FflasFfpack

Generated on Fri Feb 21 2025 00:00:00 for FflasFfpack by Doxygen 1.13.2

Fri Feb 21 2025 00:00:00

1 FFLAS-FFPACK Documentation. 1
1.1 Introduction . . . 1
1.2 Goals . . . 1
1.3 Design . . . 1
1.4 Using FFLAS-FFPACK. . . . 1
1.5 Contributing to fflas-ffpack, getting assistance. . . . 2
1.6 Copying and Licence . . . 2
1.7 Tutorial . . . 2
1.8 Configuring and Installing FFLAS-FFPACK . . . 2
1.9 Architecture of the library. . . . 2

2 Bug List 3

3 Bibliography 7

4 Todo List 9

5 Topic Index 13
5.1 Topics . . . 13

6 Namespace Index 15
6.1 Namespace List . . . 15

7 Hierarchical Index 17
7.1 Class Hierarchy . . . 17

8 Data Structure Index 25
8.1 Data Structures . . . 25

9 File Index 33
9.1 File List . . . 33

10 Topic Documentation 41
10.1 CHECKER . . . 41
10.2 FFLAS-FFPACK . . . 41
10.2.1 Detailed Description . . . 41
10.2.2 FFLAS . . . 42
10.2.3 Interfaces . . . 42
10.3 Matrix Multiplication Algorithms . . . 42
10.3.1 Detailed Description . . . 42
10.4 SIMD wrapper . . . 42
10.5 FFPACK . . . 43
10.5.1 Detailed Description . . . 43
10.6 FFLAS-FFPACK fields . . . 43
10.6.1 Detailed Description . . . 43
10.7 RNS . . . 43

11 Namespace Documentation **45**

11.1 FFLAS Namespace Reference . . . 45

11.1.1 Typedef Documentation . . . 72

11.1.1.1 Checker_fgemm . . . 72

11.1.1.2 Checker_ftrsm . . . 72

11.1.1.3 ForceCheck_fgemm . . . 72

11.1.1.4 ForceCheck_ftrsm . . . 72

11.1.1.5 ZOSparseMatrix . . . 72

11.1.1.6 NotZOSparseMatrix . . . 72

11.1.1.7 SimdSparseMatrix . . . 72

11.1.1.8 NoSimdSparseMatrix . . . 72

11.1.1.9 MKLSparseMatrixFormat . . . 72

11.1.1.10 NotMKLSparseMatrixFormat . . . 73

11.1.1.11 has_plus . . . 73

11.1.1.12 has_minus . . . 73

11.1.1.13 has_equal . . . 73

11.1.1.14 has_plus_eq . . . 73

11.1.1.15 has_minus_eq . . . 73

11.1.1.16 has_mul . . . 73

11.1.1.17 has_mul_eq . . . 73

11.1.1.18 Timer . . . 74

11.1.1.19 BaseTimer . . . 74

11.1.1.20 UserTimer . . . 74

11.1.1.21 SysTimer . . . 74

11.1.2 Enumeration Type Documentation . . . 74

11.1.2.1 FFLAS_ORDER . . . 74

11.1.2.2 FFLAS_TRANSPOSE . . . 74

11.1.2.3 FFLAS_UPLO . . . 74

11.1.2.4 FFLAS_DIAG . . . 75

11.1.2.5 FFLAS_SIDE . . . 75

11.1.2.6 FFLAS_BASE . . . 75

11.1.2.7 number_kind . . . 75

11.1.2.8 SparseMatrix_t . . . 76

11.1.2.9 FFLAS_FORMAT . . . 76

11.1.3 Function Documentation . . . 76

11.1.3.1 InfNorm() . . . 76

11.1.3.2 min3() . . . 77

11.1.3.3 max3() . . . 77

11.1.3.4 min4() . . . 77

11.1.3.5 max4() . . . 77

11.1.3.6 fadd() [1/8] . . . 77

11.1.3.7 faddin() [1/5] . . . 78

11.1.3.8 fsub() [1/4] . . . 78
11.1.3.9 fsubin() [1/3] . . . 78
11.1.3.10 fadd() [2/8] . . . 78
11.1.3.11 pfadd() . . . 79
11.1.3.12 pfsub() . . . 79
11.1.3.13 pfaddin() . . . 79
11.1.3.14 pfsubin() . . . 79
11.1.3.15 fadd() [3/8] . . . 80
11.1.3.16 fsub() [2/4] . . . 80
11.1.3.17 faddin() [2/5] . . . 81
11.1.3.18 faddin() [3/5] . . . 81
11.1.3.19 fsubin() [2/3] . . . 81
11.1.3.20 fadd() [4/8] . . . 82
11.1.3.21 fassign() [1/10] . . . 82
11.1.3.22 fassign() [2/10] . . . 83
11.1.3.23 fassign() [3/10] . . . 83
11.1.3.24 fassign() [4/10] . . . 83
11.1.3.25 fassign() [5/10] . . . 83
11.1.3.26 fassign() [6/10] . . . 83
11.1.3.27 fassign() [7/10] . . . 84
11.1.3.28 fassign() [8/10] . . . 84
11.1.3.29 faxpy() [1/6] . . . 84
11.1.3.30 faxpy() [2/6] . . . 85
11.1.3.31 faxpy() [3/6] . . . 85
11.1.3.32 faxpy() [4/6] . . . 85
11.1.3.33 fdot() [1/11] . . . 86
11.1.3.34 fdot() [2/11] . . . 86
11.1.3.35 fdot() [3/11] . . . 86
11.1.3.36 fdot() [4/11] . . . 87
11.1.3.37 fdot() [5/11] . . . 87
11.1.3.38 fdot() [6/11] . . . 87
11.1.3.39 fdot() [7/11] . . . 87
11.1.3.40 fdot() [8/11] . . . 88
11.1.3.41 fgemm() [1/23] . . . 88
11.1.3.42 fgemm() [2/23] . . . 88
11.1.3.43 fgemm() [3/23] . . . 89
11.1.3.44 fgemm() [4/23] . . . 89
11.1.3.45 fgemm() [5/23] . . . 90
11.1.3.46 fgemm() [6/23] . . . 90
11.1.3.47 fsquare() [1/6] . . . 91
11.1.3.48 fsquare() [2/6] . . . 91
11.1.3.49 fsquare() [3/6] . . . 92

11.1.3.50 fsquare() [4/6] . . . 92
11.1.3.51 fsquare() [5/6] . . . 92
11.1.3.52 fgemm() [7/23] . . . 93
11.1.3.53 fgemm() [8/23] . . . 93
11.1.3.54 fgemm() [9/23] . . . 93
11.1.3.55 fgemm() [10/23] . . . 94
11.1.3.56 fgemm() [11/23] . . . 94
11.1.3.57 fgemm() [12/23] . . . 95
11.1.3.58 fgemm() [13/23] . . . 95
11.1.3.59 fgemm() [14/23] . . . 95
11.1.3.60 fgemm() [15/23] . . . 96
11.1.3.61 fgemm() [16/23] . . . 96
11.1.3.62 fgemm() [17/23] . . . 96
11.1.3.63 fgemm() [18/23] . . . 97
11.1.3.64 fgemv() [1/19] . . . 97
11.1.3.65 fgemv() [2/19] . . . 98
11.1.3.66 fgemv() [3/19] . . . 98
11.1.3.67 fgemv() [4/19] . . . 98
11.1.3.68 fgemv() [5/19] . . . 99
11.1.3.69 fgemv() [6/19] . . . 99
11.1.3.70 fgemv() [7/19] . . . 100
11.1.3.71 fgemv() [8/19] . . . 100
11.1.3.72 fgemv() [9/19] . . . 100
11.1.3.73 fgemv() [10/19] . . . 101
11.1.3.74 fgemv() [11/19] . . . 101
11.1.3.75 fgemv() [12/19] . . . 101
11.1.3.76 fgemv() [13/19] . . . 102
11.1.3.77 fgemv() [14/19] . . . 102
11.1.3.78 fgemv() [15/19] . . . 102
11.1.3.79 fgemv() [16/19] . . . 103
11.1.3.80 fger() [1/12] . . . 103
11.1.3.81 fger() [2/12] . . . 104
11.1.3.82 fger() [3/12] . . . 104
11.1.3.83 fger() [4/12] . . . 104
11.1.3.84 fger() [5/12] . . . 105
11.1.3.85 fger() [6/12] . . . 105
11.1.3.86 fger() [7/12] . . . 105
11.1.3.87 fger() [8/12] . . . 106
11.1.3.88 fger() [9/12] . . . 106
11.1.3.89 fger() [10/12] . . . 106
11.1.3.90 fger() [11/12] . . . 107
11.1.3.91 freduce() [1/11] . . . 107

11.1.3.92 freduce() [2/11] . . . 107
11.1.3.93 freduce_constoverride() [1/2] . . . 108
11.1.3.94 finit() [1/8] . . . 108
11.1.3.95 finit() [2/8] . . . 108
11.1.3.96 freduce() [3/11] . . . 109
11.1.3.97 freduce() [4/11] . . . 109
11.1.3.98 pfreduce() . . . 109
11.1.3.99 freduce() [5/11] . . . 109
11.1.3.100 freduce_constoverride() [2/2] . . . 110
11.1.3.101 finit() [3/8] . . . 110
11.1.3.102 finit() [4/8] . . . 110
11.1.3.103 freduce() [6/11] . . . 111
11.1.3.104 freduce() [7/11] . . . 111
11.1.3.105 freivalds() . . . 111
11.1.3.106 fscalin() [1/10] . . . 112
11.1.3.107 fscal() [1/10] . . . 112
11.1.3.108 fscal() [2/10] . . . 113
11.1.3.109 fscal() [3/10] . . . 113
11.1.3.110 fscalin() [2/10] . . . 113
11.1.3.111 fscalin() [3/10] . . . 113
11.1.3.112 fscalin() [4/10] . . . 113
11.1.3.113 fscal() [4/10] . . . 114
11.1.3.114 fscalin() [5/10] . . . 114
11.1.3.115 fscal() [5/10] . . . 115
11.1.3.116 fscalin() [6/10] . . . 115
11.1.3.117 fscal() [6/10] . . . 115
11.1.3.118 fscalin() [7/10] . . . 115
11.1.3.119 fscal() [7/10] . . . 116
11.1.3.120 fscalin() [8/10] . . . 116
11.1.3.121 fscal() [8/10] . . . 116
11.1.3.122 fsyr2k() . . . 116
11.1.3.123 fsyrk() [1/16] . . . 117
11.1.3.124 fsyrk() [2/16] . . . 118
11.1.3.125 fsyrk() [3/16] . . . 118
11.1.3.126 fsyrk() [4/16] . . . 118
11.1.3.127 fsyrk() [5/16] . . . 119
11.1.3.128 fsyrk() [6/16] . . . 119
11.1.3.129 fsyrk() [7/16] . . . 119
11.1.3.130 fsyrk() [8/16] . . . 120
11.1.3.131 fsyrk() [9/16] . . . 120
11.1.3.132 fsyrk() [10/16] . . . 120
11.1.3.133 fsyrk() [11/16] . . . 121

11.1.3.134 fsyrk() [12/16] . . . 121
11.1.3.135 fsyrk() [13/16] . . . 122
11.1.3.136 fsyrk() [14/16] . . . 122
11.1.3.137 computeS1S2() . . . 123
11.1.3.138 fsyrk() [15/16] . . . 123
11.1.3.139 fsyrk() [16/16] . . . 124
11.1.3.140 fsyrk_strassen() [1/2] . . . 124
11.1.3.141 ftrmm() [1/3] . . . 124
11.1.3.142 ftrmm() [2/3] . . . 125
11.1.3.143 ftrsm() [1/9] . . . 126
11.1.3.144 ftrsm() [2/9] . . . 126
11.1.3.145 ftrsm() [3/9] . . . 127
11.1.3.146 ftrsm() [4/9] . . . 127
11.1.3.147 ftrsm() [5/9] . . . 127
11.1.3.148 cblas_imptrsm() . . . 128
11.1.3.149 ftrsv() [1/2] . . . 128
11.1.3.150 igemm_() . . . 128
11.1.3.151 finit() [5/8] . . . 129
11.1.3.152 fconvert() [1/3] . . . 129
11.1.3.153 fnegin() [1/4] . . . 130
11.1.3.154 fneg() [1/4] . . . 130
11.1.3.155 fzero() [1/5] . . . 131
11.1.3.156 frand() [1/2] . . . 131
11.1.3.157 fiszero() [1/4] . . . 131
11.1.3.158 fequal() [1/4] . . . 132
11.1.3.159 faxpby() [1/2] . . . 132
11.1.3.160 fdot() [9/11] . . . 133
11.1.3.161 fswap() [1/2] . . . 133
11.1.3.162 fzero() [2/5] . . . 134
11.1.3.163 fzero() [3/5] . . . 135
11.1.3.164 frand() [2/2] . . . 135
11.1.3.165 fequal() [2/4] . . . 136
11.1.3.166 fiszero() [2/4] . . . 136
11.1.3.167 fidentity() [1/4] . . . 137
11.1.3.168 fidentity() [2/4] . . . 137
11.1.3.169 finit() [6/8] . . . 137
11.1.3.170 fconvert() [2/3] . . . 137
11.1.3.171 fnegin() [2/4] . . . 138
11.1.3.172 fneg() [2/4] . . . 138
11.1.3.173 faxpby() [2/2] . . . 139
11.1.3.174 fmove() [1/2] . . . 139
11.1.3.175 bitsize() . . . 140

11.1.3.176 bitsize< Givaro::ZRing< Givaro::Integer > >() . . . 140
11.1.3.177 ftrmv() . . . 141
11.1.3.178 ftrsm() [6/9] . . . 141
11.1.3.179 fsyrk_strassen() [2/2] . . . 142
11.1.3.180 pfgemm() [1/7] . . . 142
11.1.3.181 pfgemm_1D_rec() . . . 143
11.1.3.182 pfgemm_2D_rec() . . . 143
11.1.3.183 pfgemm_3D_rec() . . . 143
11.1.3.184 pfgemm_3D_rec2() . . . 144
11.1.3.185 fgemm() [19/23] . . . 144
11.1.3.186 ftrsm() [7/9] . . . 144
11.1.3.187 ftrsm() [8/9] . . . 145
11.1.3.188 fspmv() [1/2] . . . 145
11.1.3.189 fspmm() . . . 145
11.1.3.190 sparse_init() [1/16] . . . 146
11.1.3.191 sparse_init() [2/16] . . . 146
11.1.3.192 sparse_delete() [1/12] . . . 146
11.1.3.193 sparse_delete() [2/12] . . . 146
11.1.3.194 sparse_init() [3/16] . . . 146
11.1.3.195 sparse_init() [4/16] . . . 147
11.1.3.196 sparse_delete() [3/12] . . . 147
11.1.3.197 sparse_delete() [4/12] . . . 147
11.1.3.198 sparse_print() [1/3] . . . 147
11.1.3.199 sparse_init() [5/16] . . . 147
11.1.3.200 sparse_init() [6/16] . . . 148
11.1.3.201 sparse_init() [7/16] . . . 148
11.1.3.202 sparse_init() [8/16] . . . 148
11.1.3.203 sparse_delete() [5/12] . . . 148
11.1.3.204 sparse_init() [9/16] . . . 149
11.1.3.205 sparse_init() [10/16] . . . 149
11.1.3.206 sparse_init() [11/16] . . . 149
11.1.3.207 sparse_delete() [6/12] . . . 149
11.1.3.208 sparse_delete() [7/12] . . . 149
11.1.3.209 sparse_init() [12/16] . . . 150
11.1.3.210 sparse_init() [13/16] . . . 150
11.1.3.211 sparse_delete() [8/12] . . . 150
11.1.3.212 sparse_delete() [9/12] . . . 150
11.1.3.213 sparse_print() [2/3] . . . 150
11.1.3.214 sparse_delete() [10/12] . . . 150
11.1.3.215 sparse_init() [14/16] . . . 151
11.1.3.216 operator<<() . . . 151
11.1.3.217 readSmsFormat() . . . 151

11.1.3.218 readSprFormat() . . . 151
11.1.3.219 getDataType() [1/4] . . . 151
11.1.3.220 getDataType() [2/4] . . . 152
11.1.3.221 getDataType() [3/4] . . . 152
11.1.3.222 getDataType() [4/4] . . . 152
11.1.3.223 readMachineType() . . . 152
11.1.3.224 readDnsFormat() . . . 152
11.1.3.225 writeDnsFormat() . . . 152
11.1.3.226 fspmv() [2/2] . . . 153
11.1.3.227 sparse_delete() [11/12] . . . 153
11.1.3.228 sparse_delete() [12/12] . . . 153
11.1.3.229 sparse_print() [3/3] . . . 153
11.1.3.230 sparse_init() [15/16] . . . 153
11.1.3.231 sparse_init() [16/16] . . . 153
11.1.3.232 computeDeviation() . . . 154
11.1.3.233 getStat() . . . 154
11.1.3.234 ftranspose() . . . 154
11.1.3.235 maxCardinality() . . . 154
11.1.3.236 maxCardinality< Givaro::Modular< int64_t > >() . . . 154
11.1.3.237 maxCardinality< Givaro::Modular< int32_t > >() . . . 154
11.1.3.238 minCardinality() . . . 155
11.1.3.239 fflas_delete() [1/4] . . . 155
11.1.3.240 fflas_delete() [2/4] . . . 155
11.1.3.241 fflas_new() [1/7] . . . 155
11.1.3.242 fflas_new() [2/7] . . . 155
11.1.3.243 finit_rns() [1/2] . . . 155
11.1.3.244 finit_trans_rns() . . . 156
11.1.3.245 fconvert_rns() [1/2] . . . 156
11.1.3.246 fconvert_trans_rns() . . . 156
11.1.3.247 fflas_new() [3/7] . . . 156
11.1.3.248 fflas_new() [4/7] . . . 156
11.1.3.249 finit_rns() [2/2] . . . 157
11.1.3.250 fconvert_rns() [2/2] . . . 157
11.1.3.251 freduce() [8/11] . . . 157
11.1.3.252 freduce() [9/11] . . . 157
11.1.3.253 finit() [7/8] . . . 158
11.1.3.254 fconvert() [3/3] . . . 158
11.1.3.255 fnegin() [3/4] . . . 159
11.1.3.256 fneg() [3/4] . . . 159
11.1.3.257 fzero() [4/5] . . . 160
11.1.3.258 fiszero() [3/4] . . . 160
11.1.3.259 fequal() [3/4] . . . 160

11.1.3.260 fassign() [9/10] . . . 161
11.1.3.261 fscalin() [9/10] . . . 161
11.1.3.262 fscal() [9/10] . . . 162
11.1.3.263 faxpy() [5/6] . . . 162
11.1.3.264 fdot() [10/11] . . . 163
11.1.3.265 fswap() [2/2] . . . 163
11.1.3.266 fadd() [5/8] . . . 164
11.1.3.267 fsub() [3/4] . . . 164
11.1.3.268 faddin() [4/5] . . . 164
11.1.3.269 fadd() [6/8] . . . 164
11.1.3.270 fassign() [10/10] . . . 165
11.1.3.271 fzero() [5/5] . . . 165
11.1.3.272 fequal() [4/4] . . . 166
11.1.3.273 fiszero() [4/4] . . . 166
11.1.3.274 fidentity() [3/4] . . . 166
11.1.3.275 fidentity() [4/4] . . . 167
11.1.3.276 freduce() [10/11] . . . 167
11.1.3.277 freduce() [11/11] . . . 167
11.1.3.278 finit() [8/8] . . . 168
11.1.3.279 fnegin() [4/4] . . . 168
11.1.3.280 fneg() [4/4] . . . 168
11.1.3.281 fscalin() [10/10] . . . 169
11.1.3.282 fscal() [10/10] . . . 169
11.1.3.283 faxpy() [6/6] . . . 170
11.1.3.284 fmove() [2/2] . . . 170
11.1.3.285 fadd() [7/8] . . . 171
11.1.3.286 fsub() [4/4] . . . 171
11.1.3.287 fsubin() [3/3] . . . 172
11.1.3.288 fadd() [8/8] . . . 172
11.1.3.289 faddin() [5/5] . . . 173
11.1.3.290 fgemv() [17/19] . . . 173
11.1.3.291 fger() [12/12] . . . 174
11.1.3.292 ftrsv() [2/2] . . . 174
11.1.3.293 ftrsm() [9/9] . . . 175
11.1.3.294 ftrmm() [3/3] . . . 176
11.1.3.295 fgemm() [20/23] . . . 176
11.1.3.296 fgemm() [21/23] . . . 177
11.1.3.297 fgemm() [22/23] . . . 177
11.1.3.298 fgemm() [23/23] . . . 178
11.1.3.299 fsquare() [6/6] . . . 178
11.1.3.300 BlockCuts() [1/2] . . . 179
11.1.3.301 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >() . . . 179

11.1.3.302 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >() 179
11.1.3.303 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >() 179
11.1.3.304 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >() 180
11.1.3.305 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >() 180
11.1.3.306 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >() 180
11.1.3.307 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >() 180
11.1.3.308 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >() 180
11.1.3.309 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >() 181
11.1.3.310 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >() 181
11.1.3.311 BlockCuts() [2/2] 181
11.1.3.312 pfzero() 181
11.1.3.313 pfrand() 181
11.1.3.314 fdot() [11/11] 182
11.1.3.315 pfgemm() [2/7] 182
11.1.3.316 pfgemm() [3/7] 182
11.1.3.317 pfgemm() [4/7] 183
11.1.3.318 pfgemm() [5/7] 183
11.1.3.319 pfgemm() [6/7] 183
11.1.3.320 pfgemm() [7/7] 184
11.1.3.321 fgemv() [18/19] 184
11.1.3.322 fgemv() [19/19] 184
11.1.3.323 parseArguments() 185
11.1.3.324 getArgumentValue() 185
11.1.3.325 writeCommandString() 185
11.1.3.326 WriteMatrix() [1/2] 186
11.1.3.327 preamble() 186
11.1.3.328 ReadMatrix() [1/2] 186
11.1.3.329 ReadMatrix() [2/2] 187
11.1.3.330 WriteMatrix() [2/2] 187
11.1.3.331 WritePermutation() 187
11.1.3.332 alignable() 188
11.1.3.333 alignable< Givaro::Integer * >() 188
11.1.3.334 fflas_new() [5/7] 188
11.1.3.335 fflas_new() [6/7] 188
11.1.3.336 fflas_new() [7/7] 188
11.1.3.337 fflas_delete() [3/4] 188
11.1.3.338 fflas_delete() [4/4] 189
11.1.3.339 prefetch() 189
11.1.3.340 getTLBSize() 189
11.1.3.341 queryCacheSizes() 189
11.1.3.342 queryL1CacheSize() 189
11.1.3.343 queryTopLevelCacheSize() 189

11.1.3.344 getSeed() . . . 189
11.2 FFLAS::_ftranspose_impl Namespace Reference . . . 190
11.2.1 Function Documentation . . . 190
11.2.1.1 not_inplace() . . . 190
11.2.1.2 square_inplace() . . . 190
11.2.1.3 nonsquare_inplace_v1() . . . 190
11.2.1.4 nonsquare_inplace_v2() . . . 191
11.3 FFLAS::BLAS3 Namespace Reference . . . 191
11.3.1 Function Documentation . . . 192
11.3.1.1 Bini() . . . 192
11.3.1.2 WinoPar() . . . 193
11.3.1.3 Winograd() . . . 193
11.3.1.4 WinogradAcc_3_23() . . . 193
11.3.1.5 WinogradAcc_3_21() . . . 194
11.3.1.6 WinogradAcc_2_24() . . . 194
11.3.1.7 WinogradAcc_2_27() . . . 195
11.3.1.8 WinogradAcc_LR() . . . 195
11.3.1.9 WinogradAcc_R_S() . . . 195
11.3.1.10 WinogradAcc_L_S() . . . 196
11.3.1.11 Winograd_LR_S() . . . 196
11.3.1.12 Winograd_L_S() . . . 196
11.3.1.13 Winograd_R_S() . . . 197
11.4 FFLAS::csr_hyb_details Namespace Reference . . . 197
11.5 FFLAS::CuttingStrategy Namespace Reference . . . 197
11.5.1 Typedef Documentation . . . 198
11.5.1.1 RNSModulus . . . 198
11.6 FFLAS::details Namespace Reference . . . 198
11.6.1 Function Documentation . . . 199
11.6.1.1 fadd() [1/5] . . . 199
11.6.1.2 fadd() [2/5] . . . 200
11.6.1.3 fadd() [3/5] . . . 200
11.6.1.4 fadd() [4/5] . . . 200
11.6.1.5 fadd() [5/5] . . . 201
11.6.1.6 faxpy() [1/2] . . . 201
11.6.1.7 faxpy() [2/2] . . . 201
11.6.1.8 freduce() [1/4] . . . 201
11.6.1.9 freduce() [2/4] . . . 202
11.6.1.10 freduce() [3/4] . . . 202
11.6.1.11 freduce() [4/4] . . . 202
11.6.1.12 fscalin() [1/2] . . . 202
11.6.1.13 fscal() [1/2] . . . 203
11.6.1.14 fscalin() [2/2] . . . 203

11.6.1.15 fscal() [2/2] . . . 203
11.6.1.16 igebb44() . . . 203
11.6.1.17 igebb24() . . . 204
11.6.1.18 igebb14() . . . 204
11.6.1.19 igebb41() . . . 204
11.6.1.20 igebb21() . . . 204
11.6.1.21 igebb11() . . . 205
11.6.1.22 igebp() . . . 205
11.6.1.23 pack_lhs() . . . 205
11.6.1.24 pack_rhs() . . . 205
11.6.1.25 gebp() . . . 206
11.6.1.26 BlockingFactor() . . . 206
11.7 FFLAS::details_spmv Namespace Reference . . . 206
11.8 FFLAS::ElementCategories Namespace Reference . . . 206
11.9 FFLAS::FieldCategories Namespace Reference . . . 206
11.9.1 Detailed Description . . . 207
11.10 FFLAS::MMHelperAlgo Namespace Reference . . . 207
11.11 FFLAS::ModeCategories Namespace Reference . . . 207
11.11.1 Detailed Description . . . 207
11.12 FFLAS::ParSeqHelper Namespace Reference . . . 208
11.12.1 Detailed Description . . . 208
11.13 FFLAS::Protected Namespace Reference . . . 208
11.13.1 Function Documentation . . . 211
11.13.1.1 computeFactorClassic() [1/3] . . . 211
11.13.1.2 computeFactorClassic() [2/3] . . . 211
11.13.1.3 computeFactorClassic() [3/3] . . . 212
11.13.1.4 DotProdBoundClassic() . . . 212
11.13.1.5 TRSMBound() [1/3] . . . 212
11.13.1.6 TRSMBound() [2/3] . . . 212
11.13.1.7 TRSMBound() [3/3] . . . 212
11.13.1.8 fgemm_convert() . . . 213
11.13.1.9 NeedPreAddReduction() [1/2] . . . 213
11.13.1.10 NeedPreAddReduction() [2/2] . . . 213
11.13.1.11 NeedPreSubReduction() [1/2] . . . 213
11.13.1.12 NeedPreSubReduction() [2/2] . . . 214
11.13.1.13 NeedDoublePreAddReduction() [1/2] . . . 214
11.13.1.14 NeedDoublePreAddReduction() [2/2] . . . 214
11.13.1.15 ScalAndReduce() [1/3] . . . 214
11.13.1.16 ScalAndReduce() [2/3] . . . 215
11.13.1.17 fsquareCommon() . . . 215
11.13.1.18 WinogradThreshold() [1/4] . . . 215
11.13.1.19 WinogradThreshold() [2/4] . . . 215

11.13.1.20 WinogradThreshold() [3/4] . . . 215
11.13.1.21 WinogradThreshold() [4/4] . . . 216
11.13.1.22 WinogradSteps() . . . 216
11.13.1.23 DynamicPeeling() . . . 216
11.13.1.24 DynamicPeeling2() . . . 217
11.13.1.25 WinogradCalc() . . . 217
11.13.1.26 fgemv_convert() . . . 217
11.13.1.27 fger_convert() . . . 218
11.13.1.28 fsyrk_convert() . . . 218
11.13.1.29 ScalAndReduce() [3/3] . . . 218
11.13.1.30 NeedPreScalReduction() [1/2] . . . 219
11.13.1.31 NeedPreScalReduction() [2/2] . . . 219
11.13.1.32 NeedPreAxpyReduction() [1/2] . . . 219
11.13.1.33 NeedPreAxpyReduction() [2/2] . . . 219
11.13.1.34 min_types() [1/7] . . . 219
11.13.1.35 min_types() [2/7] . . . 220
11.13.1.36 min_types() [3/7] . . . 220
11.13.1.37 min_types() [4/7] . . . 220
11.13.1.38 min_types() [5/7] . . . 220
11.13.1.39 min_types() [6/7] . . . 220
11.13.1.40 min_types() [7/7] . . . 220
11.13.1.41 unfit() [1/4] . . . 220
11.13.1.42 unfit() [2/4] . . . 220
11.13.1.43 unfit() [3/4] . . . 221
11.13.1.44 unfit() [4/4] . . . 221
11.13.1.45 igemm_colmajor() [1/2] . . . 221
11.13.1.46 igemm_colmajor() [2/2] . . . 221
11.13.1.47 igemm() . . . 221
11.13.1.48 MatF2MatD_Triangular() . . . 222
11.13.1.49 MatF2MatFl_Triangular() . . . 222
11.14 FFLAS::sell_details Namespace Reference . . . 222
11.15 FFLAS::sparse_details Namespace Reference . . . 223
11.15.1 Function Documentation . . . 226
11.15.1.1 init_y() [1/2] . . . 226
11.15.1.2 init_y() [2/2] . . . 226
11.15.1.3 fspmv_dispatch() [1/2] . . . 226
11.15.1.4 fspmv_dispatch() [2/2] . . . 226
11.15.1.5 fspmv() [1/12] . . . 227
11.15.1.6 fspmv() [2/12] . . . 227
11.15.1.7 fspmv() [3/12] . . . 227
11.15.1.8 fspmv() [4/12] . . . 227
11.15.1.9 fspmv() [5/12] . . . 227

11.15.1.10 fspmv() [6/12] . . . 228
11.15.1.11 fspmv() [7/12] . . . 228
11.15.1.12 fspmv() [8/12] . . . 228
11.15.1.13 fspmv() [9/12] . . . 228
11.15.1.14 fspmm_dispatch() [1/2] . . . 229
11.15.1.15 fspmm_dispatch() [2/2] . . . 229
11.15.1.16 fspmm() [1/9] . . . 229
11.15.1.17 fspmm() [2/9] . . . 229
11.15.1.18 fspmm() [3/9] . . . 230
11.15.1.19 fspmm() [4/9] . . . 230
11.15.1.20 fspmm() [5/9] . . . 230
11.15.1.21 fspmm() [6/9] . . . 231
11.15.1.22 fspmm() [7/9] . . . 231
11.15.1.23 fspmm() [8/9] . . . 231
11.15.1.24 fspmm() [9/9] . . . 231
11.15.1.25 pfspmm_dispatch() [1/2] . . . 232
11.15.1.26 pfspmm_dispatch() [2/2] . . . 232
11.15.1.27 pfspmm() [1/9] . . . 232
11.15.1.28 pfspmm() [2/9] . . . 232
11.15.1.29 pfspmm() [3/9] . . . 233
11.15.1.30 pfspmm() [4/9] . . . 233
11.15.1.31 pfspmm() [5/9] . . . 233
11.15.1.32 pfspmm() [6/9] . . . 234
11.15.1.33 pfspmm() [7/9] . . . 234
11.15.1.34 pfspmm() [8/9] . . . 234
11.15.1.35 pfspmm() [9/9] . . . 234
11.15.1.36 pfspmv() [1/6] . . . 235
11.15.1.37 pfspmv() [2/6] . . . 235
11.15.1.38 pfspmv() [3/6] . . . 235
11.15.1.39 pfspmv() [4/6] . . . 235
11.15.1.40 pfspmv() [5/6] . . . 235
11.15.1.41 pfspmv() [6/6] . . . 236
11.15.1.42 fspmv() [10/12] . . . 236
11.15.1.43 fspmv() [11/12] . . . 236
11.15.1.44 fspmv() [12/12] . . . 236
11.16 FFLAS::sparse_details_impl Namespace Reference . . . 237
11.16.1 Function Documentation . . . 245
11.16.1.1 fspmm() [1/15] . . . 245
11.16.1.2 fspmm() [2/15] . . . 245
11.16.1.3 fspmm() [3/15] . . . 246
11.16.1.4 fspmm_simd_aligned() [1/2] . . . 246
11.16.1.5 fspmm_simd_unaligned() [1/2] . . . 246

11.16.1.6 fspmm_one() [1/4] . . . 246
11.16.1.7 fspmm_mone() [1/4] . . . 247
11.16.1.8 fspmm_one_simd_aligned() [1/3] . . . 247
11.16.1.9 fspmm_one_simd_unaligned() [1/3] . . . 247
11.16.1.10 fspmm_mone_simd_aligned() [1/3] . . . 247
11.16.1.11 fspmm_mone_simd_unaligned() [1/3] . . . 248
11.16.1.12 fspmv() [1/21] . . . 248
11.16.1.13 fspmv() [2/21] . . . 248
11.16.1.14 fspmv() [3/21] . . . 248
11.16.1.15 fspmv_one() [1/10] . . . 248
11.16.1.16 fspmv_mone() [1/10] . . . 249
11.16.1.17 fspmv_one() [2/10] . . . 249
11.16.1.18 fspmv_mone() [2/10] . . . 249
11.16.1.19 pfspmm() [1/18] . . . 249
11.16.1.20 pfspmm() [2/18] . . . 249
11.16.1.21 pfspmm() [3/18] . . . 250
11.16.1.22 pfspmm_one() [1/2] . . . 250
11.16.1.23 pfspmm_mone() [1/2] . . . 250
11.16.1.24 pfspmm_one() [2/2] . . . 250
11.16.1.25 pfspmm_mone() [2/2] . . . 251
11.16.1.26 pfspmv() [1/18] . . . 251
11.16.1.27 pfspmv_task() . . . 251
11.16.1.28 pfspmv() [2/18] . . . 251
11.16.1.29 pfspmv() [3/18] . . . 251
11.16.1.30 pfspmv_one() [1/8] . . . 252
11.16.1.31 pfspmv_mone() [1/8] . . . 252
11.16.1.32 pfspmv_one() [2/8] . . . 252
11.16.1.33 pfspmv_mone() [2/8] . . . 252
11.16.1.34 fspmm() [4/15] . . . 252
11.16.1.35 fspmm() [5/15] . . . 253
11.16.1.36 fspmm_simd_aligned() [2/2] . . . 253
11.16.1.37 fspmm_simd_unaligned() [2/2] . . . 253
11.16.1.38 fspmm() [6/15] . . . 253
11.16.1.39 fspmm_one() [2/4] . . . 254
11.16.1.40 fspmm_mone() [2/4] . . . 254
11.16.1.41 fspmm_one_simd_aligned() [2/3] . . . 254
11.16.1.42 fspmm_one_simd_unaligned() [2/3] . . . 254
11.16.1.43 fspmm_mone_simd_aligned() [2/3] . . . 255
11.16.1.44 fspmm_mone_simd_unaligned() [2/3] . . . 255
11.16.1.45 fspmv() [4/21] . . . 255
11.16.1.46 fspmv() [5/21] . . . 255
11.16.1.47 fspmv() [6/21] . . . 255

11.16.1.48 fspmv_one() [3/10] . . . 256
11.16.1.49 fspmv_mone() [3/10] . . . 256
11.16.1.50 fspmv_one() [4/10] . . . 256
11.16.1.51 fspmv_mone() [4/10] . . . 256
11.16.1.52 pfspmm() [4/18] . . . 256
11.16.1.53 pfspmm() [5/18] . . . 257
11.16.1.54 pfspmm() [6/18] . . . 257
11.16.1.55 pfspmm() [7/18] . . . 257
11.16.1.56 pfspmm() [8/18] . . . 257
11.16.1.57 pfspmm() [9/18] . . . 258
11.16.1.58 pfspmv() [4/18] . . . 258
11.16.1.59 pfspmv() [5/18] . . . 258
11.16.1.60 pfspmv() [6/18] . . . 258
11.16.1.61 fspmm() [7/15] . . . 258
11.16.1.62 fspmm() [8/15] . . . 259
11.16.1.63 fspmm() [9/15] . . . 259
11.16.1.64 fspmv() [7/21] . . . 259
11.16.1.65 fspmv() [8/21] . . . 259
11.16.1.66 fspmv() [9/21] . . . 259
11.16.1.67 pfspmm() [10/18] . . . 260
11.16.1.68 pfspmm() [11/18] . . . 260
11.16.1.69 pfspmm() [12/18] . . . 260
11.16.1.70 pfspmm() [13/18] . . . 260
11.16.1.71 pfspmm() [14/18] . . . 261
11.16.1.72 pfspmm() [15/18] . . . 261
11.16.1.73 pfspmm_zo() [1/2] . . . 261
11.16.1.74 pfspmm_zo() [2/2] . . . 261
11.16.1.75 pfspmv() [7/18] . . . 262
11.16.1.76 pfspmv() [8/18] . . . 262
11.16.1.77 pfspmv() [9/18] . . . 262
11.16.1.78 pfspmv_one() [3/8] . . . 262
11.16.1.79 pfspmv_mone() [3/8] . . . 262
11.16.1.80 pfspmv_one() [4/8] . . . 263
11.16.1.81 pfspmv_mone() [4/8] . . . 263
11.16.1.82 fspmm() [10/15] . . . 263
11.16.1.83 fspmm() [11/15] . . . 263
11.16.1.84 fspmm() [12/15] . . . 264
11.16.1.85 fspmm_mone() [3/4] . . . 264
11.16.1.86 fspmm_one() [3/4] . . . 264
11.16.1.87 fspmm_mone() [4/4] . . . 264
11.16.1.88 fspmm_one() [4/4] . . . 265
11.16.1.89 fspmm_one_simd_aligned() [3/3] . . . 265

11.16.1.90 fspmm_one_simd_unaligned() [3/3] . . . 265
11.16.1.91 fspmm_mone_simd_aligned() [3/3] . . . 265
11.16.1.92 fspmm_mone_simd_unaligned() [3/3] . . . 266
11.16.1.93 fspmv() [10/21] . . . 266
11.16.1.94 fspmv() [11/21] . . . 266
11.16.1.95 fspmv() [12/21] . . . 266
11.16.1.96 fspmv_one() [5/10] . . . 266
11.16.1.97 fspmv_mone() [5/10] . . . 267
11.16.1.98 fspmv_one() [6/10] . . . 267
11.16.1.99 fspmv_mone() [6/10] . . . 267
11.16.1.100 pfspmv() [10/18] . . . 267
11.16.1.101 pfspmv() [11/18] . . . 267
11.16.1.102 pfspmv() [12/18] . . . 268
11.16.1.103 pfspmv_one() [5/8] . . . 268
11.16.1.104 pfspmv_mone() [5/8] . . . 268
11.16.1.105 pfspmv_one() [6/8] . . . 268
11.16.1.106 pfspmv_mone() [6/8] . . . 268
11.16.1.107 fspmv() [13/21] . . . 269
11.16.1.108 fspmv_simd() [1/4] . . . 269
11.16.1.109 fspmv() [14/21] . . . 269
11.16.1.110 fspmv_simd() [2/4] . . . 269
11.16.1.111 fspmv() [15/21] . . . 269
11.16.1.112 fspmv_one() [7/10] . . . 270
11.16.1.113 fspmv_mone() [7/10] . . . 270
11.16.1.114 fspmv_one() [8/10] . . . 270
11.16.1.115 fspmv_mone() [8/10] . . . 270
11.16.1.116 fspmv_one_simd() [1/2] . . . 270
11.16.1.117 fspmv_mone_simd() [1/2] . . . 271
11.16.1.118 pfspmm() [16/18] . . . 271
11.16.1.119 pfspmm() [17/18] . . . 271
11.16.1.120 pfspmm() [18/18] . . . 271
11.16.1.121 pfspmv() [13/18] . . . 272
11.16.1.122 pfspmv() [14/18] . . . 272
11.16.1.123 pfspmv() [15/18] . . . 272
11.16.1.124 fspmm() [13/15] . . . 272
11.16.1.125 fspmm() [14/15] . . . 272
11.16.1.126 fspmm() [15/15] . . . 273
11.16.1.127 fspmv() [16/21] . . . 273
11.16.1.128 fspmv() [17/21] . . . 273
11.16.1.129 fspmv() [18/21] . . . 273
11.16.1.130 pfspmv() [16/18] . . . 273
11.16.1.131 pfspmv() [17/18] . . . 274

11.16.1.132 pfspmv() [18/18] . . . 274
11.16.1.133 pfspmv_one() [7/8] . . . 274
11.16.1.134 pfspmv_mone() [7/8] . . . 274
11.16.1.135 pfspmv_one() [8/8] . . . 274
11.16.1.136 pfspmv_mone() [8/8] . . . 275
11.16.1.137 fspmv() [19/21] . . . 275
11.16.1.138 fspmv_simd() [3/4] . . . 275
11.16.1.139 fspmv() [20/21] . . . 275
11.16.1.140 fspmv_simd() [4/4] . . . 275
11.16.1.141 fspmv() [21/21] . . . 276
11.16.1.142 fspmv_one() [9/10] . . . 276
11.16.1.143 fspmv_mone() [9/10] . . . 276
11.16.1.144 fspmv_one_simd() [2/2] . . . 276
11.16.1.145 fspmv_mone_simd() [2/2] . . . 276
11.16.1.146 fspmv_one() [10/10] . . . 277
11.16.1.147 fspmv_mone() [10/10] . . . 277
11.17 FFLAS::StrategyParameter Namespace Reference . . . 277
11.18 FFLAS::StructureHelper Namespace Reference . . . 277
11.18.1 Detailed Description . . . 277
11.19 FFLAS::vectorised Namespace Reference . . . 278
11.19.1 Function Documentation . . . 279
11.19.1.1 VEC_ADD() . . . 279
11.19.1.2 addp() . . . 279
11.19.1.3 VEC_SUB() . . . 280
11.19.1.4 subp() . . . 280
11.19.1.5 add() . . . 280
11.19.1.6 sub() . . . 280
11.19.1.7 axpyp() [1/2] . . . 280
11.19.1.8 axpyp() [2/2] . . . 281
11.19.1.9 reduce() [1/9] . . . 281
11.19.1.10 reduce() [2/9] . . . 281
11.19.1.11 reduce() [3/9] . . . 281
11.19.1.12 reduce() [4/9] . . . 281
11.19.1.13 reduce() [5/9] . . . 281
11.19.1.14 reduce() [6/9] . . . 282
11.19.1.15 reduce() [7/9] . . . 282
11.19.1.16 reduce() [8/9] . . . 282
11.19.1.17 reduce() [9/9] . . . 282
11.19.1.18 modp() [1/2] . . . 282
11.19.1.19 modp() [2/2] . . . 282
11.19.1.20 scalp() [1/3] . . . 283
11.19.1.21 scalp() [2/3] . . . 283

11.19.1.22 scalp() [3/3] . . . 283
11.20 FFLAS::vectorised::unswitch Namespace Reference . . . 283
11.20.1 Function Documentation . . . 284
11.20.1.1 axpyp() [1/2] . . . 284
11.20.1.2 axpyp() [2/2] . . . 284
11.20.1.3 modp() [1/2] . . . 285
11.20.1.4 modp() [2/2] . . . 285
11.20.1.5 scalp() [1/3] . . . 285
11.20.1.6 scalp() [2/3] . . . 285
11.20.1.7 scalp() [3/3] . . . 286
11.21 FFPACK Namespace Reference . . . 286
11.21.1 Detailed Description . . . 302
11.21.2 Typedef Documentation . . . 302
11.21.2.1 Checker_PLUQ . . . 302
11.21.2.2 Checker_Det . . . 303
11.21.2.3 Checker_invert . . . 303
11.21.2.4 Checker_charpoly . . . 303
11.21.2.5 ForceCheck_PLUQ . . . 303
11.21.2.6 ForceCheck_Det . . . 303
11.21.2.7 ForceCheck_invert . . . 303
11.21.2.8 ForceCheck_charpoly . . . 303
11.21.3 Function Documentation . . . 303
11.21.3.1 LAPACKPerm2MathPerm() . . . 303
11.21.3.2 MathPerm2LAPACKPerm() . . . 304
11.21.3.3 applyP() [1/4] . . . 304
11.21.3.4 applyP() [2/4] . . . 305
11.21.3.5 applyP() [3/4] . . . 305
11.21.3.6 MonotonicApplyP() . . . 305
11.21.3.7 fgetrs() [1/4] . . . 306
11.21.3.8 fgetrs() [2/4] . . . 307
11.21.3.9 fgesv() [1/4] . . . 307
11.21.3.10 fgesv() [2/4] . . . 308
11.21.3.11 ftrtri() [1/2] . . . 309
11.21.3.12 trinv_left() [1/2] . . . 309
11.21.3.13 ftrtrm() [1/2] . . . 310
11.21.3.14 ftrstr() . . . 310
11.21.3.15 ftrssyr2k() . . . 311
11.21.3.16 fsytrf() [1/3] . . . 311
11.21.3.17 fsytrf() [2/3] . . . 312
11.21.3.18 fsytrf() [3/3] . . . 312
11.21.3.19 fsytrf_nonunit() [1/3] . . . 312
11.21.3.20 PLUQ() [1/6] . . . 313

11.21.3.21 pPLUQ() . . . 314
11.21.3.22 PLUQ() [2/6] . . . 314
11.21.3.23 PLUQ() [3/6] . . . 314
11.21.3.24 LUdivine() [1/4] . . . 314
11.21.3.25 ColumnEchelonForm() [1/3] . . . 315
11.21.3.26 pColumnEchelonForm() . . . 316
11.21.3.27 ColumnEchelonForm() [2/3] . . . 316
11.21.3.28 RowEchelonForm() [1/3] . . . 316
11.21.3.29 pRowEchelonForm() . . . 317
11.21.3.30 RowEchelonForm() [2/3] . . . 317
11.21.3.31 ReducedColumnEchelonForm() [1/3] . . . 318
11.21.3.32 pReducedColumnEchelonForm() . . . 318
11.21.3.33 ReducedColumnEchelonForm() [2/3] . . . 318
11.21.3.34 ReducedRowEchelonForm() [1/3] . . . 319
11.21.3.35 pReducedRowEchelonForm() . . . 319
11.21.3.36 ReducedRowEchelonForm() [2/3] . . . 320
11.21.3.37 Invert() [1/4] . . . 320
11.21.3.38 Invert() [2/4] . . . 320
11.21.3.39 Invert2() [1/2] . . . 321
11.21.3.40 CharPoly() [1/8] . . . 322
11.21.3.41 CharPoly() [2/8] . . . 322
11.21.3.42 CharPoly() [3/8] . . . 323
11.21.3.43 MinPoly() [1/4] . . . 323
11.21.3.44 MinPoly() [2/4] . . . 324
11.21.3.45 MatVecMinPoly() [1/2] . . . 324
11.21.3.46 Rank() [1/3] . . . 325
11.21.3.47 pRank() . . . 325
11.21.3.48 Rank() [2/3] . . . 325
11.21.3.49 IsSingular() [1/2] . . . 326
11.21.3.50 Det() [1/6] . . . 326
11.21.3.51 pDet() . . . 327
11.21.3.52 Det() [2/6] . . . 327
11.21.3.53 Solve() [1/3] . . . 327
11.21.3.54 Solve() [2/3] . . . 328
11.21.3.55 pSolve() . . . 328
11.21.3.56 RandomNullSpaceVector() [1/3] . . . 328
11.21.3.57 NullSpaceBasis() [1/2] . . . 329
11.21.3.58 RowRankProfile() [1/3] . . . 329
11.21.3.59 pRowRankProfile() . . . 330
11.21.3.60 RowRankProfile() [2/3] . . . 330
11.21.3.61 ColumnRankProfile() [1/3] . . . 330
11.21.3.62 pColumnRankProfile() . . . 331

11.21.3.63 ColumnRankProfile() [2/3] . . . 331
11.21.3.64 RankProfileFromLU() . . . 331
11.21.3.65 LeadingSubmatrixRankProfiles() . . . 332
11.21.3.66 RowRankProfileSubmatrixIndices() [1/2] . . . 332
11.21.3.67 ColRankProfileSubmatrixIndices() [1/2] . . . 333
11.21.3.68 RowRankProfileSubmatrix() [1/2] . . . 334
11.21.3.69 ColRankProfileSubmatrix() [1/2] . . . 334
11.21.3.70 getTriangular() [1/2] . . . 335
11.21.3.71 getTriangular() [2/2] . . . 335
11.21.3.72 getEchelonForm() [1/2] . . . 336
11.21.3.73 getEchelonForm() [2/2] . . . 337
11.21.3.74 getEchelonTransform() . . . 337
11.21.3.75 getReducedEchelonForm() [1/2] . . . 338
11.21.3.76 getReducedEchelonForm() [2/2] . . . 339
11.21.3.77 getReducedEchelonTransform() . . . 339
11.21.3.78 PLUQtoEchelonPermutation() . . . 340
11.21.3.79 LTBruhatGen() . . . 340
11.21.3.80 getLTBruhatGen() [1/2] . . . 341
11.21.3.81 getLTBruhatGen() [2/2] . . . 341
11.21.3.82 LTQSorder() . . . 342
11.21.3.83 CompressToBlockBiDiagonal() . . . 342
11.21.3.84 ExpandBlockBiDiagonalToBruhat() . . . 343
11.21.3.85 Bruhat2EchelonPermutation() . . . 344
11.21.3.86 TInverter() [1/2] . . . 344
11.21.3.87 ComputeRPermutation() [1/2] . . . 344
11.21.3.88 productBruhatxTS() [1/2] . . . 344
11.21.3.89 LQUPtoInverseOfFullRankMinor() [1/2] . . . 345
11.21.3.90 RandomNullSpaceVector() [2/3] . . . 346
11.21.3.91 solveLB() [1/2] . . . 346
11.21.3.92 solveLB2() [1/2] . . . 346
11.21.3.93 TInverter() [2/2] . . . 347
11.21.3.94 ComputeRPermutation() [2/2] . . . 347
11.21.3.95 expandLCRE() . . . 347
11.21.3.96 productBruhatxTS() [2/2] . . . 348
11.21.3.97 Danilevski() . . . 349
11.21.3.98 buildMatrix() . . . 349
11.21.3.99 CharPoly() [4/8] . . . 349
11.21.3.100 CharPoly() [5/8] . . . 350
11.21.3.101 Det() [3/6] . . . 350
11.21.3.102 Det() [4/6] . . . 350
11.21.3.103 fsytrf_BC_Crout() . . . 350
11.21.3.104 fsytrf_BC_RL() . . . 351

11.21.3.105 fsytrf_UP_RPM_BC_RL() . . . 351
11.21.3.106 fsytrf_LOW_RPM_BC_Crout() . . . 351
11.21.3.107 fsytrf_UP_RPM_BC_Crout() . . . 351
11.21.3.108 fsytrf_UP_RPM() . . . 352
11.21.3.109 fsytrf_nonunit() [2/3] . . . 352
11.21.3.110 fsytrf_nonunit() [3/3] . . . 352
11.21.3.111 fsytrf_RPM() . . . 352
11.21.3.112 getTridiagonal() . . . 353
11.21.3.113 LUdivine_gauss() [1/2] . . . 353
11.21.3.114 LUdivine_small() [1/2] . . . 353
11.21.3.115 LUdivine() [2/4] . . . 353
11.21.3.116 LUdivine() [3/4] . . . 354
11.21.3.117 MonotonicCompress() . . . 354
11.21.3.118 MonotonicCompressMorePivots() . . . 354
11.21.3.119 MonotonicCompressCycles() . . . 355
11.21.3.120 MonotonicExpand() . . . 355
11.21.3.121 applyP_block() . . . 355
11.21.3.122 doApplyS() . . . 355
11.21.3.123 MatrixApplyS() [1/3] . . . 356
11.21.3.124 MatrixApplyS() [2/3] . . . 356
11.21.3.125 MatrixApplyS() [3/3] . . . 356
11.21.3.126 PermApplyS() . . . 356
11.21.3.127 doApplyT() . . . 357
11.21.3.128 MatrixApplyT() [1/3] . . . 357
11.21.3.129 MatrixApplyT() [2/3] . . . 357
11.21.3.130 MatrixApplyT() [3/3] . . . 357
11.21.3.131 PermApplyT() . . . 358
11.21.3.132 composePermutationsLLL() . . . 358
11.21.3.133 composePermutationsLLM() . . . 358
11.21.3.134 composePermutationsMLM() . . . 359
11.21.3.135 cyclic_shift_mathPerm() . . . 359
11.21.3.136 cyclic_shift_row_col() [1/2] . . . 359
11.21.3.137 cyclic_shift_row() [1/3] . . . 359
11.21.3.138 cyclic_shift_row() [2/3] . . . 360
11.21.3.139 cyclic_shift_col() [1/3] . . . 360
11.21.3.140 cyclic_shift_col() [2/3] . . . 360
11.21.3.141 PLUQ_basecaseV3() . . . 360
11.21.3.142 PLUQ_basecaseV2() . . . 360
11.21.3.143 PLUQ_basecaseCrout() . . . 361
11.21.3.144 _PLUQ() . . . 361
11.21.3.145 PLUQ() [4/6] . . . 361
11.21.3.146 threads_fgemm() . . . 361

11.21.3.147 threads_ftrsm() . . . 362
11.21.3.148 PLUQ() [5/6] . . . 362
11.21.3.149 fflas_const_cast() [1/3] . . . 362
11.21.3.150 fflas_const_cast() [2/3] . . . 362
11.21.3.151 cyclic_shift_row_col() [2/2] . . . 362
11.21.3.152 cyclic_shift_row() [3/3] . . . 363
11.21.3.153 cyclic_shift_col() [3/3] . . . 363
11.21.3.154 applyP() [4/4] . . . 363
11.21.3.155 fgetrs() [3/4] . . . 363
11.21.3.156 fgetrs() [4/4] . . . 364
11.21.3.157 fgesv() [3/4] . . . 364
11.21.3.158 fgesv() [4/4] . . . 364
11.21.3.159 ftrtri() [2/2] . . . 364
11.21.3.160 trinv_left() [2/2] . . . 365
11.21.3.161 ftrtrm() [2/2] . . . 365
11.21.3.162 PLUQ() [6/6] . . . 365
11.21.3.163 LUdivine() [4/4] . . . 365
11.21.3.164 LUdivine_small() [2/2] . . . 366
11.21.3.165 LUdivine_gauss() [2/2] . . . 366
11.21.3.166 RowEchelonForm() [3/3] . . . 366
11.21.3.167 ReducedRowEchelonForm() [3/3] . . . 366
11.21.3.168 ColumnEchelonForm() [3/3] . . . 367
11.21.3.169 ReducedColumnEchelonForm() [3/3] . . . 367
11.21.3.170 Invert() [3/4] . . . 367
11.21.3.171 Invert() [4/4] . . . 367
11.21.3.172 Invert2() [2/2] . . . 367
11.21.3.173 CharPoly() [6/8] . . . 368
11.21.3.174 CharPoly() [7/8] . . . 368
11.21.3.175 CharPoly() [8/8] . . . 368
11.21.3.176 MinPoly() [3/4] . . . 368
11.21.3.177 MinPoly() [4/4] . . . 368
11.21.3.178 MatVecMinPoly() [2/2] . . . 369
11.21.3.179 KrylovElim() . . . 369
11.21.3.180 SpecRankProfile() . . . 369
11.21.3.181 Rank() [3/3] . . . 369
11.21.3.182 IsSingular() [2/2] . . . 369
11.21.3.183 Det() [5/6] . . . 370
11.21.3.184 Det() [6/6] . . . 370
11.21.3.185 Solve() [3/3] . . . 370
11.21.3.186 solveLB() [2/2] . . . 370
11.21.3.187 solveLB2() [2/2] . . . 371
11.21.3.188 RandomNullSpaceVector() [3/3] . . . 371

11.21.3.189 NullSpaceBasis() [2/2] . . . 371
11.21.3.190 RowRankProfile() [3/3] . . . 371
11.21.3.191 ColumnRankProfile() [3/3] . . . 372
11.21.3.192 RowRankProfileSubmatrixIndices() [2/2] . . . 372
11.21.3.193 ColRankProfileSubmatrixIndices() [2/2] . . . 372
11.21.3.194 RowRankProfileSubmatrix() [2/2] . . . 372
11.21.3.195 ColRankProfileSubmatrix() [2/2] . . . 372
11.21.3.196 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2] . . . 373
11.21.3.197 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2] . . . 373
11.21.3.198 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2] . . . 373
11.21.3.199 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2] . . . 373
11.21.3.200 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >() . . . 374
11.21.3.201 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2] . . . 374
11.21.3.202 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2] . . . 374
11.21.3.203 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >() . . . 374
11.21.3.204 LQUPtoInverseOfFullRankMinor() [2/2] . . . 375
11.21.3.205 fflas_const_cast() [3/3] . . . 375
11.21.3.206 failure() . . . 375
11.21.3.207 isOdd() [1/3] . . . 375
11.21.3.208 isOdd() [2/3] . . . 375
11.21.3.209 isOdd() [3/3] . . . 375
11.21.3.210 NonZeroRandomMatrix() [1/2] . . . 376
11.21.3.211 NonZeroRandomMatrix() [2/2] . . . 376
11.21.3.212 RandomMatrix() [1/2] . . . 377
11.21.3.213 RandomMatrix() [2/2] . . . 377
11.21.3.214 RandomTriangularMatrix() [1/2] . . . 378
11.21.3.215 RandomTriangularMatrix() [2/2] . . . 378
11.21.3.216 RandInt() . . . 379
11.21.3.217 RandomSymmetricMatrix() . . . 379
11.21.3.218 RandomMatrixWithRank() [1/2] . . . 380
11.21.3.219 RandomMatrixWithRank() [2/2] . . . 380
11.21.3.220 RandomIndexSubset() . . . 381
11.21.3.221 RandomPermutation() . . . 381
11.21.3.222 RandomRankProfileMatrix() . . . 381
11.21.3.223 swapval() . . . 382
11.21.3.224 RandomSymmetricRankProfileMatrix() . . . 382
11.21.3.225 RandomLTQSRankProfileMatrix() . . . 382
11.21.3.226 RandomMatrixWithRankandRPM() [1/2] . . . 382
11.21.3.227 RandomMatrixWithRankandRPM() [2/2] . . . 383
11.21.3.228 RandomSymmetricMatrixWithRankandRPM() [1/2] . . . 384
11.21.3.229 RandomSymmetricMatrixWithRankandRPM() [2/2] . . . 384
11.21.3.230 RandomMatrixWithRankandRandomRPM() [1/2] . . . 385

11.21.3.231 RandomMatrixWithRankandRandomRPM() [2/2] . . . 385
11.21.3.232 RandomSymmetricMatrixWithRankandRandomRPM() [1/2] . . . 386
11.21.3.233 RandomSymmetricMatrixWithRankandRandomRPM() [2/2] . . . 386
11.21.3.234 RandomMatrixWithDet() [1/2] . . . 387
11.21.3.235 RandomMatrixWithDet() [2/2] . . . 387
11.21.3.236 RandomLTQSMatrixWithRankandQSorder() . . . 388
11.21.3.237 chooseField() . . . 388
11.21.3.238 chooseField< Givaro::ZRing< int32_t > >() . . . 388
11.21.3.239 chooseField< Givaro::ZRing< int64_t > >() . . . 388
11.21.3.240 chooseField< Givaro::ZRing< float > >() . . . 389
11.21.3.241 chooseField< Givaro::ZRing< double > >() . . . 389
11.22 FFPACK::Protected Namespace Reference . . . 389
11.22.1 Function Documentation . . . 390
11.22.1.1 LUdivine_construct() [1/2] . . . 390
11.22.1.2 GaussJordan() . . . 391
11.22.1.3 KellerGehrig() . . . 392
11.22.1.4 KGFast() . . . 392
11.22.1.5 KGFast_generalized() . . . 392
11.22.1.6 fgemv_kgf() . . . 392
11.22.1.7 LUKrylov() . . . 392
11.22.1.8 Danilevski() . . . 393
11.22.1.9 RandomKrylovPrecond() . . . 393
11.22.1.10 ArithProg() . . . 393
11.22.1.11 LUKrylov_KGFast() . . . 393
11.22.1.12 MatVecMinPoly() . . . 394
11.22.1.13 Hybrid_KGF_LUK_MinPoly() . . . 394
11.22.1.14 updateD() . . . 394
11.22.1.15 newD() . . . 394
11.22.1.16 CompressRows() . . . 394
11.22.1.17 CompressRowsQK() . . . 395
11.22.1.18 DeCompressRows() . . . 395
11.22.1.19 DeCompressRowsQK() . . . 395
11.22.1.20 CompressRowsQA() . . . 395
11.22.1.21 DeCompressRowsQA() . . . 396
11.22.1.22 LUdivine_construct() [2/2] . . . 396
11.23 Givaro Namespace Reference . . . 396
11.24 MKL_CONFIG Namespace Reference . . . 396
11.25 RecInt Namespace Reference . . . 396

12 Data Structure Documentation **397**
12.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference . . . 397
12.1.1 Member Typedef Documentation . . . 397

12.1.1.1 value . . . 397
12.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference . . . 397
12.2.1 Member Typedef Documentation . . . 397
12.2.1.1 value [1/2] . . . 397
12.2.1.2 value [2/2] . . . 397
12.3 ALL< v > Struct Template Reference . . . 397
12.4 ALL< false, v... > Struct Template Reference . . . 398
12.4.1 Field Documentation . . . 398
12.4.1.1 value . . . 398
12.5 ALL< true, v... > Struct Template Reference . . . 398
12.5.1 Field Documentation . . . 398
12.5.1.1 value . . . 398
12.6 ALL<> Struct Reference . . . 398
12.6.1 Field Documentation . . . 398
12.6.1.1 value . . . 398
12.7 ArbitraryPrecIntTag Struct Reference . . . 398
12.7.1 Detailed Description . . . 398
12.8 AreEqual< X, Y > Class Template Reference . . . 399
12.8.1 Field Documentation . . . 399
12.8.1.1 value . . . 399
12.9 AreEqual< X, X > Class Template Reference . . . 399
12.9.1 Field Documentation . . . 399
12.9.1.1 value [1/2] . . . 399
12.9.1.2 value [2/2] . . . 399
12.10 Argument Struct Reference . . . 399
12.10.1 Field Documentation . . . 399
12.10.1.1 c . . . 399
12.10.1.2 example . . . 400
12.10.1.3 helpString . . . 400
12.10.1.4 type . . . 400
12.10.1.5 data . . . 400
12.11 array< T > Class Template Reference . . . 400
12.11.1 Detailed Description . . . 400
12.11.2 Field Documentation . . . 400
12.11.2.1 elements . . . 400
12.12 associatedDelayedField< Field > Struct Template Reference . . . 400
12.12.1 Member Typedef Documentation . . . 401
12.12.1.1 field . . . 401
12.12.1.2 type . . . 401
12.13 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference 401
12.13.1 Member Typedef Documentation . . . 401

12.13.1.1 field [1/2] . . . 401
12.13.1.2 type [1/2] . . . 401
12.13.1.3 field [2/2] . . . 401
12.13.1.4 type [2/2] . . . 401
12.14 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference . . . 401
12.14.1 Member Typedef Documentation . . . 402
12.14.1.1 field [1/2] . . . 402
12.14.1.2 type [1/2] . . . 402
12.14.1.3 field [2/2] . . . 402
12.14.1.4 type [2/2] . . . 402
12.15 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference . . . 402
12.15.1 Member Typedef Documentation . . . 402
12.15.1.1 field [1/2] . . . 402
12.15.1.2 type [1/2] . . . 402
12.15.1.3 field [2/2] . . . 402
12.15.1.4 type [2/2] . . . 403
12.16 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference . . . 403
12.16.1 Member Typedef Documentation . . . 403
12.16.1.1 field [1/2] . . . 403
12.16.1.2 type [1/2] . . . 403
12.16.1.3 field [2/2] . . . 403
12.16.1.4 type [2/2] . . . 403
12.17 Auto Struct Reference . . . 403
12.18 Bench< Elt > Class Template Reference . . . 403
12.18.1 Member Typedef Documentation . . . 404
12.18.1.1 Field . . . 404
12.18.1.2 Elt_ptr . . . 404
12.18.1.3 Residu . . . 404
12.18.1.4 enable_if_t . . . 404
12.18.1.5 is_same_element . . . 405
12.18.1.6 enable_if_no_simd_t . . . 405
12.18.1.7 enable_if_simd128_t . . . 405
12.18.1.8 enable_if_simd256_t . . . 405
12.18.1.9 enable_if_simd512_t . . . 405
12.18.2 Constructor & Destructor Documentation . . . 405
12.18.2.1 Bench() . . . 405
12.18.3 Member Function Documentation . . . 405
12.18.3.1 cardinality() [1/2] . . . 405
12.18.3.2 cardinality() [2/2] . . . 405
12.18.3.3 doBenchs() . . . 405
12.18.3.4 run() . . . 406
12.18.4 Field Documentation . . . 406

12.18.4.1 F . . . 406
12.18.4.2 m . . . 406
12.18.4.3 n . . . 406
12.18.4.4 iters . . . 406
12.18.4.5 inplace . . . 406
12.19 Bini Struct Reference . . . 406
12.20 Block Struct Reference . . . 406
12.21 BlockTransposeSIMD< Field, Simd, > Struct Template Reference . . . 406
12.21.1 Member Function Documentation . . . 407
12.21.1.1 size() . . . 407
12.21.1.2 info() . . . 407
12.21.1.3 transpose() [1/5] . . . 407
12.21.1.4 transpose() [2/5] . . . 407
12.21.1.5 transpose() [3/5] . . . 408
12.21.1.6 transpose() [4/5] . . . 408
12.21.1.7 transpose() [5/5] . . . 408
12.22 callLUdivine_small< Element > Class Template Reference . . . 408
12.22.1 Member Function Documentation . . . 408
12.22.1.1 operator()() . . . 408
12.23 callLUdivine_small< double > Class Reference . . . 409
12.23.1 Member Function Documentation . . . 409
12.23.1.1 operator()() [1/2] . . . 409
12.23.1.2 operator()() [2/2] . . . 409
12.24 callLUdivine_small< float > Class Reference . . . 410
12.24.1 Member Function Documentation . . . 410
12.24.1.1 operator()() [1/2] . . . 410
12.24.1.2 operator()() [2/2] . . . 410
12.25 CharpolyFailed Class Reference . . . 410
12.26 Checker_Empty< Field > Struct Template Reference . . . 410
12.26.1 Constructor & Destructor Documentation . . . 411
12.26.1.1 Checker_Empty() . . . 411
12.26.2 Member Function Documentation . . . 411
12.26.2.1 check() . . . 411
12.27 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference . . . 411
12.27.1 Constructor & Destructor Documentation . . . 411
12.27.1.1 CheckerImplem_charpoly() [1/2] . . . 411
12.27.1.2 CheckerImplem_charpoly() [2/2] . . . 411
12.27.1.3 ~CheckerImplem_charpoly() . . . 412
12.27.2 Member Function Documentation . . . 412
12.27.2.1 check() . . . 412
12.28 CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > Class Template Reference . . . 412

12.28.1 Member Typedef Documentation . . . 412
12.28.1.1 Ring . . . 412
12.28.2 Constructor & Destructor Documentation . . . 412
12.28.2.1 CheckerImplem_charpoly() [1/4] . . . 412
12.28.2.2 CheckerImplem_charpoly() [2/4] . . . 413
12.28.2.3 ∼CheckerImplem_charpoly() [1/2] . . . 413
12.28.2.4 CheckerImplem_charpoly() [3/4] . . . 413
12.28.2.5 CheckerImplem_charpoly() [4/4] . . . 413
12.28.2.6 ∼CheckerImplem_charpoly() [2/2] . . . 413
12.28.3 Member Function Documentation . . . 413
12.28.3.1 check() [1/2] . . . 413
12.28.3.2 check() [2/2] . . . 413
12.29 CheckerImplem_Det< Field > Class Template Reference . . . 413
12.29.1 Constructor & Destructor Documentation . . . 414
12.29.1.1 CheckerImplem_Det() [1/2] . . . 414
12.29.1.2 CheckerImplem_Det() [2/2] . . . 414
12.29.1.3 ∼CheckerImplem_Det() . . . 414
12.29.2 Member Function Documentation . . . 414
12.29.2.1 check() . . . 414
12.30 CheckerImplem_fgemm< Field > Class Template Reference . . . 414
12.30.1 Constructor & Destructor Documentation . . . 415
12.30.1.1 CheckerImplem_fgemm() [1/2] . . . 415
12.30.1.2 CheckerImplem_fgemm() [2/2] . . . 415
12.30.1.3 ∼CheckerImplem_fgemm() . . . 415
12.30.2 Member Function Documentation . . . 415
12.30.2.1 check() . . . 415
12.31 CheckerImplem_ftrsm< Field > Class Template Reference . . . 416
12.31.1 Constructor & Destructor Documentation . . . 416
12.31.1.1 CheckerImplem_ftrsm() [1/2] . . . 416
12.31.1.2 CheckerImplem_ftrsm() [2/2] . . . 416
12.31.1.3 ∼CheckerImplem_ftrsm() . . . 416
12.31.2 Member Function Documentation . . . 416
12.31.2.1 check() . . . 416
12.32 CheckerImplem_invert< Field > Class Template Reference . . . 417
12.32.1 Constructor & Destructor Documentation . . . 417
12.32.1.1 CheckerImplem_invert() [1/2] . . . 417
12.32.1.2 CheckerImplem_invert() [2/2] . . . 417
12.32.1.3 ∼CheckerImplem_invert() . . . 417
12.32.2 Member Function Documentation . . . 417
12.32.2.1 check() . . . 417
12.33 CheckerImplem_PLUQ< Field > Class Template Reference . . . 417
12.33.1 Constructor & Destructor Documentation . . . 418

12.33.1.1 CheckerImplem_PLUQ() [1/2] . . . 418
12.33.1.2 CheckerImplem_PLUQ() [2/2] . . . 418
12.33.1.3 ~CheckerImplem_PLUQ() . . . 418
12.33.2 Member Function Documentation . . . 418
12.33.2.1 check() . . . 418
12.34 Classic Struct Reference . . . 418
12.35 Column Struct Reference . . . 419
12.36 CompactElement< Element > Struct Template Reference . . . 419
12.36.1 Member Typedef Documentation . . . 419
12.36.1.1 type . . . 419
12.37 CompactElement< double > Struct Reference . . . 419
12.37.1 Member Typedef Documentation . . . 419
12.37.1.1 type [1/2] . . . 419
12.37.1.2 type [2/2] . . . 419
12.38 CompactElement< float > Struct Reference . . . 419
12.38.1 Member Typedef Documentation . . . 419
12.38.1.1 type [1/2] . . . 419
12.38.1.2 type [2/2] . . . 419
12.39 CompactElement< int16_t > Struct Reference . . . 420
12.39.1 Member Typedef Documentation . . . 420
12.39.1.1 type [1/2] . . . 420
12.39.1.2 type [2/2] . . . 420
12.40 CompactElement< int32_t > Struct Reference . . . 420
12.40.1 Member Typedef Documentation . . . 420
12.40.1.1 type [1/2] . . . 420
12.40.1.2 type [2/2] . . . 420
12.41 CompactElement< int64_t > Struct Reference . . . 420
12.41.1 Member Typedef Documentation . . . 420
12.41.1.1 type [1/2] . . . 420
12.41.1.2 type [2/2] . . . 420
12.42 compatible_data_type< Field > Struct Template Reference . . . 421
12.42.1 Field Documentation . . . 421
12.42.1.1 value . . . 421
12.43 compatible_data_type< Givaro::ZRing< double > > Struct Reference . . . 421
12.43.1 Field Documentation . . . 421
12.43.1.1 value [1/2] . . . 421
12.43.1.2 value [2/2] . . . 421
12.44 compatible_data_type< Givaro::ZRing< float > > Struct Reference . . . 421
12.44.1 Field Documentation . . . 421
12.44.1.1 value [1/2] . . . 421
12.44.1.2 value [2/2] . . . 421
12.45 Compose< H1, H2 > Struct Template Reference . . . 422

12.45.1 Constructor & Destructor Documentation . . . 422
12.45.1.1 Compose() [1/5] . . . 422
12.45.1.2 Compose() [2/5] . . . 422
12.45.1.3 Compose() [3/5] . . . 422
12.45.1.4 Compose() [4/5] . . . 422
12.45.1.5 Compose() [5/5] . . . 422
12.45.2 Member Function Documentation . . . 422
12.45.2.1 first_component() . . . 422
12.45.2.2 second_component() . . . 422
12.45.3 Friends And Related Symbol Documentation . . . 423
12.45.3.1 operator<< . . . 423
12.46 array< T >::const_iterator Class Reference . . . 423
12.46.1 Detailed Description . . . 423
12.47 string::const_iterator Class Reference . . . 423
12.47.1 Detailed Description . . . 423
12.48 vector< T >::const_iterator Class Reference . . . 423
12.48.1 Detailed Description . . . 423
12.49 array< T >::const_reverse_iterator Class Reference . . . 423
12.49.1 Detailed Description . . . 423
12.50 string::const_reverse_iterator Class Reference . . . 423
12.50.1 Detailed Description . . . 423
12.51 vector< T >::const_reverse_iterator Class Reference . . . 424
12.51.1 Detailed Description . . . 424
12.52 Simd128_impl< true, true, false, 2 >::Converter Union Reference . . . 424
12.52.1 Field Documentation . . . 424
12.52.1.1 v . . . 424
12.52.1.2 t . . . 424
12.53 Simd128_impl< true, true, false, 4 >::Converter Union Reference . . . 424
12.53.1 Field Documentation . . . 424
12.53.1.1 v . . . 424
12.53.1.2 t . . . 424
12.54 Simd128_impl< true, true, false, 8 >::Converter Union Reference . . . 424
12.54.1 Field Documentation . . . 425
12.54.1.1 v . . . 425
12.54.1.2 t . . . 425
12.55 Simd128_impl< true, true, true, 2 >::Converter Union Reference . . . 425
12.55.1 Field Documentation . . . 425
12.55.1.1 v . . . 425
12.55.1.2 t . . . 425
12.56 Simd128_impl< true, true, true, 4 >::Converter Union Reference . . . 425
12.56.1 Field Documentation . . . 425
12.56.1.1 v . . . 425

12.56.1.2 t . . . 425
12.57 Simd128_impl< true, true, true, 8 >::Converter Union Reference . . . 425
12.57.1 Field Documentation . . . 426
12.57.1.1 v . . . 426
12.57.1.2 t . . . 426
12.58 Simd256_impl< true, false, true, 8 >::Converter Union Reference . . . 426
12.58.1 Field Documentation . . . 426
12.58.1.1 v . . . 426
12.58.1.2 t . . . 426
12.59 Simd256_impl< true, true, false, 2 >::Converter Union Reference . . . 426
12.59.1 Field Documentation . . . 426
12.59.1.1 v . . . 426
12.59.1.2 t . . . 426
12.60 Simd256_impl< true, true, false, 4 >::Converter Union Reference . . . 426
12.60.1 Field Documentation . . . 427
12.60.1.1 v . . . 427
12.60.1.2 t . . . 427
12.61 Simd256_impl< true, true, false, 8 >::Converter Union Reference . . . 427
12.61.1 Field Documentation . . . 427
12.61.1.1 v . . . 427
12.61.1.2 t . . . 427
12.62 Simd256_impl< true, true, true, 2 >::Converter Union Reference . . . 427
12.62.1 Field Documentation . . . 427
12.62.1.1 v . . . 427
12.62.1.2 t . . . 427
12.63 Simd256_impl< true, true, true, 4 >::Converter Union Reference . . . 427
12.63.1 Field Documentation . . . 428
12.63.1.1 v . . . 428
12.63.1.2 t . . . 428
12.64 Simd256_impl< true, true, true, 8 >::Converter Union Reference . . . 428
12.64.1 Field Documentation . . . 428
12.64.1.1 v . . . 428
12.64.1.2 t . . . 428
12.65 Simd512_impl< true, true, false, 8 >::Converter Union Reference . . . 428
12.65.1 Field Documentation . . . 428
12.65.1.1 v . . . 428
12.65.1.2 t . . . 428
12.66 Simd512_impl< true, true, true, 8 >::Converter Union Reference . . . 428
12.66.1 Field Documentation . . . 429
12.66.1.1 v . . . 429
12.66.1.2 t . . . 429
12.67 ConvertTo< T > Struct Template Reference . . . 429

12.67.1 Detailed Description . . . 429
12.68 Coo< ValT, IdxT > Struct Template Reference . . . 429
12.68.1 Member Typedef Documentation . . . 430
12.68.1.1 Self . . . 430
12.68.2 Constructor & Destructor Documentation . . . 430
12.68.2.1 Coo() [1/4] . . . 430
12.68.2.2 Coo() [2/4] . . . 430
12.68.2.3 Coo() [3/4] . . . 430
12.68.2.4 Coo() [4/4] . . . 430
12.68.3 Member Function Documentation . . . 430
12.68.3.1 operator=() [1/2] . . . 430
12.68.3.2 operator=() [2/2] . . . 430
12.68.4 Field Documentation . . . 430
12.68.4.1 val . . . 430
12.68.4.2 row . . . 430
12.68.4.3 col . . . 431
12.69 Coo< Field > Struct Template Reference . . . 431
12.69.1 Constructor & Destructor Documentation . . . 431
12.69.1.1 Coo() [1/4] . . . 431
12.69.1.2 Coo() [2/4] . . . 431
12.69.1.3 Coo() [3/4] . . . 431
12.69.1.4 Coo() [4/4] . . . 431
12.69.2 Member Function Documentation . . . 432
12.69.2.1 operator=() [1/2] . . . 432
12.69.2.2 operator=() [2/2] . . . 432
12.69.3 Field Documentation . . . 432
12.69.3.1 val . . . 432
12.69.3.2 col . . . 432
12.69.3.3 row . . . 432
12.69.3.4 deleted . . . 432
12.70 Coo< ValT, IdxT > Struct Template Reference . . . 432
12.70.1 Member Typedef Documentation . . . 433
12.70.1.1 Self . . . 433
12.70.2 Constructor & Destructor Documentation . . . 433
12.70.2.1 Coo() [1/4] . . . 433
12.70.2.2 Coo() [2/4] . . . 433
12.70.2.3 Coo() [3/4] . . . 433
12.70.2.4 Coo() [4/4] . . . 433
12.70.3 Member Function Documentation . . . 433
12.70.3.1 operator=() [1/2] . . . 433
12.70.3.2 operator=() [2/2] . . . 433
12.70.4 Field Documentation . . . 433

12.70.4.1 val . . . 433
12.70.4.2 row . . . 433
12.70.4.3 col . . . 434
12.71 CooMat< Field > Struct Template Reference . . . 434
12.71.1 Field Documentation . . . 434
12.71.1.1 _coo16 . . . 434
12.71.1.2 _coo32 . . . 434
12.71.1.3 _coo64 . . . 434
12.71.1.4 _coo16_zo . . . 434
12.71.1.5 _coo32_zo . . . 434
12.71.1.6 _coo64_zo . . . 434
12.72 count_nonconst_lvalue_reference< T > Struct Template Reference . . . 435
12.73 count_nonconst_lvalue_reference< const T &, O... > Struct Template Reference . . . 435
12.73.1 Field Documentation . . . 435
12.73.1.1 n . . . 435
12.74 count_nonconst_lvalue_reference< T &, O... > Struct Template Reference . . . 435
12.74.1 Field Documentation . . . 435
12.74.1.1 n . . . 435
12.75 count_nonconst_lvalue_reference< T, O... > Struct Template Reference . . . 435
12.75.1 Field Documentation . . . 435
12.75.1.1 n . . . 435
12.76 count_nonconst_lvalue_reference<> Struct Reference . . . 436
12.76.1 Field Documentation . . . 436
12.76.1.1 n . . . 436
12.77 CsrMat< Field > Struct Template Reference . . . 436
12.77.1 Field Documentation . . . 436
12.77.1.1 _csr16 . . . 436
12.77.1.2 _csr32 . . . 436
12.77.1.3 _csr64 . . . 436
12.77.1.4 _csr16_zo . . . 436
12.77.1.5 _csr32_zo . . . 436
12.77.1.6 _csr64_zo . . . 436
12.78 DefaultBoundedTag Struct Reference . . . 437
12.78.1 Detailed Description . . . 437
12.79 DefaultTag Struct Reference . . . 437
12.79.1 Detailed Description . . . 437
12.80 DelayedTag Struct Reference . . . 437
12.80.1 Detailed Description . . . 437
12.81 DivideAndConquer Struct Reference . . . 437
12.82 ElementTraits< Element > Struct Template Reference . . . 437
12.82.1 Detailed Description . . . 437
12.82.2 Member Typedef Documentation . . . 438

12.82.2.1 value . . . 438
12.83 ElementTraits< double > Struct Reference . . . 438
12.83.1 Member Typedef Documentation . . . 438
12.83.1.1 value [1/2] . . . 438
12.83.1.2 value [2/2] . . . 438
12.84 ElementTraits< FFPACK::rns_double_elt > Struct Reference . . . 438
12.84.1 Member Typedef Documentation . . . 438
12.84.1.1 value [1/2] . . . 438
12.84.1.2 value [2/2] . . . 438
12.85 ElementTraits< float > Struct Reference . . . 438
12.85.1 Member Typedef Documentation . . . 439
12.85.1.1 value [1/2] . . . 439
12.85.1.2 value [2/2] . . . 439
12.86 ElementTraits< Givaro::Integer > Struct Reference . . . 439
12.86.1 Member Typedef Documentation . . . 439
12.86.1.1 value [1/2] . . . 439
12.86.1.2 value [2/2] . . . 439
12.87 ElementTraits< int16_t > Struct Reference . . . 439
12.87.1 Member Typedef Documentation . . . 439
12.87.1.1 value [1/2] . . . 439
12.87.1.2 value [2/2] . . . 439
12.88 ElementTraits< int32_t > Struct Reference . . . 440
12.88.1 Member Typedef Documentation . . . 440
12.88.1.1 value [1/2] . . . 440
12.88.1.2 value [2/2] . . . 440
12.89 ElementTraits< int64_t > Struct Reference . . . 440
12.89.1 Member Typedef Documentation . . . 440
12.89.1.1 value [1/2] . . . 440
12.89.1.2 value [2/2] . . . 440
12.90 ElementTraits< int8_t > Struct Reference . . . 440
12.90.1 Member Typedef Documentation . . . 440
12.90.1.1 value [1/2] . . . 440
12.90.1.2 value [2/2] . . . 441
12.91 ElementTraits< RecInt::rint< K > > Struct Template Reference . . . 441
12.91.1 Member Typedef Documentation . . . 441
12.91.1.1 value [1/2] . . . 441
12.91.1.2 value [2/2] . . . 441
12.92 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference . . . 441
12.92.1 Member Typedef Documentation . . . 441
12.92.1.1 value [1/2] . . . 441
12.92.1.2 value [2/2] . . . 441
12.93 ElementTraits< RecInt::ruint< K > > Struct Template Reference . . . 441

12.93.1 Member Typedef Documentation . . . 442
12.93.1.1 value [1/2] . . . 442
12.93.1.2 value [2/2] . . . 442
12.94 ElementTraits< uint16_t > Struct Reference . . . 442
12.94.1 Member Typedef Documentation . . . 442
12.94.1.1 value [1/2] . . . 442
12.94.1.2 value [2/2] . . . 442
12.95 ElementTraits< uint32_t > Struct Reference . . . 442
12.95.1 Member Typedef Documentation . . . 442
12.95.1.1 value [1/2] . . . 442
12.95.1.2 value [2/2] . . . 443
12.96 ElementTraits< uint64_t > Struct Reference . . . 443
12.96.1 Member Typedef Documentation . . . 443
12.96.1.1 value [1/2] . . . 443
12.96.1.2 value [2/2] . . . 443
12.97 ElementTraits< uint8_t > Struct Reference . . . 443
12.97.1 Member Typedef Documentation . . . 443
12.97.1.1 value [1/2] . . . 443
12.97.1.2 value [2/2] . . . 443
12.98 EllMat< Field > Struct Template Reference . . . 443
12.98.1 Field Documentation . . . 444
12.98.1.1 _ell16 . . . 444
12.98.1.2 _ell32 . . . 444
12.98.1.3 _ell64 . . . 444
12.98.1.4 _ell16_zo . . . 444
12.98.1.5 _ell32_zo . . . 444
12.98.1.6 _ell64_zo . . . 444
12.99 Failure Class Reference . . . 444
12.99.1 Detailed Description . . . 445
12.99.2 Constructor & Destructor Documentation . . . 445
12.99.2.1 Failure() . . . 445
12.99.3 Member Function Documentation . . . 445
12.99.3.1 operator()() [1/2] . . . 445
12.99.3.2 operator()() [2/2] . . . 445
12.99.3.3 setErrorStream() . . . 445
12.99.3.4 print() . . . 445
12.99.4 Field Documentation . . . 446
12.99.4.1 _errorStream . . . 446
12.100 FailureCharpolyCheck Class Reference . . . 446
12.101 FailureDetCheck Class Reference . . . 446
12.102 FailureFgemmCheck Class Reference . . . 446
12.103 FailureInvertCheck Class Reference . . . 446

12.104 FailurePLUQCheck Class Reference . . . 446
12.105 FailureTrsmCheck Class Reference . . . 446
12.106 FieldSimd< _Field > Class Template Reference . . . 447
12.106.1 Member Typedef Documentation . . . 448
12.106.1.1 Field . . . 448
12.106.1.2 Element . . . 448
12.106.1.3 simd . . . 448
12.106.1.4 vect_t . . . 448
12.106.1.5 scalar_t . . . 448
12.106.2 Constructor & Destructor Documentation . . . 448
12.106.2.1 FieldSimd() [1/3] . . . 448
12.106.2.2 FieldSimd() [2/3] . . . 448
12.106.2.3 FieldSimd() [3/3] . . . 448
12.106.3 Member Function Documentation . . . 448
12.106.3.1 operator=() [1/2] . . . 448
12.106.3.2 operator=() [2/2] . . . 448
12.106.3.3 init() [1/2] . . . 449
12.106.3.4 init() [2/2] . . . 449
12.106.3.5 add() [1/2] . . . 449
12.106.3.6 add() [2/2] . . . 449
12.106.3.7 addin() . . . 449
12.106.3.8 add_r() [1/2] . . . 449
12.106.3.9 add_r() [2/2] . . . 449
12.106.3.10 addin_r() . . . 449
12.106.3.11 sub() [1/2] . . . 450
12.106.3.12 sub() [2/2] . . . 450
12.106.3.13 subin() . . . 450
12.106.3.14 sub_r() [1/2] . . . 450
12.106.3.15 sub_r() [2/2] . . . 450
12.106.3.16 subin_r() . . . 450
12.106.3.17 zero() [1/2] . . . 450
12.106.3.18 zero() [2/2] . . . 450
12.106.3.19 mod() . . . 450
12.106.3.20 mul() [1/2] . . . 451
12.106.3.21 mul() [2/2] . . . 451
12.106.3.22 mulin() . . . 451
12.106.3.23 mul_r() [1/2] . . . 451
12.106.3.24 mul_r() [2/2] . . . 451
12.106.3.25 axpy() [1/2] . . . 451
12.106.3.26 axpy() [2/2] . . . 451
12.106.3.27 axpyin() . . . 451
12.106.3.28 axpy_r() [1/2] . . . 452

12.106.3.29 axpy_r() [2/2] . . . 452
12.106.3.30 axpyin_r() . . . 452
12.106.3.31 maxpy() [1/2] . . . 452
12.106.3.32 maxpy() [2/2] . . . 452
12.106.3.33 maxpyin() . . . 452
12.106.4 Field Documentation . . . 452
12.106.4.1 vect_size . . . 452
12.106.4.2 alignment . . . 453
12.107 FieldTraits< Field > Struct Template Reference . . . 453
12.107.1 Detailed Description . . . 453
12.107.2 Member Typedef Documentation . . . 453
12.107.2.1 category . . . 453
12.107.3 Field Documentation . . . 453
12.107.3.1 balanced . . . 453
12.108 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference . . . 453
12.108.1 Member Typedef Documentation . . . 454
12.108.1.1 category [1/2] . . . 454
12.108.1.2 category [2/2] . . . 454
12.108.2 Field Documentation . . . 454
12.108.2.1 balanced [1/2] . . . 454
12.108.2.2 balanced [2/2] . . . 454
12.109 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference . . . 454
12.109.1 Member Typedef Documentation . . . 454
12.109.1.1 category [1/2] . . . 454
12.109.1.2 category [2/2] . . . 454
12.109.2 Field Documentation . . . 454
12.109.2.1 balanced [1/2] . . . 454
12.109.2.2 balanced [2/2] . . . 455
12.110 FieldTraits< Givaro::Modular< Element > > Struct Template Reference . . . 455
12.110.1 Member Typedef Documentation . . . 455
12.110.1.1 category [1/2] . . . 455
12.110.1.2 category [2/2] . . . 455
12.110.2 Field Documentation . . . 455
12.110.2.1 balanced [1/2] . . . 455
12.110.2.2 balanced [2/2] . . . 455
12.111 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference . . . 455
12.111.1 Member Typedef Documentation . . . 456
12.111.1.1 category [1/2] . . . 456
12.111.1.2 category [2/2] . . . 456
12.111.2 Field Documentation . . . 456
12.111.2.1 balanced [1/2] . . . 456
12.111.2.2 balanced [2/2] . . . 456

12.112 FieldTraits< Givaro::ZRing< double > > Struct Reference . . . 456
12.112.1 Member Typedef Documentation . . . 456
12.112.1.1 category [1/2] . . . 456
12.112.1.2 category [2/2] . . . 456
12.112.2 Field Documentation . . . 456
12.112.2.1 balanced [1/2] . . . 456
12.112.2.2 balanced [2/2] . . . 457
12.113 FieldTraits< Givaro::ZRing< float > > Struct Reference . . . 457
12.113.1 Member Typedef Documentation . . . 457
12.113.1.1 category [1/2] . . . 457
12.113.1.2 category [2/2] . . . 457
12.113.2 Field Documentation . . . 457
12.113.2.1 balanced [1/2] . . . 457
12.113.2.2 balanced [2/2] . . . 457
12.114 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference . . . 457
12.114.1 Member Typedef Documentation . . . 458
12.114.1.1 category [1/2] . . . 458
12.114.1.2 category [2/2] . . . 458
12.114.2 Field Documentation . . . 458
12.114.2.1 balanced [1/2] . . . 458
12.114.2.2 balanced [2/2] . . . 458
12.115 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference . . . 458
12.115.1 Member Typedef Documentation . . . 458
12.115.1.1 category [1/2] . . . 458
12.115.1.2 category [2/2] . . . 458
12.115.2 Field Documentation . . . 458
12.115.2.1 balanced [1/2] . . . 458
12.115.2.2 balanced [2/2] . . . 458
12.116 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference . . . 459
12.116.1 Member Typedef Documentation . . . 459
12.116.1.1 category [1/2] . . . 459
12.116.1.2 category [2/2] . . . 459
12.116.2 Field Documentation . . . 459
12.116.2.1 balanced [1/2] . . . 459
12.116.2.2 balanced [2/2] . . . 459
12.117 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference . . . 459
12.117.1 Member Typedef Documentation . . . 459
12.117.1.1 category [1/2] . . . 459
12.117.1.2 category [2/2] . . . 459
12.117.2 Field Documentation . . . 460
12.117.2.1 balanced [1/2] . . . 460
12.117.2.2 balanced [2/2] . . . 460

12.118 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference 460
12.118.1 Member Typedef Documentation 460
12.118.1.1 category [1/2] 460
12.118.1.2 category [2/2] 460
12.118.2 Field Documentation 460
12.118.2.1 balanced [1/2] 460
12.118.2.2 balanced [2/2] 460
12.119 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference 460
12.119.1 Member Typedef Documentation 461
12.119.1.1 category [1/2] 461
12.119.1.2 category [2/2] 461
12.119.2 Field Documentation 461
12.119.2.1 balanced [1/2] 461
12.119.2.2 balanced [2/2] 461
12.120 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference 461
12.120.1 Member Typedef Documentation 461
12.120.1.1 category [1/2] 461
12.120.1.2 category [2/2] 461
12.120.2 Field Documentation 462
12.120.2.1 balanced [1/2] 462
12.120.2.2 balanced [2/2] 462
12.121 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference 462
12.121.1 Member Typedef Documentation 462
12.121.1.1 category [1/2] 462
12.121.1.2 category [2/2] 462
12.121.2 Field Documentation 462
12.121.2.1 balanced [1/2] 462
12.121.2.2 balanced [2/2] 462
12.122 Fixed Struct Reference 462
12.123 FixedPrecIntTag Struct Reference 462
12.123.1 Detailed Description 463
12.124 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution Class Reference 463
12.124.1 Member Typedef Documentation 463
12.124.1.1 IntType 463
12.124.2 Constructor & Destructor Documentation 463
12.124.2.1 FloatingPointTestDistribution() 463
12.124.3 Member Function Documentation 463
12.124.3.1 operator()() 463
12.125 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference 463
12.125.1 Constructor & Destructor Documentation 464
12.125.1.1 ForStrategy1D() [1/2] 464

12.125.1.2 ForStrategy1D() [2/2] . . . 464
12.125.2 Member Function Documentation . . . 464
12.125.2.1 build() . . . 464
12.125.2.2 initialize() . . . 464
12.125.2.3 isTerminated() . . . 464
12.125.2.4 begin() . . . 465
12.125.2.5 end() . . . 465
12.125.2.6 numblocks() . . . 465
12.125.2.7 blockindex() . . . 465
12.125.2.8 operator++() . . . 465
12.125.3 Field Documentation . . . 465
12.125.3.1 ibeg . . . 465
12.125.3.2 iend . . . 465
12.125.3.3 current . . . 465
12.125.3.4 firstBlockSize . . . 465
12.125.3.5 lastBlockSize . . . 465
12.125.3.6 changeBS . . . 466
12.125.3.7 numBlock . . . 466
12.126 ForStrategy2D< blocksize_t, Cut, Param > Struct Template Reference . . . 466
12.126.1 Constructor & Destructor Documentation . . . 467
12.126.1.1 ForStrategy2D() . . . 467
12.126.2 Member Function Documentation . . . 467
12.126.2.1 initialize() . . . 467
12.126.2.2 isTerminated() . . . 467
12.126.2.3 ibegin() . . . 467
12.126.2.4 jbegin() . . . 467
12.126.2.5 iend() . . . 467
12.126.2.6 jend() . . . 467
12.126.2.7 operator++() . . . 467
12.126.2.8 rownumblocks() . . . 467
12.126.2.9 colnumblocks() . . . 468
12.126.2.10 blockindex() . . . 468
12.126.2.11 rowblockindex() . . . 468
12.126.2.12 colblockindex() . . . 468
12.126.3 Friends And Related Symbol Documentation . . . 468
12.126.3.1 operator<< . . . 468
12.126.4 Field Documentation . . . 468
12.126.4.1 _ibeg . . . 468
12.126.4.2 _iend . . . 468
12.126.4.3 _jbeg . . . 468
12.126.4.4 _jend . . . 468
12.126.4.5 rowBlockSize . . . 469

12.126.4.6 colBlockSize . . . 469
12.126.4.7 current . . . 469
12.126.4.8 lastRBS . . . 469
12.126.4.9 lastCBS . . . 469
12.126.4.10 changeRBS . . . 469
12.126.4.11 changeCBS . . . 469
12.126.4.12 numRowBlock . . . 469
12.126.4.13 numColBlock . . . 469
12.126.4.14 BLOCKS . . . 469
12.127 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . 470
12.128 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference . . . 470
12.129 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference . . . 470
12.130 ftrmmLeftLowerTransUnit< Element > Class Template Reference . . . 470
12.131 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . 470
12.132 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference . . . 470
12.133 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference . . . 470
12.134 ftrmmLeftUpperTransUnit< Element > Class Template Reference . . . 471
12.135 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference . . . 471
12.136 ftrmmRightLowerNoTransUnit< Element > Class Template Reference . . . 471
12.137 ftrmmRightLowerTransNonUnit< Element > Class Template Reference . . . 471
12.138 ftrmmRightLowerTransUnit< Element > Class Template Reference . . . 471
12.139 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference . . . 471
12.140 ftrmmRightUpperNoTransUnit< Element > Class Template Reference . . . 471
12.141 ftrmmRightUpperTransNonUnit< Element > Class Template Reference . . . 471
12.142 ftrmmRightUpperTransUnit< Element > Class Template Reference . . . 472
12.143 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . 472
12.144 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference . . . 472
12.145 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference . . . 472
12.146 ftrsmLeftLowerTransUnit< Element > Class Template Reference . . . 472
12.147 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . 472
12.147.1 Detailed Description . . . 472
12.148 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference . . . 473
12.149 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference . . . 473
12.150 ftrsmLeftUpperTransUnit< Element > Class Template Reference . . . 473
12.151 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference . . . 473
12.152 ftrsmRightLowerNoTransUnit< Element > Class Template Reference . . . 473
12.153 ftrsmRightLowerTransNonUnit< Element > Class Template Reference . . . 473
12.154 ftrsmRightLowerTransUnit< Element > Class Template Reference . . . 473
12.155 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference . . . 473
12.156 ftrsmRightUpperNoTransUnit< Element > Class Template Reference . . . 474
12.157 ftrsmRightUpperTransNonUnit< Element > Class Template Reference . . . 474
12.158 ftrsmRightUpperTransUnit< Element > Class Template Reference . . . 474

12.159 GenericTag Struct Reference . . . 474
12.159.1 Detailed Description . . . 474
12.160 GenericTag Struct Reference . . . 474
12.160.1 Detailed Description . . . 474
12.161 Grain Struct Reference . . . 474
12.162 has_minus_eq_impl< C > Struct Template Reference . . . 474
12.162.1 Field Documentation . . . 475
12.162.1.1 value . . . 475
12.163 has_minus_impl< C > Struct Template Reference . . . 475
12.163.1 Field Documentation . . . 475
12.163.1.1 value . . . 475
12.164 has_mul_eq_impl< C > Struct Template Reference . . . 475
12.164.1 Field Documentation . . . 475
12.164.1.1 value . . . 475
12.165 has_mul_impl< C > Struct Template Reference . . . 475
12.165.1 Field Documentation . . . 476
12.165.1.1 value . . . 476
12.166 has_operation< T > Struct Template Reference . . . 476
12.166.1 Field Documentation . . . 476
12.166.1.1 value . . . 476
12.167 has_plus_eq_impl< C > Struct Template Reference . . . 476
12.167.1 Field Documentation . . . 476
12.167.1.1 value . . . 476
12.168 has_plus_impl< C > Struct Template Reference . . . 476
12.168.1 Field Documentation . . . 477
12.168.1.1 value . . . 477
12.169 HelperFlag Struct Reference . . . 477
12.169.1 Field Documentation . . . 477
12.169.1.1 none . . . 477
12.169.1.2 coo . . . 477
12.169.1.3 csr . . . 477
12.169.1.4 ell . . . 477
12.169.1.5 aut . . . 477
12.169.1.6 pm1 . . . 477
12.170 HelperMod< Field, ElementTraits > Struct Template Reference . . . 477
12.171 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference . . . 478
12.171.1 Constructor & Destructor Documentation . . . 478
12.171.1.1 HelperMod() [1/2] . . . 478
12.171.1.2 HelperMod() [2/2] . . . 478
12.171.2 Field Documentation . . . 478
12.171.2.1 p . . . 478
12.171.2.2 invp . . . 478

12.171.2.3 min . . . 478
12.171.2.4 max . . . 478
12.171.2.5 pow50rem . . . 478
12.172 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference 479
12.172.1 Constructor & Destructor Documentation . . . 479
12.172.1.1 HelperMod() [1/2] . . . 479
12.172.1.2 HelperMod() [2/2] . . . 479
12.172.2 Field Documentation . . . 479
12.172.2.1 p . . . 479
12.173 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference . 479
12.173.1 Constructor & Destructor Documentation . . . 479
12.173.1.1 HelperMod() [1/2] . . . 479
12.173.1.2 HelperMod() [2/2] . . . 479
12.173.2 Field Documentation . . . 480
12.173.2.1 p . . . 480
12.174 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference . 480
12.174.1 Constructor & Destructor Documentation . . . 480
12.174.1.1 HelperMod() [1/2] . . . 480
12.174.1.2 HelperMod() [2/2] . . . 480
12.174.2 Field Documentation . . . 480
12.174.2.1 p . . . 480
12.174.2.2 invp . . . 480
12.174.2.3 min . . . 480
12.174.2.4 max . . . 480
12.175 Hybrid Struct Reference . . . 481
12.176 Info Struct Reference . . . 481
12.176.1 Constructor & Destructor Documentation . . . 481
12.176.1.1 Info() [1/4] . . . 481
12.176.1.2 Info() [2/4] . . . 481
12.176.1.3 Info() [3/4] . . . 481
12.176.1.4 Info() [4/4] . . . 481
12.176.2 Member Function Documentation . . . 481
12.176.2.1 operator=() [1/2] . . . 481
12.176.2.2 operator=() [2/2] . . . 481
12.176.3 Field Documentation . . . 482
12.176.3.1 size . . . 482
12.176.3.2 perm . . . 482
12.176.3.3 begin . . . 482
12.177 Info Struct Reference . . . 482
12.177.1 Constructor & Destructor Documentation . . . 482
12.177.1.1 Info() [1/4] . . . 482
12.177.1.2 Info() [2/4] . . . 482

12.177.1.3 Info() [3/4] . . . 482
12.177.1.4 Info() [4/4] . . . 482
12.177.2 Member Function Documentation . . . 483
12.177.2.1 operator=() [1/2] . . . 483
12.177.2.2 operator=() [2/2] . . . 483
12.177.3 Field Documentation . . . 483
12.177.3.1 size . . . 483
12.177.3.2 perm . . . 483
12.177.3.3 begin . . . 483
12.178 is_all_same< Args > Struct Template Reference . . . 483
12.179 is_all_same< T, Args... > Struct Template Reference . . . 483
12.179.1 Field Documentation . . . 483
12.179.1.1 value . . . 483
12.180 is_all_same<> Struct Reference . . . 483
12.180.1 Field Documentation . . . 484
12.180.1.1 value . . . 484
12.181 is_simd< T > Struct Template Reference . . . 484
12.181.1 Member Typedef Documentation . . . 484
12.181.1.1 type . . . 484
12.181.2 Field Documentation . . . 484
12.181.2.1 value . . . 484
12.182 isSparseMatrix< Field, M > Struct Template Reference . . . 484
12.183 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference . . . 484
12.184 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference 485
12.185 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference . . . 485
12.186 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference 485
12.187 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference 486
12.188 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference . . . 486
12.189 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference 486
12.190 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference . . . 486
12.191 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference 487
12.192 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference 487
12.193 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference . . . 487
12.194 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference 487
12.195 isSparseMatrixMKLFormat< F, M > Struct Template Reference . . . 488
12.196 isSparseMatrixSimdFormat< F, M > Struct Template Reference . . . 488
12.197 isZOSparseMatrix< F, M > Struct Template Reference . . . 488
12.198 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference . . . 489
12.199 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference 489
12.200 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference . . . 489

12.201 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference 489
12.202 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference . . . 490
12.203 Iterative Struct Reference . . . 490
12.204 array< T >::iterator Class Reference . . . 490
12.204.1 Detailed Description . . . 490
12.205 string::iterator Class Reference . . . 490
12.205.1 Detailed Description . . . 490
12.206 vector< T >::iterator Class Reference . . . 490
12.206.1 Detailed Description . . . 490
12.207 LazyTag Struct Reference . . . 490
12.207.1 Detailed Description . . . 491
12.208 limits< T > Struct Template Reference . . . 491
12.209 limits< char > Struct Reference . . . 491
12.209.1 Member Typedef Documentation . . . 491
12.209.1.1 T . . . 491
12.209.2 Member Function Documentation . . . 491
12.209.2.1 max() . . . 491
12.209.2.2 min() . . . 491
12.209.2.3 digits() . . . 491
12.210 limits< double > Struct Reference . . . 491
12.210.1 Member Typedef Documentation . . . 492
12.210.1.1 T . . . 492
12.210.2 Member Function Documentation . . . 492
12.210.2.1 max() . . . 492
12.210.2.2 min() . . . 492
12.210.2.3 digits() . . . 492
12.211 limits< float > Struct Reference . . . 492
12.211.1 Member Typedef Documentation . . . 492
12.211.1.1 T . . . 492
12.211.2 Member Function Documentation . . . 492
12.211.2.1 max() . . . 492
12.211.2.2 min() . . . 492
12.211.2.3 digits() . . . 493
12.212 limits< Givaro::Integer > Struct Reference . . . 493
12.212.1 Member Typedef Documentation . . . 493
12.212.1.1 T . . . 493
12.212.2 Member Function Documentation . . . 493
12.212.2.1 max() . . . 493
12.212.2.2 min() . . . 493
12.213 limits< int > Struct Reference . . . 493
12.213.1 Member Typedef Documentation . . . 493

12.213.1.1 T . . . 493
12.213.2 Member Function Documentation . . . 494
12.213.2.1 max() . . . 494
12.213.2.2 min() . . . 494
12.213.2.3 digits() . . . 494
12.214 limits< long > Struct Reference . . . 494
12.214.1 Member Typedef Documentation . . . 494
12.214.1.1 T . . . 494
12.214.2 Member Function Documentation . . . 494
12.214.2.1 max() . . . 494
12.214.2.2 min() . . . 494
12.214.2.3 digits() . . . 494
12.215 limits< long long > Struct Reference . . . 494
12.215.1 Member Typedef Documentation . . . 495
12.215.1.1 T . . . 495
12.215.2 Member Function Documentation . . . 495
12.215.2.1 max() . . . 495
12.215.2.2 min() . . . 495
12.215.2.3 digits() . . . 495
12.216 limits< RecInt::rint< K > > Struct Template Reference . . . 495
12.216.1 Member Typedef Documentation . . . 495
12.216.1.1 T . . . 495
12.216.2 Member Function Documentation . . . 495
12.216.2.1 max() . . . 495
12.216.2.2 min() . . . 496
12.217 limits< RecInt::ruint< K > > Struct Template Reference . . . 496
12.217.1 Member Typedef Documentation . . . 496
12.217.1.1 T . . . 496
12.217.2 Member Function Documentation . . . 496
12.217.2.1 max() . . . 496
12.217.2.2 min() . . . 496
12.218 limits< short int > Struct Reference . . . 496
12.218.1 Member Typedef Documentation . . . 497
12.218.1.1 T . . . 497
12.218.2 Member Function Documentation . . . 497
12.218.2.1 max() . . . 497
12.218.2.2 min() . . . 497
12.218.2.3 digits() . . . 497
12.219 limits< signed char > Struct Reference . . . 497
12.219.1 Member Typedef Documentation . . . 497
12.219.1.1 T . . . 497
12.219.2 Member Function Documentation . . . 497

12.219.2.1 max() . . . 497
12.219.2.2 min() . . . 497
12.219.2.3 digits() . . . 497
12.220 limits< unsigned char > Struct Reference . . . 498
12.220.1 Member Typedef Documentation . . . 498
12.220.1.1 T . . . 498
12.220.2 Member Function Documentation . . . 498
12.220.2.1 max() . . . 498
12.220.2.2 min() . . . 498
12.220.2.3 digits() . . . 498
12.221 limits< unsigned int > Struct Reference . . . 498
12.221.1 Member Typedef Documentation . . . 498
12.221.1.1 T . . . 498
12.221.2 Member Function Documentation . . . 499
12.221.2.1 max() . . . 499
12.221.2.2 min() . . . 499
12.221.2.3 digits() . . . 499
12.222 limits< unsigned long > Struct Reference . . . 499
12.222.1 Member Typedef Documentation . . . 499
12.222.1.1 T . . . 499
12.222.2 Member Function Documentation . . . 499
12.222.2.1 max() . . . 499
12.222.2.2 min() . . . 499
12.222.2.3 digits() . . . 499
12.223 limits< unsigned long long > Struct Reference . . . 499
12.223.1 Member Typedef Documentation . . . 500
12.223.1.1 T . . . 500
12.223.2 Member Function Documentation . . . 500
12.223.2.1 max() . . . 500
12.223.2.2 min() . . . 500
12.223.2.3 digits() . . . 500
12.224 limits< unsigned short int > Struct Reference . . . 500
12.224.1 Member Typedef Documentation . . . 500
12.224.1.1 T . . . 500
12.224.2 Member Function Documentation . . . 500
12.224.2.1 max() . . . 500
12.224.2.2 min() . . . 500
12.224.2.3 digits() . . . 501
12.225 MachineFloatTag Struct Reference . . . 501
12.225.1 Detailed Description . . . 501
12.226 MachineIntTag Struct Reference . . . 501
12.226.1 Detailed Description . . . 501

12.227 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference 501
12.227.1 Member Typedef Documentation 502
12.227.1.1 Self_t 502
12.227.1.2 DelayedField_t 502
12.227.1.3 DelayedField 502
12.227.1.4 DFElt 502
12.227.2 Constructor & Destructor Documentation 502
12.227.2.1 MMHelper() [1/5] 502
12.227.2.2 MMHelper() [2/5] 503
12.227.2.3 MMHelper() [3/5] 503
12.227.2.4 MMHelper() [4/5] 503
12.227.2.5 MMHelper() [5/5] 503
12.227.3 Member Function Documentation 503
12.227.3.1 initC() 503
12.227.3.2 initA() 503
12.227.3.3 initB() 503
12.227.3.4 initOut() 503
12.227.3.5 MaxDelayedDim() 504
12.227.3.6 Aunfit() 504
12.227.3.7 Bunfit() 504
12.227.3.8 setOutBounds() 504
12.227.3.9 checkA() 504
12.227.3.10 checkB() 504
12.227.3.11 checkOut() [1/2] 504
12.227.3.12 checkOut() [2/2] 504
12.227.4 Friends And Related Symbol Documentation 505
12.227.4.1 operator<< 505
12.227.5 Field Documentation 505
12.227.5.1 recLevel 505
12.227.5.2 FieldMin 505
12.227.5.3 FieldMax 505
12.227.5.4 Amin 505
12.227.5.5 Amax 505
12.227.5.6 Bmin 505
12.227.5.7 Bmax 505
12.227.5.8 Cmin 505
12.227.5.9 Cmax 506
12.227.5.10 Outmin 506
12.227.5.11 Outmax 506
12.227.5.12 MaxStorableValue 506
12.227.5.13 delayedField 506
12.227.5.14 parseq 506

12.228 ModeTraits< Field > Struct Template Reference 506
12.228.1 Detailed Description 506
12.228.2 Member Typedef Documentation 506
12.228.2.1 value 506
12.229 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference 507
12.229.1 Member Typedef Documentation 507
12.229.1.1 value [1/2] 507
12.229.1.2 value [2/2] 507
12.230 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference 507
12.230.1 Member Typedef Documentation 507
12.230.1.1 value [1/2] 507
12.230.1.2 value [2/2] 507
12.231 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference 507
12.231.1 Member Typedef Documentation 508
12.231.1.1 value [1/2] 508
12.231.1.2 value [2/2] 508
12.232 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference 508
12.232.1 Member Typedef Documentation 508
12.232.1.1 value [1/2] 508
12.232.1.2 value [2/2] 508
12.233 ModeTraits< Givaro::Modular< int64_t, uint64_t > > Struct Reference 508
12.233.1 Member Typedef Documentation 508
12.233.1.1 value [1/2] 508
12.233.1.2 value [2/2] 508
12.234 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference 509
12.234.1 Member Typedef Documentation 509
12.234.1.1 value [1/2] 509
12.234.1.2 value [2/2] 509
12.235 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference . . 509
12.235.1 Member Typedef Documentation 509
12.235.1.1 value [1/2] 509
12.235.1.2 value [2/2] 509
12.236 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference 509
12.236.1 Member Typedef Documentation 510
12.236.1.1 value [1/2] 510
12.236.1.2 value [2/2] 510
12.237 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference 510
12.237.1 Member Typedef Documentation 510
12.237.1.1 value [1/2] 510
12.237.1.2 value [2/2] 510
12.238 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference 510
12.238.1 Member Typedef Documentation 510

12.238.1.1 value [1/2] . . . 510
12.238.1.2 value [2/2] . . . 510
12.239 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference . . . 511
12.239.1 Member Typedef Documentation . . . 511
12.239.1.1 value [1/2] . . . 511
12.239.1.2 value [2/2] . . . 511
12.240 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference . . . 511
12.240.1 Member Typedef Documentation . . . 511
12.240.1.1 value [1/2] . . . 511
12.240.1.2 value [2/2] . . . 511
12.241 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference . . . 511
12.241.1 Member Typedef Documentation . . . 512
12.241.1.1 value [1/2] . . . 512
12.241.1.2 value [2/2] . . . 512
12.242 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference . . . 512
12.242.1 Member Typedef Documentation . . . 512
12.242.1.1 value [1/2] . . . 512
12.242.1.2 value [2/2] . . . 512
12.243 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference . . . 512
12.243.1 Member Typedef Documentation . . . 512
12.243.1.1 value [1/2] . . . 512
12.243.1.2 value [2/2] . . . 512
12.244 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference . . . 513
12.244.1 Member Typedef Documentation . . . 513
12.244.1.1 value [1/2] . . . 513
12.244.1.2 value [2/2] . . . 513
12.245 ModeTraits< Givaro::ZRing< double > > Struct Reference . . . 513
12.245.1 Member Typedef Documentation . . . 513
12.245.1.1 value [1/2] . . . 513
12.245.1.2 value [2/2] . . . 513
12.246 ModeTraits< Givaro::ZRing< float > > Struct Reference . . . 513
12.246.1 Member Typedef Documentation . . . 513
12.246.1.1 value [1/2] . . . 513
12.246.1.2 value [2/2] . . . 514
12.247 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference . . . 514
12.247.1 Member Typedef Documentation . . . 514
12.247.1.1 value [1/2] . . . 514
12.247.1.2 value [2/2] . . . 514
12.248 ModularBalanced< T > Class Template Reference . . . 514
12.249 ModularBalanced< T > Class Template Reference . . . 514
12.250 ModularTag Struct Reference . . . 514
12.250.1 Detailed Description . . . 514

12.251 Montgomery< T > Class Template Reference . . . 514
12.252 need_field_characteristic< Field > Struct Template Reference . . . 515
12.252.1 Field Documentation . . . 515
12.252.1.1 value . . . 515
12.253 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference . . . 515
12.253.1 Field Documentation . . . 515
12.253.1.1 value [1/2] . . . 515
12.253.1.2 value [2/2] . . . 515
12.254 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference . . . 515
12.254.1 Field Documentation . . . 515
12.254.1.1 value [1/2] . . . 515
12.254.1.2 value [2/2] . . . 515
12.255 NoSimd< T > Struct Template Reference . . . 516
12.255.1 Member Typedef Documentation . . . 516
12.255.1.1 vect_t . . . 516
12.255.1.2 scalar_t . . . 516
12.255.1.3 aligned_allocator . . . 516
12.255.1.4 aligned_vector . . . 516
12.255.1.5 is_same_element . . . 516
12.255.2 Member Function Documentation . . . 516
12.255.2.1 type_string() . . . 516
12.255.2.2 valid() . . . 517
12.255.2.3 compliant() . . . 517
12.255.3 Field Documentation . . . 517
12.255.3.1 vect_size . . . 517
12.255.3.2 alignment . . . 517
12.256 Parallel< C, P > Struct Template Reference . . . 517
12.256.1 Member Typedef Documentation . . . 517
12.256.1.1 Cut . . . 517
12.256.1.2 Param . . . 517
12.256.2 Constructor & Destructor Documentation . . . 518
12.256.2.1 Parallel() . . . 518
12.256.3 Member Function Documentation . . . 518
12.256.3.1 numthreads() . . . 518
12.256.3.2 set_numthreads() . . . 518
12.256.4 Friends And Related Symbol Documentation . . . 518
12.256.4.1 operator<< . . . 518
12.257 RNSInteger< RNS >::RandIter Class Reference . . . 518
12.257.1 Constructor & Destructor Documentation . . . 519
12.257.1.1 RandIter() . . . 519
12.257.2 Member Function Documentation . . . 519
12.257.2.1 random() [1/2] . . . 519

12.257.2.2 random() [2/2] . . . 519
12.257.2.3 operator()() [1/2] . . . 519
12.257.2.4 operator()() [2/2] . . . 519
12.257.2.5 ring() . . . 519
12.258 RNSIntegerMod< RNS >::RandIter Class Reference . . . 519
12.258.1 Constructor & Destructor Documentation . . . 520
12.258.1.1 RandIter() . . . 520
12.258.2 Member Function Documentation . . . 520
12.258.2.1 random() [1/2] . . . 520
12.258.2.2 random() [2/2] . . . 520
12.258.2.3 operator()() [1/2] . . . 520
12.258.2.4 operator()() [2/2] . . . 520
12.258.2.5 ring() . . . 520
12.259 readMyMachineType< Field, T > Struct Template Reference . . . 520
12.259.1 Member Typedef Documentation . . . 521
12.259.1.1 Element . . . 521
12.259.1.2 Element_ptr . . . 521
12.259.2 Member Function Documentation . . . 521
12.259.2.1 operator()() . . . 521
12.260 readMyMachineType< Field, mpz_t > Struct Template Reference . . . 521
12.260.1 Member Typedef Documentation . . . 521
12.260.1.1 Element [1/2] . . . 521
12.260.1.2 Element_ptr [1/2] . . . 522
12.260.1.3 Element [2/2] . . . 522
12.260.1.4 Element_ptr [2/2] . . . 522
12.260.2 Member Function Documentation . . . 522
12.260.2.1 operator()() [1/2] . . . 522
12.260.2.2 operator()() [2/2] . . . 522
12.261 Recursive Struct Reference . . . 522
12.262 Recursive Struct Reference . . . 522
12.263 array< T >::reverse_iterator Class Reference . . . 522
12.263.1 Detailed Description . . . 523
12.264 string::reverse_iterator Class Reference . . . 523
12.264.1 Detailed Description . . . 523
12.265 vector< T >::reverse_iterator Class Reference . . . 523
12.265.1 Detailed Description . . . 523
12.266 rint< K > Class Template Reference . . . 523
12.267 rns_double Struct Reference . . . 523
12.267.1 Member Typedef Documentation . . . 524
12.267.1.1 integer . . . 524
12.267.1.2 ModField . . . 524
12.267.1.3 BasisElement . . . 524

12.267.1.4 Element . . . 524
12.267.1.5 Element_ptr . . . 524
12.267.1.6 ConstElement_ptr . . . 524
12.267.2 Constructor & Destructor Documentation . . . 525
12.267.2.1 rns_double() [1/4] . . . 525
12.267.2.2 rns_double() [2/4] . . . 525
12.267.2.3 rns_double() [3/4] . . . 525
12.267.2.4 rns_double() [4/4] . . . 525
12.267.3 Member Function Documentation . . . 525
12.267.3.1 precompute_cst() . . . 525
12.267.3.2 init() [1/3] . . . 525
12.267.3.3 init() [2/3] . . . 525
12.267.3.4 init_transpose() . . . 526
12.267.3.5 convert() [1/2] . . . 526
12.267.3.6 convert_transpose() . . . 526
12.267.3.7 reduce() . . . 526
12.267.3.8 init() [3/3] . . . 526
12.267.3.9 convert() [2/2] . . . 527
12.267.4 Field Documentation . . . 527
12.267.4.1 _basis . . . 527
12.267.4.2 _basisMax . . . 527
12.267.4.3 _negbasis . . . 527
12.267.4.4 _invbasis . . . 527
12.267.4.5 _field_rns . . . 527
12.267.4.6 _M . . . 527
12.267.4.7 _Mi . . . 527
12.267.4.8 _MMi . . . 527
12.267.4.9 _crt_in . . . 527
12.267.4.10 _crt_out . . . 527
12.267.4.11 _size . . . 528
12.267.4.12 _pbits . . . 528
12.267.4.13 _ldm . . . 528
12.267.4.14 _mi_sum . . . 528
12.268 rns_double_elt Struct Reference . . . 528
12.268.1 Constructor & Destructor Documentation . . . 528
12.268.1.1 rns_double_elt() [1/3] . . . 528
12.268.1.2 ~rns_double_elt() . . . 528
12.268.1.3 rns_double_elt() [2/3] . . . 529
12.268.1.4 rns_double_elt() [3/3] . . . 529
12.268.2 Member Function Documentation . . . 529
12.268.2.1 operator&() [1/2] . . . 529
12.268.2.2 operator&() [2/2] . . . 529

12.268.3 Field Documentation . . . 529
12.268.3.1 _ptr . . . 529
12.268.3.2 _stride . . . 529
12.268.3.3 _alloc . . . 529
12.269 rns_double_elt_cstptr Struct Reference . . . 529
12.269.1 Constructor & Destructor Documentation . . . 530
12.269.1.1 rns_double_elt_cstptr() [1/5] . . . 530
12.269.1.2 rns_double_elt_cstptr() [2/5] . . . 530
12.269.1.3 rns_double_elt_cstptr() [3/5] . . . 530
12.269.1.4 rns_double_elt_cstptr() [4/5] . . . 530
12.269.1.5 rns_double_elt_cstptr() [5/5] . . . 530
12.269.2 Member Function Documentation . . . 530
12.269.2.1 operator&() [1/2] . . . 530
12.269.2.2 operator∗() . . . 530
12.269.2.3 operator[]() [1/2] . . . 531
12.269.2.4 operator[]() [2/2] . . . 531
12.269.2.5 operator++() . . . 531
12.269.2.6 operator--() . . . 531
12.269.2.7 operator+() . . . 531
12.269.2.8 operator-() . . . 531
12.269.2.9 operator+=() . . . 531
12.269.2.10 operator-=() . . . 531
12.269.2.11 operator=() . . . 531
12.269.2.12 operator<() . . . 531
12.269.2.13 operator"!=() . . . 531
12.269.2.14 operator&() [2/2] . . . 531
12.269.3 Field Documentation . . . 532
12.269.3.1 other . . . 532
12.269.3.2 _ptr . . . 532
12.269.3.3 _stride . . . 532
12.269.3.4 _alloc . . . 532
12.270 rns_double_elt_ptr Struct Reference . . . 532
12.270.1 Constructor & Destructor Documentation . . . 533
12.270.1.1 rns_double_elt_ptr() [1/5] . . . 533
12.270.1.2 rns_double_elt_ptr() [2/5] . . . 533
12.270.1.3 rns_double_elt_ptr() [3/5] . . . 533
12.270.1.4 rns_double_elt_ptr() [4/5] . . . 533
12.270.1.5 rns_double_elt_ptr() [5/5] . . . 533
12.270.2 Member Function Documentation . . . 533
12.270.2.1 operator&() [1/2] . . . 533
12.270.2.2 operator∗() . . . 533
12.270.2.3 operator[]() [1/2] . . . 533

12.270.2.4 operator[]() [2/2] . . . 533
12.270.2.5 operator++() . . . 533
12.270.2.6 operator--() . . . 533
12.270.2.7 operator+() . . . 534
12.270.2.8 operator-() . . . 534
12.270.2.9 operator+=() . . . 534
12.270.2.10 operator-=() . . . 534
12.270.2.11 operator=() . . . 534
12.270.2.12 operator< () . . . 534
12.270.2.13 operator"!=() . . . 534
12.270.2.14 operator&() [2/2] . . . 534
12.270.3 Field Documentation . . . 534
12.270.3.1 other . . . 534
12.270.3.2 _ptr . . . 534
12.270.3.3 _stride . . . 534
12.270.3.4 _alloc . . . 534
12.271 rns_double_extended Struct Reference . . . 535
12.271.1 Member Typedef Documentation . . . 535
12.271.1.1 integer . . . 535
12.271.1.2 ModField . . . 535
12.271.1.3 BasisElement . . . 536
12.271.1.4 Element . . . 536
12.271.1.5 Element_ptr . . . 536
12.271.1.6 ConstElement_ptr . . . 536
12.271.2 Constructor & Destructor Documentation . . . 536
12.271.2.1 rns_double_extended() [1/3] . . . 536
12.271.2.2 rns_double_extended() [2/3] . . . 536
12.271.2.3 rns_double_extended() [3/3] . . . 536
12.271.3 Member Function Documentation . . . 536
12.271.3.1 precompute_cst() . . . 536
12.271.3.2 init() [1/3] . . . 536
12.271.3.3 init() [2/3] . . . 537
12.271.3.4 convert() [1/2] . . . 537
12.271.3.5 init() [3/3] . . . 537
12.271.3.6 convert() [2/2] . . . 537
12.271.3.7 reduce() . . . 537
12.271.4 Field Documentation . . . 537
12.271.4.1 _basis . . . 537
12.271.4.2 _basisMax . . . 537
12.271.4.3 _negbasis . . . 537
12.271.4.4 _invbasis . . . 538
12.271.4.5 _field_rns . . . 538

12.271.4.6 _M . . . 538
12.271.4.7 _Mi . . . 538
12.271.4.8 _MMi . . . 538
12.271.4.9 _crt_in . . . 538
12.271.4.10 _crt_out . . . 538
12.271.4.11 _size . . . 538
12.271.4.12 _pbits . . . 538
12.271.4.13 _ldm . . . 538
12.272 RNSElementTag Struct Reference . . . 538
12.272.1 Detailed Description . . . 538
12.273 RNSInteger< RNS > Class Template Reference . . . 538
12.273.1 Member Typedef Documentation . . . 539
12.273.1.1 BasisElement . . . 539
12.273.1.2 integer . . . 539
12.273.1.3 Element . . . 540
12.273.1.4 Element_ptr . . . 540
12.273.1.5 ConstElement_ptr . . . 540
12.273.2 Constructor & Destructor Documentation . . . 540
12.273.2.1 RNSInteger() [1/2] . . . 540
12.273.2.2 RNSInteger() [2/2] . . . 540
12.273.3 Member Function Documentation . . . 540
12.273.3.1 rns() . . . 540
12.273.3.2 size() . . . 540
12.273.3.3 isOne() . . . 540
12.273.3.4 isMOne() . . . 540
12.273.3.5 isZero() . . . 540
12.273.3.6 characteristic() . . . 541
12.273.3.7 cardinality() . . . 541
12.273.3.8 init() [1/2] . . . 541
12.273.3.9 init() [2/2] . . . 541
12.273.3.10 reduce() [1/2] . . . 541
12.273.3.11 reduce() [2/2] . . . 541
12.273.3.12 convert() . . . 541
12.273.3.13 assign() . . . 541
12.273.3.14 write() [1/2] . . . 541
12.273.3.15 write() [2/2] . . . 542
12.273.4 Field Documentation . . . 542
12.273.4.1 _rns . . . 542
12.273.4.2 one . . . 542
12.273.4.3 mOne . . . 542
12.273.4.4 zero . . . 542
12.274 RNSIntegerMod< RNS > Class Template Reference . . . 542

12.274.1 Member Typedef Documentation . . . 543
12.274.1.1 Element . . . 543
12.274.1.2 Element_ptr . . . 543
12.274.1.3 ConstElement_ptr . . . 544
12.274.1.4 BasisElement . . . 544
12.274.1.5 ModField . . . 544
12.274.1.6 integer . . . 544
12.274.2 Constructor & Destructor Documentation . . . 544
12.274.2.1 RNSIntegerMod() . . . 544
12.274.3 Member Function Documentation . . . 544
12.274.3.1 rns() . . . 544
12.274.3.2 delayed() . . . 544
12.274.3.3 size() . . . 544
12.274.3.4 isOne() . . . 544
12.274.3.5 isMOne() . . . 544
12.274.3.6 isZero() . . . 545
12.274.3.7 characteristic() [1/2] . . . 545
12.274.3.8 characteristic() [2/2] . . . 545
12.274.3.9 cardinality() [1/2] . . . 545
12.274.3.10 cardinality() [2/2] . . . 545
12.274.3.11 minElement() . . . 545
12.274.3.12 maxElement() . . . 545
12.274.3.13 init() [1/3] . . . 545
12.274.3.14 init() [2/3] . . . 545
12.274.3.15 reduce() [1/2] . . . 545
12.274.3.16 reduce() [2/2] . . . 546
12.274.3.17 init() [3/3] . . . 546
12.274.3.18 convert() . . . 546
12.274.3.19 assign() . . . 546
12.274.3.20 add() . . . 546
12.274.3.21 sub() . . . 546
12.274.3.22 neg() . . . 546
12.274.3.23 mul() . . . 546
12.274.3.24 axpyin() . . . 547
12.274.3.25 inv() . . . 547
12.274.3.26 areEqual() . . . 547
12.274.3.27 write() [1/2] . . . 547
12.274.3.28 write() [2/2] . . . 547
12.274.3.29 reduce_modp() [1/2] . . . 547
12.274.3.30 write_matrix() . . . 547
12.274.3.31 write_matrix_long() . . . 547
12.274.3.32 reduce_modp() [2/2] . . . 548

12.274.3.33 reduce_modp_rnsmajor() . . . 548
12.274.4 Field Documentation . . . 548
12.274.4.1 _p . . . 548
12.274.4.2 _Mi_modp_rns . . . 548
12.274.4.3 _iM_modp_rns . . . 548
12.274.4.4 _rns . . . 548
12.274.4.5 _F . . . 548
12.274.4.6 _RNSdelayed . . . 548
12.274.4.7 one . . . 548
12.274.4.8 mOne . . . 549
12.274.4.9 zero . . . 549
12.275 rnsRandIter< RNS > Class Template Reference . . . 549
12.275.1 Constructor & Destructor Documentation . . . 549
12.275.1.1 rnsRandIter() . . . 549
12.275.2 Member Function Documentation . . . 549
12.275.2.1 random() [1/2] . . . 549
12.275.2.2 operator()() [1/2] . . . 550
12.275.2.3 operator()() [2/2] . . . 550
12.275.2.4 random() [2/2] . . . 550
12.275.2.5 ring() . . . 550
12.276 Row Struct Reference . . . 550
12.277 ruint< K > Class Template Reference . . . 550
12.278 ScalFunctions< Element > Struct Template Reference . . . 550
12.278.1 Member Typedef Documentation . . . 551
12.278.1.1 vectElt . . . 551
12.278.2 Member Function Documentation . . . 551
12.278.2.1 genInputs() . . . 551
12.278.2.2 genInputsWithZero() . . . 551
12.278.2.3 zero() . . . 551
12.278.2.4 vand() . . . 552
12.278.2.5 vor() . . . 552
12.278.2.6 vxor() . . . 552
12.278.2.7 vandnot() . . . 552
12.278.2.8 add() . . . 552
12.278.2.9 addin() . . . 552
12.278.2.10 sub() . . . 552
12.278.2.11 subin() . . . 552
12.278.2.12 mul() . . . 552
12.278.2.13 mulin() . . . 553
12.278.2.14 div() . . . 553
12.278.2.15 fmadd() . . . 553
12.278.2.16 fmaddin() . . . 553

12.278.2.17 fmsub() . . . 553
12.278.2.18 fmsubin() . . . 553
12.278.2.19 fnmadd() . . . 553
12.278.2.20 fnmaddin() . . . 554
12.278.2.21 lesser() . . . 554
12.278.2.22 lesser_eq() . . . 554
12.278.2.23 greater() . . . 554
12.278.2.24 greater_eq() . . . 554
12.278.2.25 eq() . . . 554
12.278.2.26 unpacklo() . . . 554
12.278.2.27 unpackhi() . . . 554
12.278.2.28 unpacklohi() . . . 555
12.278.2.29 pack_even() . . . 555
12.278.2.30 pack_odd() . . . 555
12.278.2.31 pack() . . . 555
12.278.2.32 blend() . . . 555
12.279 ScalFunctionsBase< Element, Enable > Struct Template Reference . . . 555
12.280 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference . . . 556
12.280.1 Member Function Documentation . . . 556
12.280.1.1 get_default_random_generator() . . . 556
12.280.1.2 ceil() . . . 556
12.280.1.3 floor() . . . 556
12.280.1.4 round() . . . 556
12.280.1.5 blendv() . . . 556
12.280.1.6 fma() . . . 557
12.280.2 Field Documentation . . . 557
12.280.2.1 _zero . . . 557
12.280.2.2 cmp_true . . . 557
12.280.2.3 cmp_false . . . 557
12.281 ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference . . . 557
12.281.1 Member Function Documentation . . . 558
12.281.1.1 get_default_random_generator() . . . 558
12.281.1.2 round() . . . 558
12.281.1.3 fma() . . . 558
12.281.1.4 mullo() . . . 558
12.281.1.5 mulhi() . . . 558
12.281.1.6 mulx() . . . 558
12.281.1.7 fmaddx() . . . 558
12.281.1.8 fmaddxin() . . . 558
12.281.1.9 fmsubx() . . . 559
12.281.1.10 fmsubxin() . . . 559

12.281.1.11 fnmaddx() . . . 559
12.281.1.12 fnmaddxin() . . . 559
12.281.1.13 sra() . . . 559
12.281.1.14 srl() . . . 559
12.281.1.15 sll() . . . 559
12.281.2 Field Documentation . . . 559
12.281.2.1 _zero . . . 559
12.281.2.2 cmp_true . . . 560
12.281.2.3 cmp_false . . . 560
12.282 Sequential Struct Reference . . . 560
12.282.1 Constructor & Destructor Documentation . . . 560
12.282.1.1 Sequential() [1/3] . . . 560
12.282.1.2 Sequential() [2/3] . . . 560
12.282.1.3 Sequential() [3/3] . . . 560
12.282.2 Member Function Documentation . . . 560
12.282.2.1 numthreads() . . . 560
12.282.3 Friends And Related Symbol Documentation . . . 560
12.282.3.1 operator<< . . . 560
12.283 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . 561
12.284 Simd128_impl< true, false, true, 4 > Struct Reference . . . 561
12.285 Simd128_impl< true, false, true, 8 > Struct Reference . . . 561
12.286 Simd128_impl< true, true, false, 2 > Struct Reference . . . 561
12.286.1 Member Typedef Documentation . . . 563
12.286.1.1 scalar_t . . . 563
12.286.1.2 aligned_allocator . . . 563
12.286.1.3 aligned_vector . . . 563
12.286.1.4 is_same_element . . . 563
12.286.1.5 vect_t . . . 563
12.286.2 Member Function Documentation . . . 563
12.286.2.1 type_string() . . . 563
12.286.2.2 set1() [1/2] . . . 563
12.286.2.3 set() [1/2] . . . 564
12.286.2.4 gather() [1/2] . . . 564
12.286.2.5 load() [1/2] . . . 564
12.286.2.6 loadu() [1/2] . . . 564
12.286.2.7 store() [1/2] . . . 564
12.286.2.8 storeu() [1/2] . . . 564
12.286.2.9 stream() [1/2] . . . 564
12.286.2.10 sra() . . . 564
12.286.2.11 greater() . . . 564
12.286.2.12 lesser() . . . 565
12.286.2.13 greater_eq() . . . 565

12.286.2.14 lesser_eq() . . . 565
12.286.2.15 mulhi() . . . 565
12.286.2.16 mulx() . . . 565
12.286.2.17 fmaddx() . . . 565
12.286.2.18 fmaddxin() . . . 565
12.286.2.19 fnmaddx() . . . 565
12.286.2.20 fnmaddxin() . . . 565
12.286.2.21 fmsubx() . . . 566
12.286.2.22 fmsubxin() . . . 566
12.286.2.23 hadd_to_scal() . . . 566
12.286.2.24 valid() . . . 566
12.286.2.25 compliant() . . . 566
12.286.2.26 set1() [2/2] . . . 566
12.286.2.27 set() [2/2] . . . 566
12.286.2.28 gather() [2/2] . . . 566
12.286.2.29 load() [2/2] . . . 566
12.286.2.30 loadu() [2/2] . . . 567
12.286.2.31 store() [2/2] . . . 567
12.286.2.32 storeu() [2/2] . . . 567
12.286.2.33 stream() [2/2] . . . 567
12.286.2.34 sll() . . . 567
12.286.2.35 srl() . . . 567
12.286.2.36 shuffle() . . . 567
12.286.2.37 unpacklo_intrinsic() . . . 567
12.286.2.38 unpackhi_intrinsic() . . . 567
12.286.2.39 unpacklo() . . . 567
12.286.2.40 unpackhi() . . . 568
12.286.2.41 unpacklohi() . . . 568
12.286.2.42 pack_even() . . . 568
12.286.2.43 pack_odd() . . . 568
12.286.2.44 pack() . . . 568
12.286.2.45 transpose() . . . 568
12.286.2.46 blend() . . . 568
12.286.2.47 add() . . . 568
12.286.2.48 addin() . . . 569
12.286.2.49 sub() . . . 569
12.286.2.50 subin() . . . 569
12.286.2.51 mullo() . . . 569
12.286.2.52 mul() . . . 569
12.286.2.53 fmadd() . . . 569
12.286.2.54 fmaddin() . . . 569
12.286.2.55 fnmadd() . . . 569

12.286.2.56 fnmaddin() . . . 569
12.286.2.57 fmsub() . . . 570
12.286.2.58 fmsubin() . . . 570
12.286.2.59 eq() . . . 570
12.286.2.60 round() . . . 570
12.286.2.61 mod() . . . 570
12.286.2.62 zero() . . . 570
12.286.2.63 sll128() . . . 570
12.286.2.64 srl128() . . . 570
12.286.2.65 vand() . . . 570
12.286.2.66 vor() . . . 571
12.286.2.67 vxor() . . . 571
12.286.2.68 vandnot() . . . 571
12.286.3 Field Documentation . . . 571
12.286.3.1 vect_size . . . 571
12.286.3.2 alignment . . . 571
12.287 Simd128_impl< true, true, false, 4 > Struct Reference . . . 571
12.287.1 Member Typedef Documentation . . . 573
12.287.1.1 scalar_t . . . 573
12.287.1.2 aligned_allocator . . . 573
12.287.1.3 aligned_vector . . . 573
12.287.1.4 is_same_element . . . 573
12.287.1.5 vect_t . . . 573
12.287.2 Member Function Documentation . . . 574
12.287.2.1 type_string() . . . 574
12.287.2.2 set1() [1/2] . . . 574
12.287.2.3 set() [1/2] . . . 574
12.287.2.4 gather() [1/2] . . . 574
12.287.2.5 load() [1/2] . . . 574
12.287.2.6 loadu() [1/2] . . . 574
12.287.2.7 store() [1/2] . . . 574
12.287.2.8 storeu() [1/2] . . . 574
12.287.2.9 stream() [1/2] . . . 574
12.287.2.10 sra() . . . 574
12.287.2.11 greater() . . . 575
12.287.2.12 lesser() . . . 575
12.287.2.13 greater_eq() . . . 575
12.287.2.14 lesser_eq() . . . 575
12.287.2.15 mulhi() . . . 575
12.287.2.16 mulx() . . . 575
12.287.2.17 fmaddx() . . . 575
12.287.2.18 fmaddxin() . . . 575

12.287.2.19 fnmaddx() . . . 575
12.287.2.20 fnmaddxin() . . . 576
12.287.2.21 fmsubx() . . . 576
12.287.2.22 fmsubxin() . . . 576
12.287.2.23 hadd_to_scal() . . . 576
12.287.2.24 valid() . . . 576
12.287.2.25 compliant() . . . 576
12.287.2.26 set1() [2/2] . . . 576
12.287.2.27 set() [2/2] . . . 576
12.287.2.28 gather() [2/2] . . . 576
12.287.2.29 load() [2/2] . . . 577
12.287.2.30 loadu() [2/2] . . . 577
12.287.2.31 store() [2/2] . . . 577
12.287.2.32 storeu() [2/2] . . . 577
12.287.2.33 stream() [2/2] . . . 577
12.287.2.34 sll() . . . 577
12.287.2.35 srl() . . . 577
12.287.2.36 shuffle() . . . 577
12.287.2.37 unpacklo_intrinsic() . . . 577
12.287.2.38 unpackhi_intrinsic() . . . 577
12.287.2.39 unpacklo() . . . 578
12.287.2.40 unpackhi() . . . 578
12.287.2.41 unpacklohi() . . . 578
12.287.2.42 pack_even() . . . 578
12.287.2.43 pack_odd() . . . 578
12.287.2.44 pack() . . . 578
12.287.2.45 transpose() . . . 578
12.287.2.46 blend() . . . 578
12.287.2.47 add() . . . 578
12.287.2.48 addin() . . . 579
12.287.2.49 sub() . . . 579
12.287.2.50 subin() . . . 579
12.287.2.51 mullo() . . . 579
12.287.2.52 mul() . . . 579
12.287.2.53 fmadd() . . . 579
12.287.2.54 fmaddin() . . . 579
12.287.2.55 fnmadd() . . . 579
12.287.2.56 fnmaddin() . . . 579
12.287.2.57 fmsub() . . . 580
12.287.2.58 fmsubin() . . . 580
12.287.2.59 eq() . . . 580
12.287.2.60 round() . . . 580

12.287.2.61 mod() . . . 580
12.287.2.62 zero() . . . 580
12.287.2.63 sll128() . . . 580
12.287.2.64 srl128() . . . 580
12.287.2.65 vand() . . . 580
12.287.2.66 vor() . . . 581
12.287.2.67 vxor() . . . 581
12.287.2.68 vandnot() . . . 581
12.287.3 Field Documentation . . . 581
12.287.3.1 vect_size . . . 581
12.287.3.2 alignment . . . 581
12.288 Simd128_impl< true, true, false, 8 > Struct Reference . . . 581
12.288.1 Member Typedef Documentation . . . 583
12.288.1.1 scalar_t . . . 583
12.288.1.2 aligned_allocator . . . 583
12.288.1.3 aligned_vector . . . 583
12.288.1.4 is_same_element . . . 583
12.288.1.5 vect_t . . . 584
12.288.2 Member Function Documentation . . . 584
12.288.2.1 type_string() . . . 584
12.288.2.2 set1() [1/2] . . . 584
12.288.2.3 set() [1/2] . . . 584
12.288.2.4 gather() [1/2] . . . 584
12.288.2.5 load() [1/2] . . . 584
12.288.2.6 loadu() [1/2] . . . 584
12.288.2.7 store() [1/2] . . . 584
12.288.2.8 storeu() [1/2] . . . 584
12.288.2.9 stream() [1/2] . . . 584
12.288.2.10 sra() . . . 584
12.288.2.11 greater() . . . 585
12.288.2.12 lesser() . . . 585
12.288.2.13 greater_eq() . . . 585
12.288.2.14 lesser_eq() . . . 585
12.288.2.15 mullo() . . . 585
12.288.2.16 mulhi() . . . 585
12.288.2.17 mulx() . . . 585
12.288.2.18 fmaddx() . . . 585
12.288.2.19 fmaddxin() . . . 585
12.288.2.20 fnmaddx() . . . 586
12.288.2.21 fnmaddxin() . . . 586
12.288.2.22 fmsubx() . . . 586
12.288.2.23 fmsubxin() . . . 586

12.288.2.24 hadd_to_scal() . . . 586
12.288.2.25 valid() . . . 586
12.288.2.26 compliant() . . . 586
12.288.2.27 set1() [2/2] . . . 586
12.288.2.28 set() [2/2] . . . 586
12.288.2.29 gather() [2/2] . . . 587
12.288.2.30 get() . . . 587
12.288.2.31 load() [2/2] . . . 587
12.288.2.32 loadu() [2/2] . . . 587
12.288.2.33 store() [2/2] . . . 587
12.288.2.34 storeu() [2/2] . . . 587
12.288.2.35 stream() [2/2] . . . 587
12.288.2.36 sll() . . . 587
12.288.2.37 srl() . . . 587
12.288.2.38 shuffle() . . . 587
12.288.2.39 unpacklo_intrinsic() . . . 588
12.288.2.40 unpackhi_intrinsic() . . . 588
12.288.2.41 unpacklo() . . . 588
12.288.2.42 unpackhi() . . . 588
12.288.2.43 unpacklohi() . . . 588
12.288.2.44 pack_even() . . . 588
12.288.2.45 pack_odd() . . . 588
12.288.2.46 pack() . . . 588
12.288.2.47 transpose() . . . 588
12.288.2.48 blend() . . . 589
12.288.2.49 add() . . . 589
12.288.2.50 addin() . . . 589
12.288.2.51 sub() . . . 589
12.288.2.52 subin() . . . 589
12.288.2.53 mul() . . . 589
12.288.2.54 fmadd() . . . 589
12.288.2.55 fmaddin() . . . 589
12.288.2.56 fnmadd() . . . 589
12.288.2.57 fnmaddin() . . . 590
12.288.2.58 fmsub() . . . 590
12.288.2.59 fmsubin() . . . 590
12.288.2.60 eq() . . . 590
12.288.2.61 round() . . . 590
12.288.2.62 mask_high() . . . 590
12.288.2.63 mulhi_fast() . . . 590
12.288.2.64 mod() . . . 590
12.288.2.65 signbits() . . . 590

12.288.2.66 zero() . . . 591
12.288.2.67 sll128() . . . 591
12.288.2.68 srl128() . . . 591
12.288.2.69 vand() . . . 591
12.288.2.70 vor() . . . 591
12.288.2.71 vxor() . . . 591
12.288.2.72 vandnot() . . . 591
12.288.3 Field Documentation . . . 591
12.288.3.1 vect_size . . . 591
12.288.3.2 alignment . . . 591
12.289 Simd128_impl< true, true, true, 2 > Struct Reference . . . 591
12.289.1 Member Typedef Documentation . . . 593
12.289.1.1 vect_t . . . 593
12.289.1.2 scalar_t . . . 593
12.289.1.3 aligned_allocator . . . 594
12.289.1.4 aligned_vector . . . 594
12.289.1.5 is_same_element . . . 594
12.289.2 Member Function Documentation . . . 594
12.289.2.1 type_string() . . . 594
12.289.2.2 valid() . . . 594
12.289.2.3 compliant() . . . 594
12.289.2.4 set1() . . . 594
12.289.2.5 set() . . . 594
12.289.2.6 gather() . . . 594
12.289.2.7 load() . . . 594
12.289.2.8 loadu() . . . 595
12.289.2.9 store() . . . 595
12.289.2.10 storeu() . . . 595
12.289.2.11 stream() . . . 595
12.289.2.12 sll() . . . 595
12.289.2.13 srl() . . . 595
12.289.2.14 sra() . . . 595
12.289.2.15 shuffle() . . . 595
12.289.2.16 unpacklo_intrinsic() . . . 595
12.289.2.17 unpackhi_intrinsic() . . . 595
12.289.2.18 unpacklo() . . . 596
12.289.2.19 unpackhi() . . . 596
12.289.2.20 unpacklohi() . . . 596
12.289.2.21 pack_even() . . . 596
12.289.2.22 pack_odd() . . . 596
12.289.2.23 pack() . . . 596
12.289.2.24 transpose() . . . 596

12.289.2.25 blend() . . . 596
12.289.2.26 add() . . . 597
12.289.2.27 addin() . . . 597
12.289.2.28 sub() . . . 597
12.289.2.29 subin() . . . 597
12.289.2.30 mullo() . . . 597
12.289.2.31 mul() . . . 597
12.289.2.32 mulhi() . . . 597
12.289.2.33 mulx() . . . 597
12.289.2.34 fmadd() . . . 597
12.289.2.35 fmaddin() . . . 597
12.289.2.36 fmaddx() . . . 598
12.289.2.37 fmaddxin() . . . 598
12.289.2.38 fnmadd() . . . 598
12.289.2.39 fnmaddin() . . . 598
12.289.2.40 fnmaddx() . . . 598
12.289.2.41 fnmaddxin() . . . 598
12.289.2.42 fmsub() . . . 598
12.289.2.43 fmsubin() . . . 598
12.289.2.44 fmsubx() . . . 598
12.289.2.45 fmsubxin() . . . 599
12.289.2.46 eq() . . . 599
12.289.2.47 greater() . . . 599
12.289.2.48 lesser() . . . 599
12.289.2.49 greater_eq() . . . 599
12.289.2.50 lesser_eq() . . . 599
12.289.2.51 hadd_to_scal() . . . 599
12.289.2.52 round() . . . 599
12.289.2.53 mod() . . . 599
12.289.2.54 zero() . . . 600
12.289.2.55 sll128() . . . 600
12.289.2.56 srl128() . . . 600
12.289.2.57 vand() . . . 600
12.289.2.58 vor() . . . 600
12.289.2.59 vxor() . . . 600
12.289.2.60 vandnot() . . . 600
12.289.3 Field Documentation . . . 600
12.289.3.1 vect_size . . . 600
12.289.3.2 alignment . . . 600
12.290 Simd128_impl< true, true, true, 4 > Struct Reference . . . 601
12.290.1 Member Typedef Documentation . . . 602
12.290.1.1 vect_t . . . 602

12.290.1.2 scalar_t . . . 602
12.290.1.3 aligned_allocator . . . 603
12.290.1.4 aligned_vector . . . 603
12.290.1.5 is_same_element . . . 603
12.290.2 Member Function Documentation . . . 603
12.290.2.1 type_string() . . . 603
12.290.2.2 valid() . . . 603
12.290.2.3 compliant() . . . 603
12.290.2.4 set1() . . . 603
12.290.2.5 set() . . . 603
12.290.2.6 gather() . . . 603
12.290.2.7 load() . . . 603
12.290.2.8 loadu() . . . 603
12.290.2.9 store() . . . 604
12.290.2.10 storeu() . . . 604
12.290.2.11 stream() . . . 604
12.290.2.12 sll() . . . 604
12.290.2.13 srl() . . . 604
12.290.2.14 sra() . . . 604
12.290.2.15 shuffle() . . . 604
12.290.2.16 unpacklo_intrinsic() . . . 604
12.290.2.17 unpackhi_intrinsic() . . . 604
12.290.2.18 unpacklo() . . . 604
12.290.2.19 unpackhi() . . . 605
12.290.2.20 unpacklohi() . . . 605
12.290.2.21 pack_even() . . . 605
12.290.2.22 pack_odd() . . . 605
12.290.2.23 pack() . . . 605
12.290.2.24 transpose() . . . 605
12.290.2.25 blend() . . . 605
12.290.2.26 add() . . . 605
12.290.2.27 addin() . . . 605
12.290.2.28 sub() . . . 606
12.290.2.29 subin() . . . 606
12.290.2.30 mullo() . . . 606
12.290.2.31 mul() . . . 606
12.290.2.32 mulhi() . . . 606
12.290.2.33 mulx() . . . 606
12.290.2.34 fmadd() . . . 606
12.290.2.35 fmaddin() . . . 606
12.290.2.36 fmaddx() . . . 606
12.290.2.37 fmaddxin() . . . 607

12.290.2.38 fnmadd() . . . 607
12.290.2.39 fnmaddin() . . . 607
12.290.2.40 fnmaddx() . . . 607
12.290.2.41 fnmaddxin() . . . 607
12.290.2.42 fmsub() . . . 607
12.290.2.43 fmsubin() . . . 607
12.290.2.44 fmsubx() . . . 607
12.290.2.45 fmsubxin() . . . 607
12.290.2.46 eq() . . . 608
12.290.2.47 greater() . . . 608
12.290.2.48 lesser() . . . 608
12.290.2.49 greater_eq() . . . 608
12.290.2.50 lesser_eq() . . . 608
12.290.2.51 hadd_to_scal() . . . 608
12.290.2.52 round() . . . 608
12.290.2.53 mod() . . . 608
12.290.2.54 zero() . . . 608
12.290.2.55 sll128() . . . 609
12.290.2.56 srl128() . . . 609
12.290.2.57 vand() . . . 609
12.290.2.58 vor() . . . 609
12.290.2.59 vxor() . . . 609
12.290.2.60 vandnot() . . . 609
12.290.3 Field Documentation . . . 609
12.290.3.1 vect_size . . . 609
12.290.3.2 alignment . . . 609
12.291 Simd128_impl< true, true, true, 8 > Struct Reference . . . 609
12.291.1 Member Typedef Documentation . . . 611
12.291.1.1 vect_t . . . 611
12.291.1.2 scalar_t . . . 612
12.291.1.3 aligned_allocator . . . 612
12.291.1.4 aligned_vector . . . 612
12.291.1.5 is_same_element . . . 612
12.291.2 Member Function Documentation . . . 612
12.291.2.1 type_string() . . . 612
12.291.2.2 valid() . . . 612
12.291.2.3 compliant() . . . 612
12.291.2.4 set1() . . . 612
12.291.2.5 set() . . . 612
12.291.2.6 gather() . . . 612
12.291.2.7 get() . . . 612
12.291.2.8 load() . . . 613

12.291.2.9 loadu() . . . 613
12.291.2.10 store() . . . 613
12.291.2.11 storeu() . . . 613
12.291.2.12 stream() . . . 613
12.291.2.13 sll() . . . 613
12.291.2.14 srl() . . . 613
12.291.2.15 sra() . . . 613
12.291.2.16 shuffle() . . . 613
12.291.2.17 unpacklo_intrinsic() . . . 613
12.291.2.18 unpackhi_intrinsic() . . . 614
12.291.2.19 unpacklo() . . . 614
12.291.2.20 unpackhi() . . . 614
12.291.2.21 unpacklohi() . . . 614
12.291.2.22 pack_even() . . . 614
12.291.2.23 pack_odd() . . . 614
12.291.2.24 pack() . . . 614
12.291.2.25 transpose() . . . 614
12.291.2.26 blend() . . . 614
12.291.2.27 add() . . . 615
12.291.2.28 addin() . . . 615
12.291.2.29 sub() . . . 615
12.291.2.30 subin() . . . 615
12.291.2.31 mullo() . . . 615
12.291.2.32 mul() . . . 615
12.291.2.33 mulhi() . . . 615
12.291.2.34 mulx() . . . 615
12.291.2.35 fmadd() . . . 615
12.291.2.36 fmaddin() . . . 615
12.291.2.37 fmaddx() . . . 616
12.291.2.38 fmaddxin() . . . 616
12.291.2.39 fnmadd() . . . 616
12.291.2.40 fnmaddin() . . . 616
12.291.2.41 fnmaddx() . . . 616
12.291.2.42 fnmaddxin() . . . 616
12.291.2.43 fmsub() . . . 616
12.291.2.44 fmsubin() . . . 616
12.291.2.45 fmsubx() . . . 616
12.291.2.46 fmsubxin() . . . 617
12.291.2.47 eq() . . . 617
12.291.2.48 greater() . . . 617
12.291.2.49 lesser() . . . 617
12.291.2.50 greater_eq() . . . 617

12.291.2.51 lesser_eq() . . . 617
12.291.2.52 hadd_to_scal() . . . 617
12.291.2.53 round() . . . 617
12.291.2.54 mask_high() . . . 617
12.291.2.55 mulhi_fast() . . . 617
12.291.2.56 mod() . . . 618
12.291.2.57 signbits() . . . 618
12.291.2.58 zero() . . . 618
12.291.2.59 sll128() . . . 618
12.291.2.60 srl128() . . . 618
12.291.2.61 vand() . . . 618
12.291.2.62 vor() . . . 618
12.291.2.63 vxor() . . . 618
12.291.2.64 vandnot() . . . 618
12.291.3 Field Documentation . . . 619
12.291.3.1 vect_size . . . 619
12.291.3.2 alignment . . . 619
12.292 Simd128i_base Struct Reference . . . 619
12.292.1 Member Typedef Documentation . . . 619
12.292.1.1 vect_t . . . 619
12.292.2 Member Function Documentation . . . 619
12.292.2.1 zero() . . . 619
12.292.2.2 sll128() . . . 619
12.292.2.3 srl128() . . . 620
12.292.2.4 vand() . . . 620
12.292.2.5 vor() . . . 620
12.292.2.6 vxor() . . . 620
12.292.2.7 vandnot() . . . 620
12.293 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . 620
12.294 Simd256_impl< true, false, true, 4 > Struct Reference . . . 620
12.295 Simd256_impl< true, false, true, 8 > Struct Reference . . . 621
12.295.1 Member Typedef Documentation . . . 622
12.295.1.1 vect_t . . . 622
12.295.1.2 scalar_t . . . 622
12.295.1.3 aligned_allocator . . . 622
12.295.1.4 aligned_vector . . . 622
12.295.1.5 is_same_element . . . 622
12.295.2 Member Function Documentation . . . 622
12.295.2.1 type_string() . . . 622
12.295.2.2 valid() . . . 623
12.295.2.3 compliant() . . . 623
12.295.2.4 zero() . . . 623

12.295.2.5 set1() . . . 623
12.295.2.6 set() . . . 623
12.295.2.7 gather() . . . 623
12.295.2.8 load() . . . 623
12.295.2.9 loadu() . . . 623
12.295.2.10 store() . . . 623
12.295.2.11 storeu() . . . 623
12.295.2.12 stream() . . . 624
12.295.2.13 unpacklo_intrinsic() . . . 624
12.295.2.14 unpackhi_intrinsic() . . . 624
12.295.2.15 unpacklo() . . . 624
12.295.2.16 unpackhi() . . . 624
12.295.2.17 unpacklohi() . . . 624
12.295.2.18 pack_even() . . . 624
12.295.2.19 pack_odd() . . . 624
12.295.2.20 pack() . . . 624
12.295.2.21 transpose() . . . 625
12.295.2.22 blend() . . . 625
12.295.2.23 blendv() . . . 625
12.295.2.24 add() . . . 625
12.295.2.25 addin() . . . 625
12.295.2.26 sub() . . . 625
12.295.2.27 subin() . . . 625
12.295.2.28 mul() . . . 625
12.295.2.29 mulin() . . . 625
12.295.2.30 div() . . . 626
12.295.2.31 fmadd() . . . 626
12.295.2.32 fmaddin() . . . 626
12.295.2.33 fnmadd() . . . 626
12.295.2.34 fnmaddin() . . . 626
12.295.2.35 fmsub() . . . 626
12.295.2.36 fmsubin() . . . 626
12.295.2.37 eq() . . . 626
12.295.2.38 lesser() . . . 626
12.295.2.39 lesser_eq() . . . 627
12.295.2.40 greater() . . . 627
12.295.2.41 greater_eq() . . . 627
12.295.2.42 vand() . . . 627
12.295.2.43 vor() . . . 627
12.295.2.44 vxor() . . . 627
12.295.2.45 vandnot() . . . 627
12.295.2.46 floor() . . . 627

12.295.2.47 ceil() . . . 627
12.295.2.48 round() . . . 627
12.295.2.49 hadd() . . . 628
12.295.2.50 hadd_to_scal() . . . 628
12.295.2.51 mod() . . . 628
12.295.3 Field Documentation . . . 628
12.295.3.1 vect_size . . . 628
12.295.3.2 alignment . . . 628
12.296 Simd256_impl< true, true, false, 2 > Struct Reference . . . 628
12.296.1 Member Typedef Documentation . . . 630
12.296.1.1 scalar_t . . . 630
12.296.1.2 aligned_allocator . . . 630
12.296.1.3 aligned_vector . . . 630
12.296.1.4 is_same_element . . . 630
12.296.1.5 simdHalf . . . 631
12.296.1.6 vect_t . . . 631
12.296.1.7 half_t . . . 631
12.296.2 Member Function Documentation . . . 631
12.296.2.1 type_string() . . . 631
12.296.2.2 set1() [1/2] . . . 631
12.296.2.3 set() [1/2] . . . 631
12.296.2.4 gather() [1/2] . . . 631
12.296.2.5 load() [1/2] . . . 631
12.296.2.6 loadu() [1/2] . . . 631
12.296.2.7 store() [1/2] . . . 632
12.296.2.8 storeu() [1/2] . . . 632
12.296.2.9 stream() [1/2] . . . 632
12.296.2.10 sra() . . . 632
12.296.2.11 greater() . . . 632
12.296.2.12 lesser() . . . 632
12.296.2.13 greater_eq() . . . 632
12.296.2.14 lesser_eq() . . . 632
12.296.2.15 mulhi() . . . 632
12.296.2.16 mulx() . . . 632
12.296.2.17 fmaddx() . . . 633
12.296.2.18 fmaddxin() . . . 633
12.296.2.19 fnmaddx() . . . 633
12.296.2.20 fnmaddxin() . . . 633
12.296.2.21 fmsubx() . . . 633
12.296.2.22 fmsubxin() . . . 633
12.296.2.23 hadd_to_scal() . . . 633
12.296.2.24 valid() . . . 633

12.296.2.25 compliant() . . . 633
12.296.2.26 set1() [2/2] . . . 634
12.296.2.27 set() [2/2] . . . 634
12.296.2.28 gather() [2/2] . . . 634
12.296.2.29 load() [2/2] . . . 634
12.296.2.30 loadu() [2/2] . . . 634
12.296.2.31 store() [2/2] . . . 634
12.296.2.32 storeu() [2/2] . . . 634
12.296.2.33 stream() [2/2] . . . 634
12.296.2.34 sll() . . . 635
12.296.2.35 srl() . . . 635
12.296.2.36 shuffle() . . . 635
12.296.2.37 unpacklo_intrinsic() . . . 635
12.296.2.38 unpackhi_intrinsic() . . . 635
12.296.2.39 unpacklo() . . . 635
12.296.2.40 unpackhi() . . . 635
12.296.2.41 unpacklohi() . . . 635
12.296.2.42 pack_even() . . . 635
12.296.2.43 pack_odd() . . . 635
12.296.2.44 pack() . . . 636
12.296.2.45 transpose() . . . 636
12.296.2.46 blend() [1/2] . . . 636
12.296.2.47 blend() [2/2] . . . 636
12.296.2.48 add() . . . 636
12.296.2.49 addin() . . . 636
12.296.2.50 sub() . . . 636
12.296.2.51 subin() . . . 637
12.296.2.52 mullo() . . . 637
12.296.2.53 mul() . . . 637
12.296.2.54 fmadd() . . . 637
12.296.2.55 fmaddin() . . . 637
12.296.2.56 fnmadd() . . . 637
12.296.2.57 fnmaddin() . . . 637
12.296.2.58 fmsub() . . . 637
12.296.2.59 fmsubin() . . . 637
12.296.2.60 eq() . . . 638
12.296.2.61 round() . . . 638
12.296.2.62 mod() . . . 638
12.296.2.63 zero() . . . 638
12.296.3 Field Documentation . . . 638
12.296.3.1 vect_size . . . 638
12.296.3.2 alignment . . . 638

12.297 Simd256_impl< true, true, false, 4 > Struct Reference 638
12.297.1 Member Typedef Documentation 642
12.297.1.1 scalar_t [1/2] 642
12.297.1.2 aligned_allocator [1/2] 642
12.297.1.3 aligned_vector [1/2] 642
12.297.1.4 is_same_element [1/2] 642
12.297.1.5 simdHalf [1/2] 642
12.297.1.6 scalar_t [2/2] 642
12.297.1.7 aligned_allocator [2/2] 642
12.297.1.8 aligned_vector [2/2] 642
12.297.1.9 is_same_element [2/2] 642
12.297.1.10 simdHalf [2/2] 642
12.297.1.11 vect_t [1/2] 642
12.297.1.12 vect_t [2/2] 642
12.297.1.13 half_t [1/2] 642
12.297.1.14 half_t [2/2] 643
12.297.2 Member Function Documentation 643
12.297.2.1 type_string() [1/2] 643
12.297.2.2 set1() [1/3] 643
12.297.2.3 set() [1/4] 643
12.297.2.4 gather() [1/3] 643
12.297.2.5 load() [1/3] 643
12.297.2.6 loadu() [1/3] 643
12.297.2.7 store() [1/3] 643
12.297.2.8 storeu() [1/3] 643
12.297.2.9 stream() [1/3] 643
12.297.2.10 sra() [1/2] 644
12.297.2.11 greater() [1/2] 644
12.297.2.12 lesser() [1/2] 644
12.297.2.13 greater_eq() [1/2] 644
12.297.2.14 lesser_eq() [1/2] 644
12.297.2.15 mulhi() [1/2] 644
12.297.2.16 mulx() [1/2] 644
12.297.2.17 fmaddx() [1/2] 644
12.297.2.18 fmaddxin() [1/2] 644
12.297.2.19 fnmaddx() [1/2] 645
12.297.2.20 fnmaddxin() [1/2] 645
12.297.2.21 fmsubx() [1/2] 645
12.297.2.22 fmsubxin() [1/2] 645
12.297.2.23 hadd_to_scal() [1/2] 645
12.297.2.24 type_string() [2/2] 645
12.297.2.25 set1() [2/3] 645

12.297.2.26 set() [2/4] . . . 645
12.297.2.27 gather() [2/3] . . . 645
12.297.2.28 load() [2/3] . . . 646
12.297.2.29 loadu() [2/3] . . . 646
12.297.2.30 store() [2/3] . . . 646
12.297.2.31 storeu() [2/3] . . . 646
12.297.2.32 stream() [2/3] . . . 646
12.297.2.33 sra() [2/2] . . . 646
12.297.2.34 greater() [2/2] . . . 646
12.297.2.35 lesser() [2/2] . . . 646
12.297.2.36 greater_eq() [2/2] . . . 646
12.297.2.37 lesser_eq() [2/2] . . . 646
12.297.2.38 mulhi() [2/2] . . . 647
12.297.2.39 mulx() [2/2] . . . 647
12.297.2.40 fmaddx() [2/2] . . . 647
12.297.2.41 fmaddxin() [2/2] . . . 647
12.297.2.42 fnmaddx() [2/2] . . . 647
12.297.2.43 fnmaddxin() [2/2] . . . 647
12.297.2.44 fmsubx() [2/2] . . . 647
12.297.2.45 fmsubxin() [2/2] . . . 647
12.297.2.46 hadd_to_scal() [2/2] . . . 647
12.297.2.47 valid() [1/2] . . . 648
12.297.2.48 valid() [2/2] . . . 648
12.297.2.49 compliant() [1/2] . . . 648
12.297.2.50 compliant() [2/2] . . . 648
12.297.2.51 set1() [3/3] . . . 648
12.297.2.52 set() [3/4] . . . 648
12.297.2.53 set() [4/4] . . . 648
12.297.2.54 gather() [3/3] . . . 649
12.297.2.55 load() [3/3] . . . 649
12.297.2.56 loadu() [3/3] . . . 649
12.297.2.57 store() [3/3] . . . 649
12.297.2.58 storeu() [3/3] . . . 649
12.297.2.59 stream() [3/3] . . . 649
12.297.2.60 sll() [1/2] . . . 649
12.297.2.61 sll() [2/2] . . . 649
12.297.2.62 srl() [1/2] . . . 649
12.297.2.63 srl() [2/2] . . . 649
12.297.2.64 shuffle_twice() [1/2] . . . 650
12.297.2.65 shuffle_twice() [2/2] . . . 650
12.297.2.66 shuffle() [1/2] . . . 650
12.297.2.67 shuffle() [2/2] . . . 650

12.297.2.68 unpacklo_intrinsic() [1/2] . . . 650
12.297.2.69 unpacklo_intrinsic() [2/2] . . . 650
12.297.2.70 unpackhi_intrinsic() [1/2] . . . 650
12.297.2.71 unpackhi_intrinsic() [2/2] . . . 650
12.297.2.72 unpacklo() [1/2] . . . 650
12.297.2.73 unpacklo() [2/2] . . . 650
12.297.2.74 unpackhi() [1/2] . . . 651
12.297.2.75 unpackhi() [2/2] . . . 651
12.297.2.76 unpacklohi() [1/2] . . . 651
12.297.2.77 unpacklohi() [2/2] . . . 651
12.297.2.78 pack_even() [1/2] . . . 651
12.297.2.79 pack_even() [2/2] . . . 651
12.297.2.80 pack_odd() [1/2] . . . 651
12.297.2.81 pack_odd() [2/2] . . . 651
12.297.2.82 pack() [1/2] . . . 651
12.297.2.83 pack() [2/2] . . . 652
12.297.2.84 transpose() [1/2] . . . 652
12.297.2.85 transpose() [2/2] . . . 652
12.297.2.86 blend() [1/2] . . . 652
12.297.2.87 blend() [2/2] . . . 652
12.297.2.88 add() [1/2] . . . 652
12.297.2.89 add() [2/2] . . . 653
12.297.2.90 addin() [1/2] . . . 653
12.297.2.91 addin() [2/2] . . . 653
12.297.2.92 sub() [1/2] . . . 653
12.297.2.93 sub() [2/2] . . . 653
12.297.2.94 subin() [1/2] . . . 653
12.297.2.95 subin() [2/2] . . . 653
12.297.2.96 mullo() [1/2] . . . 653
12.297.2.97 mullo() [2/2] . . . 653
12.297.2.98 mul() [1/2] . . . 653
12.297.2.99 mul() [2/2] . . . 654
12.297.2.100 fmadd() [1/2] . . . 654
12.297.2.101 fmadd() [2/2] . . . 654
12.297.2.102 fmaddin() [1/2] . . . 654
12.297.2.103 fmaddin() [2/2] . . . 654
12.297.2.104 fnmadd() [1/2] . . . 654
12.297.2.105 fnmadd() [2/2] . . . 654
12.297.2.106 fnmaddin() [1/2] . . . 654
12.297.2.107 fnmaddin() [2/2] . . . 654
12.297.2.108 fmsub() [1/2] . . . 655
12.297.2.109 fmsub() [2/2] . . . 655

12.297.2.110 fmsubin() [1/2] . . . 655
12.297.2.111 fmsubin() [2/2] . . . 655
12.297.2.112 eq() [1/2] . . . 655
12.297.2.113 eq() [2/2] . . . 655
12.297.2.114 round() [1/2] . . . 655
12.297.2.115 round() [2/2] . . . 655
12.297.2.116 mod() [1/2] . . . 655
12.297.2.117 mod() [2/2] . . . 656
12.297.2.118 zero() [1/2] . . . 656
12.297.2.119 zero() [2/2] . . . 656
12.297.2.120 vor() . . . 656
12.297.2.121 vxor() . . . 656
12.297.2.122 vand() . . . 656
12.297.2.123 vandnot() . . . 656
12.297.3 Field Documentation . . . 656
12.297.3.1 vect_size . . . 656
12.297.3.2 alignment . . . 656
12.298 Simd256_impl< true, true, false, 8 > Struct Reference . . . 657
12.298.1 Member Typedef Documentation . . . 659
12.298.1.1 scalar_t . . . 659
12.298.1.2 aligned_allocator . . . 659
12.298.1.3 aligned_vector . . . 659
12.298.1.4 is_same_element . . . 659
12.298.1.5 simdHalf . . . 659
12.298.1.6 vect_t . . . 659
12.298.1.7 half_t . . . 659
12.298.2 Member Function Documentation . . . 659
12.298.2.1 type_string() . . . 659
12.298.2.2 set1() [1/2] . . . 659
12.298.2.3 set() [1/2] . . . 659
12.298.2.4 gather() [1/2] . . . 660
12.298.2.5 load() [1/2] . . . 660
12.298.2.6 loadu() [1/2] . . . 660
12.298.2.7 store() [1/2] . . . 660
12.298.2.8 storeu() [1/2] . . . 660
12.298.2.9 stream() [1/2] . . . 660
12.298.2.10 sra() . . . 660
12.298.2.11 greater() . . . 660
12.298.2.12 lesser() . . . 660
12.298.2.13 greater_eq() . . . 660
12.298.2.14 lesser_eq() . . . 661
12.298.2.15 mullo() . . . 661

12.298.2.16 mulhi() . . . 661
12.298.2.17 mulx() . . . 661
12.298.2.18 fmaddx() . . . 661
12.298.2.19 fmaddxin() . . . 661
12.298.2.20 fnmaddx() . . . 661
12.298.2.21 fnmaddxin() . . . 661
12.298.2.22 fmsubx() . . . 661
12.298.2.23 fmsubxin() . . . 662
12.298.2.24 hadd_to_scal() . . . 662
12.298.2.25 valid() . . . 662
12.298.2.26 compliant() . . . 662
12.298.2.27 set1() [2/2] . . . 662
12.298.2.28 set() [2/2] . . . 662
12.298.2.29 gather() [2/2] . . . 662
12.298.2.30 get() . . . 662
12.298.2.31 load() [2/2] . . . 662
12.298.2.32 loadu() [2/2] . . . 662
12.298.2.33 store() [2/2] . . . 663
12.298.2.34 storeu() [2/2] . . . 663
12.298.2.35 stream() [2/2] . . . 663
12.298.2.36 sll() . . . 663
12.298.2.37 srl() . . . 663
12.298.2.38 shuffle() . . . 663
12.298.2.39 unpacklo_intrinsic() . . . 663
12.298.2.40 unpackhi_intrinsic() . . . 663
12.298.2.41 unpacklo() . . . 663
12.298.2.42 unpackhi() . . . 663
12.298.2.43 unpacklohi() . . . 664
12.298.2.44 pack_even() . . . 664
12.298.2.45 pack_odd() . . . 664
12.298.2.46 pack() . . . 664
12.298.2.47 transpose() . . . 664
12.298.2.48 blend() . . . 664
12.298.2.49 add() . . . 664
12.298.2.50 addin() . . . 664
12.298.2.51 sub() . . . 664
12.298.2.52 subin() . . . 665
12.298.2.53 mul() . . . 665
12.298.2.54 fmadd() . . . 665
12.298.2.55 fmaddin() . . . 665
12.298.2.56 fnmadd() . . . 665
12.298.2.57 fnmaddin() . . . 665

12.298.2.58 fmsub() . . . 665
12.298.2.59 fmsubin() . . . 665
12.298.2.60 eq() . . . 665
12.298.2.61 round() . . . 666
12.298.2.62 mask_high() . . . 666
12.298.2.63 mulhi_fast() . . . 666
12.298.2.64 mod() . . . 666
12.298.2.65 signbits() . . . 666
12.298.2.66 zero() . . . 666
12.298.3 Field Documentation . . . 666
12.298.3.1 vect_size . . . 666
12.298.3.2 alignment . . . 666
12.299 Simd256_impl< true, true, true, 2 > Struct Reference . . . 666
12.299.1 Member Typedef Documentation . . . 668
12.299.1.1 vect_t . . . 668
12.299.1.2 half_t . . . 668
12.299.1.3 scalar_t . . . 668
12.299.1.4 simdHalf . . . 669
12.299.1.5 aligned_allocator . . . 669
12.299.1.6 aligned_vector . . . 669
12.299.1.7 is_same_element . . . 669
12.299.2 Member Function Documentation . . . 669
12.299.2.1 type_string() . . . 669
12.299.2.2 valid() . . . 669
12.299.2.3 compliant() . . . 669
12.299.2.4 set1() . . . 669
12.299.2.5 set() . . . 669
12.299.2.6 gather() . . . 670
12.299.2.7 load() . . . 670
12.299.2.8 loadu() . . . 670
12.299.2.9 store() . . . 670
12.299.2.10 storeu() . . . 670
12.299.2.11 stream() . . . 670
12.299.2.12 sll() . . . 670
12.299.2.13 srl() . . . 670
12.299.2.14 sra() . . . 670
12.299.2.15 shuffle() . . . 670
12.299.2.16 unpacklo_intrinsic() . . . 671
12.299.2.17 unpackhi_intrinsic() . . . 671
12.299.2.18 unpacklo() . . . 671
12.299.2.19 unpackhi() . . . 671
12.299.2.20 unpacklohi() . . . 671

12.299.2.21 pack_even() . . . 671
12.299.2.22 pack_odd() . . . 671
12.299.2.23 pack() . . . 671
12.299.2.24 transpose() . . . 671
12.299.2.25 blend() [1/2] . . . 672
12.299.2.26 blend() [2/2] . . . 672
12.299.2.27 add() . . . 672
12.299.2.28 addin() . . . 672
12.299.2.29 sub() . . . 672
12.299.2.30 subin() . . . 672
12.299.2.31 mullo() . . . 672
12.299.2.32 mul() . . . 673
12.299.2.33 mulhi() . . . 673
12.299.2.34 mulx() . . . 673
12.299.2.35 fmadd() . . . 673
12.299.2.36 fmaddin() . . . 673
12.299.2.37 fmaddx() . . . 673
12.299.2.38 fmaddxin() . . . 673
12.299.2.39 fnmadd() . . . 673
12.299.2.40 fnmaddin() . . . 673
12.299.2.41 fnmaddx() . . . 674
12.299.2.42 fnmaddxin() . . . 674
12.299.2.43 fmsub() . . . 674
12.299.2.44 fmsubin() . . . 674
12.299.2.45 fmsubx() . . . 674
12.299.2.46 fmsubxin() . . . 674
12.299.2.47 eq() . . . 674
12.299.2.48 greater() . . . 674
12.299.2.49 lesser() . . . 674
12.299.2.50 greater_eq() . . . 675
12.299.2.51 lesser_eq() . . . 675
12.299.2.52 hadd_to_scal() . . . 675
12.299.2.53 round() . . . 675
12.299.2.54 mod() . . . 675
12.299.2.55 zero() . . . 675
12.299.3 Field Documentation . . . 675
12.299.3.1 vect_size . . . 675
12.299.3.2 alignment . . . 675
12.300 Simd256_impl< true, true, true, 4 > Struct Reference . . . 675
12.300.1 Member Typedef Documentation . . . 679
12.300.1.1 vect_t [1/2] . . . 679
12.300.1.2 half_t [1/2] . . . 679

12.300.1.3 scalar_t [1/2] . . . 679
12.300.1.4 simdHalf [1/2] . . . 679
12.300.1.5 aligned_allocator [1/2] . . . 679
12.300.1.6 aligned_vector [1/2] . . . 679
12.300.1.7 is_same_element [1/2] . . . 679
12.300.1.8 vect_t [2/2] . . . 679
12.300.1.9 half_t [2/2] . . . 679
12.300.1.10 scalar_t [2/2] . . . 679
12.300.1.11 simdHalf [2/2] . . . 679
12.300.1.12 aligned_allocator [2/2] . . . 680
12.300.1.13 aligned_vector [2/2] . . . 680
12.300.1.14 is_same_element [2/2] . . . 680
12.300.2 Member Function Documentation . . . 680
12.300.2.1 type_string() [1/2] . . . 680
12.300.2.2 valid() [1/2] . . . 680
12.300.2.3 compliant() [1/2] . . . 680
12.300.2.4 set1() [1/2] . . . 680
12.300.2.5 set() [1/2] . . . 680
12.300.2.6 gather() [1/2] . . . 680
12.300.2.7 load() [1/2] . . . 680
12.300.2.8 loadu() [1/2] . . . 681
12.300.2.9 store() [1/2] . . . 681
12.300.2.10 storeu() [1/2] . . . 681
12.300.2.11 stream() [1/2] . . . 681
12.300.2.12 sll() [1/2] . . . 681
12.300.2.13 srl() [1/2] . . . 681
12.300.2.14 sra() [1/2] . . . 681
12.300.2.15 shuffle_twice() [1/2] . . . 681
12.300.2.16 shuffle() [1/2] . . . 681
12.300.2.17 unpacklo_intrinsic() [1/2] . . . 681
12.300.2.18 unpackhi_intrinsic() [1/2] . . . 682
12.300.2.19 unpacklo() [1/2] . . . 682
12.300.2.20 unpackhi() [1/2] . . . 682
12.300.2.21 unpacklohi() [1/2] . . . 682
12.300.2.22 pack_even() [1/2] . . . 682
12.300.2.23 pack_odd() [1/2] . . . 682
12.300.2.24 pack() [1/2] . . . 682
12.300.2.25 transpose() [1/2] . . . 682
12.300.2.26 blend() [1/2] . . . 683
12.300.2.27 add() [1/2] . . . 683
12.300.2.28 addin() [1/2] . . . 683
12.300.2.29 sub() [1/2] . . . 683

12.300.2.30 subin() [1/2] . . . 683
12.300.2.31 mullo() [1/2] . . . 683
12.300.2.32 mul() [1/2] . . . 683
12.300.2.33 mulhi() [1/2] . . . 683
12.300.2.34 mulx() [1/2] . . . 683
12.300.2.35 fmadd() [1/2] . . . 683
12.300.2.36 fmaddin() [1/2] . . . 684
12.300.2.37 fmaddx() [1/2] . . . 684
12.300.2.38 fmaddxin() [1/2] . . . 684
12.300.2.39 fnmadd() [1/2] . . . 684
12.300.2.40 fnmaddin() [1/2] . . . 684
12.300.2.41 fnmaddx() [1/2] . . . 684
12.300.2.42 fnmaddxin() [1/2] . . . 684
12.300.2.43 fmsub() [1/2] . . . 684
12.300.2.44 fmsubin() [1/2] . . . 684
12.300.2.45 fmsubx() [1/2] . . . 685
12.300.2.46 fmsubxin() [1/2] . . . 685
12.300.2.47 eq() [1/2] . . . 685
12.300.2.48 greater() [1/2] . . . 685
12.300.2.49 lesser() [1/2] . . . 685
12.300.2.50 greater_eq() [1/2] . . . 685
12.300.2.51 lesser_eq() [1/2] . . . 685
12.300.2.52 hadd_to_scal() [1/2] . . . 685
12.300.2.53 round() [1/2] . . . 685
12.300.2.54 mod() [1/2] . . . 686
12.300.2.55 type_string() [2/2] . . . 686
12.300.2.56 valid() [2/2] . . . 686
12.300.2.57 compliant() [2/2] . . . 686
12.300.2.58 set1() [2/2] . . . 686
12.300.2.59 set() [2/2] . . . 686
12.300.2.60 gather() [2/2] . . . 686
12.300.2.61 load() [2/2] . . . 687
12.300.2.62 loadu() [2/2] . . . 687
12.300.2.63 store() [2/2] . . . 687
12.300.2.64 storeu() [2/2] . . . 687
12.300.2.65 stream() [2/2] . . . 687
12.300.2.66 sll() [2/2] . . . 687
12.300.2.67 srl() [2/2] . . . 687
12.300.2.68 sra() [2/2] . . . 687
12.300.2.69 shuffle_twice() [2/2] . . . 687
12.300.2.70 shuffle() [2/2] . . . 687
12.300.2.71 unpacklo_intrinsic() [2/2] . . . 688

12.300.2.72 unpackhi_intrinsic() [2/2] . . . 688
12.300.2.73 unpacklo() [2/2] . . . 688
12.300.2.74 unpackhi() [2/2] . . . 688
12.300.2.75 unpacklohi() [2/2] . . . 688
12.300.2.76 pack_even() [2/2] . . . 688
12.300.2.77 pack_odd() [2/2] . . . 688
12.300.2.78 pack() [2/2] . . . 688
12.300.2.79 transpose() [2/2] . . . 688
12.300.2.80 blend() [2/2] . . . 689
12.300.2.81 add() [2/2] . . . 689
12.300.2.82 addin() [2/2] . . . 689
12.300.2.83 sub() [2/2] . . . 689
12.300.2.84 subin() [2/2] . . . 689
12.300.2.85 mullo() [2/2] . . . 689
12.300.2.86 mul() [2/2] . . . 689
12.300.2.87 mulhi() [2/2] . . . 689
12.300.2.88 mulx() [2/2] . . . 690
12.300.2.89 fmadd() [2/2] . . . 690
12.300.2.90 fmaddin() [2/2] . . . 690
12.300.2.91 fmaddx() [2/2] . . . 690
12.300.2.92 fmaddxin() [2/2] . . . 690
12.300.2.93 fnmadd() [2/2] . . . 690
12.300.2.94 fnmaddin() [2/2] . . . 690
12.300.2.95 fnmaddx() [2/2] . . . 690
12.300.2.96 fnmaddxin() [2/2] . . . 690
12.300.2.97 fmsub() [2/2] . . . 691
12.300.2.98 fmsubin() [2/2] . . . 691
12.300.2.99 fmsubx() [2/2] . . . 691
12.300.2.100 fmsubxin() [2/2] . . . 691
12.300.2.101 eq() [2/2] . . . 691
12.300.2.102 greater() [2/2] . . . 691
12.300.2.103 lesser() [2/2] . . . 691
12.300.2.104 greater_eq() [2/2] . . . 691
12.300.2.105 lesser_eq() [2/2] . . . 691
12.300.2.106 hadd_to_scal() [2/2] . . . 692
12.300.2.107 round() [2/2] . . . 692
12.300.2.108 mod() [2/2] . . . 692
12.300.2.109 zero() [1/2] . . . 692
12.300.2.110 zero() [2/2] . . . 692
12.300.2.111 vor() . . . 692
12.300.2.112 vxor() . . . 692
12.300.2.113 vand() . . . 692

12.300.2.114 vandnot() . . . 692
12.300.3 Field Documentation . . . 692
12.300.3.1 vect_size . . . 692
12.300.3.2 alignment . . . 693
12.301 Simd256_impl< true, true, true, 8 > Struct Reference . . . 693
12.301.1 Member Typedef Documentation . . . 695
12.301.1.1 vect_t . . . 695
12.301.1.2 half_t . . . 695
12.301.1.3 scalar_t . . . 695
12.301.1.4 simdHalf . . . 695
12.301.1.5 aligned_allocator . . . 695
12.301.1.6 aligned_vector . . . 695
12.301.1.7 is_same_element . . . 695
12.301.2 Member Function Documentation . . . 695
12.301.2.1 type_string() . . . 695
12.301.2.2 valid() . . . 695
12.301.2.3 compliant() . . . 695
12.301.2.4 set1() . . . 695
12.301.2.5 set() . . . 696
12.301.2.6 gather() . . . 696
12.301.2.7 get() . . . 696
12.301.2.8 load() . . . 696
12.301.2.9 loadu() . . . 696
12.301.2.10 store() . . . 696
12.301.2.11 storeu() . . . 696
12.301.2.12 stream() . . . 696
12.301.2.13 sll() . . . 696
12.301.2.14 srl() . . . 696
12.301.2.15 sra() . . . 697
12.301.2.16 shuffle() . . . 697
12.301.2.17 unpacklo_intrinsic() . . . 697
12.301.2.18 unpackhi_intrinsic() . . . 697
12.301.2.19 unpacklo() . . . 697
12.301.2.20 unpackhi() . . . 697
12.301.2.21 unpacklohi() . . . 697
12.301.2.22 pack_even() . . . 697
12.301.2.23 pack_odd() . . . 697
12.301.2.24 pack() . . . 698
12.301.2.25 transpose() . . . 698
12.301.2.26 blend() . . . 698
12.301.2.27 add() . . . 698
12.301.2.28 addin() . . . 698

12.301.2.29 sub() . . . 698
12.301.2.30 subin() . . . 698
12.301.2.31 mullo() . . . 698
12.301.2.32 mul() . . . 698
12.301.2.33 mulhi() . . . 699
12.301.2.34 mulx() . . . 699
12.301.2.35 fmadd() . . . 699
12.301.2.36 fmaddin() . . . 699
12.301.2.37 fmaddx() . . . 699
12.301.2.38 fmaddxin() . . . 699
12.301.2.39 fnmadd() . . . 699
12.301.2.40 fnmaddin() . . . 699
12.301.2.41 fnmaddx() . . . 699
12.301.2.42 fnmaddxin() . . . 700
12.301.2.43 fmsub() . . . 700
12.301.2.44 fmsubin() . . . 700
12.301.2.45 fmsubx() . . . 700
12.301.2.46 fmsubxin() . . . 700
12.301.2.47 eq() . . . 700
12.301.2.48 greater() . . . 700
12.301.2.49 lesser() . . . 700
12.301.2.50 greater_eq() . . . 700
12.301.2.51 lesser_eq() . . . 701
12.301.2.52 hadd_to_scal() . . . 701
12.301.2.53 round() . . . 701
12.301.2.54 mask_high() . . . 701
12.301.2.55 mulhi_fast() . . . 701
12.301.2.56 mod() . . . 701
12.301.2.57 signbits() . . . 701
12.301.2.58 zero() . . . 701
12.301.3 Field Documentation . . . 701
12.301.3.1 vect_size . . . 701
12.301.3.2 alignment . . . 701
12.302 Simd256fp_base Struct Reference . . . 702
12.303 Simd256i_base Struct Reference . . . 702
12.303.1 Member Typedef Documentation . . . 702
12.303.1.1 vect_t . . . 702
12.303.2 Member Function Documentation . . . 702
12.303.2.1 zero() . . . 702
12.304 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . 702
12.305 Simd512_impl< true, false, true, 4 > Struct Reference . . . 702
12.306 Simd512_impl< true, false, true, 8 > Struct Reference . . . 703

12.306.1 Member Typedef Documentation . . . 704
12.306.1.1 vect_t . . . 704
12.306.1.2 scalar_t . . . 704
12.306.1.3 aligned_allocator . . . 704
12.306.1.4 aligned_vector . . . 704
12.306.1.5 is_same_element . . . 704
12.306.2 Member Function Documentation . . . 704
12.306.2.1 type_string() . . . 704
12.306.2.2 valid() . . . 704
12.306.2.3 compliant() . . . 704
12.306.2.4 zero() . . . 705
12.306.2.5 set1() . . . 705
12.306.2.6 set() . . . 705
12.306.2.7 gather() . . . 705
12.306.2.8 load() . . . 705
12.306.2.9 loadu() . . . 705
12.306.2.10 store() . . . 705
12.306.2.11 storeu() . . . 705
12.306.2.12 stream() . . . 705
12.306.2.13 shuffle() . . . 705
12.306.2.14 unpacklo_intrinsic() . . . 706
12.306.2.15 unpackhi_intrinsic() . . . 706
12.306.2.16 unpacklo() . . . 706
12.306.2.17 unpackhi() . . . 706
12.306.2.18 unpacklohi() . . . 706
12.306.2.19 pack_even() . . . 706
12.306.2.20 pack_odd() . . . 706
12.306.2.21 pack() . . . 706
12.306.2.22 transpose() . . . 706
12.306.2.23 blend() . . . 707
12.306.2.24 blendv() . . . 707
12.306.2.25 add() . . . 707
12.306.2.26 addin() . . . 707
12.306.2.27 sub() . . . 707
12.306.2.28 subin() . . . 707
12.306.2.29 mul() . . . 707
12.306.2.30 mulin() . . . 707
12.306.2.31 div() . . . 707
12.306.2.32 fmadd() . . . 708
12.306.2.33 fmaddin() . . . 708
12.306.2.34 fnmadd() . . . 708
12.306.2.35 fnmaddin() . . . 708

12.306.2.36 fmsub() . . . 708
12.306.2.37 fmsubin() . . . 708
12.306.2.38 eq() . . . 708
12.306.2.39 lesser() . . . 708
12.306.2.40 lesser_eq() . . . 708
12.306.2.41 greater() . . . 709
12.306.2.42 greater_eq() . . . 709
12.306.2.43 floor() . . . 709
12.306.2.44 ceil() . . . 709
12.306.2.45 round() . . . 709
12.306.2.46 hadd() . . . 709
12.306.2.47 hadd_to_scal() . . . 709
12.306.3 Field Documentation . . . 709
12.306.3.1 vect_size . . . 709
12.306.3.2 alignment . . . 709
12.307 Simd512_impl< true, true, false, 8 > Struct Reference . . . 709
12.307.1 Member Typedef Documentation . . . 712
12.307.1.1 scalar_t . . . 712
12.307.1.2 aligned_allocator . . . 712
12.307.1.3 aligned_vector . . . 712
12.307.1.4 is_same_element . . . 712
12.307.1.5 simdHalf . . . 712
12.307.1.6 vect_t . . . 712
12.307.1.7 half_t . . . 712
12.307.2 Member Function Documentation . . . 712
12.307.2.1 type_string() . . . 712
12.307.2.2 set1() [1/2] . . . 712
12.307.2.3 set() [1/3] . . . 712
12.307.2.4 gather() [1/2] . . . 713
12.307.2.5 load() [1/2] . . . 713
12.307.2.6 loadu() [1/2] . . . 713
12.307.2.7 store() [1/2] . . . 713
12.307.2.8 maskstore() [1/2] . . . 713
12.307.2.9 storeu() [1/2] . . . 713
12.307.2.10 stream() [1/2] . . . 713
12.307.2.11 sra() . . . 713
12.307.2.12 greater() . . . 713
12.307.2.13 lesser() . . . 713
12.307.2.14 greater_eq() . . . 714
12.307.2.15 lesser_eq() . . . 714
12.307.2.16 mullo() . . . 714
12.307.2.17 mulhi() . . . 714

12.307.2.18 mulx() . . . 714
12.307.2.19 fmaddx() . . . 714
12.307.2.20 fmaddxin() . . . 714
12.307.2.21 fnmaddx() . . . 714
12.307.2.22 fnmaddxin() . . . 714
12.307.2.23 fmsubx() . . . 715
12.307.2.24 fmsubxin() . . . 715
12.307.2.25 hadd_to_scal() . . . 715
12.307.2.26 valid() . . . 715
12.307.2.27 compliant() . . . 715
12.307.2.28 set1() [2/2] . . . 715
12.307.2.29 set() [2/3] . . . 715
12.307.2.30 set() [3/3] . . . 715
12.307.2.31 gather() [2/2] . . . 715
12.307.2.32 load() [2/2] . . . 716
12.307.2.33 loadu() [2/2] . . . 716
12.307.2.34 store() [2/2] . . . 716
12.307.2.35 maskstore() [2/2] . . . 716
12.307.2.36 storeu() [2/2] . . . 716
12.307.2.37 stream() [2/2] . . . 716
12.307.2.38 sll() . . . 716
12.307.2.39 srl() . . . 716
12.307.2.40 shuffle() . . . 716
12.307.2.41 unpacklo_intrinsic() . . . 717
12.307.2.42 unpackhi_intrinsic() . . . 717
12.307.2.43 unpacklo() . . . 717
12.307.2.44 unpackhi() . . . 717
12.307.2.45 unpacklohi() . . . 717
12.307.2.46 pack_even() . . . 717
12.307.2.47 pack_odd() . . . 717
12.307.2.48 pack() . . . 717
12.307.2.49 transpose() . . . 717
12.307.2.50 blend() . . . 718
12.307.2.51 add() . . . 718
12.307.2.52 addin() . . . 718
12.307.2.53 sub() . . . 718
12.307.2.54 subin() . . . 718
12.307.2.55 mul() . . . 718
12.307.2.56 fmadd() . . . 718
12.307.2.57 fmaddin() . . . 718
12.307.2.58 fnmadd() . . . 718
12.307.2.59 fnmaddin() . . . 719

12.307.2.60 fmsub() . . . 719
12.307.2.61 fmsubin() . . . 719
12.307.2.62 eq() . . . 719
12.307.2.63 round() . . . 719
12.307.2.64 mask_high() . . . 719
12.307.2.65 mulhi_fast() . . . 719
12.307.2.66 mod() . . . 719
12.307.2.67 signbits() . . . 720
12.307.2.68 zero() . . . 720
12.307.2.69 vor() . . . 720
12.307.2.70 vxor() . . . 720
12.307.2.71 vand() . . . 720
12.307.2.72 vandnot() . . . 720
12.307.3 Field Documentation . . . 720
12.307.3.1 vect_size . . . 720
12.307.3.2 alignment . . . 720
12.308 Simd512_impl< true, true, true, 8 > Struct Reference . . . 720
12.308.1 Member Typedef Documentation . . . 722
12.308.1.1 vect_t . . . 722
12.308.1.2 half_t . . . 722
12.308.1.3 scalar_t . . . 722
12.308.1.4 simdHalf . . . 723
12.308.1.5 aligned_allocator . . . 723
12.308.1.6 aligned_vector . . . 723
12.308.1.7 is_same_element . . . 723
12.308.2 Member Function Documentation . . . 723
12.308.2.1 type_string() . . . 723
12.308.2.2 valid() . . . 723
12.308.2.3 compliant() . . . 723
12.308.2.4 set1() . . . 723
12.308.2.5 set() [1/2] . . . 723
12.308.2.6 set() [2/2] . . . 723
12.308.2.7 gather() . . . 724
12.308.2.8 load() . . . 724
12.308.2.9 loadu() . . . 724
12.308.2.10 store() . . . 724
12.308.2.11 maskstore() . . . 724
12.308.2.12 storeu() . . . 724
12.308.2.13 stream() . . . 724
12.308.2.14 sll() . . . 724
12.308.2.15 srl() . . . 724
12.308.2.16 sra() . . . 724

12.308.2.17 shuffle() . . . 725
12.308.2.18 unpacklo_intrinsic() . . . 725
12.308.2.19 unpackhi_intrinsic() . . . 725
12.308.2.20 unpacklo() . . . 725
12.308.2.21 unpackhi() . . . 725
12.308.2.22 unpacklohi() . . . 725
12.308.2.23 pack_even() . . . 725
12.308.2.24 pack_odd() . . . 725
12.308.2.25 pack() . . . 725
12.308.2.26 transpose() . . . 726
12.308.2.27 blend() . . . 726
12.308.2.28 add() . . . 726
12.308.2.29 addin() . . . 726
12.308.2.30 sub() . . . 726
12.308.2.31 subin() . . . 726
12.308.2.32 mullo() . . . 726
12.308.2.33 mul() . . . 726
12.308.2.34 mulhi() . . . 726
12.308.2.35 mulx() . . . 727
12.308.2.36 fmadd() . . . 727
12.308.2.37 fmaddin() . . . 727
12.308.2.38 fmaddx() . . . 727
12.308.2.39 fmaddxin() . . . 727
12.308.2.40 fnmadd() . . . 727
12.308.2.41 fnmaddin() . . . 727
12.308.2.42 fnmaddx() . . . 727
12.308.2.43 fnmaddxin() . . . 727
12.308.2.44 fmsub() . . . 728
12.308.2.45 fmsubin() . . . 728
12.308.2.46 fmsubx() . . . 728
12.308.2.47 fmsubxin() . . . 728
12.308.2.48 eq() . . . 728
12.308.2.49 greater() . . . 728
12.308.2.50 lesser() . . . 728
12.308.2.51 greater_eq() . . . 728
12.308.2.52 lesser_eq() . . . 728
12.308.2.53 hadd_to_scal() . . . 729
12.308.2.54 round() . . . 729
12.308.2.55 mask_high() . . . 729
12.308.2.56 mulhi_fast() . . . 729
12.308.2.57 mod() . . . 729
12.308.2.58 signbits() . . . 729

12.308.2.59 zero() . . . 729
12.308.2.60 vor() . . . 729
12.308.2.61 vxor() . . . 729
12.308.2.62 vand() . . . 729
12.308.2.63 vandnot() . . . 730
12.308.3 Field Documentation . . . 730
12.308.3.1 vect_size . . . 730
12.308.3.2 alignment . . . 730
12.309 Simd512i_base Struct Reference . . . 730
12.309.1 Member Typedef Documentation . . . 730
12.309.1.1 vect_t . . . 730
12.309.2 Member Function Documentation . . . 730
12.309.2.1 zero() . . . 730
12.309.2.2 vor() . . . 730
12.309.2.3 vxor() . . . 731
12.309.2.4 vand() . . . 731
12.309.2.5 vandnot() . . . 731
12.310 SimdChooser< T, bool, bool > Struct Template Reference . . . 731
12.311 SimdChooser< T, false, b > Struct Template Reference . . . 731
12.311.1 Member Typedef Documentation . . . 731
12.311.1.1 value . . . 731
12.312 SimdChooser< T, true, false > Struct Template Reference . . . 731
12.312.1 Member Typedef Documentation . . . 732
12.312.1.1 value . . . 732
12.313 SimdChooser< T, true, true > Struct Template Reference . . . 732
12.313.1 Member Typedef Documentation . . . 732
12.313.1.1 value . . . 732
12.314 simdToType< T > Struct Template Reference . . . 732
12.315 Single Struct Reference . . . 732
12.316 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference . . . 732
12.317 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference . . . 732
12.317.1 Member Typedef Documentation . . . 733
12.317.1.1 Field . . . 733
12.317.2 Field Documentation . . . 733
12.317.2.1 col . . . 733
12.317.2.2 row . . . 733
12.317.2.3 dat . . . 733
12.317.2.4 delayed . . . 733
12.317.2.5 kmax . . . 733
12.317.2.6 m . . . 733
12.317.2.7 n . . . 733
12.317.2.8 nnz . . . 734

12.317.2.9 nElements . . . 734
12.317.2.10 maxrow . . . 734
12.318 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference . . . 734
12.318.1 Member Typedef Documentation . . . 734
12.318.1.1 Field . . . 734
12.318.2 Field Documentation . . . 735
12.318.2.1 cst . . . 735
12.318.2.2 col . . . 735
12.318.2.3 row . . . 735
12.318.2.4 dat . . . 735
12.318.2.5 delayed . . . 735
12.318.2.6 kmax . . . 735
12.318.2.7 m . . . 735
12.318.2.8 n . . . 735
12.318.2.9 nnz . . . 735
12.318.2.10 nElements . . . 735
12.318.2.11 maxrow . . . 735
12.319 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference . . . 736
12.319.1 Member Typedef Documentation . . . 736
12.319.1.1 Field . . . 736
12.319.2 Field Documentation . . . 736
12.319.2.1 delayed . . . 736
12.319.2.2 kmax . . . 736
12.319.2.3 m . . . 736
12.319.2.4 n . . . 736
12.319.2.5 nnz . . . 737
12.319.2.6 nElements . . . 737
12.319.2.7 maxrow . . . 737
12.319.2.8 col . . . 737
12.319.2.9 st . . . 737
12.319.2.10 stend . . . 737
12.319.2.11 dat . . . 737
12.320 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference . . . 737
12.320.1 Member Typedef Documentation . . . 738
12.320.1.1 Field . . . 738
12.320.2 Field Documentation . . . 738
12.320.2.1 delayed . . . 738
12.320.2.2 col . . . 738
12.320.2.3 st . . . 738
12.320.2.4 dat . . . 738
12.320.2.5 kmax . . . 738
12.320.2.6 m . . . 738

12.320.2.7 n . . . 738
12.320.2.8 nnz . . . 738
12.320.2.9 nElements . . . 738
12.320.2.10 maxrow . . . 739
12.320.2.11 nOnes . . . 739
12.320.2.12 nMOnes . . . 739
12.320.2.13 nOthers . . . 739
12.321 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference . . . 739
12.321.1 Member Typedef Documentation . . . 740
12.321.1.1 Field . . . 740
12.321.2 Field Documentation . . . 740
12.321.2.1 cst . . . 740
12.321.2.2 delayed . . . 740
12.321.2.3 kmax . . . 740
12.321.2.4 m . . . 740
12.321.2.5 n . . . 740
12.321.2.6 nnz . . . 740
12.321.2.7 nElements . . . 740
12.321.2.8 maxrow . . . 740
12.321.2.9 col . . . 740
12.321.2.10 st . . . 740
12.321.2.11 stend . . . 741
12.321.2.12 dat . . . 741
12.322 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference . . . 741
12.322.1 Member Typedef Documentation . . . 741
12.322.1.1 Field . . . 741
12.322.2 Field Documentation . . . 741
12.322.2.1 delayed . . . 741
12.322.2.2 kmax . . . 742
12.322.2.3 m . . . 742
12.322.2.4 n . . . 742
12.322.2.5 ld . . . 742
12.322.2.6 nnz . . . 742
12.322.2.7 nElements . . . 742
12.322.2.8 maxrow . . . 742
12.322.2.9 col . . . 742
12.322.2.10 dat . . . 742
12.323 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference . . . 742
12.323.1 Field Documentation . . . 743
12.323.1.1 delayed . . . 743
12.323.1.2 chunk . . . 743
12.323.1.3 m . . . 743

12.323.1.4 n . . . 743
12.323.1.5 ld . . . 743
12.323.1.6 kmax . . . 743
12.323.1.7 nnz . . . 743
12.323.1.8 nElements . . . 743
12.323.1.9 maxrow . . . 744
12.323.1.10 nChunks . . . 744
12.323.1.11 col . . . 744
12.323.1.12 dat . . . 744
12.324 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference . . . 744
12.324.1 Field Documentation . . . 744
12.324.1.1 cst . . . 744
12.324.1.2 delayed . . . 745
12.324.1.3 chunk . . . 745
12.324.1.4 m . . . 745
12.324.1.5 n . . . 745
12.324.1.6 ld . . . 745
12.324.1.7 kmax . . . 745
12.324.1.8 nnz . . . 745
12.324.1.9 nElements . . . 745
12.324.1.10 maxrow . . . 745
12.324.1.11 nChunks . . . 745
12.324.1.12 col . . . 745
12.324.1.13 dat . . . 745
12.325 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference . . . 746
12.325.1 Member Typedef Documentation . . . 746
12.325.1.1 Field . . . 746
12.325.2 Field Documentation . . . 746
12.325.2.1 cst . . . 746
12.325.2.2 delayed . . . 746
12.325.2.3 kmax . . . 746
12.325.2.4 m . . . 746
12.325.2.5 n . . . 747
12.325.2.6 ld . . . 747
12.325.2.7 nnz . . . 747
12.325.2.8 nElements . . . 747
12.325.2.9 maxrow . . . 747
12.325.2.10 col . . . 747
12.325.2.11 dat . . . 747
12.326 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference . . . 747
12.326.1 Member Typedef Documentation . . . 748
12.326.1.1 Field . . . 748

12.326.2 Field Documentation . . . 748
12.326.2.1 delayed . . . 748
12.326.2.2 chunk . . . 748
12.326.2.3 kmax . . . 748
12.326.2.4 m . . . 748
12.326.2.5 n . . . 748
12.326.2.6 maxrow . . . 748
12.326.2.7 sigma . . . 749
12.326.2.8 nChunks . . . 749
12.326.2.9 nnz . . . 749
12.326.2.10 nElements . . . 749
12.326.2.11 perm . . . 749
12.326.2.12 st . . . 749
12.326.2.13 chunkSize . . . 749
12.326.2.14 col . . . 749
12.326.2.15 dat . . . 749
12.327 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference . . . 749
12.327.1 Member Typedef Documentation . . . 750
12.327.1.1 Field . . . 750
12.327.2 Field Documentation . . . 750
12.327.2.1 cst . . . 750
12.327.2.2 delayed . . . 750
12.327.2.3 chunk . . . 750
12.327.2.4 kmax . . . 750
12.327.2.5 m . . . 750
12.327.2.6 n . . . 751
12.327.2.7 maxrow . . . 751
12.327.2.8 sigma . . . 751
12.327.2.9 nChunks . . . 751
12.327.2.10 nnz . . . 751
12.327.2.11 nElements . . . 751
12.327.2.12 perm . . . 751
12.327.2.13 st . . . 751
12.327.2.14 chunkSize . . . 751
12.327.2.15 col . . . 751
12.327.2.16 dat . . . 751
12.328 SpMat< Field, flag > Struct Template Reference . . . 751
12.328.1 Field Documentation . . . 752
12.328.1.1 _coo . . . 752
12.328.1.2 _csr . . . 752
12.328.1.3 _ell . . . 752
12.329 StatsMatrix Struct Reference . . . 752

12.329.1 Field Documentation . . . 753
12.329.1.1 rowdim . . . 753
12.329.1.2 coldim . . . 753
12.329.1.3 nOnes . . . 753
12.329.1.4 nMOnes . . . 753
12.329.1.5 nOthers . . . 753
12.329.1.6 nnz . . . 753
12.329.1.7 maxRow . . . 753
12.329.1.8 minRow . . . 753
12.329.1.9 averageRow . . . 753
12.329.1.10 deviationRow . . . 753
12.329.1.11 maxCol . . . 753
12.329.1.12 minCol . . . 753
12.329.1.13 averageCol . . . 753
12.329.1.14 deviationCol . . . 754
12.329.1.15 minColDifference . . . 754
12.329.1.16 maxColDifference . . . 754
12.329.1.17 averageColDifference . . . 754
12.329.1.18 deviationColDifference . . . 754
12.329.1.19 minRowDifference . . . 754
12.329.1.20 maxRowDifference . . . 754
12.329.1.21 averageRowDifference . . . 754
12.329.1.22 deviationRowDifference . . . 754
12.329.1.23 nDenseRows . . . 754
12.329.1.24 nDenseCols . . . 754
12.329.1.25 nEmptyRows . . . 754
12.329.1.26 nEmptyCols . . . 754
12.329.1.27 nEmptyColsEnd . . . 754
12.329.1.28 denseRows . . . 754
12.329.1.29 denseCols . . . 755
12.330 string Class Reference . . . 755
12.330.1 Detailed Description . . . 755
12.331 support_fast_mod< T > Struct Template Reference . . . 755
12.332 support_fast_mod< double > Struct Reference . . . 755
12.333 support_fast_mod< float > Struct Reference . . . 756
12.334 support_fast_mod< int64_t > Struct Reference . . . 756
12.335 support_simd< T > Struct Template Reference . . . 756
12.336 support_simd_add< T > Struct Template Reference . . . 756
12.337 support_simd_mod< T > Struct Template Reference . . . 757
12.338 Test< Elt > Class Template Reference . . . 757
12.338.1 Member Typedef Documentation . . . 758
12.338.1.1 Field . . . 758

12.338.1.2 Elt_ptr . . . 758
12.338.1.3 Residu . . . 758
12.338.1.4 enable_if_t . . . 758
12.338.1.5 is_same_element . . . 758
12.338.1.6 enable_if_no_simd_t . . . 758
12.338.1.7 enable_if_simd128_t . . . 758
12.338.1.8 enable_if_simd256_t . . . 758
12.338.1.9 enable_if_simd512_t . . . 758
12.338.2 Constructor & Destructor Documentation . . . 759
12.338.2.1 Test() . . . 759
12.338.3 Member Function Documentation . . . 759
12.338.3.1 cardinality() [1/2] . . . 759
12.338.3.2 cardinality() [2/2] . . . 759
12.338.3.3 test_ftranspose() . . . 759
12.338.3.4 doTests() . . . 759
12.338.3.5 run() . . . 759
12.338.4 Field Documentation . . . 759
12.338.4.1 F . . . 759
12.338.4.2 _mm . . . 759
12.338.4.3 _nn . . . 760
12.339 TestOneMethod< Simd > Class Template Reference . . . 760
12.339.1 Member Typedef Documentation . . . 761
12.339.1.1 Element . . . 761
12.339.1.2 vect_t . . . 761
12.339.1.3 vectElt . . . 761
12.339.1.4 enable_if_t . . . 761
12.339.2 Constructor & Destructor Documentation . . . 761
12.339.2.1 TestOneMethod() . . . 761
12.339.3 Member Function Documentation . . . 761
12.339.3.1 evaluate_scalar_method() [1/3] . . . 761
12.339.3.2 evaluate_scalar_method() [2/3] . . . 761
12.339.3.3 evaluate_scalar_method() [3/3] . . . 761
12.339.3.4 evaluate_simd_method() [1/2] . . . 762
12.339.3.5 evaluate_simd_method() [2/2] . . . 762
12.339.3.6 getStatus() . . . 762
12.339.3.7 getTestName() . . . 762
12.339.3.8 writeResultLine() . . . 762
12.339.3.9 writeDebugData() . . . 762
12.339.4 Field Documentation . . . 762
12.339.4.1 vect_size . . . 762
12.339.4.2 nb_lref . . . 762
12.339.4.3 name . . . 762

12.339.4.4 inputs . . . 762
12.339.4.5 outputs_simd . . . 763
12.339.4.6 outputs_scalar . . . 763
12.340 tfn_minus Struct Reference . . . 763
12.340.1 Member Function Documentation . . . 763
12.340.1.1 operator()() . . . 763
12.341 tfn_minus_eq Struct Reference . . . 763
12.341.1 Member Function Documentation . . . 763
12.341.1.1 operator()() . . . 763
12.342 tfn_mul Struct Reference . . . 763
12.342.1 Member Function Documentation . . . 764
12.342.1.1 operator()() . . . 764
12.343 tfn_mul_eq Struct Reference . . . 764
12.343.1 Member Function Documentation . . . 764
12.343.1.1 operator()() . . . 764
12.344 tfn_plus Struct Reference . . . 764
12.344.1 Member Function Documentation . . . 764
12.344.1.1 operator()() . . . 764
12.345 tfn_plus_eq Struct Reference . . . 765
12.345.1 Member Function Documentation . . . 765
12.345.1.1 operator()() . . . 765
12.346 Threads Struct Reference . . . 765
12.347 ThreeD Struct Reference . . . 765
12.348 ThreeDAdaptive Struct Reference . . . 765
12.349 ThreeDInPlace Struct Reference . . . 765
12.350 TRSMHelper< RecIterTrait, ParSeqTrait > Struct Template Reference . . . 765
12.350.1 Detailed Description . . . 766
12.350.2 Constructor & Destructor Documentation . . . 766
12.350.2.1 TRSMHelper() [1/3] . . . 766
12.350.2.2 TRSMHelper() [2/3] . . . 766
12.350.2.3 TRSMHelper() [3/3] . . . 766
12.350.3 Member Function Documentation . . . 766
12.350.3.1 pMMH() [1/2] . . . 766
12.350.3.2 pMMH() [2/2] . . . 767
12.350.4 Field Documentation . . . 767
12.350.4.1 parseq . . . 767
12.351 TwoD Struct Reference . . . 767
12.352 TwoDAdaptive Struct Reference . . . 767
12.353 UnparametricTag Struct Reference . . . 767
12.353.1 Detailed Description . . . 767
12.354 vector< T > Class Template Reference . . . 767
12.354.1 Detailed Description . . . 768

12.354.2 Field Documentation . . . 768
12.354.2.1 elements . . . 768
12.355 width< T > Struct Template Reference . . . 768
12.355.1 Field Documentation . . . 768
12.355.1.1 value . . . 768
12.356 width< double > Struct Reference . . . 768
12.356.1 Field Documentation . . . 768
12.356.1.1 value [1/2] . . . 768
12.356.1.2 value [2/2] . . . 768
12.357 width< float > Struct Reference . . . 769
12.357.1 Field Documentation . . . 769
12.357.1.1 value [1/2] . . . 769
12.357.1.2 value [2/2] . . . 769
12.358 Winograd Struct Reference . . . 769
12.359 WinogradPar Struct Reference . . . 769

13 File Documentation **771**

13.1 arithprog.C File Reference . . . 771
13.1.1 Typedef Documentation . . . 771
13.1.1.1 TTimer . . . 771
13.1.2 Function Documentation . . . 771
13.1.2.1 main() . . . 771
13.2 autotune/charpoly.C File Reference . . . 771
13.2.1 Macro Definition Documentation . . . 772
13.2.1.1 CUBE . . . 772
13.2.1.2 GFOPS . . . 772
13.2.2 Typedef Documentation . . . 772
13.2.2.1 TTimer . . . 772
13.2.3 Function Documentation . . . 772
13.2.3.1 main() . . . 772
13.3 examples/charpoly.C File Reference . . . 772
13.3.1 Function Documentation . . . 772
13.3.1.1 main() . . . 772
13.4 fsyrk.C File Reference . . . 773
13.4.1 Macro Definition Documentation . . . 773
13.4.1.1 CUBE . . . 773
13.4.1.2 GFOPS . . . 773
13.4.2 Function Documentation . . . 773
13.4.2.1 main() . . . 773
13.5 fsytrf.C File Reference . . . 773
13.5.1 Macro Definition Documentation . . . 774
13.5.1.1 CUBE . . . 774

13.5.1.2 GFOPS . . . 774
13.5.2 Function Documentation . . . 774
13.5.2.1 main() . . . 774
13.6 ftrtri.C File Reference . . . 774
13.6.1 Macro Definition Documentation . . . 774
13.6.1.1 CUBE . . . 774
13.6.1.2 GFOPS . . . 775
13.6.2 Function Documentation . . . 775
13.6.2.1 main() . . . 775
13.7 autotune/pluq.C File Reference . . . 775
13.7.1 Macro Definition Documentation . . . 775
13.7.1.1 CUBE . . . 775
13.7.1.2 GFOPS . . . 775
13.7.2 Function Documentation . . . 776
13.7.2.1 main() . . . 776
13.8 examples/pluq.C File Reference . . . 776
13.8.1 Function Documentation . . . 776
13.8.1.1 main() . . . 776
13.9 winograd.C File Reference . . . 776
13.9.1 Macro Definition Documentation . . . 776
13.9.1.1 DOUBLE_TO_FLOAT_CROSSOVER . . . 776
13.9.1.2 GFOPS . . . 777
13.9.2 Function Documentation . . . 777
13.9.2.1 balanced() [1/2] . . . 777
13.9.2.2 balanced() [2/2] . . . 777
13.9.2.3 main() . . . 777
13.10 benchmark-charpoly-mp.C File Reference . . . 777
13.10.1 Macro Definition Documentation . . . 777
13.10.1.1 __FFLASFFPACK_FORCE_SEQ . . . 777
13.10.2 Function Documentation . . . 777
13.10.2.1 main() . . . 777
13.11 benchmark-charpoly.C File Reference . . . 778
13.11.1 Macro Definition Documentation . . . 778
13.11.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 778
13.11.2 Function Documentation . . . 778
13.11.2.1 run_with_field() . . . 778
13.11.2.2 main() . . . 778
13.12 benchmark-checkers.C File Reference . . . 778
13.12.1 Macro Definition Documentation . . . 779
13.12.1.1 ENABLE_ALL_CHECKINGS . . . 779
13.12.1.2 _NR_TESTS . . . 779
13.12.1.3 _MAX_SIZE_MATRICES . . . 779

13.12.1.4 CUBE . . . 779
13.12.2 Function Documentation . . . 779
13.12.2.1 main() . . . 779
13.13 benchmark-dgemm.C File Reference . . . 779
13.13.1 Macro Definition Documentation . . . 780
13.13.1.1 CBLAS_GEMM . . . 780
13.13.2 Typedef Documentation . . . 780
13.13.2.1 TTimer . . . 780
13.13.2.2 Floats . . . 780
13.13.3 Function Documentation . . . 780
13.13.3.1 main() . . . 780
13.14 benchmark-dgetrf.C File Reference . . . 780
13.14.1 Macro Definition Documentation . . . 781
13.14.1.1 __FFLASFFPACK_HAVE_DGETRF . . . 781
13.14.2 Function Documentation . . . 781
13.14.2.1 main() . . . 781
13.15 benchmark-dgetri.C File Reference . . . 781
13.15.1 Function Documentation . . . 781
13.15.1.1 main() . . . 781
13.16 benchmark-dsytrf.C File Reference . . . 781
13.16.1 Macro Definition Documentation . . . 782
13.16.1.1 EFFGFF . . . 782
13.16.2 Function Documentation . . . 782
13.16.2.1 main() . . . 782
13.17 benchmark-dtrsm.C File Reference . . . 782
13.17.1 Function Documentation . . . 782
13.17.1.1 main() . . . 782
13.18 benchmark-dtrtri.C File Reference . . . 782
13.18.1 Macro Definition Documentation . . . 783
13.18.1.1 __FFLASFFPACK_HAVE_DTRTRI . . . 783
13.18.2 Function Documentation . . . 783
13.18.2.1 main() . . . 783
13.19 benchmark-fadd-lvl2.C File Reference . . . 783
13.19.1 Macro Definition Documentation . . . 783
13.19.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 783
13.19.2 Function Documentation . . . 783
13.19.2.1 main() . . . 783
13.20 benchmark-fdot.C File Reference . . . 783
13.20.1 Macro Definition Documentation . . . 784
13.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 784
13.20.2 Function Documentation . . . 784
13.20.2.1 run_with_field() . . . 784

13.20.2.2 main() . . . 784
13.21 benchmark-fgemm-mp.C File Reference . . . 784
13.21.1 Macro Definition Documentation . . . 785
13.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 785
13.21.1.2 MG_DEFAULT . . . 785
13.21.1.3 STD_RECINT_SIZE . . . 785
13.21.2 Function Documentation . . . 785
13.21.2.1 tmain() . . . 785
13.21.2.2 main() . . . 785
13.22 benchmark-fgemm-rns.C File Reference . . . 785
13.22.1 Macro Definition Documentation . . . 786
13.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 786
13.22.2 Typedef Documentation . . . 786
13.22.2.1 RNS . . . 786
13.22.2.2 Field . . . 786
13.22.2.3 Element_ptr . . . 786
13.22.2.4 ConstElement_ptr . . . 786
13.22.2.5 THREADS . . . 786
13.22.2.6 GRAIN . . . 786
13.22.2.7 TWOD . . . 786
13.22.2.8 TWODA . . . 786
13.22.2.9 THREED . . . 786
13.22.2.10 THREEDA . . . 786
13.22.2.11 THREEDIP . . . 787
13.22.2.12 PSeq . . . 787
13.22.3 Function Documentation . . . 787
13.22.3.1 main() . . . 787
13.23 benchmark-fgemm.C File Reference . . . 787
13.23.1 Macro Definition Documentation . . . 787
13.23.1.1 CLASSIC_HYBRID . . . 787
13.23.2 Function Documentation . . . 787
13.23.2.1 main() . . . 787
13.24 benchmark-fgemv-mp.C File Reference . . . 787
13.24.1 Macro Definition Documentation . . . 788
13.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 788
13.24.1.2 MG_DEFAULT . . . 788
13.24.1.3 STD_RECINT_SIZE . . . 788
13.24.2 Function Documentation . . . 788
13.24.2.1 write_matrix() . . . 788
13.24.2.2 tmain() . . . 788
13.24.2.3 main() . . . 788
13.25 benchmark-fgemv.C File Reference . . . 789

13.25.1 Macro Definition Documentation . . . 789
13.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 789
13.25.2 Function Documentation . . . 790
13.25.2.1 fill_value() . . . 790
13.25.2.2 genData() . . . 790
13.25.2.3 check_result() . . . 790
13.25.2.4 benchmark_with_timer() . . . 790
13.25.2.5 benchmark_disp() . . . 791
13.25.2.6 benchmark_in_Field() . . . 791
13.25.2.7 benchmark_with_field() [1/2] . . . 791
13.25.2.8 benchmark_with_field() [2/2] . . . 791
13.25.2.9 main() . . . 792
13.26 benchmark-fgesv.C File Reference . . . 792
13.26.1 Macro Definition Documentation . . . 792
13.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 792
13.26.2 Function Documentation . . . 792
13.26.2.1 main() . . . 792
13.27 benchmark-fsyr2k.C File Reference . . . 792
13.27.1 Function Documentation . . . 793
13.27.1.1 main() . . . 793
13.28 benchmark-fsyrk.C File Reference . . . 793
13.28.1 Macro Definition Documentation . . . 793
13.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 793
13.28.2 Function Documentation . . . 793
13.28.2.1 main() . . . 793
13.29 benchmark-fsytrf.C File Reference . . . 793
13.29.1 Macro Definition Documentation . . . 794
13.29.1.1 __FFPACK_FSYTRF_BC_CROUT . . . 794
13.29.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 794
13.29.1.3 CUBE . . . 794
13.29.2 Function Documentation . . . 794
13.29.2.1 main() . . . 794
13.30 benchmark-ftrsm-mp.C File Reference . . . 794
13.30.1 Macro Definition Documentation . . . 794
13.30.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 794
13.30.2 Function Documentation . . . 795
13.30.2.1 main() . . . 795
13.31 benchmark-ftrsm.C File Reference . . . 795
13.31.1 Macro Definition Documentation . . . 795
13.31.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 795
13.31.2 Function Documentation . . . 795
13.31.2.1 main() . . . 795

13.32 benchmark-ftrsv.C File Reference . . . 795
13.32.1 Macro Definition Documentation . . . 796
13.32.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 796
13.32.2 Function Documentation . . . 796
13.32.2.1 main() . . . 796
13.33 benchmark-ftrtri.C File Reference . . . 796
13.33.1 Macro Definition Documentation . . . 796
13.33.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 796
13.33.1.2 CUBE . . . 796
13.33.2 Function Documentation . . . 796
13.33.2.1 main() . . . 796
13.34 benchmark-inverse.C File Reference . . . 796
13.34.1 Macro Definition Documentation . . . 797
13.34.1.1 CUBE . . . 797
13.34.2 Function Documentation . . . 797
13.34.2.1 main() . . . 797
13.35 benchmark-lqup-mp.C File Reference . . . 797
13.35.1 Function Documentation . . . 797
13.35.1.1 main() . . . 797
13.36 benchmark-lqup.C File Reference . . . 798
13.36.1 Macro Definition Documentation . . . 798
13.36.1.1 CUBE . . . 798
13.36.2 Function Documentation . . . 798
13.36.2.1 main() . . . 798
13.37 benchmark-pluq.C File Reference . . . 798
13.37.1 Macro Definition Documentation . . . 799
13.37.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 799
13.37.1.2 CUBE . . . 799
13.37.2 Typedef Documentation . . . 799
13.37.2.1 Field . . . 799
13.37.3 Function Documentation . . . 799
13.37.3.1 verification_PLUQ() . . . 799
13.37.3.2 Rec_Initialize() . . . 799
13.37.3.3 main() . . . 799
13.38 benchmark-quasisep.C File Reference . . . 799
13.38.1 Macro Definition Documentation . . . 800
13.38.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . 800
13.38.2 Function Documentation . . . 800
13.38.2.1 run_with_field() . . . 800
13.38.2.2 main() . . . 800
13.39 benchmark-storage-transpose.C File Reference . . . 800
13.39.1 Function Documentation . . . 801

13.39.1.1 main() . . . 801
13.40 benchmark-wino.C File Reference . . . 801
13.40.1 Macro Definition Documentation . . . 801
13.40.1.1 CUBE . . . 801
13.40.2 Function Documentation . . . 801
13.40.2.1 launch_wino() . . . 801
13.40.2.2 main() . . . 802
13.41 mainpage.doxy File Reference . . . 802
13.42 det.C File Reference . . . 802
13.42.1 Function Documentation . . . 802
13.42.1.1 main() . . . 802
13.43 matmul.C File Reference . . . 802
13.43.1 Function Documentation . . . 802
13.43.1.1 main() . . . 802
13.44 rank.C File Reference . . . 803
13.44.1 Function Documentation . . . 803
13.44.1.1 main() . . . 803
13.45 solve.C File Reference . . . 803
13.45.1 Function Documentation . . . 803
13.45.1.1 main() . . . 803
13.46 checker_charpoly.inl File Reference . . . 803
13.46.1 Macro Definition Documentation . . . 804
13.46.1.1 __FFLASFFPACK_checker_charpoly_INL . . . 804
13.47 checker_det.inl File Reference . . . 804
13.47.1 Macro Definition Documentation . . . 804
13.47.1.1 __FFLASFFPACK_checker_det_INL . . . 804
13.48 checker_empty.h File Reference . . . 804
13.49 checker_fgemm.inl File Reference . . . 804
13.49.1 Macro Definition Documentation . . . 805
13.49.1.1 __FFLASFFPACK_checker_fgemm_INL . . . 805
13.50 checker_ftrsm.inl File Reference . . . 805
13.50.1 Macro Definition Documentation . . . 805
13.50.1.1 __FFLASFFPACK_checker_ftrsm_INL . . . 805
13.51 checker_invert.inl File Reference . . . 805
13.51.1 Macro Definition Documentation . . . 805
13.51.1.1 __FFLASFFPACK_checker_invert_INL . . . 805
13.52 checker_pluq.inl File Reference . . . 805
13.52.1 Macro Definition Documentation . . . 806
13.52.1.1 __FFLASFFPACK_checker_pluq_INL . . . 806
13.53 checkers.doxy File Reference . . . 806
13.54 checkers_fflas.h File Reference . . . 806
13.55 checkers_fflas.inl File Reference . . . 806

13.55.1 Macro Definition Documentation . . . 807
13.55.1.1 FFLASFFPACK_checkers_fflas_inl_H . . . 807
13.56 checkers_ffpack.h File Reference . . . 807
13.57 checkers_ffpack.inl File Reference . . . 807
13.57.1 Macro Definition Documentation . . . 808
13.57.1.1 FFLASFFPACK_checkers_ffpack_inl_H . . . 808
13.58 config-blas.h File Reference . . . 808
13.58.1 Macro Definition Documentation . . . 809
13.58.1.1 CBLAS_INT . . . 809
13.58.1.2 CBLAS_ENUM_DEFINED_H . . . 809
13.58.1.3 CBLAS_EXTERNALS . . . 809
13.58.1.4 blas_enum . . . 809
13.58.2 Enumeration Type Documentation . . . 809
13.58.2.1 CBLAS_ORDER . . . 809
13.58.2.2 CBLAS_TRANSPOSE . . . 809
13.58.2.3 CBLAS_UPLO . . . 810
13.58.2.4 CBLAS_DIAG . . . 810
13.58.2.5 CBLAS_SIDE . . . 810
13.58.3 Function Documentation . . . 810
13.58.3.1 daxpy_() . . . 810
13.58.3.2 saxpy_() . . . 810
13.58.3.3 ddot_() . . . 811
13.58.3.4 sdot_() . . . 811
13.58.3.5 dasum_() . . . 811
13.58.3.6 idamax_() . . . 811
13.58.3.7 dnrm2_() . . . 811
13.58.3.8 dgemv_() . . . 811
13.58.3.9 sgemv_() . . . 811
13.58.3.10 dger_() . . . 812
13.58.3.11 sger_() . . . 812
13.58.3.12 dcopy_() . . . 812
13.58.3.13 scopy_() . . . 812
13.58.3.14 dscal_() . . . 812
13.58.3.15 sscal_() . . . 813
13.58.3.16 dtrsm_() . . . 813
13.58.3.17 strsm_() . . . 813
13.58.3.18 dtrmm_() . . . 813
13.58.3.19 strmm_() . . . 813
13.58.3.20 sgemm_() . . . 814
13.58.3.21 dgemm_() . . . 814
13.58.3.22 cblas_dsyrk() . . . 814
13.59 config.h File Reference . . . 815

13.59.1 Macro Definition Documentation . . . 815
13.59.1.1 HAVE_BLAS . . . 815
13.59.1.2 HAVE_CBLAS . . . 815
13.59.1.3 HAVE_CXX11 . . . 816
13.59.1.4 HAVE_DLFCN_H . . . 816
13.59.1.5 HAVE_FLOAT_H . . . 816
13.59.1.6 HAVE_INT128 . . . 816
13.59.1.7 HAVE_INTTYPES_H . . . 816
13.59.1.8 HAVE_LAPACK . . . 816
13.59.1.9 HAVE_LIMITS_H . . . 816
13.59.1.10 HAVE_LITTLE_ENDIAN . . . 816
13.59.1.11 HAVE_PTHREAD_H . . . 816
13.59.1.12 HAVE_STDDEF_H . . . 816
13.59.1.13 HAVE_STDINT_H . . . 816
13.59.1.14 HAVE_STDIO_H . . . 816
13.59.1.15 HAVE_STDLIB_H . . . 816
13.59.1.16 HAVE_STRINGS_H . . . 816
13.59.1.17 HAVE_STRING_H . . . 816
13.59.1.18 HAVE_SYS_STAT_H . . . 817
13.59.1.19 HAVE_SYS_TIME_H . . . 817
13.59.1.20 HAVE_SYS_TYPES_H . . . 817
13.59.1.21 HAVE_UNISTD_H . . . 817
13.59.1.22 LT_OBJDIR . . . 817
13.59.1.23 OPENBLAS_NUM_THREADS . . . 817
13.59.1.24 PACKAGE . . . 817
13.59.1.25 PACKAGE_BUGREPORT . . . 817
13.59.1.26 PACKAGE_NAME . . . 817
13.59.1.27 PACKAGE_STRING . . . 817
13.59.1.28 PACKAGE_TARNAME . . . 817
13.59.1.29 PACKAGE_URL . . . 817
13.59.1.30 PACKAGE_VERSION . . . 817
13.59.1.31 SIZEOF_CHAR . . . 817
13.59.1.32 SIZEOF_INT . . . 817
13.59.1.33 SIZEOF_LONG . . . 818
13.59.1.34 SIZEOF_LONG_LONG . . . 818
13.59.1.35 SIZEOF_SHORT . . . 818
13.59.1.36 SIZEOF___INT64_T . . . 818
13.59.1.37 STDC_HEADERS . . . 818
13.59.1.38 USE_OPENMP . . . 818
13.59.1.39 VERSION . . . 818
13.60 fflas-ffpack/config.h File Reference . . . 818
13.60.1 Macro Definition Documentation . . . 819

13.60.1.1 __FFLASFFPACK_HAVE_BLAS . . . 819
13.60.1.2 __FFLASFFPACK_HAVE_CBLAS . . . 819
13.60.1.3 __FFLASFFPACK_HAVE_CXX11 . . . 819
13.60.1.4 __FFLASFFPACK_HAVE_DLFCN_H . . . 819
13.60.1.5 __FFLASFFPACK_HAVE_FLOAT_H . . . 819
13.60.1.6 __FFLASFFPACK_HAVE_INT128 . . . 819
13.60.1.7 __FFLASFFPACK_HAVE_INTTYPES_H . . . 819
13.60.1.8 __FFLASFFPACK_HAVE_LAPACK . . . 819
13.60.1.9 __FFLASFFPACK_HAVE_LIMITS_H . . . 819
13.60.1.10 __FFLASFFPACK_HAVE_LITTLE_ENDIAN . . . 819
13.60.1.11 __FFLASFFPACK_HAVE_PTHREAD_H . . . 819
13.60.1.12 __FFLASFFPACK_HAVE_STDDEF_H . . . 820
13.60.1.13 __FFLASFFPACK_HAVE_STDINT_H . . . 820
13.60.1.14 __FFLASFFPACK_HAVE_STDIO_H . . . 820
13.60.1.15 __FFLASFFPACK_HAVE_STDLIB_H . . . 820
13.60.1.16 __FFLASFFPACK_HAVE_STRINGS_H . . . 820
13.60.1.17 __FFLASFFPACK_HAVE_STRING_H . . . 820
13.60.1.18 __FFLASFFPACK_HAVE_SYS_STAT_H . . . 820
13.60.1.19 __FFLASFFPACK_HAVE_SYS_TIME_H . . . 820
13.60.1.20 __FFLASFFPACK_HAVE_SYS_TYPES_H . . . 820
13.60.1.21 __FFLASFFPACK_HAVE_UNISTD_H . . . 820
13.60.1.22 __FFLASFFPACK_LT_OBJDIR . . . 820
13.60.1.23 __FFLASFFPACK_OPENBLAS_NUM_THREADS . . . 820
13.60.1.24 __FFLASFFPACK_PACKAGE . . . 820
13.60.1.25 __FFLASFFPACK_PACKAGE_BUGREPORT . . . 820
13.60.1.26 __FFLASFFPACK_PACKAGE_NAME . . . 820
13.60.1.27 __FFLASFFPACK_PACKAGE_STRING . . . 821
13.60.1.28 __FFLASFFPACK_PACKAGE_TARNAME . . . 821
13.60.1.29 __FFLASFFPACK_PACKAGE_URL . . . 821
13.60.1.30 __FFLASFFPACK_PACKAGE_VERSION . . . 821
13.60.1.31 __FFLASFFPACK_SIZEOF_CHAR . . . 821
13.60.1.32 __FFLASFFPACK_SIZEOF_INT . . . 821
13.60.1.33 __FFLASFFPACK_SIZEOF_LONG . . . 821
13.60.1.34 __FFLASFFPACK_SIZEOF_LONG_LONG . . . 821
13.60.1.35 __FFLASFFPACK_SIZEOF_SHORT . . . 821
13.60.1.36 __FFLASFFPACK_SIZEOF___INT64_T . . . 821
13.60.1.37 __FFLASFFPACK_STDC_HEADERS . . . 821
13.60.1.38 __FFLASFFPACK_USE_OPENMP . . . 821
13.60.1.39 __FFLASFFPACK_VERSION . . . 821
13.61 fflas-ffpack-config.h File Reference . . . 821
13.61.1 Detailed Description . . . 822
13.61.2 Macro Definition Documentation . . . 822

13.61.2.1 GCC_VERSION . . . 822
13.62 fflas-ffpack-default-thresholds.h File Reference . . . 822
13.62.1 Macro Definition Documentation . . . 822
13.62.1.1 __FFLASFFPACK_WINOTHRESHOLD . . . 822
13.62.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT . . . 822
13.62.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL . . . 822
13.62.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT . . . 822
13.62.1.5 __FFLASFFPACK_PLUQ_THRESHOLD . . . 822
13.62.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD . . . 822
13.62.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD . . . 823
13.62.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD . . . 823
13.62.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD . . . 823
13.62.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD . . . 823
13.62.1.11 __FFLASFFPACK_FSYRK_THRESHOLD . . . 823
13.63 fflas-ffpack-thresholds.h File Reference . . . 823
13.64 fflas-ffpack.doxy File Reference . . . 823
13.65 fflas-ffpack.h File Reference . . . 823
13.65.1 Detailed Description . . . 823
13.66 fflas.doxy File Reference . . . 823
13.67 fflas.h File Reference . . . 823
13.67.1 Detailed Description . . . 824
13.67.2 Macro Definition Documentation . . . 824
13.67.2.1 WINOTHRESHOLD . . . 824
13.67.2.2 DOUBLE_TO_FLOAT_CROSSOVER . . . 824
13.68 fflas_bounds.inl File Reference . . . 825
13.68.1 Macro Definition Documentation . . . 825
13.68.1.1 __FFLASFFPACK_fflas_bounds_INL . . . 825
13.68.1.2 FFLAS_INT_TYPE . . . 825
13.69 fflas_enum.h File Reference . . . 825
13.70 fflas_fadd.h File Reference . . . 826
13.71 fflas_fadd.inl File Reference . . . 827
13.71.1 Macro Definition Documentation . . . 828
13.71.1.1 __FFLASFFPACK_fadd_INL . . . 828
13.72 fflas_fassign.h File Reference . . . 828
13.73 fflas_fassign.inl File Reference . . . 829
13.73.1 Macro Definition Documentation . . . 829
13.73.1.1 __FFLASFFPACK_fassign_INL . . . 829
13.74 fflas_faxpy.inl File Reference . . . 829
13.74.1 Macro Definition Documentation . . . 830
13.74.1.1 __FFLASFFPACK_faxpy_INL . . . 830
13.75 fflas_fdot.inl File Reference . . . 830
13.75.1 Macro Definition Documentation . . . 831

13.75.1.1 __FFLASFFPACK_fdot_INL . . . 831
13.76 fflas_fgemm.inl File Reference . . . 831
13.76.1 Macro Definition Documentation . . . 833
13.76.1.1 __FFLASFFPACK_fgemm_INL . . . 833
13.77 fgemm_classical.inl File Reference . . . 833
13.78 fgemm_classical_mp.inl File Reference . . . 833
13.78.1 Detailed Description . . . 835
13.78.2 Macro Definition Documentation . . . 835
13.78.2.1 __FFPACK_fgemm_classical_INL . . . 835
13.79 fgemm_winograd.inl File Reference . . . 835
13.79.1 Macro Definition Documentation . . . 836
13.79.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL . . . 836
13.79.1.2 NEWWINO . . . 837
13.80 matmul.doxy File Reference . . . 837
13.81 schedule_bini.inl File Reference . . . 837
13.81.1 Detailed Description . . . 837
13.81.2 Macro Definition Documentation . . . 837
13.81.2.1 __FFLASFFPACK_fgemm_bini_INL . . . 837
13.82 schedule_winograd.inl File Reference . . . 837
13.82.1 Macro Definition Documentation . . . 838
13.82.1.1 __FFLASFFPACK_fgemm_winograd_INL . . . 838
13.83 schedule_winograd_acc.inl File Reference . . . 838
13.83.1 Macro Definition Documentation . . . 838
13.83.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL . . . 838
13.84 schedule_winograd_acc_ip.inl File Reference . . . 839
13.84.1 Macro Definition Documentation . . . 839
13.84.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL . . . 839
13.85 schedule_winograd_ip.inl File Reference . . . 839
13.85.1 Macro Definition Documentation . . . 840
13.85.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL . . . 840
13.86 fflas_fgemv.inl File Reference . . . 840
13.86.1 Macro Definition Documentation . . . 841
13.86.1.1 __FFLASFFPACK_fgemv_INL . . . 841
13.87 fflas_fgemv_mp.inl File Reference . . . 841
13.87.1 Macro Definition Documentation . . . 842
13.87.1.1 __FFLASFFPACK_fgemv_mp_INL . . . 842
13.88 fflas_fger.inl File Reference . . . 842
13.88.1 Macro Definition Documentation . . . 843
13.88.1.1 __FFLASFFPACK_fger_INL . . . 843
13.89 fflas_fger_mp.inl File Reference . . . 843
13.89.1 Macro Definition Documentation . . . 844
13.89.1.1 __FFPACK_fger_mp_INL . . . 844

13.90 fflas_freduce.h File Reference . . . 844
13.91 fflas_freduce.inl File Reference . . . 845
13.91.1 Macro Definition Documentation . . . 846
13.91.1.1 __FFLASFFPACK_fflas_freduce_INL . . . 846
13.91.1.2 FFLASFFPACK_COPY_REDUCE . . . 846
13.92 fflas_freduce_mp.inl File Reference . . . 847
13.92.1 Macro Definition Documentation . . . 847
13.92.1.1 __FFLASFFPACK_fflas_freduce_mp_INL . . . 847
13.93 fflas_freivalds.inl File Reference . . . 847
13.93.1 Macro Definition Documentation . . . 847
13.93.1.1 __FFLASFFPACK_freivalds_INL . . . 847
13.94 fflas_fscal.h File Reference . . . 847
13.95 fflas_fscal.inl File Reference . . . 848
13.95.1 Macro Definition Documentation . . . 849
13.95.1.1 __FFLASFFPACK_fscal_INL . . . 849
13.96 fflas_fscal_mp.inl File Reference . . . 849
13.96.1 Macro Definition Documentation . . . 850
13.96.1.1 __FFLASFFPACK_fscal_mp_INL . . . 850
13.97 fflas_fsyr2k.inl File Reference . . . 850
13.97.1 Macro Definition Documentation . . . 850
13.97.1.1 __FFLASFFPACK_fflas_fsyr2k_INL . . . 850
13.98 fflas_fsyrk.inl File Reference . . . 850
13.98.1 Macro Definition Documentation . . . 852
13.98.1.1 __FFLASFFPACK_fflas_fsyrk_INL . . . 852
13.99 fflas_fsyrk_strassen.inl File Reference . . . 852
13.99.1 Macro Definition Documentation . . . 853
13.99.1.1 __FFLASFFPACK_fflas_fsyrk_strassen_INL . . . 853
13.100 fflas_ftrmm.inl File Reference . . . 853
13.100.1 Macro Definition Documentation . . . 854
13.100.1.1 __FFLASFFPACK_ftrmm_INL . . . 854
13.101 fflas_ftrsm.inl File Reference . . . 854
13.101.1 Macro Definition Documentation . . . 854
13.101.1.1 __FFLASFFPACK_ftrsm_INL . . . 854
13.102 fflas_ftrsm_mp.inl File Reference . . . 854
13.102.1 Detailed Description . . . 855
13.102.2 Macro Definition Documentation . . . 855
13.102.2.1 __FFPACK_ftrsm_mp_INL . . . 855
13.103 fflas_ftrsv.inl File Reference . . . 855
13.103.1 Macro Definition Documentation . . . 855
13.103.1.1 __FFLASFFPACK_ftrsv_INL . . . 855
13.104 fflas_helpers.inl File Reference . . . 855
13.104.1 Macro Definition Documentation . . . 856

13.104.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL . . . 856
13.105 igemm.doxy File Reference . . . 857
13.106 igemm.h File Reference . . . 857
13.107 igemm.inl File Reference . . . 857
13.107.1 Macro Definition Documentation . . . 858
13.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL . . . 858
13.108 igemm_kernels.h File Reference . . . 858
13.109 igemm_kernels.inl File Reference . . . 858
13.109.1 Macro Definition Documentation . . . 859
13.109.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL . . . 859
13.110 igemm_tools.h File Reference . . . 859
13.111 igemm_tools.inl File Reference . . . 859
13.111.1 Macro Definition Documentation . . . 860
13.111.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL . . . 860
13.112 fflas_level1.inl File Reference . . . 860
13.112.1 Macro Definition Documentation . . . 862
13.112.1.1 __FFLASFFPACK_fflas_fflas_level1_INL . . . 862
13.113 fflas_level2.inl File Reference . . . 862
13.113.1 Macro Definition Documentation . . . 865
13.113.1.1 __FFLASFFPACK_fflas_fflas_level2_INL . . . 865
13.114 fflas_level3.inl File Reference . . . 865
13.114.1 Macro Definition Documentation . . . 867
13.114.1.1 __FFLASFFPACK_fflas_fflas_level3_INL . . . 867
13.114.1.2 __FFLAS__TRSM_READONLY . . . 867
13.115 fflas_pfgemm.inl File Reference . . . 867
13.115.1 Macro Definition Documentation . . . 868
13.115.1.1 __FFLASFFPACK_fflas_pfgemm_INL . . . 868
13.115.1.2 __FFLASFFPACK_SEQPARTHRESHOLD . . . 868
13.115.1.3 __FFLASFFPACK_DIMKPENALTY . . . 868
13.116 fflas_pftrsm.inl File Reference . . . 868
13.116.1 Macro Definition Documentation . . . 869
13.116.1.1 __FFLASFFPACK_fflas_pftrsm_INL . . . 869
13.116.1.2 PTRSM_HYBRID_THRESHOLD . . . 869
13.117 fflas_simd.h File Reference . . . 869
13.117.1 Macro Definition Documentation . . . 870
13.117.1.1 SIMD_INT . . . 870
13.117.1.2 INLINE . . . 870
13.117.1.3 CONST . . . 870
13.117.1.4 PURE . . . 870
13.117.1.5 NORML_MOD . . . 870
13.117.1.6 FLOAT_MOD . . . 870
13.117.2 Typedef Documentation . . . 870

13.117.2.1 Simd . . . 870
13.118 simd.doxy File Reference . . . 871
13.119 simd128.inl File Reference . . . 871
13.119.1 Macro Definition Documentation . . . 871
13.119.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_INL . . . 871
13.119.2 Typedef Documentation . . . 871
13.119.2.1 Simd128 . . . 871
13.120 simd128_double.inl File Reference . . . 871
13.120.1 Macro Definition Documentation . . . 872
13.120.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL . . . 872
13.121 simd128_float.inl File Reference . . . 872
13.121.1 Macro Definition Documentation . . . 872
13.121.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL . . . 872
13.122 simd128_int16.inl File Reference . . . 872
13.122.1 Macro Definition Documentation . . . 872
13.122.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL . . . 872
13.123 simd128_int32.inl File Reference . . . 872
13.123.1 Macro Definition Documentation . . . 873
13.123.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL . . . 873
13.124 simd128_int64.inl File Reference . . . 873
13.124.1 Macro Definition Documentation . . . 873
13.124.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL . . . 873
13.124.1.2 vect_t . . . 873
13.125 simd256.inl File Reference . . . 873
13.125.1 Macro Definition Documentation . . . 874
13.125.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_INL . . . 874
13.125.2 Typedef Documentation . . . 874
13.125.2.1 Simd256 . . . 874
13.126 simd256_double.inl File Reference . . . 874
13.126.1 Macro Definition Documentation . . . 874
13.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL . . . 874
13.127 simd256_float.inl File Reference . . . 874
13.127.1 Macro Definition Documentation . . . 875
13.127.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL . . . 875
13.128 simd256_int16.inl File Reference . . . 875
13.128.1 Macro Definition Documentation . . . 875
13.128.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL . . . 875
13.129 simd256_int32.inl File Reference . . . 875
13.129.1 Macro Definition Documentation . . . 876
13.129.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL . . . 876
13.130 simd256_int64.inl File Reference . . . 876
13.130.1 Macro Definition Documentation . . . 876

13.130.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL . . . 876
13.130.1.2 vect_t . . . 876
13.131 simd512.inl File Reference . . . 876
13.131.1 Macro Definition Documentation . . . 877
13.131.1.1 __FFLASFFPACK_simd512_INL . . . 877
13.131.2 Typedef Documentation . . . 877
13.131.2.1 Simd512 . . . 877
13.132 simd512_double.inl File Reference . . . 877
13.132.1 Macro Definition Documentation . . . 877
13.132.1.1 __FFLASFFPACK_simd512_double_INL . . . 877
13.133 simd512_float.inl File Reference . . . 877
13.133.1 Macro Definition Documentation . . . 877
13.133.1.1 __FFLASFFPACK_simd512_float_INL . . . 877
13.134 simd512_int32.inl File Reference . . . 878
13.134.1 Macro Definition Documentation . . . 878
13.134.1.1 __FFLASFFPACK_simd512_int32_INL . . . 878
13.135 simd512_int64.inl File Reference . . . 878
13.135.1 Macro Definition Documentation . . . 878
13.135.1.1 _simd512_int64_INL . . . 878
13.135.1.2 vect_t . . . 878
13.136 simd_modular.inl File Reference . . . 879
13.137 fflas_sparse.h File Reference . . . 879
13.137.1 Macro Definition Documentation . . . 883
13.137.1.1 index_t . . . 883
13.137.1.2 ROUND_DOWN . . . 883
13.137.1.3 __FFLASFFPACK_CACHE_LINE_SIZE . . . 883
13.137.1.4 assume_aligned . . . 883
13.137.1.5 DENSE_THRESHOLD . . . 883
13.138 fflas_sparse.inl File Reference . . . 883
13.138.1 Macro Definition Documentation . . . 885
13.138.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL . . . 885
13.139 coo.h File Reference . . . 885
13.140 coo_spmm.inl File Reference . . . 886
13.140.1 Macro Definition Documentation . . . 886
13.140.1.1 __FFLASFFPACK_fflas_sparse_coo_spmm_INL . . . 886
13.141 coo_spmv.inl File Reference . . . 887
13.141.1 Macro Definition Documentation . . . 887
13.141.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL . . . 887
13.142 coo_utils.inl File Reference . . . 887
13.142.1 Macro Definition Documentation . . . 888
13.142.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL . . . 888
13.143 csr.h File Reference . . . 888

13.144 csr_pspmm.inl File Reference . . . 888
13.144.1 Macro Definition Documentation . . . 889
13.144.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL . . . 889
13.145 csr_pspmv.inl File Reference . . . 889
13.145.1 Macro Definition Documentation . . . 890
13.145.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL . . . 890
13.146 csr_spmm.inl File Reference . . . 890
13.146.1 Macro Definition Documentation . . . 891
13.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmm_INL . . . 891
13.147 csr_spmv.inl File Reference . . . 891
13.147.1 Macro Definition Documentation . . . 892
13.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL . . . 892
13.148 csr_utils.inl File Reference . . . 892
13.149 csr_hyb.h File Reference . . . 893
13.150 csr_hyb_pspmm.inl File Reference . . . 893
13.150.1 Macro Definition Documentation . . . 894
13.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL . . . 894
13.151 csr_hyb_pspmv.inl File Reference . . . 894
13.151.1 Macro Definition Documentation . . . 894
13.151.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL . . . 894
13.152 csr_hyb_spmm.inl File Reference . . . 894
13.152.1 Macro Definition Documentation . . . 895
13.152.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL . . . 895
13.153 csr_hyb_spmv.inl File Reference . . . 895
13.153.1 Macro Definition Documentation . . . 895
13.153.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL . . . 895
13.154 csr_hyb_utils.inl File Reference . . . 895
13.154.1 Macro Definition Documentation . . . 896
13.154.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL . . . 896
13.155 ell.h File Reference . . . 896
13.156 ell_pspmm.inl File Reference . . . 896
13.156.1 Macro Definition Documentation . . . 897
13.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL . . . 897
13.157 ell_pspmv.inl File Reference . . . 897
13.157.1 Macro Definition Documentation . . . 898
13.157.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL . . . 898
13.158 ell_spmm.inl File Reference . . . 898
13.158.1 Macro Definition Documentation . . . 899
13.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmm_INL . . . 899
13.159 ell_spmv.inl File Reference . . . 899
13.159.1 Macro Definition Documentation . . . 899
13.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL . . . 899

13.160 ell_utils.inl File Reference . . . 899
13.160.1 Macro Definition Documentation . . . 900
13.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL . . . 900
13.161 ell_simd.h File Reference . . . 900
13.162 ell_simd_pspmv.inl File Reference . . . 901
13.162.1 Macro Definition Documentation . . . 901
13.162.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL . . . 901
13.163 ell_simd_spmv.inl File Reference . . . 901
13.163.1 Macro Definition Documentation . . . 902
13.163.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL . . . 902
13.164 ell_simd_utils.inl File Reference . . . 902
13.164.1 Macro Definition Documentation . . . 903
13.164.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL . . . 903
13.165 hyb_zo.h File Reference . . . 903
13.166 hyb_zo_pspmm.inl File Reference . . . 903
13.166.1 Macro Definition Documentation . . . 903
13.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL . . . 903
13.167 hyb_zo_pspmv.inl File Reference . . . 903
13.167.1 Macro Definition Documentation . . . 904
13.167.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL . . . 904
13.168 hyb_zo_spmm.inl File Reference . . . 904
13.168.1 Macro Definition Documentation . . . 904
13.168.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL . . . 904
13.169 hyb_zo_spmv.inl File Reference . . . 904
13.169.1 Macro Definition Documentation . . . 905
13.169.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL . . . 905
13.170 hyb_zo_utils.inl File Reference . . . 905
13.170.1 Macro Definition Documentation . . . 905
13.170.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL . . . 905
13.171 read_sparse.h File Reference . . . 905
13.171.1 Macro Definition Documentation . . . 906
13.171.1.1 DNS_BIN_VER . . . 906
13.171.1.2 mask_t . . . 906
13.172 sell.h File Reference . . . 906
13.173 sell_pspmv.inl File Reference . . . 907
13.173.1 Macro Definition Documentation . . . 907
13.173.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL . . . 907
13.174 sell_spmv.inl File Reference . . . 907
13.174.1 Macro Definition Documentation . . . 908
13.174.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL . . . 908
13.175 sell_utils.inl File Reference . . . 908
13.175.1 Macro Definition Documentation . . . 909

13.175.1.1 __FFLASFFPACK_fflas_sparse_sell_utils_INL . . . 909
13.176 sparse_matrix_traits.h File Reference . . . 909
13.177 utils.h File Reference . . . 910
13.178 fflas_transpose.h File Reference . . . 911
13.178.1 Detailed Description . . . 912
13.178.2 Macro Definition Documentation . . . 912
13.178.2.1 FFLAS_TRANSPOSE_BLOCKSIZE . . . 912
13.178.2.2 LD . . . 912
13.178.2.3 ST . . . 912
13.179 ffpack.doxy File Reference . . . 912
13.180 ffpack.h File Reference . . . 912
13.180.1 Detailed Description . . . 921
13.180.2 Macro Definition Documentation . . . 921
13.180.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD . . . 921
13.180.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD . . . 921
13.181 ffpack.inl File Reference . . . 921
13.181.1 Macro Definition Documentation . . . 922
13.181.1.1 __FFLASFFPACK_ffpack_INL . . . 922
13.182 ffpack_bruhatgen.inl File Reference . . . 922
13.182.1 Macro Definition Documentation . . . 924
13.182.1.1 __FFLASFFPACK_ffpack_bruhatgen_inl . . . 924
13.183 ffpack_charpoly.inl File Reference . . . 924
13.183.1 Macro Definition Documentation . . . 924
13.183.1.1 __FFLASFFPACK_charpoly_INL . . . 924
13.184 ffpack_charpoly_danilevski.inl File Reference . . . 924
13.184.1 Macro Definition Documentation . . . 925
13.184.1.1 __FFLASFFPACK_ffpack_charpoly_danilveski_INL . . . 925
13.185 ffpack_charpoly_kgfast.inl File Reference . . . 925
13.185.1 Macro Definition Documentation . . . 925
13.185.1.1 __FFLASFFPACK_ffpack_charpoly_kgfast_INL . . . 925
13.186 ffpack_charpoly_kgfastgeneralized.inl File Reference . . . 925
13.186.1 Macro Definition Documentation . . . 926
13.186.1.1 __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL . . . 926
13.187 ffpack_charpoly_kglu.inl File Reference . . . 926
13.187.1 Macro Definition Documentation . . . 926
13.187.1.1 __FFLASFFPACK_ffpack_charpoly_kglu_INL . . . 926
13.188 ffpack_charpoly_mp.inl File Reference . . . 926
13.188.1 Macro Definition Documentation . . . 927
13.188.1.1 __FFPACK_charpoly_mp_INL . . . 927
13.189 ffpack_det_mp.inl File Reference . . . 927
13.189.1 Macro Definition Documentation . . . 927
13.189.1.1 __FFPACK_det_mp_INL . . . 927

13.190 ffpack_echelonforms.inl File Reference . . . 928
13.190.1 Macro Definition Documentation . . . 929
13.190.1.1 __FFLASFFPACK_ffpack_echelon_forms_INL . . . 929
13.190.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE . . . 929
13.191 ffpack_fgesv.inl File Reference . . . 929
13.191.1 Macro Definition Documentation . . . 929
13.191.1.1 __FFLASFFPACK_ffpack_fgesv_INL . . . 929
13.192 ffpack_fgetrs.inl File Reference . . . 929
13.192.1 Macro Definition Documentation . . . 930
13.192.1.1 __FFLASFFPACK_ffpack_fgetrs_INL . . . 930
13.193 ffpack_frobenius.inl File Reference . . . 930
13.194 ffpack_fsytrf.inl File Reference . . . 931
13.194.1 Macro Definition Documentation . . . 932
13.194.1.1 __FFLASFFPACK_ffpack_fsytrf_INL . . . 932
13.195 ffpack_ftrssyr2k.inl File Reference . . . 932
13.195.1 Macro Definition Documentation . . . 932
13.195.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL . . . 932
13.196 ffpack_ftrstr.inl File Reference . . . 932
13.196.1 Macro Definition Documentation . . . 933
13.196.1.1 __FFLASFFPACK_ffpack_ftrstr_INL . . . 933
13.197 ffpack_ftrtr.inl File Reference . . . 933
13.197.1 Macro Definition Documentation . . . 933
13.197.1.1 ENABLE_ALL_CHECKINGS . . . 933
13.197.1.2 __FFLASFFPACK_ffpack_ftrtr_INL . . . 933
13.198 ffpack_invert.inl File Reference . . . 933
13.198.1 Macro Definition Documentation . . . 934
13.198.1.1 __FFLASFFPACK_ffpack_invert_INL . . . 934
13.199 ffpack_krylovelim.inl File Reference . . . 934
13.199.1 Macro Definition Documentation . . . 934
13.199.1.1 __FFLASFFPACK_ffpack_krylovelim_INL . . . 934
13.200 ffpack_ludivine.inl File Reference . . . 934
13.200.1 Macro Definition Documentation . . . 935
13.200.1.1 __FFLASFFPACK_ffpack_ludivine_INL . . . 935
13.201 ffpack_ludivine_mp.inl File Reference . . . 935
13.201.1 Macro Definition Documentation . . . 935
13.201.1.1 __FFPACK_ludivine_mp_INL . . . 935
13.202 ffpack_minpoly.inl File Reference . . . 936
13.202.1 Macro Definition Documentation . . . 936
13.202.1.1 __FFLASFFPACK_ffpack_minpoly_INL . . . 936
13.203 ffpack_permutation.inl File Reference . . . 936
13.203.1 Macro Definition Documentation . . . 938
13.203.1.1 __FFLASFFPACK_ffpack_permutation_INL . . . 938

13.203.1.2 FFLASFFPACK_PERM_BKSIZE . . . 938
13.204 ffpack_pluq.inl File Reference . . . 939
13.204.1 Macro Definition Documentation . . . 939
13.204.1.1 __FFLASFFPACK_ffpack_pluq_INL . . . 939
13.204.1.2 CROUT . . . 939
13.205 ffpack_pluq_mp.inl File Reference . . . 939
13.205.1 Macro Definition Documentation . . . 940
13.205.1.1 __FFPACK_pluq_mp_INL . . . 940
13.206 ffpack_ppluq.inl File Reference . . . 940
13.206.1 Macro Definition Documentation . . . 940
13.206.1.1 __FFLASFFPACK_ffpack_ppluq_INL . . . 940
13.206.1.2 __FFLAS__TRSM_READONLY . . . 940
13.206.1.3 PBASECASE_K . . . 940
13.207 ffpack_rankprofiles.inl File Reference . . . 941
13.207.1 Macro Definition Documentation . . . 942
13.207.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL . . . 942
13.208 field-traits.h File Reference . . . 942
13.208.1 Detailed Description . . . 944
13.209 field.doxy File Reference . . . 944
13.210 rns-double-elt.h File Reference . . . 944
13.210.1 Detailed Description . . . 944
13.211 rns-double-recint.inl File Reference . . . 945
13.211.1 Macro Definition Documentation . . . 945
13.211.1.1 __FFLASFFPACK_field_rns_double_recint_INL . . . 945
13.212 rns-double.h File Reference . . . 945
13.212.1 Detailed Description . . . 946
13.212.2 Macro Definition Documentation . . . 946
13.212.2.1 ROUND_DOWN . . . 946
13.213 rns-double.inl File Reference . . . 946
13.213.1 Macro Definition Documentation . . . 946
13.213.1.1 __FFLASFFPACK_field_rns_double_INL . . . 946
13.214 rns-integer-mod.h File Reference . . . 946
13.214.1 Detailed Description . . . 947
13.215 rns-integer.h File Reference . . . 947
13.215.1 Detailed Description . . . 948
13.216 rns.h File Reference . . . 948
13.217 rns.inl File Reference . . . 948
13.217.1 Macro Definition Documentation . . . 948
13.217.1.1 __FFLASFFPACK_field_rns_INL . . . 948
13.218 interfaces.doxy File Reference . . . 948
13.219 fflas_c.h File Reference . . . 948
13.219.1 Macro Definition Documentation . . . 950

13.219.1.1 FFLAS_COMPILED . . . 950
13.219.2 Enumeration Type Documentation . . . 950
13.219.2.1 FFLAS_C_ORDER . . . 950
13.219.2.2 FFLAS_C_TRANSPOSE . . . 950
13.219.2.3 FFLAS_C_UPLO . . . 951
13.219.2.4 FFLAS_C_DIAG . . . 951
13.219.2.5 FFLAS_C_SIDE . . . 951
13.219.2.6 FFLAS_C_BASE . . . 951
13.219.3 Function Documentation . . . 952
13.219.3.1 freducein_1_modular_double() . . . 952
13.219.3.2 freduce_1_modular_double() . . . 952
13.219.3.3 fnegin_1_modular_double() . . . 952
13.219.3.4 fneg_1_modular_double() . . . 952
13.219.3.5 fzero_1_modular_double() . . . 952
13.219.3.6 fiszero_1_modular_double() . . . 952
13.219.3.7 fequal_1_modular_double() . . . 953
13.219.3.8 fassign_1_modular_double() . . . 953
13.219.3.9 fscalin_1_modular_double() . . . 953
13.219.3.10 fscal_1_modular_double() . . . 953
13.219.3.11 faxpy_1_modular_double() . . . 953
13.219.3.12 fdot_1_modular_double() . . . 953
13.219.3.13 fswap_1_modular_double() . . . 954
13.219.3.14 fadd_1_modular_double() . . . 954
13.219.3.15 fsub_1_modular_double() . . . 954
13.219.3.16 faddin_1_modular_double() . . . 954
13.219.3.17 fsubin_1_modular_double() . . . 954
13.219.3.18 fassign_2_modular_double() . . . 955
13.219.3.19 fzero_2_modular_double() . . . 955
13.219.3.20 fequal_2_modular_double() . . . 955
13.219.3.21 fiszero_2_modular_double() . . . 955
13.219.3.22 fidentity_2_modular_double() . . . 955
13.219.3.23 freducein_2_modular_double() . . . 956
13.219.3.24 freduce_2_modular_double() . . . 956
13.219.3.25 fnegin_2_modular_double() . . . 956
13.219.3.26 fneg_2_modular_double() . . . 956
13.219.3.27 fscalin_2_modular_double() . . . 956
13.219.3.28 fscal_2_modular_double() . . . 957
13.219.3.29 faxpy_2_modular_double() . . . 957
13.219.3.30 fmove_2_modular_double() . . . 957
13.219.3.31 fadd_2_modular_double() . . . 957
13.219.3.32 fsub_2_modular_double() . . . 957
13.219.3.33 fsubin_2_modular_double() . . . 958

13.219.3.34 faddin_2_modular_double() . . . 958
13.219.3.35 fgemv_2_modular_double() . . . 958
13.219.3.36 fger_2_modular_double() . . . 958
13.219.3.37 ftrsv_2_modular_double() . . . 959
13.219.3.38 ftrsm_3_modular_double() . . . 959
13.219.3.39 ftrmm_3_modular_double() . . . 959
13.219.3.40 fgemm_3_modular_double() . . . 959
13.219.3.41 fsquare_3_modular_double() . . . 960
13.220 fflas_L1_inst.C File Reference . . . 960
13.220.1 Macro Definition Documentation . . . 960
13.220.1.1 __FFLAS_L1_INST_C . . . 960
13.220.1.2 INST_OR_DECL . . . 960
13.220.1.3 FFLAS_FIELD [1/2] . . . 961
13.220.1.4 FFLAS_ELT [1/6] . . . 961
13.220.1.5 FFLAS_ELT [2/6] . . . 961
13.220.1.6 FFLAS_ELT [3/6] . . . 961
13.220.1.7 FFLAS_FIELD [2/2] . . . 961
13.220.1.8 FFLAS_ELT [4/6] . . . 961
13.220.1.9 FFLAS_ELT [5/6] . . . 961
13.220.1.10 FFLAS_ELT [6/6] . . . 961
13.221 fflas_L1_inst.h File Reference . . . 961
13.221.1 Macro Definition Documentation . . . 961
13.221.1.1 INST_OR_DECL . . . 961
13.221.1.2 FFLAS_FIELD [1/2] . . . 962
13.221.1.3 FFLAS_ELT [1/6] . . . 962
13.221.1.4 FFLAS_ELT [2/6] . . . 962
13.221.1.5 FFLAS_ELT [3/6] . . . 962
13.221.1.6 FFLAS_FIELD [2/2] . . . 962
13.221.1.7 FFLAS_ELT [4/6] . . . 962
13.221.1.8 FFLAS_ELT [5/6] . . . 962
13.221.1.9 FFLAS_ELT [6/6] . . . 962
13.222 fflas_L1_inst_implem.inl File Reference . . . 962
13.223 fflas_L2_inst.C File Reference . . . 963
13.223.1 Macro Definition Documentation . . . 964
13.223.1.1 __FFLAS_L2_INST_C . . . 964
13.223.1.2 INST_OR_DECL . . . 964
13.223.1.3 FFLAS_FIELD [1/2] . . . 964
13.223.1.4 FFLAS_ELT [1/6] . . . 964
13.223.1.5 FFLAS_ELT [2/6] . . . 964
13.223.1.6 FFLAS_ELT [3/6] . . . 964
13.223.1.7 FFLAS_FIELD [2/2] . . . 964
13.223.1.8 FFLAS_ELT [4/6] . . . 964

13.223.1.9 FFLAS_ELT [5/6] . . . 964
13.223.1.10 FFLAS_ELT [6/6] . . . 964
13.224 fflas_L2_inst.h File Reference . . . 964
13.224.1 Macro Definition Documentation . . . 965
13.224.1.1 INST_OR_DECL . . . 965
13.224.1.2 FFLAS_FIELD [1/2] . . . 965
13.224.1.3 FFLAS_ELT [1/6] . . . 965
13.224.1.4 FFLAS_ELT [2/6] . . . 965
13.224.1.5 FFLAS_ELT [3/6] . . . 965
13.224.1.6 FFLAS_FIELD [2/2] . . . 965
13.224.1.7 FFLAS_ELT [4/6] . . . 965
13.224.1.8 FFLAS_ELT [5/6] . . . 965
13.224.1.9 FFLAS_ELT [6/6] . . . 965
13.225 fflas_L2_inst_implem.inl File Reference . . . 965
13.226 fflas_L3_inst.C File Reference . . . 967
13.226.1 Macro Definition Documentation . . . 967
13.226.1.1 __FFLAS_L3_INST_C . . . 967
13.226.1.2 INST_OR_DECL . . . 968
13.226.1.3 FFLAS_FIELD [1/2] . . . 968
13.226.1.4 FFLAS_ELT [1/6] . . . 968
13.226.1.5 FFLAS_ELT [2/6] . . . 968
13.226.1.6 FFLAS_ELT [3/6] . . . 968
13.226.1.7 FFLAS_FIELD [2/2] . . . 968
13.226.1.8 FFLAS_ELT [4/6] . . . 968
13.226.1.9 FFLAS_ELT [5/6] . . . 968
13.226.1.10 FFLAS_ELT [6/6] . . . 968
13.227 fflas_L3_inst.h File Reference . . . 968
13.227.1 Macro Definition Documentation . . . 969
13.227.1.1 INST_OR_DECL . . . 969
13.227.1.2 FFLAS_FIELD [1/2] . . . 969
13.227.1.3 FFLAS_ELT [1/6] . . . 969
13.227.1.4 FFLAS_ELT [2/6] . . . 969
13.227.1.5 FFLAS_ELT [3/6] . . . 969
13.227.1.6 FFLAS_FIELD [2/2] . . . 969
13.227.1.7 FFLAS_ELT [4/6] . . . 969
13.227.1.8 FFLAS_ELT [5/6] . . . 969
13.227.1.9 FFLAS_ELT [6/6] . . . 969
13.228 fflas_L3_inst_implem.inl File Reference . . . 969
13.228.1 Macro Definition Documentation . . . 970
13.228.1.1 __FFLAS__TRSM_READONLY . . . 970
13.229 fflas_lvl1.C File Reference . . . 970
13.229.1 Detailed Description . . . 971

13.229.2 Function Documentation . . . 971
13.229.2.1 freducein_1_modular_double() . . . 971
13.229.2.2 freduce_1_modular_double() . . . 971
13.229.2.3 fnegin_1_modular_double() . . . 971
13.229.2.4 fneg_1_modular_double() . . . 972
13.229.2.5 fzero_1_modular_double() . . . 972
13.229.2.6 fiszero_1_modular_double() . . . 972
13.229.2.7 fequal_1_modular_double() . . . 972
13.229.2.8 fassign_1_modular_double() . . . 972
13.229.2.9 fscalin_1_modular_double() . . . 972
13.229.2.10 fscal_1_modular_double() . . . 973
13.229.2.11 faxpy_1_modular_double() . . . 973
13.229.2.12 fdot_1_modular_double() . . . 973
13.229.2.13 fswap_1_modular_double() . . . 973
13.229.2.14 fadd_1_modular_double() . . . 973
13.229.2.15 fsub_1_modular_double() . . . 974
13.229.2.16 faddin_1_modular_double() . . . 974
13.229.2.17 fsubin_1_modular_double() . . . 974
13.230 fflas_lvl2.C File Reference . . . 974
13.230.1 Detailed Description . . . 975
13.230.2 Function Documentation . . . 975
13.230.2.1 fassign_2_modular_double() . . . 975
13.230.2.2 fzero_2_modular_double() . . . 976
13.230.2.3 fequal_2_modular_double() . . . 976
13.230.2.4 fiszero_2_modular_double() . . . 976
13.230.2.5 fidentity_2_modular_double() . . . 976
13.230.2.6 freducein_2_modular_double() . . . 976
13.230.2.7 freduce_2_modular_double() . . . 976
13.230.2.8 fnegin_2_modular_double() . . . 977
13.230.2.9 fneg_2_modular_double() . . . 977
13.230.2.10 fscalin_2_modular_double() . . . 977
13.230.2.11 fscal_2_modular_double() . . . 977
13.230.2.12 faxpy_2_modular_double() . . . 977
13.230.2.13 fmove_2_modular_double() . . . 978
13.230.2.14 fadd_2_modular_double() . . . 978
13.230.2.15 fsub_2_modular_double() . . . 978
13.230.2.16 fsubin_2_modular_double() . . . 978
13.230.2.17 faddin_2_modular_double() . . . 979
13.230.2.18 fgemv_2_modular_double() . . . 979
13.230.2.19 fger_2_modular_double() . . . 979
13.230.2.20 ftrsv_2_modular_double() . . . 979
13.231 fflas_lvl3.C File Reference . . . 980

13.231.1 Detailed Description . . . 980
13.231.2 Function Documentation . . . 980
13.231.2.1 ftrsm_3_modular_double() . . . 980
13.231.2.2 ftrmm_3_modular_double() . . . 981
13.231.2.3 fgemm_3_modular_double() . . . 981
13.231.2.4 fsquare_3_modular_double() . . . 981
13.232 fflas_sparse.C File Reference . . . 981
13.232.1 Detailed Description . . . 981
13.233 ffpack.C File Reference . . . 982
13.233.1 Detailed Description . . . 985
13.233.2 Function Documentation . . . 985
13.233.2.1 LAPACKPerm2MathPerm() . . . 985
13.233.2.2 MathPerm2LAPACKPerm() . . . 985
13.233.2.3 MatrixApplyS_modular_double() . . . 985
13.233.2.4 PermApplyS_double() . . . 986
13.233.2.5 MatrixApplyT_modular_double() . . . 986
13.233.2.6 PermApplyT_double() . . . 986
13.233.2.7 composePermutationsLLM() . . . 986
13.233.2.8 composePermutationsLLL() . . . 986
13.233.2.9 composePermutationsMLM() . . . 987
13.233.2.10 cyclic_shift_mathPerm() . . . 987
13.233.2.11 cyclic_shift_row_modular_double() . . . 987
13.233.2.12 cyclic_shift_col_modular_double() . . . 987
13.233.2.13 applyP_modular_double() . . . 987
13.233.2.14 fgetrsin_modular_double() . . . 987
13.233.2.15 fgetrsv_modular_double() . . . 988
13.233.2.16 fgesvin_modular_double() . . . 988
13.233.2.17 fgesv_modular_double() . . . 988
13.233.2.18 ftrtri_modular_double() . . . 989
13.233.2.19 trinv_left_modular_double() . . . 989
13.233.2.20 ftrtrm_modular_double() . . . 989
13.233.2.21 PLUQ_modular_double() . . . 989
13.233.2.22 LUdivine_modular_double() . . . 989
13.233.2.23 ColumnEchelonForm_modular_double() . . . 990
13.233.2.24 RowEchelonForm_modular_double() . . . 990
13.233.2.25 ReducedColumnEchelonForm_modular_double() . . . 990
13.233.2.26 ReducedRowEchelonForm_modular_double() . . . 990
13.233.2.27 ColumnEchelonForm_modular_float() . . . 991
13.233.2.28 RowEchelonForm_modular_float() . . . 991
13.233.2.29 ReducedColumnEchelonForm_modular_float() . . . 991
13.233.2.30 ReducedRowEchelonForm_modular_float() . . . 991
13.233.2.31 ColumnEchelonForm_modular_int32_t() . . . 991

13.233.2.32 RowEchelonForm_modular_int32_t() . . . 992
13.233.2.33 ReducedColumnEchelonForm_modular_int32_t() . . . 992
13.233.2.34 ReducedRowEchelonForm_modular_int32_t() . . . 992
13.233.2.35 pColumnEchelonForm_modular_double() . . . 992
13.233.2.36 pRowEchelonForm_modular_double() . . . 993
13.233.2.37 pReducedColumnEchelonForm_modular_double() . . . 993
13.233.2.38 pReducedRowEchelonForm_modular_double() . . . 993
13.233.2.39 pColumnEchelonForm_modular_float() . . . 993
13.233.2.40 pRowEchelonForm_modular_float() . . . 994
13.233.2.41 pReducedColumnEchelonForm_modular_float() . . . 994
13.233.2.42 pReducedRowEchelonForm_modular_float() . . . 994
13.233.2.43 pColumnEchelonForm_modular_int32_t() . . . 994
13.233.2.44 pRowEchelonForm_modular_int32_t() . . . 995
13.233.2.45 pReducedColumnEchelonForm_modular_int32_t() . . . 995
13.233.2.46 pReducedRowEchelonForm_modular_int32_t() . . . 995
13.233.2.47 Invertin_modular_double() . . . 995
13.233.2.48 Invert_modular_double() . . . 995
13.233.2.49 Invert2_modular_double() . . . 996
13.233.2.50 KrylovElim_modular_double() . . . 996
13.233.2.51 SpecRankProfile_modular_double() . . . 996
13.233.2.52 Rank_modular_double() . . . 996
13.233.2.53 IsSingular_modular_double() . . . 996
13.233.2.54 Det_modular_double() . . . 997
13.233.2.55 Solve_modular_double() . . . 997
13.233.2.56 solveLB_modular_double() . . . 997
13.233.2.57 solveLB2_modular_double() . . . 997
13.233.2.58 RandomNullSpaceVector_modular_double() . . . 998
13.233.2.59 NullSpaceBasis_modular_double() . . . 998
13.233.2.60 RowRankProfile_modular_double() . . . 998
13.233.2.61 ColumnRankProfile_modular_double() . . . 998
13.233.2.62 RankProfileFromLU() . . . 998
13.233.2.63 LeadingSubmatrixRankProfiles() . . . 999
13.233.2.64 RowRankProfileSubmatrixIndices_modular_double() . . . 999
13.233.2.65 ColRankProfileSubmatrixIndices_modular_double() . . . 999
13.233.2.66 RowRankProfileSubmatrix_modular_double() . . . 999
13.233.2.67 ColRankProfileSubmatrix_modular_double() . . . 999
13.233.2.68 getTriangular_modular_double() . . . 1000
13.233.2.69 getTriangularin_modular_double() . . . 1000
13.233.2.70 getEchelonForm_modular_double() . . . 1000
13.233.2.71 getEchelonFormin_modular_double() . . . 1000
13.233.2.72 getEchelonTransform_modular_double() . . . 1001
13.233.2.73 getReducedEchelonForm_modular_double() . . . 1001

13.233.2.74 getReducedEchelonFormin_modular_double() . . . 1001
13.233.2.75 getReducedEchelonTransform_modular_double() . . . 1001
13.233.2.76 PLUQtoEchelonPermutation() . . . 1002
13.234 ffpack_c.h File Reference . . . 1002
13.234.1 Macro Definition Documentation . . . 1005
13.234.1.1 FFPACK_COMPILED . . . 1005
13.234.2 Enumeration Type Documentation . . . 1005
13.234.2.1 FFLAS_C_ORDER . . . 1005
13.234.2.2 FFLAS_C_TRANSPOSE . . . 1005
13.234.2.3 FFLAS_C_UPLO . . . 1005
13.234.2.4 FFLAS_C_DIAG . . . 1006
13.234.2.5 FFLAS_C_SIDE . . . 1006
13.234.2.6 FFPACK_C_LU_TAG . . . 1006
13.234.2.7 FFPACK_C_CHARPOLY_TAG . . . 1006
13.234.2.8 FFPACK_C_MINPOLY_TAG . . . 1006
13.234.3 Function Documentation . . . 1007
13.234.3.1 LAPACKPerm2MathPerm() . . . 1007
13.234.3.2 MathPerm2LAPACKPerm() . . . 1007
13.234.3.3 MatrixApplyS_modular_double() . . . 1007
13.234.3.4 PermApplyS_double() . . . 1007
13.234.3.5 MatrixApplyT_modular_double() . . . 1007
13.234.3.6 PermApplyT_double() . . . 1008
13.234.3.7 composePermutationsLLM() . . . 1008
13.234.3.8 composePermutationsLLL() . . . 1008
13.234.3.9 composePermutationsMLM() . . . 1008
13.234.3.10 cyclic_shift_mathPerm() . . . 1008
13.234.3.11 cyclic_shift_row_modular_double() . . . 1008
13.234.3.12 cyclic_shift_col_modular_double() . . . 1008
13.234.3.13 applyP_modular_double() . . . 1009
13.234.3.14 fgetrsin_modular_double() . . . 1009
13.234.3.15 fgetrs_modular_double() . . . 1009
13.234.3.16 fgesvin_modular_double() . . . 1010
13.234.3.17 fgesv_modular_double() . . . 1010
13.234.3.18 ftrtri_modular_double() . . . 1010
13.234.3.19 trinv_left_modular_double() . . . 1010
13.234.3.20 ftrtrm_modular_double() . . . 1010
13.234.3.21 PLUQ_modular_double() . . . 1011
13.234.3.22 LUdivine_modular_double() . . . 1011
13.234.3.23 LUdivine_small_modular_double() . . . 1011
13.234.3.24 LUdivine_gauss_modular_double() . . . 1011
13.234.3.25 ColumnEchelonForm_modular_double() . . . 1012
13.234.3.26 RowEchelonForm_modular_double() . . . 1012

13.234.3.27 ColumnEchelonForm_modular_float() . . . 1012
13.234.3.28 RowEchelonForm_modular_float() . . . 1012
13.234.3.29 ColumnEchelonForm_modular_int32_t() . . . 1013
13.234.3.30 RowEchelonForm_modular_int32_t() . . . 1013
13.234.3.31 ReducedColumnEchelonForm_modular_double() . . . 1013
13.234.3.32 ReducedRowEchelonForm_modular_double() . . . 1013
13.234.3.33 ReducedColumnEchelonForm_modular_float() . . . 1013
13.234.3.34 ReducedRowEchelonForm_modular_float() . . . 1014
13.234.3.35 ReducedColumnEchelonForm_modular_int32_t() . . . 1014
13.234.3.36 ReducedRowEchelonForm_modular_int32_t() . . . 1014
13.234.3.37 ReducedRowEchelonForm2_modular_double() . . . 1014
13.234.3.38 REF_modular_double() . . . 1015
13.234.3.39 Invertin_modular_double() . . . 1015
13.234.3.40 Invert_modular_double() . . . 1015
13.234.3.41 Invert2_modular_double() . . . 1015
13.234.3.42 KrylovElim_modular_double() . . . 1015
13.234.3.43 SpecRankProfile_modular_double() . . . 1016
13.234.3.44 Rank_modular_double() . . . 1016
13.234.3.45 IsSingular_modular_double() . . . 1016
13.234.3.46 Det_modular_double() . . . 1016
13.234.3.47 Solve_modular_double() . . . 1016
13.234.3.48 solveLB_modular_double() . . . 1017
13.234.3.49 solveLB2_modular_double() . . . 1017
13.234.3.50 RandomNullSpaceVector_modular_double() . . . 1017
13.234.3.51 NullSpaceBasis_modular_double() . . . 1017
13.234.3.52 RowRankProfile_modular_double() . . . 1018
13.234.3.53 ColumnRankProfile_modular_double() . . . 1018
13.234.3.54 RankProfileFromLU() . . . 1018
13.234.3.55 LeadingSubmatrixRankProfiles() . . . 1018
13.234.3.56 RowRankProfileSubmatrixIndices_modular_double() . . . 1018
13.234.3.57 ColRankProfileSubmatrixIndices_modular_double() . . . 1019
13.234.3.58 RowRankProfileSubmatrix_modular_double() . . . 1019
13.234.3.59 ColRankProfileSubmatrix_modular_double() . . . 1019
13.234.3.60 getTriangular_modular_double() . . . 1019
13.234.3.61 getTriangularin_modular_double() . . . 1020
13.234.3.62 getEchelonForm_modular_double() . . . 1020
13.234.3.63 getEchelonFormin_modular_double() . . . 1020
13.234.3.64 getEchelonTransform_modular_double() . . . 1020
13.234.3.65 getReducedEchelonForm_modular_double() . . . 1021
13.234.3.66 getReducedEchelonFormin_modular_double() . . . 1021
13.234.3.67 getReducedEchelonTransform_modular_double() . . . 1021
13.234.3.68 PLUQtoEchelonPermutation() . . . 1021

13.235 ffpack_inst.C File Reference . . . 1022
13.235.1 Macro Definition Documentation . . . 1022
13.235.1.1 __FFPACK_INST_C . . . 1022
13.235.1.2 FFLAS_COMPILED . . . 1022
13.235.1.3 INST_OR_DECL . . . 1022
13.235.1.4 FFLAS_FIELD [1/2] . . . 1022
13.235.1.5 FFLAS_ELT [1/6] . . . 1022
13.235.1.6 FFLAS_ELT [2/6] . . . 1022
13.235.1.7 FFLAS_ELT [3/6] . . . 1022
13.235.1.8 FFLAS_FIELD [2/2] . . . 1022
13.235.1.9 FFLAS_ELT [4/6] . . . 1023
13.235.1.10 FFLAS_ELT [5/6] . . . 1023
13.235.1.11 FFLAS_ELT [6/6] . . . 1023
13.236 ffpack_inst.h File Reference . . . 1023
13.236.1 Macro Definition Documentation . . . 1023
13.236.1.1 FFLAS_COMPILED . . . 1023
13.236.1.2 INST_OR_DECL . . . 1023
13.236.1.3 FFLAS_FIELD [1/2] . . . 1023
13.236.1.4 FFLAS_ELT [1/6] . . . 1023
13.236.1.5 FFLAS_ELT [2/6] . . . 1023
13.236.1.6 FFLAS_ELT [3/6] . . . 1023
13.236.1.7 FFLAS_FIELD [2/2] . . . 1024
13.236.1.8 FFLAS_ELT [4/6] . . . 1024
13.236.1.9 FFLAS_ELT [5/6] . . . 1024
13.236.1.10 FFLAS_ELT [6/6] . . . 1024
13.237 ffpack_inst_implem.inl File Reference . . . 1024
13.238 blockcuts.inl File Reference . . . 1027
13.238.1 Macro Definition Documentation . . . 1028
13.238.1.1 __FFLASFFPACK_fflas_blockcuts_INL . . . 1028
13.238.1.2 __FFLASFFPACK_MINBLOCKCUTS . . . 1028
13.239 fflas_plevel1.h File Reference . . . 1029
13.240 kaapi_routines.inl File Reference . . . 1029
13.240.1 Macro Definition Documentation . . . 1029
13.240.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL . . . 1029
13.241 parallel.h File Reference . . . 1029
13.241.1 Macro Definition Documentation . . . 1030
13.241.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1030
13.241.1.2 index_t . . . 1030
13.241.1.3 TASK . . . 1030
13.241.1.4 WAIT . . . 1030
13.241.1.5 CHECK_DEPENDENCIES . . . 1030
13.241.1.6 BARRIER . . . 1031

13.241.1.7 PAR_BLOCK . . . 1031
13.241.1.8 SYNCH_GROUP . . . 1031
13.241.1.9 THREAD_INDEX . . . 1031
13.241.1.10 NUM_THREADS . . . 1031
13.241.1.11 SET_THREADS . . . 1031
13.241.1.12 MAX_THREADS . . . 1031
13.241.1.13 READ . . . 1031
13.241.1.14 WRITE . . . 1031
13.241.1.15 READWRITE . . . 1031
13.241.1.16 CONSTREFERENCE . . . 1031
13.241.1.17 VALUE . . . 1031
13.241.1.18 BEGIN_PARALLEL_MAIN . . . 1032
13.241.1.19 END_PARALLEL_MAIN . . . 1032
13.241.1.20 FORBLOCK1D . . . 1032
13.241.1.21 FOR1D . . . 1032
13.241.1.22 PARFORBLOCK1D . . . 1032
13.241.1.23 PARFOR1D . . . 1032
13.241.1.24 FORBLOCK2D . . . 1033
13.241.1.25 FOR2D . . . 1033
13.241.1.26 PARFORBLOCK2D . . . 1033
13.241.1.27 PARFOR2D . . . 1033
13.241.1.28 COMMA . . . 1033
13.241.1.29 MODE . . . 1033
13.241.1.30 RETURNPARAM . . . 1034
13.241.1.31 NUMARGS . . . 1034
13.241.1.32 PP_NARG_ . . . 1034
13.241.1.33 PP_ARG_N . . . 1034
13.241.1.34 PP_RSEQ_N . . . 1035
13.241.1.35 NOSPLIT . . . 1035
13.241.1.36 splitting_0 . . . 1035
13.241.1.37 splitting_1 . . . 1036
13.241.1.38 splitting_2 . . . 1036
13.241.1.39 splitting_3 . . . 1036
13.241.1.40 splitt . . . 1036
13.241.1.41 SPLITTER . . . 1036
13.242 pfgemm_variants.inl File Reference . . . 1036
13.243 pfgemv.inl File Reference . . . 1037
13.244 align-allocator.h File Reference . . . 1037
13.245 args-parser.h File Reference . . . 1038
13.245.1 Macro Definition Documentation . . . 1038
13.245.1.1 TYPE_BOOL . . . 1038
13.245.1.2 END_OF_ARGUMENTS . . . 1038

13.245.1.3 type_integer . . . 1038
13.245.2 Enumeration Type Documentation . . . 1039
13.245.2.1 ArgumentType . . . 1039
13.245.3 Function Documentation . . . 1039
13.245.3.1 printHelpMessage() . . . 1039
13.245.3.2 findArgument() . . . 1039
13.245.3.3 getListArgs() . . . 1039
13.246 bit_manipulation.h File Reference . . . 1039
13.246.1 Macro Definition Documentation . . . 1040
13.246.1.1 __has_builtin . . . 1040
13.246.2 Function Documentation . . . 1040
13.246.2.1 clz() [1/2] . . . 1040
13.246.2.2 clz() [2/2] . . . 1040
13.246.2.3 ctz() [1/2] . . . 1040
13.246.2.4 ctz() [2/2] . . . 1040
13.247 cast.h File Reference . . . 1040
13.248 debug.h File Reference . . . 1040
13.248.1 Detailed Description . . . 1041
13.248.2 Macro Definition Documentation . . . 1041
13.248.2.1 FFLASFFPACK_check . . . 1041
13.248.2.2 FFLASFFPACK_abort . . . 1041
13.249 fflas_intrinsic.h File Reference . . . 1041
13.250 fflas_io.h File Reference . . . 1041
13.251 fflas_memory.h File Reference . . . 1042
13.252 fflas_randommatrix.h File Reference . . . 1043
13.253 flimits.h File Reference . . . 1045
13.253.1 Function Documentation . . . 1046
13.253.1.1 in_range() [1/3] . . . 1046
13.253.1.2 in_range() [2/3] . . . 1046
13.253.1.3 in_range() [3/3] . . . 1046
13.254 Matio.h File Reference . . . 1046
13.254.1 Function Documentation . . . 1046
13.254.1.1 read_field() . . . 1046
13.254.1.2 write_field() . . . 1046
13.255 test-utils.h File Reference . . . 1047
13.256 timer.h File Reference . . . 1047
13.257 cblas.C File Reference . . . 1048
13.257.1 Macro Definition Documentation . . . 1048
13.257.1.1 __FFLASFFPACK_CONFIGURATION . . . 1048
13.257.1.2 __FFLASFFPACK_HAVE_CBLAS . . . 1048
13.257.2 Function Documentation . . . 1048
13.257.2.1 main() . . . 1048

13.258 clapack.C File Reference . . . 1048
13.258.1 Macro Definition Documentation . . . 1048
13.258.1.1 __FFLASFFPACK_CONFIGURATION . . . 1048
13.258.1.2 __FFLASFFPACK_HAVE_LAPACK . . . 1048
13.258.1.3 __FFLASFFPACK_HAVE_CLAPACK . . . 1048
13.258.2 Function Documentation . . . 1049
13.258.2.1 main() . . . 1049
13.259 cuda.C File Reference . . . 1049
13.259.1 Function Documentation . . . 1049
13.259.1.1 main() . . . 1049
13.260 fblas.C File Reference . . . 1049
13.260.1 Macro Definition Documentation . . . 1049
13.260.1.1 __FFLASFFPACK_CONFIGURATION . . . 1049
13.260.2 Function Documentation . . . 1049
13.260.2.1 dgemm_() . . . 1049
13.260.2.2 main() . . . 1050
13.261 lapack.C File Reference . . . 1050
13.261.1 Macro Definition Documentation . . . 1050
13.261.1.1 __FFLASFFPACK_CONFIGURATION . . . 1050
13.261.1.2 __FFLASFFPACK_HAVE_LAPACK . . . 1050
13.261.2 Function Documentation . . . 1050
13.261.2.1 main() . . . 1050
13.262 regression-check.C File Reference . . . 1050
13.262.1 Function Documentation . . . 1051
13.262.1.1 check1() . . . 1051
13.262.1.2 check2() . . . 1051
13.262.1.3 check3() . . . 1051
13.262.1.4 check4() . . . 1051
13.262.1.5 checkZeroDimCharpoly() . . . 1051
13.262.1.6 checkZeroDimMinPoly() . . . 1051
13.262.1.7 gf2ModularBalanced() . . . 1051
13.262.1.8 main() . . . 1051
13.263 test-charpoly-check.C File Reference . . . 1051
13.263.1 Macro Definition Documentation . . . 1052
13.263.1.1 ENABLE_CHECKER_charpoly . . . 1052
13.263.1.2 TIME_CHECKER_CHARPOLY . . . 1052
13.263.2 Function Documentation . . . 1052
13.263.2.1 printPolynomial() . . . 1052
13.263.2.2 main() . . . 1052
13.264 test-charpoly.C File Reference . . . 1052
13.264.1 Function Documentation . . . 1052
13.264.1.1 launch_test() . . . 1052

13.264.1.2 run_with_field() . . . 1053
13.264.1.3 main() . . . 1053
13.265 test-compressQ.C File Reference . . . 1053
13.265.1 Typedef Documentation . . . 1053
13.265.1.1 Field . . . 1053
13.265.2 Function Documentation . . . 1053
13.265.2.1 printvect() . . . 1053
13.265.2.2 main() . . . 1054
13.266 test-det-check.C File Reference . . . 1054
13.266.1 Macro Definition Documentation . . . 1054
13.266.1.1 ENABLE_CHECKER_Det . . . 1054
13.266.1.2 TIME_CHECKER_Det . . . 1054
13.266.2 Function Documentation . . . 1054
13.266.2.1 main() . . . 1054
13.267 test-det.C File Reference . . . 1054
13.267.1 Function Documentation . . . 1055
13.267.1.1 test_det() . . . 1055
13.267.1.2 main() . . . 1055
13.268 test-echelon.C File Reference . . . 1055
13.268.1 Macro Definition Documentation . . . 1056
13.268.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1056
13.268.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE . . . 1056
13.268.1.3 __FFLASFFPACK_PLUQ_THRESHOLD . . . 1056
13.268.2 Function Documentation . . . 1056
13.268.2.1 test_colechelon() . . . 1056
13.268.2.2 test_rowechelon() . . . 1056
13.268.2.3 test_redcolechelon() . . . 1056
13.268.2.4 test_redrowechelon() . . . 1057
13.268.2.5 run_with_field() . . . 1057
13.268.2.6 main() . . . 1057
13.269 test-fadd.C File Reference . . . 1057
13.269.1 Function Documentation . . . 1058
13.269.1.1 test_fadd() . . . 1058
13.269.1.2 test_faddin() . . . 1058
13.269.1.3 test_fsub() . . . 1058
13.269.1.4 test_fsubin() . . . 1058
13.269.1.5 main() . . . 1059
13.270 test-fdot.C File Reference . . . 1059
13.270.1 Macro Definition Documentation . . . 1059
13.270.1.1 ENABLE_ALL_CHECKINGS . . . 1059
13.270.2 Function Documentation . . . 1059
13.270.2.1 check_fdot() . . . 1059

13.270.2.2 run_with_field() . . . 1059
13.270.2.3 run_with_Integer() . . . 1060
13.270.2.4 main() . . . 1060
13.271 test-fgemm-check.C File Reference . . . 1060
13.271.1 Macro Definition Documentation . . . 1060
13.271.1.1 ENABLE_ALL_CHECKINGS . . . 1060
13.271.2 Function Documentation . . . 1060
13.271.2.1 launch_MM_dispatch() . . . 1060
13.271.2.2 run_with_field() . . . 1061
13.271.2.3 main() . . . 1061
13.272 test-fgemm.C File Reference . . . 1061
13.272.1 Macro Definition Documentation . . . 1062
13.272.1.1 ENABLE_CHECKER_fgemm . . . 1062
13.272.2 Function Documentation . . . 1062
13.272.2.1 check_MM() . . . 1062
13.272.2.2 launch_MM() . . . 1062
13.272.2.3 launch_MM_dispatch() . . . 1063
13.272.2.4 run_with_field() . . . 1063
13.272.2.5 main() . . . 1063
13.273 test-fgemv.C File Reference . . . 1063
13.273.1 Function Documentation . . . 1064
13.273.1.1 check_MV() . . . 1064
13.273.1.2 launch_MV() . . . 1064
13.273.1.3 launch_MV_dispatch() . . . 1064
13.273.1.4 run_with_field() . . . 1065
13.273.1.5 main() . . . 1065
13.274 test-fger.C File Reference . . . 1065
13.274.1 Macro Definition Documentation . . . 1066
13.274.1.1 TIME . . . 1066
13.274.2 Function Documentation . . . 1066
13.274.2.1 check_fger() . . . 1066
13.274.2.2 launch_fger() . . . 1066
13.274.2.3 launch_fger_dispatch() . . . 1066
13.274.2.4 run_with_field() . . . 1067
13.274.2.5 main() . . . 1067
13.275 test-fgesv.C File Reference . . . 1067
13.275.1 Function Documentation . . . 1067
13.275.1.1 test_square_fgesv() . . . 1067
13.275.1.2 test_rect_fgesv() . . . 1067
13.275.1.3 run_with_field() . . . 1068
13.275.1.4 main() . . . 1068
13.276 test-finit.C File Reference . . . 1068

13.276.1 Function Documentation . . . 1069
13.276.1.1 test_freduce() . . . 1069
13.276.1.2 run_with_field() . . . 1069
13.276.1.3 main() . . . 1069
13.277 test-fscal.C File Reference . . . 1069
13.277.1 Function Documentation . . . 1070
13.277.1.1 test_fscal() [1/2] . . . 1070
13.277.1.2 test_fscal() [2/2] . . . 1070
13.277.1.3 test_fscalin() [1/2] . . . 1070
13.277.1.4 test_fscalin() [2/2] . . . 1070
13.277.1.5 main() . . . 1070
13.277.1.6 RandomMatrix() . . . 1071
13.278 test-fsyr2k.C File Reference . . . 1071
13.278.1 Macro Definition Documentation . . . 1071
13.278.1.1 ENABLE_ALL_CHECKINGS . . . 1071
13.278.2 Function Documentation . . . 1072
13.278.2.1 check_fsyr2k() . . . 1072
13.278.2.2 run_with_field() . . . 1072
13.278.2.3 main() . . . 1072
13.279 test-fsyrk.C File Reference . . . 1072
13.279.1 Macro Definition Documentation . . . 1073
13.279.1.1 ENABLE_ALL_CHECKINGS . . . 1073
13.279.2 Function Documentation . . . 1073
13.279.2.1 check_fsyrk() . . . 1073
13.279.2.2 check_fsyrk_diag() . . . 1073
13.279.2.3 check_fsyrk_bkdiag() . . . 1073
13.279.2.4 check_computeS1S2() . . . 1074
13.279.2.5 run_with_field() . . . 1074
13.279.2.6 main() . . . 1074
13.280 test-fsytrf.C File Reference . . . 1074
13.280.1 Function Documentation . . . 1075
13.280.1.1 operator<<() . . . 1075
13.280.1.2 test_RPM_fsytrf() . . . 1075
13.280.1.3 test_generic_fsytrf() . . . 1075
13.280.1.4 run_with_field() . . . 1075
13.280.1.5 main() . . . 1075
13.281 test-ftrmm.C File Reference . . . 1075
13.281.1 Macro Definition Documentation . . . 1076
13.281.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1076
13.281.2 Function Documentation . . . 1076
13.281.2.1 check_ftrmm() . . . 1076
13.281.2.2 run_with_field() . . . 1076

13.281.2.3 main() . . . 1077
13.282 test-ftrmv.C File Reference . . . 1077
13.282.1 Macro Definition Documentation . . . 1077
13.282.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1077
13.282.1.2 ENABLE_ALL_CHECKINGS . . . 1077
13.282.2 Function Documentation . . . 1077
13.282.2.1 check_ftrmv() . . . 1077
13.282.2.2 run_with_field() . . . 1078
13.282.2.3 main() . . . 1078
13.283 test-ftrsm-check.C File Reference . . . 1078
13.283.1 Macro Definition Documentation . . . 1078
13.283.1.1 ENABLE_ALL_CHECKINGS . . . 1078
13.283.2 Function Documentation . . . 1078
13.283.2.1 main() . . . 1078
13.284 test-ftrsm.C File Reference . . . 1078
13.284.1 Macro Definition Documentation . . . 1079
13.284.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1079
13.284.1.2 ENABLE_ALL_CHECKINGS . . . 1079
13.284.2 Function Documentation . . . 1079
13.284.2.1 check_ftrsm() . . . 1079
13.284.2.2 run_with_field() . . . 1079
13.284.2.3 main() . . . 1080
13.285 test-ftrssyr2k.C File Reference . . . 1080
13.285.1 Macro Definition Documentation . . . 1080
13.285.1.1 ENABLE_ALL_CHECKINGS . . . 1080
13.285.2 Function Documentation . . . 1080
13.285.2.1 check_ftrssyr2k() . . . 1080
13.285.2.2 run_with_field() . . . 1080
13.285.2.3 main() . . . 1081
13.286 test-ftrstr.C File Reference . . . 1081
13.286.1 Macro Definition Documentation . . . 1081
13.286.1.1 ENABLE_ALL_CHECKINGS . . . 1081
13.286.2 Function Documentation . . . 1081
13.286.2.1 check_ftrstr() . . . 1081
13.286.2.2 run_with_field() . . . 1082
13.286.2.3 main() . . . 1082
13.287 test-ftrsv.C File Reference . . . 1082
13.287.1 Macro Definition Documentation . . . 1082
13.287.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1082
13.287.1.2 ENABLE_ALL_CHECKINGS . . . 1082
13.287.2 Function Documentation . . . 1082
13.287.2.1 check_ftrsv() . . . 1082

13.287.2.2 run_with_field() . . . 1083
13.287.2.3 main() . . . 1083
13.288 test-ftrtri.C File Reference . . . 1083
13.288.1 Macro Definition Documentation . . . 1083
13.288.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1083
13.288.1.2 ENABLE_ALL_CHECKINGS . . . 1084
13.288.2 Function Documentation . . . 1084
13.288.2.1 check_ftrtri() . . . 1084
13.288.2.2 run_with_field() . . . 1084
13.288.2.3 main() . . . 1084
13.289 test-interfaces-c.c File Reference . . . 1084
13.289.1 Function Documentation . . . 1084
13.289.1.1 main() . . . 1084
13.290 test-invert-check.C File Reference . . . 1084
13.290.1 Macro Definition Documentation . . . 1085
13.290.1.1 ENABLE_ALL_CHECKINGS . . . 1085
13.290.2 Function Documentation . . . 1085
13.290.2.1 main() . . . 1085
13.291 test-io.C File Reference . . . 1085
13.291.1 Function Documentation . . . 1085
13.291.1.1 run_with_field() . . . 1085
13.291.1.2 main() . . . 1086
13.292 test-lu.C File Reference . . . 1086
13.292.1 Macro Definition Documentation . . . 1087
13.292.1.1 BASECASE_K . . . 1087
13.292.1.2 __FFLASFFPACK_SEQUENTIAL . . . 1087
13.292.1.3 __LUDIVINE_CUTOFF . . . 1087
13.292.2 Function Documentation . . . 1087
13.292.2.1 test_LUdivine() . . . 1087
13.292.2.2 verifPLUQ() . . . 1088
13.292.2.3 test_pluq() . . . 1088
13.292.2.4 launch_test() . . . 1089
13.292.2.5 run_with_field() . . . 1089
13.292.2.6 main() . . . 1089
13.292.3 Variable Documentation . . . 1089
13.292.3.1 tperm . . . 1089
13.292.3.2 tgemm . . . 1089
13.292.3.3 tBC . . . 1089
13.292.3.4 ttrsm . . . 1089
13.292.3.5 trest . . . 1090
13.292.3.6 timtot . . . 1090
13.292.3.7 mvcnt . . . 1090

13.293 test-maxdelayeddim.C File Reference . . . 1090
13.293.1 Macro Definition Documentation . . . 1090
13.293.1.1 MAX_WITH_SIZE_T . . . 1090
13.293.2 Function Documentation . . . 1090
13.293.2.1 test() . . . 1090
13.293.2.2 main() . . . 1090
13.294 test-minpoly.C File Reference . . . 1090
13.294.1 Function Documentation . . . 1091
13.294.1.1 check_minpoly() . . . 1091
13.294.1.2 run_with_field() . . . 1091
13.294.1.3 main() . . . 1091
13.295 test-multifile1.C File Reference . . . 1091
13.296 test-multifile2.C File Reference . . . 1091
13.296.1 Function Documentation . . . 1092
13.296.1.1 main() . . . 1092
13.297 test-nullspace.C File Reference . . . 1092
13.297.1 Function Documentation . . . 1092
13.297.1.1 checkingMessage() . . . 1092
13.297.1.2 readOrRandomMatrixWithRankAndRandomRPM() . . . 1092
13.297.1.3 test_nullspace() . . . 1093
13.297.1.4 run_with_field() . . . 1093
13.297.1.5 main() . . . 1093
13.298 test-permutations.C File Reference . . . 1093
13.298.1 Function Documentation . . . 1094
13.298.1.1 checkMonotonicApplyP() . . . 1094
13.298.1.2 main() . . . 1094
13.298.2 Variable Documentation . . . 1094
13.298.2.1 tperm . . . 1094
13.298.2.2 tgemm . . . 1094
13.298.2.3 tBC . . . 1094
13.298.2.4 ttrsm . . . 1094
13.298.2.5 trest . . . 1094
13.298.2.6 timtot . . . 1094
13.299 test-pluq-check.C File Reference . . . 1094
13.299.1 Macro Definition Documentation . . . 1095
13.299.1.1 ENABLE_ALL_CHECKINGS . . . 1095
13.299.2 Function Documentation . . . 1095
13.299.2.1 main() . . . 1095
13.300 test-quasisep.C File Reference . . . 1095
13.300.1 Function Documentation . . . 1095
13.300.1.1 test_BruhatGenerator() . . . 1095
13.300.1.2 launch_test() . . . 1096

13.300.1.3 testLTQSRPM() . . . 1096
13.300.1.4 run_with_field() . . . 1096
13.300.1.5 main() . . . 1096
13.301 test-rankprofiles.C File Reference . . . 1096
13.301.1 Macro Definition Documentation . . . 1097
13.301.1.1 __FFLASFFPACK_SEQUENTIAL . . . 1097
13.301.2 Function Documentation . . . 1097
13.301.2.1 run_with_field() . . . 1097
13.301.2.2 main() . . . 1097
13.302 test-rpm.C File Reference . . . 1097
13.302.1 Function Documentation . . . 1097
13.302.1.1 checkRPM() . . . 1097
13.302.1.2 checkSymmetricRPM() . . . 1098
13.302.1.3 main() . . . 1098
13.303 test-simd.C File Reference . . . 1098
13.303.1 Macro Definition Documentation . . . 1099
13.303.1.1 _TEST_ONE . . . 1099
13.303.1.2 TEST_ONE_OP . . . 1099
13.303.1.3 TEST_ONE_OP_WZ . . . 1100
13.303.1.4 TEST_IMPL . . . 1100
13.303.2 Function Documentation . . . 1100
13.303.2.1 check_eq() [1/2] . . . 1100
13.303.2.2 check_eq() [2/2] . . . 1100
13.303.2.3 cmp() . . . 1100
13.303.2.4 eval_func_on_array() [1/3] . . . 1100
13.303.2.5 eval_func_on_array() [2/3] . . . 1100
13.303.2.6 eval_func_on_array() [3/3] . . . 1100
13.303.2.7 operator<<() . . . 1101
13.303.2.8 test_impl_base() [1/2] . . . 1101
13.303.2.9 test_impl_base() [2/2] . . . 1101
13.303.2.10 test_impl() . . . 1101
13.303.2.11 main() . . . 1101
13.304 test-solve.C File Reference . . . 1101
13.304.1 Function Documentation . . . 1101
13.304.1.1 check_solve() . . . 1101
13.304.1.2 run_with_field() . . . 1102
13.304.1.3 main() . . . 1102
13.305 test-storage-transpose.C File Reference . . . 1102
13.305.1 Function Documentation . . . 1102
13.305.1.1 main() . . . 1102
13.306 101-fgemm.C File Reference . . . 1102
13.306.1 Function Documentation . . . 1103

13.306.1.1 main() . . . 1103
13.307 2x2-fgemm.C File Reference . . . 1103
13.307.1 Function Documentation . . . 1103
13.307.1.1 main() . . . 1103
13.308 2x2-ftrsv.C File Reference . . . 1103
13.308.1 Function Documentation . . . 1104
13.308.1.1 main() . . . 1104
13.309 2x2-pluq.C File Reference . . . 1104
13.309.1 Function Documentation . . . 1104
13.309.1.1 main() . . . 1104
13.310 fflas-101_1.C File Reference . . . 1104
13.310.1 Function Documentation . . . 1104
13.310.1.1 main() . . . 1104
13.311 fflas-101_3.C File Reference . . . 1104
13.311.1 Function Documentation . . . 1105
13.311.1.1 main() . . . 1105
13.312 fflas_101.C File Reference . . . 1105
13.312.1 Function Documentation . . . 1105
13.312.1.1 main() . . . 1105
13.313 fflas_101_lvl1.C File Reference . . . 1105
13.313.1 Function Documentation . . . 1105
13.313.1.1 main() . . . 1105
13.314 ffpack-fgesv.C File Reference . . . 1106
13.314.1 Function Documentation . . . 1106
13.314.1.1 main() . . . 1106
13.315 ffpack-solve.C File Reference . . . 1106
13.315.1 Function Documentation . . . 1106
13.315.1.1 main() . . . 1106

Index **1107**

Chapter 1

FFLAS-FFPACK Documentation.

1.1 Introduction

FFLAS-FFPACK is a LGPL-2.1+ source code library for basic linear algebra operations over a finite field. It is inspired by BLAS interface (Basic Linear Algebra Subprograms) and the LAPACK library for numerical linear algebra, and shares part of their design. Yet it differs in many aspects due to the specifities of computing over a finite field:

- it is generic with respect to the finite field, so as to accomodate a large variety of field sizes and implementations;
- it is a pure source code library, to be included and compiled in the user's software. Its build system is only used for tests and benchmarks.

1.2 Goals

1.3 Design

1.4 Using FFLAS-FFPACK.

- Copying and Licence.
- Tutorial. This is a brief introduction to FFLAS-FFPACK capabilities.
- Configuring and Installing FFLAS-FFPACK. Explains how to configure/install from sources or from the latest svn version.
- Architecture of the library.. Describes how FFLAS-FFPACK is organized
- `Documentation for Users`. If everything around is blue, then you are reading the lighter, user-oriented, documentation.
- `Documentation for Developers`. If everything around is green, then you can get to everything (not necessarily yet) documented.

1.5 Contributing to fflas-ffpack, getting assistance.

Version

2.5.0

1.6 Copying and Licence

The FFLAS-FFPACK library is licensed under the terms of the GNU LGPL v2.1 or later.

See https://www.gnu.org/licenses/lgpl-2.1.html

1.7 Tutorial

no doc.

1.8 Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Hower configuration process can be tweaked a lot. Configure looks for BLAS routines and Givaro library which are both mandatory dependencies. See the output of `./configure -help` for information about the LAPACK/↩ BLAS discovering strategies.

1.9 Architecture of the library.

no doc.

Chapter 2

Bug List

Global DOUBLE_TO_FLOAT_CROSSOVER

to be benchmarked.

Global FFLAS::details::pack_lhs (int64_t ∗XX, const int64_t ∗X, size_t ldx, size_t rows, size_t cols)

this is fassign

this is fassign

Global FFLAS::details::pack_rhs (int64_t ∗XX, const int64_t ∗X, size_t ldx, size_t rows, size_t cols)

this is fassign

this is fassign

Global FFLAS::fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

use cblas_(d)scal when possible

Global FFLAS::fconvert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT ∗X, const size_t incX, const FFLAS_ELT ∗Y, const size_t incY)

use cblas_(d)scal when possible

Global FFLAS::finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT ∗Y, const size_t incY, FFLAS_ELT ∗X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT ∗Y, const size_t incY, FFLAS_ELT ∗X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

use cblas_(d)scal when possible

Global FFLAS::fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

use cblas_(d)scal when possible

Global FFLAS::fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global FFLAS::fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

why double ?

Global FFLAS::fswap (const Field &F, const size_t N, typename Field::Element_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

use cblas_dswap when double

Global FFLAS::fswap (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

use cblas_dswap when double

Global FFLAS::ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

α must be non zero.

Global FFLAS::ftrsm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

α must be non zero.

Global FFPACK::buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)

is this :

Global FFPACK::Invert2 (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

not tested.

Global launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const size_t iters, RandIter &G)

test for incx equal

test for transpo

Global launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, RandIter &G)

test for ldX equal

test for transpo

Global launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const int nbw, const bool par, RandIter &G)

test for ldX equal

test for transpo

Global printvect (std::ostream &o, vector< T > &vect)

does not belong here

Chapter 3

Bibliography

Global FFLAS::Protected::TRSMBound (const Givaro::ModularBalanced< Element > &F)

- Dumas Giorgi Pernet 06, arXiv:cs/0601133

Global FFPACK::LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles* , ISSAC'13.

Global FFPACK::LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition* , J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

Global FFPACK::PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles* , ISSAC'13, 2013

Global FFPACK::productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

Pernet C. and Storjohann A. *Time and space efficient generators for quasiseparable matrices* , JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010

Global FFPACK::Protected::GaussJordan (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

- Algorithm 2.8 of A. Storjohann Thesis 2000,
- Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. `Rank-profile` *revealing Gaussian elimination and the CUP matrix decomposition* , J. of Symbolic Comp., 2013

Class ftrsmLeftUpperNoTransNonUnit< Element >

- Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Chapter 4

Todo List

File debug.h

we should put vector printing elsewhere.

Global FFLAS::fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)

optimise here

Global FFLAS::fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

variant for triagular matrix

Global FFLAS::fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

variant for triagular matrix

Global FFLAS::fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)

check if n == lda

Global FFLAS::fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

check if n == lda

Global FFLAS::fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

check if n == lda

Global FFLAS::fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

check if comparison with +/-1,0 is necessary.

Global FFLAS::fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

check if comparison with +/-1,0 is necessary.

Global FFLAS::fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

check if comparison with +/-1,0 is necessary.

Global FFLAS::fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

check if comparison with +/-1,0 is necessary.

Global FFLAS::Protected::igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)

use primitive (no Field()) and specialise for int64.

Global FFLAS::Protected::MatF2MatFl_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)

do finit(...,FFLAS_TRANS,FFLAS_DIAG)

do fconvert(...,FFLAS_TRANS,FFLAS_DIAG)

Global FFPACK::getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)

just one triangular fzero+fassign ?

Global FFPACK::getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)

just one triangular fzero+fassign ?

Global FFPACK::Invert2 (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

this init is not all necessary (done after ftrtri)

Global FFPACK::LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)

std::swap ?

Global FFPACK::Protected::RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::↩ Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing↩ ::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

swap to save space ??

don't assing K2 c*noc x N but only mas (c,noc) x N and store each one after the other

Module field

biblio

Global launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const size_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const int nbw, const bool par, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Module MMalgos

biblio

Module simd

biblio

Global test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check lda

Global test_det (Field &F, size_t n, int iter, RandIter &G)

test with stride

Global test_redcolechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check lda

Global test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check lda

Global test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check lda

Chapter 5

Topic Index

5.1 Topics

Here is a list of all topics with brief descriptions:

CHECKER 41
FFLAS-FFPACK 41
FFLAS 42
Interfaces 42
Matrix Multiplication Algorithms 42
SIMD wrapper 42
FFPACK 43
FFLAS-FFPACK fields 43
RNS 43

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

FFLAS . . . 45
FFLAS::_ftranspose_impl . . . 190
FFLAS::BLAS3 . . . 191
FFLAS::csr_hyb_details . . . 197
FFLAS::CuttingStrategy . . . 197
FFLAS::details . . . 198
FFLAS::details_spmv . . . 206
FFLAS::ElementCategories . . . 206
FFLAS::FieldCategories
Traits and categories will need to be placed in a proper file later . . . 206
FFLAS::MMHelperAlgo . . . 207
FFLAS::ModeCategories
Specifies the mode of action for an algorithm w.r.t . . . 207
FFLAS::ParSeqHelper
ParSeqHelper for both fgemm and ftrsm . . . 208
FFLAS::Protected . . . 208
FFLAS::sell_details . . . 222
FFLAS::sparse_details . . . 223
FFLAS::sparse_details_impl . . . 237
FFLAS::StrategyParameter . . . 277
FFLAS::StructureHelper
StructureHelper for ftrsm . . . 277
FFLAS::vectorised . . . 278
FFLAS::vectorised::unswitch . . . 283
FFPACK
Finite **F**ield **PACK** Set of elimination based routines for dense linear algebra . . . 286
FFPACK::Protected . . . 389
Givaro . . . 396
MKL_CONFIG . . . 396
RecInt . . . 396

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AlgoChooser< ModeT, ParSeq > . . . 397
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > . . . 397
ALL< v > . . . 397
ALL< false, v... > . . . 398
ALL< true, v... > . . . 398
ALL<> . . . 398
ArbitraryPrecIntTag . . . 398
AreEqual< X, Y > . . . 399
AreEqual< X, X > . . . 399
Argument . . . 399
array< T > . . . 400
associatedDelayedField< Field > . . . 400
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > . . . 401
associatedDelayedField< const Givaro::Modular< T, X > > . . . 401
associatedDelayedField< const Givaro::ModularBalanced< T > > . . . 402
associatedDelayedField< const Givaro::ZRing< T > > . . . 403
Auto . . . 403
Bench< Elt > . . . 403
Bini . . . 406
Block . . . 406
BlockTransposeSIMD< Field, Simd, > . . . 406
callLUdivine_small< Element > . . . 408
callLUdivine_small< double > . . . 409
callLUdivine_small< float > . . . 410
CharpolyFailed . . . 410
Checker_Empty< Field > . . . 410
CheckerImplem_charpoly< Field, Polynomial > . . . 411
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > . . . 412
CheckerImplem_Det< Field > . . . 413
CheckerImplem_fgemm< Field > . . . 414
CheckerImplem_ftrsm< Field > . . . 416
CheckerImplem_invert< Field > . . . 417
CheckerImplem_PLUQ< Field > . . . 417
Classic . . . 418
Column . . . 419

CompactElement< Element > . . . 419
CompactElement< double > . . . 419
CompactElement< float > . . . 419
CompactElement< int16_t > . . . 420
CompactElement< int32_t > . . . 420
CompactElement< int64_t > . . . 420
compatible_data_type< Field > . . . 421
compatible_data_type< Givaro::ZRing< double > > . . . 421
compatible_data_type< Givaro::ZRing< float > > . . . 421
Compose< H1, H2 > . . . 422
array< T >::const_iterator . . . 423
string::const_iterator . . . 423
vector< T >::const_iterator . . . 423
array< T >::const_reverse_iterator . . . 423
string::const_reverse_iterator . . . 423
vector< T >::const_reverse_iterator . . . 424
Simd128_impl< true, true, false, 2 >::Converter . . . 424
Simd128_impl< true, true, false, 4 >::Converter . . . 424
Simd128_impl< true, true, false, 8 >::Converter . . . 424
Simd128_impl< true, true, true, 2 >::Converter . . . 425
Simd128_impl< true, true, true, 4 >::Converter . . . 425
Simd128_impl< true, true, true, 8 >::Converter . . . 425
Simd256_impl< true, false, true, 8 >::Converter . . . 426
Simd256_impl< true, true, false, 2 >::Converter . . . 426
Simd256_impl< true, true, false, 4 >::Converter . . . 426
Simd256_impl< true, true, false, 8 >::Converter . . . 427
Simd256_impl< true, true, true, 2 >::Converter . . . 427
Simd256_impl< true, true, true, 4 >::Converter . . . 427
Simd256_impl< true, true, true, 8 >::Converter . . . 428
Simd512_impl< true, true, false, 8 >::Converter . . . 428
Simd512_impl< true, true, true, 8 >::Converter . . . 428
ConvertTo< T > . . . 429
Coo< ValT, IdxT > . . . 429
Coo< Field > . . . 431
Coo< ValT, IdxT > . . . 432
CooMat< Field > . . . 434
count_nonconst_lvalue_reference< T > . . . 435
count_nonconst_lvalue_reference< const T &, O... > . . . 435
count_nonconst_lvalue_reference< T &, O... > . . . 435
count_nonconst_lvalue_reference< T, O... > . . . 435
count_nonconst_lvalue_reference<> . . . 436
CsrMat< Field > . . . 436
DefaultBoundedTag . . . 437
DefaultTag . . . 437
DelayedTag . . . 437
DivideAndConquer . . . 437
ElementTraits< Element > . . . 437
ElementTraits< double > . . . 438
ElementTraits< FFPACK::rns_double_elt > . . . 438
ElementTraits< float > . . . 438
ElementTraits< Givaro::Integer > . . . 439
ElementTraits< int16_t > . . . 439
ElementTraits< int32_t > . . . 440
ElementTraits< int64_t > . . . 440
ElementTraits< int8_t > . . . 440
ElementTraits< RecInt::rint< K > > . . . 441
ElementTraits< RecInt::rmint< K, MG > > . . . 441
ElementTraits< RecInt::ruint< K > > . . . 441

ElementTraits< uint16_t > . . . 442
ElementTraits< uint32_t > . . . 442
ElementTraits< uint64_t > . . . 443
ElementTraits< uint8_t > . . . 443
EllMat< Field > . . . 443
Failure . . . 444
FailureCharpolyCheck . . . 446
FailureDetCheck . . . 446
FailureFgemmCheck . . . 446
FailureInvertCheck . . . 446
FailurePLUQCheck . . . 446
FailureTrsmCheck . . . 446
false_type
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > . . . 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 487
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 490
support_fast_mod< double > . . . 755
support_fast_mod< float > . . . 756
support_fast_mod< int64_t > . . . 756
isSparseMatrix< Field, M > . . . 484
isSparseMatrixMKLFormat< F, M > . . . 488
isSparseMatrixSimdFormat< F, M > . . . 488
isZOSparseMatrix< F, M > . . . 488
support_fast_mod< T > . . . 755
support_simd< T > . . . 756
support_simd_add< T > . . . 756
support_simd_mod< T > . . . 757
FieldSimd< _Field > . . . 447
FieldTraits< Field > . . . 453
FieldTraits< FFPACK::RNSInteger< T > > . . . 453
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . 454
FieldTraits< Givaro::Modular< Element > > . . . 455
FieldTraits< Givaro::ModularBalanced< Element > > . . . 455
FieldTraits< Givaro::ZRing< double > > . . . 456
FieldTraits< Givaro::ZRing< float > > . . . 457
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . 457
FieldTraits< Givaro::ZRing< int16_t > > . . . 458
FieldTraits< Givaro::ZRing< int32_t > > . . . 459
FieldTraits< Givaro::ZRing< int64_t > > . . . 459
FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > . . . 460
FieldTraits< Givaro::ZRing< uint16_t > > . . . 460
FieldTraits< Givaro::ZRing< uint32_t > > . . . 461
FieldTraits< Givaro::ZRing< uint64_t > > . . . 462
Fixed . . . 462

FixedPrecIntTag 462
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >↩
::FloatingPointTestDistribution 463
ForStrategy1D< blocksize_t, Cut, Param > 463
ForStrategy2D< blocksize_t, Cut, Param > 466
ftrmmLeftLowerNoTransNonUnit< Element > 470
ftrmmLeftLowerNoTransUnit< Element > 470
ftrmmLeftLowerTransNonUnit< Element > 470
ftrmmLeftLowerTransUnit< Element > 470
ftrmmLeftUpperNoTransNonUnit< Element > 470
ftrmmLeftUpperNoTransUnit< Element > 470
ftrmmLeftUpperTransNonUnit< Element > 470
ftrmmLeftUpperTransUnit< Element > 471
ftrmmRightLowerNoTransNonUnit< Element > 471
ftrmmRightLowerNoTransUnit< Element > 471
ftrmmRightLowerTransNonUnit< Element > 471
ftrmmRightLowerTransUnit< Element > 471
ftrmmRightUpperNoTransNonUnit< Element > 471
ftrmmRightUpperNoTransUnit< Element > 471
ftrmmRightUpperTransNonUnit< Element > 471
ftrmmRightUpperTransUnit< Element > 472
ftrsmLeftLowerNoTransNonUnit< Element > 472
ftrsmLeftLowerNoTransUnit< Element > 472
ftrsmLeftLowerTransNonUnit< Element > 472
ftrsmLeftLowerTransUnit< Element > 472
ftrsmLeftUpperNoTransNonUnit< Element > 472
ftrsmLeftUpperNoTransUnit< Element > 473
ftrsmLeftUpperTransNonUnit< Element > 473
ftrsmLeftUpperTransUnit< Element > 473
ftrsmRightLowerNoTransNonUnit< Element > 473
ftrsmRightLowerNoTransUnit< Element > 473
ftrsmRightLowerTransNonUnit< Element > 473
ftrsmRightLowerTransUnit< Element > 473
ftrsmRightUpperNoTransNonUnit< Element > 473
ftrsmRightUpperNoTransUnit< Element > 474
ftrsmRightUpperTransNonUnit< Element > 474
ftrsmRightUpperTransUnit< Element > 474
GenericTag 474
GenericTag 474
Grain 474
has_minus_eq_impl< C > 474
has_minus_impl< C > 475
has_mul_eq_impl< C > 475
has_mul_impl< C > 475
has_operation< T > 476
has_plus_eq_impl< C > 476
has_plus_impl< C > 476
HelperFlag 477
HelperMod< Field, ElementTraits > 477
HelperMod< Field, ElementCategories::MachineIntTag > 478
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > 479
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > 479
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > 480
Hybrid 481
Info 481
Info 482
is_all_same< Args > 483
is_all_same< T, Args... > 483

is_all_same<> . . . 483
is_simd< T > . . . 484
Iterative . . . 490
array< T >::iterator . . . 490
string::iterator . . . 490
vector< T >::iterator . . . 490
LazyTag . . . 490
limits< T > . . . 491
limits< char > . . . 491
limits< double > . . . 491
limits< float > . . . 492
limits< Givaro::Integer > . . . 493
limits< int > . . . 493
limits< long > . . . 494
limits< long long > . . . 494
limits< RecInt::rint< K > > . . . 495
limits< RecInt::ruint< K > > . . . 496
limits< short int > . . . 496
limits< signed char > . . . 497
limits< unsigned char > . . . 498
limits< unsigned int > . . . 498
limits< unsigned long > . . . 499
limits< unsigned long long > . . . 499
limits< unsigned short int > . . . 500
MachineFloatTag . . . 501
MachineIntTag . . . 501
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > . . . 501
ModeTraits< Field > . . . 506
ModeTraits< Givaro::Modular< Element, Compute > > . . . 507
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > . . . 507
ModeTraits< Givaro::Modular< int16_t, Compute > > . . . 507
ModeTraits< Givaro::Modular< int32_t, Compute > > . . . 508
ModeTraits< Givaro::Modular< int64_t, uint64_t > > . . . 508
ModeTraits< Givaro::Modular< int8_t, Compute > > . . . 509
ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > . . . 509
ModeTraits< Givaro::Modular< uint16_t, Compute > > . . . 509
ModeTraits< Givaro::Modular< uint32_t, Compute > > . . . 510
ModeTraits< Givaro::Modular< uint8_t, Compute > > . . . 510
ModeTraits< Givaro::ModularBalanced< Element > > . . . 511
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . 511
ModeTraits< Givaro::ModularBalanced< int16_t > > . . . 511
ModeTraits< Givaro::ModularBalanced< int32_t > > . . . 512
ModeTraits< Givaro::ModularBalanced< int8_t > > . . . 512
ModeTraits< Givaro::Montgomery< T > > . . . 513
ModeTraits< Givaro::ZRing< double > > . . . 513
ModeTraits< Givaro::ZRing< float > > . . . 513
ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . 514
ModularBalanced< T > . . . 514
ModularBalanced< T > . . . 514
ModularTag . . . 514
Montgomery< T > . . . 514
need_field_characteristic< Field > . . . 515
need_field_characteristic< Givaro::Modular< Field > > . . . 515
need_field_characteristic< Givaro::ModularBalanced< Field > > . . . 515
NoSimd< T > . . . 516
Parallel< C, P > . . . 517
readMyMachineType< Field, T > . . . 520
readMyMachineType< Field, mpz_t > . . . 521

Recursive . . . 522
Recursive . . . 522
array< T >::reverse_iterator . . . 522
string::reverse_iterator . . . 523
vector< T >::reverse_iterator . . . 523
rint< K > . . . 523
rns_double . . . 523
rns_double_elt . . . 528
  rns_double_elt_cstptr . . . 529
  rns_double_elt_ptr . . . 532
rns_double_extended . . . 535
RNSElementTag . . . 538
RNSInteger< RNS > . . . 538
RNSIntegerMod< RNS > . . . 542
rnsRandIter< RNS > . . . 549
  RNSInteger< RNS >::RandIter . . . 518
  RNSIntegerMod< RNS >::RandIter . . . 519
Row . . . 550
ruint< K > . . . 550
ScalFunctionsBase< Element, Enable > . . . 555
  ScalFunctions< Element > . . . 550
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > . . . 556
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > . . . 557
Sequential . . . 560
Simd128_impl< ArithType, Int, Signed, Size > . . . 561
Simd128_impl< true, false, true, 4 > . . . 561
Simd128_impl< true, false, true, 8 > . . . 561
Simd128i_base . . . 619
  Simd128_impl< true, true, true, 2 > . . . 591
    Simd128_impl< true, true, false, 2 > . . . 561
  Simd128_impl< true, true, true, 4 > . . . 601
    Simd128_impl< true, true, false, 4 > . . . 571
  Simd128_impl< true, true, true, 8 > . . . 609
    Simd128_impl< true, true, false, 8 > . . . 581
Simd256_impl< ArithType, Int, Signed, Size > . . . 620
Simd256fp_base . . . 702
  Simd256_impl< true, false, true, 4 > . . . 620
  Simd256_impl< true, false, true, 8 > . . . 621
Simd256i_base . . . 702
  Simd256_impl< true, true, true, 2 > . . . 666
    Simd256_impl< true, true, false, 2 > . . . 628
  Simd256_impl< true, true, true, 4 > . . . 675
    Simd256_impl< true, true, false, 4 > . . . 638
    Simd256_impl< true, true, false, 4 > . . . 638
  Simd256_impl< true, true, true, 8 > . . . 693
    Simd256_impl< true, true, false, 8 > . . . 657
Simd512_impl< ArithType, Int, Signed, Size > . . . 702
Simd512_impl< true, false, true, 4 > . . . 702
Simd512_impl< true, false, true, 8 > . . . 703
Simd512i_base . . . 730
  Simd256_impl< true, true, true, 4 > . . . 675
  Simd512_impl< true, true, true, 8 > . . . 720
    Simd512_impl< true, true, false, 8 > . . . 709
SimdChooser< T, bool, bool > . . . 731
SimdChooser< T, false, b > . . . 731
SimdChooser< T, true, false > . . . 731

SimdChooser< T, true, true > . . . 732
simdToType< T > . . . 732
Single . . . 732
Sparse< Field, SparseMatrix_t, IdxT, PtrT > . . . 732
Sparse< _Field, SparseMatrix_t::COO > . . . 732
Sparse< _Field, SparseMatrix_t::COO_ZO > . . . 734
Sparse< _Field, SparseMatrix_t::CSR > . . . 736
Sparse< _Field, SparseMatrix_t::CSR_ZO > . . . 739
Sparse< _Field, SparseMatrix_t::CSR_HYB > . . . 737
Sparse< _Field, SparseMatrix_t::ELL > . . . 741
Sparse< _Field, SparseMatrix_t::ELL_ZO > . . . 746
Sparse< _Field, SparseMatrix_t::ELL_simd > . . . 742
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > . . . 744
Sparse< _Field, SparseMatrix_t::SELL > . . . 747
Sparse< _Field, SparseMatrix_t::SELL_ZO > . . . 749
SpMat< Field, flag > . . . 751
StatsMatrix . . . 752
string . . . 755
Test< Elt > . . . 757
TestOneMethod< Simd > . . . 760
tfn_minus . . . 763
tfn_minus_eq . . . 763
tfn_mul . . . 763
tfn_mul_eq . . . 764
tfn_plus . . . 764
tfn_plus_eq . . . 765
Threads . . . 765
ThreeD . . . 765
ThreeDAdaptive . . . 765
ThreeDInPlace . . . 765
TRSMHelper< RecIterTrait, ParSeqTrait > . . . 765
true_type
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > . . . 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 487
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 490
support_fast_mod< double > . . . 755
support_fast_mod< float > . . . 756
support_fast_mod< int64_t > . . . 756
TwoD . . . 767
TwoDAdaptive . . . 767
UnparametricTag . . . 767
vector< T > . . . 767
width< T > . . . 768

width< double > 768
width< float > 769
Winograd 769
WinogradPar 769

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

AlgoChooser< ModeT, ParSeq > . . . 397
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > . . . 397
ALL< v > . . . 397
ALL< false, v... > . . . 398
ALL< true, v... > . . . 398
ALL<> . . . 398
ArbitraryPrecIntTag
    Arbitrary precision integers: GMP . . . 398
AreEqual< X, Y > . . . 399
AreEqual< X, X > . . . 399
Argument . . . 399
array< T >
    STL class . . . 400
associatedDelayedField< Field > . . . 400
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > . . . 401
associatedDelayedField< const Givaro::Modular< T, X > > . . . 401
associatedDelayedField< const Givaro::ModularBalanced< T > > . . . 402
associatedDelayedField< const Givaro::ZRing< T > > . . . 403
Auto . . . 403
Bench< Elt > . . . 403
Bini . . . 406
Block . . . 406
BlockTransposeSIMD< Field, Simd, > . . . 406
callLUdivine_small< Element > . . . 408
callLUdivine_small< double > . . . 409
callLUdivine_small< float > . . . 410
CharpolyFailed . . . 410
Checker_Empty< Field > . . . 410
CheckerImplem_charpoly< Field, Polynomial > . . . 411
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > . . . 412
CheckerImplem_Det< Field > . . . 413
CheckerImplem_fgemm< Field > . . . 414
CheckerImplem_ftrsm< Field > . . . 416
CheckerImplem_invert< Field > . . . 417
CheckerImplem_PLUQ< Field > . . . 417

Classic . . . 418
Column . . . 419
CompactElement< Element > . . . 419
CompactElement< double > . . . 419
CompactElement< float > . . . 419
CompactElement< int16_t > . . . 420
CompactElement< int32_t > . . . 420
CompactElement< int64_t > . . . 420
compatible_data_type< Field > . . . 421
compatible_data_type< Givaro::ZRing< double > > . . . 421
compatible_data_type< Givaro::ZRing< float > > . . . 421
Compose< H1, H2 > . . . 422
array< T >::const_iterator
STL iterator class . . . 423
string::const_iterator
STL iterator class . . . 423
vector< T >::const_iterator
STL iterator class . . . 423
array< T >::const_reverse_iterator
STL iterator class . . . 423
string::const_reverse_iterator
STL iterator class . . . 423
vector< T >::const_reverse_iterator
STL iterator class . . . 424
Simd128_impl< true, true, false, 2 >::Converter . . . 424
Simd128_impl< true, true, false, 4 >::Converter . . . 424
Simd128_impl< true, true, false, 8 >::Converter . . . 424
Simd128_impl< true, true, true, 2 >::Converter . . . 425
Simd128_impl< true, true, true, 4 >::Converter . . . 425
Simd128_impl< true, true, true, 8 >::Converter . . . 425
Simd256_impl< true, false, true, 8 >::Converter . . . 426
Simd256_impl< true, true, false, 2 >::Converter . . . 426
Simd256_impl< true, true, false, 4 >::Converter . . . 426
Simd256_impl< true, true, false, 8 >::Converter . . . 427
Simd256_impl< true, true, true, 2 >::Converter . . . 427
Simd256_impl< true, true, true, 4 >::Converter . . . 427
Simd256_impl< true, true, true, 8 >::Converter . . . 428
Simd512_impl< true, true, false, 8 >::Converter . . . 428
Simd512_impl< true, true, true, 8 >::Converter . . . 428
ConvertTo< T >
Force conversion to appropriate element type of ElementCategory T . . . 429
Coo< ValT, IdxT > . . . 429
Coo< Field > . . . 431
Coo< ValT, IdxT > . . . 432
CooMat< Field > . . . 434
count_nonconst_lvalue_reference< T > . . . 435
count_nonconst_lvalue_reference< const T &, O... > . . . 435
count_nonconst_lvalue_reference< T &, O... > . . . 435
count_nonconst_lvalue_reference< T, O... > . . . 435
count_nonconst_lvalue_reference<> . . . 436
CsrMat< Field > . . . 436
DefaultBoundedTag
Use standard field operations, but keeps track of bounds on input and output . . . 437
DefaultTag
No specific mode of action: use standard field operations . . . 437
DelayedTag
Performs field operations with delayed mod reductions. Ensures result is reduced . . . 437
DivideAndConquer . . . 437

ElementTraits< Element >
ElementTraits . . . 437
ElementTraits< double > . . . 438
ElementTraits< FFPACK::rns_double_elt > . . . 438
ElementTraits< float > . . . 438
ElementTraits< Givaro::Integer > . . . 439
ElementTraits< int16_t > . . . 439
ElementTraits< int32_t > . . . 440
ElementTraits< int64_t > . . . 440
ElementTraits< int8_t > . . . 440
ElementTraits< RecInt::rint< K > > . . . 441
ElementTraits< RecInt::rmint< K, MG > > . . . 441
ElementTraits< RecInt::ruint< K > > . . . 441
ElementTraits< uint16_t > . . . 442
ElementTraits< uint32_t > . . . 442
ElementTraits< uint64_t > . . . 443
ElementTraits< uint8_t > . . . 443
EllMat< Field > . . . 443
Failure
A precondtion failed . . . 444
FailureCharpolyCheck . . . 446
FailureDetCheck . . . 446
FailureFgemmCheck . . . 446
FailureInvertCheck . . . 446
FailurePLUQCheck . . . 446
FailureTrsmCheck . . . 446
FieldSimd< _Field > . . . 447
FieldTraits< Field >
FieldTrait . . . 453
FieldTraits< FFPACK::RNSInteger< T > > . . . 453
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . 454
FieldTraits< Givaro::Modular< Element > > . . . 455
FieldTraits< Givaro::ModularBalanced< Element > > . . . 455
FieldTraits< Givaro::ZRing< double > > . . . 456
FieldTraits< Givaro::ZRing< float > > . . . 457
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . 457
FieldTraits< Givaro::ZRing< int16_t > > . . . 458
FieldTraits< Givaro::ZRing< int32_t > > . . . 459
FieldTraits< Givaro::ZRing< int64_t > > . . . 459
FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > . . . 460
FieldTraits< Givaro::ZRing< uint16_t > > . . . 460
FieldTraits< Givaro::ZRing< uint32_t > > . . . 461
FieldTraits< Givaro::ZRing< uint64_t > > . . . 462
Fixed . . . 462
FixedPrecIntTag
Fixed precision integers above machine precision: Givaro::recInt . . . 462
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistributio
463
ForStrategy1D< blocksize_t, Cut, Param > . . . 463
ForStrategy2D< blocksize_t, Cut, Param > . . . 466
ftrmmLeftLowerNoTransNonUnit< Element > . . . 470
ftrmmLeftLowerNoTransUnit< Element > . . . 470
ftrmmLeftLowerTransNonUnit< Element > . . . 470
ftrmmLeftLowerTransUnit< Element > . . . 470
ftrmmLeftUpperNoTransNonUnit< Element > . . . 470
ftrmmLeftUpperNoTransUnit< Element > . . . 470
ftrmmLeftUpperTransNonUnit< Element > . . . 470
ftrmmLeftUpperTransUnit< Element > . . . 471

ftrmmRightLowerNoTransNonUnit< Element > . . . 471
ftrmmRightLowerNoTransUnit< Element > . . . 471
ftrmmRightLowerTransNonUnit< Element > . . . 471
ftrmmRightLowerTransUnit< Element > . . . 471
ftrmmRightUpperNoTransNonUnit< Element > . . . 471
ftrmmRightUpperNoTransUnit< Element > . . . 471
ftrmmRightUpperTransNonUnit< Element > . . . 471
ftrmmRightUpperTransUnit< Element > . . . 472
ftrsmLeftLowerNoTransNonUnit< Element > . . . 472
ftrsmLeftLowerNoTransUnit< Element > . . . 472
ftrsmLeftLowerTransNonUnit< Element > . . . 472
ftrsmLeftLowerTransUnit< Element > . . . 472
ftrsmLeftUpperNoTransNonUnit< Element >
Computes the maximal size for delaying the modular reduction in a triangular system resolution . . . 472
ftrsmLeftUpperNoTransUnit< Element > . . . 473
ftrsmLeftUpperTransNonUnit< Element > . . . 473
ftrsmLeftUpperTransUnit< Element > . . . 473
ftrsmRightLowerNoTransNonUnit< Element > . . . 473
ftrsmRightLowerNoTransUnit< Element > . . . 473
ftrsmRightLowerTransNonUnit< Element > . . . 473
ftrsmRightLowerTransUnit< Element > . . . 473
ftrsmRightUpperNoTransNonUnit< Element > . . . 473
ftrsmRightUpperNoTransUnit< Element > . . . 474
ftrsmRightUpperTransNonUnit< Element > . . . 474
ftrsmRightUpperTransUnit< Element > . . . 474
GenericTag
Default is generic . . . 474
GenericTag
Generic ring . . . 474
Grain . . . 474
has_minus_eq_impl< C > . . . 474
has_minus_impl< C > . . . 475
has_mul_eq_impl< C > . . . 475
has_mul_impl< C > . . . 475
has_operation< T > . . . 476
has_plus_eq_impl< C > . . . 476
has_plus_impl< C > . . . 476
HelperFlag . . . 477
HelperMod< Field, ElementTraits > . . . 477
HelperMod< Field, ElementCategories::MachineIntTag > . . . 478
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > . . . 479
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > . . . 479
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > . . . 480
Hybrid . . . 481
Info . . . 481
Info . . . 482
is_all_same< Args > . . . 483
is_all_same< T, Args... > . . . 483
is_all_same<> . . . 483
is_simd< T > . . . 484
isSparseMatrix< Field, M > . . . 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > . . . 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > . . . 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > . . . 486

isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > . . . 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 487
isSparseMatrixMKLFormat< F, M > . . . 488
isSparseMatrixSimdFormat< F, M > . . . 488
isZOSparseMatrix< F, M > . . . 488
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . 490
Iterative . . . 490
array< T >::iterator
STL iterator class . . . 490
string::iterator
STL iterator class . . . 490
vector< T >::iterator
STL iterator class . . . 490
LazyTag
Performs field operations with delayed mod only when necessary. Result may not be reduced . . . 490
limits< T > . . . 491
limits< char > . . . 491
limits< double > . . . 491
limits< float > . . . 492
limits< Givaro::Integer > . . . 493
limits< int > . . . 493
limits< long > . . . 494
limits< long long > . . . 494
limits< RecInt::rint< K > > . . . 495
limits< RecInt::ruint< K > > . . . 496
limits< short int > . . . 496
limits< signed char > . . . 497
limits< unsigned char > . . . 498
limits< unsigned int > . . . 498
limits< unsigned long > . . . 499
limits< unsigned long long > . . . 499
limits< unsigned short int > . . . 500
MachineFloatTag
Float or double . . . 501
MachineIntTag
Short, int, long, long long, and unsigned variants . . . 501
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > . . . 501
ModeTraits< Field >
ModeTraits . . . 506
ModeTraits< Givaro::Modular< Element, Compute > > . . . 507
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > . . . 507
ModeTraits< Givaro::Modular< int16_t, Compute > > . . . 507
ModeTraits< Givaro::Modular< int32_t, Compute > > . . . 508
ModeTraits< Givaro::Modular< int64_t, uint64_t > > . . . 508
ModeTraits< Givaro::Modular< int8_t, Compute > > . . . 509
ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > . . . 509
ModeTraits< Givaro::Modular< uint16_t, Compute > > . . . 509
ModeTraits< Givaro::Modular< uint32_t, Compute > > . . . 510
ModeTraits< Givaro::Modular< uint8_t, Compute > > . . . 510
ModeTraits< Givaro::ModularBalanced< Element > > . . . 511
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . 511

ModeTraits< Givaro::ModularBalanced< int16_t > > . . . 511
ModeTraits< Givaro::ModularBalanced< int32_t > > . . . 512
ModeTraits< Givaro::ModularBalanced< int8_t > > . . . 512
ModeTraits< Givaro::Montgomery< T > > . . . 513
ModeTraits< Givaro::ZRing< double > > . . . 513
ModeTraits< Givaro::ZRing< float > > . . . 513
ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . 514
ModularBalanced< T > . . . 514
ModularBalanced< T > . . . 514
ModularTag
This is a modular field like e.g. Modular<T> or ModularBalanced<T> . . . 514
Montgomery< T > . . . 514
need_field_characteristic< Field > . . . 515
need_field_characteristic< Givaro::Modular< Field > > . . . 515
need_field_characteristic< Givaro::ModularBalanced< Field > > . . . 515
NoSimd< T > . . . 516
Parallel< C, P > . . . 517
RNSInteger< RNS >::RandIter . . . 518
RNSIntegerMod< RNS >::RandIter . . . 519
readMyMachineType< Field, T > . . . 520
readMyMachineType< Field, mpz_t > . . . 521
Recursive . . . 522
Recursive . . . 522
array< T >::reverse_iterator
STL iterator class . . . 522
string::reverse_iterator
STL iterator class . . . 523
vector< T >::reverse_iterator
STL iterator class . . . 523
rint< K > . . . 523
rns_double . . . 523
rns_double_elt . . . 528
rns_double_elt_cstptr . . . 529
rns_double_elt_ptr . . . 532
rns_double_extended . . . 535
RNSElementTag
Representation in a Residue Number System . . . 538
RNSInteger< RNS > . . . 538
RNSIntegerMod< RNS > . . . 542
rnsRandIter< RNS > . . . 549
Row . . . 550
ruint< K > . . . 550
ScalFunctions< Element > . . . 550
ScalFunctionsBase< Element, Enable > . . . 555
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > . . . 556
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > . . . 557
Sequential . . . 560
Simd128_impl< ArithType, Int, Signed, Size > . . . 561
Simd128_impl< true, false, true, 4 > . . . 561
Simd128_impl< true, false, true, 8 > . . . 561
Simd128_impl< true, true, false, 2 > . . . 561
Simd128_impl< true, true, false, 4 > . . . 571
Simd128_impl< true, true, false, 8 > . . . 581
Simd128_impl< true, true, true, 2 > . . . 591
Simd128_impl< true, true, true, 4 > . . . 601
Simd128_impl< true, true, true, 8 > . . . 609
Simd128i_base . . . 619
Simd256_impl< ArithType, Int, Signed, Size > . . . 620

Simd256_impl< true, false, true, 4 > . . . 620
Simd256_impl< true, false, true, 8 > . . . 621
Simd256_impl< true, true, false, 2 > . . . 628
Simd256_impl< true, true, false, 4 > . . . 638
Simd256_impl< true, true, false, 8 > . . . 657
Simd256_impl< true, true, true, 2 > . . . 666
Simd256_impl< true, true, true, 4 > . . . 675
Simd256_impl< true, true, true, 8 > . . . 693
Simd256fp_base . . . 702
Simd256i_base . . . 702
Simd512_impl< ArithType, Int, Signed, Size > . . . 702
Simd512_impl< true, false, true, 4 > . . . 702
Simd512_impl< true, false, true, 8 > . . . 703
Simd512_impl< true, true, false, 8 > . . . 709
Simd512_impl< true, true, true, 8 > . . . 720
Simd512i_base . . . 730
SimdChooser< T, bool, bool > . . . 731
SimdChooser< T, false, b > . . . 731
SimdChooser< T, true, false > . . . 731
SimdChooser< T, true, true > . . . 732
simdToType< T > . . . 732
Single . . . 732
Sparse< Field, SparseMatrix_t, IdxT, PtrT > . . . 732
Sparse< _Field, SparseMatrix_t::COO > . . . 732
Sparse< _Field, SparseMatrix_t::COO_ZO > . . . 734
Sparse< _Field, SparseMatrix_t::CSR > . . . 736
Sparse< _Field, SparseMatrix_t::CSR_HYB > . . . 737
Sparse< _Field, SparseMatrix_t::CSR_ZO > . . . 739
Sparse< _Field, SparseMatrix_t::ELL > . . . 741
Sparse< _Field, SparseMatrix_t::ELL_simd > . . . 742
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > . . . 744
Sparse< _Field, SparseMatrix_t::ELL_ZO > . . . 746
Sparse< _Field, SparseMatrix_t::SELL > . . . 747
Sparse< _Field, SparseMatrix_t::SELL_ZO > . . . 749
SpMat< Field, flag > . . . 751
StatsMatrix . . . 752
string
STL class . . . 755
support_fast_mod< T > . . . 755
support_fast_mod< double > . . . 755
support_fast_mod< float > . . . 756
support_fast_mod< int64_t > . . . 756
support_simd< T > . . . 756
support_simd_add< T > . . . 756
support_simd_mod< T > . . . 757
Test< Elt > . . . 757
TestOneMethod< Simd > . . . 760
tfn_minus . . . 763
tfn_minus_eq . . . 763
tfn_mul . . . 763
tfn_mul_eq . . . 764
tfn_plus . . . 764
tfn_plus_eq . . . 765
Threads . . . 765
ThreeD . . . 765
ThreeDAdaptive . . . 765
ThreeDInPlace . . . 765

TRSMHelper< RecIterTrait, ParSeqTrait >
TRSM Helper . . . 765
TwoD . . . 767
TwoDAdaptive . . . 767
UnparametricTag
If the field uses a representation with infix operators . . . 767
vector< T >
STL class . . . 767
width< T > . . . 768
width< double > . . . 768
width< float > . . . 769
Winograd . . . 769
WinogradPar . . . 769

Chapter 9

File Index

9.1 File List

Here is a list of all files with brief descriptions:

arithprog.C . . . 771
autotune/charpoly.C . . . 771
examples/charpoly.C . . . 772
fsyrk.C . . . 773
fsytrf.C . . . 773
ftrtri.C . . . 774
autotune/pluq.C . . . 775
examples/pluq.C . . . 776
winograd.C . . . 776
benchmark-charpoly-mp.C . . . 777
benchmark-charpoly.C . . . 778
benchmark-checkers.C . . . 778
benchmark-dgemm.C . . . 779
benchmark-dgetrf.C . . . 780
benchmark-dgetri.C . . . 781
benchmark-dsytrf.C . . . 781
benchmark-dtrsm.C . . . 782
benchmark-dtrtri.C . . . 782
benchmark-fadd-lvl2.C . . . 783
benchmark-fdot.C . . . 783
benchmark-fgemm-mp.C . . . 784
benchmark-fgemm-rns.C . . . 785
benchmark-fgemm.C . . . 787
benchmark-fgemv-mp.C . . . 787
benchmark-fgemv.C . . . 789
benchmark-fgesv.C . . . 792
benchmark-fsyr2k.C . . . 792
benchmark-fsyrk.C . . . 793
benchmark-fsytrf.C . . . 793
benchmark-ftrsm-mp.C . . . 794
benchmark-ftrsm.C . . . 795
benchmark-ftrsv.C . . . 795
benchmark-ftrtri.C . . . 796
benchmark-inverse.C . . . 796
benchmark-lqup-mp.C . . . 797

benchmark-lqup.C . . . 798
benchmark-pluq.C . . . 798
benchmark-quasisep.C . . . 799
benchmark-storage-transpose.C . . . 800
benchmark-wino.C . . . 801
mainpage.doxy . . . 802
det.C . . . 802
matmul.C . . . 802
rank.C . . . 803
solve.C . . . 803
checker_charpoly.inl . . . 803
checker_det.inl . . . 804
checker_empty.h . . . 804
checker_fgemm.inl . . . 804
checker_ftrsm.inl . . . 805
checker_invert.inl . . . 805
checker_pluq.inl . . . 805
checkers.doxy . . . 806
checkers_fflas.h . . . 806
checkers_fflas.inl . . . 806
checkers_ffpack.h . . . 807
checkers_ffpack.inl . . . 807
config-blas.h . . . 808
config.h . . . 815
fflas-ffpack/config.h . . . 818
fflas-ffpack-config.h
Defaults for optimised values . . . 821
fflas-ffpack-default-thresholds.h . . . 822
fflas-ffpack-thresholds.h . . . 823
fflas-ffpack.doxy . . . 823
fflas-ffpack.h
Includes FFLAS and FFPACK . . . 823
fflas.doxy . . . 823
fflas.h
Finite **F**ield **L**inear **A**lgebra **S**ubroutines . . . 823
fflas_bounds.inl . . . 825
fflas_enum.h . . . 825
fflas_fadd.h . . . 826
fflas_fadd.inl . . . 827
fflas_fassign.h . . . 828
fflas_fassign.inl . . . 829
fflas_faxpy.inl . . . 829
fflas_fdot.inl . . . 830
fflas_fgemm.inl . . . 831
fgemm_classical.inl . . . 833
fgemm_classical_mp.inl
Matrix multiplication with multiprecision input (either over Z or over Z/pZ) . . . 833
fgemm_winograd.inl . . . 835
matmul.doxy . . . 837
schedule_bini.inl
Bini implementation . . . 837
schedule_winograd.inl . . . 837
schedule_winograd_acc.inl . . . 838
schedule_winograd_acc_ip.inl . . . 839
schedule_winograd_ip.inl . . . 839
fflas_fgemv.inl . . . 840
fflas_fgemv_mp.inl . . . 841
fflas_fger.inl . . . 842

fflas_fger_mp.inl . . . 843
fflas_freduce.h . . . 844
fflas_freduce.inl . . . 845
fflas_freduce_mp.inl . . . 847
fflas_freivalds.inl . . . 847
fflas_fscal.h . . . 847
fflas_fscal.inl . . . 848
fflas_fscal_mp.inl . . . 849
fflas_fsyr2k.inl . . . 850
fflas_fsyrk.inl . . . 850
fflas_fsyrk_strassen.inl . . . 852
fflas_ftrmm.inl . . . 853
fflas_ftrsm.inl . . . 854
fflas_ftrsm_mp.inl
Triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ) . . . 854
fflas_ftrsv.inl . . . 855
fflas_helpers.inl . . . 855
igemm.doxy . . . 857
igemm.h . . . 857
igemm.inl . . . 857
igemm_kernels.h . . . 858
igemm_kernels.inl . . . 858
igemm_tools.h . . . 859
igemm_tools.inl . . . 859
fflas_level1.inl . . . 860
fflas_level2.inl . . . 862
fflas_level3.inl . . . 865
fflas_pfgemm.inl . . . 867
fflas_pftrsm.inl . . . 868
fflas_simd.h . . . 869
simd.doxy . . . 871
simd128.inl . . . 871
simd128_double.inl . . . 871
simd128_float.inl . . . 872
simd128_int16.inl . . . 872
simd128_int32.inl . . . 872
simd128_int64.inl . . . 873
simd256.inl . . . 873
simd256_double.inl . . . 874
simd256_float.inl . . . 874
simd256_int16.inl . . . 875
simd256_int32.inl . . . 875
simd256_int64.inl . . . 876
simd512.inl . . . 876
simd512_double.inl . . . 877
simd512_float.inl . . . 877
simd512_int32.inl . . . 878
simd512_int64.inl . . . 878
simd_modular.inl . . . 879
fflas_sparse.h . . . 879
fflas_sparse.inl . . . 883
coo.h . . . 885
coo_spmm.inl . . . 886
coo_spmv.inl . . . 887
coo_utils.inl . . . 887
csr.h . . . 888
csr_pspmm.inl . . . 888

csr_pspmv.inl . . . 889
csr_spmm.inl . . . 890
csr_spmv.inl . . . 891
csr_utils.inl . . . 892
csr_hyb.h . . . 893
csr_hyb_pspmm.inl . . . 893
csr_hyb_pspmv.inl . . . 894
csr_hyb_spmm.inl . . . 894
csr_hyb_spmv.inl . . . 895
csr_hyb_utils.inl . . . 895
ell.h . . . 896
ell_pspmm.inl . . . 896
ell_pspmv.inl . . . 897
ell_spmm.inl . . . 898
ell_spmv.inl . . . 899
ell_utils.inl . . . 899
ell_simd.h . . . 900
ell_simd_pspmv.inl . . . 901
ell_simd_spmv.inl . . . 901
ell_simd_utils.inl . . . 902
hyb_zo.h . . . 903
hyb_zo_pspmm.inl . . . 903
hyb_zo_pspmv.inl . . . 903
hyb_zo_spmm.inl . . . 904
hyb_zo_spmv.inl . . . 904
hyb_zo_utils.inl . . . 905
read_sparse.h . . . 905
sell.h . . . 906
sell_pspmv.inl . . . 907
sell_spmv.inl . . . 907
sell_utils.inl . . . 908
sparse_matrix_traits.h . . . 909
utils.h . . . 910
fflas_transpose.h
Transpose the storage of the matrix (switch between row and col major mode) . . . 911
ffpack.doxy . . . 912
ffpack.h
Set of elimination based routines for dense linear algebra . . . 912
ffpack.inl . . . 921
ffpack_bruhatgen.inl . . . 922
ffpack_charpoly.inl . . . 924
ffpack_charpoly_danilevski.inl . . . 924
ffpack_charpoly_kgfast.inl . . . 925
ffpack_charpoly_kgfastgeneralized.inl . . . 925
ffpack_charpoly_kglu.inl . . . 926
ffpack_charpoly_mp.inl . . . 926
ffpack_det_mp.inl . . . 927
ffpack_echelonforms.inl . . . 928
ffpack_fgesv.inl . . . 929
ffpack_fgetrs.inl . . . 929
ffpack_frobenius.inl . . . 930
ffpack_fsytrf.inl . . . 931
ffpack_ftrssyr2k.inl . . . 932
ffpack_ftrstr.inl . . . 932
ffpack_ftrtr.inl . . . 933
ffpack_invert.inl . . . 933
ffpack_krylovelim.inl . . . 934
ffpack_ludivine.inl . . . 934

ffpack_ludivine_mp.inl . . . 935
ffpack_minpoly.inl . . . 936
ffpack_permutation.inl . . . 936
ffpack_pluq.inl . . . 939
ffpack_pluq_mp.inl . . . 939
ffpack_ppluq.inl . . . 940
ffpack_rankprofiles.inl . . . 941
field-traits.h
Field Traits . . . 942
field.doxy . . . 944
rns-double-elt.h
Rns elt structure with double support . . . 944
rns-double-recint.inl . . . 945
rns-double.h
Rns structure with double support . . . 945
rns-double.inl . . . 946
rns-integer-mod.h
Representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision) . . . 946
rns-integer.h
Representation of $\mathbb{Z}$ using RNS representation (note: fixed precision) . . . 947
rns.h . . . 948
rns.inl . . . 948
interfaces.doxy . . . 948
fflas_c.h . . . 948
fflas_L1_inst.C . . . 960
fflas_L1_inst.h . . . 961
fflas_L1_inst_implem.inl . . . 962
fflas_L2_inst.C . . . 963
fflas_L2_inst.h . . . 964
fflas_L2_inst_implem.inl . . . 965
fflas_L3_inst.C . . . 967
fflas_L3_inst.h . . . 968
fflas_L3_inst_implem.inl . . . 969
fflas_lvl1.C
C functions calls for level 1 FFLAS in fflas-c.h . . . 970
fflas_lvl2.C
C functions calls for level 2 FFLAS in fflas-c.h . . . 974
fflas_lvl3.C
C functions calls for level 3 FFLAS in fflas-c.h . . . 980
fflas_sparse.C
C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h . . . 981
ffpack.C
C functions calls for FFPACK in ffpack-c.h . . . 982
ffpack_c.h . . . 1002
ffpack_inst.C . . . 1022
ffpack_inst.h . . . 1023
ffpack_inst_implem.inl . . . 1024
blockcuts.inl . . . 1027
fflas_plevel1.h . . . 1029
kaapi_routines.inl . . . 1029
parallel.h . . . 1029
pfgemm_variants.inl . . . 1036
pfgemv.inl . . . 1037
align-allocator.h . . . 1037
args-parser.h . . . 1038
bit_manipulation.h . . . 1039
cast.h . . . 1040

debug.h
Various utilities for debugging . . . 1040
fflas_intrinsic.h . . . 1041
fflas_io.h . . . 1041
fflas_memory.h . . . 1042
fflas_randommatrix.h . . . 1043
flimits.h . . . 1045
Matio.h . . . 1046
test-utils.h . . . 1047
timer.h . . . 1047
cblas.C . . . 1048
clapack.C . . . 1048
cuda.C . . . 1049
fblas.C . . . 1049
lapack.C . . . 1050
regression-check.C . . . 1050
test-charpoly-check.C . . . 1051
test-charpoly.C . . . 1052
test-compressQ.C . . . 1053
test-det-check.C . . . 1054
test-det.C . . . 1054
test-echelon.C . . . 1055
test-fadd.C . . . 1057
test-fdot.C . . . 1059
test-fgemm-check.C . . . 1060
test-fgemm.C . . . 1061
test-fgemv.C . . . 1063
test-fger.C . . . 1065
test-fgesv.C . . . 1067
test-finit.C . . . 1068
test-fscal.C . . . 1069
test-fsyr2k.C . . . 1071
test-fsyrk.C . . . 1072
test-fsytrf.C . . . 1074
test-ftrmm.C . . . 1075
test-ftrmv.C . . . 1077
test-ftrsm-check.C . . . 1078
test-ftrsm.C . . . 1078
test-ftrssyr2k.C . . . 1080
test-ftrstr.C . . . 1081
test-ftrsv.C . . . 1082
test-ftrtri.C . . . 1083
test-interfaces-c.c . . . 1084
test-invert-check.C . . . 1084
test-io.C . . . 1085
test-lu.C . . . 1086
test-maxdelayeddim.C . . . 1090
test-minpoly.C . . . 1090
test-multifile1.C . . . 1091
test-multifile2.C . . . 1091
test-nullspace.C . . . 1092
test-permutations.C . . . 1093
test-pluq-check.C . . . 1094
test-quasisep.C . . . 1095
test-rankprofiles.C . . . 1096
test-rpm.C . . . 1097
test-simd.C . . . 1098
test-solve.C . . . 1101

test-storage-transpose.C . . . 1102
101-fgemm.C . . . 1102
2x2-fgemm.C . . . 1103
2x2-ftrsv.C . . . 1103
2x2-pluq.C . . . 1104
fflas-101_1.C . . . 1104
fflas-101_3.C . . . 1104
fflas_101.C . . . 1105
fflas_101_lvl1.C . . . 1105
ffpack-fgesv.C . . . 1106
ffpack-solve.C . . . 1106

Chapter 10

Topic Documentation

10.1 CHECKER

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

10.2 FFLAS-FFPACK

the FFLAS FFPACK library

Topics

- FFLAS
  The C-style wrapper of BLAS for finite field linear algebra.
- Interfaces
  Intefaces for FFLAS-FFPACK.

10.2.1 Detailed Description

the FFLAS FFPACK library

C++ header library for fast exact dense linear algebra

See also

FFLAS

FFPACK

10.2.2 FFLAS

The C-style wrapper of BLAS for finite field linear algebra.

The C-style wrapper of BLAS for finite field linear algebra.

FFLAS, Finite Field Linear Algebra Subroutines, provide basic linear algebra subroutines based on the BLAS interface. Therefore, the specifications are in C style; only the field given as a template parameter requires `C++`.

As much as possible, these routines use `ATLAS/BLAS` computations and achieve therefore high efficiency.

10.2.3 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

libs

10.3 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

Files

- file schedule_bini.inl
  Bini implementation.

10.3.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

Todo biblio

10.4 SIMD wrapper

wraps SIMD functions Supporst SSE4.1, AVX, AVX2.

wraps SIMD functions Supporst SSE4.1, AVX, AVX2.

Todo biblio

10.5 FFPACK

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

10.5.1 Detailed Description

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

10.6 FFLAS-FFPACK fields

fields in the FFLAS-FFPACK library

Files

- file rns-double-elt.h

  rns elt structure with double support
- file rns-double.h

  rns structure with double support
- file rns-integer-mod.h

  representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)
- file rns-integer.h

  representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)
- file rns.h

10.6.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

Todo biblio

10.7 RNS

just include them all

just include them all

Chapter 11

Namespace Documentation

11.1 FFLAS Namespace Reference

Namespaces

- namespace _ftranspose_impl
- namespace BLAS3
- namespace csr_hyb_details
- namespace CuttingStrategy
- namespace details
- namespace details_spmv
- namespace ElementCategories
- namespace FieldCategories

  Traits and categories will need to be placed in a proper file later.
- namespace MMHelperAlgo
- namespace ModeCategories

  Specifies the mode of action for an algorithm w.r.t.
- namespace ParSeqHelper

  ParSeqHelper for both fgemm and ftrsm.
- namespace Protected
- namespace sell_details
- namespace sparse_details
- namespace sparse_details_impl
- namespace StrategyParameter
- namespace StructureHelper

  StructureHelper for ftrsm.
- namespace vectorised

Data Structures

- struct AlgoChooser
- struct AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >
- struct associatedDelayedField
- struct associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >
- struct associatedDelayedField< const Givaro::Modular< T, X > >
- struct associatedDelayedField< const Givaro::ModularBalanced< T > >
- struct associatedDelayedField< const Givaro::ZRing< T > >

- struct BlockTransposeSIMD
- struct Checker_Empty
- class CheckerImplem_fgemm
- class CheckerImplem_ftrsm
- struct CooMat
- struct CsrMat
- struct ElementTraits

  ElementTraits.
- struct ElementTraits< double >
- struct ElementTraits< FFPACK::rns_double_elt >
- struct ElementTraits< float >
- struct ElementTraits< Givaro::Integer >
- struct ElementTraits< int16_t >
- struct ElementTraits< int32_t >
- struct ElementTraits< int64_t >
- struct ElementTraits< int8_t >
- struct ElementTraits< RecInt::rint< K > >
- struct ElementTraits< RecInt::rmint< K, MG > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< uint16_t >
- struct ElementTraits< uint32_t >
- struct ElementTraits< uint64_t >
- struct ElementTraits< uint8_t >
- struct EllMat
- struct FieldTraits

  FieldTrait.
- struct FieldTraits< FFPACK::RNSInteger< T > >
- struct FieldTraits< FFPACK::RNSIntegerMod< T > >
- struct FieldTraits< Givaro::Modular< Element > >
- struct FieldTraits< Givaro::ModularBalanced< Element > >
- struct FieldTraits< Givaro::ZRing< double > >
- struct FieldTraits< Givaro::ZRing< float > >
- struct FieldTraits< Givaro::ZRing< Givaro::Integer > >
- struct FieldTraits< Givaro::ZRing< int16_t > >
- struct FieldTraits< Givaro::ZRing< int32_t > >
- struct FieldTraits< Givaro::ZRing< int64_t > >
- struct FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >
- struct FieldTraits< Givaro::ZRing< uint16_t > >
- struct FieldTraits< Givaro::ZRing< uint32_t > >
- struct FieldTraits< Givaro::ZRing< uint64_t > >
- struct ForStrategy1D
- struct ForStrategy2D
- struct has_minus_eq_impl
- struct has_minus_impl
- struct has_mul_eq_impl
- struct has_mul_impl
- struct has_operation
- struct has_plus_eq_impl
- struct has_plus_impl
- struct HelperFlag
- struct isSparseMatrix
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >

- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >
- struct isSparseMatrixMKLFormat
- struct isSparseMatrixSimdFormat
- struct isZOSparseMatrix
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >
- struct MMHelper
- struct ModeTraits

  ModeTraits.
- struct ModeTraits< Givaro::Modular< Element, Compute > >
- struct ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >
- struct ModeTraits< Givaro::Modular< int16_t, Compute > >
- struct ModeTraits< Givaro::Modular< int32_t, Compute > >
- struct ModeTraits< Givaro::Modular< int64_t, uint64_t > >
- struct ModeTraits< Givaro::Modular< int8_t, Compute > >
- struct ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >
- struct ModeTraits< Givaro::Modular< uint16_t, Compute > >
- struct ModeTraits< Givaro::Modular< uint32_t, Compute > >
- struct ModeTraits< Givaro::Modular< uint8_t, Compute > >
- struct ModeTraits< Givaro::ModularBalanced< Element > >
- struct ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >
- struct ModeTraits< Givaro::ModularBalanced< int16_t > >
- struct ModeTraits< Givaro::ModularBalanced< int32_t > >
- struct ModeTraits< Givaro::ModularBalanced< int8_t > >
- struct ModeTraits< Givaro::Montgomery< T > >
- struct ModeTraits< Givaro::ZRing< double > >
- struct ModeTraits< Givaro::ZRing< float > >
- struct ModeTraits< Givaro::ZRing< Givaro::Integer > >
- struct readMyMachineType
- struct readMyMachineType< Field, mpz_t >
- struct Sparse
- struct Sparse< _Field, SparseMatrix_t::COO >
- struct Sparse< _Field, SparseMatrix_t::COO_ZO >
- struct Sparse< _Field, SparseMatrix_t::CSR >
- struct Sparse< _Field, SparseMatrix_t::CSR_HYB >
- struct Sparse< _Field, SparseMatrix_t::CSR_ZO >
- struct Sparse< _Field, SparseMatrix_t::ELL >
- struct Sparse< _Field, SparseMatrix_t::ELL_simd >
- struct Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >
- struct Sparse< _Field, SparseMatrix_t::ELL_ZO >
- struct Sparse< _Field, SparseMatrix_t::SELL >
- struct Sparse< _Field, SparseMatrix_t::SELL_ZO >
- struct SpMat

- struct StatsMatrix
- struct support_fast_mod
- struct support_fast_mod< double >
- struct support_fast_mod< float >
- struct support_fast_mod< int64_t >
- struct support_simd
- struct support_simd_add
- struct support_simd_mod
- struct tfn_minus
- struct tfn_minus_eq
- struct tfn_mul
- struct tfn_mul_eq
- struct tfn_plus
- struct tfn_plus_eq
- struct TRSMHelper

  TRSM Helper.

Typedefs

- template<class Field>
  using Checker_fgemm = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_ftrsm = FFLAS::Checker_Empty<Field>
- template<class Field>
  using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
- template<class Field>
  using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
- using ZOSparseMatrix = std::true_type
- using NotZOSparseMatrix = std::false_type
- using SimdSparseMatrix = std::true_type
- using NoSimdSparseMatrix = std::false_type
- using MKLSparseMatrixFormat = std::true_type
- using NotMKLSparseMatrixFormat = std::false_type
- template<class T>
  using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_plus_impl<T>>↩
  ::type
- template<class T>
  using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_minus_impl<T>>↩
  ::type
- template<class T>
  using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, std::is_copy_↩
  assignable<T>>::type
- template<class T>
  using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_plus_eq_impl<T>>↩
  ::type
- template<class T>
  using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_minus_eq_impl<T>>↩
  ::type
- template<class T>
  using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>>↩
  ::type
- template<class T>
  using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_eq_impl<T>>↩
  ::type
- typedef Givaro::Timer Timer
- typedef Givaro::BaseTimer BaseTimer
- typedef Givaro::UserTimer UserTimer
- typedef Givaro::SysTimer SysTimer

Enumerations

- enum FFLAS_ORDER { FflasRowMajor =101 , FflasColMajor =102 }

  Storage by row or col ?
- enum FFLAS_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }

  Is matrix transposed ?
- enum FFLAS_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasLeftTri = 123 , FflasRightTri = 124 }

  Is triangular matrix's shape upper ?
- enum FFLAS_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }

  Is the triangular matrix implicitly unit diagonal ?
- enum FFLAS_SIDE { FflasLeft = 141 , FflasRight = 142 }

  On what side ?
- enum FFLAS_BASE { FflasDouble = 151 , FflasFloat = 152 , FflasGeneric = 153 }

  FFLAS_BASE determines the type of the element representation for Matrix Mult kernel.
- enum number_kind { zero =0 , one =1 , mone =-1 , other =2 }
- enum class SparseMatrix_t {
  CSR , CSR_ZO , CSC , CSC_ZO ,
  COO , COO_ZO , ELL , ELL_ZO ,
  SELL , SELL_ZO , ELL_simd , ELL_simd_ZO ,
  CSR_HYB , HYB_ZO }
- enum FFLAS_FORMAT {
  FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,
  FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }

Functions

- Givaro::Integer InfNorm (const size_t M, const size_t N, const Givaro::Integer *A, const size_t lda)
- template<class T>
  const T & min3 (const T &m, const T &n, const T &k)
- template<class T>
  const T & max3 (const T &m, const T &n, const T &k)
- template<class T>
  const T & min4 (const T &m, const T &n, const T &k, const T &l)
- template<class T>
  const T & max4 (const T &m, const T &n, const T &k, const T &l)
- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)

- template<class Field>
  void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)
- template<class Field>
  void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition.
- template<class Field>
  void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsub : matrix subtraction.
- template<class Field>
  void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  faddin
- template<class Field>
  void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadding for symmetric matrices
- template<class Field>
  void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsubin C = C - B
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition with scaling.
- template<class Field>
  void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  fassign : $x \leftarrow y$.
- template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<class Field>
  void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  fassign : $A \leftarrow B$.

- template<class Field>
  void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<class Field>
  void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)
- template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<class Field, class T>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fdot: dot product $x^T y$.
- template<class Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template<typename Field, class Cut, class Param>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)

- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  *fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*
- template<typename Field, class ModeT, class ParSeq>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > &H)
- template<class Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field>
  Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsquare: Squares a matrix.
- template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)
- template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)
- template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)
- template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS_TRANSPOSE ta, const size↩
  _t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)
- template<typename RNS, typename ParSeqTrait>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)
- template<typename RNS>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)
- template<typename RNS, typename ParSeqTrait>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS

>, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)

- template<typename RNS, typename Cut, typename Param>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class ParSeq>
  Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<typename RNS, class ModeT>
  RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)
- template<typename RNS>
  RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)
- Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<class ParSeq>
  Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<size_t K1, size_t K2, class ParSeq>
  RecInt::ruint< K1 > * fgemm (const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const RecInt::ruint< K1 > alpha, const RecInt::ruint< K1 > *A, const size_t lda, const RecInt::ruint< K1 > *B, const size_t ldb, RecInt::ruint< K1 > beta, RecInt::ruint< K1 > *C, const size_t ldc, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<class Field, class ModeT>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)
- template<class Field, class ModeT, class Cut, class Param>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,

const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)

- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)

  finite prime Field GEneral Matrix Vector multiplication.
- Givaro::ZRing< int64_t >::Element_ptr fgemv (const Givaro::ZRing< int64_t > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *X, const size_t incX, const int64_t beta, int64_t *Y, const size_t incY, MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::DoubleDomain::Element_ptr fgemv (const Givaro::DoubleDomain &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::ConstElement_ptr X, const size_t incX, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fgemv (const Givaro::FloatDomain &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::ConstElement_ptr X, const size_t incX, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field, class Cut, class Param>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Parallel< Cut, Param > &parH)

- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Sequential &seqH)
- FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::Integer * fgemv (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- Givaro::Integer * fgemv (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<size_t K1, size_t K2, class ParSeq>
  RecInt::ruint< K1 > * fgemv (const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const RecInt::ruint< K1 > alpha, const RecInt::ruint< K1 > *A, const size_t lda, const RecInt::ruint< K1 > *X, const size_t incx, RecInt::ruint< K1 > beta, RecInt::ruint< K1 > *Y, const size_t incy, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)

  fger: rank one update of a general matrix
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)
- template<class Field, class AnyTag>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)
- void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)

- void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- void fger (const Givaro::Modular< Givaro::Integer > &F, const size_t M, const size_t N, const typename Givaro::Integer alpha, typename Givaro::Integer *x, const size_t incx, typename Givaro::Integer *y, const size_t incy, typename Givaro::Integer *A, const size_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename RNS>
  void fger (const FFPACK::RNSInteger< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSInteger< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSInteger< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename RNS>
  void fger (const FFPACK::RNSIntegerMod< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSIntegerMod< RNS >::Element alpha, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > &H)
- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow y mod F$.
- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow x mod F$.
- template<class Field>
  void freduce_constoverride (const Field &F, const size_t m, typename Field::ConstElement_ptr A, const size_t incX)
- template<class Field, class ConstOtherElement_ptr>
  void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)
- template<class Field>
  void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  finit Initializes X *in* $F\$$.
- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow A mod F$.
- template<class Field>
  void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

  freduce for square symmetric matrices
- template<class Field>
  void pfreduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const size_t numths)

- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow BmodF$.
- template<class Field>
  void freduce_constoverride (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  finit $A \leftarrow BmodF$.
- template<class Field>
  void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)
- template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, size_t inc)
- template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, FFPACK::rns_double::Element_ptr A, size_t lda)
- template<class Field>
  bool freivalds (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::ConstElement_ptr C, const size_t ldc)

  *freivalds: **F**reivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*
- template<class Field>
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- template<class Field>
  void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  fscal $y \leftarrow \alpha \cdot x$.
- template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element↩ _ptr y, const size_t incy)
- template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<class Field>
  void fscalin (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)

  fscalin $A \leftarrow a \cdot A$.
- template<class Field>
  void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  fscal $B \leftarrow a \cdot A$.
- template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t inc)
- template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)
- template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)

- template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)
- template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)
- template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)
- template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)
- template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)
- template<class Field>
  Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyr2k: Symmetric Rank 2K update
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyrk: Symmetric Rank K update
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field, typename Mode>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > &H)

- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fsyrk (const Givaro::FloatDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::FloatDomain::Element alpha, Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::DoubleDomain::Element_ptr fsyrk (const Givaro::DoubleDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::DoubleDomain::Element alpha, Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq, const size_t threshold)
- template<class Field, class Cut, class Param>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &twoBlock, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling
- template<class Field, class FieldTrait>
  void computeS1S2 (const Field &F, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element x, const typename Field::Element y, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr S, const size_t lds, typename Field::Element_ptr T, const size_t ldt, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field, class Mode>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Mode > &H)
- template<class Field, class FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO uplo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element

y2, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  *ftrmm: **TR**iangular **M**atrix **M**ultiply.*
- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  *ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ *or* $C \leftarrow \alpha B \mathrm{op}(A) + betaC$.
- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)
- template<class Field, class Cut, class Param>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)
- template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)
- void ftrsm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, Givaro::Integer *B, const size_t ldb)
- void cblas_imptrsm (const enum FFLAS_ORDER Order, const enum FFLAS_SIDE Side, const enum FFLAS_UPLO Uplo, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_DIAG Diag, const int M, const int N, const FFPACK::rns_double_elt alpha, FFPACK::rns_double_elt_cstptr A, const int lda, FFPACK::rns_double_elt_ptr B, const int ldb)
- template<class Field>
  void ftrsv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$
- void igemm_ (const enum FFLAS_ORDER Order, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, const size_t M, const size_t N, const size_t K, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *B, const size_t ldb, const int64_t beta, int64_t *C, const size_t ldc)
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  finit $x \leftarrow ymodF$.
- template<class Field, class OtherElement_ptr>
  void fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fconvert $x \leftarrow ymodF$.

- template<class Field>
  void fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  fnegin $x \leftarrow -x$.
- template<class Field>
  void fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  fneg $x \leftarrow -y$.
- template<class Field>
  void fzero (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  fzero : $A \leftarrow 0$.
- template<class Field, class RandIter>
  void frand (const Field &F, RandIter &G, const size_t n, typename Field::Element_ptr X, const size_t incX)

  frand : $A \leftarrow random$.
- template<class Field>
  bool fiszero (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX)

  fiszero : test $X = 0$.
- template<class Field>
  bool fequal (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fequal : test $X = Y$.
- template<class Field>
  void faxpby (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template<typename Field, class Cut, class Param>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field>
  void fswap (const Field &F, const size_t N, typename Field::Element_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  fswap: $X \leftrightarrow Y$.
- template<class Field>
  void fzero (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fzero : $A \leftarrow 0$.
- template<class Field>
  void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fzero : $A \leftarrow 0$ *for a triangular matrix.*
- template<class Field, class RandIter>
  void frand (const Field &F, RandIter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  frand : $A \leftarrow random$.
- template<class Field>
  bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)

  fequal : test $A = B$.
- template<class Field>
  bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size↩_t lda)

  fiszero : test $A = 0$.
- template<class Field>
  void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const typename Field::Element &d)

creates a diagonal matrix

- template<class Field>
  void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  creates a diagonal matrix
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  finit Initializes A *in* $F\$$.
- template<class Field, class OtherElement_ptr>
  void fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)

  fconvert $A \leftarrow BmodF$.
- template<class Field>
  void fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fnegin $A \leftarrow -A$.
- template<class Field>
  void fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  fneg $A \leftarrow -B$.
- template<class Field>
  void faxpby (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t ldy)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template<class Field>
  void fmove (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  fmove : $A \leftarrow B$ *and* $B \leftarrow 0$.
- template<class Field>
  size_t bitsize (const Field &F, size_t M, size_t N, const typename Field::ConstElement_ptr A, size_t lda)

  bitsize: Computes the largest bitsize of the matrix' coefficients.
- template<> size_t bitsize< Givaro::ZRing< Givaro::Integer > > (const Givaro::ZRing< Givaro::Integer > &F, size_t M, size_t N, const Givaro::Integer *A, size_t lda)
- template<class Field>
  void ftrmv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)

  ftrsm: TRiangular Matrix Vector prodcut Computes $X \leftarrow \mathrm{op}(A)X$
- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  ftrsm: ***TR****iangular* ***S****ystem solve with* ***M****atrix.*
- template<class Field, typename FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &H)
- template<typename Field>
  Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t numthreads=0)

- template<class Field>
  Field::Element * pfgemm_1D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)
- template<class Field>
  Field::Element * pfgemm_2D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)
- template<class Field>
  Field::Element * pfgemm_3D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)
- template<class Field>
  Field::Element_ptr pfgemm_3D_rec2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)
- template<class Field, class ModeTrait, class Strat, class Param>
  std::enable_if< !std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > >::value, typenameField::Element_ptr >::type fgemm (const Field &F, const FFLAS::FFLAS_TRANSPOSE ta, const FFLAS::FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > > &H)
- template<class Field, class Cut, class Param>
  Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class Field, class Cut, class Param>
  Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)
- template<class Field>
  std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)
- template<class IndexT>
  void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT>
  void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT, size_t RECINT_SIZE>
  void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT, size_t RECINT_SIZE>
  void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)
- template<class Field>
  void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<typename _Field>
  std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)
- template<class Field, bool sorted = true, bool read_integer = false>
  void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)
- template<class Field>
  void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)
- template<class T>
  std::enable_if< std::is_integral< T >::value, int > getDataType ()
- template<class T>
  std::enable_if< std::is_floating_point< T >::value, int > getDataType ()
- template<class T>
  std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()
- template<class T>
  int getDataType ()
- template<class Field>
  void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)
- template<class Field>
  void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)
- template<class Field>
  void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t ldA)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)
- template<class Field>
  void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class It>
  double computeDeviation (It begin, It end)
- template<class Field>
  StatsMatrix getStat (const Field &F, const index_t *row, const index_t *col, typename Field::ConstElement_ptr val, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value >::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type * = nullptr>
  Field::Element_ptr ftranspose (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

- template<class Field, class enable = void>
  Field::Residu_t maxCardinality ()
- template<> uint64_t maxCardinality< Givaro::Modular< int64_t > > ()
- template<> uint32_t maxCardinality< Givaro::Modular< int32_t > > ()
- template<class Field>
  Field::Residu_t minCardinality ()
- template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)
- template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const Alignment align)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)
- template<typename RNS>
  void finit_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)
- template<typename RNS>
  void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)
- template<typename RNS>
  void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)
- template<typename RNS>
  void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const Alignment align)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)
- template<typename RNS>
  void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)
- template<typename RNS>
  void fconvert_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  freduce $x \leftarrow xmodF$.
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  freduce $x \leftarrow ymodF$.
- template INST_OR_DECL void finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  finit $x \leftarrow ymodF$.
- template INST_OR_DECL void fconvert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  fconvert $x \leftarrow ymodF$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  fnegin $x \leftarrow -x$.
- template INST_OR_DECL void fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  fneg $x \leftarrow -y$.
- template INST_OR_DECL void fzero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  fzero : $A \leftarrow 0$.

- template INST_OR_DECL bool fiszero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX)

  fiszero : test $X = 0$.
- template INST_OR_DECL bool fequal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  fequal : test $X = Y$.
- template INST_OR_DECL void fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  fassign : $x \leftarrow y$.
- template INST_OR_DECL void fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- template INST_OR_DECL void fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  fscal $y \leftarrow \alpha \cdot x$.
- template INST_OR_DECL void faxpy (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template INST_OR_DECL FFLAS_ELT fdot (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template INST_OR_DECL void fswap (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  fswap: $X \leftrightarrow Y$.
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void fsub (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void faddin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT alpha, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  fassign : $A \leftarrow B$.
- template INST_OR_DECL void fzero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  fzero : $A \leftarrow 0$.
- template INST_OR_DECL bool fequal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size↩ _t n, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb)

  fequal : test $A = B$.
- template INST_OR_DECL bool fiszero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *A, const size_t lda)

  fiszero : test $A = 0$.
- template INST_OR_DECL void fidentity (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda, const FFLAS_ELT &d)

  creates a diagonal matrix
- template INST_OR_DECL void fidentity (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  creates a diagonal matrix

- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  freduce $A \leftarrow AmodF$.
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  freduce $A \leftarrow BmodF$.
- template INST_OR_DECL void finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  finit $A \leftarrow BmodF$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  fnegin $A \leftarrow -A$.
- template INST_OR_DECL void fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  fneg $A \leftarrow -B$.
- template INST_OR_DECL void fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *A, const size_t lda)

  fscalin $A \leftarrow a \cdot A$.
- template INST_OR_DECL void fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  fscal $B \leftarrow a \cdot A$.
- template INST_OR_DECL void faxpy (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t ldx, FFLAS_ELT *Y, const size_t ldy)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template INST_OR_DECL void fmove (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fadd : matrix addition.
- template INST_OR_DECL void fsub (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fsub : matrix subtraction.
- template INST_OR_DECL void fsubin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fsubin C = C - B
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT alpha, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fadd : matrix addition with scaling.
- template INST_OR_DECL void faddin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  faddin
- template INST_OR_DECL FFLAS_ELT * fgemv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT beta, FFLAS_ELT *Y, const size_t incY)

  finite prime FFLAS_FIELD<FFLAS_ELT> GEneral Matrix Vector multiplication.
- template INST_OR_DECL void fger (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *x, const size_t incx, const FFLAS_ELT *y, const size_t incy, FFLAS_ELT *A, const size_t lda)

fger: rank one update of a general matrix

- template INST_OR_DECL void ftrsv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$
- template INST_OR_DECL void ftrsm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  ftrsm: ***TR****iangular* ***S****ystem solve with* ***M****atrix.*
- template INST_OR_DECL void ftrmm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  ftrmm: ***TR****iangular* ***M****atrix* ***M****ultiply.*
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc)

  fgemm: ***F****ield* ***GE****neral* ***M****atrix* ***M****ultiply.*
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > par)
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > par)
- template INST_OR_DECL FFLAS_ELT * fsquare (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc)

  fsquare: Squares a matrix.
- template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>
  void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)

- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>
  void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)
- template<class Field>
  void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)
- template<class Field, class RandIter>
  void pfrand (const Field &F, RandIter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)
- template<class Field, class Cut, class Param>
  Field::Element & fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace > > &H)

- template<class Field, class AlgoT, class FieldTrait>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)
- template<class Field, class AlgoT, class FieldTrait, class Cut>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)
- void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)
- char * getArgumentValue (int argc, char **argv, int i)

  Get the value of an argument and avoid core dump when no value was given after an argument.
- std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)

  writes the values of all arguments, preceded by the programName
- template<class Field>
  std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format, bool column_major)

  WriteMatrix: write a matrix to an output stream.
- void preamble (std::ifstream &ifs, FFLAS_FORMAT &format)
- template<class Field>
  Field::Element_ptr ReadMatrix (std::ifstream &ifs, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)

  ReadMatrix: read a matrix from an input stream.
- template<class Field>
  Field::Element_ptr ReadMatrix (const std::string &matrix_file, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)

  ReadMatrix: read a matrix from a file.
- template<class Field>
  void WriteMatrix (std::string &matrix_file, const Field &F, int m, int n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format=FflasDense, bool column_major=false)

  WriteMatrix: write a matrix to a file.
- std::ostream & WritePermutation (std::ostream &c, const size_t *P, size_t N)

  WritePermutation: write a permutation matrix to an output stream.
- template<class Element>
  bool alignable ()
- template<> bool alignable< Givaro::Integer * > ()
- template<class Field>
  Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)
- template<class Field>
  Field::Element_ptr fflas_new (const Field &F, const size_t m, const size_t n, const Alignment align=Alignment::DEFAULT)
- template<class Element>
  Element * fflas_new (const size_t m, const Alignment align=Alignment::DEFAULT)
- template<class Element_ptr>
  void fflas_delete (Element_ptr A)
- template<class Ptr, class ... Args>
  void fflas_delete (Ptr p, Args ... args)
- void prefetch (const int64_t *)
- void getTLBSize (int &tlb)
- void queryCacheSizes (int &l1, int &l2, int &l3)
- int queryL1CacheSize ()
- int queryTopLevelCacheSize ()
- uint64_t getSeed ()

11.1.1 Typedef Documentation

11.1.1.1 Checker_fgemm

```
template<class Field>
using Checker_fgemm = FFLAS::Checker_Empty<Field>
```

11.1.1.2 Checker_ftrsm

```
template<class Field>
using Checker_ftrsm = FFLAS::Checker_Empty<Field>
```

11.1.1.3 ForceCheck_fgemm

```
template<class Field>
using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
```

11.1.1.4 ForceCheck_ftrsm

```
template<class Field>
using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
```

11.1.1.5 ZOSparseMatrix

```
using ZOSparseMatrix = std::true_type
```

11.1.1.6 NotZOSparseMatrix

```
using NotZOSparseMatrix = std::false_type
```

11.1.1.7 SimdSparseMatrix

```
using SimdSparseMatrix = std::true_type
```

11.1.1.8 NoSimdSparseMatrix

```
using NoSimdSparseMatrix = std::false_type
```

11.1.1.9 MKLSparseMatrixFormat

```
using MKLSparseMatrixFormat = std::true_type
```

11.1.1.10 NotMKLSparseMatrixFormat

```
using NotMKLSparseMatrixFormat = std::false_type
```

11.1.1.11 has_plus

```
template<class T>
using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_plus_impl<T>>::type
```

11.1.1.12 has_minus

```
template<class T>
using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_minus_impl<T>>::type
```

11.1.1.13 has_equal

```
template<class T>
using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
std::is_copy_assignable<T>>::type
```

11.1.1.14 has_plus_eq

```
template<class T>
using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_plus_eq_impl<T>>::type
```

11.1.1.15 has_minus_eq

```
template<class T>
using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_minus_eq_impl<T>>::type
```

11.1.1.16 has_mul

```
template<class T>
using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>>↩
::type
```

11.1.1.17 has_mul_eq

```
template<class T>
using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_mul_eq_impl<T>>::type
```

11.1.1.18 Timer

```
typedef Givaro::Timer Timer
```

11.1.1.19 BaseTimer

```
typedef Givaro::BaseTimer BaseTimer
```

11.1.1.20 UserTimer

```
typedef Givaro::UserTimer UserTimer
```

11.1.1.21 SysTimer

```
typedef Givaro::SysTimer SysTimer
```

11.1.2 Enumeration Type Documentation

11.1.2.1 FFLAS_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

11.1.2.2 FFLAS_TRANSPOSE

```
enum FFLAS_TRANSPOSE
```

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

11.1.2.3 FFLAS_UPLO

```
enum FFLAS_UPLO
```

Is triangular matrix's shape upper ?

Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$)
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$)
FflasLeftTri	Triangular matrix is Left triangular (if $j > n - i - 1$ then $T_{i,j} = 0$)
FflasRightTri	Triangular matrix is Right triangular (if $j < n - i - 1$ then $T_{i,j} = 0$)

11.1.2.4 FFLAS_DIAG

`enum FFLAS_DIAG`

Is the triangular matrix implicitly unit diagonal ?

Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ($T_{i,i} = 1$)

11.1.2.5 FFLAS_SIDE

`enum FFLAS_SIDE`

On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the rigth.

11.1.2.6 FFLAS_BASE

`enum FFLAS_BASE`

`FFLAS_BASE` determines the type of the element representation for Matrix Mult kernel.

(deprecated, should not be used)

Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

11.1.2.7 number_kind

`enum number_kind`

Enumerator

zero	
one	
mone	
other	

11.1.2.8 SparseMatrix_t

```
enum class SparseMatrix_t  [strong]
```

Enumerator

CSR	
CSR_ZO	
CSC	
CSC_ZO	
COO	
COO_ZO	
ELL	
ELL_ZO	
SELL	
SELL_ZO	
ELL_simd	
ELL_simd_ZO	
CSR_HYB	
HYB_ZO	

11.1.2.9 FFLAS_FORMAT

```
enum FFLAS_FORMAT
```

Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

11.1.3 Function Documentation

11.1.3.1 InfNorm()

```
Givaro::Integer InfNorm (
            const size_t M,
            const size_t N,
            const Givaro::Integer * A,
            const size_t lda)  [inline]
```

11.1.3.2 min3()

```
template<class T>
const T & min3 (
             const T & m,
             const T & n,
             const T & k)
```

11.1.3.3 max3()

```
template<class T>
const T & max3 (
             const T & m,
             const T & n,
             const T & k)
```

11.1.3.4 min4()

```
template<class T>
const T & min4 (
             const T & m,
             const T & n,
             const T & k,
             const T & l)
```

11.1.3.5 max4()

```
template<class T>
const T & max4 (
             const T & m,
             const T & n,
             const T & k,
             const T & l)
```

11.1.3.6 fadd() [1/8]

```
template<class Field>
void fadd (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t inca,
             typename Field::ConstElement_ptr B,
             const size_t incb,
             typename Field::Element_ptr C,
             const size_t incc)
```

11.1.3.7 faddin() [1/5]

```
template<class Field>
void faddin (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t incb,
             typename Field::Element_ptr C,
             const size_t incc)
```

11.1.3.8 fsub() [1/4]

```
template<class Field>
void fsub (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t inca,
             typename Field::ConstElement_ptr B,
             const size_t incb,
             typename Field::Element_ptr C,
             const size_t incc)
```

11.1.3.9 fsubin() [1/3]

```
template<class Field>
void fsubin (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t incb,
             typename Field::Element_ptr C,
             const size_t incc)
```

11.1.3.10 fadd() [2/8]

```
template<class Field>
void fadd (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t inca,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr B,
             const size_t incb,
             typename Field::Element_ptr C,
             const size_t incc)
```

Todo optimise here

11.1.3.11 pfadd()

```
template<class Field>
void pfadd (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc,
             const size_t numths)
```

11.1.3.12 pfsub()

```
template<class Field>
void pfsub (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc,
             const size_t numths)
```

11.1.3.13 pfaddin()

```
template<class Field>
void pfaddin (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc,
             size_t numths)
```

11.1.3.14 pfsubin()

```
template<class Field>
void pfsubin (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc,
             size_t numths)
```

11.1.3.15 fadd() [3/8]

```
template<class Field>
void fadd (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              typename Field::Element_ptr C,
              const size_t ldc)
```

fadd : matrix addition.

Computes C = A + B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.16 fsub() [2/4]

```
template<class Field>
void fsub (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              typename Field::Element_ptr C,
              const size_t ldc)
```

fsub : matrix subtraction.

Computes C = A - B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN

Parameters

lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.17 faddin() [2/5]

```
template<class Field>
void faddin (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc)
```

faddin

11.1.3.18 faddin() [3/5]

```
template<class Field>
void faddin (
             const Field & F,
             const FFLAS_UPLO uplo,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc)
```

fadding for symmetric matrices

11.1.3.19 fsubin() [2/3]

```
template<class Field>
void fsubin (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr C,
             const size_t ldc)
```

fsubin C = C - B

11.1.3.20 fadd() [4/8]

```
template<class Field>
void fadd (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            typename Field::Element_ptr C,
            const size_t ldc)
```

fadd : matrix addition with scaling.

Computes C = A + alpha B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
alpha	some scalar
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.21 fassign() [1/10]

```
template<class Field>
void fassign (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr Y,
            const size_t incY,
            typename Field::Element_ptr X,
            const size_t incX)  [inline]
```

fassign : $x \leftarrow y$.

X is preallocated

Todo variant for triagular matrix

Parameters

	F	field
	N	size of the vectors
out	*X*	vector in F
	incX	stride of X
in	*Y*	vector in F
	incY	stride of Y

11.1.3.22 fassign() [2/10]

```
template<>
void fassign (
            const Givaro::Modular< float > & F,
            const size_t N,
            const float * Y,
            const size_t incY,
            float * X,
            const size_t incX)  [inline]
```

11.1.3.23 fassign() [3/10]

```
template<>
void fassign (
            const Givaro::ModularBalanced< float > & F,
            const size_t N,
            const float * Y,
            const size_t incY,
            float * X,
            const size_t incX)  [inline]
```

11.1.3.24 fassign() [4/10]

```
template<>
void fassign (
            const Givaro::ZRing< float > & F,
            const size_t N,
            const float * Y,
            const size_t incY,
            float * X,
            const size_t incX)  [inline]
```

11.1.3.25 fassign() [5/10]

```
template<>
void fassign (
            const Givaro::Modular< double > & F,
            const size_t N,
            const double * Y,
            const size_t incY,
            double * X,
            const size_t incX)  [inline]
```

11.1.3.26 fassign() [6/10]

```
template<>
void fassign (
            const Givaro::ModularBalanced< double > & F,
            const size_t N,
            const double * Y,
            const size_t incY,
            double * X,
            const size_t incX)  [inline]
```

11.1.3.27 fassign() [7/10]

```
template<>
void fassign (
              const Givaro::ZRing< double > & F,
              const size_t N,
              const double * Y,
              const size_t incY,
              double * X,
              const size_t incX)  [inline]
```

11.1.3.28 fassign() [8/10]

```
template<class Field>
void fassign (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              typename Field::Element_ptr A,
              const size_t lda)
```

fassign : $A \leftarrow B$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	vector in F
ldb	stride of B

11.1.3.29 faxpy() [1/6]

```
template<class Field>
void faxpy (
              const Field & F,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY)  [inline]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field

Parameters

	N	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
in,out	*Y*	vector in F
	incY	stride of Y

11.1.3.30 faxpy() [2/6]

```
template<>
void faxpy (
             const Givaro::DoubleDomain & ,
             const size_t N,
             const Givaro::DoubleDomain::Element a,
             Givaro::DoubleDomain::ConstElement_ptr x,
             const size_t incx,
             Givaro::DoubleDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.31 faxpy() [3/6]

```
template<>
void faxpy (
             const Givaro::FloatDomain & ,
             const size_t N,
             const Givaro::FloatDomain::Element a,
             Givaro::FloatDomain::ConstElement_ptr x,
             const size_t incx,
             Givaro::FloatDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.32 faxpy() [4/6]

```
template<class Field>
void faxpy (
             const Field & F,
             const size_t m,
             const size_t n,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr X,
             const size_t ldx,
             typename Field::Element_ptr Y,
             const size_t ldy)  [inline]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field

Parameters

	m	row dimension
	n	column dimension
	alpha	scalar
in	*X*	vector in F
	ldx	leading dimension of X
in,out	*Y*	vector in F
	ldy	leading dimension of Y

11.1.3.33 fdot() [1/11]

```
template<class Field>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::DefaultTag & MT)  [inline]
```

11.1.3.34 fdot() [2/11]

```
template<class Field>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::DelayedTag & MT)  [inline]
```

11.1.3.35 fdot() [3/11]

```
template<>
Givaro::DoubleDomain::Element fdot (
            const Givaro::DoubleDomain & ,
            const size_t N,
            Givaro::DoubleDomain::ConstElement_ptr x,
            const size_t incx,
            Givaro::DoubleDomain::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::DefaultTag & MT)  [inline]
```

11.1.3.36 fdot() [4/11]

```
template<>
Givaro::FloatDomain::Element fdot (
            const Givaro::FloatDomain & ,
            const size_t N,
            Givaro::FloatDomain::ConstElement_ptr x,
            const size_t incx,
            Givaro::FloatDomain::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::DefaultTag & MT)  [inline]
```

11.1.3.37 fdot() [5/11]

```
template<class Field, class T>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::ConvertTo< T > & MT)  [inline]
```

11.1.3.38 fdot() [6/11]

```
template<class Field>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            ModeCategories::DefaultBoundedTag & dbt)  [inline]
```

11.1.3.39 fdot() [7/11]

```
template<class Field>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            const ParSeqHelper::Sequential seq)  [inline]
```

11.1.3.40 fdot() [8/11]

```
template<class Field>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr X,
            const size_t incX,
            typename Field::ConstElement_ptr Y,
            const size_t incY)  [inline]
```

fdot: dot product $x^T y$.

Parameters

F	field
N	size of the vectors
X	vector in F
incX	stride of X
Y	vector in F
incY	stride of Y

11.1.3.41 fgemm() [1/23]

```
template<class Field>
Field::Element_ptr fgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFl
>, ParSeqHelper::Sequential > & H)  [inline]
```

11.1.3.42 fgemm() [2/23]

```
template<typename Field>
Field::Element_ptr fgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
```

```
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const ParSeqHelper::Sequential seq)  [inline]
```

11.1.3.43 fgemm() [3/23]

```
template<typename Field, class Cut, class Param>
Field::Element_ptr fgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.1.3.44 fgemm() [4/23]

```
template<typename Field>
Field::Element_ptr fgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc)  [inline]
```

fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.

Computes $C = \alpha \mathrm{op}(A) \times \mathrm{op}(B) + \beta C$ Automatically set Winograd recursion level

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ is $k \times n$
C	C is $m \times n$
lda	leading dimension of `A`
ldb	leading dimension of `B`
ldc	leading dimension of `C`
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of `w`.

Warning

α *must* be invertible

11.1.3.45 fgemm() [5/23]

```
template<typename Field, class ModeT, class ParSeq>
Field::Element_ptr fgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > & H)  [inline]
```

11.1.3.46 fgemm() [6/23]

```
template<class Field>
Field::Element_ptr fgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
```

```
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H)  [inline]
```

11.1.3.47 fsquare() [1/6]

```
template<class Field>
Field::Element_ptr fsquare (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const size_t n,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc)  [inline]
```

fsquare: Squares a matrix.

compute $C \leftarrow \alpha \mathrm{op}(A)\mathrm{op}(A) + \beta C$ over a Field F Avoid the conversion of B

Parameters

ta	if ta==FflasTrans, $\mathrm{op}(A) = A^T$.
F	field
n	size of A
alpha	scalar
beta	scalar
A	dense matrix of size nxn
lda	leading dimension of A
C	dense matrix of size nxn
ldc	leading dimension of C

Bug why double ?

11.1.3.48 fsquare() [2/6]

```
template<>
double * fsquare (
            const Givaro::ModularBalanced< double > & F,
            const FFLAS_TRANSPOSE ta,
```

```
            const size_t n,
            const double alpha,
            const double * A,
            const size_t lda,
            const double beta,
            double * C,
            const size_t ldc)  [inline]
```

11.1.3.49 fsquare() [3/6]

```
template<>
float * fsquare (
            const Givaro::ModularBalanced< float > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t n,
            const float alpha,
            const float * A,
            const size_t lda,
            const float beta,
            float * C,
            const size_t ldc)  [inline]
```

11.1.3.50 fsquare() [4/6]

```
template<>
double * fsquare (
            const Givaro::Modular< double > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t lda,
            const double beta,
            double * C,
            const size_t ldc)  [inline]
```

11.1.3.51 fsquare() [5/6]

```
template<>
float * fsquare (
            const Givaro::Modular< float > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t n,
            const float alpha,
            const float * A,
            const size_t lda,
            const float beta,
            float * C,
            const size_t ldc)  [inline]
```

11.1.3.52 fgemm() [7/23]

```
template<typename RNS, typename ParSeqTrait>
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
             const FFPACK::RNSInteger< RNS > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename FFPACK::RNSInteger< RNS >::Element alpha,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
             const size_t lda,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
             const size_t ldb,
             const typename FFPACK::RNSInteger< RNS >::Element beta,
             typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
             const size_t ldc,
             MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > & H)  [inline]
```

11.1.3.53 fgemm() [8/23]

```
template<typename RNS>
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
             const FFPACK::RNSInteger< RNS > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename FFPACK::RNSInteger< RNS >::Element alpha,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
             const size_t lda,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
             const size_t ldb,
             const typename FFPACK::RNSInteger< RNS >::Element beta,
             typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
             const size_t ldc,
             MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Sequential > & H)  [inline]
```

11.1.3.54 fgemm() [9/23]

```
template<typename RNS, typename ParSeqTrait>
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
             const FFPACK::RNSInteger< RNS > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename FFPACK::RNSInteger< RNS >::Element alpha,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
```

```
            const size_t lda,
            typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
            const size_t ldb,
            const typename FFPACK::RNSInteger< RNS >::Element beta,
            typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
            const size_t ldc,
            MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads
>, ParSeqTrait > > & H)  [inline]
```

11.1.3.55 fgemm() [10/23]

```
template<typename RNS, typename Cut, typename Param>
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
            const FFPACK::RNSInteger< RNS > & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename FFPACK::RNSInteger< RNS >::Element alpha,
            typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
            const size_t lda,
            typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
            const size_t ldb,
            const typename FFPACK::RNSInteger< RNS >::Element beta,
            typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
            const size_t ldc,
            MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
ParSeqHelper::Parallel< Cut, Param > > & H)  [inline]
```

11.1.3.56 fgemm() [11/23]

```
template<class ParSeq>
Givaro::Integer * fgemm (
            const Givaro::ZRing< Givaro::Integer > & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const Givaro::Integer alpha,
            const Givaro::Integer * A,
            const size_t lda,
            const Givaro::Integer * B,
            const size_t ldb,
            Givaro::Integer beta,
            Givaro::Integer * C,
            const size_t ldc,
            MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H)  [inline]
```

11.1.3.57 fgemm() [12/23]

```
template<typename RNS, class ModeT>
RNS::Element_ptr fgemm (
             const FFPACK::RNSInteger< RNS > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename RNS::Element alpha,
             typename RNS::ConstElement_ptr Ad,
             const size_t lda,
             typename RNS::ConstElement_ptr Bd,
             const size_t ldb,
             const typename RNS::Element beta,
             typename RNS::Element_ptr Cd,
             const size_t ldc,
             MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential
> & H)  [inline]
```

11.1.3.58 fgemm() [13/23]

```
template<typename RNS>
RNS::Element_ptr fgemm (
             const FFPACK::RNSIntegerMod< RNS > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename RNS::Element alpha,
             typename RNS::ConstElement_ptr Ad,
             const size_t lda,
             typename RNS::ConstElement_ptr Bd,
             const size_t ldb,
             const typename RNS::Element beta,
             typename RNS::Element_ptr Cd,
             const size_t ldc,
             MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > & H)  [inline]
```

11.1.3.59 fgemm() [14/23]

```
Givaro::Integer * fgemm (
             const Givaro::Modular< Givaro::Integer > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const Givaro::Integer alpha,
             const Givaro::Integer * A,
             const size_t lda,
             const Givaro::Integer * B,
```

```
            const size_t ldb,
            const Givaro::Integer beta,
            Givaro::Integer * C,
            const size_t ldc,
            MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H)  [inline]
```

11.1.3.60 fgemm() [15/23]

```
template<class ParSeq>
Givaro::Integer * fgemm (
            const Givaro::Modular< Givaro::Integer > & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const Givaro::Integer alpha,
            const Givaro::Integer * A,
            const size_t lda,
            const Givaro::Integer * B,
            const size_t ldb,
            const Givaro::Integer beta,
            Givaro::Integer * C,
            const size_t ldc,
            MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H)  [inline]
```

11.1.3.61 fgemm() [16/23]

```
template<size_t K1, size_t K2, class ParSeq>
RecInt::ruint< K1 > * fgemm (
            const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const RecInt::ruint< K1 > alpha,
            const RecInt::ruint< K1 > * A,
            const size_t lda,
            const RecInt::ruint< K1 > * B,
            const size_t ldb,
            RecInt::ruint< K1 > beta,
            RecInt::ruint< K1 > * C,
            const size_t ldc,
            MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H)  [inline]
```

11.1.3.62 fgemm() [17/23]

```
template<class Field, class ModeT>
Field::Element_ptr fgemm (
```

```
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeT > & H)  [inline]
```

11.1.3.63 fgemm() [18/23]

```
template<class Field, class ModeT, class Cut, class Param>
Field::Element_ptr fgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut,
Param > > & H)  [inline]
```

11.1.3.64 fgemv() [1/19]

```
template<class Field>
Field::Element_ptr fgemv (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr X,
            const size_t incX,
            const typename Field::Element beta,
            typename Field::Element_ptr Y,
            const size_t incY,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H)  [inline]
```

11.1.3.65 fgemv() [2/19]

```
template<class Field>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY,
              MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H)  [inline]
```

11.1.3.66 fgemv() [3/19]

```
template<class Field>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY,
              MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H)  [inline]
```

11.1.3.67 fgemv() [4/19]

```
template<class Field>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY,
              MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H)  [inline]
```

11.1.3.68 fgemv() [5/19]

```
template<class Field>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE TransA,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY)  [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

Computes $Y \leftarrow \alpha \mathrm{op}(A) X + \beta Y$.

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	lda	leading dimension of `A`
	X	dense vector of size `N`
	incX	stride of `X`
	beta	scalar
`out`	*Y*	dense vector of size `M`
	incY	stride of `Y`

11.1.3.69 fgemv() [6/19]

```
Givaro::ZRing< int64_t >::Element_ptr fgemv (
              const Givaro::ZRing< int64_t > & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
              const size_t N,
              const int64_t alpha,
              const int64_t * A,
              const size_t lda,
              const int64_t * X,
              const size_t incX,
              const int64_t beta,
              int64_t * Y,
              const size_t incY,
              MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.70 fgemv() [7/19]

```
Givaro::DoubleDomain::Element_ptr fgemv (
            const Givaro::DoubleDomain & F,
            const FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            const Givaro::DoubleDomain::Element alpha,
            const Givaro::DoubleDomain::ConstElement_ptr A,
            const size_t lda,
            const Givaro::DoubleDomain::ConstElement_ptr X,
            const size_t incX,
            const Givaro::DoubleDomain::Element beta,
            Givaro::DoubleDomain::Element_ptr Y,
            const size_t incY,
            MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.71 fgemv() [8/19]

```
template<class Field>
Field::Element_ptr fgemv (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::ConstElement_ptr X,
            const size_t incX,
            const typename Field::Element beta,
            typename Field::Element_ptr Y,
            const size_t incY,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H)
[inline]
```

11.1.3.72 fgemv() [9/19]

```
Givaro::FloatDomain::Element_ptr fgemv (
            const Givaro::FloatDomain & F,
            const FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            const Givaro::FloatDomain::Element alpha,
            const Givaro::FloatDomain::ConstElement_ptr A,
            const size_t lda,
            const Givaro::FloatDomain::ConstElement_ptr X,
            const size_t incX,
            const Givaro::FloatDomain::Element beta,
            Givaro::FloatDomain::Element_ptr Y,
            const size_t incY,
            MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.73 fgemv() [10/19]

```
template<class Field, class Cut, class Param>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t m,
              const size_t n,
              const typename Field::Element alpha,
              const typename Field::ConstElement_ptr A,
              const size_t lda,
              const typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY,
              ParSeqHelper::Parallel< Cut, Param > & parH)
```

11.1.3.74 fgemv() [11/19]

```
template<class Field>
Field::Element_ptr fgemv (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t m,
              const size_t n,
              const typename Field::Element alpha,
              const typename Field::ConstElement_ptr A,
              const size_t lda,
              const typename Field::ConstElement_ptr X,
              const size_t incX,
              const typename Field::Element beta,
              typename Field::Element_ptr Y,
              const size_t incY,
              ParSeqHelper::Sequential & seqH)
```

11.1.3.75 fgemv() [12/19]

```
FFPACK::rns_double::Element_ptr fgemv (
              const FFPACK::RNSInteger< FFPACK::rns_double > & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
              const size_t N,
              const FFPACK::rns_double::Element alpha,
              FFPACK::rns_double::ConstElement_ptr A,
              const size_t lda,
              FFPACK::rns_double::ConstElement_ptr X,
              const size_t incX,
              const FFPACK::rns_double::Element beta,
              FFPACK::rns_double::Element_ptr Y,
              const size_t incY,
              MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::Defau
> & H)  [inline]
```

11.1.3.76 fgemv() [13/19]

```
FFPACK::rns_double::Element_ptr fgemv (
            const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            const FFPACK::rns_double::Element alpha,
            FFPACK::rns_double::ConstElement_ptr A,
            const size_t lda,
            FFPACK::rns_double::ConstElement_ptr X,
            const size_t incX,
            const FFPACK::rns_double::Element beta,
            FFPACK::rns_double::Element_ptr Y,
            const size_t incY,
            MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic,
ModeCategories::DefaultTag > & H)  [inline]
```

11.1.3.77 fgemv() [14/19]

```
Givaro::Integer * fgemv (
            const Givaro::ZRing< Givaro::Integer > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t m,
            const size_t n,
            const Givaro::Integer alpha,
            Givaro::Integer * A,
            const size_t lda,
            Givaro::Integer * X,
            const size_t ldx,
            Givaro::Integer beta,
            Givaro::Integer * Y,
            const size_t ldy,
            MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H)  [inline]
```

11.1.3.78 fgemv() [15/19]

```
Givaro::Integer * fgemv (
            const Givaro::Modular< Givaro::Integer > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t m,
            const size_t n,
            const Givaro::Integer alpha,
            Givaro::Integer * A,
            const size_t lda,
            Givaro::Integer * X,
            const size_t ldx,
            Givaro::Integer beta,
            Givaro::Integer * Y,
            const size_t ldy,
            MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H)  [inline]
```

11.1.3.79 fgemv() [16/19]

```
template<size_t K1, size_t K2, class ParSeq>
RecInt::ruint< K1 > * fgemv (
             const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
             const FFLAS_TRANSPOSE ta,
             const size_t m,
             const size_t n,
             const RecInt::ruint< K1 > alpha,
             const RecInt::ruint< K1 > * A,
             const size_t lda,
             const RecInt::ruint< K1 > * X,
             const size_t incx,
             RecInt::ruint< K1 > beta,
             RecInt::ruint< K1 > * Y,
             const size_t incy,
             MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H)  [inline]
```

11.1.3.80 fger() [1/12]

```
template<class Field>
void fger (
             const Field & F,
             const size_t M,
             const size_t N,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr x,
             const size_t incx,
             typename Field::ConstElement_ptr y,
             const size_t incy,
             typename Field::Element_ptr A,
             const size_t lda)  [inline]
```

fger: rank one update of a general matrix

Computes $A \leftarrow \alpha x.y^T + A$

Parameters

	F	field
	M	rows
	N	cols
	alpha	scalar
in,out	*A*	dense matrix of size MxN and leading dimension lda
	lda	leading dimension of A
	x	dense vector of size M
	incx	stride of X
	y	dense vector of size N
	incy	stride of Y

11.1.3.81 fger() [2/12]

```
template<class Field>
void fger (
            const Field & F,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            typename Field::Element_ptr A,
            const size_t lda,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H)  [inline]
```

11.1.3.82 fger() [3/12]

```
template<class Field, class AnyTag>
void fger (
            const Field & F,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            typename Field::Element_ptr A,
            const size_t lda,
            MMHelper< Field, MMHelperAlgo::Classic, AnyTag > & H)  [inline]
```

11.1.3.83 fger() [4/12]

```
void fger (
            const Givaro::DoubleDomain & F,
            const size_t M,
            const size_t N,
            const Givaro::DoubleDomain::Element alpha,
            const Givaro::DoubleDomain::ConstElement_ptr x,
            const size_t incx,
            const Givaro::DoubleDomain::ConstElement_ptr y,
            const size_t incy,
            Givaro::DoubleDomain::Element_ptr A,
            const size_t lda,
            MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.84 fger() [5/12]

```
template<class Field>
void fger (
             const Field & F,
             const size_t M,
             const size_t N,
             const typename Field::Element alpha,
             const typename Field::ConstElement_ptr x,
             const size_t incx,
             const typename Field::ConstElement_ptr y,
             const size_t incy,
             typename Field::Element_ptr A,
             const size_t lda,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H)
[inline]
```

11.1.3.85 fger() [6/12]

```
void fger (
             const Givaro::FloatDomain & F,
             const size_t M,
             const size_t N,
             const Givaro::FloatDomain::Element alpha,
             const Givaro::FloatDomain::ConstElement_ptr x,
             const size_t incx,
             const Givaro::FloatDomain::ConstElement_ptr y,
             const size_t incy,
             Givaro::FloatDomain::Element_ptr A,
             const size_t lda,
             MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.86 fger() [7/12]

```
template<class Field>
void fger (
             const Field & F,
             const size_t M,
             const size_t N,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr x,
             const size_t incx,
             typename Field::ConstElement_ptr y,
             const size_t incy,
             typename Field::Element_ptr A,
             const size_t lda,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H)  [inline]
```

11.1.3.87 fger() [8/12]

```
template<class Field>
void fger (
            const Field & F,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            typename Field::Element_ptr A,
            const size_t lda,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H)  [inline]
```

11.1.3.88 fger() [9/12]

```
void fger (
            const Givaro::Modular< Givaro::Integer > & F,
            const size_t M,
            const size_t N,
            const typename Givaro::Integer alpha,
            typename Givaro::Integer * x,
            const size_t incx,
            typename Givaro::Integer * y,
            const size_t incy,
            typename Givaro::Integer * A,
            const size_t lda,
            MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H)  [inline]
```

11.1.3.89 fger() [10/12]

```
template<typename RNS>
void fger (
            const FFPACK::RNSInteger< RNS > & F,
            const size_t M,
            const size_t N,
            const typename FFPACK::RNSInteger< RNS >::Element alpha,
            typename FFPACK::RNSInteger< RNS >::Element_ptr x,
            const size_t incx,
            typename FFPACK::RNSInteger< RNS >::Element_ptr y,
            const size_t incy,
            typename FFPACK::RNSInteger< RNS >::Element_ptr A,
            const size_t lda,
            MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.90 fger() [11/12]

```
template<typename RNS>
void fger (
              const FFPACK::RNSIntegerMod< RNS > & F,
              const size_t M,
              const size_t N,
              const typename FFPACK::RNSIntegerMod< RNS >::Element alpha,
              typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x,
              const size_t incx,
              typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y,
              const size_t incy,
              typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A,
              const size_t lda,
              MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > & H)  [inline]
```

11.1.3.91 freduce() [1/11]

```
template<class Field>
void freduce (
              const Field & F,
              const size_t n,
              typename Field::ConstElement_ptr Y,
              const size_t incY,
              typename Field::Element_ptr X,
              const size_t incX)
```

freduce $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors
Y	vector of `Element`
incY	stride of `Y`
X	vector in `F`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.92 freduce() [2/11]

```
template<class Field>
void freduce (
              const Field & F,
              const size_t n,
              typename Field::Element_ptr X,
              const size_t incX)
```

freduce $x \leftarrow xmodF$.

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.93 freduce_constoverride() [1/2]

```
template<class Field>
void freduce_constoverride (
             const Field & F,
             const size_t m,
             typename Field::ConstElement_ptr A,
             const size_t incX)
```

11.1.3.94 finit() [1/8]

```
template<class Field, class ConstOtherElement_ptr>
void finit (
             const Field & F,
             const size_t n,
             ConstOtherElement_ptr Y,
             const size_t incY,
             typename Field::Element_ptr X,
             const size_t incX)
```

11.1.3.95 finit() [2/8]

```
template<class Field>
void finit (
             const Field & F,
             const size_t n,
             typename Field::Element_ptr X,
             const size_t incX)
```

finit Initializes X in F$.

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X

11.1.3.96 freduce() [3/11]

```
template<class Field>
void freduce (
             const Field & F,
             const size_t m,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda)
```

freduce $A \leftarrow AmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.97 freduce() [4/11]

```
template<class Field>
void freduce (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda)
```

freduce for square symmetric matrices

11.1.3.98 pfreduce()

```
template<class Field>
void pfreduce (
             const Field & F,
             const size_t m,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t numths)
```

11.1.3.99 freduce() [5/11]

```
template<class Field>
void freduce (
             const Field & F,
             const size_t m,
             const size_t n,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             typename Field::Element_ptr A,
             const size_t lda)
```

freduce $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`
B	matrix in `Element`
ldb	stride of `B`

11.1.3.100 freduce_constoverride() [2/2]

```
template<class Field>
void freduce_constoverride (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::ConstElement_ptr A,
              const size_t lda)
```

11.1.3.101 finit() [3/8]

```
template<class Field, class OtherElement_ptr>
void finit (
              const Field & F,
              const size_t m,
              const size_t n,
              const OtherElement_ptr B,
              const size_t ldb,
              typename Field::Element_ptr A,
              const size_t lda)
```

finit $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`
B	matrix in `OtherElement`
ldb	stride of `B`

11.1.3.102 finit() [4/8]

```
template<class Field>
void finit (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::Element_ptr A,
              const size_t lda)
```

11.1.3.103 freduce() [6/11]

```
template<>
void freduce (
              const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
              const size_t n,
              FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
              size_t inc)  [inline]
```

11.1.3.104 freduce() [7/11]

```
template<>
void freduce (
              const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
              const size_t m,
              const size_t n,
              FFPACK::rns_double::Element_ptr A,
              size_t lda)  [inline]
```

11.1.3.105 freivalds()

```
template<class Field>
bool freivalds (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const FFLAS_TRANSPOSE tb,
              const size_t m,
              const size_t n,
              const size_t k,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              typename Field::ConstElement_ptr C,
              const size_t ldc)  [inline]
```

freivalds: **F**reivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.

Randomly Checks $C = \alpha \mathrm{op}(A) \times \mathrm{op}(B)$

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ is $k \times n$
C	C is $m \times n$
lda	leading dimension of `A`
ldb	leading dimension of `B`
ldc	leading dimension of `C`

11.1.3.106 fscalin() [1/10]

```
template<class Field>
void fscalin (
              const Field & F,
              const size_t n,
              const typename Field::Element alpha,
              typename Field::Element_ptr X,
              const size_t incX)  [inline]
```

fscalin $x \leftarrow \alpha \cdot x$.

Parameters

F	field
n	size of the vectors
alpha	scalar
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.107 fscal() [1/10]

```
template<class Field>
void fscal (
              const Field & F,
              const size_t n,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY)  [inline]
```

fscal $y \leftarrow \alpha \cdot x$.

Parameters

	F	field
	n	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
out	*Y*	vector in F
	incY	stride of Y

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.108 fscal() [2/10]

```
template<>
void fscal (
             const Givaro::DoubleDomain & ,
             const size_t N,
             const Givaro::DoubleDomain::Element a,
             Givaro::DoubleDomain::ConstElement_ptr x,
             const size_t incx,
             Givaro::DoubleDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.109 fscal() [3/10]

```
template<>
void fscal (
             const Givaro::FloatDomain & ,
             const size_t N,
             const Givaro::FloatDomain::Element a,
             Givaro::FloatDomain::ConstElement_ptr x,
             const size_t incx,
             Givaro::FloatDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.110 fscalin() [2/10]

```
template<>
void fscalin (
             const Givaro::DoubleDomain & ,
             const size_t N,
             const Givaro::DoubleDomain::Element a,
             Givaro::DoubleDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.111 fscalin() [3/10]

```
template<>
void fscalin (
             const Givaro::FloatDomain & ,
             const size_t N,
             const Givaro::FloatDomain::Element a,
             Givaro::FloatDomain::Element_ptr y,
             const size_t incy)  [inline]
```

11.1.3.112 fscalin() [4/10]

```
template<class Field>
void fscalin (
             const Field & F,
             const size_t m,
             const size_t n,
             const typename Field::Element alpha,
             typename Field::Element_ptr A,
             const size_t lda)  [inline]
```

fscalin $A \leftarrow a \cdot A$.

Parameters

F	field
m	number of rows
n	number of cols
alpha	homotecie scalar
A	matrix in F
lda	stride of A

11.1.3.113 fscal() [4/10]

```
template<class Field>
void fscal (
             const Field & F,
             const size_t m,
             const size_t n,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
             const size_t ldb)  [inline]
```

fscal $B \leftarrow a \cdot A$.

Parameters

	F	field
	m	number of rows
	n	number of cols
	alpha	homotecie scalar
in	*A*	matrix in F
	lda	stride of A
out	*B*	matrix in F
	ldb	stride of B

11.1.3.114 fscalin() [5/10]

```
template<>
void fscalin (
             const FFPACK::RNSInteger< FFPACK::rns_double > & F,
             const size_t n,
             const FFPACK::rns_double::Element alpha,
             FFPACK::rns_double::Element_ptr A,
             const size_t inc)  [inline]
```

11.1.3.115 fscal() [5/10]

```
template<>
void fscal (
            const FFPACK::RNSInteger< FFPACK::rns_double > & F,
            const size_t n,
            const FFPACK::rns_double::Element alpha,
            FFPACK::rns_double::ConstElement_ptr A,
            const size_t Ainc,
            FFPACK::rns_double::Element_ptr B,
            const size_t Binc)  [inline]
```

11.1.3.116 fscalin() [6/10]

```
template<>
void fscalin (
            const FFPACK::RNSInteger< FFPACK::rns_double > & F,
            const size_t m,
            const size_t n,
            const FFPACK::rns_double::Element alpha,
            FFPACK::rns_double::Element_ptr A,
            const size_t lda)  [inline]
```

11.1.3.117 fscal() [6/10]

```
template<>
void fscal (
            const FFPACK::RNSInteger< FFPACK::rns_double > & F,
            const size_t m,
            const size_t n,
            const FFPACK::rns_double::Element alpha,
            FFPACK::rns_double::ConstElement_ptr A,
            const size_t lda,
            FFPACK::rns_double::Element_ptr B,
            const size_t ldb)  [inline]
```

11.1.3.118 fscalin() [7/10]

```
template<>
void fscalin (
            const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
            const size_t n,
            const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha,
            typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
            const size_t inc)  [inline]
```

11.1.3.119 fscal() [7/10]

```
template<>
void fscal (
             const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
             const size_t n,
             const FFPACK::rns_double::Element alpha,
             FFPACK::rns_double::ConstElement_ptr A,
             const size_t Ainc,
             FFPACK::rns_double::Element_ptr B,
             const size_t Binc)  [inline]
```

11.1.3.120 fscalin() [8/10]

```
template<>
void fscalin (
             const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
             const size_t m,
             const size_t n,
             const FFPACK::rns_double::Element alpha,
             FFPACK::rns_double::Element_ptr A,
             const size_t lda)  [inline]
```

11.1.3.121 fscal() [8/10]

```
template<>
void fscal (
             const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
             const size_t m,
             const size_t n,
             const FFPACK::rns_double::Element alpha,
             FFPACK::rns_double::ConstElement_ptr A,
             const size_t lda,
             FFPACK::rns_double::Element_ptr B,
             const size_t ldb)  [inline]
```

11.1.3.122 fsyr2k()

```
template<class Field>
Field::Element_ptr fsyr2k (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc)  [inline]
```

fsyr2k: Symmetric Rank 2K update

Computes the Lower or Upper triangular part of $C = \alpha(A \times B^T + B \times A^T) + \beta C$ or $C = \alpha(A^T \times B + B^T \times A) + \beta C$

Parameters

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then compute $C = \alpha(A \times B^T + B \times A^T) + \beta C$, else $C = \alpha(A^T \times B + B^T \times A) + \beta C$
n	order of matrix C
k	see A
alpha	scalar
A	A is $n \times k$ (FflasNoTrans) or A is $k \times n$ (FflasTrans)
lda	leading dimension of A
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.123 fsyrk() [1/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc)  [inline]
```

fsyrk: Symmetric Rank K update

Computes the Lower or Upper triangular part of $C = \alpha A \times A^T + \beta C$ or $C = \alpha A^T \times A + \beta C$

Parameters

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then comput $C = \alpha A \times A^T + \beta C$, else $C = \alpha A^T \times A + \beta C$
n	order of matrix C
k	see A
alpha	scalar
A	A is $n \times k$ or A is $k \times n$
lda	leading dimension of A
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.124 fsyrk() [2/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             const ParSeqHelper::Sequential seq)  [inline]
```

11.1.3.125 fsyrk() [3/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H)  [inline]
```

11.1.3.126 fsyrk() [4/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
>, ParSeqHelper::Sequential > & H)  [inline]
```

11.1.3.127 fsyrk() [5/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H)  [inline]
```

11.1.3.128 fsyrk() [6/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H)  [inline]
```

11.1.3.129 fsyrk() [7/16]

```
template<class Field, typename Mode>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > & H)  [inline]
```

11.1.3.130 fsyrk() [8/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H)
[inline]
```

11.1.3.131 fsyrk() [9/16]

```
Givaro::FloatDomain::Element_ptr fsyrk (
             const Givaro::FloatDomain & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const Givaro::FloatDomain::Element alpha,
             Givaro::FloatDomain::ConstElement_ptr A,
             const size_t lda,
             const Givaro::FloatDomain::Element beta,
             Givaro::FloatDomain::Element_ptr C,
             const size_t ldc,
             MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.132 fsyrk() [10/16]

```
Givaro::DoubleDomain::Element_ptr fsyrk (
             const Givaro::DoubleDomain & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const Givaro::DoubleDomain::Element alpha,
             Givaro::DoubleDomain::ConstElement_ptr A,
             const size_t lda,
             const Givaro::DoubleDomain::Element beta,
             Givaro::DoubleDomain::Element_ptr C,
             const size_t ldc,
             MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H)  [inline]
```

11.1.3.133 fsyrk() [11/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr D,
             const size_t incD,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD)  [inline]
```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of $C = \alpha A \times D \times A^T + \beta C$ or $C = \alpha A^T \times D \times A + \beta C$ where D is a diagonal matrix. Matrix A is updated into $D \times A$ (if trans = FflasTrans) or $A \times D$ (if trans = FflasNoTrans).

Parameters

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then compute $C = \alpha A \times A^T + \beta C$, else $C = \alpha A^T \times A + \beta C$
n	order of matrix C
k	see A
alpha	scalar
A	A is $n \times k$ or A is $k \times n$
lda	leading dimension of A
D	D is $k \times k$ diagonal matrix, stored as a vector of k coefficients
lda	leading dimension of A
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.134 fsyrk() [12/16]

```
template<class Field>
Field::Element_ptr fsyrk (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
```

```
            const size_t K,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr D,
            const size_t incD,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const ParSeqHelper::Sequential seq,
            const size_t threshold)  [inline]
```

11.1.3.135 fsyrk() [13/16]

```
template<class Field, class Cut, class Param>
Field::Element_ptr fsyrk (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr D,
            const size_t incD,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const ParSeqHelper::Parallel< Cut, Param > par,
            const size_t threshold)  [inline]
```

11.1.3.136 fsyrk() [14/16]

```
template<class Field>
Field::Element_ptr fsyrk (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr D,
            const size_t incD,
            const std::vector< bool > & twoBlock,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD)  [inline]
```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of $C = \alpha A \times DeltaD \times A^T + \beta C$ or $C = \alpha A^T \times DeltaD \times A + \beta C$ where `D` is a diagonal matrix and `Delta` is a block diagonal with either 1 on the diagonal or 2x2 swap blocks Matrix `A` is updated into $D \times A$ (if trans = FflasTrans) or $A \times D$ (if trans = FflasNoTrans).

Parameters

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then compute $C = \alpha ADeltaD \times A^T + \beta C$, else $C = \alpha A^T DeltaD \times A + \beta C$
n	see B
k	see A
alpha	scalar
A	A is $n \times k$ or A is $k \times n$
lda	leading dimension of A
D	D is $k \times k$ diagonal matrix, stored as a vector of k coefficients
twoBlocks	a vector boolean indicating the beginning of each 2x2 blocs in Delta
lda	leading dimension of A
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.137 computeS1S2()

```
template<class Field, class FieldTrait>
void computeS1S2 (
              const Field & F,
              const FFLAS_TRANSPOSE trans,
              const size_t N,
              const size_t K,
              const typename Field::Element x,
              const typename Field::Element y,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr S,
              const size_t lds,
              typename Field::Element_ptr T,
              const size_t ldt,
              MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.1.3.138 fsyrk() [15/16]

```
template<class Field>
Field::Element_ptr fsyrk (
              const Field & F,
              const FFLAS_UPLO UpLo,
              const FFLAS_TRANSPOSE trans,
              const size_t N,
              const size_t K,
              const typename Field::Element alpha,
```

```
            typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H)  [inline]
```

11.1.3.139 fsyrk() [16/16]

```
template<class Field, class Mode>
Field::Element_ptr fsyrk (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, Mode > & H)  [inline]
```

11.1.3.140 fsyrk_strassen() [1/2]

```
template<class Field, class FieldTrait>
Field::Element_ptr fsyrk_strassen (
            const Field & F,
            const FFLAS_UPLO uplo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element y1,
            const typename Field::Element y2,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.1.3.141 ftrmm() [1/3]

```
template<class Field>
void ftrmm (
            const Field & F,
            const FFLAS_SIDE Side,
            const FFLAS_UPLO Uplo,
            const FFLAS_TRANSPOSE TransA,
            const FFLAS_DIAG Diag,
```

```
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb)  [inline]
```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes $B \leftarrow \alpha \mathrm{op}(A) B$ or $B \leftarrow \alpha B \mathrm{op}(A)$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`

11.1.3.142 ftrmm() [2/3]

```
template<class Field>
void ftrmm (
            const Field & F,
            const FFLAS_SIDE Side,
            const FFLAS_UPLO Uplo,
            const FFLAS_TRANSPOSE TransA,
            const FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc)  [inline]
```

ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha \mathrm{op}(A) B + beta C$ or $C \leftarrow \alpha B \mathrm{op}(A) + beta C$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.

Parameters

Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`
beta	scalar
C	matrix of size `MxN`
ldc	leading dim of `C`

11.1.3.143 ftrsm() [1/9]

```
template<class Field>
void ftrsm (
              const Field & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr B,
              const size_t ldb)  [inline]
```

11.1.3.144 ftrsm() [2/9]

```
template<class Field>
void ftrsm (
              const Field & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr B,
              const size_t ldb,
              const ParSeqHelper::Sequential & PSH)  [inline]
```

11.1.3.145 ftrsm() [3/9]

```
template<class Field, class Cut, class Param>
void ftrsm (
              const Field & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr B,
              const size_t ldb,
              const ParSeqHelper::Parallel< Cut, Param > & PSH)  [inline]
```

11.1.3.146 ftrsm() [4/9]

```
template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>
void ftrsm (
              const Field & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr B,
              const size_t ldb,
              TRSMHelper< StructureHelper::Recursive, ParSeqTrait > & H)  [inline]
```

11.1.3.147 ftrsm() [5/9]

```
void ftrsm (
              const Givaro::Modular< Givaro::Integer > & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const Givaro::Integer alpha,
              const Givaro::Integer * A,
              const size_t lda,
              Givaro::Integer * B,
              const size_t ldb)  [inline]
```

11.1.3.148 cblas_imptrsm()

```
void cblas_imptrsm (
            const enum FFLAS_ORDER Order,
            const enum FFLAS_SIDE Side,
            const enum FFLAS_UPLO Uplo,
            const enum FFLAS_TRANSPOSE TransA,
            const enum FFLAS_DIAG Diag,
            const int M,
            const int N,
            const FFPACK::rns_double_elt alpha,
            FFPACK::rns_double_elt_cstptr A,
            const int lda,
            FFPACK::rns_double_elt_ptr B,
            const int ldb)  [inline]
```

11.1.3.149 ftrsv() [1/2]

```
template<class Field>
void ftrsv (
            const Field & F,
            const FFLAS_UPLO Uplo,
            const FFLAS_TRANSPOSE TransA,
            const FFLAS_DIAG Diag,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            int incX)  [inline]
```

ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

Parameters

F	field
X	vector of size `N` on a field `F`
incX	stride of `X`
A	a matrix of leading dimension `lda` and size `N`
lda	leading dimension of `A`
N	number of rows or columns of `A` according to `TransA`
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

11.1.3.150 igemm_()

```
void igemm_ (
            const enum FFLAS_ORDER Order,
            const enum FFLAS_TRANSPOSE TransA,
            const enum FFLAS_TRANSPOSE TransB,
            const size_t M,
```

```
            const size_t N,
            const size_t K,
            const int64_t alpha,
            const int64_t * A,
            const size_t lda,
            const int64_t * B,
            const size_t ldb,
            const int64_t beta,
            int64_t * C,
            const size_t ldc)  [inline]
```

11.1.3.151 finit() [5/8]

```
template<class Field, class OtherElement_ptr>
void finit (
            const Field & F,
            const size_t n,
            const OtherElement_ptr Y,
            const size_t incY,
            typename Field::Element_ptr X,
            const size_t incX)
```

finit $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors
Y	vector of `OtherElement`
incY	stride of `Y`
X	vector in `F`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.152 fconvert() [1/3]

```
template<class Field, class OtherElement_ptr>
void fconvert (
            const Field & F,
            const size_t n,
            OtherElement_ptr X,
            const size_t incX,
            typename Field::ConstElement_ptr Y,
            const size_t incY)
```

fconvert $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors

Y	vector of F
incY	stride of Y
X	vector in OtherElement
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.153 fnegin() [1/4]

```
template<class Field>
void fnegin (
              const Field & F,
              const size_t n,
              typename Field::Element_ptr X,
              const size_t incX)
```

fnegin $x \leftarrow -x$.

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.154 fneg() [1/4]

```
template<class Field>
void fneg (
              const Field & F,
              const size_t n,
              typename Field::ConstElement_ptr Y,
              const size_t incY,
              typename Field::Element_ptr X,
              const size_t incX)
```

fneg $x \leftarrow -y$.

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X
Y	vector in F
incY	stride of Y

Bug use cblas_(d)scal when possible

11.1.3.155 fzero() [1/5]

```
template<class Field>
void fzero (
              const Field & F,
              const size_t n,
              typename Field::Element_ptr X,
              const size_t incX)
```

fzero : $A \leftarrow 0$.

Parameters

F	field
n	number of elements to zero
X	vector in F
incX	stride of X

11.1.3.156 frand() [1/2]

```
template<class Field, class RandIter>
void frand (
              const Field & F,
              RandIter & G,
              const size_t n,
              typename Field::Element_ptr X,
              const size_t incX)
```

frand : $A \leftarrow random$.

Parameters

F	field
G	randomiterator
n	number of elements to randomize
X	vector in F
incX	stride of X

11.1.3.157 fiszero() [1/4]

```
template<class Field>
bool fiszero (
              const Field & F,
              const size_t n,
              typename Field::ConstElement_ptr X,
              const size_t incX)
```

fiszero : test $X = 0$.

Parameters

F	field
n	vector dimension
X	vector in F
incX	increment of X

11.1.3.158 fequal() [1/4]

```
template<class Field>
bool fequal (
             const Field & F,
             const size_t n,
             typename Field::ConstElement_ptr X,
             const size_t incX,
             typename Field::ConstElement_ptr Y,
             const size_t incY)
```

fequal : test $X = Y$.

Parameters

F	field
n	vector dimension
X	vector in F
incX	increment of X
Y	vector in F
incY	increment of Y

11.1.3.159 faxpby() [1/2]

```
template<class Field>
void faxpby (
             const Field & F,
             const size_t N,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr X,
             const size_t incX,
             const typename Field::Element beta,
             typename Field::Element_ptr Y,
             const size_t incY)
```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	N	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X

	beta	scalar
in,out	*Y*	vector in F
	incY	stride of Y

Note

this is a catlas function

11.1.3.160 fdot() [9/11]

```
template<typename Field, class Cut, class Param>
Field::Element fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr X,
            const size_t incX,
            typename Field::ConstElement_ptr Y,
            const size_t incY,
            const ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.1.3.161 fswap() [1/2]

```
template<class Field>
void fswap (
            const Field & F,
            const size_t N,
            typename Field::Element_ptr X,
            const size_t incX,
            typename Field::Element_ptr Y,
            const size_t incY)
```

fswap: $X \leftrightarrow Y$.

Bug use cblas_dswap when double

Parameters

F	field
N	size of the vectors
X	vector in F
incX	stride of X
Y	vector in F
incY	stride of Y

11.1.3.162 fzero() [2/5]

```
template<class Field>
void fzero (
             const Field & F,
             const size_t m,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda)
```

fzero : $A \leftarrow 0$.

Parameters

F	field
m	number of rows to zero
n	number of cols to zero
A	matrix in F
lda	stride of A

Warning

may be buggy if Element is larger than int

11.1.3.163 fzero() [3/5]

```
template<class Field>
void fzero (
             const Field & F,
             const FFLAS_UPLO shape,
             const FFLAS_DIAG diag,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda)
```

fzero : $A \leftarrow 0$ for a triangular matrix.

Parameters

F	field
shape	shape of the triangular matrix
m	number of rows to zero
n	number of cols to zero
A	matrix in F
lda	stride of A

Warning

may be buggy if Element is larger than int

11.1.3.164 frand() [2/2]

```
template<class Field, class RandIter>
void frand (
             const Field & F,
             RandIter & G,
             const size_t m,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda)
```

frand : $A \leftarrow random.$

Parameters

F	field
G	randomiterator
m	number of rows to randomize
n	number of cols to randomize
A	matrix in $\mathbb{F}$
lda	stride of $\mathbb{A}$

11.1.3.165 fequal() [2/4]

```
template<class Field>
bool fequal (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb)
```

fequal : test $A = B$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in $\mathbb{F}$
lda	leading dimension of A
B	m x n matrix in $\mathbb{F}$
ldb	leading dimension of B

11.1.3.166 fiszero() [2/4]

```
template<class Field>
bool fiszero (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::ConstElement_ptr A,
              const size_t lda)
```

fiszero : test $A = 0$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in $\mathbb{F}$
lda	leading dimension of A

11.1.3.167 fidentity() [1/4]

```
template<class Field>
void fidentity (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::Element_ptr A,
              const size_t lda,
              const typename Field::Element & d)
```

creates a diagonal matrix

11.1.3.168 fidentity() [2/4]

```
template<class Field>
void fidentity (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::Element_ptr A,
              const size_t lda)
```

creates a diagonal matrix

11.1.3.169 finit() [6/8]

```
template<class Field, class OtherElement_ptr>
void finit (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::Element_ptr A,
              const size_t lda)
```

finit Initializes A in F$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.170 fconvert() [2/3]

```
template<class Field, class OtherElement_ptr>
void fconvert (
              const Field & F,
              const size_t m,
              const size_t n,
              OtherElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb)
```

fconvert $A \leftarrow B mod F$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `OtherElement`
lda	stride of `A`
B	matrix in `F`
ldb	stride of `B`

Todo check if n == lda

11.1.3.171 fnegin() [2/4]

```
template<class Field>
void fnegin (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::Element_ptr A,
              const size_t lda)
```

fnegin $A \leftarrow -A$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`

Todo check if n == lda

11.1.3.172 fneg() [2/4]

```
template<class Field>
void fneg (
              const Field & F,
              const size_t m,
              const size_t n,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              typename Field::Element_ptr A,
              const size_t lda)
```

fneg $A \leftarrow -B$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

Todo check if n == lda

11.1.3.173 faxpby() [2/2]

```
template<class Field>
void faxpby (
             const Field & F,
             const size_t m,
             const size_t n,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr X,
             const size_t ldx,
             const typename Field::Element beta,
             typename Field::Element_ptr Y,
             const size_t ldy)
```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	m	row dimension
	n	column dimension
	alpha	scalar
in	*X*	vector in F
	ldx	leading dimension of X
	beta	scalar
in,out	*Y*	vector in F
	ldy	leading dimension of Y

Note

this is a catlas function

11.1.3.174 fmove() [1/2]

```
template<class Field>
void fmove (
             const Field & F,
             const size_t m,
             const size_t n,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
             const size_t ldb)
```

fmove : $A \leftarrow B$ and $B \leftarrow 0$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in `F`
lda	stride of `A`
B	matrix in `F`
ldb	stride of `B`

11.1.3.175 bitsize()

```
template<class Field>
size_t bitsize (
             const Field & F,
             size_t M,
             size_t N,
             const typename Field::ConstElement_ptr A,
             size_t lda)  [inline]
```

bitsize: Computes the largest bitsize of the matrix' coefficients.

If the matrix is over a modular prime field, it returns the bitsize of the largest element (in a bsolute value)

Parameters

F	field
M	rows
N	cols
incX	stride of `X`
A	a matrix of leading dimension `lda` and size `MxN`
lda	leading dimension of `A`

11.1.3.176 bitsize< Givaro::ZRing< Givaro::Integer > >()

```
template<>
size_t bitsize< Givaro::ZRing< Givaro::Integer > > (
             const Givaro::ZRing< Givaro::Integer > & F,
             size_t M,
             size_t N,
             const Givaro::Integer * A,
             size_t lda)  [inline]
```

11.1.3.177 ftrmv()

```
template<class Field>
void ftrmv (
              const Field & F,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::Element_ptr X,
              int incX)
```

ftrsm: TRiangular Matrix Vector prodcut Computes $X \leftarrow \mathrm{op}(A)X$

Parameters

F	field
X	vector of size `N` on a field `F`
incX	stride of `X`
A	a matrix of leading dimension `lda` and size `N`
lda	leading dimension of `A`
N	number of rows and columns of `A`
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^T$.
Diag	if `Diag==FflasUnit` then `A` is unit diagonal.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

11.1.3.178 ftrsm() [6/9]

```
template<class Field>
void ftrsm (
              const Field & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::Element_ptr B,
              const size_t ldb)
```

ftrsm: **TR**iangular **S**ystem solve with **M**atrix.

Computes $B \leftarrow \alpha\mathrm{op}(A^{-1})B$ or $B \leftarrow \alpha B\mathrm{op}(A^{-1})$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha\mathrm{op}(A^{-1})B$ is computed.

Parameters

Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular invertible matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`

Bug α must be non zero.

11.1.3.179 fsyrk_strassen() [2/2]

```
template<class Field, typename FieldTrait>
Field::Element_ptr fsyrk_strassen (
             const Field & F,
             const FFLAS_UPLO UpLo,
             const FFLAS_TRANSPOSE trans,
             const size_t N,
             const size_t K,
             const typename Field::Element y1,
             const typename Field::Element y2,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & H)  [inline]
```

11.1.3.180 pfgemm() [1/7]

```
template<typename Field>
Field::Element_ptr pfgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             size_t numthreads = 0)
```

11.1.3.181 pfgemm_1D_rec()

```
template<class Field>
Field::Element * pfgemm_1D_rec (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::Element_ptr A,
             const size_t lda,
             const typename Field::Element_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element * C,
             const size_t ldc,
             size_t seuil)
```

11.1.3.182 pfgemm_2D_rec()

```
template<class Field>
Field::Element * pfgemm_2D_rec (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::Element_ptr A,
             const size_t lda,
             const typename Field::Element_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element * C,
             const size_t ldc,
             size_t seuil)
```

11.1.3.183 pfgemm_3D_rec()

```
template<class Field>
Field::Element * pfgemm_3D_rec (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::Element_ptr A,
             const size_t lda,
             const typename Field::Element_ptr B,
```

```
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            size_t seuil,
            size_t * x)
```

11.1.3.184 pfgemm_3D_rec2()

```
template<class Field>
Field::Element_ptr pfgemm_3D_rec2 (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            const typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            size_t seuil,
            size_t * x)
```

11.1.3.185 fgemm() [19/23]

```
template<class Field, class ModeTrait, class Strat, class Param>
std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag
> >::value, typenameField::Element_ptr >::type fgemm (
            const Field & F,
            const FFLAS::FFLAS_TRANSPOSE ta,
            const FFLAS::FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,
Param > > & H)  [inline]
```

11.1.3.186 ftrsm() [7/9]

```
template<class Field, class Cut, class Param>
Field::Element_ptr ftrsm (
```

```
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const FFLAS::FFLAS_UPLO UpLo,
            const FFLAS::FFLAS_TRANSPOSE TA,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb,
            TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &
H)  [inline]
```

11.1.3.187 ftrsm() [8/9]

```
template<class Field, class Cut, class Param>
Field::Element_ptr ftrsm (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const FFLAS::FFLAS_UPLO UpLo,
            const FFLAS::FFLAS_TRANSPOSE TA,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb,
            TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > & H)
[inline]
```

11.1.3.188 fspmv() [1/2]

```
template<class Field, class SM>
void fspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            const typename Field::Element & beta,
            typename Field::Element_ptr y)  [inline]
```

11.1.3.189 fspmm()

```
template<class Field, class SM>
void fspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            const typename Field::Element & beta,
            typename Field::Element_ptr y,
            int ldy)  [inline]
```

11.1.3.190 sparse_init() [1/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::COO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.191 sparse_init() [2/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::COO_ZO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.192 sparse_delete() [1/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::COO > & A)  [inline]
```

11.1.3.193 sparse_delete() [2/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::COO_ZO > & A)  [inline]
```

11.1.3.194 sparse_init() [3/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::CSR > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.195 sparse_init() [4/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.196 sparse_delete() [3/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::CSR > & A)  [inline]
```

11.1.3.197 sparse_delete() [4/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A)  [inline]
```

11.1.3.198 sparse_print() [1/3]

```
template<class Field>
std::ostream & sparse_print (
             std::ostream & os,
             const Sparse< Field, SparseMatrix_t::CSR > & A)  [inline]
```

11.1.3.199 sparse_init() [5/16]

```
template<class IndexT>
void sparse_init (
             const Givaro::Modular< Givaro::Integer > & F,
             Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > & A,
             const IndexT * row,
             const IndexT * col,
             Givaro::Integer * dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.200 sparse_init() [6/16]

```
template<class IndexT>
void sparse_init (
            const Givaro::ZRing< Givaro::Integer > & F,
            Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > & A,
            const IndexT * row,
            const IndexT * col,
            Givaro::Integer * dat,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)  [inline]
```

11.1.3.201 sparse_init() [7/16]

```
template<class IndexT, size_t RECINT_SIZE>
void sparse_init (
            const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
            Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO >
& A,
            const IndexT * row,
            const IndexT * col,
            typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)  [inline]
```

11.1.3.202 sparse_init() [8/16]

```
template<class IndexT, size_t RECINT_SIZE>
void sparse_init (
            const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
            Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &
A,
            const IndexT * row,
            const IndexT * col,
            typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)  [inline]
```

11.1.3.203 sparse_delete() [5/12]

```
template<class Field>
void sparse_delete (
            const Sparse< Field, SparseMatrix_t::CSR_HYB > & A)  [inline]
```

11.1.3.204 sparse_init() [9/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.205 sparse_init() [10/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::ELL > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.206 sparse_init() [11/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.207 sparse_delete() [6/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::ELL > & A)  [inline]
```

11.1.3.208 sparse_delete() [7/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A)  [inline]
```

11.1.3.209 sparse_init() [12/16]

```
template<class Field, class IndexT>
void sparse_init (
            const Field & F,
            Sparse< Field, SparseMatrix_t::ELL_simd > & A,
            const IndexT * row,
            const IndexT * col,
            typename Field::ConstElement_ptr dat,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)  [inline]
```

11.1.3.210 sparse_init() [13/16]

```
template<class Field, class IndexT>
void sparse_init (
            const Field & F,
            Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            const IndexT * row,
            const IndexT * col,
            typename Field::ConstElement_ptr dat,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)  [inline]
```

11.1.3.211 sparse_delete() [8/12]

```
template<class Field>
void sparse_delete (
            const Sparse< Field, SparseMatrix_t::ELL_simd > & A)  [inline]
```

11.1.3.212 sparse_delete() [9/12]

```
template<class Field>
void sparse_delete (
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A)  [inline]
```

11.1.3.213 sparse_print() [2/3]

```
template<class Field>
void sparse_print (
            const Sparse< Field, SparseMatrix_t::ELL_simd > & A)  [inline]
```

11.1.3.214 sparse_delete() [10/12]

```
template<class Field>
void sparse_delete (
            const Sparse< Field, SparseMatrix_t::HYB_ZO > & A)  [inline]
```

11.1.3.215 sparse_init() [14/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.216 operator<<()

```
template<typename _Field>
std::ostream & operator<< (
             std::ostream & os,
             const Sparse< _Field, SparseMatrix_t::HYB_ZO > & A)
```

11.1.3.217 readSmsFormat()

```
template<class Field, bool sorted = true, bool read_integer = false>
void readSmsFormat (
             const std::string & path,
             const Field & f,
             index_t *& row,
             index_t *& col,
             typename Field::Element_ptr & val,
             index_t & rowdim,
             index_t & coldim,
             uint64_t & nnz)
```

11.1.3.218 readSprFormat()

```
template<class Field>
void readSprFormat (
             const std::string & path,
             const Field & f,
             index_t *& row,
             index_t *& col,
             typename Field::Element_ptr & val,
             index_t & rowdim,
             index_t & coldim,
             uint64_t & nnz)
```

11.1.3.219 getDataType() [1/4]

```
template<class T>
std::enable_if< std::is_integral< T >::value, int > getDataType ()
```

11.1.3.220 getDataType() [2/4]

```
template<class T>
std::enable_if< std::is_floating_point< T >::value, int > getDataType ()
```

11.1.3.221 getDataType() [3/4]

```
template<class T>
std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()
```

11.1.3.222 getDataType() [4/4]

```
template<class T>
int getDataType ()
```

11.1.3.223 readMachineType()

```
template<class Field>
void readMachineType (
            const Field & F,
            typename Field::Element & modulo,
            typename Field::Element_ptr val,
            std::ifstream & file,
            const uint64_t dims,
            const mask_t data_type,
            const mask_t field_desc)
```

11.1.3.224 readDnsFormat()

```
template<class Field>
void readDnsFormat (
            const std::string & path,
            const Field & F,
            index_t & rowdim,
            index_t & coldim,
            typename Field::Element_ptr & val)
```

11.1.3.225 writeDnsFormat()

```
template<class Field>
void writeDnsFormat (
            const std::string & path,
            const Field & F,
            const index_t & rowdim,
            const index_t & coldim,
            typename Field::Element_ptr A,
            index_t ldA)
```

11.1.3.226 fspmv() [2/2]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag )  [inline]
```

11.1.3.227 sparse_delete() [11/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::SELL > & A)  [inline]
```

11.1.3.228 sparse_delete() [12/12]

```
template<class Field>
void sparse_delete (
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A)  [inline]
```

11.1.3.229 sparse_print() [3/3]

```
template<class Field>
void sparse_print (
             const Sparse< Field, SparseMatrix_t::SELL > & A)  [inline]
```

11.1.3.230 sparse_init() [15/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::SELL > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz,
             uint64_t sigma = 0)  [inline]
```

11.1.3.231 sparse_init() [16/16]

```
template<class Field, class IndexT>
void sparse_init (
             const Field & F,
             Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             const IndexT * row,
             const IndexT * col,
             typename Field::ConstElement_ptr dat,
             uint64_t rowdim,
             uint64_t coldim,
             uint64_t nnz)  [inline]
```

11.1.3.232 computeDeviation()

```
template<class It>
double computeDeviation (
            It begin,
            It end)
```

11.1.3.233 getStat()

```
template<class Field>
StatsMatrix getStat (
            const Field & F,
            const index_t * row,
            const index_t * col,
            typename Field::ConstElement_ptr val,
            uint64_t rowdim,
            uint64_t coldim,
            uint64_t nnz)
```

11.1.3.234 ftranspose()

```
template<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_↩
TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value
>::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type *
= nullptr>
Field::Element_ptr ftranspose (
            const Field & F,
            const size_t m,
            const size_t n,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb)  [inline]
```

11.1.3.235 maxCardinality()

```
template<class Field, class enable = void>
Field::Residu_t maxCardinality ()  [inline]
```

11.1.3.236 maxCardinality< Givaro::Modular< int64_t > >()

```
template<>
uint64_t maxCardinality< Givaro::Modular< int64_t > > ()  [inline]
```

11.1.3.237 maxCardinality< Givaro::Modular< int32_t > >()

```
template<>
uint32_t maxCardinality< Givaro::Modular< int32_t > > ()  [inline]
```

11.1.3.238 minCardinality()

```
template<class Field>
Field::Residu_t minCardinality ()  [inline]
```

11.1.3.239 fflas_delete() [1/4]

```
template<>
void fflas_delete (
            FFPACK::rns_double_elt_ptr A)  [inline]
```

11.1.3.240 fflas_delete() [2/4]

```
template<>
void fflas_delete (
            FFPACK::rns_double_elt_cstptr A)  [inline]
```

11.1.3.241 fflas_new() [1/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
            const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
            const size_t m,
            const Alignment align)  [inline]
```

11.1.3.242 fflas_new() [2/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
            const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
            const size_t m,
            const size_t n,
            const Alignment align)  [inline]
```

11.1.3.243 finit_rns() [1/2]

```
template<typename RNS>
void finit_rns (
            const FFPACK::RNSIntegerMod< RNS > & F,
            const size_t m,
            const size_t n,
            size_t k,
            const Givaro::Integer * B,
            const size_t ldb,
            typename RNS::Element_ptr A)
```

11.1.3.244 finit_trans_rns()

```
template<typename RNS>
void finit_trans_rns (
             const FFPACK::RNSIntegerMod< RNS > & F,
             const size_t m,
             const size_t n,
             size_t k,
             const Givaro::Integer * B,
             const size_t ldb,
             typename RNS::Element_ptr A)
```

11.1.3.245 fconvert_rns() [1/2]

```
template<typename RNS>
void fconvert_rns (
             const FFPACK::RNSIntegerMod< RNS > & F,
             const size_t m,
             const size_t n,
             Givaro::Integer alpha,
             Givaro::Integer * B,
             const size_t ldb,
             typename RNS::ConstElement_ptr A)
```

11.1.3.246 fconvert_trans_rns()

```
template<typename RNS>
void fconvert_trans_rns (
             const FFPACK::RNSIntegerMod< RNS > & F,
             const size_t m,
             const size_t n,
             Givaro::Integer alpha,
             Givaro::Integer * B,
             const size_t ldb,
             typename RNS::ConstElement_ptr A)
```

11.1.3.247 fflas_new() [3/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
             const FFPACK::RNSInteger< FFPACK::rns_double > & F,
             const size_t m,
             const Alignment align)  [inline]
```

11.1.3.248 fflas_new() [4/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
             const FFPACK::RNSInteger< FFPACK::rns_double > & F,
             const size_t m,
             const size_t n,
             const Alignment align)  [inline]
```

11.1.3.249 finit_rns() [2/2]

```
template<typename RNS>
void finit_rns (
             const FFPACK::RNSInteger< RNS > & F,
             const size_t m,
             const size_t n,
             size_t k,
             const Givaro::Integer * B,
             const size_t ldb,
             typename FFPACK::RNSInteger< RNS >::Element_ptr A)
```

11.1.3.250 fconvert_rns() [2/2]

```
template<typename RNS>
void fconvert_rns (
             const FFPACK::RNSInteger< RNS > & F,
             const size_t m,
             const size_t n,
             Givaro::Integer alpha,
             Givaro::Integer * B,
             const size_t ldb,
             typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)
```

11.1.3.251 freduce() [8/11]

```
template INST_OR_DECL void freduce (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             FFLAS_ELT * X,
             const size_t incX)
```

freduce $x \leftarrow xmodF$.

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.252 freduce() [9/11]

```
template INST_OR_DECL void freduce (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             const FFLAS_ELT * Y,
             const size_t incY,
             FFLAS_ELT * X,
             const size_t incX)
```

freduce $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors
Y	vector of `Element`
incY	stride of `Y`
X	vector in `F`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.253 finit() [7/8]

```
template INST_OR_DECL void finit (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t n,
              const FFLAS_ELT * Y,
              const size_t incY,
              FFLAS_ELT * X,
              const size_t incX)
```

finit $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors
Y	vector of `OtherElement`
incY	stride of `Y`
X	vector in `F`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.254 fconvert() [3/3]

```
template INST_OR_DECL void fconvert (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t n,
              FFLAS_ELT * X,
              const size_t incX,
              const FFLAS_ELT * Y,
              const size_t incY)
```

fconvert $x \leftarrow ymodF$.

Parameters

F	field
n	size of the vectors
Y	vector of `F`
incY	stride of `Y`
X	vector in `OtherElement`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.255 fnegin() [3/4]

```
template INST_OR_DECL void fnegin (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t n,
              FFLAS_ELT * X,
              const size_t incX)
```

fnegin $x \leftarrow -x$.

Parameters

F	field
n	size of the vectors
X	vector in `F`
incX	stride of `X`

Bug use cblas_(d)scal when possible

11.1.3.256 fneg() [3/4]

```
template INST_OR_DECL void fneg (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t n,
              const FFLAS_ELT * Y,
              const size_t incY,
              FFLAS_ELT * X,
              const size_t incX)
```

fneg $x \leftarrow -y$.

Parameters

F	field
n	size of the vectors
X	vector in `F`
incX	stride of `X`
Y	vector in `F`
incY	stride of `Y`

Bug use cblas_(d)scal when possible

11.1.3.257 fzero() [4/5]

```
template INST_OR_DECL void fzero (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             FFLAS_ELT * X,
             const size_t incX)
```

fzero : $A \leftarrow 0$.

Parameters

F	field
n	number of elements to zero
X	vector in F
incX	stride of X

11.1.3.258 fiszero() [3/4]

```
template INST_OR_DECL bool fiszero (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             const FFLAS_ELT * X,
             const size_t incX)
```

fiszero : test $X = 0$.

Parameters

F	field
n	vector dimension
X	vector in F
incX	increment of X

11.1.3.259 fequal() [3/4]

```
template INST_OR_DECL bool fequal (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             const FFLAS_ELT * X,
             const size_t incX,
             const FFLAS_ELT * Y,
             const size_t incY)
```

fequal : test $X = Y$.

Parameters

F	field
n	vector dimension
X	vector in F
incX	increment of X
Y	vector in F
incY	increment of Y

11.1.3.260 fassign() [9/10]

```
template INST_OR_DECL void fassign (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t N,
            const FFLAS_ELT * Y,
            const size_t incY,
            FFLAS_ELT * X,
            const size_t incX)
```

fassign : $x \leftarrow y$.

X is preallocated

Todo variant for triagular matrix

Parameters

	F	field
	N	size of the vectors
out	*X*	vector in F
	incX	stride of X
in	*Y*	vector in F
	incY	stride of Y

11.1.3.261 fscalin() [9/10]

```
template INST_OR_DECL void fscalin (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t n,
            const FFLAS_ELT alpha,
            FFLAS_ELT * X,
            const size_t incX)
```

fscalin $x \leftarrow \alpha \cdot x$.

Parameters

F	field
n	size of the vectors
alpha	scalar
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.262 fscal() [9/10]

```
template INST_OR_DECL void fscal (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t n,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * X,
             const size_t incX,
             FFLAS_ELT * Y,
             const size_t incY)
```

fscal $y \leftarrow \alpha \cdot x$.

Parameters

	F	field
	n	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
out	*Y*	vector in F
	incY	stride of Y

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.263 faxpy() [5/6]

```
template INST_OR_DECL void faxpy (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t N,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * X,
             const size_t incX,
             FFLAS_ELT * Y,
             const size_t incY)
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field
	N	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
in,out	*Y*	vector in F
	incY	stride of Y

11.1.3.264 fdot() [10/11]

```
template INST_OR_DECL FFLAS_ELT fdot (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t N,
            const FFLAS_ELT * X,
            const size_t incX,
            const FFLAS_ELT * Y,
            const size_t incY)
```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	N	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
	beta	scalar
in,out	*Y*	vector in F
	incY	stride of Y

Note

this is a catlas function

fdot: dot product $x^T y$.

Parameters

F	field
N	size of the vectors
X	vector in F
incX	stride of X
Y	vector in F
incY	stride of Y

11.1.3.265 fswap() [2/2]

```
template INST_OR_DECL void fswap (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t N,
            FFLAS_ELT * X,
            const size_t incX,
            FFLAS_ELT * Y,
            const size_t incY)
```

fswap: $X \leftrightarrow Y$.

Bug use cblas_dswap when double

Parameters

F	field
N	size of the vectors
X	vector in F
incX	stride of X
Y	vector in F
incY	stride of Y

11.1.3.266 fadd() [5/8]

```
template INST_OR_DECL void fadd (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t N,
             const FFLAS_ELT * A,
             const size_t inca,
             const FFLAS_ELT * B,
             const size_t incb,
             FFLAS_ELT * C,
             const size_t incc)
```

11.1.3.267 fsub() [3/4]

```
template INST_OR_DECL void fsub (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t N,
             const FFLAS_ELT * A,
             const size_t inca,
             const FFLAS_ELT * B,
             const size_t incb,
             FFLAS_ELT * C,
             const size_t incc)
```

11.1.3.268 faddin() [4/5]

```
template INST_OR_DECL void faddin (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t N,
             const FFLAS_ELT * B,
             const size_t incb,
             FFLAS_ELT * C,
             const size_t incc)
```

11.1.3.269 fadd() [6/8]

```
template INST_OR_DECL void fadd (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t N,
             const FFLAS_ELT * A,
             const size_t inca,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * B,
             const size_t incb,
             FFLAS_ELT * C,
             const size_t incc)
```

11.1.3.270 fassign() [10/10]

```
template INST_OR_DECL void fassign (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t m,
              const size_t n,
              const FFLAS_ELT * B,
              const size_t ldb,
              FFLAS_ELT * A,
              const size_t lda)
```

fassign : $A \leftarrow B$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	vector in F
ldb	stride of B

11.1.3.271 fzero() [5/5]

```
template INST_OR_DECL void fzero (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t m,
              const size_t n,
              FFLAS_ELT * A,
              const size_t lda)
```

fzero : $A \leftarrow 0$.

Parameters

F	field
m	number of rows to zero
n	number of cols to zero
A	matrix in F
lda	stride of A

Warning

may be buggy if Element is larger than int

11.1.3.272 fequal() [4/4]

```
template INST_OR_DECL bool fequal (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             const FFLAS_ELT * A,
             const size_t lda,
             const FFLAS_ELT * B,
             const size_t ldb)
```

fequal : test $A = B$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in F
lda	leading dimension of A
B	m x n matrix in F
ldb	leading dimension of B

11.1.3.273 fiszero() [4/4]

```
template INST_OR_DECL bool fiszero (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             const FFLAS_ELT * A,
             const size_t lda)
```

fiszero : test $A = 0$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in F
lda	leading dimension of A

11.1.3.274 fidentity() [3/4]

```
template INST_OR_DECL void fidentity (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             FFLAS_ELT * A,
             const size_t lda,
             const FFLAS_ELT & d)
```

creates a diagonal matrix

11.1.3.275 fidentity() [4/4]

```
template INST_OR_DECL void fidentity (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             FFLAS_ELT * A,
             const size_t lda)
```

creates a diagonal matrix

11.1.3.276 freduce() [10/11]

```
template INST_OR_DECL void freduce (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             FFLAS_ELT * A,
             const size_t lda)
```

freduce $A \leftarrow AmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`

11.1.3.277 freduce() [11/11]

```
template INST_OR_DECL void freduce (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             const FFLAS_ELT * B,
             const size_t ldb,
             FFLAS_ELT * A,
             const size_t lda)
```

freduce $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`
B	matrix in `Element`
ldb	stride of `B`

11.1.3.278 finit() [8/8]

```
template INST_OR_DECL void finit (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t m,
            const size_t n,
            const FFLAS_ELT * B,
            const size_t ldb,
            FFLAS_ELT * A,
            const size_t lda)
```

finit $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in F
ldb	stride of B

11.1.3.279 fnegin() [4/4]

```
template INST_OR_DECL void fnegin (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t m,
            const size_t n,
            FFLAS_ELT * A,
            const size_t lda)
```

fnegin $A \leftarrow -A$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.280 fneg() [4/4]

```
template INST_OR_DECL void fneg (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t m,
            const size_t n,
            const FFLAS_ELT * B,
            const size_t ldb,
            FFLAS_ELT * A,
            const size_t lda)
```

fneg $A \leftarrow -B$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.281 fscalin() [10/10]

```
template INST_OR_DECL void fscalin (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t m,
              const size_t n,
              const FFLAS_ELT alpha,
              FFLAS_ELT * A,
              const size_t lda)
```

fscalin $A \leftarrow a \cdot A$.

Parameters

F	field
m	number of rows
n	number of cols
alpha	homotecie scalar
A	matrix in F
lda	stride of A

11.1.3.282 fscal() [10/10]

```
template INST_OR_DECL void fscal (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t m,
              const size_t n,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * A,
              const size_t lda,
              FFLAS_ELT * B,
              const size_t ldb)
```

fscal $B \leftarrow a \cdot A$.

Parameters

	F	field
	m	number of rows
	n	number of cols
	alpha	homotecie scalar
in	*A*	matrix in F
	lda	stride of A
out	*B*	matrix in F
	ldb	stride of B

11.1.3.283 faxpy() [6/6]

```
template INST_OR_DECL void faxpy (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * X,
             const size_t ldx,
             FFLAS_ELT * Y,
             const size_t ldy)
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field
	m	row dimension
	n	column dimension
	alpha	scalar
in	*X*	vector in F
	ldx	leading dimension of X
in,out	*Y*	vector in F
	ldy	leading dimension of Y

11.1.3.284 fmove() [2/2]

```
template INST_OR_DECL void fmove (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t m,
             const size_t n,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT * B,
             const size_t ldb)
```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	m	row dimension
	n	column dimension
	alpha	scalar
in	*X*	vector in F
	ldx	leading dimension of X
	beta	scalar
in,out	*Y*	vector in F
	ldy	leading dimension of Y

Note

this is a catlas function

fmove : $A \leftarrow B$ and $B \leftarrow 0$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	vector in F
ldb	stride of B

11.1.3.285 fadd() [7/8]

```
template INST_OR_DECL void fadd (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             const FFLAS_ELT * A,
             const size_t lda,
             const FFLAS_ELT * B,
             const size_t ldb,
             FFLAS_ELT * C,
             const size_t ldc)
```

fadd : matrix addition.

Computes C = A + B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.286 fsub() [4/4]

```
template INST_OR_DECL void fsub (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             const FFLAS_ELT * A,
             const size_t lda,
             const FFLAS_ELT * B,
             const size_t ldb,
             FFLAS_ELT * C,
             const size_t ldc)
```

fsub : matrix subtraction.

Computes C = A - B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.287 fsubin() [3/3]

```
template INST_OR_DECL void fsubin (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t M,
              const size_t N,
              const FFLAS_ELT * B,
              const size_t ldb,
              FFLAS_ELT * C,
              const size_t ldc)
```

fsubin C = C - B

11.1.3.288 fadd() [8/8]

```
template INST_OR_DECL void fadd (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t M,
              const size_t N,
              const FFLAS_ELT * A,
              const size_t lda,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * B,
              const size_t ldb,
              FFLAS_ELT * C,
              const size_t ldc)
```

fadd : matrix addition with scaling.

Computes C = A + alpha B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
alpha	some scalar
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.289 faddin() [5/5]

```
template INST_OR_DECL void faddin (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t M,
              const size_t N,
              const FFLAS_ELT * B,
              const size_t ldb,
              FFLAS_ELT * C,
              const size_t ldc)
```

faddin

11.1.3.290 fgemv() [17/19]

```
template INST_OR_DECL FFLAS_ELT * fgemv (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const FFLAS_TRANSPOSE TransA,
              const size_t M,
              const size_t N,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * A,
              const size_t lda,
              const FFLAS_ELT * X,
              const size_t incX,
              const FFLAS_ELT beta,
              FFLAS_ELT * Y,
              const size_t incY)
```

finite prime FFLAS_FIELD<FFLAS_ELT> GEneral Matrix Vector multiplication.

Computes $Y \leftarrow \alpha \mathrm{op}(A) X + \beta Y$.

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	lda	leading dimension of `A`
	X	dense vector of size `N`
	incX	stride of `X`
	beta	scalar
`out`	*Y*	dense vector of size `M`
	incY	stride of `Y`

11.1.3.291 fger() [12/12]

```
template INST_OR_DECL void fger (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const size_t M,
              const size_t N,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * x,
              const size_t incx,
              const FFLAS_ELT * y,
              const size_t incy,
              FFLAS_ELT * A,
              const size_t lda)
```

fger: rank one update of a general matrix

Computes $A \leftarrow \alpha x.y^T + A$

Parameters

	F	field
	M	rows
	N	cols
	alpha	scalar
in,out	*A*	dense matrix of size MxN and leading dimension lda
	lda	leading dimension of A
	x	dense vector of size M
	incx	stride of X
	y	dense vector of size N
	incy	stride of Y

11.1.3.292 ftrsv() [2/2]

```
template INST_OR_DECL void ftrsv (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t N,
              const FFLAS_ELT * A,
              const size_t lda,
              FFLAS_ELT * X,
              int incX)
```

ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

Parameters

F	field
X	vector of size N on a field F
incX	stride of X
A	a matrix of leading dimension lda and size N
lda	leading dimension of A

Parameters

N	number of rows or columns of `A` according to `TransA`
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

11.1.3.293 ftrsm() [9/9]

```
template INST_OR_DECL void ftrsm (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const FFLAS_SIDE Side,
              const FFLAS_UPLO Uplo,
              const FFLAS_TRANSPOSE TransA,
              const FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * A,
              const size_t lda,
              FFLAS_ELT * B,
              const size_t ldb)
```

ftrsm: **TR**iangular **S**ystem solve with **M**atrix.

Computes $B \leftarrow \alpha \mathrm{op}(A^{-1})B$ or $B \leftarrow \alpha B \mathrm{op}(A^{-1})$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A^{-1})B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular invertible matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`

Bug α must be non zero.

11.1.3.294 ftrmm() [3/3]

```
template INST_OR_DECL void ftrmm (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS_SIDE Side,
             const FFLAS_UPLO Uplo,
             const FFLAS_TRANSPOSE TransA,
             const FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT * B,
             const size_t ldb)
```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes $B \leftarrow \alpha \mathrm{op}(A) B$ or $B \leftarrow \alpha B \mathrm{op}(A)$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`

11.1.3.295 fgemm() [20/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const FFLAS_ELT alpha,
             const FFLAS_ELT * A,
             const size_t lda,
             const FFLAS_ELT * B,
             const size_t ldb,
             const FFLAS_ELT beta,
             FFLAS_ELT * C,
             const size_t ldc)
```

fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.

Computes $C = \alpha \mathrm{op}(A) \times \mathrm{op}(B) + \beta C$ Automatically set Winograd recursion level

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ is $k \times n$
C	C is $m \times n$
lda	leading dimension of `A`
ldb	leading dimension of `B`
ldc	leading dimension of `C`
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of `w`.

Warning

α *must* be invertible

11.1.3.296 fgemm() [21/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const FFLAS_TRANSPOSE ta,
              const FFLAS_TRANSPOSE tb,
              const size_t m,
              const size_t n,
              const size_t k,
              const FFLAS_ELT alpha,
              const FFLAS_ELT * A,
              const size_t lda,
              const FFLAS_ELT * B,
              const size_t ldb,
              const FFLAS_ELT beta,
              FFLAS_ELT * C,
              const size_t ldc,
              const ParSeqHelper::Sequential seq)
```

11.1.3.297 fgemm() [22/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
              const FFLAS_FIELD< FFLAS_ELT > & F,
              const FFLAS_TRANSPOSE ta,
              const FFLAS_TRANSPOSE tb,
              const size_t m,
              const size_t n,
              const size_t k,
```

```
            const FFLAS_ELT alpha,
            const FFLAS_ELT * A,
            const size_t lda,
            const FFLAS_ELT * B,
            const size_t ldb,
            const FFLAS_ELT beta,
            FFLAS_ELT * C,
            const size_t ldc,
            const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive
> par)
```

11.1.3.298 fgemm() [23/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const FFLAS_ELT alpha,
            const FFLAS_ELT * A,
            const size_t lda,
            const FFLAS_ELT * B,
            const size_t ldb,
            const FFLAS_ELT beta,
            FFLAS_ELT * C,
            const size_t ldc,
            const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads
> par)
```

11.1.3.299 fsquare() [6/6]

```
template INST_OR_DECL FFLAS_ELT * fsquare (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS_TRANSPOSE ta,
            const size_t n,
            const FFLAS_ELT alpha,
            const FFLAS_ELT * A,
            const size_t lda,
            const FFLAS_ELT beta,
            FFLAS_ELT * C,
            const size_t ldc)
```

fsquare: Squares a matrix.

compute $C \leftarrow \alpha \mathrm{op}(A)\mathrm{op}(A) + \beta C$ over a FFLAS_FIELD <FFLAS_ELT> F Avoid the conversion of B

Parameters

ta	if ta==FflasTrans, $\mathrm{op}(A) = A^T$.
F	field
n	size of A
alpha	scalar

Parameters

beta	scalar
A	dense matrix of size `nxn`
lda	leading dimension of `A`
C	dense matrix of size `nxn`
ldc	leading dimension of `C`

11.1.3.300 BlockCuts() [1/2]

```
template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>
void BlockCuts (
             size_t & RBLOCKSIZE,
             size_t & CBLOCKSIZE,
             const size_t m,
             const size_t n,
             const size_t numthreads)  [inline]
```

11.1.3.301 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (
             size_t & RBLOCKSIZE,
             size_t & CBLOCKSIZE,
             const size_t m,
             const size_t n,
             const size_t numthreads)  [inline]
```

11.1.3.302 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (
             size_t & RBLOCKSIZE,
             size_t & CBLOCKSIZE,
             const size_t m,
             const size_t n,
             const size_t numthreads)  [inline]
```

11.1.3.303 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (
             size_t & RBLOCKSIZE,
             size_t & CBLOCKSIZE,
             const size_t m,
             const size_t n,
             const size_t grainsize)  [inline]
```

11.1.3.304 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t grainsize)  [inline]
```

11.1.3.305 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.306 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t grainsize)  [inline]
```

11.1.3.307 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.308 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.309 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.310 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (
            size_t & RBLOCKSIZE,
            size_t & CBLOCKSIZE,
            const size_t m,
            const size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.311 BlockCuts() [2/2]

```
template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>
void BlockCuts (
            size_t & rowBlockSize,
            size_t & colBlockSize,
            size_t & lastRBS,
            size_t & lastCBS,
            size_t & changeRBS,
            size_t & changeCBS,
            size_t & numRowBlock,
            size_t & numColBlock,
            size_t m,
            size_t n,
            const size_t numthreads)  [inline]
```

11.1.3.312 pfzero()

```
template<class Field>
void pfzero (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr C,
            size_t BS = 0)
```

11.1.3.313 pfrand()

```
template<class Field, class RandIter>
void pfrand (
            const Field & F,
            RandIter & G,
            size_t m,
            size_t n,
            typename Field::Element_ptr C,
            size_t BS = 0)
```

11.1.3.314 fdot() [11/11]

```
template<class Field, class Cut, class Param>
Field::Element & fdot (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            typename Field::Element & d,
            const ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.1.3.315 pfgemm() [2/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element * pfgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element * C,
            const size_t ldc,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block,
StrategyParameter::Threads > > & H)
```

11.1.3.316 pfgemm() [3/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element * pfgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr AA,
            const size_t lda,
            const typename Field::ConstElement_ptr BB,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element * C,
            const size_t ldc,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDAdaptive > > & H)
```

11.1.3.317 pfgemm() [4/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element * pfgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::ConstElement_ptr AA,
             const size_t lda,
             const typename Field::ConstElement_ptr BB,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element * C,
             const size_t ldc,
             MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoDAdaptive > > & H)
```

11.1.3.318 pfgemm() [5/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element * pfgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::ConstElement_ptr AA,
             const size_t lda,
             const typename Field::ConstElement_ptr BB,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element * C,
             const size_t ldc,
             MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoD > > & H)
```

11.1.3.319 pfgemm() [6/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element_ptr pfgemm (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::ConstElement_ptr A,
```

```
            const size_t lda,
            const typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeD > > & H)
```

11.1.3.320 pfgemm() [7/7]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element * pfgemm (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDInPlace > > & H)
```

11.1.3.321 fgemv() [18/19]

```
template<class Field, class AlgoT, class FieldTrait>
Field::Element_ptr fgemv (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::ConstElement_ptr X,
            const size_t incX,
            const typename Field::Element beta,
            typename Field::Element_ptr Y,
            const size_t incY,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::Threads > > & H)
```

11.1.3.322 fgemv() [19/19]

```
template<class Field, class AlgoT, class FieldTrait, class Cut>
Field::Element_ptr fgemv (
```

```
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::ConstElement_ptr X,
            const size_t incX,
            const typename Field::Element beta,
            typename Field::Element_ptr Y,
            const size_t incY,
            MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row,
Cut > > & H)
```

11.1.3.323 parseArguments()

```
void parseArguments (
            int argc,
            char ** argv,
            Argument * args,
            bool printDefaults = true)
```

11.1.3.324 getArgumentValue()

```
char * getArgumentValue (
            int argc,
            char ** argv,
            int i)
```

Get the value of an argument and avoid core dump when no value was given after an argument.

Parameters

argv	argument value list
i	argument index

Returns

char* argument value

11.1.3.325 writeCommandString()

```
std::ostream & writeCommandString (
            std::ostream & os,
            Argument * args,
            const char * programName = nullptr)
```

writes the values of all arguments, preceded by the programName

11.1.3.326 WriteMatrix() [1/2]

```
template<class Field>
std::ostream & WriteMatrix (
             std::ostream & c,
             const Field & F,
             size_t m,
             size_t n,
             typename Field::ConstElement_ptr A,
             size_t lda,
             FFLAS_FORMAT format,
             bool column_major)  [inline]
```

WriteMatrix: write a matrix to an output stream.

Parameters

c	output stream
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

11.1.3.327 preamble()

```
void preamble (
             std::ifstream & ifs,
             FFLAS_FORMAT & format)  [inline]
```

11.1.3.328 ReadMatrix() [1/2]

```
template<class Field>
Field::Element_ptr ReadMatrix (
             std::ifstream & ifs,
             Field & F,
             size_t & m,
             size_t & n,
             typename Field::Element_ptr & A,
             FFLAS_FORMAT format = FflasAuto)
```

ReadMatrix: read a matrix from an input stream.

Parameters

	ifs	input stream
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.329 ReadMatrix() [2/2]

```
template<class Field>
Field::Element_ptr ReadMatrix (
            const std::string & matrix_file,
            Field & F,
            size_t & m,
            size_t & n,
            typename Field::Element_ptr & A,
            FFLAS_FORMAT format = FflasAuto)  [inline]
```

ReadMatrix: read a matrix from a file.

Parameters

	matrix_file	filename
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.330 WriteMatrix() [2/2]

```
template<class Field>
void WriteMatrix (
            std::string & matrix_file,
            const Field & F,
            int m,
            int n,
            typename Field::ConstElement_ptr A,
            size_t lda,
            FFLAS_FORMAT format = FflasDense,
            bool column_major = false)
```

WriteMatrix: write a matrix to a file.

Parameters

matrix_file	file name
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

11.1.3.331 WritePermutation()

```
std::ostream & WritePermutation (
            std::ostream & c,
            const size_t * P,
            size_t N)  [inline]
```

WritePermutation: write a permutation matrix to an output stream.

Parameters

c	output stream
P	permutation
N	size of the permutation

11.1.3.332 alignable()

```
template<class Element>
bool alignable ()  [inline]
```

11.1.3.333 alignable< Givaro::Integer * >()

```
template<>
bool alignable< Givaro::Integer * > ()  [inline]
```

11.1.3.334 fflas_new() [5/7]

```
template<class Field>
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.335 fflas_new() [6/7]

```
template<class Field>
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const size_t n,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.336 fflas_new() [7/7]

```
template<class Element>
Element * fflas_new (
            const size_t m,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.337 fflas_delete() [3/4]

```
template<class Element_ptr>
void fflas_delete (
            Element_ptr A)  [inline]
```

11.1.3.338 fflas_delete() [4/4]

```
template<class Ptr, class ...  Args>
void fflas_delete (
            Ptr p,
            Args ...  args)  [inline]
```

11.1.3.339 prefetch()

```
void prefetch (
            const int64_t * )  [inline]
```

11.1.3.340 getTLBSize()

```
void getTLBSize (
            int & tlb)  [inline]
```

11.1.3.341 queryCacheSizes()

```
void queryCacheSizes (
            int & l1,
            int & l2,
            int & l3)  [inline]
```

Queries and returns the cache sizes in Bytes of the L1, L2, and L3 data caches respectively

11.1.3.342 queryL1CacheSize()

```
int queryL1CacheSize ()  [inline]
```

Returns

the size in Bytes of the L1 data cache

11.1.3.343 queryTopLevelCacheSize()

```
int queryTopLevelCacheSize ()  [inline]
```

Returns

the size in Bytes of the L2 or L3 cache if this later is present

11.1.3.344 getSeed()

```
uint64_t getSeed ()
```

11.2 FFLAS::_ftranspose_impl Namespace Reference

Functions

- template<size_t bs, typename Field, typename BTSimd>
  void not_inplace (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<size_t bs, typename Field, typename BTSimd>
  void square_inplace (const Field &F, const BTSimd &BTS, const size_t m, typename Field::Element_ptr A, const size_t lda)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v1 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v2 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)

11.2.1 Function Documentation

11.2.1.1 not_inplace()

```
template<size_t bs, typename Field, typename BTSimd>
void not_inplace (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb)
```

11.2.1.2 square_inplace()

```
template<size_t bs, typename Field, typename BTSimd>
void square_inplace (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            typename Field::Element_ptr A,
            const size_t lda)
```

11.2.1.3 nonsquare_inplace_v1()

```
template<size_t bs, typename Field, typename BTSimd>
void nonsquare_inplace_v1 (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::Element_ptr A)
```

11.2.1.4 nonsquare_inplace_v2()

```
template<size_t bs, typename Field, typename BTSimd>
void nonsquare_inplace_v2 (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::Element_ptr A)
```

11.3 FFLAS::BLAS3 Namespace Reference

Functions

- template<class Field>
  void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE base, const size_t rec_level)
- template<class Field, class FieldTrait, class Strat, class Param>
  Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)
- template<class Field, class FieldTrait>
  void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Field↩ Trait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

- template<class Field, class FieldTrait>
  void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

11.3.1 Function Documentation

11.3.1.1 Bini()

```
template<class Field>
void Bini (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            const typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
```

```
            typename Field::Element_ptr C,
            const size_t ldc,
            const size_t kmax,
            const size_t w,
            const FFLAS_BASE base,
            const size_t rec_level)  [inline]
```

11.3.1.2 WinoPar()

```
template<class Field, class FieldTrait, class Strat, class Param>
Field::Element_ptr WinoPar (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel<
Strat, Param > > & WH)  [inline]
```

11.3.1.3 Winograd()

```
template<class Field, class FieldTrait>
void Winograd (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.4 WinogradAcc_3_23()

```
template<class Field, class FieldTrait>
void WinogradAcc_3_23 (
            const Field & F,
```

```
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.5 WinogradAcc_3_21()

```
template<class Field, class FieldTrait>
void WinogradAcc_3_21 (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.6 WinogradAcc_2_24()

```
template<class Field, class FieldTrait>
void WinogradAcc_2_24 (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            const typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.7 WinogradAcc_2_27()

```
template<class Field, class FieldTrait>
void WinogradAcc_2_27 (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t mr,
             const size_t nr,
             const size_t kr,
             const typename Field::Element alpha,
             const typename Field::Element_ptr A,
             const size_t lda,
             const typename Field::Element_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.8 WinogradAcc_LR()

```
template<class Field, class FieldTrait>
void WinogradAcc_LR (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t mr,
             const size_t nr,
             const size_t kr,
             const typename Field::Element alpha,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
             const size_t ldb,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc,
             const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.9 WinogradAcc_R_S()

```
template<class Field, class FieldTrait>
void WinogradAcc_R_S (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const FFLAS_TRANSPOSE tb,
             const size_t mr,
             const size_t nr,
             const size_t kr,
             const typename Field::Element alpha,
             const typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
```

```
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.10 WinogradAcc_L_S()

```
template<class Field, class FieldTrait>
void WinogradAcc_L_S (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.11 Winograd_LR_S()

```
template<class Field, class FieldTrait>
void Winograd_LR_S (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.12 Winograd_L_S()

```
template<class Field, class FieldTrait>
void Winograd_L_S (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
```

```
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.3.1.13 Winograd_R_S()

```
template<class Field, class FieldTrait>
void Winograd_R_S (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            const typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.4 FFLAS::csr_hyb_details Namespace Reference

Data Structures

- struct Coo
- struct Info

11.5 FFLAS::CuttingStrategy Namespace Reference

Data Structures

- struct Block
- struct Column
- struct Recursive
- struct Row
- struct Single

Typedefs

- typedef Row RNSModulus

11.5.1 Typedef Documentation

11.5.1.1 RNSModulus

```
typedef Row RNSModulus
```

11.6 FFLAS::details Namespace Reference

Functions

- template<class Field, bool ADD>
  std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)
- template<class Field, bool ADD>
  std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)
- template<class Field, bool ADD>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::GenericTag)
- template<class Field, bool ADD>
  std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)
- template<class Field, bool ADD>
  std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size↩ _t incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)
- template<class Field, class FC>
  void faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)

- template<class Field, class FC>
  void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)
- template<class Field, class FC>
  void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename Field::Element_ptr A, const size_t incX, FC)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin (const Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX, FieldCategories::ModularTag)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)
- template<class Field, class FC>
  void fscalin (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX, FC)
- template<class Field, class FC>
  void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)
- template<enum number_kind K>
  void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)
- template<size_t k, bool transpose>
  void pack_lhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- template<size_t k, bool transpose>
  void pack_rhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- void gebp (size_t rows, size_t cols, size_t depth, int64_t *C, size_t ldc, const int64_t *blockA, size_t lda, const int64_t *BlockB, size_t ldb, int64_t *BlockW)
- void BlockingFactor (size_t &m, size_t &n, size_t &k)

11.6.1 Function Documentation

11.6.1.1 fadd() [1/5]

```
template<class Field, bool ADD>
std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
            const Field & F,
```

```
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t inca,
            typename Field::ConstElement_ptr B,
            const size_t incb,
            typename Field::Element_ptr C,
            const size_t incc,
            FieldCategories::ModularTag )
```

11.6.1.2 fadd() [2/5]

```
template<class Field, bool ADD>
std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t inca,
            typename Field::ConstElement_ptr B,
            const size_t incb,
            typename Field::Element_ptr C,
            const size_t incc,
            FieldCategories::ModularTag )
```

11.6.1.3 fadd() [3/5]

```
template<class Field, bool ADD>
void fadd (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t inca,
            typename Field::ConstElement_ptr B,
            const size_t incb,
            typename Field::Element_ptr C,
            const size_t incc,
            FieldCategories::GenericTag )
```

11.6.1.4 fadd() [4/5]

```
template<class Field, bool ADD>
std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t inca,
            typename Field::ConstElement_ptr B,
            const size_t incb,
            typename Field::Element_ptr C,
            const size_t incc,
            FieldCategories::UnparametricTag )  [inline]
```

11.6.1.5 fadd() [5/5]

```
template<class Field, bool ADD>
std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
              const Field & F,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t inca,
              typename Field::ConstElement_ptr B,
              const size_t incb,
              typename Field::Element_ptr C,
              const size_t incc,
              FieldCategories::UnparametricTag )  [inline]
```

11.6.1.6 faxpy() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy
(
              const Field & F,
              const size_t N,
              const typename Field::Element a,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY,
              FieldCategories::ModularTag )  [inline]
```

11.6.1.7 faxpy() [2/2]

```
template<class Field, class FC>
void faxpy (
              const Field & F,
              const size_t N,
              const typename Field::Element a,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY,
              FC )  [inline]
```

11.6.1.8 freduce() [1/4]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
              const Field & F,
              const size_t m,
              typename Field::Element_ptr A,
              const size_t incX,
              FieldCategories::ModularTag )  [inline]
```

11.6.1.9 freduce() [2/4]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
              const Field & F,
              const size_t m,
              typename Field::ConstElement_ptr B,
              const size_t incY,
              typename Field::Element_ptr A,
              const size_t incX,
              FieldCategories::ModularTag )  [inline]
```

11.6.1.10 freduce() [3/4]

```
template<class Field, class FC>
void freduce (
              const Field & F,
              const size_t m,
              typename Field::Element_ptr A,
              const size_t incX,
              FC )  [inline]
```

11.6.1.11 freduce() [4/4]

```
template<class Field, class FC>
void freduce (
              const Field & F,
              const size_t m,
              typename Field::ConstElement_ptr B,
              const size_t incY,
              typename Field::Element_ptr A,
              const size_t incX,
              FC )  [inline]
```

11.6.1.12 fscalin() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin
(
              const Field & F,
              const size_t N,
              const typename Field::Element a,
              typename Field::Element_ptr X,
              const size_t incX,
              FieldCategories::ModularTag )  [inline]
```

11.6.1.13 fscal() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal
(
              const Field & F,
              const size_t N,
              const typename Field::Element a,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY,
              FieldCategories::ModularTag )  [inline]
```

11.6.1.14 fscalin() [2/2]

```
template<class Field, class FC>
void fscalin (
              const Field & F,
              const size_t n,
              const typename Field::Element a,
              typename Field::Element_ptr X,
              const size_t incX,
              FC )  [inline]
```

11.6.1.15 fscal() [2/2]

```
template<class Field, class FC>
void fscal (
              const Field & F,
              const size_t N,
              const typename Field::Element a,
              typename Field::ConstElement_ptr X,
              const size_t incX,
              typename Field::Element_ptr Y,
              const size_t incY,
              FC )  [inline]
```

11.6.1.16 igebb44()

```
template<enum number_kind K>
void igebb44 (
              size_t i,
              size_t j,
              size_t depth,
              size_t pdeth,
              const int64_t alpha,
              const int64_t * blA,
              const int64_t * blB,
              int64_t * C,
              size_t ldc)  [inline]
```

11.6.1.17 igebb24()

```
template<enum number_kind K>
void igebb24 (
             size_t i,
             size_t j,
             size_t depth,
             size_t pdeth,
             const int64_t alpha,
             const int64_t * blA,
             const int64_t * blB,
             int64_t * C,
             size_t ldc)  [inline]
```

11.6.1.18 igebb14()

```
template<enum number_kind K>
void igebb14 (
             size_t i,
             size_t j,
             size_t depth,
             size_t pdeth,
             const int64_t alpha,
             const int64_t * blA,
             const int64_t * blB,
             int64_t * C,
             size_t ldc)  [inline]
```

11.6.1.19 igebb41()

```
template<enum number_kind K>
void igebb41 (
             size_t i,
             size_t j,
             size_t depth,
             size_t pdeth,
             const int64_t alpha,
             const int64_t * blA,
             const int64_t * blB,
             int64_t * C,
             size_t ldc)  [inline]
```

bug ,B_0 dans VEC_MADD_32 ?

11.6.1.20 igebb21()

```
template<enum number_kind K>
void igebb21 (
             size_t i,
             size_t j,
             size_t depth,
             size_t pdeth,
             const int64_t alpha,
             const int64_t * blA,
             const int64_t * blB,
             int64_t * C,
             size_t ldc)  [inline]
```

11.6.1.21 igebb11()

```
template<enum number_kind K>
void igebb11 (
             size_t i,
             size_t j,
             size_t depth,
             size_t pdeth,
             const int64_t alpha,
             const int64_t * blA,
             const int64_t * blB,
             int64_t * C,
             size_t ldc)  [inline]
```

11.6.1.22 igebp()

```
template<enum number_kind K>
void igebp (
             size_t rows,
             size_t cols,
             size_t depth,
             const int64_t alpha,
             const int64_t * blockA,
             size_t lda,
             const int64_t * blockB,
             size_t ldb,
             int64_t * C,
             size_t ldc)
```

11.6.1.23 pack_lhs()

```
template<size_t k, bool transpose>
void pack_lhs (
             int64_t * XX,
             const int64_t * X,
             size_t ldx,
             size_t rows,
             size_t cols)
```

Bug this is fassign

Bug this is fassign

11.6.1.24 pack_rhs()

```
template<size_t k, bool transpose>
void pack_rhs (
             int64_t * XX,
             const int64_t * X,
             size_t ldx,
             size_t rows,
             size_t cols)
```

Bug this is fassign

Bug this is fassign

11.6.1.25 gebp()

```
void gebp (
            size_t rows,
            size_t cols,
            size_t depth,
            int64_t * C,
            size_t ldc,
            const int64_t * blockA,
            size_t lda,
            const int64_t * BlockB,
            size_t ldb,
            int64_t * BlockW)
```

11.6.1.26 BlockingFactor()

```
void BlockingFactor (
            size_t & m,
            size_t & n,
            size_t & k)  [inline]
```

11.7 FFLAS::details_spmv Namespace Reference

Data Structures

- struct Coo

11.8 FFLAS::ElementCategories Namespace Reference

Data Structures

- struct ArbitraryPrecIntTag

  Arbitrary precision integers: GMP.
- struct FixedPrecIntTag

  Fixed precision integers above machine precision: Givaro::recInt.
- struct GenericTag

  default is generic
- struct MachineFloatTag

  float or double
- struct MachineIntTag

  short, int, long, long long, and unsigned variants
- struct RNSElementTag

  Representation in a Residue Number System.

11.9 FFLAS::FieldCategories Namespace Reference

Traits and categories will need to be placed in a proper file later.

Data Structures

- struct GenericTag

  generic ring.
- struct ModularTag

  This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
- struct UnparametricTag

  If the field uses a representation with infix operators.

11.9.1 Detailed Description

Traits and categories will need to be placed in a proper file later.

11.10 FFLAS::MMHelperAlgo Namespace Reference

Data Structures

- struct Auto
- struct Bini
- struct Classic
- struct DivideAndConquer
- struct Winograd
- struct WinogradPar

11.11 FFLAS::ModeCategories Namespace Reference

Specifies the mode of action for an algorithm w.r.t.

Data Structures

- struct ConvertTo

  Force conversion to appropriate element type of ElementCategory T.
- struct DefaultBoundedTag

  Use standard field operations, but keeps track of bounds on input and output.
- struct DefaultTag

  No specific mode of action: use standard field operations.
- struct DelayedTag

  Performs field operations with delayed mod reductions. Ensures result is reduced.
- struct LazyTag

  Performs field operations with delayed mod only when necessary. Result may not be reduced.

11.11.1 Detailed Description

Specifies the mode of action for an algorithm w.r.t.

its field

11.12 FFLAS::ParSeqHelper Namespace Reference

ParSeqHelper for both fgemm and ftrsm.

Data Structures

- struct Compose
- struct Parallel
- struct Sequential

11.12.1 Detailed Description

ParSeqHelper for both fgemm and ftrsm.

ParSeqHelper for both fgemm and ftrsm

11.13 FFLAS::Protected Namespace Reference

Data Structures

- class AreEqual
- class AreEqual< X, X >
- class ftrmmLeftLowerNoTransNonUnit
- class ftrmmLeftLowerNoTransUnit
- class ftrmmLeftLowerTransNonUnit
- class ftrmmLeftLowerTransUnit
- class ftrmmLeftUpperNoTransNonUnit
- class ftrmmLeftUpperNoTransUnit
- class ftrmmLeftUpperTransNonUnit
- class ftrmmLeftUpperTransUnit
- class ftrmmRightLowerNoTransNonUnit
- class ftrmmRightLowerNoTransUnit
- class ftrmmRightLowerTransNonUnit
- class ftrmmRightLowerTransUnit
- class ftrmmRightUpperNoTransNonUnit
- class ftrmmRightUpperNoTransUnit
- class ftrmmRightUpperTransNonUnit
- class ftrmmRightUpperTransUnit
- class ftrsmLeftLowerNoTransNonUnit
- class ftrsmLeftLowerNoTransUnit
- class ftrsmLeftLowerTransNonUnit
- class ftrsmLeftLowerTransUnit
- class ftrsmLeftUpperNoTransNonUnit

  Computes the maximal size for delaying the modular reduction in a triangular system resolution.
- class ftrsmLeftUpperNoTransUnit
- class ftrsmLeftUpperTransNonUnit
- class ftrsmLeftUpperTransUnit
- class ftrsmRightLowerNoTransNonUnit
- class ftrsmRightLowerNoTransUnit
- class ftrsmRightLowerTransNonUnit
- class ftrsmRightLowerTransUnit
- class ftrsmRightUpperNoTransNonUnit
- class ftrsmRightUpperNoTransUnit
- class ftrsmRightUpperTransNonUnit
- class ftrsmRightUpperTransUnit

Functions

- template<class Field>
  double computeFactorClassic (const Field &F)
- template<> double computeFactorClassic (const Givaro::ModularBalanced< double > &F)
- template<> double computeFactorClassic (const Givaro::ModularBalanced< float > &F)
- template<class Field>
  size_t DotProdBoundClassic (const Field &F, const typename Field::Element &beta)
- template<class Field>
  size_t TRSMBound (const Field &)

  TRSMBound.
- template<class Element>
  size_t TRSMBound (const Givaro::Modular< Element > &F)

  Specialization for positive modular representation over double Computes nmax s.t.
- template<class Element>
  size_t TRSMBound (const Givaro::ModularBalanced< Element > &F)

  Specialization for balanced modular representation over double.
- template<class NewField, class Field, class FieldMode>
  Field::Element_ptr fgemm_convert (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, Par↩ SeqTrait > &WH)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, Par↩ SeqTrait > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class Field>
  Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

- template<class Field>
  int WinogradThreshold (const Field &F)

  Computes the number of recursive levels to perform.
- template<> int WinogradThreshold (const Givaro::Modular< float > &F)
- template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)
- template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)
- template<class Field>
  int WinogradSteps (const Field &F, const size_t &m)

  Computes the number of recursive levels to perform.
- template<class Field, class FieldMode>
  void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field, class FieldMode>
  void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field, class FieldMode>
  void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)
- template<typename FloatElement, class Field>
  Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)
- template<class FloatElement, class Field>
  void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)
- template<class NewField, class Field, class FieldMode>
  Field::Element_ptr fsyrk_convert (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, FieldMode > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const FFLAS_UPLO UpLo, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element

&Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)

- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class DFE>
  size_t min_types (const DFE &k)
- template<> size_t min_types (const RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::rint< 10 > &k)
- template<> size_t min_types (const Givaro::Integer &k)
- template<class T>
  bool unfit (T x)
- template<> bool unfit (int64_t x)
- template<size_t K>
  bool unfit (RecInt::rint< K > x)
- template<> bool unfit (RecInt::rint< 6 > x)
- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)
- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)
- void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)
- template<class Field>
  void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)
- template<class Field>
  void MatF2MatFl_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)

11.13.1 Function Documentation

11.13.1.1 computeFactorClassic() [1/3]

```
template<class Field>
double computeFactorClassic (
            const Field & F)  [inline]
```

11.13.1.2 computeFactorClassic() [2/3]

```
template<>
double computeFactorClassic (
            const Givaro::ModularBalanced< double > & F)  [inline]
```

11.13.1.3 computeFactorClassic() [3/3]

```
template<>
double computeFactorClassic (
            const Givaro::ModularBalanced< float > & F)  [inline]
```

11.13.1.4 DotProdBoundClassic()

```
template<class Field>
size_t DotProdBoundClassic (
            const Field & F,
            const typename Field::Element & beta)  [inline]
```

11.13.1.5 TRSMBound() [1/3]

```
template<class Field>
size_t TRSMBound (
            const Field & )  [inline]
```

TRSMBound.

computes the maximal size for delaying the modular reduction in a triangular system resolution

This is the default version over an arbitrary field. It is currently never used (the recursive algorithm is run until n=1 in this case)

Parameters

F	Finite Field/Ring of the computation

11.13.1.6 TRSMBound() [2/3]

```
template<class Element>
size_t TRSMBound (
            const Givaro::Modular< Element > & F)  [inline]
```

Specialization for positive modular representation over double Computes nmax s.t.

(p-1)/2∗(p^{nmax-1} + (p-2)^{nmax-1}) $< 2^{53}$ See [Dumas Giorgi Pernet 06, arXiv:cs/0601133] Specialization for positive modular representation over float. Computes nmax s.t. (p-1)/2∗(p^{nmax-1} + (p-2)^{nmax-1}) $< 2^{24}$ @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

11.13.1.7 TRSMBound() [3/3]

```
template<class Element>
size_t TRSMBound (
            const Givaro::ModularBalanced< Element > & F)  [inline]
```

Specialization for balanced modular representation over double.

Computes nmax s.t. (p-1)/2∗(((p+1)/2)^{nmax-1}) $< 2^{53}$

Bibliography

• Dumas Giorgi Pernet 06, arXiv:cs/0601133

11.13.1.8 fgemm_convert()

```
template<class NewField, class Field, class FieldMode>
Field::Element_ptr fgemm_convert (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H)  [inline]
```

11.13.1.9 NeedPreAddReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.10 NeedPreAddReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.11 NeedPreSubReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreSubReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.12 NeedPreSubReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreSubReduction (
              Element & Outmin,
              Element & Outmax,
              Element & Op1min,
              Element & Op1max,
              Element & Op2min,
              Element & Op2max,
              MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.13 NeedDoublePreAddReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedDoublePreAddReduction (
              Element & Outmin,
              Element & Outmax,
              Element & Op1min,
              Element & Op1max,
              Element & Op2min,
              Element & Op2max,
              Element beta,
              MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.14 NeedDoublePreAddReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedDoublePreAddReduction (
              Element & Outmin,
              Element & Outmax,
              Element & Op1min,
              Element & Op1max,
              Element & Op2min,
              Element & Op2max,
              Element beta,
              MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.15 ScalAndReduce() [1/3]

```
template<class Field, class AlgoT, class ParSeqTrait>
void ScalAndReduce (
              const Field & F,
              const size_t N,
              const typename Field::Element alpha,
              typename Field::Element_ptr X,
              const size_t incX,
              const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H)  [inline]
```

11.13.1.16 ScalAndReduce() [2/3]

```
template<class Field, class AlgoT, class ParSeqTrait>
void ScalAndReduce (
             const Field & F,
             const size_t M,
             const size_t N,
             const typename Field::Element alpha,
             typename Field::Element_ptr A,
             const size_t lda,
             const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H)  [inline]
```

11.13.1.17 fsquareCommon()

```
template<class Field>
Field::Element_ptr fsquareCommon (
             const Field & F,
             const FFLAS_TRANSPOSE ta,
             const size_t n,
             const typename Field::Element alpha,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             const typename Field::Element beta,
             typename Field::Element_ptr C,
             const size_t ldc)  [inline]
```

11.13.1.18 WinogradThreshold() [1/4]

```
template<class Field>
int WinogradThreshold (
             const Field & F)  [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product AxB

11.13.1.19 WinogradThreshold() [2/4]

```
template<>
int WinogradThreshold (
             const Givaro::Modular< float > & F)  [inline]
```

11.13.1.20 WinogradThreshold() [3/4]

```
template<>
int WinogradThreshold (
             const Givaro::ModularBalanced< double > & F)  [inline]
```

11.13.1.21 WinogradThreshold() [4/4]

```
template<>
int WinogradThreshold (
            const Givaro::ModularBalanced< float > & F)  [inline]
```

11.13.1.22 WinogradSteps()

```
template<class Field>
int WinogradSteps (
            const Field & F,
            const size_t & m)  [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product AxB

11.13.1.23 DynamicPeeling()

```
template<class Field, class FieldMode>
void DynamicPeeling (
            const Field & F,
            const FFLAS_TRANSPOSE ta,
            const FFLAS_TRANSPOSE tb,
            const size_t m,
            const size_t n,
            const size_t k,
            const size_t mr,
            const size_t nr,
            const size_t kr,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
            const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↩
Field::Element Cmin,
            const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↩
Field::Element Cmax)  [inline]
```

11.13.1.24 DynamicPeeling2()

```
template<class Field, class FieldMode>
void DynamicPeeling2 (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const FFLAS_TRANSPOSE tb,
              const size_t m,
              const size_t n,
              const size_t k,
              const size_t mr,
              const size_t nr,
              const size_t kr,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              const typename Field::Element beta,
              typename Field::Element_ptr C,
              const size_t ldc,
              MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
              const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↩
Field::Element Cmin,
              const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↩
Field::Element Cmax)  [inline]
```

11.13.1.25 WinogradCalc()

```
template<class Field, class FieldMode>
void WinogradCalc (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const FFLAS_TRANSPOSE tb,
              const size_t mr,
              const size_t nr,
              const size_t kr,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::ConstElement_ptr B,
              const size_t ldb,
              const typename Field::Element beta,
              typename Field::Element_ptr C,
              const size_t ldc,
              MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H)  [inline]
```

11.13.1.26 fgemv_convert()

```
template<typename FloatElement, class Field>
Field::Element_ptr fgemv_convert (
              const Field & F,
              const FFLAS_TRANSPOSE ta,
              const size_t M,
```

```
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr X,
            const size_t incX,
            const typename Field::Element beta,
            typename Field::Element_ptr Y,
            const size_t incY)  [inline]
```

11.13.1.27 fger_convert()

```
template<class FloatElement, class Field>
void fger_convert (
            const Field & F,
            const size_t M,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr x,
            const size_t incx,
            typename Field::ConstElement_ptr y,
            const size_t incy,
            typename Field::Element_ptr A,
            const size_t lda)  [inline]
```

11.13.1.28 fsyrk_convert()

```
template<class NewField, class Field, class FieldMode>
Field::Element_ptr fsyrk_convert (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Classic, FieldMode > & H)  [inline]
```

11.13.1.29 ScalAndReduce() [3/3]

```
template<class Field, class AlgoT, class ParSeqTrait>
void ScalAndReduce (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const size_t N,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H)  [inline]
```

11.13.1.30 NeedPreScalReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreScalReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.31 NeedPreScalReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreScalReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.32 NeedPreAxpyReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreAxpyReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.33 NeedPreAxpyReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreAxpyReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.34 min_types() [1/7]

```
template<class DFE>
size_t min_types (
            const DFE & k)  [inline]
```

11.13.1.35 min_types() [2/7]

```
template<>
size_t min_types (
            const RecInt::rint< 6 > & k)  [inline]
```

11.13.1.36 min_types() [3/7]

```
template<>
size_t min_types (
            const RecInt::rint< 7 > & k)  [inline]
```

11.13.1.37 min_types() [4/7]

```
template<>
size_t min_types (
            const RecInt::rint< 8 > & k)  [inline]
```

11.13.1.38 min_types() [5/7]

```
template<>
size_t min_types (
            const RecInt::rint< 9 > & k)  [inline]
```

11.13.1.39 min_types() [6/7]

```
template<>
size_t min_types (
            const RecInt::rint< 10 > & k)  [inline]
```

11.13.1.40 min_types() [7/7]

```
template<>
size_t min_types (
            const Givaro::Integer & k)  [inline]
```

11.13.1.41 unfit() [1/4]

```
template<class T>
bool unfit (
            T x)  [inline]
```

11.13.1.42 unfit() [2/4]

```
template<>
bool unfit (
            int64_t x)  [inline]
```

11.13.1.43 unfit() [3/4]

```
template<size_t K>
bool unfit (
            RecInt::rint< K > x)  [inline]
```

11.13.1.44 unfit() [4/4]

```
template<>
bool unfit (
            RecInt::rint< 6 > x)  [inline]
```

11.13.1.45 igemm_colmajor() [1/2]

```
template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>
void igemm_colmajor (
            size_t rows,
            size_t cols,
            size_t depth,
            const int64_t alpha,
            const int64_t * A,
            size_t lda,
            const int64_t * B,
            size_t ldb,
            int64_t * C,
            size_t ldc)
```

11.13.1.46 igemm_colmajor() [2/2]

```
template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>
void igemm_colmajor (
            size_t rows,
            size_t cols,
            size_t depth,
            const int64_t alpha,
            const int64_t * A,
            size_t lda,
            const int64_t * B,
            size_t ldb,
            int64_t * C,
            size_t ldc)
```

11.13.1.47 igemm()

```
void igemm (
            const enum FFLAS_TRANSPOSE TransA,
            const enum FFLAS_TRANSPOSE TransB,
            size_t rows,
            size_t cols,
            size_t depth,
            const int64_t alpha,
```

```
            const int64_t * A,
            size_t lda,
            const int64_t * B,
            size_t ldb,
            const int64_t beta,
            int64_t * C,
            size_t ldc)  [inline]
```

Todo use primitive (no Field()) and specialise for int64.

11.13.1.48 MatF2MatD_Triangular()

```
template<class Field>
void MatF2MatD_Triangular (
            const Field & F,
            Givaro::DoubleDomain::Element_ptr S,
            const size_t lds,
            typename Field::ConstElement_ptr const E,
            const size_t lde,
            const size_t m,
            const size_t n)
```

11.13.1.49 MatF2MatFl_Triangular()

```
template<class Field>
void MatF2MatFl_Triangular (
            const Field & F,
            Givaro::FloatDomain::Element_ptr S,
            const size_t lds,
            typename Field::ConstElement_ptr const E,
            const size_t lde,
            const size_t m,
            const size_t n)
```

Todo do finit(...,FFLAS_TRANS,FFLAS_DIAG)
do fconvert(...,FFLAS_TRANS,FFLAS_DIAG)

11.14 FFLAS::sell_details Namespace Reference

Data Structures

- struct Coo
- struct Info

11.15 FFLAS::sparse_details Namespace Reference

Functions

- template<class Field>
  void init_y (const Field &F, const size_t m, const typename Field::Element b, typename Field::Element_ptr y)
- template<class Field>
  void init_y (const Field &F, const size_t m, const size_t n, const typename Field::Element b, typename Field::Element_ptr y, const int ldy)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)

- template<class Field, class SM, class FCat, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM

&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)

- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)

11.15.1 Function Documentation

11.15.1.1 init_y() [1/2]

```
template<class Field>
void init_y (
            const Field & F,
            const size_t m,
            const typename Field::Element b,
            typename Field::Element_ptr y)  [inline]
```

11.15.1.2 init_y() [2/2]

```
template<class Field>
void init_y (
            const Field & F,
            const size_t m,
            const size_t n,
            const typename Field::Element b,
            typename Field::Element_ptr y,
            const int ldy)  [inline]
```

11.15.1.3 fspmv_dispatch() [1/2]

```
template<class Field, class SM, class FC, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type fspmv_dispatch (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FC fc,
            MZO mzo)  [inline]
```

11.15.1.4 fspmv_dispatch() [2/2]

```
template<class Field, class SM, class FC, class MZO>
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type fspmv_dispatch (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FC fc,
            MZO mzo)  [inline]
```

11.15.1.5 fspmv() [1/12]

```
template<class Field, class SM>
void fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.6 fspmv() [2/12]

```
template<class Field, class SM>
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.7 fspmv() [3/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.8 fspmv() [4/12]

```
template<class Field, class SM>
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.9 fspmv() [5/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.10 fspmv() [6/12]

```
template<class Field, class SM>
void fspmv (
              const Field & F,
              const SM & A,
              typename Field::ConstElement_ptr x,
              typename Field::Element_ptr y,
              FieldCategories::GenericTag ,
              ZOSparseMatrix )  [inline]
```

11.15.1.11 fspmv() [7/12]

```
template<class Field, class SM>
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
              const Field & F,
              const SM & A,
              typename Field::ConstElement_ptr x,
              typename Field::Element_ptr y,
              FieldCategories::UnparametricTag ,
              ZOSparseMatrix )  [inline]
```

11.15.1.12 fspmv() [8/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
              const Field & F,
              const SM & A,
              typename Field::ConstElement_ptr x,
              typename Field::Element_ptr y,
              FieldCategories::UnparametricTag ,
              ZOSparseMatrix )  [inline]
```

11.15.1.13 fspmv() [9/12]

```
template<class Field, class SM>
void fspmv (
              const Field & F,
              const SM & A,
              typename Field::ConstElement_ptr x,
              typename Field::Element_ptr y,
              FieldCategories::ModularTag ,
              std::true_type )  [inline]
```

11.15.1.14 fspmm_dispatch() [1/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type fspmm_dispatch (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FCat ,
             MZO )  [inline]
```

11.15.1.15 fspmm_dispatch() [2/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type fspmm_dispatch (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FCat ,
             MZO )  [inline]
```

11.15.1.16 fspmm() [1/9]

```
template<class Field, class SM>
void fspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.17 fspmm() [2/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
             const Field & F,
             const SM & A,
```

```
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::UnparametricTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.18 fspmm() [3/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::UnparametricTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.19 fspmm() [4/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::ModularTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.20 fspmm() [5/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::ModularTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.21 fspmm() [6/9]

```
template<class Field, class SM>
void fspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.22 fspmm() [7/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.23 fspmm() [8/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.24 fspmm() [9/9]

```
template<class Field, class SM>
void fspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::ModularTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.25 pfspmm_dispatch() [1/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type pfspmm_dispatch (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FCat ,
             MZO )  [inline]
```

11.15.1.26 pfspmm_dispatch() [2/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type pfspmm_dispatch (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FCat ,
             MZO )  [inline]
```

11.15.1.27 pfspmm() [1/9]

```
template<class Field, class SM>
void pfspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.28 pfspmm() [2/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
             const Field & F,
             const SM & A,
```

```
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::UnparametricTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.29 pfspmm() [3/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::UnparametricTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.30 pfspmm() [4/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::ModularTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.31 pfspmm() [5/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (
            const Field & F,
            const SM & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            int ldx,
            typename Field::Element_ptr y,
            int ldy,
            FieldCategories::ModularTag ,
            NotZOSparseMatrix )  [inline]
```

11.15.1.32 pfspmm() [6/9]

```
template<class Field, class SM>
void pfspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.33 pfspmm() [7/9]

```
template<class Field, class SM>
std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.34 pfspmm() [8/9]

```
template<class Field, class SM>
std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.35 pfspmm() [9/9]

```
template<class Field, class SM>
void pfspmm (
             const Field & F,
             const SM & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::ModularTag ,
             ZOSparseMatrix )  [inline]
```

11.15.1.36 pfspmv() [1/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag ,
             std::false_type )  [inline]
```

11.15.1.37 pfspmv() [2/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             std::false_type )  [inline]
```

11.15.1.38 pfspmv() [3/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag ,
             std::false_type )  [inline]
```

11.15.1.39 pfspmv() [4/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag ,
             std::true_type )  [inline]
```

11.15.1.40 pfspmv() [5/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             std::true_type )  [inline]
```

11.15.1.41 pfspmv() [6/6]

```
template<class Field, class SM>
void pfspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag ,
             std::true_type )  [inline]
```

11.15.1.42 fspmv() [10/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↩
::Element >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.43 fspmv() [11/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↩
::Element >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::ModularTag ,
             NotZOSparseMatrix )  [inline]
```

11.15.1.44 fspmv() [12/12]

```
template<class Field, class SM>
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↩
::Element >::value >::type fspmv (
             const Field & F,
             const SM & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag ,
             ZOSparseMatrix )  [inline]
```

11.16 FFLAS::sparse_details_impl Namespace Reference

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)

- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)

- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)
- template<class Field, class Func>
  void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)
- template<class Field, class Func>
  void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)

- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)

- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

11.16.1 Function Documentation

11.16.1.1 fspmm() [1/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.2 fspmm() [2/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.3 fspmm() [3/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.4 fspmm_simd_aligned() [1/2]

```
template<class Field>
void fspmm_simd_aligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.5 fspmm_simd_unaligned() [1/2]

```
template<class Field>
void fspmm_simd_unaligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.6 fspmm_one() [1/4]

```
template<class Field>
void fspmm_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.7 fspmm_mone() [1/4]

```
template<class Field>
void fspmm_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.8 fspmm_one_simd_aligned() [1/3]

```
template<class Field>
void fspmm_one_simd_aligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.9 fspmm_one_simd_unaligned() [1/3]

```
template<class Field>
void fspmm_one_simd_unaligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.10 fspmm_mone_simd_aligned() [1/3]

```
template<class Field>
void fspmm_mone_simd_aligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.11 fspmm_mone_simd_unaligned() [1/3]

```
template<class Field>
void fspmm_mone_simd_unaligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.12 fspmv() [1/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.13 fspmv() [2/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.14 fspmv() [3/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const uint64_t kmax)  [inline]
```

11.16.1.15 fspmv_one() [1/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.16 fspmv_mone() [1/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.17 fspmv_one() [2/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.18 fspmv_mone() [2/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.19 pfspmm() [1/18]

```
template<class Field>
void pfspmm (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.20 pfspmm() [2/18]

```
template<class Field>
void pfspmm (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.21 pfspmm() [3/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.22 pfspmm_one() [1/2]

```
template<class Field>
void pfspmm_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.23 pfspmm_mone() [1/2]

```
template<class Field>
void pfspmm_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.24 pfspmm_one() [2/2]

```
template<class Field>
void pfspmm_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.25 pfspmm_mone() [2/2]

```
template<class Field>
void pfspmm_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.26 pfspmv() [1/18]

```
template<class Field>
void pfspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.27 pfspmv_task()

```
template<class Field>
void pfspmv_task (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const index_t iStart,
            const index_t iStop,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.28 pfspmv() [2/18]

```
template<class Field>
void pfspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.29 pfspmv() [3/18]

```
template<class Field>
void pfspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const int64_t kmax)  [inline]
```

11.16.1.30 pfspmv_one() [1/8]

```
template<class Field>
void pfspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.31 pfspmv_mone() [1/8]

```
template<class Field>
void pfspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.32 pfspmv_one() [2/8]

```
template<class Field>
void pfspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.33 pfspmv_mone() [2/8]

```
template<class Field>
void pfspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.34 fspmm() [4/15]

```
template<class Field>
void fspmm (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.35 fspmm() [5/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR > & A,
             index_t blockSize,
             typename Field::ConstElement_ptr x_,
             index_t ldx,
             typename Field::Element_ptr y_,
             index_t ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.36 fspmm_simd_aligned() [2/2]

```
template<class Field>
void fspmm_simd_aligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.37 fspmm_simd_unaligned() [2/2]

```
template<class Field>
void fspmm_simd_unaligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.38 fspmm() [6/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.39 fspmm_one() [2/4]

```
template<class Field>
void fspmm_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.40 fspmm_mone() [2/4]

```
template<class Field>
void fspmm_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.41 fspmm_one_simd_aligned() [2/3]

```
template<class Field>
void fspmm_one_simd_aligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.42 fspmm_one_simd_unaligned() [2/3]

```
template<class Field>
void fspmm_one_simd_unaligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.43 fspmm_mone_simd_aligned() [2/3]

```
template<class Field>
void fspmm_mone_simd_aligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.44 fspmm_mone_simd_unaligned() [2/3]

```
template<class Field>
void fspmm_mone_simd_unaligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.45 fspmv() [4/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.46 fspmv() [5/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.47 fspmv() [6/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const int64_t kmax)  [inline]
```

11.16.1.48 fspmv_one() [3/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.49 fspmv_mone() [3/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.50 fspmv_one() [4/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.51 fspmv_mone() [4/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.52 pfspmm() [4/18]

```
template<class Field>
void pfspmm (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.53 pfspmm() [5/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.54 pfspmm() [6/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.55 pfspmm() [7/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.56 pfspmm() [8/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             const int64_t kmax)  [inline]
```

11.16.1.57 pfspmm() [9/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.58 pfspmv() [4/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.59 pfspmv() [5/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.60 pfspmv() [6/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const int64_t kmax)  [inline]
```

11.16.1.61 fspmm() [7/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.62 fspmm() [8/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.63 fspmm() [9/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.64 fspmv() [7/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.65 fspmv() [8/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.66 fspmv() [9/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const uint64_t kmax)  [inline]
```

11.16.1.67 pfspmm() [10/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.68 pfspmm() [11/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.69 pfspmm() [12/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.70 pfspmm() [13/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.71 pfspmm() [14/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             const int64_t kmax)  [inline]
```

11.16.1.72 pfspmm() [15/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             const int64_t kmax)  [inline]
```

11.16.1.73 pfspmm_zo() [1/2]

```
template<class Field, class Func>
void pfspmm_zo (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             Func && func)  [inline]
```

11.16.1.74 pfspmm_zo() [2/2]

```
template<class Field, class Func>
void pfspmm_zo (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             Func && func)  [inline]
```

11.16.1.75 pfspmv() [7/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.76 pfspmv() [8/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.77 pfspmv() [9/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const int64_t kmax)  [inline]
```

11.16.1.78 pfspmv_one() [3/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.79 pfspmv_mone() [3/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.80 pfspmv_one() [4/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.81 pfspmv_mone() [4/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.82 fspmm() [10/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.83 fspmm() [11/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.84 fspmm() [12/15]

```
template<class Field>
void fspmm (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            const int64_t kmax)  [inline]
```

11.16.1.85 fspmm_mone() [3/4]

```
template<class Field>
void fspmm_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.86 fspmm_one() [3/4]

```
template<class Field>
void fspmm_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.87 fspmm_mone() [4/4]

```
template<class Field>
void fspmm_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.88 fspmm_one() [4/4]

```
template<class Field>
void fspmm_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.89 fspmm_one_simd_aligned() [3/3]

```
template<class Field>
void fspmm_one_simd_aligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.90 fspmm_one_simd_unaligned() [3/3]

```
template<class Field>
void fspmm_one_simd_unaligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.91 fspmm_mone_simd_aligned() [3/3]

```
template<class Field>
void fspmm_mone_simd_aligned (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x_,
             int ldx,
             typename Field::Element_ptr y_,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.92 fspmm_mone_simd_unaligned() [3/3]

```
template<class Field>
void fspmm_mone_simd_unaligned (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
            size_t blockSize,
            typename Field::ConstElement_ptr x_,
            int ldx,
            typename Field::Element_ptr y_,
            int ldy,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.93 fspmv() [10/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.94 fspmv() [11/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.95 fspmv() [12/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const uint64_t kmax)  [inline]
```

11.16.1.96 fspmv_one() [5/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.97 fspmv_mone() [5/10]

```
template<class Field>
void fspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.98 fspmv_one() [6/10]

```
template<class Field>
void fspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.99 fspmv_mone() [6/10]

```
template<class Field>
void fspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.100 pfspmv() [10/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.101 pfspmv() [11/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.102 pfspmv() [12/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const uint64_t kmax)  [inline]
```

11.16.1.103 pfspmv_one() [5/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.104 pfspmv_mone() [5/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.105 pfspmv_one() [6/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.106 pfspmv_mone() [6/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.107 fspmv() [13/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.108 fspmv_simd() [1/4]

```
template<class Field>
void fspmv_simd (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.109 fspmv() [14/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.110 fspmv_simd() [2/4]

```
template<class Field>
void fspmv_simd (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const uint64_t kmax)  [inline]
```

11.16.1.111 fspmv() [15/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const uint64_t kmax)  [inline]
```

11.16.1.112 fspmv_one() [7/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.113 fspmv_mone() [7/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.114 fspmv_one() [8/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.115 fspmv_mone() [8/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.116 fspmv_one_simd() [1/2]

```
template<class Field>
void fspmv_one_simd (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.117 fspmv_mone_simd() [1/2]

```
template<class Field>
void fspmv_mone_simd (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.118 pfspmm() [16/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.119 pfspmm() [17/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.120 pfspmm() [18/18]

```
template<class Field>
void pfspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             uint64_t kmax)  [inline]
```

11.16.1.121 pfspmv() [13/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.122 pfspmv() [14/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.123 pfspmv() [15/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             uint64_t kmax)  [inline]
```

11.16.1.124 fspmm() [13/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.125 fspmm() [14/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.126 fspmm() [15/15]

```
template<class Field>
void fspmm (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             size_t blockSize,
             typename Field::ConstElement_ptr x,
             int ldx,
             typename Field::Element_ptr y,
             int ldy,
             uint64_t kmax)  [inline]
```

11.16.1.127 fspmv() [16/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.128 fspmv() [17/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.129 fspmv() [18/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
             typename Field::ConstElement_ptr x,
             typename Field::Element_ptr y,
             uint64_t kmax)  [inline]
```

11.16.1.130 pfspmv() [16/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.131 pfspmv() [17/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.132 pfspmv() [18/18]

```
template<class Field>
void pfspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const int64_t kmax)  [inline]
```

11.16.1.133 pfspmv_one() [7/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.134 pfspmv_mone() [7/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.135 pfspmv_one() [8/8]

```
template<class Field>
void pfspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.136 pfspmv_mone() [8/8]

```
template<class Field>
void pfspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.137 fspmv() [19/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::GenericTag )  [inline]
```

11.16.1.138 fspmv_simd() [3/4]

```
template<class Field>
void fspmv_simd (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.139 fspmv() [20/21]

```
template<class Field>
void fspmv (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.140 fspmv_simd() [4/4]

```
template<class Field>
void fspmv_simd (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             const uint64_t kmax)  [inline]
```

11.16.1.141 fspmv() [21/21]

```
template<class Field>
void fspmv (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::SELL > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            const uint64_t kmax)  [inline]
```

11.16.1.142 fspmv_one() [9/10]

```
template<class Field>
void fspmv_one (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.143 fspmv_mone() [9/10]

```
template<class Field>
void fspmv_mone (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::GenericTag )  [inline]
```

11.16.1.144 fspmv_one_simd() [2/2]

```
template<class Field>
void fspmv_one_simd (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.145 fspmv_mone_simd() [2/2]

```
template<class Field>
void fspmv_mone_simd (
            const Field & F,
            const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
            typename Field::ConstElement_ptr x_,
            typename Field::Element_ptr y_,
            FieldCategories::UnparametricTag )  [inline]
```

11.16.1.146 fspmv_one() [10/10]

```
template<class Field>
void fspmv_one (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.16.1.147 fspmv_mone() [10/10]

```
template<class Field>
void fspmv_mone (
             const Field & F,
             const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
             typename Field::ConstElement_ptr x_,
             typename Field::Element_ptr y_,
             FieldCategories::UnparametricTag )  [inline]
```

11.17 FFLAS::StrategyParameter Namespace Reference

Data Structures

- struct Fixed
- struct Grain
- struct Threads
- struct ThreeD
- struct ThreeDAdaptive
- struct ThreeDInPlace
- struct TwoD
- struct TwoDAdaptive

11.18 FFLAS::StructureHelper Namespace Reference

StructureHelper for ftrsm.

Data Structures

- struct Hybrid
- struct Iterative
- struct Recursive

11.18.1 Detailed Description

StructureHelper for ftrsm.

11.19 FFLAS::vectorised Namespace Reference

Namespaces

- namespace unswitch

Data Structures

- struct HelperMod
- struct HelperMod< Field, ElementCategories::MachineIntTag >
- struct HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >
- struct HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >

Functions

- template<class SimdT, class Element, bool positive>
  std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)
- template<class SimdT, class Element, bool positive>
  std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)
- template<class Element>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class Element>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY)
- template<class T>
  std::enable_if<!std::is_integral< T >::value, T >::type reduce (T A, T B)
- template<class T>
  std::enable_if< std::is_integral< T >::value, T >::type reduce (T A, T B)
- template<> Givaro::Integer reduce (Givaro::Integer A, Givaro::Integer B)
- float reduce (float A, float B, float invB, float min, float max)
- double reduce (double A, double B, double invB, double min, double max)
- int64_t reduce (int64_t A, int64_t p, double invp, double min, double max, int64_t pow50rem)
- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineIntTag > &H)

- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineFloatTag > &H)
- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY)

11.19.1 Function Documentation

11.19.1.1 VEC_ADD()

```
template<class SimdT, class Element, bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (
              SimdT & C,
              SimdT & A,
              SimdT & B,
              SimdT & Q,
              SimdT & T,
              SimdT & P,
              SimdT & NEGP,
              SimdT & MIN,
              SimdT & MAX)  [inline]
```

11.19.1.2 addp()

```
template<bool positive, class Element, class T1, class T2>
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (
              Element * T,
              const Element * TA,
              const Element * TB,
              size_t n,
              Element p,
              T1 min_,
              T2 max_)  [inline]
```

11.19.1.3 VEC_SUB()

```
template<class SimdT, class Element, bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (
            SimdT & C,
            SimdT & A,
            SimdT & B,
            SimdT & Q,
            SimdT & T,
            SimdT & P,
            SimdT & NEGP,
            SimdT & MIN,
            SimdT & MAX)  [inline]
```

11.19.1.4 subp()

```
template<bool positive, class Element, class T1, class T2>
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (
            Element * T,
            const Element * TA,
            const Element * TB,
            const size_t n,
            const Element p,
            const T1 min_,
            const T2 max_)  [inline]
```

11.19.1.5 add()

```
template<class Element>
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (
            Element * T,
            const Element * TA,
            const Element * TB,
            size_t n)  [inline]
```

11.19.1.6 sub()

```
template<class Element>
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (
            Element * T,
            const Element * TA,
            const Element * TB,
            size_t n)  [inline]
```

11.19.1.7 axpyp() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n)  [inline]
```

11.19.1.8 axpyp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            const size_t incX,
            const size_t incY)  [inline]
```

11.19.1.9 reduce() [1/9]

```
template<class T>
std::enable_if<!std::is_integral< T >::value, T >::type reduce (
            T A,
            T B)  [inline]
```

11.19.1.10 reduce() [2/9]

```
template<class T>
std::enable_if< std::is_integral< T >::value, T >::type reduce (
            T A,
            T B)  [inline]
```

11.19.1.11 reduce() [3/9]

```
template<>
Givaro::Integer reduce (
            Givaro::Integer A,
            Givaro::Integer B)  [inline]
```

11.19.1.12 reduce() [4/9]

```
float reduce (
            float A,
            float B,
            float invB,
            float min,
            float max)  [inline]
```

11.19.1.13 reduce() [5/9]

```
double reduce (
            double A,
            double B,
            double invB,
            double min,
            double max)  [inline]
```

11.19.1.14 reduce() [6/9]

```
int64_t reduce (
            int64_t A,
            int64_t p,
            double invp,
            double min,
            double max,
            int64_t pow50rem)  [inline]
```

11.19.1.15 reduce() [7/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineIntTag > & H)  [inline]
```

11.19.1.16 reduce() [8/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineFloatTag > & H)  [inline]
```

11.19.1.17 reduce() [9/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H)  [inline]
```

11.19.1.18 modp() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
            const Field & F,
            typename Field::ConstElement_ptr U,
            const size_t & n,
            typename Field::Element_ptr T)  [inline]
```

11.19.1.19 modp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
            const Field & F,
            typename Field::ConstElement_ptr U,
            const size_t & n,
            const size_t & incX,
            typename Field::Element_ptr T)  [inline]
```

11.19.1.20 scalp() [1/3]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
              const Field & F,
              typename Field::Element_ptr T,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr U,
              const size_t n)  [inline]
```

11.19.1.21 scalp() [2/3]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
              const Field & F,
              typename Field::Element_ptr T,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr U,
              const size_t n,
              const size_t & incX)  [inline]
```

11.19.1.22 scalp() [3/3]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
              const Field & F,
              typename Field::Element_ptr T,
              const typename Field::Element alpha,
              typename Field::ConstElement_ptr U,
              const size_t n,
              const size_t & incX,
              const size_t & incY)  [inline]
```

11.20 FFLAS::vectorised::unswitch Namespace Reference

Functions

- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY, HelperMod< Field > &H)

- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T, HelperMod< Field > &H)
- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY, HelperMod< Field > &H)

11.20.1 Function Documentation

11.20.1.1 axpyp() [1/2]

```
template<class Field>
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type axpyp (
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            HelperMod< Field > & H)  [inline]
```

11.20.1.2 axpyp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            const size_t incX,
            const size_t incY,
            HelperMod< Field > & H)  [inline]
```

11.20.1.3 modp() [1/2]

```
template<class Field>
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type modp (
            const Field & F,
            typename Field::ConstElement_ptr U,
            const size_t & n,
            typename Field::Element_ptr T,
            HelperMod< Field > & H)  [inline]
```

11.20.1.4 modp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
            const Field & F,
            typename Field::ConstElement_ptr U,
            const size_t & n,
            const size_t & incX,
            typename Field::Element_ptr T,
            HelperMod< Field > & H)  [inline]
```

11.20.1.5 scalp() [1/3]

```
template<class Field>
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type scalp (
            const Field & F,
            typename Field::Element_ptr T,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr U,
            const size_t n,
            HelperMod< Field > & H)  [inline]
```

11.20.1.6 scalp() [2/3]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
            const Field & F,
            typename Field::Element_ptr T,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr U,
            const size_t n,
            const size_t & incX,
            HelperMod< Field > & H)  [inline]
```

11.20.1.7 scalp() [3/3]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
            const Field & F,
            typename Field::Element_ptr T,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr U,
            const size_t n,
            const size_t & incX,
            const size_t & incY,
            HelperMod< Field > & H)  [inline]
```

11.21 FFPACK Namespace Reference

Finite **F**ield **PACK** Set of elimination based routines for dense linear algebra.

Namespaces

- namespace Protected

Data Structures

- class callLUdivine_small
- class callLUdivine_small< double >
- class callLUdivine_small< float >
- class CharpolyFailed
- class CheckerImplem_charpoly
- class CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >
- class CheckerImplem_Det
- class CheckerImplem_invert
- class CheckerImplem_PLUQ
- class Failure

  A precondtion failed.
- struct rns_double
- struct rns_double_elt
- struct rns_double_elt_cstptr
- struct rns_double_elt_ptr
- struct rns_double_extended
- class RNSInteger
- class RNSIntegerMod
- class rnsRandIter

Typedefs

- template<class Field>
  using Checker_PLUQ = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_Det = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_invert = FFLAS::Checker_Empty<Field>
- template<class Field, class Polynomial>
  using Checker_charpoly = FFLAS::Checker_Empty<Field>
- template<class Field>
  using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
- template<class Field>
  using ForceCheck_Det = CheckerImplem_Det<Field>
- template<class Field>
  using ForceCheck_invert = CheckerImplem_invert<Field>
- template<class Field, class Polynomial>
  using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>

Functions

- void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)

  Conversion of a permutation from LAPACK format to Math format.
- void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)

  Conversion of a permutation from Maths format to LAPACK format.
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field, class Cut, class Param>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field>
  void MonotonicApplyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t R)

  Apply a R-monotonically increasing permutation P, to the matrix A.
- template<class Field>
  void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)

  Solve the system $AX = B$ or $XA = B$.
- template<class Field>
  Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Solve the system A X = B or X A = B.

- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)

  Square system solver.
- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Rectangular system solver.
- template<class Field>
  void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)

  Compute the inverse of a triangular matrix.
- template<class Field>
  void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)
- template<class Field>
  void ftrtrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Compute the product of two triangular matrices of opposite shape.
- template<class Field>
  void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)

  Solve a triangular system with a triangular right hand side of the same shape.
- template<class Field>
  void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)

  Solve a triangular system in a symmetric sum: find B upper/lower triangular such that A^T B + B^T A = C where C is symmetric.
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)

  Triangular factorization of symmetric matrices.
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOL
- template<class Field, class Cut, class Param>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_
- template<class Field>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)

  Triangular factorization of symmetric matrices.
- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  Compute a PLUQ factorization of the given matrix.
- template<class Field>
  size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)
- template<class Field, class Cut, class Param>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)
- template<class Field>
  size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)

  Compute the CUP or PLE factorization of the given matrix.
- template<class Field>
  size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Compute the Column Echelon form of the input matrix in-place.
- template<class Field>
  size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Compute the Row Echelon form of the input matrix in-place.
- template<class Field>
  size_t pRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Compute the Reduced Column Echelon form of the input matrix in-place.
- template<class Field>
  size_t pReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

Compute the Reduced Row Echelon form of the input matrix in-place.

- template<class Field>
  size_t pReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, int &nullity)

  Invert the given matrix in place or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert2 (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.
- template<class PolRing>
  std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class PolRing>
  PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class PolRing>
  PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class Field, class Polynomial>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial, class RandIter>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, RandIter &G)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr v, const size_t incv)

  Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, ..., A^N v)$.
- template<class Field>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Computes the rank of the given matrix using a PLUQ factorization.
- template<class Field>
  size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0)

- template<class Field, class PSHelper>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH)
- template<class Field>
  bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Returns true if the given matrix is singular.
- template<class Field>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P=NULL, size_t *Q=NULL)

  Returns the determinant of the given square matrix.
- template<class Field>
  Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)
- template<class Field, class PSHelper>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)
- template<class Field>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)

  Solves a linear system AX = b using PLUQ factorization.
- template<class Field, class PSHelper>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)
- template<class Field>
  Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)
- template<class Field>
  *void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)

  Solve L X = B or X L = B in place.
- template<class Field>
  size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &NS, size_t &ldn, size_t &NSdim)

  Computes a basis of the Left/Right nullspace of the matrix A.
- template<class Field>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the row rank profile of A.
- template<class Field>
  size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- template<class Field>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the column rank profile of A.
- template<class Field>
  size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)

- template<class Field, class PSHelper>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_↩ LU_TAG LuTag)

  Recovers the column/row rank profile from the permutation of an LU decomposition.
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)

  Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.
- template<class Field>
  size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  RowRankProfileSubmatrixIndices.
- template<class Field>
  size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  *Computes the indices of the submatrix r*r X of A whose columns correspond to the column rank profile of A.*
- template<class Field>
  size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  *Computes the r*r submatrix X of A, by picking the row rank profile rows of A.*
- template<class Field>
  size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.
- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)

  Extracts a triangular matrix from a compact storage A=L\U of rank R.
- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)

  Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by RowEchelonForm or Column↩ EchelonForm.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Cleans up a compact storage A=L\U obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.
- template<class Field>
  void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK↩ _LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.

- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG Lu↩ Tag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.
- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_↩ LU_TAG LuTag=FfpackSlabRecursive)

  Cleans up a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelon↩ Form with transform = true.
- template<class Field>
  void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.
- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

  Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.
- template<class Field>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.
- size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.
- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size↩ _t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.
- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size↩ _t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.
- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.
- size_t * TInverter (size_t *T, size_t r)

- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, size_t *MU, size_t *ML)
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, const typename Field::Element_ptr Xu, size_t ldu, size_t NbBlocksU, size_t *Ku, size_t *Tu, size_t *MU, const typename Field::Element_ptr Xl, size_t ldl, size_t NbBlocksL, size_t *Kl, size_t *Tl, size_t *ML, typename Field::Element_ptr B, size_t t, size_t ldb, typename Field::Element_ptr C, size_t ldc)

  productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix
- template<class Field>
  Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)

  LQUPtoInverseOfFullRankMinor.
- template<class Field>
  void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)

  Solve L X = B or X L = B in place.
- template<class Field>
  void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)
- template<class Field>
  void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)
- size_t * TInverter (const size_t *T, size_t r)
- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, const size_t *MU, const size_t *ML)
- template<class Field>
  Field::Element_ptr expandLCRE (const Field &Fi, size_t N, size_t s, size_t r, size_t *R, size_t i, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tuinv, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tlinv, typename Field::Element_ptr CRE, size_t ldcre)

  Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

  Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.
- template<class Field, class Polynomial>
  std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class Field>
  Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)
- FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)

- template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)
- template<class PSHelper>
  FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)
- template<class PSHelper>
  Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)
- template<class Field>
  bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)
- template<class Field>
  size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)
- template<class Field>
  size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)
- template<class Field>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Sequential seq, size_t threshold)
- template<class Field, class Cut, class Param>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)
- template<class Field>
  size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)
- template<class Field>
  void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)
- template<class Field>
  size_t LUdivine_gauss (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- template<class Field>
  size_t LUdivine_small (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- template<class Field>
  size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)

- template<> size_t LUdivine (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)
- template<class Field>
  void MonotonicCompress (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)
- template<class Field>
  void MonotonicCompressMorePivots (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t rowstomove, const size_t lenP)
- template<class Field>
  void MonotonicCompressCycles (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t lenP)
- template<class Field>
  void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)
- template<class Field>
  void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)
- template<class Field>
  void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field, class Cut, class Param>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class T>
  void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field, class Cut, class Param>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class T>
  void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)

Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.

- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in MathP as a MathPermutation format.

- void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)

  Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.

- void cyclic_shift_mathPerm (size_t *P, const size_t s)
- template<class Field>
  void cyclic_shift_row_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  void cyclic_shift_row (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<typename T>
  void cyclic_shift_row (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  void cyclic_shift_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<typename T>
  void cyclic_shift_col (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  size_t PLUQ_basecaseV3 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t PLUQ_basecaseCrout (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)
- template<class Cut, class Param>
  size_t PLUQ (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)
- template<class Field>
  void threads_fgemm (const size_t m, const size_t n, const size_t r, int nbthreads, size_t *W1, size_t *W2, size_t *W3, size_t gamma)
- template<class Field>
  void threads_ftrsm (const size_t m, const size_t n, int nbthreads, size_t *t1, size_t *t2)
- template<class Field>
  size_t PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &PSHelper)
- template<> rns_double_elt_ptr fflas_const_cast (rns_double_elt_cstptr x)
- template<> rns_double_elt_cstptr fflas_const_cast (rns_double_elt_ptr x)
- template<typename Base_t>
  void cyclic_shift_row_col (Base_t *A, size_t m, size_t n, size_t lda)
- template INST_OR_DECL void cyclic_shift_row (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)
- template INST_OR_DECL void cyclic_shift_col (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)

- template INST_OR_DECL void applyP (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, FFLAS_ELT *A, const size_t lda, const size_t *P)
- template INST_OR_DECL void fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL FFLAS_ELT * fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL void ftrtri (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t threshold)
- template INST_OR_DECL void trinv_left (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *L, const size_t ldl, FFLAS_ELT *X, const size_t ldx)
- template INST_OR_DECL void ftrtrm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL size_t PLUQ (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)
- template INST_OR_DECL size_t LUdivine (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag, const size_t cutoff)
- template INST_OR_DECL size_t LUdivine_small (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t LUdivine_gauss (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t RowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ReducedRowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ColumnEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ReducedColumnEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL FFLAS_ELT * Invert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, int &nullity)
- template INST_OR_DECL FFLAS_ELT * Invert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, int &nullity)
- template INST_OR_DECL FFLAS_ELT * Invert2 (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, int &nullity)
- template INST_OR_DECL std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &charp, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, const size_t degree)

- template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, const size_t degree)
- template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp, const size_t N, FFLAS_ELT *A, const size_t lda, const FFPACK_↩ CHARPOLY_TAG CharpTag, const size_t degree)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MatVecMinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *V, const size_t incv)
- template INST_OR_DECL size_t KrylovElim (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const size_t deg, size_t *iterates, size_↩ t *inviterates, const size_t maxit, size_t virt)
- template INST_OR_DECL size_t SpecRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t deg, size_t *rankProfile)
- template INST_OR_DECL size_t Rank (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL bool IsSingular (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL FFLAS_ELT & Det (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT &det, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)
- template INST_OR_DECL FFLAS_ELT & Det (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT &det, const size_t N, FFLAS_ELT *A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &parH, size_t *P, size_t *Q)
- template INST_OR_DECL FFLAS_ELT * Solve (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *x, const int incx, const FFLAS_ELT *b, const int incb)
- template INST_OR_DECL void solveLB (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *L, const size_t ldl, const size_t *Q, FFLAS_ELT *B, const size_t ldb)
- template INST_OR_DECL void solveLB2 (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *L, const size_t ldl, const size_t *Q, FFLAS_ELT *B, const size_t ldb)
- template INST_OR_DECL void RandomNullSpaceVector (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t incX)
- template INST_OR_DECL size_t NullSpaceBasis (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&NS, size_t &ldn, size_t &NSdim)
- template INST_OR_DECL size_t RowRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ColumnRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size↩ _t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t RowRankProfileSubmatrixIndices (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
- template INST_OR_DECL size_t ColRankProfileSubmatrixIndices (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
- template INST_OR_DECL size_t RowRankProfileSubmatrix (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&X, size_t &R)

- template INST_OR_DECL size_t ColRankProfileSubmatrix (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&X, size_t &R)
- template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors)
- template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, FFLAS_ELT *A, const size_t lda, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, FFLAS_ELT *A, const size_t lda, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL FFLAS_ELT * LQUPtoInverseOfFullRankMinor (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t rank, FFLAS_ELT *A_factors, const size_t lda, const size_t *QtPointer, FFLAS_ELT *X, const size_t ldx)
- template<class T, class CT = const T>
  T fflas_const_cast (CT x)
- Failure & failure ()
- template<class T>
  bool isOdd (const T &a)
- bool isOdd (const float &a)
- bool isOdd (const double &a)
- template<class Field, class RandIter>
  Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random non-zero Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)

  Random non-zero Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix.
- template<class Field>
  Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)

Random Matrix.

- template<class Field, class RandIter>
  Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Triangular Matrix.
- template<class Field>
  Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda)

  Random Triangular Matrix.
- size_t RandInt (size_t a, size_t b)
- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrix (const Field &F, size_t n, bool nonsingular, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Symmetric Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed rank.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed rank.
- size_t * RandomIndexSubset (size_t N, size_t R, size_t *P)

  Pick uniformly at random a sequence of R distinct elements from the set $\{0, \ldots, N-1\}$ using Knuth's shuffle.
- size_t * RandomPermutation (size_t N, size_t *P)

  Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.
- void RandomRankProfileMatrix (size_t M, size_t N, size_t R, size_t *rows, size_t *cols)

  Pick uniformly at random an R-subpermutation of dimension M x N : a matrix with only R non-zeros equal to one, in a random rook placement.
- void swapval (size_t k, size_t N, size_t *P, size_t val)
- void RandomSymmetricRankProfileMatrix (size_t N, size_t R, size_t *rows, size_t *cols)

  Pick uniformly at random a symmetric R-subpermutation of dimension N x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.
- void RandomLTQSRankProfileMatrix (size_t n, size_t r, size_t t, size_t *rows, size_t *cols)
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)

  Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)

  Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.
- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)

  Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an n x n symmetric matrix with random entries and rank r.
- template<class Field>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)

  Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an n x n symmetric matrix with random entries and rank r.

- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda)

  Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field>
  Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (Field &F, size_t n, size_t r, size_t t, typename Field::Element_ptr A, size_t lda, RandIter &G)
- template<typename Field>
  Field * chooseField (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (Givaro::Integer q, uint64_t b, uint64_t seed)

11.21.1 Detailed Description

Finite **F**ield **PACK** Set of elimination based routines for dense linear algebra.

This namespace enlarges the set of BLAS routines of the class FFLAS, with higher level routines based on elimination.

11.21.2 Typedef Documentation

11.21.2.1 Checker_PLUQ

```
template<class Field>
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

11.21.2.2 Checker_Det

```
template<class Field>
using Checker_Det = FFLAS::Checker_Empty<Field>
```

11.21.2.3 Checker_invert

```
template<class Field>
using Checker_invert = FFLAS::Checker_Empty<Field>
```

11.21.2.4 Checker_charpoly

```
template<class Field, class Polynomial>
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

11.21.2.5 ForceCheck_PLUQ

```
template<class Field>
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

11.21.2.6 ForceCheck_Det

```
template<class Field>
using ForceCheck_Det = CheckerImplem_Det<Field>
```

11.21.2.7 ForceCheck_invert

```
template<class Field>
using ForceCheck_invert = CheckerImplem_invert<Field>
```

11.21.2.8 ForceCheck_charpoly

```
template<class Field, class Polynomial>
using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>
```

11.21.3 Function Documentation

11.21.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N)  [inline]
```

Conversion of a permutation from LAPACK format to Math format.

11.21.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N)  [inline]
```

Conversion of a permutation from Maths format to LAPACK format.

11.21.3.3 applyP() [1/4]

```
template<class Field>
void applyP (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const FFLAS::FFLAS_TRANSPOSE Trans,
            const size_t M,
            const size_t ibeg,
            const size_t iend,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t * P)  [inline]
```

Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.

Parameters

in,out	*P1*	a LAPACK permutation of size N
	P2	a LAPACK permutation of size N-R

Applies a permutation P to the matrix A. Apply a permutation P, stored in the LAPACK format (a sequence of transpositions) between indices ibeg and iend of P to (iend-ibeg) vectors of size M stored in A (as column for NoTrans and rows for Trans). Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

Parameters

F	base field
Side	decides if rows (FflasLeft) or columns (FflasRight) are permuted
Trans	decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)
M	size of the elements to permute
ibeg	first index to consider in P
iend	last index to consider in P
A	input matrix
lda	leading dimension of A
P	permutation in LAPACK format
psh	(optional): a sequential or parallel helper, to choose between sequential or parallel execution

Warning

not sure the submatrix is still a permutation and the one we expect in all cases... examples for iend=2, ibeg=1 and P=[2,2,2]

11.21.3.4 applyP() [2/4]

```
template<class Field>
void applyP (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const FFLAS::FFLAS_TRANSPOSE Trans,
             const size_t m,
             const size_t ibeg,
             const size_t iend,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t * P,
             const FFLAS::ParSeqHelper::Sequential seq)  [inline]
```

11.21.3.5 applyP() [3/4]

```
template<class Field, class Cut, class Param>
void applyP (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const FFLAS::FFLAS_TRANSPOSE Trans,
             const size_t m,
             const size_t ibeg,
             const size_t iend,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t * P,
             const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.21.3.6 MonotonicApplyP()

```
template<class Field>
void MonotonicApplyP (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const FFLAS::FFLAS_TRANSPOSE Trans,
             const size_t M,
             const size_t ibeg,
             const size_t iend,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t * P,
             const size_t R)  [inline]
```

Apply a R-monotonically increasing permutation P, to the matrix A.

MonotonicApplyP Apply a permutation defined by the first R entries of the vector P (the pivots).

The permutation represented by P is defined as follows:

- the first R values of P is a LAPACK reprensentation (a sequence of transpositions)
- the remaining iend-ibeg-R values of the permutation are in a monotonically increasing progression Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

Parameters

F	base field
Side	selects if it is a row (FflasLeft) or column (FflasRight) permutation
Trans	inverse permutation (FflasTrans/NoTrans)
M	
ibeg	
iend	
A	input matrix
lda	leading dimension of A
P	LAPACK permuation
R	first values of P

The non pivot elements, are located in montonically increasing order.

11.21.3.7 fgetrs() [1/4]

```
template<class Field>
void fgetrs (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t R,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t * P,
             const size_t * Q,
             typename Field::Element_ptr B,
             const size_t ldb,
             int * info)
```

Solve the system $AX = B$ or $XA = B$.

Solving using the PLUQ decomposition of A already computed inplace with PLUQ (FFLAS::FflasNonUnit). Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

Parameters

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of B
N	col dimension of B
R	rank of A
A	input matrix
lda	leading dimension of A
P	row permutation of the PLUQ decomposition of A
Q	column permutation of the PLUQ decomposition of A
B	Right/Left hand side matrix. Initially stores B, finally stores the solution X.
ldb	leading dimension of B
info	Success of the computation: 0 if successfull, >0 if system is inconsistent

11.21.3.8 fgetrs() [2/4]

```
template<class Field>
Field::Element_ptr fgetrs (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            const size_t R,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t * P,
            const size_t * Q,
            typename Field::Element_ptr X,
            const size_t ldx,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            int * info)
```

Solve the system A X = B or X A = B.

Solving using the PLUQ decomposition of A already computed inplace with PLUQ(FFLAS::FflasNonUnit). Version for A rectangular. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

Parameters

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of A
N	col dimension of A
NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
R	rank of A
A	input matrix
lda	leading dimension of A
P	row permutation of the PLUQ decomposition of A
Q	column permutation of the PLUQ decomposition of A
X	solution matrix
ldx	leading dimension of X
B	Right/Left hand side matrix.
ldb	leading dimension of B
info	Succes of the computation: 0 if successfull, >0 if system is inconsistent

11.21.3.9 fgesv() [1/4]

```
template<class Field>
size_t fgesv (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
```

```
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb,
            int * info)
```

Square system solver.

Parameters

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of B
N	col dimension of B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
info	Success of the computation: 0 if successfull, >0 if system is inconsistent

Returns

the rank of the system

Solve the system A X = B or X A = B. Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

11.21.3.10 fgesv() [2/4]

```
template<class Field>
size_t fgesv (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            int * info)
```

Rectangular system solver.

Parameters

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of A
N	col dimension of A

Parameters

NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
X	
ldx	
info	Success of the computation: 0 if successfull, >0 if system is inconsistent

Returns

the rank of the system

Solve the system A X = B or X A = B. Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

11.21.3.11 ftrtri() [1/2]

```
template<class Field>
void ftrtri (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t threshold = __FFLASFFPACK_FTRTRI_THRESHOLD)
```

Compute the inverse of a triangular matrix.

Parameters

F	base field
Uplo	whether the matrix is upper or lower triangular
Diag	whether the matrix is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

11.21.3.12 trinv_left() [1/2]

```
template<class Field>
void trinv_left (
             const Field & F,
             const size_t N,
             typename Field::ConstElement_ptr L,
             const size_t ldl,
             typename Field::Element_ptr X,
             const size_t ldx)
```

11.21.3.13 ftrtrm() [1/2]

```
template<class Field>
void ftrtrm (
             const Field & F,
             const FFLAS::FFLAS_SIDE side,
             const FFLAS::FFLAS_DIAG diag,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda)
```

Compute the product of two triangular matrices of opposite shape.

Product UL or LU of the upper, resp lower triangular matrices U and L stored one above the other in the square matrix A.

Parameters

F	base field
Side	set to FflasLeft to compute the product UL, FflasRight to compute LU
diag	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

11.21.3.14 ftrstr()

```
template<class Field>
void ftrstr (
             const Field & F,
             const FFLAS::FFLAS_SIDE side,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diagA,
             const FFLAS::FFLAS_DIAG diagB,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
             const size_t ldb,
             const size_t threshold = __FFLASFFPACK_FTRSTR_THRESHOLD)  [inline]
```

Solve a triangular system with a triangular right hand side of the same shape.

Parameters

F	base field
Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
Uplo	whether the matrix A is upper or lower triangular
diag1	whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)
diag2	whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)
N	order of the input matrices
A	the input matrix to be inverted (U1 or L1)
lda	leading dimension of A
B	the input right hand side (U2 or L2)
ldb	leading dimension of B

11.21.3.15 ftrssyr2k()

```
template<class Field>
void ftrssyr2k (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diagA,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr B,
             const size_t ldb,
             const size_t threshold = __FFLASFFPACK_FTRSSYR2K_THRESHOLD)  [inline]
```

Solve a triangular system in a symmetric sum: find B upper/lower triangular such that A^T B + B^T A = C where C is symmetric.

C is overwritten by B.

Parameters

	F	base field
	Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
	Uplo	whether the matrix A is upper or lower triangular
	diagA	whether the matrix A is unit diagonal (FflasUnit/NoUnit)
	N	order of the input matrices
	A	the input matrix
	lda	leading dimension of A
in,out	*B*	the input right hand side where the output is written
	ldb	leading dimension of B

11.21.3.16 fsytrf() [1/3]

```
template<class Field>
bool fsytrf (
             const Field & F,
             const FFLAS::FFLAS_UPLO UpLo,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD)
```

Triangular factorization of symmetric matrices.

Parameters

	F	The computation domain
	UpLo	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	N	order of the matrix A
in,out	*A*	input matrix
	lda	leading dimension of A

Returns

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A: $A = L \times D \times L^T$ if UpLo = FflasLower or $A = U^T \times D \times U$ otherwise. D is a diagonal matrix. The matrices L and U are unit diagonal lower (resp. upper) triangular and overwrite the input matrix A. The matrix D is stored on the diagonal of A, as the diagonal of L or U is known to be all ones. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

11.21.3.17 fsytrf() [2/3]

```
template<class Field>
bool fsytrf (
              const Field & F,
              const FFLAS::FFLAS_UPLO UpLo,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              const FFLAS::ParSeqHelper::Sequential seq,
              const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD)  [inline]
```

11.21.3.18 fsytrf() [3/3]

```
template<class Field, class Cut, class Param>
bool fsytrf (
              const Field & F,
              const FFLAS::FFLAS_UPLO UpLo,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              const FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
              const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD)  [inline]
```

11.21.3.19 fsytrf_nonunit() [1/3]

```
template<class Field>
bool fsytrf_nonunit (
              const Field & F,
              const FFLAS::FFLAS_UPLO UpLo,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              typename Field::Element_ptr D,
              const size_t incD,
              const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD)
```

Triangular factorization of symmetric matrices.

Parameters

	F	The computation domain
	UpLo	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor

	N	order of the matrix A
in,out	*A*	input matrix
in,out	*D*	
	lda	leading dimension of A

Returns

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A: $A = L \times Dinv \times L^T$ if UpLo = FflasLower or $A = U^T \times D \times U$ otherwise. D is a diagonal matrix. The matrices L and U are lower (resp. upper) triangular and overwrite the input matrix A. The matrix D need to be stored separately, as the diagonal of L or U are not unit. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

11.21.3.20 PLUQ() [1/6]

```
template<class Field>
size_t PLUQ (
             const Field & F,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q)  [inline]
```

Compute a PLUQ factorization of the given matrix.

Return its rank. The permutations P and Q are represented using LAPACK's convention.

Parameters

F	base field
Diag	whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
M	matrix row dimension
N	matrix column dimension
A	input matrix
lda	leading dimension of A
P	the row permutation
Q	the column permutation

Returns

the rank of A

Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles* , ISSAC'13, 2013

11.21.3.21 pPLUQ()

```
template<class Field>
size_t pPLUQ (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t * P,
              size_t * Q)  [inline]
```

11.21.3.22 PLUQ() [2/6]

```
template<class Field>
size_t PLUQ (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t * P,
              size_t * Q,
              const FFLAS::ParSeqHelper::Sequential & PSHelper,
              size_t BCThreshold = __FFLASFFPACK_PLUQ_THRESHOLD)  [inline]
```

11.21.3.23 PLUQ() [3/6]

```
template<class Field, class Cut, class Param>
size_t PLUQ (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t * P,
              size_t * Q,
              const FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper)
```

11.21.3.24 LUdivine() [1/4]

```
template<class Field>
size_t LUdivine (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const FFLAS::FFLAS_TRANSPOSE trans,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
```

```
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive,
            const size_t cutoff = __FFLASFFPACK_LUDIVINE_THRESHOLD)
```

Compute the CUP or PLE factorization of the given matrix.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

Parameters

F	base field
Diag	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
trans	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
M	matrix row dimension
N	matrix column dimension
A	input matrix
lda	leading dimension of A
P	the factor of CUP or PLE
Q	a permutation indicating the pivot position in the echelon form C or E in its first r positions
LuTag	flag for setting the earling termination if the matrix is singular
cutoff	threshold to basecase

Returns

the rank of A

Bibliography

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition* , J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

11.21.3.25 ColumnEchelonForm() [1/3]

```
template<class Field>
size_t ColumnEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform = false,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation A = [M \ V] such that AU = C is a column echelon decomposition of A, with U = P^T [V] and C = M + Q [Ir] [0 In-r] [0] If LuTag == FfpackTileRecursive then A = [N \ V] such that the same holds with M = Q N

Qt = Q^T If transform=false, the matrix V is not computed. See also test-colechelon for an example of use

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix
	lda	leading dimension of A
	P	the column permutation
	Qt	the row position of the pivots in the echelon form
	transform	decides whether V is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.26 pColumnEchelonForm()

```
template<class Field>
size_t pColumnEchelonForm (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Qt,
             bool transform = false,
             size_t numthreads = 0,
             const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.27 ColumnEchelonForm() [2/3]

```
template<class Field, class PSHelper>
size_t ColumnEchelonForm (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Qt,
             const bool transform,
             const FFPACK_LU_TAG LuTag,
             const PSHelper & psH)  [inline]
```

11.21.3.28 RowEchelonForm() [1/3]

```
template<class Field>
size_t RowEchelonForm (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
```

```
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform = false,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Compute the Row Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation A = [L \ M] such that X A = R is a row echelon decomposition of A, with X = [L 0] P and R = M + [Ir 0] Q^T [In-r] If LuTag == FfpackTileRecursive then A = [L \ N] such that the same holds with M = N Q^T Qt = Q^T If transform=false, the matrix L is not computed. See also test-rowechelon for an example of use

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	the input matrix
	lda	leading dimension of A
	P	the row permutation
	Qt	the column position of the pivots in the echelon form
	transform	decides whether L is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.29 pRowEchelonForm()

```
template<class Field>
size_t pRowEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform = false,
            size_t numthreads = 0,
            const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.30 RowEchelonForm() [2/3]

```
template<class Field, class PSHelper>
size_t RowEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag,
            const PSHelper & psH)  [inline]
```

11.21.3.31 ReducedColumnEchelonForm() [1/3]

```
template<class Field>
size_t ReducedColumnEchelonForm (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Qt,
             const bool transform = false,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Compute the Reduced Column Echelon form of the input matrix in-place.

After the computation A = [V] such that AX = R is a reduced col echelon [M 0] decomposition of A, where X = P^T [V] and R = Q [Ir] [0 In-r] [M 0] Qt = Q^T If transform=false, the matrix X is not computed and the matrix A = R

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix
	lda	leading dimension of A
	P	the column permutation
	Qt	the row position of the pivots in the echelon form
	transform	decides whether X is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.32 pReducedColumnEchelonForm()

```
template<class Field>
size_t pReducedColumnEchelonForm (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Qt,
             const bool transform = false,
             size_t numthreads = 0,
             const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.33 ReducedColumnEchelonForm() [2/3]

```
template<class Field, class PSHelper>
size_t ReducedColumnEchelonForm (
             const Field & F,
```

```
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag,
            const PSHelper & psH)  [inline]
```

11.21.3.34 ReducedRowEchelonForm() [1/3]

```
template<class Field>
size_t ReducedRowEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform = false,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Compute the Reduced Row Echelon form of the input matrix in-place.

After the computation A = [V1 M] such that X A = R is a reduced row echelon [V2 0] decomposition of A, where X = [V1 0] P and R = [Ir M] Q^T [V2 In-r] [0] Qt = Q^T If transform=false, the matrix X is not computed and the matrix A = R

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix
	lda	leading dimension of A
	P	the row permutation
	Qt	the column position of the pivots in the echelon form
	transform	decides whether X is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.35 pReducedRowEchelonForm()

```
template<class Field>
size_t pReducedRowEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform = false,
            size_t numthreads = 0,
            const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.36 ReducedRowEchelonForm() [2/3]

```
template<class Field, class PSHelper>
size_t ReducedRowEchelonForm (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag,
            const PSHelper & psH)  [inline]
```

11.21.3.37 Invert() [1/4]

```
template<class Field>
Field::Element_ptr Invert (
            const Field & F,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            int & nullity)
```

Invert the given matrix in place or computes its nullity if it is singular.

An inplace $2n^3$ algorithm is used.

Parameters

	F	The computation domain
	M	order of the matrix
in,out	*A*	input matrix ($M \times M$)
	lda	leading dimension of A
	nullity	dimension of the kernel of A

Returns

pointer to A and $A \leftarrow A^{-1}$

11.21.3.38 Invert() [2/4]

```
template<class Field>
Field::Element_ptr Invert (
            const Field & F,
            const size_t M,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx,
            int & nullity)
```

Invert the given matrix or computes its nullity if it is singular.

Precondition

X is preallocated and should be large enough to store the $m \times m$ matrix A.

Parameters

	F	The computation domain
	M	order of the matrix
in	*A*	input matrix ($M \times M$)
	lda	leading dimension of A
out	*X*	this is the inverse of A if A is invertible (non NULL and nullity $= 0$). It is untouched otherwise.
	ldx	leading dimension of X
	nullity	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

11.21.3.39 Invert2() [1/2]

```
template<class Field>
Field::Element_ptr Invert2 (
            const Field & F,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx,
            int & nullity)
```

Invert the given matrix or computes its nullity if it is singular.

An $2n^3 f$ algorithm is used. This routine can be % faster than FFPACK::Invert but is not totally inplace.

Precondition

X is preallocated and should be large enough to store the $m \times m$ matrix A.

Warning

A is overwritten here !

Bug not tested.

Parameters

	F	the computation domain
	M	order of the matrix
in,out	*A*	input matrix ($M \times M$). On output, A is modified and represents a "psycological" factorisation LU.
	lda	leading dimension of A
out	*X*	this is the inverse of A if A is invertible (non NULL and nullity $= 0$). It is untouched otherwise.
	ldx	leading dimension of X
	nullity	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

Todo this init is not all necessary (done after ftrtri)

11.21.3.40 CharPoly() [1/8]

```
template<class PolRing>
std::list< typename PolRing::Element > & CharPoly (
             const PolRing & R,
             std::list< typename PolRing::Element > & charp,
             const size_t N,
             typename PolRing::Domain_t::Element_ptr A,
             const size_t lda,
             typename PolRing::Domain_t::RandIter & G,
             const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
             const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD)  [inline]
```

Compute the characteristic polynomial of the matrix A.

Parameters

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of as a list of factors
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of A
	CharpTag	the algorithmic variant
	G	a random iterator (required for the randomized variants LUKrylov and ArithProg)

11.21.3.41 CharPoly() [2/8]

```
template<class PolRing>
PolRing::Element & CharPoly (
             const PolRing & R,
             typename PolRing::Element & charp,
             const size_t N,
             typename PolRing::Domain_t::Element_ptr A,
             const size_t lda,
             typename PolRing::Domain_t::RandIter & G,
             const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
             const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD)  [inline]
```

Compute the characteristic polynomial of the matrix A.

Parameters

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of as a single polynomial
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of A
	CharpTag	the algorithmic variant
	G	a random iterator (required for the randomized variants LUKrylov and ArithProg)

11.21.3.42 CharPoly() [3/8]

```
template<class PolRing>
PolRing::Element & CharPoly (
             const PolRing & R,
             typename PolRing::Element & charp,
             const size_t N,
             typename PolRing::Domain_t::Element_ptr A,
             const size_t lda,
             const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
             const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD)  [inline]
```

Compute the characteristic polynomial of the matrix A.

Parameters

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of as a single polynomial
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of A
	CharpTag	the algorithmic variant

11.21.3.43 MinPoly() [1/4]

```
template<class Field, class Polynomial>
Polynomial & MinPoly (
             const Field & F,
             Polynomial & minP,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda)  [inline]
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector: (v, Av, .., A^kv)

Parameters

	F	the base field
out	*minP*	the minimal polynomial of A
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$)
	lda	leading dimension of A

11.21.3.44 MinPoly() [2/4]

```
template<class Field, class Polynomial, class RandIter>
Polynomial & MinPoly (
             const Field & F,
             Polynomial & minP,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             RandIter & G)  [inline]
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector: (v, Av, .., A^kv)

Parameters

	F	the base field
out	*minP*	the minimal polynomial of A
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$)
	lda	leading dimension of A
	G	a random iterator

11.21.3.45 MatVecMinPoly() [1/2]

```
template<class Field, class Polynomial>
Polynomial & MatVecMinPoly (
             const Field & F,
             Polynomial & minP,
             const size_t N,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::ConstElement_ptr v,
             const size_t incv)  [inline]
```

Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, ..., A^N v)$.

Parameters

	F	the base field
out	*minP*	the minimal polynomial of A and v
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$)
	lda	leading dimension of A
	K	an $N \times (N+1)$ matrix containing the vector v on its first row
	ldk	leading dimension of K
	P	[out] (optional) the permutation used in the elimination of the Krylov matrix K

11.21.3.46 Rank() [1/3]

```
template<class Field>
size_t Rank (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda)
```

Computes the rank of the given matrix using a PLUQ factorization.

The input matrix is modified.

Parameters

	F	base field
	M	row dimension of the matrix
	N	column dimension of the matrix
in	*A*	input matrix
	lda	leading dimension of A
	psH	(optional) a ParSeqHelper to choose between sequential and parallel execution

11.21.3.47 pRank()

```
template<class Field>
size_t pRank (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t numthreads = 0)
```

11.21.3.48 Rank() [2/3]

```
template<class Field, class PSHelper>
size_t Rank (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              const PSHelper & psH)
```

11.21.3.49 IsSingular() [1/2]

```
template<class Field>
bool IsSingular (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda)
```

Returns true if the given matrix is singular.

The method is a block elimination with early termination

using LQUP factorization with early termination. If `M != N`, then the matrix is virtually padded with zeros to make it square and it's determinant is zero.

Warning

The input matrix is modified.

Parameters

	F	base field
	M	row dimension of the matrix
	N	column dimension of the matrix.
in,out	*A*	input matrix
	lda	leading dimension of A

11.21.3.50 Det() [1/6]

```
template<class Field>
Field::Element & Det (
            const Field & F,
            typename Field::Element & det,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P = NULL,
            size_t * Q = NULL)  [inline]
```

Returns the determinant of the given square matrix.

The method is a block elimination using PLUQ factorization. The input matrix A is overwritten.

Warning

The input matrix is modified.

Parameters

	F	base field
out	*det*	the determinant of A
	N	the order of the square matrix A.
in,out	*A*	input matrix
	lda	leading dimension of A
	psH	(optional) a ParSeqHelper to choose between sequential and parallel execution
	P,Q	(optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification

11.21.3.51 pDet()

```
template<class Field>
Field::Element & pDet (
             const Field & F,
             typename Field::Element & det,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t numthreads = 0,
             size_t * P = NULL,
             size_t * Q = NULL)  [inline]
```

11.21.3.52 Det() [2/6]

```
template<class Field, class PSHelper>
Field::Element & Det (
             const Field & F,
             typename Field::Element & det,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             const PSHelper & psH,
             size_t * P = NULL,
             size_t * Q = NULL)
```

11.21.3.53 Solve() [1/3]

```
template<class Field>
Field::Element_ptr Solve (
             const Field & F,
             const size_t M,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr x,
             const int incx,
             typename Field::ConstElement_ptr b,
             const int incb)  [inline]
```

Solves a linear system AX = b using PLUQ factorization.

@oaram F base field @oaram M matrix order

Parameters

in	*A*	input matrix
	lda	leading dimension of A
out	*x*	output solution vector
	incx	increment of x
	b	input right hand side of the system
	incb	increment of b

11.21.3.54 Solve() [2/3]

```
template<class Field, class PSHelper>
Field::Element_ptr Solve (
             const Field & F,
             const size_t M,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr x,
             const int incx,
             typename Field::ConstElement_ptr b,
             const int incb,
             PSHelper & psH)
```

11.21.3.55 pSolve()

```
template<class Field>
Field::Element_ptr pSolve (
             const Field & F,
             const size_t M,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr x,
             const int incx,
             typename Field::ConstElement_ptr b,
             const int incb,
             size_t numthreads = 0)  [inline]
```

11.21.3.56 RandomNullSpaceVector() [1/3]

```
template<class Field>
*void RandomNullSpaceVector (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr X,
             const size_t incX)
```

Solve L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗R matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

Parameters

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

11.21.3.57 NullSpaceBasis() [1/2]

```
template<class Field>
size_t NullSpaceBasis (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr & NS,
             size_t & ldn,
             size_t & NSdim)
```

Computes a basis of the Left/Right nullspace of the matrix A.

return the dimension of the nullspace.

Parameters

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified
	lda	leading dimension of A
out	*NS*	output matrix of dimension N x NSdim (allocated here)
out	*ldn*	leading dimension of NS
out	*NSdim*	the dimension of the Nullspace (N-rank(A))

11.21.3.58 RowRankProfile() [1/3]

```
template<class Field>
size_t RowRankProfile (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t *& rkprofile,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Computes the row rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension M x N
	lda	leading dimension of A
out	*rkprofile*	return the rank profile as an array of row indexes, of dimension r=rank(A)
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

11.21.3.59 pRowRankProfile()

```
template<class Field>
size_t pRowRankProfile (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t *& rkprofile,
            size_t numthreads = 0,
            const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.60 RowRankProfile() [2/3]

```
template<class Field, class PSHelper>
size_t RowRankProfile (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t *& rkprofile,
            const FFPACK_LU_TAG LuTag,
            PSHelper & psH)  [inline]
```

11.21.3.61 ColumnRankProfile() [1/3]

```
template<class Field>
size_t ColumnRankProfile (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t *& rkprofile,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Computes the column rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension
	lda	leading dimension of A
out	*rkprofile*	return the rank profile as an array of row indexes, of dimension r=rank(A)
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

11.21.3.62 pColumnRankProfile()

```
template<class Field>
size_t pColumnRankProfile (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t *& rkprofile,
              size_t numthreads = 0,
              const FFPACK_LU_TAG LuTag = FfpackTileRecursive)  [inline]
```

11.21.3.63 ColumnRankProfile() [2/3]

```
template<class Field, class PSHelper>
size_t ColumnRankProfile (
              const Field & F,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t *& rkprofile,
              const FFPACK_LU_TAG LuTag,
              PSHelper & psH)  [inline]
```

11.21.3.64 RankProfileFromLU()

```
void RankProfileFromLU (
              const size_t * P,
              const size_t N,
              const size_t R,
              size_t * rkprofile,
              const FFPACK_LU_TAG LuTag)  [inline]
```

Recovers the column/row rank profile from the permutation of an LU decomposition.

Works with both the CUP/PLE decompositions (obtained by LUdivine) or the PLUQ decomposition. Assumes that the output vector containing the rank profile is already allocated.

Parameters

	P	the permutation carrying the rank profile information
	N	the row/col dimension for a row/column rank profile
	R	the rank of the matrix
out	*rkprofile*	return the rank profile as an array of indices
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.65 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t LSm,
            const size_t LSn,
            const size_t * P,
            const size_t * Q,
            size_t * RRP,
            size_t * CRP)  [inline]
```

Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.

Only works with the PLUQ decomposition Assumes that the output vectors containing the rank profiles are already allocated.

Parameters

P	the permutation carrying the rank profile information
M	the row dimension of the initial matrix
N	the column dimension of the initial matrix
R	the rank of the initial matrix
LSm	the row dimension of the leading submatrix considered
LSn	the column dimension of the leading submatrix considered
P	the row permutation of the PLUQ decomposition
Q	the column permutation of the PLUQ decomposition
RRP	return the row rank profile of the leading submatrix

Returns

the rank of the LSm x LSn leading submatrix

A is modified

Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles* , ISSAC'13.

11.21.3.66 RowRankProfileSubmatrixIndices() [1/2]

```
template<class Field>
size_t RowRankProfileSubmatrixIndices (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t *& rowindices,
            size_t *& colindices,
            size_t & R)
```

RowRankProfileSubmatrixIndices.

Computes the indices of the submatrix r∗r X of A whose rows correspond to the row rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation. A is modified

Returns

R

11.21.3.67 ColRankProfileSubmatrixIndices() [1/2]

```
template<class Field>
size_t ColRankProfileSubmatrixIndices (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t *& rowindices,
            size_t *& colindices,
            size_t & R)
```

Computes the indices of the submatrix r∗r X of A whose columns correspond to the column rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation.

Warning

A is modified

Returns

R

11.21.3.68 RowRankProfileSubmatrix() [1/2]

```
template<class Field>
size_t RowRankProfileSubmatrix (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr & X,
             size_t & R)
```

Computes the r∗r submatrix X of A, by picking the row rank profile rows of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension M x N
	lda	leading dimension of A
out	*X*	the output matrix
out	*R*	list of indices

A is not modified X is allocated during the computation.

Returns

R

11.21.3.69 ColRankProfileSubmatrix() [1/2]

```
template<class Field>
size_t ColRankProfileSubmatrix (
             const Field & F,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr & X,
             size_t & R)
```

Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension M x N
	lda	leading dimension of A
out	*X*	the output matrix
out	*R*	list of indices

A is not modified X is allocated during the computation.

Returns

R

11.21.3.70 getTriangular() [1/2]

```
template<class Field>
void getTriangular (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t M,
             const size_t N,
             const size_t R,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr T,
             const size_t ldt,
             const bool OnlyNonZeroVectors = false)  [inline]
```

Extracts a triangular matrix from a compact storage A=L\U of rank R.

if OnlyNonZeroVectors is false, then T and A have the same dimensions Otherwise, T is R x N if UpLo = FflasUpper, else T is M x R

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored

Todo just one triangular fzero+fassign ?

11.21.3.71 getTriangular() [2/2]

```
template<class Field>
void getTriangular (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t M,
             const size_t N,
             const size_t R,
             typename Field::Element_ptr A,
             const size_t lda)  [inline]
```

Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	diag	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
in,out	*A*	input/output matrix
	lda	leading dimension of A

Todo just one triangular fzero+fassign ?

11.21.3.72 getEchelonForm() [1/2]

```
template<class Field>
void getEchelonForm (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t M,
             const size_t N,
             const size_t R,
             const size_t * P,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr T,
             const size_t ldt,
             const bool OnlyNonZeroVectors = false,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by RowEchelonForm or ColumnEchelonForm.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P. row and column dimension of T are greater or equal to that of A

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.73 getEchelonForm() [2/2]

```
template<class Field>
void getEchelonForm (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t M,
             const size_t N,
             const size_t R,
             const size_t * P,
             typename Field::Element_ptr A,
             const size_t lda,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Cleans up a compact storage A=L\U obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P.

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in,out	*A*	input/output matrix
	lda	leading dimension of A
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.74 getEchelonTransform()

```
template<class Field>
void getEchelonTransform (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t M,
             const size_t N,
             const size_t R,
             const size_t * P,
             const size_t * Q,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr T,
             const size_t ldt,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by Row↩ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.75 getReducedEchelonForm() [1/2]

```
template<class Field>
void getReducedEchelonForm (
            const Field & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr T,
            const size_t ldt,
            const bool OnlyNonZeroVectors = false,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P. row and column dimension of T are greater or equal to that of A

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in	*A*	input matrix
	lda	leading dimension of A
	ldt	leading dimension of T

Parameters

	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored

11.21.3.76 getReducedEchelonForm() [2/2]

```
template<class Field>
void getReducedEchelonForm (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const size_t M,
             const size_t N,
             const size_t R,
             const size_t * P,
             typename Field::Element_ptr A,
             const size_t lda,
             const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Cleans up a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumn↩ EchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P.

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in,out	*A*	input/output matrix
	lda	leading dimension of A
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.77 getReducedEchelonTransform()

```
template<class Field>
void getReducedEchelonTransform (
             const Field & F,
             const FFLAS::FFLAS_UPLO Uplo,
             const size_t M,
             const size_t N,
             const size_t R,
             const size_t * P,
             const size_t * Q,
```

```
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr T,
            const size_t ldt,
            const FFPACK_LU_TAG LuTag = FfpackSlabRecursive)  [inline]
```

Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by Row↩ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	selects Col (FflasLower) or Row (FflasUpper) Echelon Form
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.78 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
            const size_t * P,
            size_t * outPerm)  [inline]
```

Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

11.21.3.79 LTBruhatGen()

```
template<class Field>
size_t LTBruhatGen (
            const Field & Fi,
            const FFLAS::FFLAS_DIAG diag,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Q)  [inline]
```

LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.

Parameters

Fi	base Field
diag	
N	size of A
A	the matrix we search the Bruhat representation
lda	the leading dimension of A
P	a permutation matrix
Q	a permutation matrix

11.21.3.80 getLTBruhatGen() [1/2]

```
template<class Field>
void getLTBruhatGen (
             const Field & Fi,
             const size_t N,
             const size_t r,
             const size_t * P,
             const size_t * Q,
             typename Field::Element_ptr R,
             const size_t ldr)  [inline]
```

GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.

Parameters

Fi	base Field
N	size of the matrix
r	the rank of the matrix
P	a permutation matrix
Q	a permutation matrix
R	the matrix that will contain the rank revealing matrix
ldr	the leading fimension of R

11.21.3.81 getLTBruhatGen() [2/2]

```
template<class Field>
void getLTBruhatGen (
             const Field & Fi,
             const FFLAS::FFLAS_UPLO Uplo,
             const FFLAS::FFLAS_DIAG diag,
             const size_t N,
             const size_t r,
             const size_t * P,
             const size_t * Q,
             typename Field::ConstElement_ptr A,
             const size_t lda,
             typename Field::Element_ptr T,
             const size_t ldt)  [inline]
```

GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.

Parameters

Fi	base Field
Uplo	choose if the procedure return L or U
diag	
N	size of A
r	rank of A
P	permutaion matrix
Q	permutation matrix
A	a bruhat representation
lda	leading dimension of A
T	matrix that will contains L or U
ldt	leading dimension of T

11.21.3.82 LTQSorder()

```
size_t LTQSorder (
            const size_t N,
            const size_t r,
            const size_t * P,
            const size_t * Q)  [inline]
```

LTQSorder This procedure computes the order of quasiseparability of a matrix.

Parameters

N	size of the matrix
r	rank of the matrix
P	permutation matrix
Q	permutation matrix

11.21.3.83 CompressToBlockBiDiagonal()

```
template<class Field>
size_t CompressToBlockBiDiagonal (
            const Field & Fi,
            const FFLAS::FFLAS_UPLO Uplo,
            size_t N,
            size_t s,
            size_t r,
            const size_t * P,
            const size_t * Q,
            typename Field::Element_ptr A,
            size_t lda,
            typename Field::Element_ptr X,
            size_t ldx,
            size_t * K,
            size_t * M,
            size_t * T)  [inline]
```

CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.

Parameters

Fi	base Field
Uplo	chosse if the procedure is based on row or column
N	size of the matrix
s	order of qausiseparability
r	rank
P	permutation matrix
Q	permutation matrix
A	the matrix to compact
lda	leading dimension of A
X	matrix that will stock the representation
ldx	leading dimension of X
K	stock the position of the blocks in A
M	permutation matrix
T	stock the operation done in the procedure

11.21.3.84 ExpandBlockBiDiagonalToBruhat()

```
template<class Field>
void ExpandBlockBiDiagonalToBruhat (
             const Field & Fi,
             const FFLAS::FFLAS_UPLO Uplo,
             size_t N,
             size_t s,
             size_t r,
             typename Field::Element_ptr A,
             size_t lda,
             typename Field::Element_ptr X,
             size_t ldx,
             size_t NbBlocks,
             size_t * K,
             size_t * M,
             size_t * T)  [inline]
```

ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.

Parameters

Fi	base Field
Uplo	chosse if the procedure is based on row or column
N	size of the matrix
s	order of qausiseparability
r	rank
A	the matrix that will sotck the expanded representation
lda	leading dimension of A
X	matrix to expand
ldx	leading dimension of X
K	stock the position of the blocks in A
M	permutation matrix
T	stock the operation done in the procedure

11.21.3.85 Bruhat2EchelonPermutation()

```
void Bruhat2EchelonPermutation (
            size_t N,
            size_t R,
            const size_t * P,
            const size_t * Q,
            size_t * M)  [inline]
```

Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.

Parameters

in	*N*	size of the matrix
in	*R*	rank
in	*P*	permutation Matrix
in	*Q*	permutation Matrix
out	*M*	output permutation matrix

11.21.3.86 TInverter() [1/2]

```
size_t * TInverter (
            size_t * T,
            size_t r)
```

11.21.3.87 ComputeRPermutation() [1/2]

```
template<class Field>
void ComputeRPermutation (
            const Field & Fi,
            size_t N,
            size_t r,
            const size_t * P,
            const size_t * Q,
            size_t * R,
            size_t * MU,
            size_t * ML)
```

11.21.3.88 productBruhatxTS() [1/2]

```
template<class Field>
void productBruhatxTS (
            const Field & Fi,
            size_t N,
            size_t s,
            size_t r,
            const size_t * P,
            const size_t * Q,
            const typename Field::Element_ptr Xu,
            size_t ldu,
```

```
            size_t NbBlocksU,
            size_t * Ku,
            size_t * Tu,
            size_t * MU,
            const typename Field::Element_ptr Xl,
            size_t ldl,
            size_t NbBlocksL,
            size_t * Kl,
            size_t * Tl,
            size_t * ML,
            typename Field::Element_ptr B,
            size_t t,
            size_t ldb,
            typename Field::Element_ptr C,
            size_t ldc)
```

productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix

11.21.3.89 LQUPtoInverseOfFullRankMinor() [1/2]

```
template<class Field>
Field::Element_ptr LQUPtoInverseOfFullRankMinor (
            const Field & F,
            const size_t rank,
            typename Field::Element_ptr A_factors,
            const size_t lda,
            const size_t * QtPointer,
            typename Field::Element_ptr X,
            const size_t ldx)
```

LQUPtoInverseOfFullRankMinor.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal r x r submatrix This procedure efficiently computes the inverse of this minor and puts it into X.

Note

It changes the lower entries of A_factors in the process (NB: unless A was nonsingular and square)

Parameters

F	base field
rank	rank of the matrix.
A_factors	matrix containing the L and U entries of the factorization
lda	leading dimension of A
QtPointer	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
X	desired location for output
ldx	leading dimension of X

11.21.3.90 RandomNullSpaceVector() [2/3]

```
template<class Field>
void RandomNullSpaceVector (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr X,
             const size_t incX)
```

Solve L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗R matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

Parameters

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

11.21.3.91 solveLB() [1/2]

```
template<class Field>
void solveLB (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t R,
             typename Field::Element_ptr L,
             const size_t ldl,
             const size_t * Q,
             typename Field::Element_ptr B,
             const size_t ldb)
```

11.21.3.92 solveLB2() [1/2]

```
template<class Field>
void solveLB2 (
             const Field & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
```

```
            const size_t N,
            const size_t R,
            typename Field::Element_ptr L,
            const size_t ldl,
            const size_t * Q,
            typename Field::Element_ptr B,
            const size_t ldb)
```

11.21.3.93 TInverter() [2/2]

```
size_t * TInverter (
            const size_t * T,
            size_t r)  [inline]
```

11.21.3.94 ComputeRPermutation() [2/2]

```
template<class Field>
void ComputeRPermutation (
            const Field & Fi,
            size_t N,
            size_t r,
            const size_t * P,
            const size_t * Q,
            size_t * R,
            const size_t * MU,
            const size_t * ML)  [inline]
```

11.21.3.95 expandLCRE()

```
template<class Field>
Field::Element_ptr expandLCRE (
            const Field & Fi,
            size_t N,
            size_t s,
            size_t r,
            size_t * R,
            size_t i,
            typename Field::ConstElement_ptr Xu,
            size_t ldu,
            size_t NbBlocksU,
            const size_t * Ku,
            const size_t * Tuinv,
            typename Field::ConstElement_ptr Xl,
            size_t ldl,
            size_t NbBlocksL,
            const size_t * Kl,
            const size_t * Tlinv,
            typename Field::Element_ptr CRE,
            size_t ldcre)  [inline]
```

Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.

11.21.3.96 productBruhatxTS() [2/2]

```
template<class Field>
void productBruhatxTS (
              const Field & Fi,
              size_t N,
              size_t s,
              size_t r,
              size_t t,
              const size_t * P,
              const size_t * Q,
              typename Field::ConstElement_ptr Xu,
              size_t ldu,
              size_t NbBlocksU,
              const size_t * Ku,
              const size_t * Tu,
              const size_t * MU,
              typename Field::ConstElement_ptr Xl,
              size_t ldl,
              size_t NbBlocksL,
              const size_t * Kl,
              const size_t * Tl,
              const size_t * ML,
              typename Field::Element_ptr B,
              size_t ldb,
              const typename Field::Element beta,
              typename Field::Element_ptr D,
              size_t ldd)  [inline]
```

Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.

Parameters

	F	the base field
	N	the order of A
	s	the order of quasiseparability of A
	r	the number of pivots in the left-triangular par of the rank profile matrix of A
	t	the number of columns of B
	P	the row indices of the pivots of A
	Q	the column indices of the pivots of A
	Xu	the compact storage of U: Du blocks in the first s rows, Su blocks in the last s rows
	ldxu	the leading dimension of Xu
	NbBlocksU	the number of diagonal blocks in the compact storage of U
	Ku	the list of starting column positions for each block of the storage of U
	Tu	the folding matrix for the compact storage of U: $Du + TuSu$ is in row echelon form
	Mu	a permutation matrix such that $Mu(Du + TuSu)$ is the U factor of the Bruhat generator
	Xl	the compact storage of L: Dl blocks in the first s columns, Sl blocks in the last s columns
	ldxl	the leading dimension of Xl
	NbBlocksL	the number of diagonal blocks in the compact storage of L
	Kl	the list of starting row positions for each block of the storage of L
	Tl	the folding matrix for the compact storage of L: $Dl + SlTl$ is in column echelon form

Parameters

	Ml	a permutation matrix such that $(Dl + TlSl)Ml$ is the L factor of the Bruhat generator
	B	an $N \times t$ dense matrix
	ldb	leading dimension of B
	beta	scaling constant
in,out	*C*	output matrix
	ldc	leading dimension of C

Bibliography Pernet C. and Storjohann A. *Time and space efficient generators for quasiseparable matrices* , JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010

11.21.3.97 Danilevski()

```
template<class Field, class Polynomial>
std::list< Polynomial > & Danilevski (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda)
```

11.21.3.98 buildMatrix()

```
template<class Field>
Field::Element_ptr buildMatrix (
            const Field & F,
            typename Field::ConstElement_ptr E,
            typename Field::ConstElement_ptr C,
            const size_t lda,
            const size_t * B,
            const size_t * T,
            const size_t me,
            const size_t mc,
            const size_t lambda,
            const size_t mu)
```

Bug is this :

11.21.3.99 CharPoly() [4/8]

```
FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (
            const FFPACK::RNSInteger< FFPACK::rns_double > & F,
            typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp,
            const size_t N,
            typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
            const size_t lda,
            Givaro::ZRing< Givaro::Integer >::RandIter & G,
            const FFPACK_CHARPOLY_TAG CharpTag,
            size_t degree)  [inline]
```

11.21.3.100 CharPoly() [5/8]

```
template<>
Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (
             const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > & R,
             Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & charp,
             const size_t N,
             Givaro::Integer * A,
             const size_t lda,
             Givaro::ZRing< Givaro::Integer >::RandIter & G,
             const FFPACK_CHARPOLY_TAG CharpTag,
             size_t degree)  [inline]
```

11.21.3.101 Det() [3/6]

```
template<class PSHelper>
FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (
             const FFPACK::RNSInteger< FFPACK::rns_double > & F,
             typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & det,
             const size_t N,
             typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
             const size_t lda,
             const PSHelper & psH)  [inline]
```

11.21.3.102 Det() [4/6]

```
template<class PSHelper>
Givaro::Integer & Det (
             const Givaro::ZRing< Givaro::Integer > & F,
             Givaro::Integer & det,
             const size_t N,
             Givaro::Integer * A,
             const size_t lda,
             const PSHelper & psH,
             size_t * P,
             size_t * Q)  [inline]
```

11.21.3.103 fsytrf_BC_Crout()

```
template<class Field>
bool fsytrf_BC_Crout (
             const Field & F,
             const FFLAS::FFLAS_UPLO UpLo,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr Dinv,
             const size_t incDinv)  [inline]
```

11.21.3.104 fsytrf_BC_RL()

```
template<class Field>
size_t fsytrf_BC_RL (
             const Field & F,
             const FFLAS::FFLAS_UPLO UpLo,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr Dinv,
             const size_t incDinv)  [inline]
```

11.21.3.105 fsytrf_UP_RPM_BC_RL()

```
template<class Field>
size_t fsytrf_UP_RPM_BC_RL (
             const Field & F,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr Dinv,
             const size_t incDinv,
             size_t * P)  [inline]
```

11.21.3.106 fsytrf_LOW_RPM_BC_Crout()

```
template<class Field>
size_t fsytrf_LOW_RPM_BC_Crout (
             const Field & F,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr Dinv,
             const size_t incDinv,
             size_t * P)  [inline]
```

11.21.3.107 fsytrf_UP_RPM_BC_Crout()

```
template<class Field>
size_t fsytrf_UP_RPM_BC_Crout (
             const Field & F,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr Dinv,
             const size_t incDinv,
             size_t * P)  [inline]
```

11.21.3.108 fsytrf_UP_RPM()

```
template<class Field>
size_t fsytrf_UP_RPM (
            const Field & Fi,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr Dinv,
            const size_t incDinv,
            size_t * P,
            size_t BCThreshold)  [inline]
```

MathP <-[[I] x P1 |] [I_(N1+R2)] [P2^T] |] x [P3^T] [----------—|--—] [| Q2^T]

Changing [U1 V1 | E1 E21 E22] into [U1 E11 E12 V1 E∗ E∗] [0 | L2 \ U2 V21 V22] [U4 V41 0 V42 V43] [0 | M2 0 0] [U3 0 0 V3] [---—|----------—] [0 0 0] [0 | H1 H21 H22] [0 | U3 V3] [0 | 0] where U4 is the 2R2 x 2R2 matrix formed by interleaving U2, L2^T and H1

11.21.3.109 fsytrf_nonunit() [2/3]

```
template<class Field>
bool fsytrf_nonunit (
            const Field & F,
            const FFLAS::FFLAS_UPLO UpLo,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr Dinv,
            const size_t incDinv,
            FFLAS::ParSeqHelper::Sequential seq,
            size_t threshold)  [inline]
```

11.21.3.110 fsytrf_nonunit() [3/3]

```
template<class Field, class Cut, class Param>
bool fsytrf_nonunit (
            const Field & F,
            const FFLAS::FFLAS_UPLO UpLo,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr Dinv,
            const size_t incDinv,
            FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
            size_t threshold)  [inline]
```

11.21.3.111 fsytrf_RPM()

```
template<class Field>
size_t fsytrf_RPM (
            const Field & F,
            const FFLAS::FFLAS_UPLO UpLo,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t threshold)  [inline]
```

11.21.3.112 getTridiagonal()

```
template<class Field>
void getTridiagonal (
              const Field & F,
              const size_t N,
              const size_t R,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              size_t * P,
              typename Field::Element_ptr T,
              const size_t ldt)  [inline]
```

11.21.3.113 LUdivine_gauss() [1/2]

```
template<class Field>
size_t LUdivine_gauss (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t * P,
              size_t * Q,
              const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

11.21.3.114 LUdivine_small() [1/2]

```
template<class Field>
size_t LUdivine_small (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const FFLAS::FFLAS_TRANSPOSE trans,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
              const size_t lda,
              size_t * P,
              size_t * Q,
              const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

11.21.3.115 LUdivine() [2/4]

```
template<class Field>
size_t LUdivine (
              const Field & F,
              const FFLAS::FFLAS_DIAG Diag,
              const FFLAS::FFLAS_TRANSPOSE trans,
              const size_t M,
              const size_t N,
              typename Field::Element_ptr A,
```

```
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag,
            const size_t cutoff)  [inline]
```

Todo std::swap ?

11.21.3.116 LUdivine() [3/4]

```
template<>
size_t LUdivine (
            const Givaro::Modular< Givaro::Integer > & F,
            const FFLAS::FFLAS_DIAG Diag,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            typename Givaro::Integer * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag,
            const size_t cutoff)  [inline]
```

11.21.3.117 MonotonicCompress()

```
template<class Field>
void MonotonicCompress (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t incA,
            const size_t * MathP,
            const size_t R,
            const size_t maxpiv,
            const size_t rowstomove,
            const std::vector< bool > & ispiv)  [inline]
```

11.21.3.118 MonotonicCompressMorePivots()

```
template<class Field>
void MonotonicCompressMorePivots (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t incA,
            const size_t * MathP,
            const size_t R,
            const size_t rowstomove,
            const size_t lenP)  [inline]
```

11.21.3.119 MonotonicCompressCycles()

```
template<class Field>
void MonotonicCompressCycles (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t incA,
            const size_t * MathP,
            const size_t lenP)  [inline]
```

11.21.3.120 MonotonicExpand()

```
template<class Field>
void MonotonicExpand (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t incA,
            const size_t * MathP,
            const size_t R,
            const size_t maxpiv,
            const size_t rowstomove,
            const std::vector< bool > & ispiv)
```

11.21.3.121 applyP_block()

```
template<class Field>
void applyP_block (
            const Field & F,
            const FFLAS::FFLAS_SIDE Side,
            const FFLAS::FFLAS_TRANSPOSE Trans,
            const size_t M,
            const size_t ibeg,
            const size_t iend,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t * P)  [inline]
```

11.21.3.122 doApplyS()

```
template<class Field>
void doApplyS (
            const Field & F,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)  [inline]
```

11.21.3.123 MatrixApplyS() [1/3]

```
template<class Field>
void MatrixApplyS (
            const Field & F,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)  [inline]
```

11.21.3.124 MatrixApplyS() [2/3]

```
template<class Field>
void MatrixApplyS (
            const Field & F,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4,
            const FFLAS::ParSeqHelper::Sequential seq)  [inline]
```

11.21.3.125 MatrixApplyS() [3/3]

```
template<class Field, class Cut, class Param>
void MatrixApplyS (
            const Field & F,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4,
            const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.21.3.126 PermApplyS()

```
template<class T>
void PermApplyS (
            T * A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)  [inline]
```

11.21.3.127 doApplyT()

```
template<class Field>
void doApplyT (
             const Field & F,
             typename Field::Element_ptr A,
             const size_t lda,
             typename Field::Element_ptr tmp,
             const size_t width,
             const size_t N2,
             const size_t R1,
             const size_t R2,
             const size_t R3,
             const size_t R4)  [inline]
```

11.21.3.128 MatrixApplyT() [1/3]

```
template<class Field>
void MatrixApplyT (
             const Field & F,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t width,
             const size_t N2,
             const size_t R1,
             const size_t R2,
             const size_t R3,
             const size_t R4)  [inline]
```

11.21.3.129 MatrixApplyT() [2/3]

```
template<class Field>
void MatrixApplyT (
             const Field & F,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t width,
             const size_t N2,
             const size_t R1,
             const size_t R2,
             const size_t R3,
             const size_t R4,
             const FFLAS::ParSeqHelper::Sequential seq)  [inline]
```

11.21.3.130 MatrixApplyT() [3/3]

```
template<class Field, class Cut, class Param>
void MatrixApplyT (
             const Field & F,
             typename Field::Element_ptr A,
             const size_t lda,
             const size_t width,
```

```
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4,
            const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)  [inline]
```

11.21.3.131 PermApplyT()

```
template<class T>
void PermApplyT (
            T * A,
            const size_t lda,
            const size_t width,
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)  [inline]
```

11.21.3.132 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)  [inline]
```

Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.

Parameters

in,out	*P1*	a LAPACK permutation of size N
	P2	a LAPACK permutation of size N-R

11.21.3.133 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)  [inline]
```

Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in MathP as a MathPermutation format.

Parameters

out		

a MathPermutation of size N

Parameters

P1	a LAPACK permutation of size N
P2	a LAPACK permutation of size N-R

11.21.3.134 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)  [inline]
```

Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.

Parameters

in,out	*MathP1*	a MathPermutation of size N
	P2	a LAPACK permutation of size N-R

11.21.3.135 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)  [inline]
```

11.21.3.136 cyclic_shift_row_col() [1/2]

```
template<class Field>
void cyclic_shift_row_col (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda)  [inline]
```

11.21.3.137 cyclic_shift_row() [1/3]

```
template<class Field>
void cyclic_shift_row (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda)  [inline]
```

11.21.3.138 cyclic_shift_row() [2/3]

```
template<typename T>
void cyclic_shift_row (
             const RNSIntegerMod< T > & F,
             typename T::Element_ptr A,
             size_t m,
             size_t n,
             size_t lda)  [inline]
```

11.21.3.139 cyclic_shift_col() [1/3]

```
template<class Field>
void cyclic_shift_col (
             const Field & F,
             typename Field::Element_ptr A,
             size_t m,
             size_t n,
             size_t lda)  [inline]
```

11.21.3.140 cyclic_shift_col() [2/3]

```
template<typename T>
void cyclic_shift_col (
             const RNSIntegerMod< T > & F,
             typename T::Element_ptr A,
             size_t m,
             size_t n,
             size_t lda)  [inline]
```

11.21.3.141 PLUQ_basecaseV3()

```
template<class Field>
size_t PLUQ_basecaseV3 (
             const Field & Fi,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element * A,
             const size_t lda,
             size_t * P,
             size_t * Q)  [inline]
```

11.21.3.142 PLUQ_basecaseV2()

```
template<class Field>
size_t PLUQ_basecaseV2 (
             const Field & Fi,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element * A,
             const size_t lda,
             size_t * P,
             size_t * Q)  [inline]
```

11.21.3.143 PLUQ_basecaseCrout()

```
template<class Field>
size_t PLUQ_basecaseCrout (
             const Field & Fi,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q)  [inline]
```

11.21.3.144 _PLUQ()

```
template<class Field>
size_t _PLUQ (
             const Field & Fi,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q,
             size_t BCThreshold)  [inline]
```

11.21.3.145 PLUQ() [4/6]

```
template<class Cut, class Param>
size_t PLUQ (
             const Givaro::Modular< Givaro::Integer > & F,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Givaro::Integer * A,
             const size_t lda,
             size_t * P,
             size_t * Q,
             size_t BCThreshold,
             FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper)  [inline]
```

11.21.3.146 threads_fgemm()

```
template<class Field>
void threads_fgemm (
             const size_t m,
             const size_t n,
             const size_t r,
             int nbthreads,
             size_t * W1,
             size_t * W2,
             size_t * W3,
             size_t gamma)
```

11.21.3.147 threads_ftrsm()

```
template<class Field>
void threads_ftrsm (
             const size_t m,
             const size_t n,
             int nbthreads,
             size_t * t1,
             size_t * t2)
```

11.21.3.148 PLUQ() [5/6]

```
template<class Field>
size_t PLUQ (
             const Field & Fi,
             const FFLAS::FFLAS_DIAG Diag,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q,
             const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & PSHelper)  [inline]
```

11.21.3.149 fflas_const_cast() [1/3]

```
template<>
rns_double_elt_ptr fflas_const_cast (
             rns_double_elt_cstptr x)  [inline]
```

11.21.3.150 fflas_const_cast() [2/3]

```
template<>
rns_double_elt_cstptr fflas_const_cast (
             rns_double_elt_ptr x)  [inline]
```

11.21.3.151 cyclic_shift_row_col() [2/2]

```
template<typename Base_t>
void cyclic_shift_row_col (
             Base_t * A,
             size_t m,
             size_t n,
             size_t lda)
```

11.21.3.152 cyclic_shift_row() [3/3]

```
template INST_OR_DECL void cyclic_shift_row (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             FFLAS_ELT * A,
             size_t m,
             size_t n,
             size_t lda)
```

11.21.3.153 cyclic_shift_col() [3/3]

```
template INST_OR_DECL void cyclic_shift_col (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             FFLAS_ELT * A,
             size_t m,
             size_t n,
             size_t lda)
```

11.21.3.154 applyP() [4/4]

```
template INST_OR_DECL void applyP (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const FFLAS::FFLAS_TRANSPOSE Trans,
             const size_t M,
             const size_t ibeg,
             const size_t iend,
             FFLAS_ELT * A,
             const size_t lda,
             const size_t * P)
```

11.21.3.155 fgetrs() [3/4]

```
template INST_OR_DECL void fgetrs (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t R,
             FFLAS_ELT * A,
             const size_t lda,
             const size_t * P,
             const size_t * Q,
             FFLAS_ELT * B,
             const size_t ldb,
             int * info)
```

11.21.3.156 fgetrs() [4/4]

```
template INST_OR_DECL FFLAS_ELT * fgetrs (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t NRHS,
             const size_t R,
             FFLAS_ELT * A,
             const size_t lda,
             const size_t * P,
             const size_t * Q,
             FFLAS_ELT * X,
             const size_t ldx,
             const FFLAS_ELT * B,
             const size_t ldb,
             int * info)
```

11.21.3.157 fgesv() [3/4]

```
template INST_OR_DECL size_t fgesv (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT * B,
             const size_t ldb,
             int * info)
```

11.21.3.158 fgesv() [4/4]

```
template INST_OR_DECL size_t fgesv (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t NRHS,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT * X,
             const size_t ldx,
             const FFLAS_ELT * B,
             const size_t ldb,
             int * info)
```

11.21.3.159 ftrtri() [2/2]

```
template INST_OR_DECL void ftrtri (
             const FFLAS_FIELD< FFLAS_ELT > & F,
```

```
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            const size_t threshold)
```

11.21.3.160 trinv_left() [2/2]

```
template INST_OR_DECL void trinv_left (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t N,
            const FFLAS_ELT * L,
            const size_t ldl,
            FFLAS_ELT * X,
            const size_t ldx)
```

11.21.3.161 ftrtrm() [2/2]

```
template INST_OR_DECL void ftrtrm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_SIDE side,
            const FFLAS::FFLAS_DIAG diag,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda)
```

11.21.3.162 PLUQ() [6/6]

```
template INST_OR_DECL size_t PLUQ (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Q)
```

11.21.3.163 LUdivine() [4/4]

```
template INST_OR_DECL size_t LUdivine (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_DIAG Diag,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const FFPACK_LU_TAG LuTag,
            const size_t cutoff)
```

11.21.3.164 LUdivine_small() [2/2]

```
template INST_OR_DECL size_t LUdivine_small (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_DIAG Diag,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.165 LUdivine_gauss() [2/2]

```
template INST_OR_DECL size_t LUdivine_gauss (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.166 RowEchelonForm() [3/3]

```
template INST_OR_DECL size_t RowEchelonForm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.167 ReducedRowEchelonForm() [3/3]

```
template INST_OR_DECL size_t ReducedRowEchelonForm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.168 ColumnEchelonForm() [3/3]

```
template INST_OR_DECL size_t ColumnEchelonForm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.169 ReducedColumnEchelonForm() [3/3]

```
template INST_OR_DECL size_t ReducedColumnEchelonForm (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.170 Invert() [3/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            FFLAS_ELT * A,
            const size_t lda,
            int & nullity)
```

11.21.3.171 Invert() [4/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * X,
            const size_t ldx,
            int & nullity)
```

11.21.3.172 Invert2() [2/2]

```
template INST_OR_DECL FFLAS_ELT * Invert2 (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * X,
            const size_t ldx,
            int & nullity)
```

11.21.3.173 CharPoly() [6/8]

```
template INST_OR_DECL std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &
CharPoly (
            const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
            std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & charp,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
            const FFPACK_CHARPOLY_TAG CharpTag,
            const size_t degree)
```

11.21.3.174 CharPoly() [7/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
            const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
            Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
            const FFPACK_CHARPOLY_TAG CharpTag,
            const size_t degree)
```

11.21.3.175 CharPoly() [8/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
            const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
            Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            const FFPACK_CHARPOLY_TAG CharpTag,
            const size_t degree)
```

11.21.3.176 MinPoly() [3/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            std::vector< FFLAS_ELT > & minP,
            const size_t N,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_FIELD< FFLAS_ELT >::RandIter & G)
```

11.21.3.177 MinPoly() [4/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            std::vector< FFLAS_ELT > & minP,
            const size_t N,
            const FFLAS_ELT * A,
            const size_t lda)
```

11.21.3.178 MatVecMinPoly() [2/2]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MatVecMinPoly (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            std::vector< FFLAS_ELT > & minP,
            const size_t N,
            const FFLAS_ELT * A,
            const size_t lda,
            const FFLAS_ELT * V,
            const size_t incv)
```

11.21.3.179 KrylovElim()

```
template INST_OR_DECL size_t KrylovElim (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const size_t deg,
            size_t * iterates,
            size_t * inviterates,
            const size_t maxit,
            size_t virt)
```

11.21.3.180 SpecRankProfile()

```
template INST_OR_DECL size_t SpecRankProfile (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            const size_t deg,
            size_t * rankProfile)
```

11.21.3.181 Rank() [3/3]

```
template INST_OR_DECL size_t Rank (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda)
```

11.21.3.182 IsSingular() [2/2]

```
template INST_OR_DECL bool IsSingular (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda)
```

11.21.3.183 Det() [5/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             FFLAS_ELT & det,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             size_t * P,
             size_t * Q)
```

11.21.3.184 Det() [6/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             FFLAS_ELT & det,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & parH,
             size_t * P,
             size_t * Q)
```

11.21.3.185 Solve() [3/3]

```
template INST_OR_DECL FFLAS_ELT * Solve (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT * x,
             const int incx,
             const FFLAS_ELT * b,
             const int incb)
```

11.21.3.186 solveLB() [2/2]

```
template INST_OR_DECL void solveLB (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const FFLAS::FFLAS_SIDE Side,
             const size_t M,
             const size_t N,
             const size_t R,
             FFLAS_ELT * L,
             const size_t ldl,
             const size_t * Q,
             FFLAS_ELT * B,
             const size_t ldb)
```

11.21.3.187 solveLB2() [2/2]

```
template INST_OR_DECL void solveLB2 (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            FFLAS_ELT * L,
            const size_t ldl,
            const size_t * Q,
            FFLAS_ELT * B,
            const size_t ldb)
```

11.21.3.188 RandomNullSpaceVector() [3/3]

```
template INST_OR_DECL void RandomNullSpaceVector (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * X,
            const size_t incX)
```

11.21.3.189 NullSpaceBasis() [2/2]

```
template INST_OR_DECL size_t NullSpaceBasis (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT *& NS,
            size_t & ldn,
            size_t & NSdim)
```

11.21.3.190 RowRankProfile() [3/3]

```
template INST_OR_DECL size_t RowRankProfile (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t M,
            const size_t N,
            FFLAS_ELT * A,
            const size_t lda,
            size_t *& rkprofile,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.191 ColumnRankProfile() [3/3]

```
template INST_OR_DECL size_t ColumnRankProfile (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             size_t *& rkprofile,
             const FFPACK_LU_TAG LuTag)
```

11.21.3.192 RowRankProfileSubmatrixIndices() [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrixIndices (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             size_t *& rowindices,
             size_t *& colindices,
             size_t & R)
```

11.21.3.193 ColRankProfileSubmatrixIndices() [2/2]

```
template INST_OR_DECL size_t ColRankProfileSubmatrixIndices (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             size_t *& rowindices,
             size_t *& colindices,
             size_t & R)
```

11.21.3.194 RowRankProfileSubmatrix() [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrix (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT *& X,
             size_t & R)
```

11.21.3.195 ColRankProfileSubmatrix() [2/2]

```
template INST_OR_DECL size_t ColRankProfileSubmatrix (
             const FFLAS_FIELD< FFLAS_ELT > & F,
             const size_t M,
             const size_t N,
             FFLAS_ELT * A,
             const size_t lda,
             FFLAS_ELT *& X,
             size_t & R)
```

11.21.3.196 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors)
```

11.21.3.197 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            FFLAS_ELT * A,
            const size_t lda)
```

11.21.3.198 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.199 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            FFLAS_ELT * A,
            const size_t lda,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.200 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()

```
template INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * T,
            const size_t ldt,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.201 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.202 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            FFLAS_ELT * A,
            const size_t lda,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.203 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()

```
template INST_OR_DECL void getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
```

```
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const FFLAS_ELT * A,
            const size_t lda,
            FFLAS_ELT * T,
            const size_t ldt,
            const FFPACK_LU_TAG LuTag)
```

11.21.3.204 LQUPtoInverseOfFullRankMinor() [2/2]

```
template INST_OR_DECL FFLAS_ELT * LQUPtoInverseOfFullRankMinor (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t rank,
            FFLAS_ELT * A_factors,
            const size_t lda,
            const size_t * QtPointer,
            FFLAS_ELT * X,
            const size_t ldx)
```

11.21.3.205 fflas_const_cast() [3/3]

```
template<class T, class CT = const T>
T fflas_const_cast (
            CT x)
```

11.21.3.206 failure()

```
Failure & failure ()  [inline]
```

11.21.3.207 isOdd() [1/3]

```
template<class T>
bool isOdd (
            const T & a)  [inline]
```

11.21.3.208 isOdd() [2/3]

```
bool isOdd (
            const float & a)  [inline]
```

11.21.3.209 isOdd() [3/3]

```
bool isOdd (
            const double & a)  [inline]
```

11.21.3.210 NonZeroRandomMatrix() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr NonZeroRandomMatrix (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random non-zero Matrix.

Creates a m x n matrix with random entries, and at least one of them is non zero.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.211 NonZeroRandomMatrix() [2/2]

```
template<class Field, class RandIter>
Field::Element_ptr NonZeroRandomMatrix (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random non-zero Matrix.

Creates a m x n matrix with random entries, and at least one of them is non zero.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.212 RandomMatrix() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrix (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Matrix.

Creates a m x n matrix with random entries.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.213 RandomMatrix() [2/2]

```
template<class Field>
Field::Element_ptr RandomMatrix (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Matrix.

Creates a m x n matrix with random entries.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.214 RandomTriangularMatrix() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomTriangularMatrix (
            const Field & F,
            size_t m,
            size_t n,
            const FFLAS::FFLAS_UPLO UpLo,
            const FFLAS::FFLAS_DIAG Diag,
            bool nonsingular,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Triangular Matrix.

Creates a m x n triangular matrix with random entries. The UpLo parameter defines wether it is upper or lower triangular.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	UpLo	whether A is upper or lower triangular
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.215 RandomTriangularMatrix() [2/2]

```
template<class Field>
Field::Element_ptr RandomTriangularMatrix (
            const Field & F,
            size_t m,
            size_t n,
            const FFLAS::FFLAS_UPLO UpLo,
            const FFLAS::FFLAS_DIAG Diag,
            bool nonsingular,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Triangular Matrix.

Creates a m x n triangular matrix with random entries. The UpLo parameter defines wether it is upper or lower triangular.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	UpLo	whether A is upper or lower triangular
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.216 RandInt()

```
size_t RandInt (
            size_t a,
            size_t b)  [inline]
```

11.21.3.217 RandomSymmetricMatrix()

```
template<class Field, class RandIter>
Field::Element_ptr RandomSymmetricMatrix (
            const Field & F,
            size_t n,
            bool nonsingular,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Symmetric Matrix.

Creates a m x n triangular matrix with random entries. The UpLo parameter defines wether it is upper or lower triangular.

Parameters

	F	field
	n	order of A
out	*A*	the matrix (preallocated to at least n x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.218 RandomMatrixWithRank() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrixWithRank (
            const Field & F,
            size_t m,
            size_t n,
            size_t r,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Matrix with prescribed rank.

Creates an m x n matrix with random entries and rank r.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least m x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.219 RandomMatrixWithRank() [2/2]

```
template<class Field>
Field::Element_ptr RandomMatrixWithRank (
            const Field & F,
            size_t m,
            size_t n,
            size_t r,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Matrix with prescribed rank.

Creates an m x n matrix with random entries and rank r.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	r	rank of the matrix to build
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.220 RandomIndexSubset()

```
size_t * RandomIndexSubset (
             size_t N,
             size_t R,
             size_t * P)  [inline]
```

Pick uniformly at random a sequence of R distinct elements from the set $\{0, \ldots, N-1\}$ using Knuth's shuffle.

Parameters

	N	the cardinality of the sampling set
	R	the number of elements to sample
out	*P*	the output sequence (pre-allocated to at least R indices)

11.21.3.221 RandomPermutation()

```
size_t * RandomPermutation (
             size_t N,
             size_t * P)  [inline]
```

Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.

Parameters

	N	the length of the permutation
out	*P*	the output permutation (pre-allocated to at least N indices)

11.21.3.222 RandomRankProfileMatrix()

```
void RandomRankProfileMatrix (
             size_t M,
             size_t N,
             size_t R,
             size_t * rows,
             size_t * cols)  [inline]
```

Pick uniformly at random an R-subpermutation of dimension M x N : a matrix with only R non-zeros equal to one, in a random rook placement.

Parameters

	M	row dimension
	N	column dimension
out	*rows*	the row position of each non zero element (pre-allocated)
out	*cols*	the column position of each non zero element (pre-allocated)

11.21.3.223 swapval()

```
void swapval (
            size_t k,
            size_t N,
            size_t * P,
            size_t val)  [inline]
```

11.21.3.224 RandomSymmetricRankProfileMatrix()

```
void RandomSymmetricRankProfileMatrix (
            size_t N,
            size_t R,
            size_t * rows,
            size_t * cols)  [inline]
```

Pick uniformly at random a symmetric R-subpermutation of dimension N x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

Parameters

	N	matrix order
out	*rows*	the row position of each non zero element (pre-allocated)
out	*cols*	the column position of each non zero element (pre-allocated)

11.21.3.225 RandomLTQSRankProfileMatrix()

```
void RandomLTQSRankProfileMatrix (
            size_t n,
            size_t r,
            size_t t,
            size_t * rows,
            size_t * cols)  [inline]
```

11.21.3.226 RandomMatrixWithRankandRPM() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrixWithRankandRPM (
            const Field & F,
            size_t M,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda,
            const size_t * RRP,
            const size_t * CRP,
            RandIter & G)  [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least m x lda field elements)
lda	leading dimension of A
RRP	the R dimensional array with row positions of the rank profile matrix' pivots
CRP	the R dimensional array with column positions of the rank profile matrix' pivots
G	a random iterator

Returns

A.

11.21.3.227 RandomMatrixWithRankandRPM() [2/2]

```
template<class Field>
Field::Element_ptr RandomMatrixWithRankandRPM (
            const Field & F,
            size_t M,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda,
            const size_t * RRP,
            const size_t * CRP)  [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least m x lda field elements)
lda	leading dimension of A
RRP	the R dimensional array with row positions of the rank profile matrix' pivots
CRP	the R dimensional array with column positions of the rank profile matrix' pivots

Returns

A.

11.21.3.228 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
             const Field & F,
             size_t N,
             size_t R,
             typename Field::Element_ptr A,
             size_t lda,
             const size_t * RRP,
             const size_t * CRP,
             RandIter & G)  [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an `n` x `n` symmetric matrix with random entries and rank `r`.

Parameters

F	field
n	order of `A`
r	rank of `A`
A	the matrix (preallocated to at least `n` x `lda` field elements)
lda	leading dimension of `A`
RRP	the R dimensional array with row positions of the rank profile matrix' pivots
CRP	the R dimensional array with column positions of the rank profile matrix' pivots
G	a random iterator

Returns

`A`.

11.21.3.229 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
template<class Field>
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
             const Field & F,
             size_t M,
             size_t N,
             size_t R,
             typename Field::Element_ptr A,
             size_t lda,
             const size_t * RRP,
             const size_t * CRP)  [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an `n` x `n` symmetric matrix with random entries and rank `r`.

Parameters

F	field
n	order of `A`
r	rank of `A`
A	the matrix (preallocated to at least `n` x `lda` field elements)

lda	leading dimension of A
RRP	the R dimensional array with row positions of the rank profile matrix' pivots
CRP	the R dimensional array with column positions of the rank profile matrix' pivots

Returns

A.

11.21.3.230 RandomMatrixWithRankandRandomRPM() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
            const Field & F,
            size_t M,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least m x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.231 RandomMatrixWithRankandRandomRPM() [2/2]

```
template<class Field>
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
            const Field & F,
            size_t M,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least m x lda field elements)
lda	leading dimension of A

Returns

A.

11.21.3.232 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
            const Field & F,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
n	order of A
r	rank of A
A	the matrix (preallocated to at least n x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.233 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]

```
template<class Field>
Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
            const Field & F,
            size_t N,
            size_t R,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
n	order of A
r	rank of A
A	the matrix (preallocated to at least n x lda field elements)
lda	leading dimension of A

Returns

A.

11.21.3.234 RandomMatrixWithDet() [1/2]

```
template<class Field>
Field::Element_ptr RandomMatrixWithDet (
            const Field & F,
            size_t n,
            const typename Field::Element d,
            typename Field::Element_ptr A,
            size_t lda)  [inline]
```

Random Matrix with prescribed det.

Creates a m x n matrix with random entries and rank r.

Parameters

F	field
d	the prescribed value for the determinant of A
n	number of cols in A
A	the matrix to be generated (preallocated to at least n x lda field elements)
lda	leading dimension of A

Returns

A.

11.21.3.235 RandomMatrixWithDet() [2/2]

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrixWithDet (
            const Field & F,
            size_t n,
            const typename Field::Element d,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Matrix with prescribed det.

Creates a m x n matrix with random entries and rank r.

Parameters

F	field
d	the prescribed value for the determinant of A
n	number of cols in `A`
A	the matrix to be generated (preallocated to at least `n` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

11.21.3.236 RandomLTQSMatrixWithRankandQSorder()

```
template<class Field, class RandIter>
Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (
            Field & F,
            size_t n,
            size_t r,
            size_t t,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

11.21.3.237 chooseField()

```
template<typename Field>
Field * chooseField (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.238 chooseField< Givaro::ZRing< int32_t > >()

```
template<>
Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.239 chooseField< Givaro::ZRing< int64_t > >()

```
template<>
Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.240 chooseField< Givaro::ZRing< float > >()

```
template<>
Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.241 chooseField< Givaro::ZRing< double > >()

```
template<>
Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.22 FFPACK::Protected Namespace Reference

Functions

- template<class Field>
  size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FfpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)
- template<class Field>
  size_t GaussJordan (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

  Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.
- template<class Field, class Polynomial>
  std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)
- template<class Field, class Polynomial>
  int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)
- template<class Field, class Polynomial>
  std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class Field>
  void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)
- template<class Field, class Polynomial, class RandIter>
  std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)
- template<class Field, class Polynomial>
  std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class PolRing>
  void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completed↩ Factors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_↩ t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

- template<class PolRing>
  std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::↩
  Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)
- template<class Field, class Polynomial>
  std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)
- template<class Field, class Polynomial>
  Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_↩
  mb=0, const size_t kg_j=0)
- template<class Field>
  size_t updateD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > > &minpt)
- template<class Field>
  size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)
- template<class Field>
  void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)
- template<class Field>
  void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size↩
  _t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)
- template<class Field>
  void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size↩
  _t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK↩
  _MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)

11.22.1 Function Documentation

11.22.1.1 LUdivine_construct() [1/2]

```
template<class Field>
size_t LUdivine_construct (
            const Field & F,
            const FFLAS::FFLAS_DIAG Diag,
```

```
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx,
            typename Field::Element_ptr u,
            const size_t incu,
            size_t * P,
            bool computeX,
            const FFPACK_MINPOLY_TAG MinTag = FfpackDense,
            const size_t kg_mc = 0,
            const size_t kg_mb = 0,
            const size_t kg_j = 0)
```

11.22.1.2 GaussJordan()

```
template<class Field>
size_t GaussJordan (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            const size_t colbeg,
            const size_t rowbeg,
            const size_t colsize,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.

Bibliography

- Algorithm 2.8 of A. Storjohann Thesis 2000,
- Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. `Rank-profile` *revealing Gaussian elimination and the CUP matrix decomposition* , J. of Symbolic Comp., 2013

Parameters

	M	row dimension of A
	N	column dimension of A
`in,out`	*A*	an m x n matrix
	lda	leading dimension of A
	P	row permutation
	Q	column permutation
	LuTag	set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon

| I | A11 | A12 | | |---|---|---|---| | |I | * | A22 | | | |0 | 0| A22 | | |---|---|---|---| | | 0 | A32 | | |---|---|---|---|
where the transformation matrix is stored at the pivot column position

11.22.1.3 KellerGehrig()

```
template<class Field, class Polynomial>
std::list< Polynomial > & KellerGehrig (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda)
```

11.22.1.4 KGFast()

```
template<class Field, class Polynomial>
int KGFast (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * kg_mc,
            size_t * kg_mb,
            size_t * kg_j)
```

11.22.1.5 KGFast_generalized()

```
template<class Field, class Polynomial>
std::list< Polynomial > & KGFast_generalized (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda)
```

11.22.1.6 fgemv_kgf()

```
template<class Field>
void fgemv_kgf (
            const Field & F,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr X,
            const size_t incX,
            typename Field::Element_ptr Y,
            const size_t incY,
            const size_t kg_mc,
            const size_t kg_mb,
            const size_t kg_j)
```

11.22.1.7 LUKrylov()

```
template<class Field, class Polynomial, class RandIter>
std::list< Polynomial > & LUKrylov (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr U,
```

```
            const size_t ldu,
            RandIter & G)
```

11.22.1.8 Danilevski()

```
template<class Field, class Polynomial>
std::list< Polynomial > & Danilevski (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda)
```

11.22.1.9 RandomKrylovPrecond()

```
template<class PolRing>
void RandomKrylovPrecond (
            const PolRing & PR,
            std::list< typename PolRing::Element > & completedFactors,
            const size_t N,
            typename PolRing::Domain_t::Element_ptr A,
            const size_t lda,
            size_t & Nb,
            typename PolRing::Domain_t::Element_ptr & B,
            size_t & ldb,
            typename PolRing::Domain_t::RandIter & g,
            const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD)  [inline]
```

Todo swap to save space ??

Todo

Todo don't assing K2 c∗noc x N but only mas (c,noc) x N and store each one after the other

11.22.1.10 ArithProg()

```
template<class PolRing>
std::list< typename PolRing::Element > & ArithProg (
            const PolRing & PR,
            std::list< typename PolRing::Element > & frobeniusForm,
            const size_t N,
            typename PolRing::Domain_t::Element_ptr A,
            const size_t lda,
            const size_t degree)  [inline]
```

11.22.1.11 LUKrylov_KGFast()

```
template<class Field, class Polynomial>
std::list< Polynomial > & LUKrylov_KGFast (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx)
```

11.22.1.12 MatVecMinPoly()

```
template<class Field, class Polynomial>
Polynomial & MatVecMinPoly (
              const Field & F,
              Polynomial & minP,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::Element_ptr v,
              const size_t incv,
              typename Field::Element_ptr K,
              const size_t ldk,
              size_t * P)  [inline]
```

11.22.1.13 Hybrid_KGF_LUK_MinPoly()

```
template<class Field, class Polynomial>
Polynomial & Hybrid_KGF_LUK_MinPoly (
              const Field & F,
              Polynomial & minP,
              const size_t N,
              typename Field::ConstElement_ptr A,
              const size_t lda,
              typename Field::Element_ptr X,
              const size_t ldx,
              size_t * P,
              const FFPACK_MINPOLY_TAG MinTag = FFPACK::FfpackDense,
              const size_t kg_mc = 0,
              const size_t kg_mb = 0,
              const size_t kg_j = 0)
```

11.22.1.14 updateD()

```
template<class Field>
size_t updateD (
              const Field & F,
              size_t * d,
              size_t k,
              std::vector< std::vector< typename Field::Element > > & minpt)
```

11.22.1.15 newD()

```
template<class Field>
size_t newD (
              const Field & F,
              size_t * d,
              bool & KeepOn,
              const size_t l,
              const size_t N,
              typename Field::Element_ptr X,
              const size_t * Q,
              std::vector< std::vector< typename Field::Element > > & minpt)
```

11.22.1.16 CompressRows()

```
template<class Field>
void CompressRows (
              Field & F,
```

```
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t nb_blocs)  [inline]
```

11.22.1.17 CompressRowsQK()

```
template<class Field>
void CompressRowsQK (
            Field & F,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t deg,
            const size_t nb_blocs)  [inline]
```

11.22.1.18 DeCompressRows()

```
template<class Field>
void DeCompressRows (
            Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t nb_blocs)  [inline]
```

11.22.1.19 DeCompressRowsQK()

```
template<class Field>
void DeCompressRowsQK (
            Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t deg,
            const size_t nb_blocs)  [inline]
```

11.22.1.20 CompressRowsQA()

```
template<class Field>
void CompressRowsQA (
            Field & F,
            const size_t M,
            typename Field::Element_ptr A,
            const size_t lda,
```

```
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t nb_blocs)  [inline]
```

11.22.1.21 DeCompressRowsQA()

```
template<class Field>
void DeCompressRowsQA (
            Field & F,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr tmp,
            const size_t ldtmp,
            const size_t * d,
            const size_t nb_blocs)  [inline]
```

11.22.1.22 LUdivine_construct() [2/2]

```
template<class Field>
size_t LUdivine_construct (
            const Field & F,
            const FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr X,
            const size_t ldx,
            typename Field::Element_ptr u,
            const size_t incu,
            size_t * P,
            bool computeX,
            const FFPACK::FFPACK_MINPOLY_TAG MinTag,
            const size_t kg_mc,
            const size_t kg_mb,
            const size_t kg_j)
```

11.23 Givaro Namespace Reference

Data Structures

- class ModularBalanced
- class Montgomery

11.24 MKL_CONFIG Namespace Reference

11.25 RecInt Namespace Reference

Data Structures

- class rint
- class ruint

Chapter 12

Data Structure Documentation

12.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Winograd value

12.1.1 Member Typedef Documentation

12.1.1.1 value

```
template<class ModeT, class ParSeq>
typedef MMHelperAlgo::Winograd value
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Classic value
- typedef MMHelperAlgo::Winograd value

12.2.1 Member Typedef Documentation

12.2.1.1 value [1/2]

```
template<class ParSeq>
typedef MMHelperAlgo::Classic value
```

12.2.1.2 value [2/2]

```
typedef MMHelperAlgo::Winograd value
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.3 ALL< v > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.4 ALL< false, v... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = false

12.4.1 Field Documentation

12.4.1.1 value

```
template<bool...  v>
bool value = false  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.5 ALL< true, v... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = ALL<v...>::value

12.5.1 Field Documentation

12.5.1.1 value

```
template<bool...  v>
bool value = ALL<v...>::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.6 ALL<> Struct Reference

Static Public Attributes

- static constexpr bool value = true

12.6.1 Field Documentation

12.6.1.1 value

```
bool value = true  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.7 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.
`#include <field-traits.h>`

12.7.1 Detailed Description

Arbitrary precision integers: GMP.
The documentation for this struct was generated from the following file:

- field-traits.h

12.8 AreEqual< X, Y > Class Template Reference

`#include <fflas_enum.h>`

Static Public Attributes

- static const bool value = false

12.8.1 Field Documentation

12.8.1.1 value

```
template<class X, class Y>
const bool value = false  [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

12.9 AreEqual< X, X > Class Template Reference

`#include <fflas_enum.h>`

Static Public Attributes

- static const bool value = true
- static const bool value

12.9.1 Field Documentation

12.9.1.1 value [1/2]

```
template<class X>
const bool value = true  [static]
```

12.9.1.2 value [2/2]

```
const bool value  [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

12.10 Argument Struct Reference

`#include <args-parser.h>`

Data Fields

- char c
- const char * example
- const char * helpString
- ArgumentType type
- void * data

12.10.1 Field Documentation

12.10.1.1 c

```
char c
```

12.10.1.2 example

```
const char* example
```

12.10.1.3 helpString

```
const char* helpString
```

12.10.1.4 type

```
ArgumentType type
```

12.10.1.5 data

```
void* data
```

The documentation for this struct was generated from the following file:

- args-parser.h

12.11 array< T > Class Template Reference

STL class.

Data Structures

- class const_iterator
  STL iterator class.
- class const_reverse_iterator
  STL iterator class.
- class iterator
  STL iterator class.
- class reverse_iterator
  STL iterator class.

Data Fields

- T elements
  STL member.

12.11.1 Detailed Description

STL class.

12.11.2 Field Documentation

12.11.2.1 elements

```
T elements
```

STL member.

The documentation for this class was generated from the following files:

12.12 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Field field
- typedef Field & type

12.12.1 Member Typedef Documentation

12.12.1.1 field

```
template<class Field>
typedef Field field
```

12.12.1.2 type

```
template<class Field>
typedef Field& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.13 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FFPACK::RNSInteger< RNS > field
- typedef FFPACK::RNSInteger< RNS > type
- typedef const FFPACK::RNSIntegerMod< RNS > field
- typedef const FFPACK::RNSIntegerMod< RNS > & type

12.13.1 Member Typedef Documentation

12.13.1.1 field [1/2]

```
template<typename RNS>
typedef FFPACK::RNSInteger<RNS> field
```

12.13.1.2 type [1/2]

```
template<typename RNS>
typedef FFPACK::RNSInteger<RNS> type
```

12.13.1.3 field [2/2]

```
typedef const FFPACK::RNSIntegerMod< RNS > field
```

12.13.1.4 type [2/2]

```
typedef const FFPACK::RNSIntegerMod< RNS >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.14 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type
- typedef const Givaro::Modular< T, X > field
- typedef const Givaro::Modular< T, X > & type

12.14.1 Member Typedef Documentation

12.14.1.1 field [1/2]

```
template<typename T, typename X>
typedef Givaro::ZRing<T> field
```

12.14.1.2 type [1/2]

```
template<typename T, typename X>
typedef Givaro::ZRing<T> type
```

12.14.1.3 field [2/2]

```
typedef const Givaro::Modular< T, X > field
```

12.14.1.4 type [2/2]

```
typedef const Givaro::Modular< T, X >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.15 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type
- typedef const Givaro::ModularBalanced< T > field
- typedef const Givaro::ModularBalanced< T > & type

12.15.1 Member Typedef Documentation

12.15.1.1 field [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> field
```

12.15.1.2 type [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> type
```

12.15.1.3 field [2/2]

```
typedef const Givaro::ModularBalanced< T > field
```

12.15.1.4 type [2/2]

```
typedef const Givaro::ModularBalanced< T >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.16 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type
- typedef const Givaro::ZRing< T > field
- typedef const Givaro::ZRing< T > & type

12.16.1 Member Typedef Documentation

12.16.1.1 field [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> field
```

12.16.1.2 type [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> type
```

12.16.1.3 field [2/2]

```
typedef const Givaro::ZRing< T > field
```

12.16.1.4 type [2/2]

```
typedef const Givaro::ZRing< T >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.17 Auto Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.18 Bench< Elt > Class Template Reference

Public Types

- using Field = Modular<Elt>
- using Elt_ptr = typename Field::Element_ptr
- using Residu = typename Field::Residu_t
- template<bool B, class T = void>
  using enable_if_t = typename std::enable_if<B, T>::type

- template<typename Simd>
  using is_same_element = typename Simd::template is_same_element<Field>
- template<typename E>
  using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
- template<typename E>
  using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
- template<typename E>
  using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
- template<typename E>
  using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>

Public Member Functions

- Bench (size_t m, size_t n, size_t iters, bool inplace)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * = nullptr>
  void doBenchs ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t< _E > * = nullptr>
  void run (bool allsimd)

Static Public Member Functions

- template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
  static Residu cardinality ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
  static Residu cardinality ()

Protected Attributes

- Field F
- const size_t m
- const size_t n
- const size_t iters
- const bool inplace

12.18.1 Member Typedef Documentation

12.18.1.1 Field

```
template<typename Elt>
using Field = Modular<Elt>
```

12.18.1.2 Elt_ptr

```
template<typename Elt>
using Elt_ptr = typename Field::Element_ptr
```

12.18.1.3 Residu

```
template<typename Elt>
using Residu = typename Field::Residu_t
```

12.18.1.4 enable_if_t

```
template<typename Elt>
template<bool B, class T = void>
using enable_if_t = typename std::enable_if<B, T>::type
```

12.18.1.5 is_same_element

```
template<typename Elt>
template<typename Simd>
using is_same_element = typename Simd::template is_same_element<Field>
```

12.18.1.6 enable_if_no_simd_t

```
template<typename Elt>
template<typename E>
using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
```

12.18.1.7 enable_if_simd128_t

```
template<typename Elt>
template<typename E>
using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
```

12.18.1.8 enable_if_simd256_t

```
template<typename Elt>
template<typename E>
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

12.18.1.9 enable_if_simd512_t

```
template<typename Elt>
template<typename E>
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

12.18.2 Constructor & Destructor Documentation

12.18.2.1 Bench()

```
template<typename Elt>
Bench (
            size_t m,
            size_t n,
            size_t iters,
            bool inplace)  [inline]
```

12.18.3 Member Function Documentation

12.18.3.1 cardinality() [1/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.18.3.2 cardinality() [2/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.18.3.3 doBenchs()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
void doBenchs ()  [inline]
```

12.18.3.4 run()

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t<
_E > * = nullptr>
void run (
            bool allsimd)  [inline]
```

12.18.4 Field Documentation

12.18.4.1 F

```
template<typename Elt>
Field F  [protected]
```

12.18.4.2 m

```
template<typename Elt>
const size_t m  [protected]
```

12.18.4.3 n

```
template<typename Elt>
const size_t n  [protected]
```

12.18.4.4 iters

```
template<typename Elt>
const size_t iters  [protected]
```

12.18.4.5 inplace

```
template<typename Elt>
const bool inplace  [protected]
```

The documentation for this class was generated from the following file:

- benchmark-storage-transpose.C

12.19 Bini Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.20 Block Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.21 BlockTransposeSIMD< Field, Simd, > Struct Template Reference

```
#include <fflas_transpose.h>
```

Public Member Functions

- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 1 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 2 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 4 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 8 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 16 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const

Static Public Member Functions

- static constexpr size_t size ()
- static const std::string info ()

12.21.1 Member Function Documentation

12.21.1.1 size()

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
static constexpr size_t size ()  [inline], [static], [constexpr]
```

12.21.1.2 info()

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
static const std::string info ()  [inline], [static]
```

12.21.1.3 transpose() [1/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 1 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.4 transpose() [2/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 2 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.5 transpose() [3/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 4 > * = nullptr>
void transpose (
             const Field & F,
             ConstElement_ptr A,
             size_t lda,
             Element_ptr B,
             size_t ldb) const  [inline]
```

12.21.1.6 transpose() [4/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 8 > * = nullptr>
void transpose (
             const Field & F,
             ConstElement_ptr A,
             size_t lda,
             Element_ptr B,
             size_t ldb) const  [inline]
```

12.21.1.7 transpose() [5/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 16 > * = nullptr>
void transpose (
             const Field & F,
             ConstElement_ptr A,
             size_t lda,
             Element_ptr B,
             size_t ldb) const  [inline]
```

The documentation for this struct was generated from the following file:

- fflas_transpose.h

12.22 callLUdivine_small< Element > Class Template Reference

Public Member Functions

- template<class Field>
  size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_↩ t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

12.22.1 Member Function Documentation

12.22.1.1 operator()()

```
template<class Element>
template<class Field>
size_t operator() (
             const Field & F,
             const FFLAS::FFLAS_DIAG Diag,
             const FFLAS::FFLAS_TRANSPOSE trans,
             const size_t M,
```

```
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

The documentation for this class was generated from the following file:

- ffpack_ludivine.inl

12.23 callLUdivine_small< double > Class Reference

Public Member Functions

- template<class Field>
  size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

12.23.1 Member Function Documentation

12.23.1.1 operator()() [1/2]

```
template<class Field>
size_t operator() (
            const Field & F,
            const FFLAS::FFLAS_DIAG Diag,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

12.23.1.2 operator()() [2/2]

```
size_t operator() (
            const Field & F,
            const FFLAS::FFLAS_DIAG Diag,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

The documentation for this class was generated from the following file:

- ffpack_ludivine.inl

12.24 callLUdivine_small< float > Class Reference

Public Member Functions

- template<class Field>
  size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

12.24.1 Member Function Documentation

12.24.1.1 operator()() [1/2]

```
template<class Field>
size_t operator() (
             const Field & F,
             const FFLAS::FFLAS_DIAG Diag,
             const FFLAS::FFLAS_TRANSPOSE trans,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q,
             const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

12.24.1.2 operator()() [2/2]

```
size_t operator() (
             const Field & F,
             const FFLAS::FFLAS_DIAG Diag,
             const FFLAS::FFLAS_TRANSPOSE trans,
             const size_t M,
             const size_t N,
             typename Field::Element_ptr A,
             const size_t lda,
             size_t * P,
             size_t * Q,
             const FFPACK::FFPACK_LU_TAG LuTag)  [inline]
```

The documentation for this class was generated from the following file:

- ffpack_ludivine.inl

12.25 CharpolyFailed Class Reference

`#include <ffpack.h>`

The documentation for this class was generated from the following file:

- ffpack.h

12.26 Checker_Empty< Field > Struct Template Reference

`#include <checker_empty.h>`

Public Member Functions

- template<typename... Params>
  Checker_Empty (Params... parameters)
- template<typename... Params>
  bool check (Params... parameters)

12.26.1 Constructor & Destructor Documentation

12.26.1.1 Checker_Empty()

```
template<class Field>
template<typename... Params>
Checker_Empty (
            Params... parameters) [inline]
```

12.26.2 Member Function Documentation

12.26.2.1 check()

```
template<class Field>
template<typename... Params>
bool check (
            Params... parameters) [inline]
```

The documentation for this struct was generated from the following file:

- checker_empty.h

12.27 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference

Public Member Functions

- CheckerImplem_charpoly (const Field &F_, const size_t n_, typename Field::ConstElement_ptr A, size_t lda_)
- CheckerImplem_charpoly (typename Field::RandIter &G, const size_t n_, typename Field::ConstElement_ptr A, size_t lda_)
- ~CheckerImplem_charpoly ()
- bool check (Polynomial &g)

12.27.1 Constructor & Destructor Documentation

12.27.1.1 CheckerImplem_charpoly() [1/2]

```
template<class Field, class Polynomial>
CheckerImplem_charpoly (
            const Field & F_,
            const size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.27.1.2 CheckerImplem_charpoly() [2/2]

```
template<class Field, class Polynomial>
CheckerImplem_charpoly (
            typename Field::RandIter & G,
            const size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.27.1.3 ~CheckerImplem_charpoly()

```
template<class Field, class Polynomial>
~CheckerImplem_charpoly ()  [inline]
```

12.27.2 Member Function Documentation

12.27.2.1 check()

```
template<class Field, class Polynomial>
bool check (
            Polynomial & g)  [inline]
```

The documentation for this class was generated from the following file:

- checker_charpoly.inl

12.28 CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > Class Template Reference

Public Types

- typedef Givaro::ZRing< Givaro::Integer > Ring

Public Member Functions

- CheckerImplem_charpoly (const Ring &F_, const size_t n_, typename Ring::ConstElement_ptr A, size_t lda_)
- CheckerImplem_charpoly (typename Ring::RandIter &G, const size_t n_, typename Ring::ConstElement_ptr A, size_t lda_)
- ~CheckerImplem_charpoly ()
- bool check (Polynomial &g)
- CheckerImplem_charpoly (const Givaro::ZRing< Givaro::Integer > &F_, const size_t n_, typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A, size_t lda_)
- CheckerImplem_charpoly (typename Givaro::ZRing< Givaro::Integer >::RandIter &G, const size_t n_, typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A, size_t lda_)
- ~CheckerImplem_charpoly ()
- bool check (Polynomial &g)

12.28.1 Member Typedef Documentation

12.28.1.1 Ring

```
template<class Polynomial>
typedef Givaro::ZRing<Givaro::Integer> Ring
```

12.28.2 Constructor & Destructor Documentation

12.28.2.1 CheckerImplem_charpoly() [1/4]

```
template<class Polynomial>
CheckerImplem_charpoly (
            const Ring & F_,
            const size_t n_,
            typename Ring::ConstElement_ptr A,
            size_t lda_)  [inline]
```

12.28.2.2 CheckerImplem_charpoly() [2/4]

```
template<class Polynomial>
CheckerImplem_charpoly (
            typename Ring::RandIter & G,
            const size_t n_,
            typename Ring::ConstElement_ptr A,
            size_t lda_)  [inline]
```

12.28.2.3 ~CheckerImplem_charpoly() [1/2]

```
template<class Polynomial>
~CheckerImplem_charpoly ()  [inline]
```

12.28.2.4 CheckerImplem_charpoly() [3/4]

```
CheckerImplem_charpoly (
            const Givaro::ZRing< Givaro::Integer > & F_,
            const size_t n_,
            typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A,
            size_t lda_)  [inline]
```

12.28.2.5 CheckerImplem_charpoly() [4/4]

```
CheckerImplem_charpoly (
            typename Givaro::ZRing< Givaro::Integer >::RandIter & G,
            const size_t n_,
            typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A,
            size_t lda_)  [inline]
```

12.28.2.6 ~CheckerImplem_charpoly() [2/2]

```
~CheckerImplem_charpoly ()  [inline]
```

12.28.3 Member Function Documentation

12.28.3.1 check() [1/2]

```
template<class Polynomial>
bool check (
            Polynomial & g)  [inline]
```

12.28.3.2 check() [2/2]

```
bool check (
            Polynomial & g)  [inline]
```

The documentation for this class was generated from the following file:

- checker_charpoly.inl

12.29 CheckerImplem_Det< Field > Class Template Reference

Public Member Functions

- CheckerImplem_Det (const Field &F_, size_t n_, typename Field::ConstElement_ptr A, size_t lda)
- CheckerImplem_Det (typename Field::RandIter &G, size_t n_, typename Field::ConstElement_ptr A, size_t lda)
- ~CheckerImplem_Det ()
- bool check (const typename Field::Element &det, typename Field::ConstElement_ptr LU, size_t lda, size_t *P, size_t *Q) const

check if the Det factorization is correct.

12.29.1 Constructor & Destructor Documentation

12.29.1.1 CheckerImplem_Det() [1/2]

```
template<class Field>
CheckerImplem_Det (
            const Field & F_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.29.1.2 CheckerImplem_Det() [2/2]

```
template<class Field>
CheckerImplem_Det (
            typename Field::RandIter & G,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.29.1.3 ~CheckerImplem_Det()

```
template<class Field>
~CheckerImplem_Det ()  [inline]
```

12.29.2 Member Function Documentation

12.29.2.1 check()

```
template<class Field>
bool check (
            const typename Field::Element & det,
            typename Field::ConstElement_ptr LU,
            size_t lda,
            size_t * P,
            size_t * Q) const  [inline]
```

check if the Det factorization is correct.
Needs matrix in LU form

Parameters

LU,storage	for L and U
det	
P	
Q	

The documentation for this class was generated from the following file:

- checker_det.inl

12.30 CheckerImplem_fgemm< Field > Class Template Reference

Public Member Functions

- CheckerImplem_fgemm (const Field &F_, const size_t m_, const size_t n_, const size_t k_, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc_)

- CheckerImplem_fgemm (typename Field::RandIter &G, const size_t m_, const size_t n_, const size_t k_, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc_)
- ~CheckerImplem_fgemm ()
- bool check (const FFLAS::FFLAS_TRANSPOSE ta, const FFLAS::FFLAS_TRANSPOSE tb, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::ConstElement_ptr C)

12.30.1 Constructor & Destructor Documentation

12.30.1.1 CheckerImplem_fgemm() [1/2]

```
template<class Field>
CheckerImplem_fgemm (
            const Field & F_,
            const size_t m_,
            const size_t n_,
            const size_t k_,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc_)  [inline]
```

12.30.1.2 CheckerImplem_fgemm() [2/2]

```
template<class Field>
CheckerImplem_fgemm (
            typename Field::RandIter & G,
            const size_t m_,
            const size_t n_,
            const size_t k_,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc_)  [inline]
```

12.30.1.3 ~CheckerImplem_fgemm()

```
template<class Field>
~CheckerImplem_fgemm ()  [inline]
```

12.30.2 Member Function Documentation

12.30.2.1 check()

```
template<class Field>
bool check (
            const FFLAS::FFLAS_TRANSPOSE ta,
            const FFLAS::FFLAS_TRANSPOSE tb,
            const typename Field::Element alpha,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            typename Field::ConstElement_ptr C)  [inline]
```

The documentation for this class was generated from the following file:

- checker_fgemm.inl

12.31 CheckerImplem_ftrsm< Field > Class Template Reference

Public Member Functions

- CheckerImplem_ftrsm (const Field &F_, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr B, const size_t ldb)
- CheckerImplem_ftrsm (typename Field::RandIter &G, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr B, const size_t ldb)
- ~CheckerImplem_ftrsm ()
- bool check (const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO uplo, const FFLAS::FFLAS_TRANSPOSE trans, const FFLAS::FFLAS_DIAG diag, const size_t m, const size_t n, typename Field::Element_ptr A, size_t lda, const typename Field::ConstElement_ptr X, size_t ldx)

12.31.1 Constructor & Destructor Documentation

12.31.1.1 CheckerImplem_ftrsm() [1/2]

```
template<class Field>
CheckerImplem_ftrsm (
            const Field & F_,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr B,
            const size_t ldb)  [inline]
```

12.31.1.2 CheckerImplem_ftrsm() [2/2]

```
template<class Field>
CheckerImplem_ftrsm (
            typename Field::RandIter & G,
            const size_t m,
            const size_t n,
            const typename Field::Element alpha,
            const typename Field::ConstElement_ptr B,
            const size_t ldb)  [inline]
```

12.31.1.3 ~CheckerImplem_ftrsm()

```
template<class Field>
~CheckerImplem_ftrsm ()  [inline]
```

12.31.2 Member Function Documentation

12.31.2.1 check()

```
template<class Field>
bool check (
            const FFLAS::FFLAS_SIDE side,
            const FFLAS::FFLAS_UPLO uplo,
            const FFLAS::FFLAS_TRANSPOSE trans,
            const FFLAS::FFLAS_DIAG diag,
            const size_t m,
            const size_t n,
            typename Field::Element_ptr A,
            size_t lda,
            const typename Field::ConstElement_ptr X,
            size_t ldx)  [inline]
```

The documentation for this class was generated from the following file:

- checker_ftrsm.inl

12.32 CheckerImplem_invert< Field > Class Template Reference

Public Member Functions

- CheckerImplem_invert (const Field &F_, const size_t m_, typename Field::ConstElement_ptr A, const size_t lda_)
- CheckerImplem_invert (typename Field::RandIter &G, const size_t m_, typename Field::ConstElement_ptr A, const size_t lda_)
- ~CheckerImplem_invert ()
- bool check (typename Field::ConstElement_ptr A, int nullity)

12.32.1 Constructor & Destructor Documentation

12.32.1.1 CheckerImplem_invert() [1/2]

```
template<class Field>
CheckerImplem_invert (
            const Field & F_,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_)  [inline]
```

12.32.1.2 CheckerImplem_invert() [2/2]

```
template<class Field>
CheckerImplem_invert (
            typename Field::RandIter & G,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_)  [inline]
```

12.32.1.3 ~CheckerImplem_invert()

```
template<class Field>
~CheckerImplem_invert ()  [inline]
```

12.32.2 Member Function Documentation

12.32.2.1 check()

```
template<class Field>
bool check (
            typename Field::ConstElement_ptr A,
            int nullity)  [inline]
```

The documentation for this class was generated from the following file:

- checker_invert.inl

12.33 CheckerImplem_PLUQ< Field > Class Template Reference

Public Member Functions

- CheckerImplem_PLUQ (const Field &F_, size_t m_, size_t n_, typename Field::ConstElement_ptr A, size_t lda)
- CheckerImplem_PLUQ (typename Field::RandIter &G, size_t m_, size_t n_, typename Field::ConstElement_ptr A, size_t lda)
- ~CheckerImplem_PLUQ ()
- bool check (typename Field::ConstElement_ptr A, size_t lda, const FFLAS::FFLAS_DIAG Diag, size_t r, size_t *P, size_t *Q) const

  check if the PLUQ factorization is correct.

12.33.1 Constructor & Destructor Documentation

12.33.1.1 CheckerImplem_PLUQ() [1/2]

```
template<class Field>
CheckerImplem_PLUQ (
            const Field & F_,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.33.1.2 CheckerImplem_PLUQ() [2/2]

```
template<class Field>
CheckerImplem_PLUQ (
            typename Field::RandIter & G,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.33.1.3 ~CheckerImplem_PLUQ()

```
template<class Field>
~CheckerImplem_PLUQ ()  [inline]
```

12.33.2 Member Function Documentation

12.33.2.1 check()

```
template<class Field>
bool check (
            typename Field::ConstElement_ptr A,
            size_t lda,
            const FFLAS::FFLAS_DIAG Diag,
            size_t r,
            size_t * P,
            size_t * Q) const  [inline]
```

check if the PLUQ factorization is correct.
Returns true if w - P(L(U(Q.v))) == 0

Parameters

A	
r	
P	
Q	

The documentation for this class was generated from the following file:

- checker_pluq.inl

12.34 Classic Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.35 Column Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.36 CompactElement< Element > Struct Template Reference

Public Types

- typedef Element type

12.36.1 Member Typedef Documentation

12.36.1.1 type

```
template<class Element>
typedef Element type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.37 CompactElement< double > Struct Reference

Public Types

- typedef int32_t type
- typedef double type

12.37.1 Member Typedef Documentation

12.37.1.1 type [1/2]

```
typedef int32_t type
```

12.37.1.2 type [2/2]

```
typedef double type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.38 CompactElement< float > Struct Reference

Public Types

- typedef int16_t type
- typedef float type

12.38.1 Member Typedef Documentation

12.38.1.1 type [1/2]

```
typedef int16_t type
```

12.38.1.2 type [2/2]

```
typedef float type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.39 CompactElement< int16_t > Struct Reference

Public Types

- typedef int8_t type
- typedef int16_t type

12.39.1 Member Typedef Documentation

12.39.1.1 type [1/2]

```
typedef int8_t type
```

12.39.1.2 type [2/2]

```
typedef int16_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.40 CompactElement< int32_t > Struct Reference

Public Types

- typedef int16_t type
- typedef int32_t type

12.40.1 Member Typedef Documentation

12.40.1.1 type [1/2]

```
typedef int16_t type
```

12.40.1.2 type [2/2]

```
typedef int32_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.41 CompactElement< int64_t > Struct Reference

Public Types

- typedef int32_t type
- typedef int64_t type

12.41.1 Member Typedef Documentation

12.41.1.1 type [1/2]

```
typedef int32_t type
```

12.41.1.2 type [2/2]

```
typedef int64_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.42 compatible_data_type< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true

12.42.1 Field Documentation

12.42.1.1 value

```
template<typename Field>
bool value = true  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.43 compatible_data_type< Givaro::ZRing< double > > Struct Reference

Static Public Attributes

- static constexpr bool value = false
- static constexpr bool value

12.43.1 Field Documentation

12.43.1.1 value [1/2]

```
bool value = false  [static], [constexpr]
```

12.43.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.44 compatible_data_type< Givaro::ZRing< float > > Struct Reference

Static Public Attributes

- static constexpr bool value = false
- static constexpr bool value

12.44.1 Field Documentation

12.44.1.1 value [1/2]

```
bool value = false  [static], [constexpr]
```

12.44.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.45 Compose< H1, H2 > Struct Template Reference

Public Member Functions

- Compose ()
- Compose (const Compose &other)
- Compose (const Sequential &S)
- Compose (size_t th1, size_t th2)
- Compose (const H1 &o1, const H2 &o2)
- H1 first_component () const
- H2 second_component () const

Friends

- std::ostream & operator<< (std::ostream &o, const Compose &c)

12.45.1 Constructor & Destructor Documentation

12.45.1.1 Compose() [1/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose ()  [inline]
```

12.45.1.2 Compose() [2/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const Compose< H1, H2 > & other)  [inline]
```

12.45.1.3 Compose() [3/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const Sequential & S)  [inline]
```

12.45.1.4 Compose() [4/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            size_t th1,
            size_t th2)  [inline]
```

12.45.1.5 Compose() [5/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const H1 & o1,
            const H2 & o2)  [inline]
```

12.45.2 Member Function Documentation

12.45.2.1 first_component()

```
template<typename H1 = Sequential, typename H2 = Sequential>
H1 first_component () const  [inline]
```

12.45.2.2 second_component()

```
template<typename H1 = Sequential, typename H2 = Sequential>
H2 second_component () const  [inline]
```

12.45.3 Friends And Related Symbol Documentation

12.45.3.1 operator<<

```
template<typename H1 = Sequential, typename H2 = Sequential>
std::ostream & operator<< (
            std::ostream & o,
            const Compose< H1, H2 > & c)  [friend]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.46 array< T >::const_iterator Class Reference

STL iterator class.

12.46.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.47 string::const_iterator Class Reference

STL iterator class.

12.47.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.48 vector< T >::const_iterator Class Reference

STL iterator class.

12.48.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.49 array< T >::const_reverse_iterator Class Reference

STL iterator class.

12.49.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.50 string::const_reverse_iterator Class Reference

STL iterator class.

12.50.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.51 vector< T >::const_reverse_iterator Class Reference

STL iterator class.

12.51.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.52 Simd128_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.52.1 Field Documentation

12.52.1.1 v

```
vect_t v
```

12.52.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int16.inl

12.53 Simd128_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.53.1 Field Documentation

12.53.1.1 v

```
vect_t v
```

12.53.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int32.inl

12.54 Simd128_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.54.1 Field Documentation

12.54.1.1 v

```
vect_t v
```

12.54.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int64.inl

12.55 Simd128_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.55.1 Field Documentation

12.55.1.1 v

```
vect_t v
```

12.55.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int16.inl

12.56 Simd128_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.56.1 Field Documentation

12.56.1.1 v

```
vect_t v
```

12.56.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int32.inl

12.57 Simd128_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.57.1 Field Documentation

12.57.1.1 v

```
vect_t v
```

12.57.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int64.inl

12.58 Simd256_impl< true, false, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.58.1 Field Documentation

12.58.1.1 v

```
vect_t v
```

12.58.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_double.inl

12.59 Simd256_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.59.1 Field Documentation

12.59.1.1 v

```
vect_t v
```

12.59.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int16.inl

12.60 Simd256_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.60.1 Field Documentation

12.60.1.1 v

```
vect_t v
```

12.60.1.2 t

```
scalar_t t
```

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.61 Simd256_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.61.1 Field Documentation

12.61.1.1 v

```
vect_t v
```

12.61.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int64.inl

12.62 Simd256_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.62.1 Field Documentation

12.62.1.1 v

```
vect_t v
```

12.62.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int16.inl

12.63 Simd256_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.63.1 Field Documentation

12.63.1.1 v

```
vect_t v
```

12.63.1.2 t

```
scalar_t t
```

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.64 Simd256_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.64.1 Field Documentation

12.64.1.1 v

```
vect_t v
```

12.64.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int64.inl

12.65 Simd512_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.65.1 Field Documentation

12.65.1.1 v

```
vect_t v
```

12.65.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd512_int64.inl

12.66 Simd512_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.66.1 Field Documentation

12.66.1.1 v

```
vect_t v
```

12.66.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd512_int64.inl

12.67 ConvertTo< T > Struct Template Reference

Force conversion to appropriate element type of ElementCategory T.
`#include <field-traits.h>`

12.67.1 Detailed Description

template<class T>
struct FFLAS::ModeCategories::ConvertTo< T >

Force conversion to appropriate element type of ElementCategory T.
e.g.

- ConvertTo<ElementCategories::MachineFloatTag> tries conversion of Modular<int> to Modular<double>
- ConvertTo<ElementCategories::FixedPrecIntTag> tries conversion of Modular<Integer> to Modular<RecInt<K> >
- ConvertTo<ElementCategories::ArbitraryPrecIntTag> tries conversion of Modular<Integer> to RNSInteger

The documentation for this struct was generated from the following file:

- field-traits.h

12.68 Coo< ValT, IdxT > Struct Template Reference

Public Types

- using Self = Coo<ValT, IdxT>

Public Member Functions

- Coo (ValT v, IdxT r, IdxT c)
- Coo ()=default
- Coo (const Self &)=default
- Coo (Self &&)=default
- Self & operator= (const Self &)=default
- Self & operator= (Self &&)=default

Data Fields

- ValT val = 0
- IdxT row = 0
- IdxT col = 0

12.68.1 Member Typedef Documentation

12.68.1.1 Self

```
template<class ValT, class IdxT>
using Self = Coo<ValT, IdxT>
```

12.68.2 Constructor & Destructor Documentation

12.68.2.1 Coo() [1/4]

```
template<class ValT, class IdxT>
Coo (
            ValT v,
            IdxT r,
            IdxT c)  [inline]
```

12.68.2.2 Coo() [2/4]

```
template<class ValT, class IdxT>
Coo ()  [default]
```

12.68.2.3 Coo() [3/4]

```
template<class ValT, class IdxT>
Coo (
            const Self & )  [default]
```

12.68.2.4 Coo() [4/4]

```
template<class ValT, class IdxT>
Coo (
            Self && )  [default]
```

12.68.3 Member Function Documentation

12.68.3.1 operator=() [1/2]

```
template<class ValT, class IdxT>
Self & operator= (
            const Self & )  [default]
```

12.68.3.2 operator=() [2/2]

```
template<class ValT, class IdxT>
Self & operator= (
            Self && )  [default]
```

12.68.4 Field Documentation

12.68.4.1 val

```
template<class ValT, class IdxT>
ValT val = 0
```

12.68.4.2 row

```
template<class ValT, class IdxT>
IdxT row = 0
```

12.68.4.3 col

```
template<class ValT, class IdxT>
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

12.69 Coo< Field > Struct Template Reference

```
#include <read_sparse.h>
```

Public Member Functions

- Coo ()=default
- Coo (typename Field::Element v, index_t r, index_t c)
- Coo (const Self &)=default
- Coo (Self &&)=default
- Self & operator= (const Self &)=default
- Self & operator= (Self &&)=default

Data Fields

- Field::Element val = 0
- index_t col = 0
- index_t row = 0
- bool deleted = false

12.69.1 Constructor & Destructor Documentation

12.69.1.1 Coo() [1/4]

```
template<class Field>
Coo () [default]
```

12.69.1.2 Coo() [2/4]

```
template<class Field>
Coo (
            typename Field::Element v,
            index_t r,
            index_t c) [inline]
```

12.69.1.3 Coo() [3/4]

```
template<class Field>
Coo (
            const Self & ) [default]
```

12.69.1.4 Coo() [4/4]

```
template<class Field>
Coo (
            Self && ) [default]
```

12.69.2 Member Function Documentation

12.69.2.1 operator=() [1/2]

```
template<class Field>
Self & operator= (
            const Self & )  [default]
```

12.69.2.2 operator=() [2/2]

```
template<class Field>
Self & operator= (
            Self && )  [default]
```

12.69.3 Field Documentation

12.69.3.1 val

```
template<class Field>
Field::Element val = 0
```

12.69.3.2 col

```
template<class Field>
index_t col = 0
```

12.69.3.3 row

```
template<class Field>
index_t row = 0
```

12.69.3.4 deleted

```
template<class Field>
bool deleted = false
```

The documentation for this struct was generated from the following file:

- read_sparse.h

12.70 Coo< ValT, IdxT > Struct Template Reference

Public Types

- using Self = Coo<ValT, IdxT>

Public Member Functions

- Coo (ValT v, IdxT r, IdxT c)
- Coo ()=default
- Coo (const Self &)=default
- Coo (Self &&)=default
- Self & operator= (const Self &)=default
- Self & operator= (Self &&)=default

Data Fields

- ValT val = 0
- IdxT row = 0
- IdxT col = 0

12.70.1 Member Typedef Documentation

12.70.1.1 Self

```
template<class ValT, class IdxT>
using Self = Coo<ValT, IdxT>
```

12.70.2 Constructor & Destructor Documentation

12.70.2.1 Coo() [1/4]

```
template<class ValT, class IdxT>
Coo (
            ValT v,
            IdxT r,
            IdxT c)  [inline]
```

12.70.2.2 Coo() [2/4]

```
template<class ValT, class IdxT>
Coo ()  [default]
```

12.70.2.3 Coo() [3/4]

```
template<class ValT, class IdxT>
Coo (
            const Self & )  [default]
```

12.70.2.4 Coo() [4/4]

```
template<class ValT, class IdxT>
Coo (
            Self && )  [default]
```

12.70.3 Member Function Documentation

12.70.3.1 operator=() [1/2]

```
template<class ValT, class IdxT>
Self & operator= (
            const Self & )  [default]
```

12.70.3.2 operator=() [2/2]

```
template<class ValT, class IdxT>
Self & operator= (
            Self && )  [default]
```

12.70.4 Field Documentation

12.70.4.1 val

```
template<class ValT, class IdxT>
ValT val = 0
```

12.70.4.2 row

```
template<class ValT, class IdxT>
IdxT row = 0
```

12.70.4.3 col

```
template<class ValT, class IdxT>
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

12.71 CooMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- FFLAS::Sparse< Field, SparseMatrix_t::COO, int16_t > * _coo16 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::COO, int32_t > * _coo32 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::COO, int64_t > * _coo64 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int16_t > * _coo16_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int32_t > * _coo32_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int64_t > * _coo64_zo = nullptr

12.71.1 Field Documentation

12.71.1.1 _coo16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int16_t>* _coo16 = nullptr
```

12.71.1.2 _coo32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int32_t>* _coo32 = nullptr
```

12.71.1.3 _coo64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int64_t>* _coo64 = nullptr
```

12.71.1.4 _coo16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int16_t>* _coo16_zo = nullptr
```

12.71.1.5 _coo32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int32_t>* _coo32_zo = nullptr
```

12.71.1.6 _coo64_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int64_t>* _coo64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.72 count_nonconst_lvalue_reference< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.73 count_nonconst_lvalue_reference< const T &, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n = count_nonconst_lvalue_reference<O...>::n

12.73.1 Field Documentation

12.73.1.1 n

```
template<typename T, typename...  O>
size_t n = count_nonconst_lvalue_reference<O...>::n  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.74 count_nonconst_lvalue_reference< T &, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n

12.74.1 Field Documentation

12.74.1.1 n

```
template<typename T, typename...  O>
size_t n  [static], [constexpr]
```

Initial value:

```
= std::integral_constant<size_t, 1>::value
                                  + count_nonconst_lvalue_reference<O...>::n
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.75 count_nonconst_lvalue_reference< T, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n = count_nonconst_lvalue_reference<O...>::n

12.75.1 Field Documentation

12.75.1.1 n

```
template<typename T, typename...  O>
size_t n = count_nonconst_lvalue_reference<O...>::n  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.76 count_nonconst_lvalue_reference<> Struct Reference

Static Public Attributes

- static constexpr size_t n = std::integral_constant<size_t, 0>::value

12.76.1 Field Documentation

12.76.1.1 n

```
size_t n = std::integral_constant<size_t, 0>::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.77 CsrMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- FFLAS::Sparse< Field, SparseMatrix_t::CSR, int16_t > * _csr16 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::CSR, int32_t > * _csr32 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::CSR, int64_t > * _csr64 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int16_t > * _csr16_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int32_t > * _csr32_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int64_t > * _csr64_zo = nullptr

12.77.1 Field Documentation

12.77.1.1 _csr16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int16_t>* _csr16 = nullptr
```

12.77.1.2 _csr32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int32_t>* _csr32 = nullptr
```

12.77.1.3 _csr64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int64_t>* _csr64 = nullptr
```

12.77.1.4 _csr16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int16_t>* _csr16_zo = nullptr
```

12.77.1.5 _csr32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int32_t>* _csr32_zo = nullptr
```

12.77.1.6 _csr64_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int64_t>* _csr64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.78 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.
`#include <field-traits.h>`

12.78.1 Detailed Description

Use standard field operations, but keeps track of bounds on input and output.
The documentation for this struct was generated from the following file:

- field-traits.h

12.79 DefaultTag Struct Reference

No specific mode of action: use standard field operations.
`#include <field-traits.h>`

12.79.1 Detailed Description

No specific mode of action: use standard field operations.
The documentation for this struct was generated from the following file:

- field-traits.h

12.80 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.
`#include <field-traits.h>`

12.80.1 Detailed Description

Performs field operations with delayed mod reductions. Ensures result is reduced.
The documentation for this struct was generated from the following file:

- field-traits.h

12.81 DivideAndConquer Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.82 ElementTraits< Element > Struct Template Reference

ElementTraits.
`#include <field-traits.h>`

Public Types

- typedef ElementCategories::GenericTag value

12.82.1 Detailed Description

template<class Element>
struct FFLAS::ElementTraits< Element >

ElementTraits.

12.82.2 Member Typedef Documentation

12.82.2.1 value

```
template<class Element>
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.83 ElementTraits< double > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineFloatTag value
- typedef ElementCategories::GenericTag value

12.83.1 Member Typedef Documentation

12.83.1.1 value [1/2]

```
typedef ElementCategories::MachineFloatTag value
```

12.83.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.84 ElementTraits< FFPACK::rns_double_elt > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::RNSElementTag value
- typedef ElementCategories::GenericTag value

12.84.1 Member Typedef Documentation

12.84.1.1 value [1/2]

```
typedef ElementCategories::RNSElementTag value
```

12.84.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.85 ElementTraits< float > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineFloatTag value
- typedef ElementCategories::GenericTag value

12.85.1 Member Typedef Documentation

12.85.1.1 value [1/2]

```
typedef ElementCategories::MachineFloatTag value
```

12.85.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.86 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::ArbitraryPrecIntTag value
- typedef ElementCategories::GenericTag value

12.86.1 Member Typedef Documentation

12.86.1.1 value [1/2]

```
typedef ElementCategories::ArbitraryPrecIntTag value
```

12.86.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.87 ElementTraits< int16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.87.1 Member Typedef Documentation

12.87.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.87.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.88 ElementTraits< int32_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.88.1 Member Typedef Documentation

12.88.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.88.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.89 ElementTraits< int64_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.89.1 Member Typedef Documentation

12.89.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.89.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.90 ElementTraits< int8_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.90.1 Member Typedef Documentation

12.90.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.90.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.91 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.91.1 Member Typedef Documentation

12.91.1.1 value [1/2]

```
template<size_t K>
typedef ElementCategories::FixedPrecIntTag value
```

12.91.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.92 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.92.1 Member Typedef Documentation

12.92.1.1 value [1/2]

```
template<size_t K, int MG>
typedef ElementCategories::FixedPrecIntTag value
```

12.92.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.93 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.93.1 Member Typedef Documentation

12.93.1.1 value [1/2]

```
template<size_t K>
typedef ElementCategories::FixedPrecIntTag value
```

12.93.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.94 ElementTraits< uint16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.94.1 Member Typedef Documentation

12.94.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.94.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.95 ElementTraits< uint32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.95.1 Member Typedef Documentation

12.95.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.95.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.96 ElementTraits< uint64_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.96.1 Member Typedef Documentation

12.96.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.96.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.97 ElementTraits< uint8_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.97.1 Member Typedef Documentation

12.97.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.97.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.98 EllMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- FFLAS::Sparse< Field, SparseMatrix_t::ELL, int16_t > * _ell16 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::ELL, int32_t > * _ell32 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::ELL, int64_t > * _ell64 = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int16_t > * _ell16_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int32_t > * _ell32_zo = nullptr
- FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int64_t > * _ell64_zo = nullptr

12.98.1 Field Documentation

12.98.1.1 _ell16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int16_t>* _ell16 = nullptr
```

12.98.1.2 _ell32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int32_t>* _ell32 = nullptr
```

12.98.1.3 _ell64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int64_t>* _ell64 = nullptr
```

12.98.1.4 _ell16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int16_t>* _ell16_zo = nullptr
```

12.98.1.5 _ell32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int32_t>* _ell32_zo = nullptr
```

12.98.1.6 _ell64_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int64_t>* _ell64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.99 Failure Class Reference

A precondtion failed.

```
#include <debug.h>
```

Public Member Functions

- Failure ()
- void operator() (const char *function, int line, const char *check)
- void operator() (const char *function, const char *file, int line, const char *check)
- void setErrorStream (std::ostream &stream)
- std::ostream & print (std::ostream &o) const

Protected Attributes

- std::ostream * _errorStream

12.99.1 Detailed Description

A precondtion failed.
The `throw` mechanism is usually used here as in

```
if (!check)
failure()(__func__,__LINE__,"this check just failed");
```

The parameters of the constructor help debugging.

12.99.2 Constructor & Destructor Documentation

12.99.2.1 Failure()

```
Failure ()  [inline]
```

12.99.3 Member Function Documentation

12.99.3.1 operator()() [1/2]

```
void operator() (
            const char * function,
            int line,
            const char * check)  [inline]
```

A precondtion failed.

Parameters

function	usually __func__, the function that threw the error
line	usually __LINE__, the line where it happened
check	a string telling what failed.

12.99.3.2 operator()() [2/2]

```
void operator() (
            const char * function,
            const char * file,
            int line,
            const char * check)  [inline]
```

A precondtion failed. The parameter help debugging. This is not much different from the previous except we can digg faster in the file where the exception was triggered.

Parameters

function	usually __func__, the function that threw the error
file	usually __FILE__, the file where this function is
line	usually __LINE__, the line where it happened
check	a string telling what failed.

12.99.3.3 setErrorStream()

```
void setErrorStream (
            std::ostream & stream)
```

12.99.3.4 print()

```
std::ostream & print (
            std::ostream & o) const  [inline]
```

overload the virtual print of LinboxError.

Parameters

o	output stream

12.99.4 Field Documentation

12.99.4.1 _errorStream

```
std::ostream* _errorStream  [protected]
```

The documentation for this class was generated from the following file:

- debug.h

12.100 FailureCharpolyCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.101 FailureDetCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.102 FailureFgemmCheck Class Reference

`#include <checkers_fflas.h>`

The documentation for this class was generated from the following file:

- checkers_fflas.h

12.103 FailureInvertCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.104 FailurePLUQCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.105 FailureTrsmCheck Class Reference

`#include <checkers_fflas.h>`

The documentation for this class was generated from the following file:

- checkers_fflas.h

12.106 FieldSimd< _Field > Class Template Reference

Public Types

- using Field = _Field
- using Element = typename Field::Element
- using simd = Simd<typename _Field::Element>
- using vect_t = typename simd::vect_t
- using scalar_t = typename simd::scalar_t

Public Member Functions

- FieldSimd (const Field &f)
- FieldSimd (const Self &)=default
- FieldSimd (Self &&)=default
- Self & operator= (const Self &)=default
- Self & operator= (Self &&)=default
- INLINE vect_t init (vect_t &x, const vect_t a) const
- INLINE vect_t init (const vect_t a) const
- INLINE vect_t add (vect_t &c, const vect_t a, const vect_t b)
- INLINE vect_t add (const vect_t a, const vect_t b)
- INLINE vect_t addin (vect_t &a, const vect_t b) const
- INLINE vect_t add_r (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t add_r (const vect_t a, const vect_t b) const
- INLINE vect_t addin_r (vect_t &a, const vect_t b) const
- INLINE vect_t sub (vect_t &c, const vect_t a, const vect_t b)
- INLINE vect_t sub (const vect_t a, const vect_t b)
- INLINE vect_t subin (vect_t &a, const vect_t b) const
- INLINE vect_t sub_r (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t sub_r (const vect_t a, const vect_t b) const
- INLINE vect_t subin_r (vect_t &a, const vect_t b) const
- INLINE vect_t zero (vect_t &x) const
- INLINE vect_t zero () const
- INLINE vect_t mod (vect_t &c) const
- INLINE vect_t mul (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t mul (const vect_t a, const vect_t b) const
- INLINE vect_t mulin (vect_t &a, const vect_t b) const
- INLINE vect_t mul_r (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t mul_r (const vect_t a, const vect_t b) const
- INLINE vect_t axpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const
- INLINE vect_t axpy (const vect_t c, const vect_t a, const vect_t b) const
- INLINE vect_t axpyin (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t axpy_r (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const
- INLINE vect_t axpy_r (const vect_t c, const vect_t a, const vect_t b) const
- INLINE vect_t axpyin_r (vect_t &c, const vect_t a, const vect_t b) const
- INLINE vect_t maxpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const
- INLINE vect_t maxpy (const vect_t c, const vect_t a, const vect_t b) const
- INLINE vect_t maxpyin (vect_t &c, const vect_t a, const vect_t b) const

Static Public Attributes

- static const constexpr size_t vect_size = simd::vect_size
- static const constexpr size_t alignment = simd::alignment

12.106.1 Member Typedef Documentation

12.106.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.106.1.2 Element

```
template<class _Field>
using Element = typename Field::Element
```

12.106.1.3 simd

```
template<class _Field>
using simd = Simd<typename _Field::Element>
```

12.106.1.4 vect_t

```
template<class _Field>
using vect_t = typename simd::vect_t
```

12.106.1.5 scalar_t

```
template<class _Field>
using scalar_t = typename simd::scalar_t
```

12.106.2 Constructor & Destructor Documentation

12.106.2.1 FieldSimd() [1/3]

```
template<class _Field>
FieldSimd (
            const Field & f)  [inline]
```

12.106.2.2 FieldSimd() [2/3]

```
template<class _Field>
FieldSimd (
            const Self & )  [default]
```

12.106.2.3 FieldSimd() [3/3]

```
template<class _Field>
FieldSimd (
            Self && )  [default]
```

12.106.3 Member Function Documentation

12.106.3.1 operator=() [1/2]

```
template<class _Field>
Self & operator= (
            const Self & )  [default]
```

12.106.3.2 operator=() [2/2]

```
template<class _Field>
Self & operator= (
            Self && )  [default]
```

12.106.3.3 init() [1/2]

```
template<class _Field>
INLINE vect_t init (
            vect_t & x,
            const vect_t a) const  [inline]
```

12.106.3.4 init() [2/2]

```
template<class _Field>
INLINE vect_t init (
            const vect_t a) const  [inline]
```

12.106.3.5 add() [1/2]

```
template<class _Field>
INLINE vect_t add (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.6 add() [2/2]

```
template<class _Field>
INLINE vect_t add (
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.7 addin()

```
template<class _Field>
INLINE vect_t addin (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.8 add_r() [1/2]

```
template<class _Field>
INLINE vect_t add_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.9 add_r() [2/2]

```
template<class _Field>
INLINE vect_t add_r (
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.10 addin_r()

```
template<class _Field>
INLINE vect_t addin_r (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.11 sub() [1/2]

```
template<class _Field>
INLINE vect_t sub (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.12 sub() [2/2]

```
template<class _Field>
INLINE vect_t sub (
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.13 subin()

```
template<class _Field>
INLINE vect_t subin (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.14 sub_r() [1/2]

```
template<class _Field>
INLINE vect_t sub_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.15 sub_r() [2/2]

```
template<class _Field>
INLINE vect_t sub_r (
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.16 subin_r()

```
template<class _Field>
INLINE vect_t subin_r (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.17 zero() [1/2]

```
template<class _Field>
INLINE vect_t zero (
            vect_t & x) const  [inline]
```

12.106.3.18 zero() [2/2]

```
template<class _Field>
INLINE vect_t zero () const  [inline]
```

12.106.3.19 mod()

```
template<class _Field>
INLINE vect_t mod (
            vect_t & c) const  [inline]
```

12.106.3.20 mul() [1/2]

```
template<class _Field>
INLINE vect_t mul (
             vect_t & c,
             const vect_t a,
             const vect_t b) const  [inline]
```

12.106.3.21 mul() [2/2]

```
template<class _Field>
INLINE vect_t mul (
             const vect_t a,
             const vect_t b) const  [inline]
```

12.106.3.22 mulin()

```
template<class _Field>
INLINE vect_t mulin (
             vect_t & a,
             const vect_t b) const  [inline]
```

12.106.3.23 mul_r() [1/2]

```
template<class _Field>
INLINE vect_t mul_r (
             vect_t & c,
             const vect_t a,
             const vect_t b) const  [inline]
```

12.106.3.24 mul_r() [2/2]

```
template<class _Field>
INLINE vect_t mul_r (
             const vect_t a,
             const vect_t b) const  [inline]
```

12.106.3.25 axpy() [1/2]

```
template<class _Field>
INLINE vect_t axpy (
             vect_t & r,
             const vect_t a,
             const vect_t b,
             const vect_t c) const  [inline]
```

12.106.3.26 axpy() [2/2]

```
template<class _Field>
INLINE vect_t axpy (
             const vect_t c,
             const vect_t a,
             const vect_t b) const  [inline]
```

12.106.3.27 axpyin()

```
template<class _Field>
INLINE vect_t axpyin (
             vect_t & c,
```

```
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.28 axpy_r() [1/2]

```
template<class _Field>
INLINE vect_t axpy_r (
            vect_t & r,
            const vect_t a,
            const vect_t b,
            const vect_t c) const  [inline]
```

12.106.3.29 axpy_r() [2/2]

```
template<class _Field>
INLINE vect_t axpy_r (
            const vect_t c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.30 axpyin_r()

```
template<class _Field>
INLINE vect_t axpyin_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.31 maxpy() [1/2]

```
template<class _Field>
INLINE vect_t maxpy (
            vect_t & r,
            const vect_t a,
            const vect_t b,
            const vect_t c) const  [inline]
```

12.106.3.32 maxpy() [2/2]

```
template<class _Field>
INLINE vect_t maxpy (
            const vect_t c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.33 maxpyin()

```
template<class _Field>
INLINE vect_t maxpyin (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.4 Field Documentation

12.106.4.1 vect_size

```
template<class _Field>
const constexpr size_t vect_size = simd::vect_size  [static], [constexpr]
```

12.106.4.2 alignment

```
template<class _Field>
const constexpr size_t alignment = simd::alignment  [static], [constexpr]
```

The documentation for this class was generated from the following file:

- simd_modular.inl

12.107 FieldTraits< Field > Struct Template Reference

FieldTrait.
`#include <field-traits.h>`

Public Types

- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false

12.107.1 Detailed Description

template<class Field>
struct FFLAS::FieldTraits< Field >

FieldTrait.

12.107.2 Member Typedef Documentation

12.107.2.1 category

```
template<class Field>
typedef FieldCategories::GenericTag category
```

12.107.3 Field Documentation

12.107.3.1 balanced

```
template<class Field>
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.108 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.108.1 Member Typedef Documentation

12.108.1.1 category [1/2]

```
template<typename T>
typedef FieldCategories::UnparametricTag category
```

12.108.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.108.2 Field Documentation

12.108.2.1 balanced [1/2]

```
template<typename T>
const bool balanced = false  [static]
```

12.108.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.109 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.109.1 Member Typedef Documentation

12.109.1.1 category [1/2]

```
template<typename T>
typedef FieldCategories::ModularTag category
```

12.109.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.109.2 Field Documentation

12.109.2.1 balanced [1/2]

```
template<typename T>
const bool balanced = false  [static]
```

12.109.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.110 FieldTraits< Givaro::Modular< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.110.1 Member Typedef Documentation

12.110.1.1 category [1/2]

```
template<class Element>
typedef FieldCategories::ModularTag category
```

12.110.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.110.2 Field Documentation

12.110.2.1 balanced [1/2]

```
template<class Element>
const bool balanced = false  [static]
```

12.110.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.111 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = true
- static const bool balanced

12.111.1 Member Typedef Documentation

12.111.1.1 category [1/2]

```
template<class Element>
typedef FieldCategories::ModularTag category
```

12.111.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.111.2 Field Documentation

12.111.2.1 balanced [1/2]

```
template<class Element>
const bool balanced = true  [static]
```

12.111.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.112 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.112.1 Member Typedef Documentation

12.112.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.112.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.112.2 Field Documentation

12.112.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.112.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.113 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.113.1 Member Typedef Documentation

12.113.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.113.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.113.2 Field Documentation

12.113.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.113.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.114 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.114.1 Member Typedef Documentation

12.114.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.114.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.114.2 Field Documentation

12.114.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.114.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.115 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.115.1 Member Typedef Documentation

12.115.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.115.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.115.2 Field Documentation

12.115.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.115.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.116 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.116.1 Member Typedef Documentation

12.116.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.116.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.116.2 Field Documentation

12.116.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.116.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.117 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.117.1 Member Typedef Documentation

12.117.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.117.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.117.2 Field Documentation

12.117.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.117.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.118 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.118.1 Member Typedef Documentation

12.118.1.1 category [1/2]

```
template<size_t K>
typedef FieldCategories::UnparametricTag category
```

12.118.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.118.2 Field Documentation

12.118.2.1 balanced [1/2]

```
template<size_t K>
const bool balanced = false  [static]
```

12.118.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.119 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.119.1 Member Typedef Documentation

12.119.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.119.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.119.2 Field Documentation

12.119.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.119.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.120 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.120.1 Member Typedef Documentation

12.120.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.120.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.120.2 Field Documentation

12.120.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.120.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.121 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.121.1 Member Typedef Documentation

12.121.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.121.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.121.2 Field Documentation

12.121.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.121.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.122 Fixed Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.123 FixedPrecIntTag Struct Reference

Fixed precision integers above machine precision: Givaro::recInt.

`#include <field-traits.h>`

12.123.1 Detailed Description

Fixed precision integers above machine precision: Givaro::recInt.
The documentation for this struct was generated from the following file:

- field-traits.h

12.124 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution Class Reference

Public Types

- using IntType = typename make_unsigned_int<Element>::type

Public Member Functions

- FloatingPointTestDistribution ()
- template<class Generator>
  Element operator() (Generator &g)

12.124.1 Member Typedef Documentation

12.124.1.1 IntType

```
template<typename Element>
using IntType = typename make_unsigned_int<Element>::type
```

12.124.2 Constructor & Destructor Documentation

12.124.2.1 FloatingPointTestDistribution()

```
template<typename Element>
FloatingPointTestDistribution () [inline]
```

12.124.3 Member Function Documentation

12.124.3.1 operator()()

```
template<typename Element>
template<class Generator>
Element operator() (
            Generator & g) [inline]
```

The documentation for this class was generated from the following file:

- test-simd.C

12.125 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference

Public Member Functions

- ForStrategy1D (const blocksize_t n, const ParSeqHelper::Parallel< Cut, Param > H)
- ForStrategy1D (const blocksize_t b, const blocksize_t e, const ParSeqHelper::Parallel< Cut, Param > H)
- void build (const blocksize_t n, const ParSeqHelper::Parallel< Cut, Param > H)
- blocksize_t initialize ()
- bool isTerminated () const
- blocksize_t begin () const

- blocksize_t end () const
- blocksize_t numblocks () const
- blocksize_t blockindex () const
- blocksize_t operator++ ()

Protected Attributes

- blocksize_t ibeg
- blocksize_t iend
- blocksize_t current
- blocksize_t firstBlockSize
- blocksize_t lastBlockSize
- blocksize_t changeBS
- blocksize_t numBlock

12.125.1 Constructor & Destructor Documentation

12.125.1.1 ForStrategy1D() [1/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy1D (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.1.2 ForStrategy1D() [2/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy1D (
            const blocksize_t b,
            const blocksize_t e,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.2 Member Function Documentation

12.125.2.1 build()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
void build (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.2.2 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ()  [inline]
```

12.125.2.3 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated () const  [inline]
```

12.125.2.4 begin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t begin () const  [inline]
```

12.125.2.5 end()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t end () const  [inline]
```

12.125.2.6 numblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numblocks () const  [inline]
```

12.125.2.7 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex () const  [inline]
```

12.125.2.8 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ ()  [inline]
```

12.125.3 Field Documentation

12.125.3.1 ibeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibeg  [protected]
```

12.125.3.2 iend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend  [protected]
```

12.125.3.3 current

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current  [protected]
```

12.125.3.4 firstBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t firstBlockSize  [protected]
```

12.125.3.5 lastBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastBlockSize  [protected]
```

12.125.3.6 changeBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeBS  [protected]
```

12.125.3.7 numBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numBlock  [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.126 ForStrategy2D< blocksize_t, Cut, Param > Struct Template Reference

Public Member Functions

- ForStrategy2D (const blocksize_t m, const blocksize_t n, const ParSeqHelper::Parallel< Cut, Param > H)
- blocksize_t initialize ()
- bool isTerminated () const
- blocksize_t ibegin () const
- blocksize_t jbegin () const
- blocksize_t iend () const
- blocksize_t jend () const
- blocksize_t operator++ ()
- blocksize_t rownumblocks () const
- blocksize_t colnumblocks () const
- blocksize_t blockindex () const
- blocksize_t rowblockindex () const
- blocksize_t colblockindex () const

Protected Attributes

- blocksize_t _ibeg
- blocksize_t _iend
- blocksize_t _jbeg
- blocksize_t _jend
- blocksize_t rowBlockSize
- blocksize_t colBlockSize
- blocksize_t current
- blocksize_t lastRBS
- blocksize_t lastCBS
- blocksize_t changeRBS
- blocksize_t changeCBS
- blocksize_t numRowBlock
- blocksize_t numColBlock
- blocksize_t BLOCKS

Friends

- std::ostream & operator<< (std::ostream &out, const ForStrategy2D &FS2D)

12.126.1 Constructor & Destructor Documentation

12.126.1.1 ForStrategy2D()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy2D (
            const blocksize_t m,
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.126.2 Member Function Documentation

12.126.2.1 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ()  [inline]
```

12.126.2.2 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated () const  [inline]
```

12.126.2.3 ibegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibegin () const  [inline]
```

12.126.2.4 jbegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jbegin () const  [inline]
```

12.126.2.5 iend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend () const  [inline]
```

12.126.2.6 jend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jend () const  [inline]
```

12.126.2.7 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ ()  [inline]
```

12.126.2.8 rownumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rownumblocks () const  [inline]
```

12.126.2.9 colnumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colnumblocks () const  [inline]
```

12.126.2.10 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex () const  [inline]
```

12.126.2.11 rowblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowblockindex () const  [inline]
```

12.126.2.12 colblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colblockindex () const  [inline]
```

12.126.3 Friends And Related Symbol Documentation

12.126.3.1 operator<<

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
std::ostream & operator<< (
            std::ostream & out,
            const ForStrategy2D< blocksize_t, Cut, Param > & FS2D)  [friend]
```

12.126.4 Field Documentation

12.126.4.1 _ibeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _ibeg  [protected]
```

12.126.4.2 _iend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _iend  [protected]
```

12.126.4.3 _jbeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jbeg  [protected]
```

12.126.4.4 _jend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jend  [protected]
```

12.126.4.5 rowBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowBlockSize  [protected]
```

12.126.4.6 colBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colBlockSize  [protected]
```

12.126.4.7 current

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current  [protected]
```

12.126.4.8 lastRBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastRBS  [protected]
```

12.126.4.9 lastCBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastCBS  [protected]
```

12.126.4.10 changeRBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeRBS  [protected]
```

12.126.4.11 changeCBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeCBS  [protected]
```

12.126.4.12 numRowBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numRowBlock  [protected]
```

12.126.4.13 numColBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numColBlock  [protected]
```

12.126.4.14 BLOCKS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t BLOCKS  [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.127 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.128 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.129 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.130 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.131 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.132 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.133 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.134 ftrmmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.135 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.136 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.137 ftrmmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.138 ftrmmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.139 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.140 ftrmmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.141 ftrmmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.142 ftrmmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.143 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.144 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.145 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.146 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.147 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

12.147.1 Detailed Description

template<class Element>
class FFLAS::Protected::ftrsmLeftUpperNoTransNonUnit< Element >

Computes the maximal size for delaying the modular reduction in a triangular system resolution.
Compute the maximal dimension k, such that a unit diagonal triangular system of dimension k can be solved over Z without overflow of the underlying floating point representation.

Bibliography • Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Parameters

F	Finite Field/Ring of the computation

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.148 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.149 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.150 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.151 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.152 ftrsmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.153 ftrsmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.154 ftrsmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.155 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.156 ftrsmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.157 ftrsmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.158 ftrsmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.159 GenericTag Struct Reference

default is generic
`#include <field-traits.h>`

12.159.1 Detailed Description

default is generic
The documentation for this struct was generated from the following file:

- field-traits.h

12.160 GenericTag Struct Reference

generic ring.
`#include <field-traits.h>`

12.160.1 Detailed Description

generic ring.
The documentation for this struct was generated from the following file:

- field-traits.h

12.161 Grain Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.162 has_minus_eq_impl< C > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Static Public Attributes

- static constexpr bool value = type::value

12.162.1 Field Documentation

12.162.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.163 has_minus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.163.1 Field Documentation

12.163.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.164 has_mul_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.164.1 Field Documentation

12.164.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.165 has_mul_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.165.1 Field Documentation

12.165.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.166 has_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value

12.166.1 Field Documentation

12.166.1.1 value

```
template<class T>
bool value  [static], [constexpr]
```

Initial value:

```
= (has_plus<T>::value && has_minus<T>::value && has_equal<T>::value &&
                                    has_plus_eq<T>::value && has_minus_eq<T>::value && has_mul<T>::value
      && has_mul_eq<T>::value)
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.167 has_plus_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.167.1 Field Documentation

12.167.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.168 has_plus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.168.1 Field Documentation

12.168.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.169 HelperFlag Struct Reference

```
#include <fflas_sparse.h>
```

Static Public Attributes

- static constexpr uint64_t none = 0_ui64
- static constexpr uint64_t coo = 1_ui64
- static constexpr uint64_t csr = 1_ui64 << 1
- static constexpr uint64_t ell = 1_ui64 << 2
- static constexpr uint64_t aut = 1_ui64 << 32
- static constexpr uint64_t pm1 = 1_ui64 << 33

12.169.1 Field Documentation

12.169.1.1 none

```
uint64_t none = 0_ui64  [static], [constexpr]
```

12.169.1.2 coo

```
uint64_t coo = 1_ui64  [static], [constexpr]
```

12.169.1.3 csr

```
uint64_t csr = 1_ui64 << 1  [static], [constexpr]
```

12.169.1.4 ell

```
uint64_t ell = 1_ui64 << 2  [static], [constexpr]
```

12.169.1.5 aut

```
uint64_t aut = 1_ui64 << 32  [static], [constexpr]
```

12.169.1.6 pm1

```
uint64_t pm1 = 1_ui64 << 33  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.170 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.171 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p
- double invp
- double min
- double max
- int64_t pow50rem

12.171.1 Constructor & Destructor Documentation

12.171.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.171.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.171.2 Field Documentation

12.171.2.1 p

```
template<class Field>
Field::Element p
```

12.171.2.2 invp

```
template<class Field>
double invp
```

12.171.2.3 min

```
template<class Field>
double min
```

12.171.2.4 max

```
template<class Field>
double max
```

12.171.2.5 pow50rem

```
template<class Field>
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.172 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

12.172.1 Constructor & Destructor Documentation

12.172.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.172.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.172.2 Field Documentation

12.172.2.1 p

```
template<class Field>
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.173 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

12.173.1 Constructor & Destructor Documentation

12.173.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.173.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.173.2 Field Documentation

12.173.2.1 p

```
template<class Field>
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.174 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p
- Field::Element invp
- Field::Element min
- Field::Element max

12.174.1 Constructor & Destructor Documentation

12.174.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.174.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.174.2 Field Documentation

12.174.2.1 p

```
template<class Field>
Field::Element p
```

12.174.2.2 invp

```
template<class Field>
Field::Element invp
```

12.174.2.3 min

```
template<class Field>
Field::Element min
```

12.174.2.4 max

```
template<class Field>
Field::Element max
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.175 Hybrid Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.176 Info Struct Reference

Public Member Functions

- Info (uint64_t it, uint64_t s, uint64_t p)
- Info ()=default
- Info (const Info &)=default
- Info (Info &&)=default
- Info & operator= (const Info &)=default
- Info & operator= (Info &&)=default

Data Fields

- uint64_t size = 0
- uint64_t perm = 0
- uint64_t begin = 0

12.176.1 Constructor & Destructor Documentation

12.176.1.1 Info() [1/4]

```
Info (
            uint64_t it,
            uint64_t s,
            uint64_t p)  [inline]
```

12.176.1.2 Info() [2/4]

```
Info ()  [default]
```

12.176.1.3 Info() [3/4]

```
Info (
            const Info & )  [default]
```

12.176.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

12.176.2 Member Function Documentation

12.176.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

12.176.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

12.176.3 Field Documentation

12.176.3.1 size

```
uint64_t size = 0
```

12.176.3.2 perm

```
uint64_t perm = 0
```

12.176.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

12.177 Info Struct Reference

Public Member Functions

- Info (uint64_t it, uint64_t s, uint64_t p)
- Info ()=default
- Info (const Info &)=default
- Info (Info &&)=default
- Info & operator= (const Info &)=default
- Info & operator= (Info &&)=default

Data Fields

- uint64_t size = 0
- uint64_t perm = 0
- uint64_t begin = 0

12.177.1 Constructor & Destructor Documentation

12.177.1.1 Info() [1/4]

```
Info (
            uint64_t it,
            uint64_t s,
            uint64_t p)  [inline]
```

12.177.1.2 Info() [2/4]

```
Info ()  [default]
```

12.177.1.3 Info() [3/4]

```
Info (
            const Info & )  [default]
```

12.177.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

12.177.2 Member Function Documentation

12.177.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

12.177.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

12.177.3 Field Documentation

12.177.3.1 size

```
uint64_t size = 0
```

12.177.3.2 perm

```
uint64_t perm = 0
```

12.177.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

12.178 is_all_same< Args > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.179 is_all_same< T, Args... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = ALL<std::is_same<typename decay<T>::type, typename decay<Args>↩ ::type>::value...>::value

12.179.1 Field Documentation

12.179.1.1 value

```
template<typename T, typename...  Args>
bool value = ALL<std::is_same<typename decay<T>::type, typename decay<Args>::type>::value...>↩
::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.180 is_all_same<> Struct Reference

Static Public Attributes

- static constexpr bool value = true

12.180.1 Field Documentation

12.180.1.1 value

```
bool value = true  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.181 is_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using type = std::integral_constant<bool, false>

Static Public Attributes

- static const constexpr bool value = false

12.181.1 Member Typedef Documentation

12.181.1.1 type

```
template<class T>
using type = std::integral_constant<bool, false>
```

12.181.2 Field Documentation

12.181.2.1 value

```
template<class T>
const constexpr bool value = false  [static], [constexpr]
```

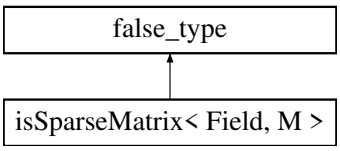
The documentation for this struct was generated from the following file:

- fflas_simd.h

12.182 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



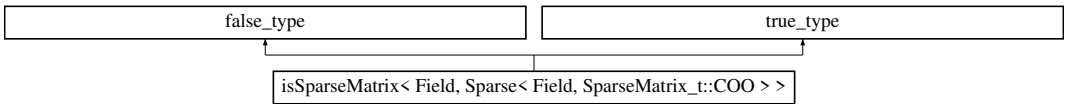
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.183 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >:



The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.184 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >:

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.185 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >:

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.186 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference

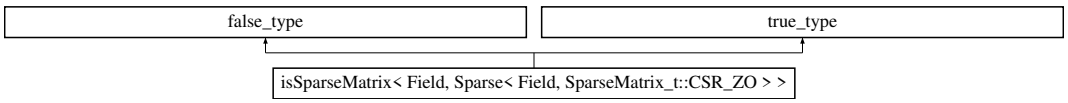
`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >:

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.187 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >:

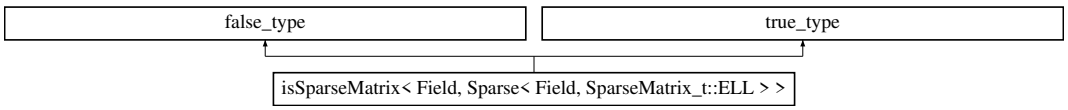


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.188 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >:

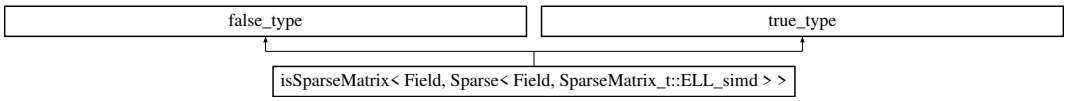


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.189 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >:

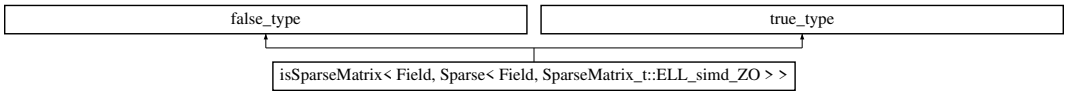


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.190 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >:

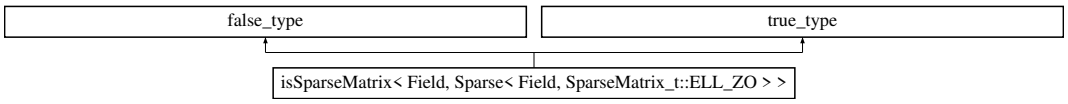


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.191 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >:

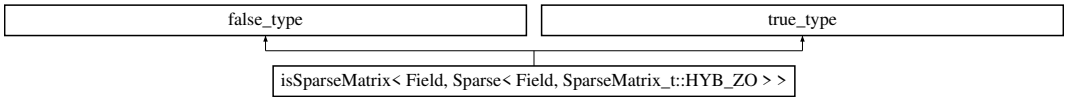


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.192 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >:

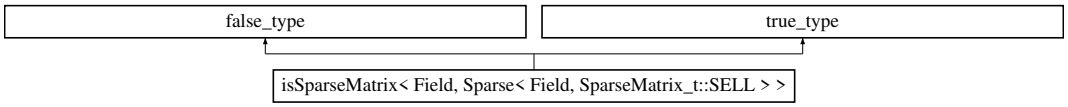


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.193 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >:

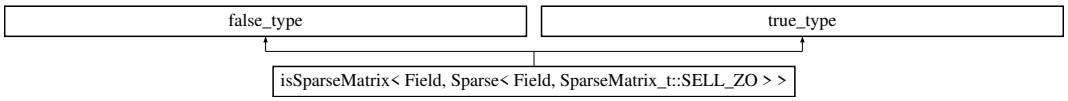


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.194 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >:

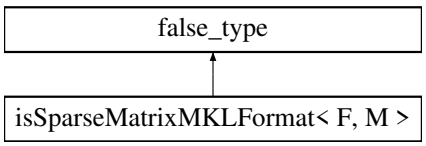


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.195 isSparseMatrixMKLFormat< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrixMKLFormat< F, M >:

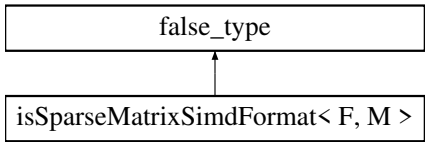


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.196 isSparseMatrixSimdFormat< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isSparseMatrixSimdFormat< F, M >:

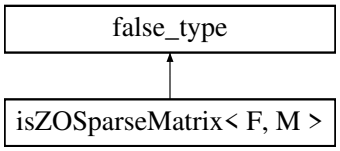


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.197 isZOSparseMatrix< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< F, M >:

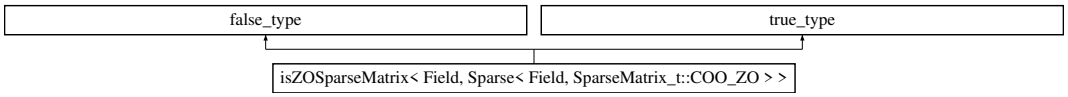


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.198 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >:

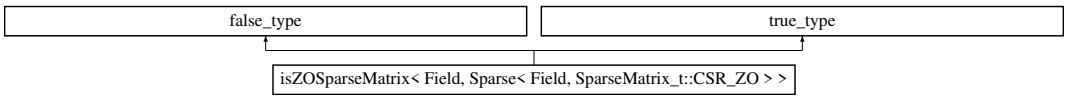


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.199 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >:

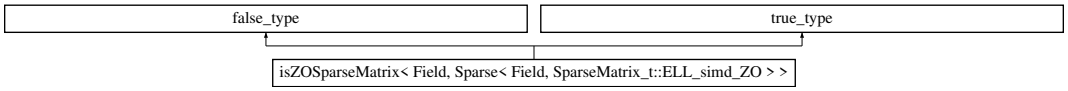


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.200 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >:

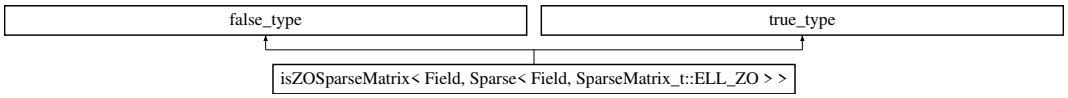


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.201 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >:

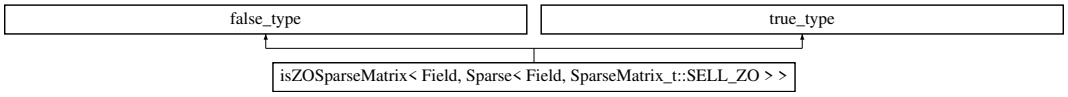


The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.202 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference

`#include <sparse_matrix_traits.h>`
Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >:



The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.203 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.204 array< T >::iterator Class Reference

STL iterator class.

12.204.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.205 string::iterator Class Reference

STL iterator class.

12.205.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.206 vector< T >::iterator Class Reference

STL iterator class.

12.206.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.207 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.
`#include <field-traits.h>`

12.207.1 Detailed Description

Performs field operations with delayed mod only when necessary. Result may not be reduced.
The documentation for this struct was generated from the following file:

- field-traits.h

12.208 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- flimits.h

12.209 limits< char > Struct Reference

`#include <flimits.h>`

Public Types

- typedef char T

Static Public Member Functions

- static constexpr char max () noexcept
- static constexpr char min () noexcept
- static constexpr int32_t digits () noexcept

12.209.1 Member Typedef Documentation

12.209.1.1 T

`typedef char T`

12.209.2 Member Function Documentation

12.209.2.1 max()

`static constexpr char max () [inline], [static], [constexpr], [noexcept]`

12.209.2.2 min()

`static constexpr char min () [inline], [static], [constexpr], [noexcept]`

12.209.2.3 digits()

`static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]`
The documentation for this struct was generated from the following file:

- flimits.h

12.210 limits< double > Struct Reference

`#include <flimits.h>`

Public Types

- typedef double T

Static Public Member Functions

- static constexpr int64_t max () noexcept
- static constexpr int64_t min () noexcept
- static constexpr int32_t digits () noexcept

12.210.1 Member Typedef Documentation

12.210.1.1 T

```
typedef double T
```

12.210.2 Member Function Documentation

12.210.2.1 max()

```
static constexpr int64_t max ()  [inline], [static], [constexpr], [noexcept]
```

12.210.2.2 min()

```
static constexpr int64_t min ()  [inline], [static], [constexpr], [noexcept]
```

12.210.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.211 limits< float > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef float T

Static Public Member Functions

- static constexpr int32_t max () noexcept
- static constexpr int32_t min () noexcept
- static constexpr int32_t digits () noexcept

12.211.1 Member Typedef Documentation

12.211.1.1 T

```
typedef float T
```

12.211.2 Member Function Documentation

12.211.2.1 max()

```
static constexpr int32_t max ()  [inline], [static], [constexpr], [noexcept]
```

12.211.2.2 min()

```
static constexpr int32_t min ()  [inline], [static], [constexpr], [noexcept]
```

12.211.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.212 limits< Givaro::Integer > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef Givaro::Integer T

Static Public Member Functions

- static constexpr int max () noexcept
- static constexpr int min () noexcept

12.212.1 Member Typedef Documentation

12.212.1.1 T

```
typedef Givaro::Integer T
```

12.212.2 Member Function Documentation

12.212.2.1 max()

```
static constexpr int max ()  [inline], [static], [constexpr], [noexcept]
```

12.212.2.2 min()

```
static constexpr int min ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.213 limits< int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef int T

Static Public Member Functions

- static constexpr int max () noexcept
- static constexpr int min () noexcept
- static constexpr int32_t digits () noexcept

12.213.1 Member Typedef Documentation

12.213.1.1 T

```
typedef int T
```

12.213.2 Member Function Documentation

12.213.2.1 max()

```
static constexpr int max ()  [inline], [static], [constexpr], [noexcept]
```

12.213.2.2 min()

```
static constexpr int min ()  [inline], [static], [constexpr], [noexcept]
```

12.213.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.214 limits< long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef long T

Static Public Member Functions

- static constexpr long max () noexcept
- static constexpr long min () noexcept
- static constexpr int32_t digits () noexcept

12.214.1 Member Typedef Documentation

12.214.1.1 T

```
typedef long T
```

12.214.2 Member Function Documentation

12.214.2.1 max()

```
static constexpr long max ()  [inline], [static], [constexpr], [noexcept]
```

12.214.2.2 min()

```
static constexpr long min ()  [inline], [static], [constexpr], [noexcept]
```

12.214.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.215 limits< long long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef long long T

Static Public Member Functions

- static constexpr long long max () noexcept
- static constexpr long long min () noexcept
- static constexpr int32_t digits () noexcept

12.215.1 Member Typedef Documentation

12.215.1.1 T

```
typedef long long T
```

12.215.2 Member Function Documentation

12.215.2.1 max()

```
static constexpr long long max ()  [inline], [static], [constexpr], [noexcept]
```

12.215.2.2 min()

```
static constexpr long long min ()  [inline], [static], [constexpr], [noexcept]
```

12.215.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.216 limits< RecInt::rint< K > > Struct Template Reference

```
#include <flimits.h>
```

Public Types

- typedef RecInt::ruint< K > T

Static Public Member Functions

- static constexpr RecInt::rint< K > max () noexcept
- static constexpr RecInt::rint< K > min () noexcept

12.216.1 Member Typedef Documentation

12.216.1.1 T

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.216.2 Member Function Documentation

12.216.2.1 max()

```
template<size_t K>
static constexpr RecInt::rint< K > max ()  [inline], [static], [constexpr], [noexcept]
```

12.216.2.2 min()

```
template<size_t K>
static constexpr RecInt::rint< K > min ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.217 limits< RecInt::ruint< K > > Struct Template Reference

`#include <flimits.h>`

Public Types

- typedef RecInt::ruint< K > T

Static Public Member Functions

- static constexpr RecInt::ruint< K > max () noexcept
- static constexpr RecInt::ruint< K > min () noexcept

12.217.1 Member Typedef Documentation

12.217.1.1 T

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.217.2 Member Function Documentation

12.217.2.1 max()

```
template<size_t K>
static constexpr RecInt::ruint< K > max ()  [inline], [static], [constexpr], [noexcept]
```

12.217.2.2 min()

```
template<size_t K>
static constexpr RecInt::ruint< K > min ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.218 limits< short int > Struct Reference

`#include <flimits.h>`

Public Types

- typedef short int T

Static Public Member Functions

- static constexpr short int max () noexcept
- static constexpr short int min () noexcept
- static constexpr int32_t digits () noexcept

12.218.1 Member Typedef Documentation

12.218.1.1 T

```
typedef short int T
```

12.218.2 Member Function Documentation

12.218.2.1 max()

```
static constexpr short int max ()  [inline], [static], [constexpr], [noexcept]
```

12.218.2.2 min()

```
static constexpr short int min ()  [inline], [static], [constexpr], [noexcept]
```

12.218.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.219 limits< signed char > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef signed char T

Static Public Member Functions

- static constexpr signed char max () noexcept
- static constexpr signed char min () noexcept
- static constexpr int32_t digits () noexcept

12.219.1 Member Typedef Documentation

12.219.1.1 T

```
typedef signed char T
```

12.219.2 Member Function Documentation

12.219.2.1 max()

```
static constexpr signed char max ()  [inline], [static], [constexpr], [noexcept]
```

12.219.2.2 min()

```
static constexpr signed char min ()  [inline], [static], [constexpr], [noexcept]
```

12.219.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.220 limits< unsigned char > Struct Reference

`#include <flimits.h>`

Public Types

- typedef unsigned char T

Static Public Member Functions

- static constexpr unsigned char max () noexcept
- static constexpr unsigned char min () noexcept
- static constexpr int32_t digits () noexcept

12.220.1 Member Typedef Documentation

12.220.1.1 T

`typedef unsigned char T`

12.220.2 Member Function Documentation

12.220.2.1 max()

`static constexpr unsigned char max ()  [inline], [static], [constexpr], [noexcept]`

12.220.2.2 min()

`static constexpr unsigned char min ()  [inline], [static], [constexpr], [noexcept]`

12.220.2.3 digits()

`static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.221 limits< unsigned int > Struct Reference

`#include <flimits.h>`

Public Types

- typedef unsigned int T

Static Public Member Functions

- static constexpr unsigned int max () noexcept
- static constexpr unsigned int min () noexcept
- static constexpr int32_t digits () noexcept

12.221.1 Member Typedef Documentation

12.221.1.1 T

`typedef unsigned int T`

12.221.2 Member Function Documentation

12.221.2.1 max()

```
static constexpr unsigned int max ()  [inline], [static], [constexpr], [noexcept]
```

12.221.2.2 min()

```
static constexpr unsigned int min ()  [inline], [static], [constexpr], [noexcept]
```

12.221.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.222 limits< unsigned long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned long T

Static Public Member Functions

- static constexpr unsigned long max () noexcept
- static constexpr unsigned long min () noexcept
- static constexpr int32_t digits () noexcept

12.222.1 Member Typedef Documentation

12.222.1.1 T

```
typedef unsigned long T
```

12.222.2 Member Function Documentation

12.222.2.1 max()

```
static constexpr unsigned long max ()  [inline], [static], [constexpr], [noexcept]
```

12.222.2.2 min()

```
static constexpr unsigned long min ()  [inline], [static], [constexpr], [noexcept]
```

12.222.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.223 limits< unsigned long long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned long long T

Static Public Member Functions

- static constexpr unsigned long long max () noexcept
- static constexpr unsigned long long min () noexcept
- static constexpr int32_t digits () noexcept

12.223.1 Member Typedef Documentation

12.223.1.1 T

```
typedef unsigned long long T
```

12.223.2 Member Function Documentation

12.223.2.1 max()

```
static constexpr unsigned long long max ()  [inline], [static], [constexpr], [noexcept]
```

12.223.2.2 min()

```
static constexpr unsigned long long min ()  [inline], [static], [constexpr], [noexcept]
```

12.223.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.224 limits< unsigned short int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned short int T

Static Public Member Functions

- static constexpr unsigned short int max () noexcept
- static constexpr unsigned short int min () noexcept
- static constexpr int32_t digits () noexcept

12.224.1 Member Typedef Documentation

12.224.1.1 T

```
typedef unsigned short int T
```

12.224.2 Member Function Documentation

12.224.2.1 max()

```
static constexpr unsigned short int max ()  [inline], [static], [constexpr], [noexcept]
```

12.224.2.2 min()

```
static constexpr unsigned short int min ()  [inline], [static], [constexpr], [noexcept]
```

12.224.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.225 MachineFloatTag Struct Reference

float or double

`#include <field-traits.h>`

12.225.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- field-traits.h

12.226 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

`#include <field-traits.h>`

12.226.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- field-traits.h

12.227 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference

Public Types

- typedef MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Self_t
- typedef associatedDelayedField< constField >::type DelayedField_t
- typedef associatedDelayedField< constField >::field DelayedField
- typedef DelayedField::Element DFElt

Public Member Functions

- void initC ()
- void initA ()
- void initB ()
- void initOut ()
- size_t MaxDelayedDim (DFElt beta)
- bool Aunfit ()
- bool Bunfit ()
- void setOutBounds (const size_t k, const DFElt alpha, const DFElt beta)
- bool checkA (const Field &F, const FFLAS::FFLAS_TRANSPOSE ta, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)
- bool checkB (const Field &F, const FFLAS::FFLAS_TRANSPOSE tb, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb)
- bool checkOut (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

- bool checkOut (const Field &F, FFLAS_UPLO uplo, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)
- MMHelper ()
- MMHelper (const Field &F, size_t m, size_t k, size_t n, ParSeqTrait _PS)
- MMHelper (const Field &F, int w, ParSeqTrait _PS=ParSeqTrait())
- template<class F2, typename AlgoT2, typename FT2, typename PS2>
  MMHelper (MMHelper< F2, AlgoT2, FT2, PS2 > &WH)
- MMHelper (const Field &F, int w, DFElt _Amin, DFElt _Amax, DFElt _Bmin, DFElt _Bmax, DFElt _Cmin, DFElt _Cmax, ParSeqTrait _PS=ParSeqTrait())

Data Fields

- int recLevel
- DFElt FieldMin
- DFElt FieldMax
- DFElt Amin
- DFElt Amax
- DFElt Bmin
- DFElt Bmax
- DFElt Cmin
- DFElt Cmax
- DFElt Outmin
- DFElt Outmax
- DFElt MaxStorableValue
- const DelayedField_t delayedField
- ParSeqTrait parseq

Friends

- std::ostream & operator<< (std::ostream &out, const Self_t &M)

12.227.1 Member Typedef Documentation

12.227.1.1 Self_t

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

12.227.1.2 DelayedField_t

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef associatedDelayedField<constField>::type DelayedField_t
```

12.227.1.3 DelayedField

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef associatedDelayedField<constField>::field DelayedField
```

12.227.1.4 DFElt

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef DelayedField::Element DFElt
```

12.227.2 Constructor & Destructor Documentation

12.227.2.1 MMHelper() [1/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper () [inline]
```

12.227.2.2 MMHelper() [2/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait _PS)  [inline]
```

12.227.2.3 MMHelper() [3/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            int w,
            ParSeqTrait _PS = ParSeqTrait())  [inline]
```

12.227.2.4 MMHelper() [4/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
template<class F2, typename AlgoT2, typename FT2, typename PS2>
MMHelper (
            MMHelper< F2, AlgoT2, FT2, PS2 > & WH)  [inline]
```

12.227.2.5 MMHelper() [5/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            int w,
            DFElt _Amin,
            DFElt _Amax,
            DFElt _Bmin,
            DFElt _Bmax,
            DFElt _Cmin,
            DFElt _Cmax,
            ParSeqTrait _PS = ParSeqTrait())  [inline]
```

12.227.3 Member Function Documentation

12.227.3.1 initC()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initC ()  [inline]
```

12.227.3.2 initA()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initA ()  [inline]
```

12.227.3.3 initB()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initB ()  [inline]
```

12.227.3.4 initOut()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initOut ()  [inline]
```

12.227.3.5 MaxDelayedDim()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
size_t MaxDelayedDim (
            DFElt beta)  [inline]
```

12.227.3.6 Aunfit()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool Aunfit ()  [inline]
```

12.227.3.7 Bunfit()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool Bunfit ()  [inline]
```

12.227.3.8 setOutBounds()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void setOutBounds (
            const size_t k,
            const DFElt alpha,
            const DFElt beta)  [inline]
```

12.227.3.9 checkA()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkA (
            const Field & F,
            const FFLAS::FFLAS_TRANSPOSE ta,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda)  [inline]
```

12.227.3.10 checkB()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkB (
            const Field & F,
            const FFLAS::FFLAS_TRANSPOSE tb,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr B,
            const size_t ldb)  [inline]
```

12.227.3.11 checkOut() [1/2]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkOut (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda)  [inline]
```

12.227.3.12 checkOut() [2/2]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkOut (
```

```
            const Field & F,
            FFLAS_UPLO uplo,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda)  [inline]
```

12.227.4 Friends And Related Symbol Documentation

12.227.4.1 operator<<

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M)  [friend]
```

12.227.5 Field Documentation

12.227.5.1 recLevel

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
int recLevel
```

12.227.5.2 FieldMin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt FieldMin
```

12.227.5.3 FieldMax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt FieldMax
```

12.227.5.4 Amin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Amin
```

12.227.5.5 Amax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Amax
```

12.227.5.6 Bmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Bmin
```

12.227.5.7 Bmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Bmax
```

12.227.5.8 Cmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Cmin
```

12.227.5.9 Cmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Cmax
```

12.227.5.10 Outmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Outmin
```

12.227.5.11 Outmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Outmax
```

12.227.5.12 MaxStorableValue

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt MaxStorableValue
```

12.227.5.13 delayedField

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
const DelayedField_t delayedField
```

12.227.5.14 parseq

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.228 ModeTraits< Field > Struct Template Reference

ModeTraits.

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultTag value

12.228.1 Detailed Description

template<class Field>
struct FFLAS::ModeTraits< Field >

ModeTraits.

12.228.2 Member Typedef Documentation

12.228.2.1 value

```
template<class Field>
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.229 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DelayedTag value
- typedef ModeCategories::DefaultTag value

12.229.1 Member Typedef Documentation

12.229.1.1 value [1/2]

```
template<typename Element, typename Compute>
typedef ModeCategories::DelayedTag value
```

12.229.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.230 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.230.1 Member Typedef Documentation

12.230.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.230.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.231 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.231.1 Member Typedef Documentation

12.231.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.231.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.232 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.232.1 Member Typedef Documentation

12.232.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.232.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.233 ModeTraits< Givaro::Modular< int64_t, uint64_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultTag value
- typedef ModeCategories::DefaultTag value

12.233.1 Member Typedef Documentation

12.233.1.1 value [1/2]

```
typedef ModeCategories::DefaultTag value
```

12.233.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.234 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.234.1 Member Typedef Documentation

12.234.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.234.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.235 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.235.1 Member Typedef Documentation

12.235.1.1 value [1/2]

```
template<typename Compute, size_t K>
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.235.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.236 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.236.1 Member Typedef Documentation

12.236.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.236.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.237 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.237.1 Member Typedef Documentation

12.237.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.237.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.238 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.238.1 Member Typedef Documentation

12.238.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.238.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.239 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DelayedTag value
- typedef ModeCategories::DefaultTag value

12.239.1 Member Typedef Documentation

12.239.1.1 value [1/2]

```
template<typename Element>
typedef ModeCategories::DelayedTag value
```

12.239.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.240 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.240.1 Member Typedef Documentation

12.240.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.240.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.241 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.241.1 Member Typedef Documentation

12.241.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.241.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.242 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.242.1 Member Typedef Documentation

12.242.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.242.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.243 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.243.1 Member Typedef Documentation

12.243.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.243.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.244 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.244.1 Member Typedef Documentation

12.244.1.1 value [1/2]

```
template<class T>
typedef ModeCategories::DefaultBoundedTag value
```

12.244.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.245 ModeTraits< Givaro::ZRing< double > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.245.1 Member Typedef Documentation

12.245.1.1 value [1/2]

```
typedef ModeCategories::DefaultBoundedTag value
```

12.245.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.246 ModeTraits< Givaro::ZRing< float > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.246.1 Member Typedef Documentation

12.246.1.1 value [1/2]

```
typedef ModeCategories::DefaultBoundedTag value
```

12.246.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.247 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.247.1 Member Typedef Documentation

12.247.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.247.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.248 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.249 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.250 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
`#include <field-traits.h>`

12.250.1 Detailed Description

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
The documentation for this struct was generated from the following file:

- field-traits.h

12.251 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.252 need_field_characteristic< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = false

12.252.1 Field Documentation

12.252.1.1 value

```
template<typename Field>
bool value = false  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.253 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true
- static constexpr bool value

12.253.1 Field Documentation

12.253.1.1 value [1/2]

```
template<typename Field>
bool value = true  [static], [constexpr]
```

12.253.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.254 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true
- static constexpr bool value

12.254.1 Field Documentation

12.254.1.1 value [1/2]

```
template<typename Field>
bool value = true  [static], [constexpr]
```

12.254.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.255 NoSimd< T > Struct Template Reference

`#include <fflas_simd.h>`

Public Types

- using vect_t = T*
- using scalar_t = T
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, T>

Static Public Member Functions

- static const std::string type_string ()
- template<class TT>
  static constexpr bool valid (TT p)
- template<class TT>
  static constexpr bool compliant (TT n)

Static Public Attributes

- static const constexpr size_t vect_size = 1
- static const constexpr size_t alignment = static_cast<size_t>(Alignment::Normal)

12.255.1 Member Typedef Documentation

12.255.1.1 vect_t

```
template<typename T>
using vect_t = T*
```

12.255.1.2 scalar_t

```
template<typename T>
using scalar_t = T
```

12.255.1.3 aligned_allocator

```
template<typename T>
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.255.1.4 aligned_vector

```
template<typename T>
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.255.1.5 is_same_element

```
template<typename T>
template<class Field>
using is_same_element = std::is_same<typename Field::Element, T>
```

12.255.2 Member Function Documentation

12.255.2.1 type_string()

```
template<typename T>
static const std::string type_string ()  [inline], [static]
```

12.255.2.2 valid()

```
template<typename T>
template<class TT>
static constexpr bool valid (
            TT p)  [inline], [static], [constexpr]
```

12.255.2.3 compliant()

```
template<typename T>
template<class TT>
static constexpr bool compliant (
            TT n)  [inline], [static], [constexpr]
```

12.255.3 Field Documentation

12.255.3.1 vect_size

```
template<typename T>
const constexpr size_t vect_size = 1  [static], [constexpr]
```

12.255.3.2 alignment

```
template<typename T>
const constexpr size_t alignment = static_cast<size_t>(Alignment::Normal)  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.256 Parallel< C, P > Struct Template Reference

Public Types

- typedef C Cut
- typedef P Param

Public Member Functions

- Parallel (size_t n=NUM_THREADS)
- size_t numthreads () const
- size_t & set_numthreads (size_t n)

Friends

- std::ostream & operator<< (std::ostream &out, const Parallel &p)

12.256.1 Member Typedef Documentation

12.256.1.1 Cut

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef C Cut
```

12.256.1.2 Param

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef P Param
```

12.256.2 Constructor & Destructor Documentation

12.256.2.1 Parallel()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
Parallel (
            size_t n = NUM_THREADS)  [inline]
```

12.256.3 Member Function Documentation

12.256.3.1 numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t numthreads () const  [inline]
```

12.256.3.2 set_numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t & set_numthreads (
            size_t n)  [inline]
```

12.256.4 Friends And Related Symbol Documentation

12.256.4.1 operator<<

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
std::ostream & operator<< (
            std::ostream & out,
            const Parallel< C, P > & p)  [friend]
```

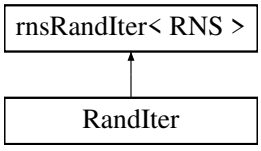
The documentation for this struct was generated from the following file:

- blockcuts.inl

12.257 RNSInteger< RNS >::RandIter Class Reference

`#include <rns-integer.h>`

Inheritance diagram for RNSInteger< RNS >::RandIter:



Public Member Functions

- RandIter (const RNSInteger< RNS > &F, uint64_t seed=0)
- RNS::Element & random (typename RNS::Element &elt) const

  RNS ring Element random assignement.
- RNS::Element random () const
- RNS::Element & operator() (typename RNS::Element &elt) const
- RNS::Element operator() () const
- const RNS & ring () const

12.257.1 Constructor & Destructor Documentation

12.257.1.1 RandIter()

```
template<typename RNS>
RandIter (
            const RNSInteger< RNS > & F,
            uint64_t seed = 0)  [inline]
```

12.257.2 Member Function Documentation

12.257.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
            typename RNS::Element & elt) const  [inline], [inherited]
```

RNS ring Element random assignement.
Element is supposed to be initialized

Returns

random ring Element

12.257.2.2 random() [2/2]

```
template<typename RNS>
RNS::Element random () const  [inline], [inherited]
```

12.257.2.3 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const  [inline], [inherited]
```

12.257.2.4 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const  [inline], [inherited]
```

12.257.2.5 ring()

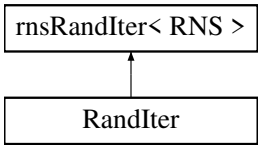
```
template<typename RNS>
const RNS & ring () const  [inline], [inherited]
```

The documentation for this class was generated from the following file:

- rns-integer.h

12.258 RNSIntegerMod< RNS >::RandIter Class Reference

`#include <rns-integer-mod.h>`
Inheritance diagram for RNSIntegerMod< RNS >::RandIter:



Public Member Functions

- RandIter (const RNSIntegerMod< RNS > &F, uint64_t seed=0)
- RNS::Element & random (typename RNS::Element &elt) const
- RNS::Element random () const
- RNS::Element & operator() (typename RNS::Element &elt) const
- RNS::Element operator() () const
- const RNS & ring () const

12.258.1 Constructor & Destructor Documentation

12.258.1.1 RandIter()

```
template<typename RNS>
RandIter (
            const RNSIntegerMod< RNS > & F,
            uint64_t seed = 0)  [inline]
```

12.258.2 Member Function Documentation

12.258.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
            typename RNS::Element & elt) const  [inline]
```

12.258.2.2 random() [2/2]

```
template<typename RNS>
RNS::Element random () const  [inline], [inherited]
```

12.258.2.3 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const  [inline], [inherited]
```

12.258.2.4 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const  [inline], [inherited]
```

12.258.2.5 ring()

```
template<typename RNS>
const RNS & ring () const  [inline], [inherited]
```

The documentation for this class was generated from the following file:

- rns-integer-mod.h

12.259 readMyMachineType< Field, T > Struct Template Reference

`#include <read_sparse.h>`

Public Types

- typedef Field::Element Element
- typedef Field::Element_ptr Element_ptr

Public Member Functions

- void operator() (const Field &F, Element &modulo, Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)

12.259.1 Member Typedef Documentation

12.259.1.1 Element

```
template<class Field, class T>
typedef Field::Element Element
```

12.259.1.2 Element_ptr

```
template<class Field, class T>
typedef Field::Element_ptr Element_ptr
```

12.259.2 Member Function Documentation

12.259.2.1 operator()()

```
template<class Field, typename T>
void operator() (
            const Field & F,
            Element & modulo,
            Element_ptr val,
            std::ifstream & file,
            const uint64_t dims,
            const mask_t data_type,
            const mask_t field_desc)
```

The documentation for this struct was generated from the following file:

- read_sparse.h

12.260 readMyMachineType< Field, mpz_t > Struct Template Reference

```
#include <read_sparse.h>
```

Public Types

- typedef Field::Element Element
- typedef Field::Element_ptr Element_ptr
- typedef Field::Element Element
- typedef Field::Element_ptr Element_ptr

Public Member Functions

- void operator() (const Field &F, Element &modulo, Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)
- void operator() (const Field &F, Element &modulo, Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)

12.260.1 Member Typedef Documentation

12.260.1.1 Element [1/2]

```
template<class Field>
typedef Field::Element Element
```

12.260.1.2 Element_ptr [1/2]

```
template<class Field>
typedef Field::Element_ptr Element_ptr
```

12.260.1.3 Element [2/2]

```
typedef Field::Element Element
```

12.260.1.4 Element_ptr [2/2]

```
typedef Field::Element_ptr Element_ptr
```

12.260.2 Member Function Documentation

12.260.2.1 operator()() [1/2]

```
template<class Field>
void operator() (
             const Field & F,
             typename Field::Element & modulo,
             typename Field::Element_ptr val,
             std::ifstream & file,
             const uint64_t dims,
             const mask_t data_type,
             const mask_t field_desc)
```

12.260.2.2 operator()() [2/2]

```
void operator() (
             const Field & F,
             Element & modulo,
             Element_ptr val,
             std::ifstream & file,
             const uint64_t dims,
             const mask_t data_type,
             const mask_t field_desc)
```

The documentation for this struct was generated from the following file:

- read_sparse.h

12.261 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.262 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.263 array< T >::reverse_iterator Class Reference

STL iterator class.

12.263.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.264 string::reverse_iterator Class Reference

STL iterator class.

12.264.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.265 vector< T >::reverse_iterator Class Reference

STL iterator class.

12.265.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.266 rint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.267 rns_double Struct Reference

`#include <rns-double.h>`

Public Types

- typedef Givaro::Integer integer
- typedef Givaro::Modular< double > ModField
- typedef double BasisElement
- typedef rns_double_elt Element
- typedef rns_double_elt_ptr Element_ptr
- typedef rns_double_elt_cstptr ConstElement_ptr

Public Member Functions

- rns_double (const integer &bound, size_t pbits, bool rnsmod=false, long seed=time(NULL))
- rns_double (size_t pbits, size_t size, long seed=time(NULL))
- template<typename Vect>
  rns_double (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- rns_double (const RNSIntegerMod< rns_double > &basis, bool rnsmod=false, long seed=time(NULL))
- void precompute_cst (size_t K=0)
- template<typename T>
  void init (size_t m, size_t n, double *Arns, size_t rda, const T *A, size_t lda, const integer &maxA, bool RNS_MAJOR=false) const
- void init (size_t m, size_t n, double *Arns, size_t rda, const integer *A, size_t lda, size_t k, bool RNS_↩
  MAJOR=false) const

- void init_transpose (size_t m, size_t n, double *Arns, size_t rda, const integer *A, size_t lda, size_t k, bool RNS_MAJOR=false) const
- void convert (size_t m, size_t n, integer gamma, integer *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false) const
- void convert_transpose (size_t m, size_t n, integer gamma, integer *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false) const
- void reduce (size_t n, double *Arns, size_t rda, bool RNS_MAJOR=false) const
- template<size_t K>
  void init (size_t m, size_t n, double *Arns, size_t rda, const RecInt::ruint< K > *A, size_t lda, size_t k, bool RNS_MAJOR=false) const
- template<size_t K>
  void convert (size_t m, size_t n, integer gamma, RecInt::ruint< K > *A, size_t lda, const double *Arns, size_t rda, integer p=0, bool RNS_MAJOR=false) const

Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis
- std::vector< ModField > _field_rns
- integer _M
- std::vector< integer > _Mi
- std::vector< double > _MMi
- std::vector< double > _crt_in
- std::vector< double > _crt_out
- size_t _size
- size_t _pbits
- size_t _ldm
- integer _mi_sum

12.267.1 Member Typedef Documentation

12.267.1.1 integer

```
typedef Givaro::Integer integer
```

12.267.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

12.267.1.3 BasisElement

```
typedef double BasisElement
```

12.267.1.4 Element

```
typedef rns_double_elt Element
```

12.267.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.267.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.267.2 Constructor & Destructor Documentation

12.267.2.1 rns_double() [1/4]

```
rns_double (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.267.2.2 rns_double() [2/4]

```
rns_double (
            size_t pbits,
            size_t size,
            long seed = time(NULL))  [inline]
```

12.267.2.3 rns_double() [3/4]

```
template<typename Vect>
rns_double (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.267.2.4 rns_double() [4/4]

```
rns_double (
            const RNSIntegerMod< rns_double > & basis,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.267.3 Member Function Documentation

12.267.3.1 precompute_cst()

```
void precompute_cst (
            size_t K = 0)  [inline]
```

12.267.3.2 init() [1/3]

```
template<typename T>
void init (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const T * A,
            size_t lda,
            const integer & maxA,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.3 init() [2/3]

```
void init (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const integer * A,
```

```
            size_t lda,
            size_t k,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.4 init_transpose()

```
void init_transpose (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const integer * A,
            size_t lda,
            size_t k,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.5 convert() [1/2]

```
void convert (
            size_t m,
            size_t n,
            integer gamma,
            integer * A,
            size_t lda,
            const double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.6 convert_transpose()

```
void convert_transpose (
            size_t m,
            size_t n,
            integer gamma,
            integer * A,
            size_t lda,
            const double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.8 init() [3/3]

```
template<size_t K>
void init (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const RecInt::ruint< K > * A,
            size_t lda,
```

```
            size_t k,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.9 convert() [2/2]

```
template<size_t K>
void convert (
            size_t m,
            size_t n,
            integer gamma,
            RecInt::ruint< K > * A,
            size_t lda,
            const double * Arns,
            size_t rda,
            integer p = 0,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.4 Field Documentation

12.267.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.267.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.267.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.267.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

12.267.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

12.267.4.6 _M

```
integer _M
```

12.267.4.7 _Mi

```
std::vector<integer> _Mi
```

12.267.4.8 _MMi

```
std::vector<double> _MMi
```

12.267.4.9 _crt_in

```
std::vector<double> _crt_in
```

12.267.4.10 _crt_out

```
std::vector<double> _crt_out
```

12.267.4.11 _size

```
size_t _size
```

12.267.4.12 _pbits

```
size_t _pbits
```

12.267.4.13 _ldm

```
size_t _ldm
```

12.267.4.14 _mi_sum

```
integer _mi_sum
```

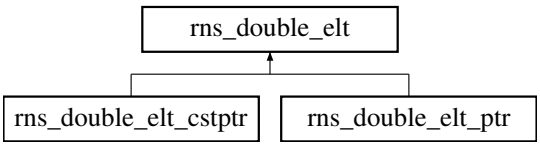
The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double-recint.inl
- rns-double.inl

12.268 rns_double_elt Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns_double_elt:



Public Member Functions

- rns_double_elt ()
- ~rns_double_elt ()
- rns_double_elt (double *p, size_t r, size_t a=false)
- rns_double_elt_ptr operator& ()
- rns_double_elt_cstptr operator& () const
- rns_double_elt (const rns_double_elt &x)

Data Fields

- double * _ptr
- size_t _stride
- bool _alloc

12.268.1 Constructor & Destructor Documentation

12.268.1.1 rns_double_elt() [1/3]

```
rns_double_elt () [inline]
```

12.268.1.2 ~rns_double_elt()

```
~rns_double_elt () [inline]
```

12.268.1.3 rns_double_elt() [2/3]

```
rns_double_elt (
            double * p,
            size_t r,
            size_t a = false)  [inline]
```

12.268.1.4 rns_double_elt() [3/3]

```
rns_double_elt (
            const rns_double_elt & x)  [inline]
```

12.268.2 Member Function Documentation

12.268.2.1 operator&() [1/2]

```
rns_double_elt_ptr operator& ()  [inline]
```

12.268.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const  [inline]
```

12.268.3 Field Documentation

12.268.3.1 _ptr

```
double* _ptr
```

12.268.3.2 _stride

```
size_t _stride
```

12.268.3.3 _alloc

```
bool _alloc
```

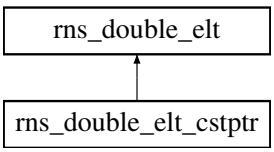
The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.269 rns_double_elt_cstptr Struct Reference

`#include <rns-double-elt.h>`

Inheritance diagram for rns_double_elt_cstptr:



Public Member Functions

- rns_double_elt_cstptr ()
- rns_double_elt_cstptr (double ∗p, size_t r)
- rns_double_elt_cstptr (const rns_double_elt_ptr &x)
- rns_double_elt_cstptr (const rns_double_elt_cstptr &x)
- rns_double_elt_cstptr (rns_double_elt_cstptr &&)=default
- rns_double_elt_cstptr ∗ operator& ()
- rns_double_elt & operator∗ () const

- rns_double_elt operator[ ] (size_t i) const
- rns_double_elt & operator[ ] (size_t i)
- rns_double_elt_cstptr operator++ ()
- rns_double_elt_cstptr operator-- ()
- rns_double_elt_cstptr operator+ (size_t inc) const
- rns_double_elt_cstptr operator- (size_t inc) const
- rns_double_elt_cstptr & operator+= (size_t inc)
- rns_double_elt_cstptr & operator-= (size_t inc)
- rns_double_elt_cstptr & operator= (const rns_double_elt_cstptr &x)
- bool operator< (const rns_double_elt_cstptr &x)
- bool operator!= (const rns_double_elt_cstptr &x)
- rns_double_elt_cstptr operator& () const

Data Fields

- rns_double_elt other
- double * _ptr
- size_t _stride
- bool _alloc

12.269.1 Constructor & Destructor Documentation

12.269.1.1 rns_double_elt_cstptr() [1/5]

```
rns_double_elt_cstptr ()  [inline]
```

12.269.1.2 rns_double_elt_cstptr() [2/5]

```
rns_double_elt_cstptr (
            double * p,
            size_t r)  [inline]
```

12.269.1.3 rns_double_elt_cstptr() [3/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_ptr & x)  [inline]
```

12.269.1.4 rns_double_elt_cstptr() [4/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_cstptr & x)  [inline]
```

12.269.1.5 rns_double_elt_cstptr() [5/5]

```
rns_double_elt_cstptr (
            rns_double_elt_cstptr && )  [default]
```

12.269.2 Member Function Documentation

12.269.2.1 operator&() [1/2]

```
rns_double_elt_cstptr * operator& ()  [inline]
```

12.269.2.2 operator*()

```
rns_double_elt & operator* () const  [inline]
```

12.269.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
            size_t i) const  [inline]
```

12.269.2.4 operator[]() [2/2]

```
rns_double_elt & operator[] (
            size_t i)  [inline]
```

12.269.2.5 operator++()

```
rns_double_elt_cstptr operator++ ()  [inline]
```

12.269.2.6 operator--()

```
rns_double_elt_cstptr operator-- ()  [inline]
```

12.269.2.7 operator+()

```
rns_double_elt_cstptr operator+ (
            size_t inc) const  [inline]
```

12.269.2.8 operator-()

```
rns_double_elt_cstptr operator- (
            size_t inc) const  [inline]
```

12.269.2.9 operator+=()

```
rns_double_elt_cstptr & operator+= (
            size_t inc)  [inline]
```

12.269.2.10 operator-=()

```
rns_double_elt_cstptr & operator-= (
            size_t inc)  [inline]
```

12.269.2.11 operator=()

```
rns_double_elt_cstptr & operator= (
            const rns_double_elt_cstptr & x)  [inline]
```

12.269.2.12 operator<()

```
bool operator< (
            const rns_double_elt_cstptr & x)  [inline]
```

12.269.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_cstptr & x)  [inline]
```

12.269.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const  [inline], [inherited]
```

12.269.3 Field Documentation

12.269.3.1 other

```
rns_double_elt other
```

12.269.3.2 _ptr

```
double* _ptr  [inherited]
```

12.269.3.3 _stride

```
size_t _stride  [inherited]
```

12.269.3.4 _alloc

```
bool _alloc  [inherited]
```

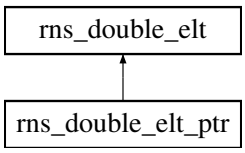
The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.270 rns_double_elt_ptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns_double_elt_ptr:



Public Member Functions

- rns_double_elt_ptr ()
- rns_double_elt_ptr (double *p, size_t r)
- rns_double_elt_ptr (const rns_double_elt_ptr &x)
- rns_double_elt_ptr (const rns_double_elt_cstptr &x)
- rns_double_elt_ptr (rns_double_elt_ptr &&)=default
- rns_double_elt_ptr * operator& ()
- rns_double_elt & operator* ()
- rns_double_elt operator[ ] (size_t i) const
- rns_double_elt & operator[ ] (size_t i)
- rns_double_elt_ptr operator++ ()
- rns_double_elt_ptr operator-- ()
- rns_double_elt_ptr operator+ (size_t inc)
- rns_double_elt_ptr operator- (size_t inc)
- rns_double_elt_ptr & operator+= (size_t inc)
- rns_double_elt_ptr & operator-= (size_t inc)
- rns_double_elt_ptr & operator= (const rns_double_elt_ptr &x)
- bool operator< (const rns_double_elt_ptr &x)
- bool operator!= (const rns_double_elt_ptr &x)
- rns_double_elt_cstptr operator& () const

Data Fields

- rns_double_elt other
- double * _ptr
- size_t _stride
- bool _alloc

12.270.1 Constructor & Destructor Documentation

12.270.1.1 rns_double_elt_ptr() [1/5]

```
rns_double_elt_ptr () [inline]
```

12.270.1.2 rns_double_elt_ptr() [2/5]

```
rns_double_elt_ptr (
            double * p,
            size_t r) [inline]
```

12.270.1.3 rns_double_elt_ptr() [3/5]

```
rns_double_elt_ptr (
            const rns_double_elt_ptr & x) [inline]
```

12.270.1.4 rns_double_elt_ptr() [4/5]

```
rns_double_elt_ptr (
            const rns_double_elt_cstptr & x) [inline]
```

12.270.1.5 rns_double_elt_ptr() [5/5]

```
rns_double_elt_ptr (
            rns_double_elt_ptr && ) [default]
```

12.270.2 Member Function Documentation

12.270.2.1 operator&() [1/2]

```
rns_double_elt_ptr * operator& () [inline]
```

12.270.2.2 operator*()

```
rns_double_elt & operator* () [inline]
```

12.270.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
            size_t i) const [inline]
```

12.270.2.4 operator[]() [2/2]

```
rns_double_elt & operator[] (
            size_t i) [inline]
```

12.270.2.5 operator++()

```
rns_double_elt_ptr operator++ () [inline]
```

12.270.2.6 operator--()

```
rns_double_elt_ptr operator-- () [inline]
```

12.270.2.7 operator+()

```
rns_double_elt_ptr operator+ (
            size_t inc)  [inline]
```

12.270.2.8 operator-()

```
rns_double_elt_ptr operator- (
            size_t inc)  [inline]
```

12.270.2.9 operator+=()

```
rns_double_elt_ptr & operator+= (
            size_t inc)  [inline]
```

12.270.2.10 operator-=()

```
rns_double_elt_ptr & operator-= (
            size_t inc)  [inline]
```

12.270.2.11 operator=()

```
rns_double_elt_ptr & operator= (
            const rns_double_elt_ptr & x)  [inline]
```

12.270.2.12 operator<()

```
bool operator< (
            const rns_double_elt_ptr & x)  [inline]
```

12.270.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_ptr & x)  [inline]
```

12.270.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const  [inline], [inherited]
```

12.270.3 Field Documentation

12.270.3.1 other

```
rns_double_elt other
```

12.270.3.2 _ptr

```
double* _ptr  [inherited]
```

12.270.3.3 _stride

```
size_t _stride  [inherited]
```

12.270.3.4 _alloc

```
bool _alloc  [inherited]
```

The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.271 rns_double_extended Struct Reference

`#include <rns-double.h>`

Public Types

- typedef Givaro::Integer integer
- typedef Givaro::ModularExtended< double > ModField
- typedef double BasisElement
- typedef rns_double_elt Element
- typedef rns_double_elt_ptr Element_ptr
- typedef rns_double_elt_cstptr ConstElement_ptr

Public Member Functions

- rns_double_extended (const integer &bound, size_t pbits, bool rnsmod=false, long seed=time(NULL))
- rns_double_extended (size_t pbits, size_t size, long seed=time(NULL))
- template<typename Vect>
  rns_double_extended (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- void precompute_cst ()
- void init (size_t m, size_t n, double *Arns, size_t rda, const integer *A, size_t lda, const integer &maxA, bool RNS_MAJOR=false) const
- void init (size_t m, size_t n, double *Arns, size_t rda, const integer *A, size_t lda, size_t k, bool RNS_↩ MAJOR=false)
- void convert (size_t m, size_t n, integer gamma, integer *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false)
- void init (size_t m, double *Arns, const integer *A, size_t lda) const
- void convert (size_t m, integer *A, const double *Arns) const
- void reduce (size_t n, double *Arns, size_t rda, bool RNS_MAJOR=false) const

Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis
- std::vector< ModField > _field_rns
- integer _M
- std::vector< integer > _Mi
- std::vector< double > _MMi
- std::vector< double > _crt_in
- std::vector< double > _crt_out
- size_t _size
- size_t _pbits
- size_t _ldm

12.271.1 Member Typedef Documentation

12.271.1.1 integer

`typedef Givaro::Integer integer`

12.271.1.2 ModField

`typedef Givaro::ModularExtended<double> ModField`

12.271.1.3 BasisElement

```
typedef double BasisElement
```

12.271.1.4 Element

```
typedef rns_double_elt Element
```

12.271.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.271.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.271.2 Constructor & Destructor Documentation

12.271.2.1 rns_double_extended() [1/3]

```
rns_double_extended (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.271.2.2 rns_double_extended() [2/3]

```
rns_double_extended (
            size_t pbits,
            size_t size,
            long seed = time(NULL))  [inline]
```

12.271.2.3 rns_double_extended() [3/3]

```
template<typename Vect>
rns_double_extended (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.271.3 Member Function Documentation

12.271.3.1 precompute_cst()

```
void precompute_cst ()  [inline]
```

12.271.3.2 init() [1/3]

```
void init (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const integer * A,
            size_t lda,
            const integer & maxA,
            bool RNS_MAJOR = false) const  [inline]
```

12.271.3.3 init() [2/3]

```
void init (
            size_t m,
            size_t n,
            double * Arns,
            size_t rda,
            const integer * A,
            size_t lda,
            size_t k,
            bool RNS_MAJOR = false)  [inline]
```

12.271.3.4 convert() [1/2]

```
void convert (
            size_t m,
            size_t n,
            integer gamma,
            integer * A,
            size_t lda,
            const double * Arns,
            size_t rda,
            bool RNS_MAJOR = false)  [inline]
```

12.271.3.5 init() [3/3]

```
void init (
            size_t m,
            double * Arns,
            const integer * A,
            size_t lda) const  [inline]
```

12.271.3.6 convert() [2/2]

```
void convert (
            size_t m,
            integer * A,
            const double * Arns) const  [inline]
```

12.271.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.271.4 Field Documentation

12.271.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.271.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.271.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.271.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

12.271.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

12.271.4.6 _M

```
integer _M
```

12.271.4.7 _Mi

```
std::vector<integer> _Mi
```

12.271.4.8 _MMi

```
std::vector<double> _MMi
```

12.271.4.9 _crt_in

```
std::vector<double> _crt_in
```

12.271.4.10 _crt_out

```
std::vector<double> _crt_out
```

12.271.4.11 _size

```
size_t _size
```

12.271.4.12 _pbits

```
size_t _pbits
```

12.271.4.13 _ldm

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double.inl

12.272 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```

12.272.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- field-traits.h

12.273 RNSInteger< RNS > Class Template Reference

```
#include <rns-integer.h>
```

Data Structures

- class RandIter

Public Types

- typedef RNS::Element Element
- typedef RNS::Element_ptr Element_ptr
- typedef RNS::ConstElement_ptr ConstElement_ptr

Public Member Functions

- RNSInteger (const RNS &myrns)
- template<typename T>
  RNSInteger (const T &F)
- const RNS & rns () const
- size_t size () const
- bool isOne (const Element &x) const
- bool isMOne (const Element &x) const
- bool isZero (const Element &x) const
- integer characteristic (integer &p) const
- integer cardinality (integer &p) const
- Element & init (Element &x) const
- Element & init (Element &x, const Givaro::Integer &y) const
- Element & reduce (Element &x, const Element &y) const
- Element & reduce (Element &x) const
- Givaro::Integer convert (Givaro::Integer &x, const Element &y) const
- Element & assign (Element &x, const Element &y) const
- std::ostream & write (std::ostream &os, const Element &y) const
- std::ostream & write (std::ostream &os) const

Data Fields

- Element one
- Element mOne
- Element zero

Protected Types

- typedef RNS::BasisElement BasisElement
- typedef Givaro::Integer integer

Protected Attributes

- const RNS * _rns

12.273.1 Member Typedef Documentation

12.273.1.1 BasisElement

```
template<typename RNS>
typedef RNS::BasisElement BasisElement  [protected]
```

12.273.1.2 integer

```
template<typename RNS>
typedef Givaro::Integer integer  [protected]
```

12.273.1.3 Element

```
template<typename RNS>
typedef RNS::Element Element
```

12.273.1.4 Element_ptr

```
template<typename RNS>
typedef RNS::Element_ptr Element_ptr
```

12.273.1.5 ConstElement_ptr

```
template<typename RNS>
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.273.2 Constructor & Destructor Documentation

12.273.2.1 RNSInteger() [1/2]

```
template<typename RNS>
RNSInteger (
            const RNS & myrns)  [inline]
```

12.273.2.2 RNSInteger() [2/2]

```
template<typename RNS>
template<typename T>
RNSInteger (
            const T & F)  [inline]
```

12.273.3 Member Function Documentation

12.273.3.1 rns()

```
template<typename RNS>
const RNS & rns () const  [inline]
```

12.273.3.2 size()

```
template<typename RNS>
size_t size () const  [inline]
```

12.273.3.3 isOne()

```
template<typename RNS>
bool isOne (
            const Element & x) const  [inline]
```

12.273.3.4 isMOne()

```
template<typename RNS>
bool isMOne (
            const Element & x) const  [inline]
```

12.273.3.5 isZero()

```
template<typename RNS>
bool isZero (
            const Element & x) const  [inline]
```

12.273.3.6 characteristic()

```
template<typename RNS>
integer characteristic (
            integer & p) const  [inline]
```

12.273.3.7 cardinality()

```
template<typename RNS>
integer cardinality (
            integer & p) const  [inline]
```

12.273.3.8 init() [1/2]

```
template<typename RNS>
Element & init (
            Element & x) const  [inline]
```

12.273.3.9 init() [2/2]

```
template<typename RNS>
Element & init (
            Element & x,
            const Givaro::Integer & y) const  [inline]
```

12.273.3.10 reduce() [1/2]

```
template<typename RNS>
Element & reduce (
            Element & x,
            const Element & y) const  [inline]
```

12.273.3.11 reduce() [2/2]

```
template<typename RNS>
Element & reduce (
            Element & x) const  [inline]
```

12.273.3.12 convert()

```
template<typename RNS>
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y) const  [inline]
```

12.273.3.13 assign()

```
template<typename RNS>
Element & assign (
            Element & x,
            const Element & y) const  [inline]
```

12.273.3.14 write() [1/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os,
            const Element & y) const  [inline]
```

12.273.3.15 write() [2/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os) const  [inline]
```

12.273.4 Field Documentation

12.273.4.1 _rns

```
template<typename RNS>
const RNS* _rns  [protected]
```

12.273.4.2 one

```
template<typename RNS>
Element one
```

12.273.4.3 mOne

```
template<typename RNS>
Element mOne
```

12.273.4.4 zero

```
template<typename RNS>
Element zero
```

The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer.h

12.274 RNSIntegerMod< RNS > Class Template Reference

`#include <rns-integer-mod.h>`

Data Structures

- class RandIter

Public Types

- typedef RNS::Element Element
- typedef RNS::Element_ptr Element_ptr
- typedef RNS::ConstElement_ptr ConstElement_ptr

Public Member Functions

- RNSIntegerMod (const integer &p, const RNS &myrns)
- const rns_double & rns () const
- const RNSInteger< RNS > & delayed () const
- size_t size () const
- bool isOne (const Element &x) const
- bool isMOne (const Element &x) const
- bool isZero (const Element &x) const
- integer & characteristic (integer &p) const
- integer characteristic () const
- integer & cardinality (integer &p) const
- integer cardinality () const

- integer minElement () const
- integer maxElement () const
- Element & init (Element &x) const
- Element & init (Element &x, const Givaro::Integer &y) const
- Element & reduce (Element &x, const Element &y) const
- Element & reduce (Element &x) const
- Element & init (Element &x, const Element &y) const
- Givaro::Integer convert (Givaro::Integer &x, const Element &y) const
- Element & assign (Element &x, const Element &y) const
- Element & add (Element &x, const Element &y, const Element &z) const
- Element & sub (Element &x, const Element &y, const Element &z) const
- Element & neg (Element &x, const Element &y) const
- Element & mul (Element &x, const Element &y, const Element &z) const
- Element & axpyin (Element &x, const Element &y, const Element &z) const
- Element & inv (Element &x, const Element &y) const
- bool areEqual (const Element &x, const Element &y) const
- std::ostream & write (std::ostream &os, const Element &y) const
- std::ostream & write (std::ostream &os) const
- void reduce_modp (size_t n, Element_ptr B) const
- std::ostream & write_matrix (std::ostream &c, const double *E, int n, int m, int lda) const
- std::ostream & write_matrix_long (std::ostream &c, const double *E, int n, int m, int lda) const
- void reduce_modp (size_t m, size_t n, Element_ptr B, size_t lda) const
- void reduce_modp_rnsmajor (size_t n, Element_ptr B) const

Data Fields

- Element one
- Element mOne
- Element zero

Protected Types

- typedef RNS::BasisElement BasisElement
- typedef Givaro::Modular< BasisElement > ModField
- typedef Givaro::Integer integer

Protected Attributes

- integer _p
- std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _Mi_modp_rns
- std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _iM_modp_rns
- const RNS * _rns
- Givaro::Modular< Givaro::Integer > _F
- RNSInteger< RNS > _RNSdelayed

12.274.1 Member Typedef Documentation

12.274.1.1 Element

```
template<typename RNS>
typedef RNS::Element Element
```

12.274.1.2 Element_ptr

```
template<typename RNS>
typedef RNS::Element_ptr Element_ptr
```

12.274.1.3 ConstElement_ptr

```
template<typename RNS>
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.274.1.4 BasisElement

```
template<typename RNS>
typedef RNS::BasisElement BasisElement  [protected]
```

12.274.1.5 ModField

```
template<typename RNS>
typedef Givaro::Modular<BasisElement> ModField  [protected]
```

12.274.1.6 integer

```
template<typename RNS>
typedef Givaro::Integer integer  [protected]
```

12.274.2 Constructor & Destructor Documentation

12.274.2.1 RNSIntegerMod()

```
template<typename RNS>
RNSIntegerMod (
            const integer & p,
            const RNS & myrns)  [inline]
```

12.274.3 Member Function Documentation

12.274.3.1 rns()

```
template<typename RNS>
const rns_double & rns () const  [inline]
```

12.274.3.2 delayed()

```
template<typename RNS>
const RNSInteger< RNS > & delayed () const  [inline]
```

12.274.3.3 size()

```
template<typename RNS>
size_t size () const  [inline]
```

12.274.3.4 isOne()

```
template<typename RNS>
bool isOne (
            const Element & x) const  [inline]
```

12.274.3.5 isMOne()

```
template<typename RNS>
bool isMOne (
            const Element & x) const  [inline]
```

12.274.3.6 isZero()

```
template<typename RNS>
bool isZero (
            const Element & x) const  [inline]
```

12.274.3.7 characteristic() [1/2]

```
template<typename RNS>
integer & characteristic (
            integer & p) const  [inline]
```

12.274.3.8 characteristic() [2/2]

```
template<typename RNS>
integer characteristic () const  [inline]
```

12.274.3.9 cardinality() [1/2]

```
template<typename RNS>
integer & cardinality (
            integer & p) const  [inline]
```

12.274.3.10 cardinality() [2/2]

```
template<typename RNS>
integer cardinality () const  [inline]
```

12.274.3.11 minElement()

```
template<typename RNS>
integer minElement () const  [inline]
```

12.274.3.12 maxElement()

```
template<typename RNS>
integer maxElement () const  [inline]
```

12.274.3.13 init() [1/3]

```
template<typename RNS>
Element & init (
            Element & x) const  [inline]
```

12.274.3.14 init() [2/3]

```
template<typename RNS>
Element & init (
            Element & x,
            const Givaro::Integer & y) const  [inline]
```

12.274.3.15 reduce() [1/2]

```
template<typename RNS>
Element & reduce (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.16 reduce() [2/2]

```
template<typename RNS>
Element & reduce (
            Element & x) const  [inline]
```

12.274.3.17 init() [3/3]

```
template<typename RNS>
Element & init (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.18 convert()

```
template<typename RNS>
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y) const  [inline]
```

12.274.3.19 assign()

```
template<typename RNS>
Element & assign (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.20 add()

```
template<typename RNS>
Element & add (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.21 sub()

```
template<typename RNS>
Element & sub (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.22 neg()

```
template<typename RNS>
Element & neg (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.23 mul()

```
template<typename RNS>
Element & mul (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.24 axpyin()

```
template<typename RNS>
Element & axpyin (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.25 inv()

```
template<typename RNS>
Element & inv (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.26 areEqual()

```
template<typename RNS>
bool areEqual (
            const Element & x,
            const Element & y) const  [inline]
```

12.274.3.27 write() [1/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os,
            const Element & y) const  [inline]
```

12.274.3.28 write() [2/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os) const  [inline]
```

12.274.3.29 reduce_modp() [1/2]

```
template<typename RNS>
void reduce_modp (
            size_t n,
            Element_ptr B) const  [inline]
```

12.274.3.30 write_matrix()

```
template<typename RNS>
std::ostream & write_matrix (
            std::ostream & c,
            const double * E,
            int n,
            int m,
            int lda) const  [inline]
```

12.274.3.31 write_matrix_long()

```
template<typename RNS>
std::ostream & write_matrix_long (
            std::ostream & c,
            const double * E,
            int n,
```

```
            int m,
            int lda) const  [inline]
```

12.274.3.32 reduce_modp() [2/2]

```
template<typename RNS>
void reduce_modp (
            size_t m,
            size_t n,
            Element_ptr B,
            size_t lda) const  [inline]
```

12.274.3.33 reduce_modp_rnsmajor()

```
template<typename RNS>
void reduce_modp_rnsmajor (
            size_t n,
            Element_ptr B) const  [inline]
```

12.274.4 Field Documentation

12.274.4.1 _p

```
template<typename RNS>
integer _p  [protected]
```

12.274.4.2 _Mi_modp_rns

```
template<typename RNS>
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↩
rns  [protected]
```

12.274.4.3 _iM_modp_rns

```
template<typename RNS>
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↩
rns  [protected]
```

12.274.4.4 _rns

```
template<typename RNS>
const RNS* _rns  [protected]
```

12.274.4.5 _F

```
template<typename RNS>
Givaro::Modular<Givaro::Integer> _F  [protected]
```

12.274.4.6 _RNSdelayed

```
template<typename RNS>
RNSInteger<RNS> _RNSdelayed  [protected]
```

12.274.4.7 one

```
template<typename RNS>
Element one
```

12.274.4.8 mOne

```
template<typename RNS>
Element mOne
```

12.274.4.9 zero

```
template<typename RNS>
Element zero
```

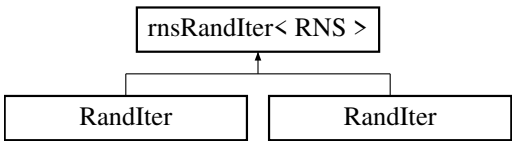
The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer-mod.h

12.275 rnsRandIter< RNS > Class Template Reference

`#include <rns-double.h>`

Inheritance diagram for rnsRandIter< RNS >:



Public Member Functions

- rnsRandIter (const RNS &R, uint64_t seed=0)
- RNS::Element & random (typename RNS::Element &elt) const

  RNS ring Element random assignement.
- RNS::Element & operator() (typename RNS::Element &elt) const
- RNS::Element operator() () const
- RNS::Element random () const
- const RNS & ring () const

12.275.1 Constructor & Destructor Documentation

12.275.1.1 rnsRandIter()

```
template<typename RNS>
rnsRandIter (
            const RNS & R,
            uint64_t seed = 0)  [inline]
```

12.275.2 Member Function Documentation

12.275.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
            typename RNS::Element & elt) const  [inline]
```

RNS ring Element random assignement.
Element is supposed to be initialized

Returns

random ring Element

12.275.2.2 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const  [inline]
```

12.275.2.3 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const  [inline]
```

12.275.2.4 random() [2/2]

```
template<typename RNS>
RNS::Element random () const  [inline]
```

12.275.2.5 ring()

```
template<typename RNS>
const RNS & ring () const  [inline]
```

The documentation for this class was generated from the following file:

- rns-double.h

12.276 Row Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

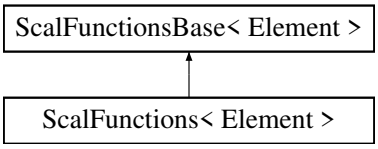
12.277 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.278 ScalFunctions< Element > Struct Template Reference

Inheritance diagram for ScalFunctions< Element >:



Public Types

- using vectElt = vector<Element>

Static Public Member Functions

- static void genInputs (vector< vectElt > &inputs)
- static void genInputsWithZero (vector< vectElt > &inputs)
- static Element zero ()
- static Element vand (Element x1, Element x2)
- static Element vor (Element x1, Element x2)

- static Element vxor (Element x1, Element x2)
- static Element vandnot (Element x1, Element x2)
- static Element add (Element x1, Element x2)
- static Element addin (Element &x1, Element x2)
- static Element sub (Element x1, Element x2)
- static Element subin (Element &x1, Element x2)
- static Element mul (Element x1, Element x2)
- static Element mulin (Element &x1, Element x2)
- static Element div (Element x1, Element x2)
- static Element fmadd (Element x1, Element x2, Element x3)
- static Element fmaddin (Element &x1, Element x2, Element x3)
- static Element fmsub (Element x1, Element x2, Element x3)
- static Element fmsubin (Element &x1, Element x2, Element x3)
- static Element fnmadd (Element x1, Element x2, Element x3)
- static Element fnmaddin (Element &x1, Element x2, Element x3)
- static Element lesser (Element x1, Element x2)
- static Element lesser_eq (Element x1, Element x2)
- static Element greater (Element x1, Element x2)
- static Element greater_eq (Element x1, Element x2)
- static Element eq (Element x1, Element x2)
- static vectElt unpacklo (vectElt a, vectElt b)
- static vectElt unpackhi (vectElt a, vectElt b)
- static void unpacklohi (vectElt &lo, vectElt &hi, vectElt a, vectElt b)
- static vectElt pack_even (vectElt a, vectElt b)
- static vectElt pack_odd (vectElt a, vectElt b)
- static void pack (vectElt &even, vectElt &odd, vectElt a, vectElt b)
- template<uint16_t s>
  static vectElt blend (vectElt a, vectElt b)

12.278.1 Member Typedef Documentation

12.278.1.1 vectElt

```
template<typename Element>
using vectElt = vector<Element>
```

12.278.2 Member Function Documentation

12.278.2.1 genInputs()

```
template<typename Element>
static void genInputs (
             vector< vectElt > & inputs)  [inline], [static]
```

12.278.2.2 genInputsWithZero()

```
template<typename Element>
static void genInputsWithZero (
             vector< vectElt > & inputs)  [inline], [static]
```

12.278.2.3 zero()

```
template<typename Element>
static Element zero ()  [inline], [static]
```

12.278.2.4 vand()

```
template<typename Element>
static Element vand (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.5 vor()

```
template<typename Element>
static Element vor (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.6 vxor()

```
template<typename Element>
static Element vxor (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.7 vandnot()

```
template<typename Element>
static Element vandnot (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.8 add()

```
template<typename Element>
static Element add (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.9 addin()

```
template<typename Element>
static Element addin (
            Element & x1,
            Element x2)  [inline], [static]
```

12.278.2.10 sub()

```
template<typename Element>
static Element sub (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.11 subin()

```
template<typename Element>
static Element subin (
            Element & x1,
            Element x2)  [inline], [static]
```

12.278.2.12 mul()

```
template<typename Element>
static Element mul (
```

```
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.13 mulin()

```
template<typename Element>
static Element mulin (
            Element & x1,
            Element x2)  [inline], [static]
```

12.278.2.14 div()

```
template<typename Element>
static Element div (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.15 fmadd()

```
template<typename Element>
static Element fmadd (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.16 fmaddin()

```
template<typename Element>
static Element fmaddin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.17 fmsub()

```
template<typename Element>
static Element fmsub (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.18 fmsubin()

```
template<typename Element>
static Element fmsubin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.19 fnmadd()

```
template<typename Element>
static Element fnmadd (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.20 fnmaddin()

```
template<typename Element>
static Element fnmaddin (
             Element & x1,
             Element x2,
             Element x3)  [inline], [static]
```

12.278.2.21 lesser()

```
template<typename Element>
static Element lesser (
             Element x1,
             Element x2)  [inline], [static]
```

12.278.2.22 lesser_eq()

```
template<typename Element>
static Element lesser_eq (
             Element x1,
             Element x2)  [inline], [static]
```

12.278.2.23 greater()

```
template<typename Element>
static Element greater (
             Element x1,
             Element x2)  [inline], [static]
```

12.278.2.24 greater_eq()

```
template<typename Element>
static Element greater_eq (
             Element x1,
             Element x2)  [inline], [static]
```

12.278.2.25 eq()

```
template<typename Element>
static Element eq (
             Element x1,
             Element x2)  [inline], [static]
```

12.278.2.26 unpacklo()

```
template<typename Element>
static vectElt unpacklo (
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.27 unpackhi()

```
template<typename Element>
static vectElt unpackhi (
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.28 unpacklohi()

```
template<typename Element>
static void unpacklohi (
             vectElt & lo,
             vectElt & hi,
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.29 pack_even()

```
template<typename Element>
static vectElt pack_even (
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.30 pack_odd()

```
template<typename Element>
static vectElt pack_odd (
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.31 pack()

```
template<typename Element>
static void pack (
             vectElt & even,
             vectElt & odd,
             vectElt a,
             vectElt b)  [inline], [static]
```

12.278.2.32 blend()

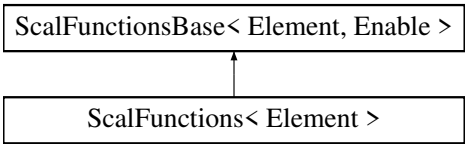
```
template<typename Element>
template<uint16_t s>
static vectElt blend (
             vectElt a,
             vectElt b)  [inline], [static]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.279 ScalFunctionsBase< Element, Enable > Struct Template Reference

Inheritance diagram for ScalFunctionsBase< Element, Enable >:



The documentation for this struct was generated from the following file:

- test-simd.C

12.280 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference

Data Structures

- class FloatingPointTestDistribution

Static Public Member Functions

- static FloatingPointTestDistribution get_default_random_generator ()
- static Element ceil (Element x)
- static Element floor (Element x)
- static Element round (Element x)
- static Element blendv (Element a, Element b, Element mask)
- static Element fma (Element x, Element y, Element z)

Static Public Attributes

- static constexpr Element _zero = 0.0
- static constexpr Element cmp_true = NAN
- static constexpr Element cmp_false = _zero

12.280.1 Member Function Documentation

12.280.1.1 get_default_random_generator()

```
template<typename Element>
static FloatingPointTestDistribution get_default_random_generator ()  [inline], [static]
```

12.280.1.2 ceil()

```
template<typename Element>
static Element ceil (
            Element x)  [inline], [static]
```

12.280.1.3 floor()

```
template<typename Element>
static Element floor (
            Element x)  [inline], [static]
```

12.280.1.4 round()

```
template<typename Element>
static Element round (
            Element x)  [inline], [static]
```

12.280.1.5 blendv()

```
template<typename Element>
static Element blendv (
            Element a,
            Element b,
            Element mask)  [inline], [static]
```

12.280.1.6 fma()

```
template<typename Element>
static Element fma (
             Element x,
             Element y,
             Element z)  [inline], [static]
```

12.280.2 Field Documentation

12.280.2.1 _zero

```
template<typename Element>
Element _zero = 0.0  [static], [constexpr]
```

12.280.2.2 cmp_true

```
template<typename Element>
Element cmp_true = NAN  [static], [constexpr]
```

12.280.2.3 cmp_false

```
template<typename Element>
Element cmp_false = _zero  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.281 ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference

Static Public Member Functions

- static uniform_int_distribution< Element > get_default_random_generator ()
- static Element round (Element x)
- static Element fma (Element x, Element y, Element z)
- static Element mullo (Element x1, Element x2)
- static Element mulhi (Element x1, Element x2)
- static Element mulx (Element x1, Element x2)
- static Element fmaddx (Element x1, Element x2, Element x3)
- static Element fmaddxin (Element &x1, Element x2, Element x3)
- static Element fmsubx (Element x1, Element x2, Element x3)
- static Element fmsubxin (Element &x1, Element x2, Element x3)
- static Element fnmaddx (Element x1, Element x2, Element x3)
- static Element fnmaddxin (Element &x1, Element x2, Element x3)
- template<int s>
  static Element sra (Element x1)
- template<int s>
  static Element srl (Element x1)
- template<int s>
  static Element sll (Element x1)

Static Public Attributes

- static constexpr Element _zero = 0
- static constexpr Element cmp_true = -1
- static constexpr Element cmp_false = _zero

12.281.1 Member Function Documentation

12.281.1.1 get_default_random_generator()

```
template<typename Element>
static uniform_int_distribution< Element > get_default_random_generator ()  [inline], [static]
```

12.281.1.2 round()

```
template<typename Element>
static Element round (
            Element x)  [inline], [static]
```

12.281.1.3 fma()

```
template<typename Element>
static Element fma (
            Element x,
            Element y,
            Element z)  [inline], [static]
```

12.281.1.4 mullo()

```
template<typename Element>
static Element mullo (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.5 mulhi()

```
template<typename Element>
static Element mulhi (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.6 mulx()

```
template<typename Element>
static Element mulx (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.7 fmaddx()

```
template<typename Element>
static Element fmaddx (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.281.1.8 fmaddxin()

```
template<typename Element>
static Element fmaddxin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.281.1.9 fmsubx()

```
template<typename Element>
static Element fmsubx (
             Element x1,
             Element x2,
             Element x3)  [inline], [static]
```

12.281.1.10 fmsubxin()

```
template<typename Element>
static Element fmsubxin (
             Element & x1,
             Element x2,
             Element x3)  [inline], [static]
```

12.281.1.11 fnmaddx()

```
template<typename Element>
static Element fnmaddx (
             Element x1,
             Element x2,
             Element x3)  [inline], [static]
```

12.281.1.12 fnmaddxin()

```
template<typename Element>
static Element fnmaddxin (
             Element & x1,
             Element x2,
             Element x3)  [inline], [static]
```

12.281.1.13 sra()

```
template<typename Element>
template<int s>
static Element sra (
             Element x1)  [inline], [static]
```

12.281.1.14 srl()

```
template<typename Element>
template<int s>
static Element srl (
             Element x1)  [inline], [static]
```

12.281.1.15 sll()

```
template<typename Element>
template<int s>
static Element sll (
             Element x1)  [inline], [static]
```

12.281.2 Field Documentation

12.281.2.1 _zero

```
template<typename Element>
Element _zero = 0  [static], [constexpr]
```

12.281.2.2 cmp_true

```
template<typename Element>
Element cmp_true = -1  [static], [constexpr]
```

12.281.2.3 cmp_false

```
template<typename Element>
Element cmp_false = _zero  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.282 Sequential Struct Reference

Public Member Functions

- Sequential ()
- Sequential (size_t nth)
- template<class Cut, class Param>
  Sequential (Parallel< Cut, Param > &)
- size_t numthreads () const

Friends

- std::ostream & operator<< (std::ostream &out, const Sequential &)

12.282.1 Constructor & Destructor Documentation

12.282.1.1 Sequential() [1/3]

```
Sequential ()  [inline]
```

12.282.1.2 Sequential() [2/3]

```
Sequential (
            size_t nth)  [inline]
```

12.282.1.3 Sequential() [3/3]

```
template<class Cut, class Param>
Sequential (
            Parallel< Cut, Param > & )  [inline]
```

12.282.2 Member Function Documentation

12.282.2.1 numthreads()

```
size_t numthreads () const  [inline]
```

12.282.3 Friends And Related Symbol Documentation

12.282.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Sequential & )  [friend]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.283 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd128.inl

12.284 Simd128_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- simd128_float.inl

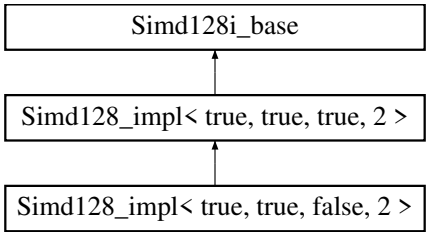
12.285 Simd128_impl< true, false, true, 8 > Struct Reference

The documentation for this struct was generated from the following file:

- simd128_double.inl

12.286 Simd128_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 2 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint16_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)

- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint32_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)

- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 16

12.286.1 Member Typedef Documentation

12.286.1.1 scalar_t

```
using scalar_t = uint16_t
```

12.286.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.286.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.286.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.286.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.286.2 Member Function Documentation

12.286.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.286.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.286.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.286.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.286.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.286.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.286.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.286.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.286.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.286.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.286.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.286.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.286.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.286.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.286.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.286.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.286.2.27 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static], [inherited]
```

12.286.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.286.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.286.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.286.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.286.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.286.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.286.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.36 shuffle()

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.286.2.46 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.47 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.48 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.49 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.50 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.51 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.52 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.53 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.54 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.55 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.56 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.57 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.58 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.59 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.60 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.61 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const __m64 & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static], [inherited]
```

12.286.2.62 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.286.2.63 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.64 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.65 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.66 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.67 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.68 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.3 Field Documentation

12.286.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

12.286.3.2 alignment

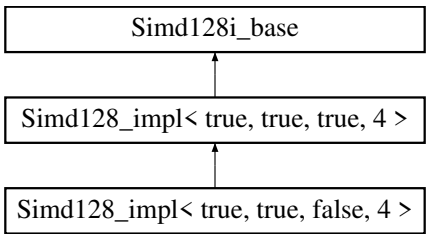
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

12.287 Simd128_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 4 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint32_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)

- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 4
- static const constexpr size_t alignment = 16

12.287.1 Member Typedef Documentation

12.287.1.1 scalar_t

```
using scalar_t = uint32_t
```

12.287.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.287.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.287.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.287.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.287.2 Member Function Documentation

12.287.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.287.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.287.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static]
```

12.287.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.287.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.287.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.287.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.287.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.287.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.287.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.287.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.287.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.287.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.287.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.287.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.287.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.287.2.27 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static], [inherited]
```

12.287.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.287.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.287.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.287.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.287.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.287.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.287.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.36 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static], [inherited]
```

12.287.2.46 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.47 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.48 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.49 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.50 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.51 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.52 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.53 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.54 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.55 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.56 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.57 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.58 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.59 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.60 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.61 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static], [inherited]
```

12.287.2.62 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.287.2.63 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.64 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.65 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.66 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.67 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.68 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.3 Field Documentation

12.287.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr], [inherited]
```

12.287.3.2 alignment

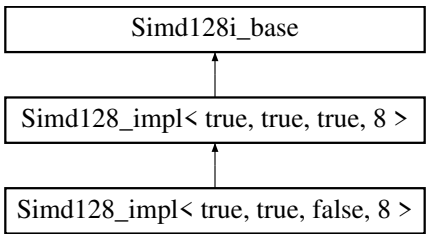
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

12.288 Simd128_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 8 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint64_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- template<int idx>
  static INLINE CONST scalar_t get (vect_t v)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)

- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 2
- static const constexpr size_t alignment = 16

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.288.1 Member Typedef Documentation

12.288.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.288.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.288.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.288.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.288.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.288.2 Member Function Documentation

12.288.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.288.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.288.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1)  [inline], [static]
```

12.288.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.288.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.288.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.288.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.288.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.288.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.288.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.288.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.288.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.288.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.15 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1)  [inline], [static]
```

12.288.2.16 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.17 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.18 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.19 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.20 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.21 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.22 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.23 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.24 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.288.2.25 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.288.2.26 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.288.2.27 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.288.2.28 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1)  [inline], [static], [inherited]
```

12.288.2.29 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.288.2.30 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v)  [inline], [static], [inherited]
```

12.288.2.31 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.288.2.32 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.288.2.33 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.288.2.34 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.288.2.35 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.288.2.36 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.37 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.38 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.39 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.40 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.43 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.44 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.45 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.46 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.47 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1)  [inline], [static], [inherited]
```

12.288.2.48 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.49 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.50 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.51 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.52 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.62 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static], [inherited]
```

12.288.2.63 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static], [inherited]
```

12.288.2.64 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m128d & P,
            const __m128d & INVP,
            const __m128d & NEGP,
            const vect_t & POW50REM,
            const __m128d & MIN,
            const __m128d & MAX,
            __m128d & Q,
            __m128d & T)  [static], [inherited]
```

12.288.2.65 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.288.2.66 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.288.2.67 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.68 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.69 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.70 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.71 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.72 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.3 Field Documentation

12.288.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr], [inherited]
```

12.288.3.2 alignment

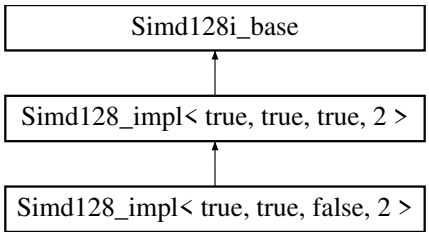
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

12.289 Simd128_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 2 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m128i
- using scalar_t = int16_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint32_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)

- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 16

12.289.1 Member Typedef Documentation

12.289.1.1 vect_t

```
using vect_t = __m128i
```

12.289.1.2 scalar_t

```
using scalar_t = int16_t
```

12.289.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.289.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.289.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.289.2 Member Function Documentation

12.289.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.289.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.289.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.289.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.289.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.289.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.289.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.289.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.289.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.289.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.289.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.289.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.289.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.289.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.289.2.15 shuffle()

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.289.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.289.2.25 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.26 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.27 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.289.2.28 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.29 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.289.2.30 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.31 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.32 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.33 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.34 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.35 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.36 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.37 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.38 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.39 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.40 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.41 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.42 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.43 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.44 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.45 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.46 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.47 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.48 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.49 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.50 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.51 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.289.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.289.2.53 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const __m64 & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
```

```
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.289.2.54 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.289.2.55 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.289.2.56 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.289.2.57 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.58 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.59 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.60 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.3 Field Documentation

12.289.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.289.3.2 alignment

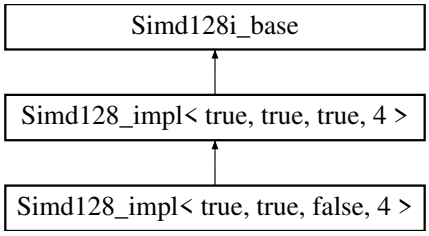
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

12.290 Simd128_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 4 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m128i
- using scalar_t = int32_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)

- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 4
- static const constexpr size_t alignment = 16

12.290.1 Member Typedef Documentation

12.290.1.1 vect_t

```
using vect_t = __m128i
```

12.290.1.2 scalar_t

```
using scalar_t = int32_t
```

12.290.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.290.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.290.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.290.2 Member Function Documentation

12.290.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.290.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.290.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.290.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.290.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static]
```

12.290.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.290.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.290.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.290.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.290.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.290.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.290.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.290.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.290.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.290.2.15 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.290.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static]
```

12.290.2.25 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.26 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.27 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.290.2.28 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.29 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.290.2.30 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.31 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.32 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.33 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.34 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.35 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.36 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.37 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.38 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.39 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.40 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.41 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.42 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.43 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.44 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.45 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.46 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.47 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.48 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.49 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.50 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.51 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.290.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.290.2.53 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.290.2.54 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.290.2.55 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.290.2.56 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.290.2.57 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.290.2.58 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.290.2.59 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.290.2.60 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.290.3 Field Documentation

12.290.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

12.290.3.2 alignment

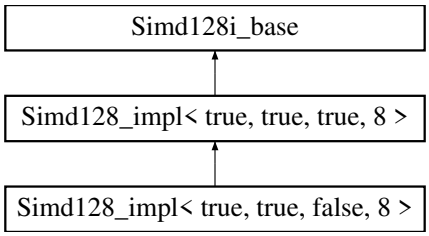
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

12.291 Simd128_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 8 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m128i
- using scalar_t = int64_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- template<int idx>
  static INLINE CONST scalar_t get (vect_t v)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1)

- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)
- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 2
- static const constexpr size_t alignment = 16

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.291.1 Member Typedef Documentation

12.291.1.1 vect_t

```
using vect_t = __m128i
```

12.291.1.2 scalar_t

```
using scalar_t = int64_t
```

12.291.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.291.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.291.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.291.2 Member Function Documentation

12.291.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.291.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.291.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.291.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.291.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1)  [inline], [static]
```

12.291.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.291.2.7 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v)  [inline], [static]
```

12.291.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.291.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.291.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.291.2.11 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.291.2.12 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.291.2.13 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.291.2.14 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.291.2.15 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.291.2.16 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.291.2.17 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.18 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.21 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.22 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.23 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.24 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.25 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1)  [inline], [static]
```

12.291.2.26 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.291.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.291.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1)  [inline], [static]
```

12.291.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.291.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.291.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.291.2.55 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.291.2.56 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m128d & P,
            const __m128d & INVP,
            const __m128d & NEGP,
            const vect_t & POW50REM,
            const __m128d & MIN,
            const __m128d & MAX,
            __m128d & Q,
            __m128d & T)  [static]
```

12.291.2.57 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.291.2.58 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.291.2.59 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.291.2.60 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.291.2.61 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.62 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.63 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.64 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.3 Field Documentation

12.291.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr]
```

12.291.3.2 alignment

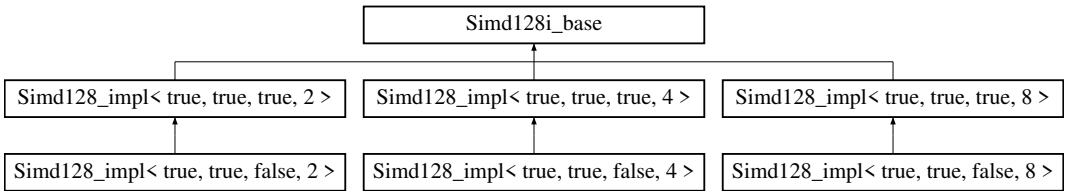
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

12.292 Simd128i_base Struct Reference

Inheritance diagram for Simd128i_base:



Public Types

- using vect_t = __m128i

Static Public Member Functions

- static INLINE CONST vect_t zero ()
- template<uint8_t s>
  static INLINE CONST vect_t sll128 (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t srl128 (const vect_t a)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

12.292.1 Member Typedef Documentation

12.292.1.1 vect_t

```
using vect_t = __m128i
```

12.292.2 Member Function Documentation

12.292.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.292.2.2 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static]
```

12.292.2.3 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static]
```

12.292.2.4 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.5 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.6 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.7 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd128.inl

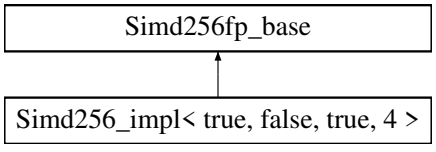
12.293 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd256.inl

12.294 Simd256_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 4 >:

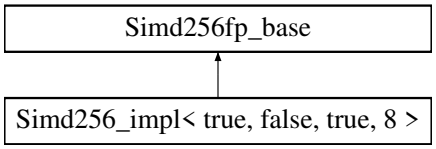


The documentation for this struct was generated from the following file:

- simd256_float.inl

12.295 Simd256_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 8 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m256d
- using scalar_t = double
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (const scalar_t *p, const vect_t v)
- static INLINE void storeu (const scalar_t *p, const vect_t v)
- static INLINE void stream (const scalar_t *p, const vect_t v)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t blendv (const vect_t a, const vect_t b, const vect_t mask)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE CONST vect_t subin (vect_t &a, const vect_t b)

- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t div (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)
- static INLINE CONST vect_t floor (const vect_t a)
- static INLINE CONST vect_t ceil (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t hadd (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)

Static Public Attributes

- static const constexpr size_t vect_size = 4
- static const constexpr size_t alignment = 32

12.295.1 Member Typedef Documentation

12.295.1.1 vect_t

```
using vect_t = __m256d
```

12.295.1.2 scalar_t

```
using scalar_t = double
```

12.295.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.295.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.295.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.295.2 Member Function Documentation

12.295.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.295.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.295.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.295.2.4 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.295.2.5 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.295.2.6 set()

```
static INLINE CONST vect_t set (
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4)  [inline], [static]
```

12.295.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.295.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.295.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.295.2.10 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.295.2.11 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.295.2.12 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.295.2.13 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.14 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.17 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.18 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.19 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.20 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.21 transpose()

```
static INLINE void transpose (
             vect_t & r0,
             vect_t & r1,
             vect_t & r2,
             vect_t & r3)  [inline], [static]
```

12.295.2.22 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.295.2.23 blendv()

```
static INLINE CONST vect_t blendv (
             const vect_t a,
             const vect_t b,
             const vect_t mask)  [inline], [static]
```

12.295.2.24 add()

```
static INLINE CONST vect_t add (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.295.2.25 addin()

```
static INLINE vect_t addin (
             vect_t & a,
             const vect_t b)  [inline], [static]
```

12.295.2.26 sub()

```
static INLINE CONST vect_t sub (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.295.2.27 subin()

```
static INLINE CONST vect_t subin (
             vect_t & a,
             const vect_t b)  [inline], [static]
```

12.295.2.28 mul()

```
static INLINE CONST vect_t mul (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.295.2.29 mulin()

```
static INLINE CONST vect_t mulin (
             vect_t & a,
             const vect_t b)  [inline], [static]
```

12.295.2.30 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.31 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.32 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.34 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.35 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.36 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.37 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.38 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.39 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.40 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.42 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.43 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.44 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.45 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.46 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a)  [inline], [static]
```

12.295.2.47 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a)  [inline], [static]
```

12.295.2.48 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.295.2.49 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.50 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.295.2.51 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.295.3 Field Documentation

12.295.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

12.295.3.2 alignment

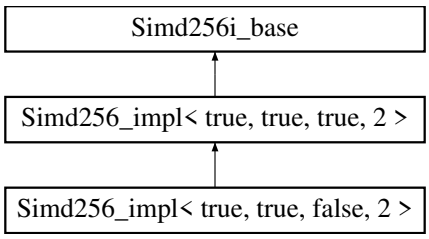
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_double.inl

12.296 Simd256_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 2 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint16_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>

- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using vect_t = __m256i
- using half_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)

- template<uint64_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 16
- static const constexpr size_t alignment = 32

12.296.1 Member Typedef Documentation

12.296.1.1 scalar_t

```
using scalar_t = uint16_t
```

12.296.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.296.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.296.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.296.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.296.1.6 vect_t

```
using vect_t = __m256i  [inherited]
```

12.296.1.7 half_t

```
using half_t = __m128i  [inherited]
```

12.296.2 Member Function Documentation

12.296.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.296.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.296.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8,
            const scalar_t x9,
            const scalar_t x10,
            const scalar_t x11,
            const scalar_t x12,
            const scalar_t x13,
            const scalar_t x14,
            const scalar_t x15)  [inline], [static]
```

12.296.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.296.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.296.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.296.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.296.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.296.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.296.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.296.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.296.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.296.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.296.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.296.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.296.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.296.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.296.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.296.2.27 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8,
            const scalar_t x9,
            const scalar_t x10,
            const scalar_t x11,
            const scalar_t x12,
            const scalar_t x13,
            const scalar_t x14,
            const scalar_t x15)  [inline], [static], [inherited]
```

12.296.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.296.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.296.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.296.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.296.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.296.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.296.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.296.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.296.2.36 shuffle()

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.296.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static], [inherited]
```

12.296.2.46 blend() [1/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.47 blend() [2/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.48 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.49 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.50 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.51 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.52 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.296.2.62 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static], [inherited]
```

12.296.2.63 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.296.3 Field Documentation

12.296.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr], [inherited]
```

12.296.3.2 alignment

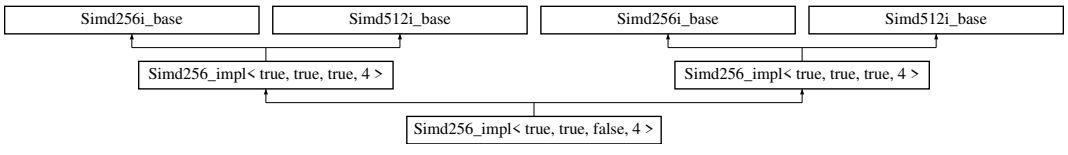
```
const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

12.297 Simd256_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 4 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint32_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>

- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using scalar_t = uint32_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using vect_t = __m256i
- using vect_t = __m512i
- using half_t = __m128i
- using half_t = __m256i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)

- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle_twice (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle_twice (const vect_t a)
- template<uint32_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- template<uint64_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)

- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- template<uint16_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 32

12.297.1 Member Typedef Documentation

12.297.1.1 scalar_t [1/2]

```
using scalar_t = uint32_t
```

12.297.1.2 aligned_allocator [1/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.297.1.3 aligned_vector [1/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.297.1.4 is_same_element [1/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.297.1.5 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.297.1.6 scalar_t [2/2]

```
using scalar_t = uint32_t
```

12.297.1.7 aligned_allocator [2/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.297.1.8 aligned_vector [2/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.297.1.9 is_same_element [2/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.297.1.10 simdHalf [2/2]

```
using simdHalf = Simd128<scalar_t>
```

12.297.1.11 vect_t [1/2]

```
using vect_t = __m256i  [inherited]
```

12.297.1.12 vect_t [2/2]

```
using vect_t = __m512i  [inherited]
```

12.297.1.13 half_t [1/2]

```
using half_t = __m128i  [inherited]
```

12.297.1.14 half_t [2/2]

```
using half_t = __m256i  [inherited]
```

12.297.2 Member Function Documentation

12.297.2.1 type_string() [1/2]

```
static const std::string type_string ()  [inline], [static]
```

12.297.2.2 set1() [1/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.297.2.3 set() [1/4]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.297.2.4 gather() [1/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.297.2.5 load() [1/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.6 loadu() [1/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.7 store() [1/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.8 storeu() [1/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.9 stream() [1/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.297.2.10 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.297.2.11 greater() [1/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.12 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.13 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.14 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.15 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.16 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.17 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.18 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.19 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.20 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.21 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.22 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.23 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.297.2.24 type_string() [2/2]

```
static const std::string type_string ()  [inline], [static]
```

12.297.2.25 set1() [2/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.297.2.26 set() [2/4]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.297.2.27 gather() [2/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.297.2.28 load() [2/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.29 loadu() [2/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.30 store() [2/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.31 storeu() [2/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.32 stream() [2/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.297.2.33 sra() [2/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.297.2.34 greater() [2/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.35 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.36 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.37 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.38 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.39 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.40 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.41 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.42 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.43 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.44 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.45 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.46 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.297.2.47 valid() [1/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.297.2.48 valid() [2/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.297.2.49 compliant() [1/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.297.2.50 compliant() [2/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.297.2.51 set1() [3/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.297.2.52 set() [3/4]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static], [inherited]
```

12.297.2.53 set() [4/4]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8,
            const scalar_t x9,
            const scalar_t x10,
            const scalar_t x11,
            const scalar_t x12,
            const scalar_t x13,
            const scalar_t x14,
            const scalar_t x15)  [inline], [static], [inherited]
```

12.297.2.54 gather() [3/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.297.2.55 load() [3/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.297.2.56 loadu() [3/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.297.2.57 store() [3/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.297.2.58 storeu() [3/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.297.2.59 stream() [3/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.297.2.60 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.61 sll() [2/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.62 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.63 srl() [2/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.64 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.65 shuffle_twice() [2/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.66 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.67 shuffle() [2/2]

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.68 unpacklo_intrinsic() [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.69 unpacklo_intrinsic() [2/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.70 unpackhi_intrinsic() [1/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.71 unpackhi_intrinsic() [2/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.72 unpacklo() [1/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.73 unpacklo() [2/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.74 unpackhi() [1/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.75 unpackhi() [2/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.76 unpacklohi() [1/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.77 unpacklohi() [2/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.78 pack_even() [1/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.79 pack_even() [2/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.80 pack_odd() [1/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.81 pack_odd() [2/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.82 pack() [1/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.83 pack() [2/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.84 transpose() [1/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.297.2.85 transpose() [2/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static], [inherited]
```

12.297.2.86 blend() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.87 blend() [2/2]

```
template<uint16_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.88 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.89 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.90 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.91 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.92 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.93 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.94 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.95 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.96 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.97 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.98 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.99 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.100 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.101 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.102 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.103 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.104 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.105 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.106 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.107 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.108 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.109 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.110 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.111 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.112 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.113 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.114 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.115 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.116 mod() [1/2]

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
```

```
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static], [inherited]
```

12.297.2.117 mod() [2/2]

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static], [inherited]
```

12.297.2.118 zero() [1/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.297.2.119 zero() [2/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.297.2.120 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.121 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.122 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.123 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.3 Field Documentation

12.297.3.1 vect_size

```
static const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

12.297.3.2 alignment

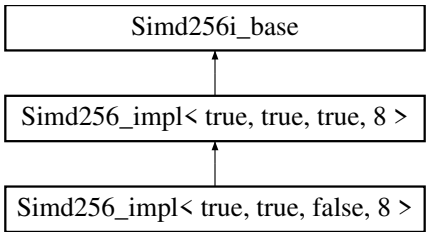
```
static const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.298 Simd256_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 8 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint64_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using vect_t = __m256i
- using half_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mullo (vect_t a, vect_t b)
- static INLINE CONST vect_t mulhi (vect_t a, vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)

- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- template<int idx>
  static INLINE CONST scalar_t get (vect_t v)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m256d &P, const __m256d &INVP, const __m256d &NEGP, const vect_t &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)
- static INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 4
- static const constexpr size_t alignment = 32

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.298.1 Member Typedef Documentation

12.298.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.298.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.298.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.298.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.298.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.298.1.6 vect_t

```
using vect_t = __m256i  [inherited]
```

12.298.1.7 half_t

```
using half_t = __m128i  [inherited]
```

12.298.2 Member Function Documentation

12.298.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.298.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.298.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static]
```

12.298.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.298.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.298.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.298.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.298.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.298.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.298.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.298.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.15 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.16 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.17 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.18 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.19 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.20 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.21 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.22 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.23 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.24 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.298.2.25 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.298.2.26 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.298.2.27 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.298.2.28 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static], [inherited]
```

12.298.2.29 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.298.2.30 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v)  [inline], [static], [inherited]
```

12.298.2.31 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.298.2.32 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.298.2.33 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.298.2.34 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.298.2.35 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.298.2.36 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.298.2.37 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.298.2.38 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.298.2.39 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.40 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.43 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.44 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.45 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.46 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.47 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static], [inherited]
```

12.298.2.48 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.49 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.50 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.51 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.52 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.298.2.62 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static], [inherited]
```

12.298.2.63 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static], [inherited]
```

12.298.2.64 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m256d & P,
            const __m256d & INVP,
            const __m256d & NEGP,
            const vect_t & POW50REM,
            const __m256d & MIN,
            const __m256d & MAX,
            __m256d & Q,
            __m256d & T)  [static], [inherited]
```

12.298.2.65 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.298.2.66 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.298.3 Field Documentation

12.298.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr], [inherited]
```

12.298.3.2 alignment

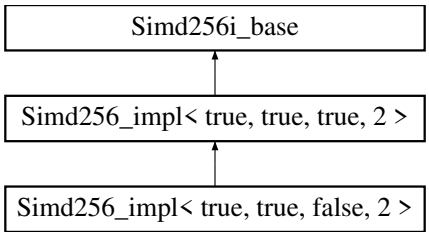
```
const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd256_int64.inl

12.299 Simd256_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 2 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m256i
- using half_t = __m128i
- using scalar_t = int16_t
- using simdHalf = Simd128<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint64_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)

- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 16
- static const constexpr size_t alignment = 32

12.299.1 Member Typedef Documentation

12.299.1.1 vect_t

```
using vect_t = __m256i
```

12.299.1.2 half_t

```
using half_t = __m128i
```

12.299.1.3 scalar_t

```
using scalar_t = int16_t
```

12.299.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.299.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.299.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.299.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.299.2 Member Function Documentation

12.299.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.299.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.299.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.299.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.299.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8,
            const scalar_t x9,
            const scalar_t x10,
            const scalar_t x11,
            const scalar_t x12,
            const scalar_t x13,
            const scalar_t x14,
            const scalar_t x15)  [inline], [static]
```

12.299.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.299.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.299.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.299.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.299.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.299.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.299.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.299.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.299.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.299.2.15 shuffle()

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.299.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static]
```

12.299.2.25 blend() [1/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.26 blend() [2/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.299.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.299.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.299.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.299.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.299.2.54 mod()

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.299.2.55 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.299.3 Field Documentation

12.299.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr]
```

12.299.3.2 alignment

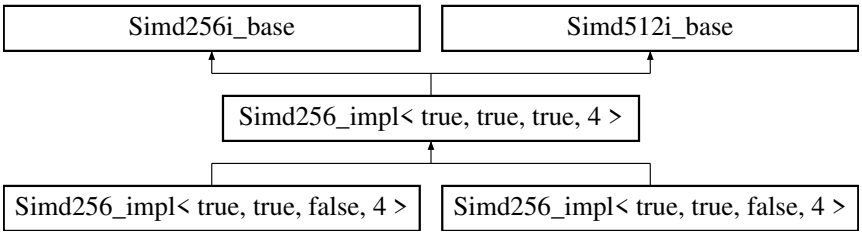
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

12.300 Simd256_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 4 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m256i
- using half_t = __m128i
- using scalar_t = int32_t
- using simdHalf = Simd128<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m512i
- using half_t = __m256i
- using scalar_t = int32_t
- using simdHalf = Simd256<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle_twice (const vect_t a)

- template<uint32_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)

- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle_twice (const vect_t a)
- template<uint64_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulx (vect_t a, vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)

- static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 32

12.300.1 Member Typedef Documentation

12.300.1.1 vect_t [1/2]

```
using vect_t = __m256i
```

12.300.1.2 half_t [1/2]

```
using half_t = __m128i
```

12.300.1.3 scalar_t [1/2]

```
using scalar_t = int32_t
```

12.300.1.4 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.300.1.5 aligned_allocator [1/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.300.1.6 aligned_vector [1/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.300.1.7 is_same_element [1/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.300.1.8 vect_t [2/2]

```
using vect_t = __m512i
```

12.300.1.9 half_t [2/2]

```
using half_t = __m256i
```

12.300.1.10 scalar_t [2/2]

```
using scalar_t = int32_t
```

12.300.1.11 simdHalf [2/2]

```
using simdHalf = Simd256<scalar_t>
```

12.300.1.12 aligned_allocator [2/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.300.1.13 aligned_vector [2/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.300.1.14 is_same_element [2/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.300.2 Member Function Documentation

12.300.2.1 type_string() [1/2]

```
static const std::string type_string ()  [inline], [static]
```

12.300.2.2 valid() [1/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.300.2.3 compliant() [1/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.300.2.4 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.300.2.5 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.300.2.6 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.300.2.7 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.8 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.9 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.10 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.11 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.300.2.12 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.300.2.13 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.300.2.14 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.300.2.15 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static]
```

12.300.2.16 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.300.2.17 unpacklo_intrinsic() [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.18 unpackhi_intrinsic() [1/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.19 unpacklo() [1/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.20 unpackhi() [1/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.21 unpacklohi() [1/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.22 pack_even() [1/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.23 pack_odd() [1/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.24 pack() [1/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.25 transpose() [1/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.300.2.26 blend() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.27 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.28 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.29 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.30 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.31 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.32 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.33 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.34 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.300.2.35 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.36 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.37 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.38 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.39 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.40 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.41 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.42 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.43 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.44 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.45 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.46 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.47 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.48 greater() [1/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.49 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.50 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.51 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.52 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.300.2.53 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.300.2.54 mod() [1/2]

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.300.2.55 type_string() [2/2]

```
static const std::string type_string ()  [inline], [static]
```

12.300.2.56 valid() [2/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.300.2.57 compliant() [2/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.300.2.58 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.300.2.59 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8,
            const scalar_t x9,
            const scalar_t x10,
            const scalar_t x11,
            const scalar_t x12,
            const scalar_t x13,
            const scalar_t x14,
            const scalar_t x15)  [inline], [static]
```

12.300.2.60 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.300.2.61 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.62 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.63 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.64 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.65 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.300.2.66 sll() [2/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.300.2.67 srl() [2/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.300.2.68 sra() [2/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.300.2.69 shuffle_twice() [2/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static]
```

12.300.2.70 shuffle() [2/2]

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.300.2.71 unpacklo_intrinsic() [2/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.72 unpackhi_intrinsic() [2/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.73 unpacklo() [2/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.74 unpackhi() [2/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.75 unpacklohi() [2/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.76 pack_even() [2/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.77 pack_odd() [2/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.78 pack() [2/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.79 transpose() [2/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static]
```

12.300.2.80 blend() [2/2]

```
template<uint16_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.81 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.82 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.83 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.84 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.85 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.86 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.87 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.88 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.300.2.89 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.90 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.91 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.92 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.93 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.94 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.95 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.96 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.97 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.98 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.99 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.100 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.101 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.102 greater() [2/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.103 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.104 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.105 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.106 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.300.2.107 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.300.2.108 mod() [2/2]

```
static INLINE vect_t mod (
            vect_t & C,
            const vect_t & P,
            const vect_t & INVP,
            const vect_t & NEGP,
            const vect_t & MIN,
            const vect_t & MAX,
            vect_t & Q,
            vect_t & T)  [inline], [static]
```

12.300.2.109 zero() [1/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.300.2.110 zero() [2/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.300.2.111 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.112 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.113 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.114 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.3 Field Documentation

12.300.3.1 vect_size

```
static const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.300.3.2 alignment

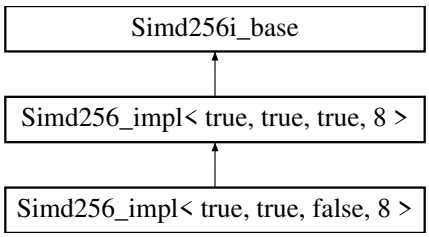
```
static const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.301 Simd256_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 8 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m256i
- using half_t = __m128i
- using scalar_t = int64_t
- using simdHalf = Simd128<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- template<int idx>
  static INLINE CONST scalar_t get (vect_t v)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)

- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (vect_t a, vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (vect_t a, vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m256d &P, const __m256d &INVP, const __m256d &NEGP, const vect_t &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)
- static INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 4
- static const constexpr size_t alignment = 32

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.301.1 Member Typedef Documentation

12.301.1.1 vect_t

```
using vect_t = __m256i
```

12.301.1.2 half_t

```
using half_t = __m128i
```

12.301.1.3 scalar_t

```
using scalar_t = int64_t
```

12.301.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.301.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.301.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.301.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.301.2 Member Function Documentation

12.301.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.301.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.301.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.301.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.301.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static]
```

12.301.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.301.2.7 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v)  [inline], [static]
```

12.301.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.301.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.301.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.301.2.11 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.301.2.12 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.301.2.13 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.301.2.14 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.301.2.15 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.301.2.16 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.301.2.17 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.18 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.21 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.22 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.23 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.24 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.25 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static]
```

12.301.2.26 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.301.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.301.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.301.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.301.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.301.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.301.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.301.2.55 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.301.2.56 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m256d & P,
            const __m256d & INVP,
            const __m256d & NEGP,
            const vect_t & POW50REM,
            const __m256d & MIN,
            const __m256d & MAX,
            __m256d & Q,
            __m256d & T)  [static]
```

12.301.2.57 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.301.2.58 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.301.3 Field Documentation

12.301.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

12.301.3.2 alignment

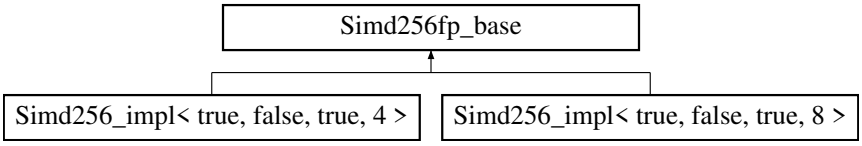
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int64.inl

12.302 Simd256fp_base Struct Reference

Inheritance diagram for Simd256fp_base:

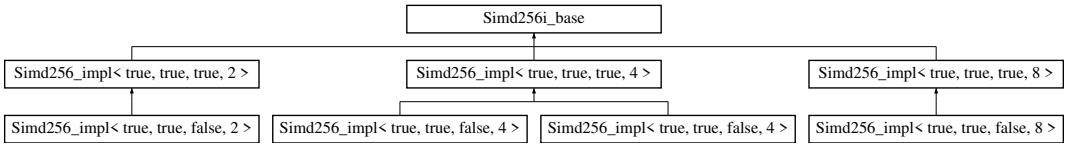


The documentation for this struct was generated from the following file:

- simd256.inl

12.303 Simd256i_base Struct Reference

Inheritance diagram for Simd256i_base:



Public Types

- using vect_t = __m256i

Static Public Member Functions

- static INLINE CONST vect_t zero ()

12.303.1 Member Typedef Documentation

12.303.1.1 vect_t

```
using vect_t = __m256i
```

12.303.2 Member Function Documentation

12.303.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd256.inl

12.304 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd512.inl

12.305 Simd512_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- simd512_float.inl

12.306 Simd512_impl< true, false, true, 8 > Struct Reference

Public Types

- using vect_t = __m512d
- using scalar_t = double
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (const scalar_t *p, const vect_t v)
- static INLINE void storeu (const scalar_t *p, const vect_t v)
- static INLINE void stream (const scalar_t *p, const vect_t v)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t blendv (const vect_t a, const vect_t b, const vect_t mask)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE CONST vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t div (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)

- static INLINE CONST vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t floor (const vect_t a)
- static INLINE CONST vect_t ceil (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t hadd (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 64

12.306.1 Member Typedef Documentation

12.306.1.1 vect_t

```
using vect_t = __m512d
```

12.306.1.2 scalar_t

```
using scalar_t = double
```

12.306.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.306.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.306.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.306.2 Member Function Documentation

12.306.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.306.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.306.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.306.2.4 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.306.2.5 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.306.2.6 set()

```
static INLINE CONST vect_t set (
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7,
            const scalar_t x8)  [inline], [static]
```

12.306.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.306.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.306.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.306.2.10 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.11 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.12 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.13 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.306.2.14 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.15 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.16 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.17 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.18 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.19 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.20 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.21 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.22 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.306.2.23 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.24 blendv()

```
static INLINE CONST vect_t blendv (
            const vect_t a,
            const vect_t b,
            const vect_t mask)  [inline], [static]
```

12.306.2.25 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.26 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.27 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.28 subin()

```
static INLINE CONST vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.29 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.30 mulin()

```
static INLINE CONST vect_t mulin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.31 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.32 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.33 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.34 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.35 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.36 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.37 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.38 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.39 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.40 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.41 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.42 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.43 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a)  [inline], [static]
```

12.306.2.44 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a)  [inline], [static]
```

12.306.2.45 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.306.2.46 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.47 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.306.3 Field Documentation

12.306.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.306.3.2 alignment

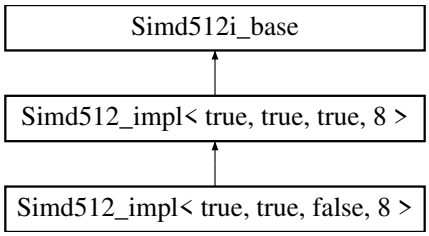
```
const constexpr size_t alignment = 64  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd512_double.inl

12.307 Simd512_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd512_impl< true, true, false, 8 >:



Data Structures

- union Converter

Public Types

- using scalar_t = uint64_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd256<scalar_t>
- using vect_t = __m512i
- using half_t = __m256i

Static Public Member Functions

- static const std::string type_string ()
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- template<uint8_t k>
  static INLINE void maskstore (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- static INLINE CONST vect_t greater (vect_t a, vect_t b)
- static INLINE CONST vect_t lesser (vect_t a, vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mullo (vect_t a, vect_t b)
- static INLINE CONST vect_t mulhi (vect_t a, vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- template<class T>
  static constexpr bool valid (T *p)

- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- template<uint8_t k>
  static INLINE void maskstore (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 64

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.307.1 Member Typedef Documentation

12.307.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.307.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.307.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.307.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.307.1.5 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.307.1.6 vect_t

```
using vect_t = __m512i  [inherited]
```

12.307.1.7 half_t

```
using half_t = __m256i  [inherited]
```

12.307.2 Member Function Documentation

12.307.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.307.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.307.2.3 set() [1/3]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.307.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.307.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.307.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.307.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.8 maskstore() [1/2]

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.9 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.10 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.307.2.11 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.307.2.12 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.13 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.14 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.15 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.16 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.17 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.18 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.19 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.20 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.21 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.22 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.23 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.24 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.307.2.25 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.307.2.26 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.307.2.27 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.307.2.28 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.307.2.29 set() [2/3]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static], [inherited]
```

12.307.2.30 set() [3/3]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static], [inherited]
```

12.307.2.31 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
```

```
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.307.2.32 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.307.2.33 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.307.2.34 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.35 maskstore() [2/2]

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.36 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.37 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.307.2.38 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.39 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.40 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.41 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.42 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.43 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.44 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.45 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.46 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.47 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.48 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.49 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.307.2.50 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.51 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.52 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.53 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.54 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.55 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.56 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.57 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.58 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.59 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.60 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.61 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.62 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.63 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.64 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static], [inherited]
```

12.307.2.65 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static], [inherited]
```

12.307.2.66 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m512d & P,
            const __m512d & INVP,
            const __m512d & NEGP,
            const vect_t & POW50REM,
            const __m512d & MIN,
            const __m512d & MAX,
            __m512d & Q,
            __m512d & T)  [static], [inherited]
```

12.307.2.67 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.307.2.68 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.307.2.69 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.70 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.71 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.72 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.3 Field Documentation

12.307.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

12.307.3.2 alignment

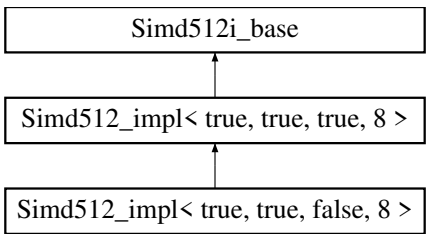
```
const constexpr size_t alignment = 64  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd512_int64.inl

12.308 Simd512_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd512_impl< true, true, true, 8 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m512i
- using half_t = __m256i
- using scalar_t = int64_t
- using simdHalf = Simd256<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- template<class T>
  static constexpr bool valid (T *p)
- template<class T>
  static constexpr bool compliant (T n)
- static INLINE CONST vect_t set1 (const scalar_t x)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)
- static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3)
- template<class T>
  static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)
- static INLINE PURE vect_t load (const scalar_t *const p)
- static INLINE PURE vect_t loadu (const scalar_t *const p)
- static INLINE void store (scalar_t *p, vect_t v)
- template<uint8_t k>
  static INLINE void maskstore (scalar_t *p, vect_t v)
- static INLINE void storeu (scalar_t *p, vect_t v)
- static INLINE void stream (scalar_t *p, const vect_t v)
- template<int s>
  static INLINE CONST vect_t sll (const vect_t a)
- template<int s>
  static INLINE CONST vect_t srl (const vect_t a)
- template<int s>
  static INLINE CONST vect_t sra (const vect_t a)
- template<uint8_t s>
  static INLINE CONST vect_t shuffle (const vect_t a)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- static INLINE CONST vect_t add (const vect_t a, const vect_t b)
- static INLINE vect_t addin (vect_t &a, const vect_t b)

- static INLINE CONST vect_t sub (const vect_t a, const vect_t b)
- static INLINE vect_t subin (vect_t &a, const vect_t b)
- static INLINE CONST vect_t mullo (vect_t a, vect_t b)
- static INLINE CONST vect_t mul (const vect_t a, const vect_t b)
- static INLINE CONST vect_t mulhi (vect_t a, vect_t b)
- static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)
- static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)
- static INLINE CONST vect_t eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)
- static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)
- static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)
- static INLINE CONST scalar_t hadd_to_scal (const vect_t a)
- static INLINE CONST vect_t round (const vect_t a)
- static INLINE CONST vect_t mask_high ()
- static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)
- static INLINE vect_t mod (vect_t &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)
- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

Static Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 64

Static Protected Member Functions

- static INLINE CONST vect_t signbits (const vect_t x)

12.308.1 Member Typedef Documentation

12.308.1.1 vect_t

```
using vect_t = __m512i
```

12.308.1.2 half_t

```
using half_t = __m256i
```

12.308.1.3 scalar_t

```
using scalar_t = int64_t
```

12.308.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.308.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.308.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.308.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.308.2 Member Function Documentation

12.308.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.308.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.308.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.308.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.308.2.5 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3,
            const scalar_t x4,
            const scalar_t x5,
            const scalar_t x6,
            const scalar_t x7)  [inline], [static]
```

12.308.2.6 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1,
            const scalar_t x2,
            const scalar_t x3)  [inline], [static]
```

12.308.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.308.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.308.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.308.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.11 maskstore()

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.12 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.13 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.308.2.14 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.308.2.15 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.308.2.16 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.308.2.17 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.308.2.18 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.19 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.20 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.21 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.22 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.23 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.24 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.25 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.26 transpose()

```
static INLINE void transpose (
             vect_t & r0,
             vect_t & r1,
             vect_t & r2,
             vect_t & r3,
             vect_t & r4,
             vect_t & r5,
             vect_t & r6,
             vect_t & r7)  [inline], [static]
```

12.308.2.27 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.308.2.28 add()

```
static INLINE CONST vect_t add (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.308.2.29 addin()

```
static INLINE vect_t addin (
             vect_t & a,
             const vect_t b)  [inline], [static]
```

12.308.2.30 sub()

```
static INLINE CONST vect_t sub (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.308.2.31 subin()

```
static INLINE vect_t subin (
             vect_t & a,
             const vect_t b)  [inline], [static]
```

12.308.2.32 mullo()

```
static INLINE CONST vect_t mullo (
             vect_t a,
             vect_t b)  [inline], [static]
```

12.308.2.33 mul()

```
static INLINE CONST vect_t mul (
             const vect_t a,
             const vect_t b)  [inline], [static]
```

12.308.2.34 mulhi()

```
static INLINE CONST vect_t mulhi (
             vect_t a,
             vect_t b)  [inline], [static]
```

12.308.2.35 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.36 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.37 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.38 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.39 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.40 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.41 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.42 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.43 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.44 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.45 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.46 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.47 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.48 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.49 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.50 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.51 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.52 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.53 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.308.2.54 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.308.2.55 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.308.2.56 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.308.2.57 mod()

```
INLINE vect_t mod (
            vect_t & C,
            const __m512d & P,
            const __m512d & INVP,
            const __m512d & NEGP,
            const vect_t & POW50REM,
            const __m512d & MIN,
            const __m512d & MAX,
            __m512d & Q,
            __m512d & T)  [static]
```

12.308.2.58 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.308.2.59 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.308.2.60 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.61 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.62 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.63 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.3 Field Documentation

12.308.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.308.3.2 alignment

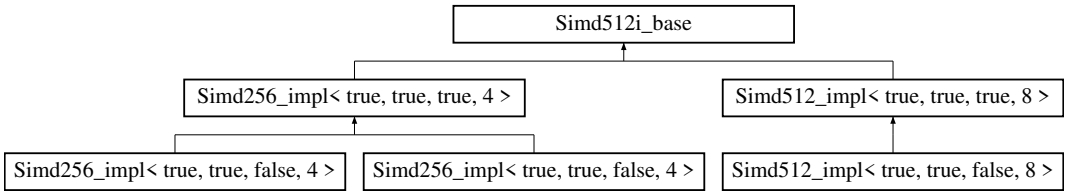
```
const constexpr size_t alignment = 64  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd512_int64.inl

12.309 Simd512i_base Struct Reference

Inheritance diagram for Simd512i_base:



Public Types

- using vect_t = __m512i

Static Public Member Functions

- static INLINE CONST vect_t zero ()
- static INLINE CONST vect_t vor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vand (const vect_t a, const vect_t b)
- static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)

12.309.1 Member Typedef Documentation

12.309.1.1 vect_t

```
using vect_t = __m512i
```

12.309.2 Member Function Documentation

12.309.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.309.2.2 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.3 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.4 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.5 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd512.inl

12.310 SimdChooser< T, bool, bool > Struct Template Reference

`#include <fflas_simd.h>`

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.311 SimdChooser< T, false, b > Struct Template Reference

`#include <fflas_simd.h>`

Public Types

- using value = NoSimd<T>

12.311.1 Member Typedef Documentation

12.311.1.1 value

```
template<class T, bool b>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.312 SimdChooser< T, true, false > Struct Template Reference

`#include <fflas_simd.h>`

Public Types

- using value = NoSimd<T>

12.312.1 Member Typedef Documentation

12.312.1.1 value

```
template<class T>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.313 SimdChooser< T, true, true > Struct Template Reference

`#include <fflas_simd.h>`

Public Types

- using value = NoSimd<T>

12.313.1 Member Typedef Documentation

12.313.1.1 value

```
template<class T>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.314 simdToType< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.315 Single Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.316 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference

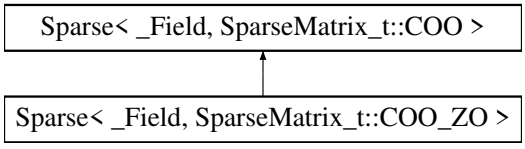
The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.317 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference

`#include <coo.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::COO >:



Public Types

- using Field = _Field

Data Fields

- index_t * col = nullptr
- index_t * row = nullptr
- _Field::Element_ptr dat
- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0

12.317.1 Member Typedef Documentation

12.317.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.317.2 Field Documentation

12.317.2.1 col

```
template<class _Field>
index_t* col = nullptr
```

12.317.2.2 row

```
template<class _Field>
index_t* row = nullptr
```

12.317.2.3 dat

```
template<class _Field>
_Field::Element_ptr dat
```

12.317.2.4 delayed

```
template<class _Field>
bool delayed = false
```

12.317.2.5 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.317.2.6 m

```
template<class _Field>
index_t m = 0
```

12.317.2.7 n

```
template<class _Field>
index_t n = 0
```

12.317.2.8 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.317.2.9 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.317.2.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

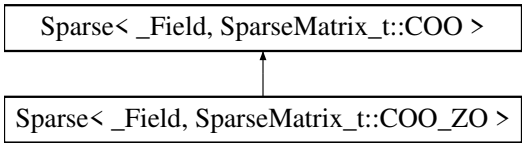
The documentation for this struct was generated from the following file:

- coo.h

12.318 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference

`#include <coo.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::COO_ZO >:



Public Types

- using Field = _Field

Data Fields

- _Field::Element cst = 1
- index_t * col = nullptr
- index_t * row = nullptr
- _Field::Element_ptr dat
- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0

12.318.1 Member Typedef Documentation

12.318.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.318.2 Field Documentation

12.318.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.318.2.2 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.318.2.3 row

```
template<class _Field>
index_t* row = nullptr  [inherited]
```

12.318.2.4 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

12.318.2.5 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.318.2.6 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.318.2.7 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.318.2.8 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.318.2.9 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.318.2.10 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.318.2.11 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

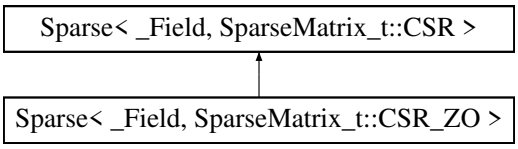
The documentation for this struct was generated from the following file:

- coo.h

12.319 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference

`#include <csr.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::CSR >:



Public Types

- using Field = _Field

Data Fields

- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- index_t * col = nullptr
- index_t * st = nullptr
- index_t * stend = nullptr
- _Field::Element_ptr dat

12.319.1 Member Typedef Documentation

12.319.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.319.2 Field Documentation

12.319.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.319.2.2 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.319.2.3 m

```
template<class _Field>
index_t m = 0
```

12.319.2.4 n

```
template<class _Field>
index_t n = 0
```

12.319.2.5 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.319.2.6 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.319.2.7 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.319.2.8 col

```
template<class _Field>
index_t* col = nullptr
```

12.319.2.9 st

```
template<class _Field>
index_t* st = nullptr
```

12.319.2.10 stend

```
template<class _Field>
index_t* stend = nullptr
```

12.319.2.11 dat

```
template<class _Field>
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- csr.h

12.320 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference

```
#include <csr_hyb.h>
```

Public Types

- using Field = _Field

Data Fields

- bool delayed = false
- index_t * col = nullptr
- index_t * st = nullptr
- _Field::Element_ptr dat
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- uint64_t nnz = 0
- uint64_t nElements = 0

- uint64_t maxrow = 0
- uint64_t nOnes = 0
- uint64_t nMOnes = 0
- uint64_t nOthers = 0

12.320.1 Member Typedef Documentation

12.320.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.320.2 Field Documentation

12.320.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.320.2.2 col

```
template<class _Field>
index_t* col = nullptr
```

12.320.2.3 st

```
template<class _Field>
index_t* st = nullptr
```

12.320.2.4 dat

```
template<class _Field>
_Field::Element_ptr dat
```

12.320.2.5 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.320.2.6 m

```
template<class _Field>
index_t m = 0
```

12.320.2.7 n

```
template<class _Field>
index_t n = 0
```

12.320.2.8 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.320.2.9 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.320.2.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.320.2.11 nOnes

```
template<class _Field>
uint64_t nOnes = 0
```

12.320.2.12 nMOnes

```
template<class _Field>
uint64_t nMOnes = 0
```

12.320.2.13 nOthers

```
template<class _Field>
uint64_t nOthers = 0
```

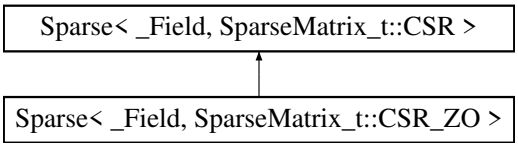
The documentation for this struct was generated from the following file:

- csr_hyb.h

12.321 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference

`#include <csr.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::CSR_ZO >:



Public Types

- using Field = _Field

Data Fields

- int64_t cst = 1
- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- index_t * col = nullptr
- index_t * st = nullptr
- index_t * stend = nullptr
- _Field::Element_ptr dat

12.321.1 Member Typedef Documentation

12.321.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.321.2 Field Documentation

12.321.2.1 cst

```
template<class _Field>
int64_t cst = 1
```

12.321.2.2 delayed

```
template<class _Field>
bool delayed = false
```

12.321.2.3 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.321.2.4 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.321.2.5 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.321.2.6 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.321.2.7 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.321.2.8 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

12.321.2.9 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.321.2.10 st

```
template<class _Field>
index_t* st = nullptr  [inherited]
```

12.321.2.11 stend

```
template<class _Field>
index_t* stend = nullptr  [inherited]
```

12.321.2.12 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

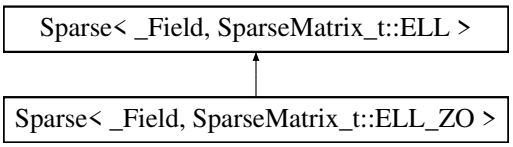
The documentation for this struct was generated from the following file:

- csr.h

12.322 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL >:



Public Types

- using Field = _Field

Data Fields

- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- index_t ld = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- index_t * col = nullptr
- _Field::Element_ptr dat

12.322.1 Member Typedef Documentation

12.322.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.322.2 Field Documentation

12.322.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.322.2.2 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.322.2.3 m

```
template<class _Field>
index_t m = 0
```

12.322.2.4 n

```
template<class _Field>
index_t n = 0
```

12.322.2.5 ld

```
template<class _Field>
index_t ld = 0
```

12.322.2.6 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.322.2.7 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.322.2.8 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.322.2.9 col

```
template<class _Field>
index_t* col = nullptr
```

12.322.2.10 dat

```
template<class _Field>
_Field::Element_ptr dat
```

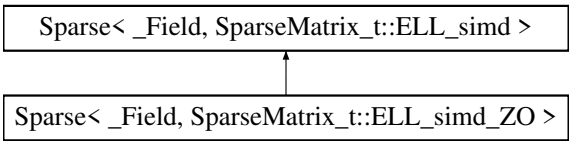
The documentation for this struct was generated from the following file:

- ell.h

12.323 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference

`#include <ell_simd.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_simd >:



Data Fields

- bool delayed = false
- int chunk = 0
- index_t m = 0
- index_t n = 0
- index_t ld = 0
- uint64_t kmax = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- uint64_t nChunks = 0
- index_t * col = nullptr
- _Field::Element_ptr dat

12.323.1 Field Documentation

12.323.1.1 delayed

```
template<class _Field>
bool delayed = false
```

12.323.1.2 chunk

```
template<class _Field>
int chunk = 0
```

12.323.1.3 m

```
template<class _Field>
index_t m = 0
```

12.323.1.4 n

```
template<class _Field>
index_t n = 0
```

12.323.1.5 ld

```
template<class _Field>
index_t ld = 0
```

12.323.1.6 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.323.1.7 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.323.1.8 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.323.1.9 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.323.1.10 nChunks

```
template<class _Field>
uint64_t nChunks = 0
```

12.323.1.11 col

```
template<class _Field>
index_t* col = nullptr
```

12.323.1.12 dat

```
template<class _Field>
_Field::Element_ptr dat
```

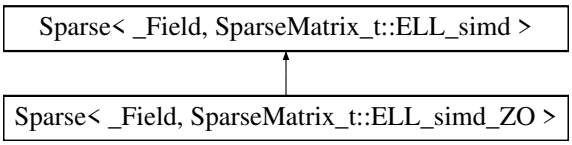
The documentation for this struct was generated from the following file:

- ell_simd.h

12.324 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference

`#include <ell_simd.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >:



Data Fields

- _Field::Element cst = 1
- bool delayed = false
- int chunk = 0
- index_t m = 0
- index_t n = 0
- index_t ld = 0
- uint64_t kmax = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- uint64_t nChunks = 0
- index_t * col = nullptr
- _Field::Element_ptr dat

12.324.1 Field Documentation

12.324.1.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.324.1.2 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.324.1.3 chunk

```
template<class _Field>
int chunk = 0  [inherited]
```

12.324.1.4 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.324.1.5 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.324.1.6 ld

```
template<class _Field>
index_t ld = 0  [inherited]
```

12.324.1.7 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.324.1.8 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.324.1.9 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.324.1.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

12.324.1.11 nChunks

```
template<class _Field>
uint64_t nChunks = 0  [inherited]
```

12.324.1.12 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.324.1.13 dat

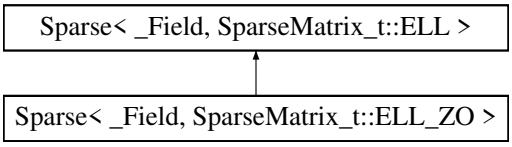
```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

The documentation for this struct was generated from the following file:

- ell_simd.h

12.325 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference

`#include <ell.h>`
Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_ZO >:



Public Types

- using Field = _Field

Data Fields

- _Field::Element cst = 1
- bool delayed = false
- uint64_t kmax = 0
- index_t m = 0
- index_t n = 0
- index_t ld = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- uint64_t maxrow = 0
- index_t * col = nullptr
- _Field::Element_ptr dat

12.325.1 Member Typedef Documentation

12.325.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.325.2 Field Documentation

12.325.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.325.2.2 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.325.2.3 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.325.2.4 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.325.2.5 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.325.2.6 ld

```
template<class _Field>
index_t ld = 0  [inherited]
```

12.325.2.7 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.325.2.8 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.325.2.9 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

12.325.2.10 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.325.2.11 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

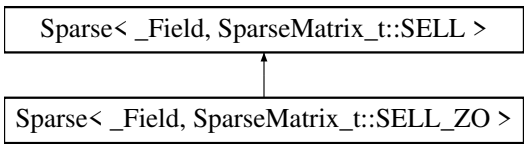
The documentation for this struct was generated from the following file:

- ell.h

12.326 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference

`#include <sell.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::SELL >:



Public Types

- using Field = _Field

Data Fields

- bool delayed = false
- int chunk = 0
- index_t kmax = 0
- index_t m = 0
- index_t n = 0
- index_t maxrow = 0
- index_t sigma = 0
- index_t nChunks = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- index_t * perm = nullptr
- uint64_t * st = nullptr
- index_t * chunkSize = nullptr
- index_t * col = nullptr
- _Field::Element_ptr dat

12.326.1 Member Typedef Documentation

12.326.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.326.2 Field Documentation

12.326.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.326.2.2 chunk

```
template<class _Field>
int chunk = 0
```

12.326.2.3 kmax

```
template<class _Field>
index_t kmax = 0
```

12.326.2.4 m

```
template<class _Field>
index_t m = 0
```

12.326.2.5 n

```
template<class _Field>
index_t n = 0
```

12.326.2.6 maxrow

```
template<class _Field>
index_t maxrow = 0
```

12.326.2.7 sigma

```
template<class _Field>
index_t sigma = 0
```

12.326.2.8 nChunks

```
template<class _Field>
index_t nChunks = 0
```

12.326.2.9 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.326.2.10 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.326.2.11 perm

```
template<class _Field>
index_t* perm = nullptr
```

12.326.2.12 st

```
template<class _Field>
uint64_t* st = nullptr
```

12.326.2.13 chunkSize

```
template<class _Field>
index_t* chunkSize = nullptr
```

12.326.2.14 col

```
template<class _Field>
index_t* col = nullptr
```

12.326.2.15 dat

```
template<class _Field>
_Field::Element_ptr dat
```

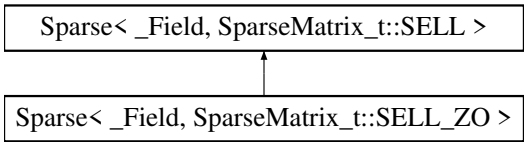
The documentation for this struct was generated from the following file:

- sell.h

12.327 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference

`#include <sell.h>`

Inheritance diagram for Sparse< _Field, SparseMatrix_t::SELL_ZO >:



Public Types

- using Field = _Field

Data Fields

- _Field::Element cst = 1
- bool delayed = false
- int chunk = 0
- index_t kmax = 0
- index_t m = 0
- index_t n = 0
- index_t maxrow = 0
- index_t sigma = 0
- index_t nChunks = 0
- uint64_t nnz = 0
- uint64_t nElements = 0
- index_t * perm = nullptr
- uint64_t * st = nullptr
- index_t * chunkSize = nullptr
- index_t * col = nullptr
- _Field::Element_ptr dat

12.327.1 Member Typedef Documentation

12.327.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.327.2 Field Documentation

12.327.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.327.2.2 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.327.2.3 chunk

```
template<class _Field>
int chunk = 0  [inherited]
```

12.327.2.4 kmax

```
template<class _Field>
index_t kmax = 0  [inherited]
```

12.327.2.5 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.327.2.6 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.327.2.7 maxrow

```
template<class _Field>
index_t maxrow = 0  [inherited]
```

12.327.2.8 sigma

```
template<class _Field>
index_t sigma = 0  [inherited]
```

12.327.2.9 nChunks

```
template<class _Field>
index_t nChunks = 0  [inherited]
```

12.327.2.10 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.327.2.11 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.327.2.12 perm

```
template<class _Field>
index_t* perm = nullptr  [inherited]
```

12.327.2.13 st

```
template<class _Field>
uint64_t* st = nullptr  [inherited]
```

12.327.2.14 chunkSize

```
template<class _Field>
index_t* chunkSize = nullptr  [inherited]
```

12.327.2.15 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.327.2.16 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

The documentation for this struct was generated from the following file:

- sell.h

12.328 SpMat< Field, flag > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- FFLAS::CooMat< Field > * _coo = nullptr
- FFLAS::CsrMat< Field > * _csr = nullptr
- FFLAS::EllMat< Field > * _ell = nullptr

12.328.1 Field Documentation

12.328.1.1 _coo

```
template<class Field, int flag = HelperFlag::none>
FFLAS::CooMat<Field>* _coo = nullptr
```

12.328.1.2 _csr

```
template<class Field, int flag = HelperFlag::none>
FFLAS::CsrMat<Field>* _csr = nullptr
```

12.328.1.3 _ell

```
template<class Field, int flag = HelperFlag::none>
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.329 StatsMatrix Struct Reference

```
#include <utils.h>
```

Data Fields

- uint64_t rowdim = 0
- uint64_t coldim = 0
- uint64_t nOnes = 0
- uint64_t nMOnes = 0
- uint64_t nOthers = 0
- uint64_t nnz = 0
- uint64_t maxRow = 0
- uint64_t minRow = 0
- uint64_t averageRow = 0
- uint64_t deviationRow = 0
- uint64_t maxCol = 0
- uint64_t minCol = 0
- uint64_t averageCol = 0
- uint64_t deviationCol = 0
- uint64_t minColDifference = 0
- uint64_t maxColDifference = 0
- uint64_t averageColDifference = 0
- uint64_t deviationColDifference = 0
- uint64_t minRowDifference = 0
- uint64_t maxRowDifference = 0
- uint64_t averageRowDifference = 0
- uint64_t deviationRowDifference = 0
- uint64_t nDenseRows = 0
- uint64_t nDenseCols = 0
- uint64_t nEmptyRows = 0

- uint64_t nEmptyCols = 0
- uint64_t nEmptyColsEnd = 0
- std::vector< uint64_t > denseRows
- std::vector< uint64_t > denseCols

12.329.1 Field Documentation

12.329.1.1 rowdim

```
uint64_t rowdim = 0
```

12.329.1.2 coldim

```
uint64_t coldim = 0
```

12.329.1.3 nOnes

```
uint64_t nOnes = 0
```

12.329.1.4 nMOnes

```
uint64_t nMOnes = 0
```

12.329.1.5 nOthers

```
uint64_t nOthers = 0
```

12.329.1.6 nnz

```
uint64_t nnz = 0
```

12.329.1.7 maxRow

```
uint64_t maxRow = 0
```

12.329.1.8 minRow

```
uint64_t minRow = 0
```

12.329.1.9 averageRow

```
uint64_t averageRow = 0
```

12.329.1.10 deviationRow

```
uint64_t deviationRow = 0
```

12.329.1.11 maxCol

```
uint64_t maxCol = 0
```

12.329.1.12 minCol

```
uint64_t minCol = 0
```

12.329.1.13 averageCol

```
uint64_t averageCol = 0
```

12.329.1.14 deviationCol

```
uint64_t deviationCol = 0
```

12.329.1.15 minColDifference

```
uint64_t minColDifference = 0
```

12.329.1.16 maxColDifference

```
uint64_t maxColDifference = 0
```

12.329.1.17 averageColDifference

```
uint64_t averageColDifference = 0
```

12.329.1.18 deviationColDifference

```
uint64_t deviationColDifference = 0
```

12.329.1.19 minRowDifference

```
uint64_t minRowDifference = 0
```

12.329.1.20 maxRowDifference

```
uint64_t maxRowDifference = 0
```

12.329.1.21 averageRowDifference

```
uint64_t averageRowDifference = 0
```

12.329.1.22 deviationRowDifference

```
uint64_t deviationRowDifference = 0
```

12.329.1.23 nDenseRows

```
uint64_t nDenseRows = 0
```

12.329.1.24 nDenseCols

```
uint64_t nDenseCols = 0
```

12.329.1.25 nEmptyRows

```
uint64_t nEmptyRows = 0
```

12.329.1.26 nEmptyCols

```
uint64_t nEmptyCols = 0
```

12.329.1.27 nEmptyColsEnd

```
uint64_t nEmptyColsEnd = 0
```

12.329.1.28 denseRows

```
std::vector<uint64_t> denseRows
```

12.329.1.29 denseCols

`std::vector<uint64_t> denseCols`
The documentation for this struct was generated from the following file:

- utils.h

12.330 string Class Reference

STL class.

Data Structures

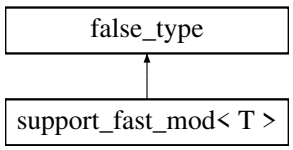
- class const_iterator
  STL iterator class.
- class const_reverse_iterator
  STL iterator class.
- class iterator
  STL iterator class.
- class reverse_iterator
  STL iterator class.

12.330.1 Detailed Description

STL class.
The documentation for this class was generated from the following files:

12.331 support_fast_mod< T > Struct Template Reference

`#include <fflas_freduce.h>`
Inheritance diagram for support_fast_mod< T >:

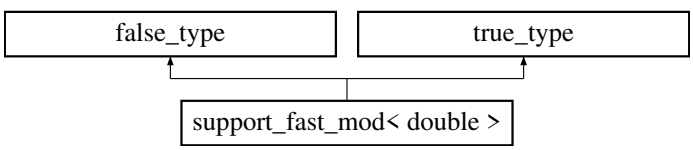


The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.332 support_fast_mod< double > Struct Reference

`#include <fflas_freduce.h>`
Inheritance diagram for support_fast_mod< double >:

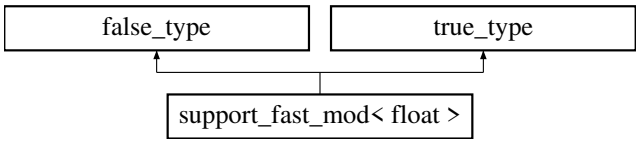


The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.333 support_fast_mod< float > Struct Reference

`#include <fflas_freduce.h>`
Inheritance diagram for support_fast_mod< float >:

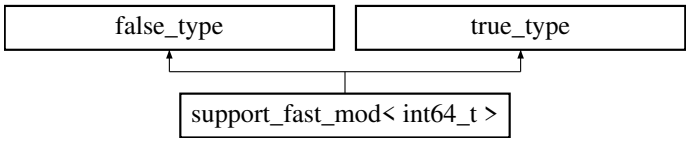


The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.334 support_fast_mod< int64_t > Struct Reference

`#include <fflas_freduce.h>`
Inheritance diagram for support_fast_mod< int64_t >:

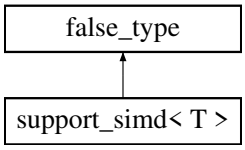


The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.335 support_simd< T > Struct Template Reference

`#include <fflas_simd.h>`
Inheritance diagram for support_simd< T >:

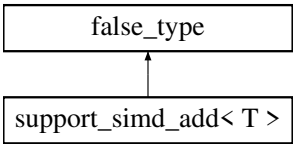


The documentation for this struct was generated from the following file:

- fflas_simd.h

12.336 support_simd_add< T > Struct Template Reference

`#include <fflas_fadd.h>`
Inheritance diagram for support_simd_add< T >:

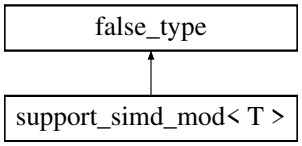


The documentation for this struct was generated from the following file:

- fflas_fadd.h

12.337 support_simd_mod< T > Struct Template Reference

`#include <fflas_freduce.h>`
Inheritance diagram for support_simd_mod< T >:



The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.338 Test< Elt > Class Template Reference

Public Types

- using Field = Modular<Elt>
- using Elt_ptr = typename Field::Element_ptr
- using Residu = typename Field::Residu_t
- template<bool B, class T = void>
  using enable_if_t = typename std::enable_if<B, T>::type
- template<typename Simd>
  using is_same_element = typename Simd::template is_same_element<Field>
- template<typename E>
  using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
- template<typename E>
  using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
- template<typename E>
  using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
- template<typename E>
  using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>

Public Member Functions

- Test (size_t mm, size_t nn)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * = nullptr>
  bool test_ftranspose (size_t m, size_t n, Elt_ptr A, size_t lda, Elt_ptr B, size_t ldb)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * = nullptr>
  bool doTests ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t< _E > * = nullptr>
  bool run ()

Static Public Member Functions

- template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
  static Residu cardinality ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
  static Residu cardinality ()

Protected Attributes

- Field F
- size_t _mm
- size_t _nn

12.338.1 Member Typedef Documentation

12.338.1.1 Field

```
template<typename Elt>
using Field = Modular<Elt>
```

12.338.1.2 Elt_ptr

```
template<typename Elt>
using Elt_ptr = typename Field::Element_ptr
```

12.338.1.3 Residu

```
template<typename Elt>
using Residu = typename Field::Residu_t
```

12.338.1.4 enable_if_t

```
template<typename Elt>
template<bool B, class T = void>
using enable_if_t = typename std::enable_if<B, T>::type
```

12.338.1.5 is_same_element

```
template<typename Elt>
template<typename Simd>
using is_same_element = typename Simd::template is_same_element<Field>
```

12.338.1.6 enable_if_no_simd_t

```
template<typename Elt>
template<typename E>
using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
```

12.338.1.7 enable_if_simd128_t

```
template<typename Elt>
template<typename E>
using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
```

12.338.1.8 enable_if_simd256_t

```
template<typename Elt>
template<typename E>
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

12.338.1.9 enable_if_simd512_t

```
template<typename Elt>
template<typename E>
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

12.338.2 Constructor & Destructor Documentation

12.338.2.1 Test()

```
template<typename Elt>
Test (
            size_t mm,
            size_t nn)  [inline]
```

12.338.3 Member Function Documentation

12.338.3.1 cardinality() [1/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.338.3.2 cardinality() [2/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.338.3.3 test_ftranspose()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
bool test_ftranspose (
            size_t m,
            size_t n,
            Elt_ptr A,
            size_t lda,
            Elt_ptr B,
            size_t ldb)  [inline]
```

12.338.3.4 doTests()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
bool doTests ()  [inline]
```

12.338.3.5 run()

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t<
_E > * = nullptr>
bool run ()  [inline]
```

12.338.4 Field Documentation

12.338.4.1 F

```
template<typename Elt>
Field F  [protected]
```

12.338.4.2 _mm

```
template<typename Elt>
size_t _mm  [protected]
```

12.338.4.3 _nn

```
template<typename Elt>
size_t _nn  [protected]
```

The documentation for this class was generated from the following file:

- test-storage-transpose.C

12.339 TestOneMethod< Simd > Class Template Reference

Public Types

- using Element = typename Simd::scalar_t
- using vect_t = typename Simd::vect_t
- using vectElt = vector<Element>
- template<bool B, typename T = void>
  using enable_if_t = typename enable_if<B, T>::type

Public Member Functions

- template<typename... AScal, typename RScal, typename... ASimd, typename RSimd, enable_if_t< sizeof...(AScal)==sizeof...(ASimd)> * = nullptr, enable_if_t< count_nonconst_lvalue_reference< AScal... >::n==count_nonconst_lvalue_reference< ASimd... >::n > * = nullptr, enable_if_t< is_all_same< AScal... >::value > * = nullptr, enable_if_t< is_all_same< vect_t, ASimd... >::value > * = nullptr>
  TestOneMethod (function< RSimd(ASimd...)> fsimd, function< RScal(AScal...)> fscal, function< void(vector< vectElt > &)> genInputs, string fname)
- template<typename Ret, typename... AScal>
  enable_if_t< is_all_same< Element, AScal... >::value &&std::is_convertible< Ret, Element >::value, void > evaluate_scalar_method (function< Ret(AScal...)> fscal)
- template<typename... AScal>
  enable_if_t< is_all_same< vectElt, AScal... >::value, void > evaluate_scalar_method (function< vectElt(AScal...)> fscal)
- template<typename... AScal>
  enable_if_t< is_all_same< vectElt, AScal... >::value, void > evaluate_scalar_method (function< void(AScal...)> fscal)
- template<typename Ret, typename... ASimd>
  enable_if_t< is_all_same< vect_t, ASimd... >::value &&std::is_convertible< Ret, vect_t >::value, void > evaluate_simd_method (function< Ret(ASimd...)> fsimd, array< vect_t, sizeof...(ASimd)> &simd_in)
- template<typename... ASimd>
  enable_if_t< is_all_same< vect_t, ASimd... >::value, void > evaluate_simd_method (function< void(ASimd...)> fsimd, array< vect_t, sizeof...(ASimd)> &simd_in)
- bool getStatus () const
- string getTestName () const
- bool writeResultLine () const
- void writeDebugData () const

Static Public Attributes

- static constexpr size_t vect_size = Simd::vect_size

Protected Attributes

- size_t nb_lref
- string name
- vector< vectElt > inputs
- vector< vectElt > outputs_simd
- vector< vectElt > outputs_scalar

12.339.1 Member Typedef Documentation

12.339.1.1 Element

```
template<typename Simd>
using Element = typename Simd::scalar_t
```

12.339.1.2 vect_t

```
template<typename Simd>
using vect_t = typename Simd::vect_t
```

12.339.1.3 vectElt

```
template<typename Simd>
using vectElt = vector<Element>
```

12.339.1.4 enable_if_t

```
template<typename Simd>
template<bool B, typename T = void>
using enable_if_t = typename enable_if<B, T>::type
```

12.339.2 Constructor & Destructor Documentation

12.339.2.1 TestOneMethod()

```
template<typename Simd>
template<typename...  AScal, typename RScal, typename...  ASimd, typename RSimd, enable_if_t<
sizeof...(AScal)==sizeof...(ASimd)> * = nullptr, enable_if_t< count_nonconst_lvalue_reference<
AScal...  >::n==count_nonconst_lvalue_reference< ASimd...  >::n > * = nullptr, enable_if_t<
is_all_same< AScal...  >::value > * = nullptr, enable_if_t< is_all_same< vect_t, ASimd...
>::value > * = nullptr>
TestOneMethod (
             function< RSimd(ASimd...)> fsimd,
             function< RScal(AScal...)> fscal,
             function< void(vector< vectElt > &)> genInputs,
             string fname)  [inline]
```

12.339.3 Member Function Documentation

12.339.3.1 evaluate_scalar_method() [1/3]

```
template<typename Simd>
template<typename Ret, typename...  AScal>
enable_if_t< is_all_same< Element, AScal...  >::value &&std::is_convertible< Ret, Element >↩
::value, void > evaluate_scalar_method (
             function< Ret(AScal...)> fscal)  [inline]
```

12.339.3.2 evaluate_scalar_method() [2/3]

```
template<typename Simd>
template<typename...  AScal>
enable_if_t< is_all_same< vectElt, AScal...  >::value, void > evaluate_scalar_method (
             function< vectElt(AScal...)> fscal)  [inline]
```

12.339.3.3 evaluate_scalar_method() [3/3]

```
template<typename Simd>
template<typename...  AScal>
```

```
enable_if_t< is_all_same< vectElt, AScal...  >::value, void > evaluate_scalar_method (
            function< void(AScal...)> fscal)  [inline]
```

12.339.3.4 evaluate_simd_method() [1/2]

```
template<typename Simd>
template<typename Ret, typename...  ASimd>
enable_if_t< is_all_same< vect_t, ASimd...  >::value &&std::is_convertible< Ret, vect_t >↩
::value, void > evaluate_simd_method (
            function< Ret(ASimd...)> fsimd,
            array< vect_t, sizeof...(ASimd)> & simd_in)  [inline]
```

12.339.3.5 evaluate_simd_method() [2/2]

```
template<typename Simd>
template<typename...  ASimd>
enable_if_t< is_all_same< vect_t, ASimd...  >::value, void > evaluate_simd_method (
            function< void(ASimd...)> fsimd,
            array< vect_t, sizeof...(ASimd)> & simd_in)  [inline]
```

12.339.3.6 getStatus()

```
template<typename Simd>
bool getStatus () const  [inline]
```

12.339.3.7 getTestName()

```
template<typename Simd>
string getTestName () const  [inline]
```

12.339.3.8 writeResultLine()

```
template<typename Simd>
bool writeResultLine () const  [inline]
```

12.339.3.9 writeDebugData()

```
template<typename Simd>
void writeDebugData () const  [inline]
```

12.339.4 Field Documentation

12.339.4.1 vect_size

```
template<typename Simd>
size_t vect_size = Simd::vect_size  [static], [constexpr]
```

12.339.4.2 nb_lref

```
template<typename Simd>
size_t nb_lref  [protected]
```

12.339.4.3 name

```
template<typename Simd>
string name  [protected]
```

12.339.4.4 inputs

```
template<typename Simd>
vector<vectElt> inputs  [protected]
```

12.339.4.5 outputs_simd

```
template<typename Simd>
vector<vectElt> outputs_simd  [protected]
```

12.339.4.6 outputs_scalar

```
template<typename Simd>
vector<vectElt> outputs_scalar  [protected]
```

The documentation for this class was generated from the following file:

- test-simd.C

12.340 tfn_minus Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.340.1 Member Function Documentation

12.340.1.1 operator()()

```
template<typename...  Args>
auto operator() (
            Args &&...  args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.341 tfn_minus_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.341.1 Member Function Documentation

12.341.1.1 operator()()

```
template<typename...  Args>
auto operator() (
            Args &&...  args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.342 tfn_mul Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.342.1 Member Function Documentation

12.342.1.1 operator()()

```
template<typename... Args>
auto operator() (
            Args &&... args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.343 tfn_mul_eq Struct Reference

`#include <sparse_matrix_traits.h>`

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.343.1 Member Function Documentation

12.343.1.1 operator()()

```
template<typename... Args>
auto operator() (
            Args &&... args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.344 tfn_plus Struct Reference

`#include <sparse_matrix_traits.h>`

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.344.1 Member Function Documentation

12.344.1.1 operator()()

```
template<typename... Args>
auto operator() (
            Args &&... args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.345 tfn_plus_eq Struct Reference

`#include <sparse_matrix_traits.h>`

Public Member Functions

- template<typename... Args>
  auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))

12.345.1 Member Function Documentation

12.345.1.1 operator()()

```
template<typename...  Args>
auto operator() (
            Args &&...  args) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.346 Threads Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.347 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.348 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.349 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.350 TRSMHelper< RecIterTrait, ParSeqTrait > Struct Template Reference

TRSM Helper.

Public Member Functions

- template<class Cut, class Param>
  TRSMHelper (ParSeqHelper::Parallel< Cut, Param > _PS)
- TRSMHelper (ParSeqHelper::Sequential _PS)
- template<typename RIT, typename PST>
  TRSMHelper (TRSMHelper< RIT, PST > &_TH)

- template<class Dom, class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value> FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n, ParSeqTrait p) const
- template<class Dom, class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value> FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n) const

Data Fields

- ParSeqTrait parseq

12.350.1 Detailed Description

template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< RecIterTrait, ParSeqTrait >

TRSM Helper.

12.350.2 Constructor & Destructor Documentation

12.350.2.1 TRSMHelper() [1/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
template<class Cut, class Param>
TRSMHelper (
            ParSeqHelper::Parallel< Cut, Param > _PS)  [inline]
```

12.350.2.2 TRSMHelper() [2/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
TRSMHelper (
            ParSeqHelper::Sequential _PS)  [inline]
```

12.350.2.3 TRSMHelper() [3/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
template<typename RIT, typename PST>
TRSMHelper (
            TRSMHelper< RIT, PST > & _TH)  [inline]
```

12.350.3 Member Function Documentation

12.350.3.1 pMMH() [1/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
template<class Dom, class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS↩
::ModeTraits<Dom>::value>
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
            Dom & D,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait p) const  [inline]
```

12.350.3.2 pMMH() [2/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
template<class Dom, class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS↩
::ModeTraits<Dom>::value>
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
             Dom & D,
             size_t m,
             size_t k,
             size_t n) const  [inline]
```

12.350.4 Field Documentation

12.350.4.1 parseq

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↩
Helper::Sequential>
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.351 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.352 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.353 UnparametricTag Struct Reference

If the field uses a representation with infix operators.
`#include <field-traits.h>`

12.353.1 Detailed Description

If the field uses a representation with infix operators.
The documentation for this struct was generated from the following file:

- field-traits.h

12.354 vector< T > Class Template Reference

STL class.

Data Structures

- class const_iterator

  STL iterator class.
- class const_reverse_iterator

  STL iterator class.

- class iterator

  STL iterator class.
- class reverse_iterator

  STL iterator class.

Data Fields

- T elements

  STL member.

12.354.1 Detailed Description

STL class.

12.354.2 Field Documentation

12.354.2.1 elements

```
T elements
```

STL member.

The documentation for this class was generated from the following files:

12.355 width< T > Struct Template Reference

Static Public Attributes

- static constexpr size_t value = 2+2*sizeof(T)

12.355.1 Field Documentation

12.355.1.1 value

```
template<typename T>
size_t value = 2+2*sizeof(T)  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.356 width< double > Struct Reference

Static Public Attributes

- static constexpr size_t value = 24
- static constexpr size_t value

12.356.1 Field Documentation

12.356.1.1 value [1/2]

```
size_t value = 24  [static], [constexpr]
```

12.356.1.2 value [2/2]

```
size_t value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.357 width< float > Struct Reference

Static Public Attributes

- static constexpr size_t value = 16
- static constexpr size_t value

12.357.1 Field Documentation

12.357.1.1 value [1/2]

```
size_t value = 16  [static], [constexpr]
```

12.357.1.2 value [2/2]

```
size_t value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.358 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.359 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

Chapter 13

File Documentation

13.1 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

Typedefs

- typedef Givaro::OMPTimer TTimer

Functions

- int main (int argc, char **argv)

13.1.1 Typedef Documentation

13.1.1.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.1.2 Function Documentation

13.1.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.2 autotune/charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define CUBE(x)
- #define GFOPS(m, n, r, t)

Typedefs

- typedef Givaro::Timer TTimer

Functions

- int main ()

13.2.1 Macro Definition Documentation

13.2.1.1 CUBE

```
#define CUBE(
                x)
```

Value:

```
((x)*(x)*(x))
```

13.2.1.2 GFOPS

```
#define GFOPS(
                m,
                n,
                r,
                t)
```

Value:

```
(2.7*CUBE(double(n)/1000.0))/t
```

13.2.2 Typedef Documentation

13.2.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.2.3 Function Documentation

13.2.3.1 main()

```
int main (
              void )
```

13.3 examples/charpoly.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (int argc, char **argv)

  This example computes the characteristic polynomial of a matrix over a defined finite field.

13.3.1 Function Documentation

13.3.1.1 main()

```
int main (
              int argc,
              char ** argv)
```

This example computes the characteristic polynomial of a matrix over a defined finite field.
Outputs the characteristic polynomial.

13.4 fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

Macros

- #define CUBE(x)
- #define GFOPS(n, t)

Functions

- int main ()

13.4.1 Macro Definition Documentation

13.4.1.1 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.4.1.2 GFOPS

```
#define GFOPS(
              n,
              t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.4.2 Function Documentation

13.4.2.1 main()

```
int main (
            void )
```

13.5 fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

Macros

- #define CUBE(x)
- #define GFOPS(n, t)

Functions

- int main ()

13.5.1 Macro Definition Documentation

13.5.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.5.1.2 GFOPS

```
#define GFOPS(
              n,
              t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.5.2 Function Documentation

13.5.2.1 main()

```
int main (
            void )
```

13.6 ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define CUBE(x)
- #define GFOPS(n, t)

Functions

- int main ()

13.6.1 Macro Definition Documentation

13.6.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.6.1.2 GFOPS

```
#define GFOPS(
              n,
              t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.6.2 Function Documentation

13.6.2.1 main()

```
int main (
            void )
```

13.7 autotune/pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define CUBE(x)
- #define GFOPS(m, n, r, t)

Functions

- int main ()

13.7.1 Macro Definition Documentation

13.7.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.7.1.2 GFOPS

```
#define GFOPS(
              m,
              n,
              r,
              t)
```

Value:

```
(2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0  -
      double(r)/1000.0*double(r)/1000.0*(m+n)/1000)/t
```

13.7.2 Function Documentation

13.7.2.1 main()

```
int main (
            void )
```

13.8 examples/pluq.C File Reference

```
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Functions

- int main (int argc, char **argv)

13.8.1 Function Documentation

13.8.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.9 winograd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <ctime>
```

Macros

- #define DOUBLE_TO_FLOAT_CROSSOVER 0
- #define GFOPS(n, t)

Functions

- template<class Field>
  bool balanced (const Field &)
- template<class T>
  bool balanced (const Givaro::ModularBalanced< T > &)
- int main ()

13.9.1 Macro Definition Documentation

13.9.1.1 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

13.9.1.2 GFOPS

```
#define GFOPS(
              n,
              t)
```

Value:

```
(2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

13.9.2 Function Documentation

13.9.2.1 balanced() [1/2]

```
template<class Field>
bool balanced (
             const Field & )
```

13.9.2.2 balanced() [2/2]

```
template<class T>
bool balanced (
             const Givaro::ModularBalanced< T > & )
```

13.9.2.3 main()

```
int main (
             void )
```

13.10 benchmark-charpoly-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/Matio.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_FORCE_SEQ

Functions

- int main (int argc, char **argv)

13.10.1 Macro Definition Documentation

13.10.1.1 __FFLASFFPACK_FORCE_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

13.10.2 Function Documentation

13.10.2.1 main()

```
int main (
             int argc,
             char ** argv)
```

13.11 benchmark-charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- template<class Field>
  void run_with_field (int q, uint64_t bits, size_t n, size_t d, size_t iter, std::string file, int variant, uint64_t seed)
- int main (int argc, char **argv)

13.11.1 Macro Definition Documentation

13.11.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.11.2 Function Documentation

13.11.2.1 run_with_field()

```
template<class Field>
void run_with_field (
            int q,
            uint64_t bits,
            size_t n,
            size_t d,
            size_t iter,
            std::string file,
            int variant,
            uint64_t seed)
```

13.11.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.12 benchmark-checkers.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include <fstream>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1
- #define _NR_TESTS 5
- #define _MAX_SIZE_MATRICES 1000
- #define CUBE(x)

Functions

- int main (int argc, char **argv)

13.12.1 Macro Definition Documentation

13.12.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.12.1.2 _NR_TESTS

```
#define _NR_TESTS 5
```

13.12.1.3 _MAX_SIZE_MATRICES

```
#define _MAX_SIZE_MATRICES 1000
```

13.12.1.4 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.12.2 Function Documentation

13.12.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.13 benchmark-dgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CBLAS_GEMM cblas_dgemm

Typedefs

- typedef FFLAS::Timer TTimer
- typedef double Floats

Functions

- int main (int argc, char **argv)

13.13.1 Macro Definition Documentation

13.13.1.1 CBLAS_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

13.13.2 Typedef Documentation

13.13.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.13.2.2 Floats

```
typedef double Floats
```

13.13.3 Function Documentation

13.13.3.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.14 benchmark-dgetrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_HAVE_DGETRF 1

Functions

- int main (int argc, char **argv)

13.14.1 Macro Definition Documentation

13.14.1.1 __FFLASFFPACK_HAVE_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

13.14.2 Function Documentation

13.14.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.15 benchmark-dgetri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Functions

- int main (int argc, char **argv)

13.15.1 Function Documentation

13.15.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.16 benchmark-dsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define EFFGFF(n, t, i)

Functions

- int main (int argc, char **argv)

13.16.1 Macro Definition Documentation

13.16.1.1 EFFGFF

```
#define EFFGFF(
              n,
              t,
              i)
```

Value:

```
( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i) / 3.)
```

13.16.2 Function Documentation

13.16.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.17 benchmark-dtrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Functions

- int main (int argc, char **argv)

13.17.1 Function Documentation

13.17.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.18 benchmark-dtrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_HAVE_DTRTRI 1

Functions

- int main (int argc, char **argv)

13.18.1 Macro Definition Documentation

13.18.1.1 __FFLASFFPACK_HAVE_DTRTRI

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

13.18.2 Function Documentation

13.18.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.19 benchmark-fadd-lvl2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.19.1 Macro Definition Documentation

13.19.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.19.2 Function Documentation

13.19.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.20 benchmark-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givrational.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- template<class Field>
  Field::Element run_with_field (int q, size_t iter, size_t N, const uint64_t BS, const size_t p, const size_t threads, uint64_t seed)
- int main (int argc, char **argv)

13.20.1 Macro Definition Documentation

13.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.20.2 Function Documentation

13.20.2.1 run_with_field()

```
template<class Field>
Field::Element run_with_field (
            int q,
            size_t iter,
            size_t N,
            const uint64_t BS,
            const size_t p,
            const size_t threads,
            uint64_t seed)
```

13.20.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.21 benchmark-fgemm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
- #define MG_DEFAULT MG_ACTIVE
- #define STD_RECINT_SIZE 8

Functions

- template<typename Ints>
  int tmain ()
- int main (int argc, char **argv)

13.21.1 Macro Definition Documentation

13.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.21.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

13.21.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

13.21.2 Function Documentation

13.21.2.1 tmain()

```
template<typename Ints>
int tmain ()
```

13.21.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.22 benchmark-fgemm-rns.C File Reference

```
#include "fflas-ffpack/fflas/fflas.h"
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Typedefs

- typedef FFPACK::rns_double RNS
- typedef FFPACK::RNSInteger< RNS > Field
- typedef Field::Element_ptr Element_ptr
- typedef Field::ConstElement_ptr ConstElement_ptr
- typedef StrategyParameter::Threads THREADS
- typedef StrategyParameter::Grain GRAIN

- typedef StrategyParameter::TwoD TWOD
- typedef StrategyParameter::TwoDAdaptive TWODA
- typedef StrategyParameter::ThreeD THREED
- typedef StrategyParameter::ThreeDAdaptive THREEDA
- typedef StrategyParameter::ThreeDInPlace THREEDIP
- typedef ParSeqHelper::Sequential PSeq

Functions

- int main (int argc, char ∗argv[ ])

13.22.1 Macro Definition Documentation

13.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.22.2 Typedef Documentation

13.22.2.1 RNS

```
typedef FFPACK::rns_double RNS
```

13.22.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

13.22.2.3 Element_ptr

```
typedef Field::Element_ptr Element_ptr
```

13.22.2.4 ConstElement_ptr

```
typedef Field::ConstElement_ptr ConstElement_ptr
```

13.22.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

13.22.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

13.22.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

13.22.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

13.22.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

13.22.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

13.22.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

13.22.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

13.22.3 Function Documentation

13.22.3.1 main()

```
int main (
            int argc,
            char * argv[])
```

13.23 benchmark-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CLASSIC_HYBRID

Functions

- int main (int argc, char **argv)

13.23.1 Macro Definition Documentation

13.23.1.1 CLASSIC_HYBRID

```
#define CLASSIC_HYBRID
```

13.23.2 Function Documentation

13.23.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.24 benchmark-fgemv-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
- #define MG_DEFAULT MG_ACTIVE
- #define STD_RECINT_SIZE 8

Functions

- template<typename T>
  std::ostream & write_matrix (std::ostream &out, Givaro::Integer p, size_t m, size_t n, T *C, size_t ldc)
- template<typename Ints>
  int tmain ()
- int main (int argc, char **argv)

13.24.1 Macro Definition Documentation

13.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.24.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

13.24.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

13.24.2 Function Documentation

13.24.2.1 write_matrix()

```
template<typename T>
std::ostream & write_matrix (
             std::ostream & out,
             Givaro::Integer p,
             size_t m,
             size_t n,
             T * C,
             size_t ldc)
```

13.24.2.2 tmain()

```
template<typename Ints>
int tmain ()
```

13.24.2.3 main()

```
int main (
             int argc,
             char ** argv)
```

13.25 benchmark-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
```

Data Structures

- struct need_field_characteristic< Field >
- struct need_field_characteristic< Givaro::Modular< Field > >
- struct need_field_characteristic< Givaro::ModularBalanced< Field > >
- struct compatible_data_type< Field >
- struct compatible_data_type< Givaro::ZRing< float > >
- struct compatible_data_type< Givaro::ZRing< double > >

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- template<class Field, class RandIter, class Matrix, class Vector>
  void fill_value (Field &F, RandIter &Rand, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK)
- template<class Field, class Matrix, class Vector>
  void genData (Field &F, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK, uint64_t bitsize, uint64_t seed)
- template<class Field, class Matrix, class Vector>
  bool check_result (Field &F, size_t m, size_t lda, Matrix &A, Vector &X, size_t incX, Vector &Y, size_t incY)
- template<class Field, class Matrix, class Vector>
  bool benchmark_with_timer (Field &F, int p, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, size_t iters, int t, double &time, size_t GrainSize)
- template<class Field, class arg>
  void benchmark_disp (Field &F, bool pass, double &time, size_t iters, int p, size_t m, size_t k, arg &as)
- template<class Field, class arg>
  void benchmark_in_Field (Field &F, int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)
- template<class Field, class arg>
  void benchmark_with_field (int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)
- template<class Field, class arg>
  void benchmark_with_field (const Givaro::Integer &q, int p, size_t m, size_t k, int NBK, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)
- int main (int argc, char **argv)

13.25.1 Macro Definition Documentation

13.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.25.2 Function Documentation

13.25.2.1 fill_value()

```
template<class Field, class RandIter, class Matrix, class Vector>
void fill_value (
            Field & F,
            RandIter & Rand,
            Matrix & A,
            Vector & X,
            Vector & Y,
            size_t m,
            size_t k,
            size_t incX,
            size_t incY,
            size_t lda,
            int NBK)
```

13.25.2.2 genData()

```
template<class Field, class Matrix, class Vector>
void genData (
            Field & F,
            Matrix & A,
            Vector & X,
            Vector & Y,
            size_t m,
            size_t k,
            size_t incX,
            size_t incY,
            size_t lda,
            int NBK,
            uint64_t bitsize,
            uint64_t seed)
```

13.25.2.3 check_result()

```
template<class Field, class Matrix, class Vector>
bool check_result (
            Field & F,
            size_t m,
            size_t lda,
            Matrix & A,
            Vector & X,
            size_t incX,
            Vector & Y,
            size_t incY)
```

13.25.2.4 benchmark_with_timer()

```
template<class Field, class Matrix, class Vector>
bool benchmark_with_timer (
            Field & F,
            int p,
            Matrix & A,
            Vector & X,
            Vector & Y,
            size_t m,
            size_t k,
            size_t incX,
```

```
            size_t incY,
            size_t lda,
            size_t iters,
            int t,
            double & time,
            size_t GrainSize)
```

13.25.2.5 benchmark_disp()

```
template<class Field, class arg>
void benchmark_disp (
            Field & F,
            bool pass,
            double & time,
            size_t iters,
            int p,
            size_t m,
            size_t k,
            arg & as)
```

13.25.2.6 benchmark_in_Field()

```
template<class Field, class arg>
void benchmark_in_Field (
            Field & F,
            int p,
            size_t m,
            size_t k,
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.7 benchmark_with_field() [1/2]

```
template<class Field, class arg>
void benchmark_with_field (
            int p,
            size_t m,
            size_t k,
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.8 benchmark_with_field() [2/2]

```
template<class Field, class arg>
void benchmark_with_field (
            const Givaro::Integer & q,
            int p,
            size_t m,
            size_t k,
```

```
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.9 main()

```
int main (
            int argc,
            char ** argv)
```

13.26 benchmark-fgesv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.26.1 Macro Definition Documentation

13.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.26.2 Function Documentation

13.26.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.27 benchmark-fsyr2k.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Functions

- int main (int argc, char **argv)

13.27.1 Function Documentation

13.27.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.28 benchmark-fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.28.1 Macro Definition Documentation

13.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.28.2 Function Documentation

13.28.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.29 benchmark-fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFPACK_FSYTRF_BC_CROUT
- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

- #define CUBE(x)

Functions

- int main (int argc, char **argv)

13.29.1 Macro Definition Documentation

13.29.1.1 __FFPACK_FSYTRF_BC_CROUT

```
#define __FFPACK_FSYTRF_BC_CROUT
```

13.29.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.29.1.3 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.29.2 Function Documentation

13.29.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.30 benchmark-ftrsm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.30.1 Macro Definition Documentation

13.30.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.30.2 Function Documentation

13.30.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.31 benchmark-ftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.31.1 Macro Definition Documentation

13.31.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.31.2 Function Documentation

13.31.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.32 benchmark-ftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- int main (int argc, char **argv)

13.32.1 Macro Definition Documentation

13.32.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.32.2 Function Documentation

13.32.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.33 benchmark-ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
- #define CUBE(x)

Functions

- int main (int argc, char **argv)

13.33.1 Macro Definition Documentation

13.33.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.33.1.2 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.33.2 Function Documentation

13.33.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.34 benchmark-inverse.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
```

```
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CUBE(x)

Functions

- int main (int argc, char **argv)

13.34.1 Macro Definition Documentation

13.34.1.1 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.34.2 Function Documentation

13.34.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.35 benchmark-lqup-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

Functions

- int main (int argc, char **argv)

13.35.1 Function Documentation

13.35.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.36 benchmark-lqup.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CUBE(x)

Functions

- int main (int argc, char **argv)

13.36.1 Macro Definition Documentation

13.36.1.1 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.36.2 Function Documentation

13.36.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.37 benchmark-pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include <givaro/givranditer.h>
#include <iostream>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
- #define CUBE(x)

Typedefs

- typedef Givaro::ModularBalanced< double > Field

Functions

- void verification_PLUQ (const Field &F, typename Field::Element ∗B, typename Field::Element ∗A, size_t ∗P, size_t ∗Q, size_t m, size_t n, size_t R)
- void Rec_Initialize (Field &F, Field::Element ∗C, size_t m, size_t n, size_t ldc)
- int main (int argc, char ∗∗argv)

13.37.1 Macro Definition Documentation

13.37.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.37.1.2 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.37.2 Typedef Documentation

13.37.2.1 Field

```
typedef Givaro::ModularBalanced<double> Field
```

13.37.3 Function Documentation

13.37.3.1 verification_PLUQ()

```
void verification_PLUQ (
             const Field & F,
             typename Field::Element * B,
             typename Field::Element * A,
             size_t * P,
             size_t * Q,
             size_t m,
             size_t n,
             size_t R)
```

13.37.3.2 Rec_Initialize()

```
void Rec_Initialize (
             Field & F,
             Field::Element * C,
             size_t m,
             size_t n,
             size_t ldc)
```

13.37.3.3 main()

```
int main (
             int argc,
             char ** argv)
```

13.38 benchmark-quasisep.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
```

```
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1

Functions

- template<class Field>
  void run_with_field (int q, size_t n, size_t m, size_t t, size_t r, size_t iter, uint64_t seed)
- int main (int argc, char **argv)

13.38.1 Macro Definition Documentation

13.38.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.38.2 Function Documentation

13.38.2.1 run_with_field()

```
template<class Field>
void run_with_field (
            int q,
            size_t n,
            size_t m,
            size_t t,
            size_t r,
            size_t iter,
            uint64_t seed)
```

13.38.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.39 benchmark-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

Data Structures

- class Bench< Elt >
- class ModularBalanced< T >

Functions

- int main (int argc, char **argv)

13.39.1 Function Documentation

13.39.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.40 benchmark-wino.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CUBE(x)

Functions

- template<class Field>
  void launch_wino (const Field &F, const size_t &n, const size_t &NB, const size_t &wino, const bool &asmax, const size_t &seed, const bool compare)
- int main (int argc, char **argv)

13.40.1 Macro Definition Documentation

13.40.1.1 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.40.2 Function Documentation

13.40.2.1 launch_wino()

```
template<class Field>
void launch_wino (
            const Field & F,
            const size_t & n,
            const size_t & NB,
            const size_t & wino,
            const bool & asmax,
```

```
            const size_t & seed,
            const bool compare)
```

13.40.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.41 mainpage.doxy File Reference

13.42 det.C File Reference

```
#include <givaro/modular.h>
#include <iostream>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Functions

- int main (int argc, char **argv)

  This example computes the determinant of a matrix over a defined finite field.

13.42.1 Function Documentation

13.42.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example computes the determinant of a matrix over a defined finite field.
Outputs the determinant.

13.43 matmul.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (int argc, char **argv)

  This example computes the matrix multiplication over a defined finite field.

13.43.1 Function Documentation

13.43.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example computes the matrix multiplication over a defined finite field.
Outputs the product of the matrix given as input.

13.44 rank.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (int argc, char **argv)

  This example computes the rank of a matrix over a defined finite field.

13.44.1 Function Documentation

13.44.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example computes the rank of a matrix over a defined finite field.
Outputs the rank.

13.45 solve.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (int argc, char **argv)

  This example solve the quare system defined by the input over a defined finite field.

13.45.1 Function Documentation

13.45.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example solve the quare system defined by the input over a defined finite field.

13.46 checker_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class CheckerImplem_charpoly< Field, Polynomial >
- class CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_charpoly_INL

13.46.1 Macro Definition Documentation

13.46.1.1 __FFLASFFPACK_checker_charpoly_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

13.47 checker_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class CheckerImplem_Det< Field >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_det_INL

13.47.1 Macro Definition Documentation

13.47.1.1 __FFLASFFPACK_checker_det_INL

```
#define __FFLASFFPACK_checker_det_INL
```

13.48 checker_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Data Structures

- struct Checker_Empty< Field >

Namespaces

- namespace FFLAS

13.49 checker_fgemm.inl File Reference

Data Structures

- class CheckerImplem_fgemm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_fgemm_INL

13.49.1 Macro Definition Documentation

13.49.1.1 __FFLASFFPACK_checker_fgemm_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

13.50 checker_ftrsm.inl File Reference

Data Structures

- class CheckerImplem_ftrsm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_ftrsm_INL

13.50.1 Macro Definition Documentation

13.50.1.1 __FFLASFFPACK_checker_ftrsm_INL

```
#define __FFLASFFPACK_checker_ftrsm_INL
```

13.51 checker_invert.inl File Reference

Data Structures

- class CheckerImplem_invert< Field >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_invert_INL

13.51.1 Macro Definition Documentation

13.51.1.1 __FFLASFFPACK_checker_invert_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

13.52 checker_pluq.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Data Structures

- class CheckerImplem_PLUQ< Field >

Namespaces

- namespace FFPACK

  Finite Field PACK Set of elimination based routines for dense linear algebra.

Macros

- #define __FFLASFFPACK_checker_pluq_INL

13.52.1 Macro Definition Documentation

13.52.1.1 __FFLASFFPACK_checker_pluq_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

13.53 checkers.doxy File Reference

13.54 checkers_fflas.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
```

Data Structures

- class FailureFgemmCheck
- class FailureTrsmCheck

Namespaces

- namespace FFLAS

Typedefs

- template<class Field>
  using Checker_fgemm = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_ftrsm = FFLAS::Checker_Empty<Field>

13.55 checkers_fflas.inl File Reference

```
#include "checker_fgemm.inl"
#include "checker_ftrsm.inl"
```

Namespaces

- namespace FFLAS

Macros

- #define FFLASFFPACK_checkers_fflas_inl_H

Typedefs

- template<class Field>
  using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
- template<class Field>
  using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>

13.55.1 Macro Definition Documentation

13.55.1.1 FFLASFFPACK_checkers_fflas_inl_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

13.56 checkers_ffpack.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class FailurePLUQCheck
- class FailureDetCheck
- class FailureInvertCheck
- class FailureCharpolyCheck

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Typedefs

- template<class Field>
  using Checker_PLUQ = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_Det = FFLAS::Checker_Empty<Field>
- template<class Field>
  using Checker_invert = FFLAS::Checker_Empty<Field>
- template<class Field, class Polynomial>
  using Checker_charpoly = FFLAS::Checker_Empty<Field>

13.57 checkers_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define FFLASFFPACK_checkers_ffpack_inl_H

Typedefs

- template<class Field>
  using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
- template<class Field>
  using ForceCheck_Det = CheckerImplem_Det<Field>
- template<class Field>
  using ForceCheck_invert = CheckerImplem_invert<Field>
- template<class Field, class Polynomial>
  using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>

13.57.1 Macro Definition Documentation

13.57.1.1 FFLASFFPACK_checkers_ffpack_inl_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

13.58 config-blas.h File Reference

Macros

- #define CBLAS_INT int
- #define CBLAS_ENUM_DEFINED_H
- #define CBLAS_EXTERNALS
- #define blas_enum enum

Enumerations

- enum CBLAS_ORDER { CblasRowMajor =101 , CblasColMajor =102 }
- enum CBLAS_TRANSPOSE { CblasNoTrans =111 , CblasTrans =112 , CblasConjTrans =113 , AtlasConj =114 }
- enum CBLAS_UPLO { CblasUpper =121 , CblasLower =122 }
- enum CBLAS_DIAG { CblasNonUnit =131 , CblasUnit =132 }
- enum CBLAS_SIDE { CblasLeft =141 , CblasRight =142 }

Functions

- void daxpy_ (const int *, const double *, const double *, const int *, double *, const int *)
- void saxpy_ (const int *, const float *, const float *, const int *, float *, const int *)
- double ddot_ (const int *, const double *, const int *, const double *, const int *)
- float sdot_ (const int *, const float *, const int *, const float *, const int *)
- double dasum_ (const int *, const double *, const int *)
- int idamax_ (const int *, const double *, const int *)
- double dnrm2_ (const int *, const double *, const int *)
- void dgemv_ (const char *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)
- void sgemv_ (const char *, const int *, const int *, const float *, const float *, const int *, const float *, const int *, const float *, float *, const int *)

- void dger_ (const int *, const int *, const double *, const double *, const int *, const double *, const int *, double *, const int *)
- void sger_ (const int *, const int *, const float *, const float *, const int *, const float *, const int *, float *, const int *)
- void dcopy_ (const int *, const double *, const int *, double *, const int *)
- void scopy_ (const int *, const float *, const int *, float *, const int *)
- void dscal_ (const int *, const double *, double *, const int *)
- void sscal_ (const int *, const float *, float *, const int *)
- void dtrsm_ (const char *, const char *, const char *, const char *, const int *, const int *, const double *, const double *, const int *, double *, const int *)
- void strsm_ (const char *, const char *, const char *, const char *, const int *, const int *, const float *, const float *, const int *, float *, const int *)
- void dtrmm_ (const char *, const char *, const char *, const char *, const int *, const int *, const double *, const double *, const int *, double *, const int *)
- void strmm_ (const char *, const char *, const char *, const char *, const int *, const int *, const float *, const float *, const int *, float *, const int *)
- void sgemm_ (const char *, const char *, const int *, const int *, const int *, const float *, const float *, const int *, const float *, const int *, const float *, float *, const int *)
- void dgemm_ (const char *, const char *, const int *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)
- void cblas_dsyrk (const enum CBLAS_ORDER Order, const enum CBLAS_UPLO Uplo, const enum CBLAS_TRANSPOSE Trans, const int N, const int K, const double alpha, const double *A, const int lda, const double beta, double *C, const int ldc)

13.58.1 Macro Definition Documentation

13.58.1.1 CBLAS_INT

```
#define CBLAS_INT int
```

13.58.1.2 CBLAS_ENUM_DEFINED_H

```
#define CBLAS_ENUM_DEFINED_H
```

13.58.1.3 CBLAS_EXTERNALS

```
#define CBLAS_EXTERNALS
```

13.58.1.4 blas_enum

```
#define blas_enum enum
```

13.58.2 Enumeration Type Documentation

13.58.2.1 CBLAS_ORDER

```
enum CBLAS_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

13.58.2.2 CBLAS_TRANSPOSE

```
enum CBLAS_TRANSPOSE
```

Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

13.58.2.3 CBLAS_UPLO

```
enum CBLAS_UPLO
```

Enumerator

CblasUpper	
CblasLower	

13.58.2.4 CBLAS_DIAG

```
enum CBLAS_DIAG
```

Enumerator

CblasNonUnit	
CblasUnit	

13.58.2.5 CBLAS_SIDE

```
enum CBLAS_SIDE
```

Enumerator

CblasLeft	
CblasRight	

13.58.3 Function Documentation

13.58.3.1 daxpy_()

```
void daxpy_ (
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.2 saxpy_()

```
void saxpy_ (
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.3 ddot_()

```
double ddot_ (
             const int * ,
             const double * ,
             const int * ,
             const double * ,
             const int * )
```

13.58.3.4 sdot_()

```
float sdot_ (
             const int * ,
             const float * ,
             const int * ,
             const float * ,
             const int * )
```

13.58.3.5 dasum_()

```
double dasum_ (
             const int * ,
             const double * ,
             const int * )
```

13.58.3.6 idamax_()

```
int idamax_ (
             const int * ,
             const double * ,
             const int * )
```

13.58.3.7 dnrm2_()

```
double dnrm2_ (
             const int * ,
             const double * ,
             const int * )
```

13.58.3.8 dgemv_()

```
void dgemv_ (
             const char * ,
             const int * ,
             const int * ,
             const double * ,
             const double * ,
             const int * ,
             const double * ,
             const int * ,
             const double * ,
             double * ,
             const int * )
```

13.58.3.9 sgemv_()

```
void sgemv_ (
             const char * ,
             const int * ,
             const int * ,
```

```
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            float * ,
            const int * )
```

13.58.3.10 dger_()

```
void dger_ (
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.11 sger_()

```
void sger_ (
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.12 dcopy_()

```
void dcopy_ (
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.13 scopy_()

```
void scopy_ (
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.14 dscal_()

```
void dscal_ (
            const int * ,
            const double * ,
```

```
            double * ,
            const int * )
```

13.58.3.15 sscal_()

```
void sscal_ (
            const int * ,
            const float * ,
            float * ,
            const int * )
```

13.58.3.16 dtrsm_()

```
void dtrsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.17 strsm_()

```
void strsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.18 dtrmm_()

```
void dtrmm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.19 strmm_()

```
void strmm_ (
```

```
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.20 sgemm_()

```
void sgemm_ (
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            float * ,
            const int * )
```

13.58.3.21 dgemm_()

```
void dgemm_ (
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            const double * ,
            const int * ,
            const double * ,
            double * ,
            const int * )
```

13.58.3.22 cblas_dsyrk()

```
void cblas_dsyrk (
            const enum CBLAS_ORDER Order,
            const enum CBLAS_UPLO Uplo,
            const enum CBLAS_TRANSPOSE Trans,
            const int N,
            const int K,
            const double alpha,
            const double * A,
            const int lda,
            const double beta,
```

```
            double * C,
            const int ldc)
```

13.59 config.h File Reference

Macros

- #define HAVE_BLAS 1
- #define HAVE_CBLAS 1
- #define HAVE_CXX11 1
- #define HAVE_DLFCN_H 1
- #define HAVE_FLOAT_H 1
- #define HAVE_INT128 1
- #define HAVE_INTTYPES_H 1
- #define HAVE_LAPACK 1
- #define HAVE_LIMITS_H 1
- #define HAVE_LITTLE_ENDIAN 1
- #define HAVE_PTHREAD_H 1
- #define HAVE_STDDEF_H 1
- #define HAVE_STDINT_H 1
- #define HAVE_STDIO_H 1
- #define HAVE_STDLIB_H 1
- #define HAVE_STRINGS_H 1
- #define HAVE_STRING_H 1
- #define HAVE_SYS_STAT_H 1
- #define HAVE_SYS_TIME_H 1
- #define HAVE_SYS_TYPES_H 1
- #define HAVE_UNISTD_H 1
- #define LT_OBJDIR ".libs/"
- #define OPENBLAS_NUM_THREADS 1
- #define PACKAGE "fflas-ffpack"
- #define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
- #define PACKAGE_NAME "FFLAS-FFPACK"
- #define PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
- #define PACKAGE_TARNAME "fflas-ffpack"
- #define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
- #define PACKAGE_VERSION "2.5.0"
- #define SIZEOF_CHAR 1
- #define SIZEOF_INT 4
- #define SIZEOF_LONG 8
- #define SIZEOF_LONG_LONG 8
- #define SIZEOF_SHORT 2
- #define SIZEOF___INT64_T 8
- #define STDC_HEADERS 1
- #define USE_OPENMP 1
- #define VERSION "2.5.0"

13.59.1 Macro Definition Documentation

13.59.1.1 HAVE_BLAS

```
#define HAVE_BLAS 1
```

13.59.1.2 HAVE_CBLAS

```
#define HAVE_CBLAS 1
```

13.59.1.3 HAVE_CXX11

```
#define HAVE_CXX11 1
```

13.59.1.4 HAVE_DLFCN_H

```
#define HAVE_DLFCN_H 1
```

13.59.1.5 HAVE_FLOAT_H

```
#define HAVE_FLOAT_H 1
```

13.59.1.6 HAVE_INT128

```
#define HAVE_INT128 1
```

13.59.1.7 HAVE_INTTYPES_H

```
#define HAVE_INTTYPES_H 1
```

13.59.1.8 HAVE_LAPACK

```
#define HAVE_LAPACK 1
```

13.59.1.9 HAVE_LIMITS_H

```
#define HAVE_LIMITS_H 1
```

13.59.1.10 HAVE_LITTLE_ENDIAN

```
#define HAVE_LITTLE_ENDIAN 1
```

13.59.1.11 HAVE_PTHREAD_H

```
#define HAVE_PTHREAD_H 1
```

13.59.1.12 HAVE_STDDEF_H

```
#define HAVE_STDDEF_H 1
```

13.59.1.13 HAVE_STDINT_H

```
#define HAVE_STDINT_H 1
```

13.59.1.14 HAVE_STDIO_H

```
#define HAVE_STDIO_H 1
```

13.59.1.15 HAVE_STDLIB_H

```
#define HAVE_STDLIB_H 1
```

13.59.1.16 HAVE_STRINGS_H

```
#define HAVE_STRINGS_H 1
```

13.59.1.17 HAVE_STRING_H

```
#define HAVE_STRING_H 1
```

13.59.1.18 HAVE_SYS_STAT_H

```
#define HAVE_SYS_STAT_H 1
```

13.59.1.19 HAVE_SYS_TIME_H

```
#define HAVE_SYS_TIME_H 1
```

13.59.1.20 HAVE_SYS_TYPES_H

```
#define HAVE_SYS_TYPES_H 1
```

13.59.1.21 HAVE_UNISTD_H

```
#define HAVE_UNISTD_H 1
```

13.59.1.22 LT_OBJDIR

```
#define LT_OBJDIR ".libs/"
```

13.59.1.23 OPENBLAS_NUM_THREADS

```
#define OPENBLAS_NUM_THREADS 1
```

13.59.1.24 PACKAGE

```
#define PACKAGE "fflas-ffpack"
```

13.59.1.25 PACKAGE_BUGREPORT

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

13.59.1.26 PACKAGE_NAME

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

13.59.1.27 PACKAGE_STRING

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

13.59.1.28 PACKAGE_TARNAME

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

13.59.1.29 PACKAGE_URL

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

13.59.1.30 PACKAGE_VERSION

```
#define PACKAGE_VERSION "2.5.0"
```

13.59.1.31 SIZEOF_CHAR

```
#define SIZEOF_CHAR 1
```

13.59.1.32 SIZEOF_INT

```
#define SIZEOF_INT 4
```

13.59.1.33 SIZEOF_LONG

```
#define SIZEOF_LONG 8
```

13.59.1.34 SIZEOF_LONG_LONG

```
#define SIZEOF_LONG_LONG 8
```

13.59.1.35 SIZEOF_SHORT

```
#define SIZEOF_SHORT 2
```

13.59.1.36 SIZEOF___INT64_T

```
#define SIZEOF___INT64_T 8
```

13.59.1.37 STDC_HEADERS

```
#define STDC_HEADERS 1
```

13.59.1.38 USE_OPENMP

```
#define USE_OPENMP 1
```

13.59.1.39 VERSION

```
#define VERSION "2.5.0"
```

13.60 fflas-ffpack/config.h File Reference

Macros

- #define __FFLASFFPACK_HAVE_BLAS 1
- #define __FFLASFFPACK_HAVE_CBLAS 1
- #define __FFLASFFPACK_HAVE_CXX11 1
- #define __FFLASFFPACK_HAVE_DLFCN_H 1
- #define __FFLASFFPACK_HAVE_FLOAT_H 1
- #define __FFLASFFPACK_HAVE_INT128 1
- #define __FFLASFFPACK_HAVE_INTTYPES_H 1
- #define __FFLASFFPACK_HAVE_LAPACK 1
- #define __FFLASFFPACK_HAVE_LIMITS_H 1
- #define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
- #define __FFLASFFPACK_HAVE_PTHREAD_H 1
- #define __FFLASFFPACK_HAVE_STDDEF_H 1
- #define __FFLASFFPACK_HAVE_STDINT_H 1
- #define __FFLASFFPACK_HAVE_STDIO_H 1
- #define __FFLASFFPACK_HAVE_STDLIB_H 1
- #define __FFLASFFPACK_HAVE_STRINGS_H 1
- #define __FFLASFFPACK_HAVE_STRING_H 1
- #define __FFLASFFPACK_HAVE_SYS_STAT_H 1
- #define __FFLASFFPACK_HAVE_SYS_TIME_H 1
- #define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
- #define __FFLASFFPACK_HAVE_UNISTD_H 1
- #define __FFLASFFPACK_LT_OBJDIR ".libs/"
- #define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
- #define __FFLASFFPACK_PACKAGE "fflas-ffpack"
- #define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"

- #define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
- #define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
- #define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
- #define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
- #define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"
- #define __FFLASFFPACK_SIZEOF_CHAR 1
- #define __FFLASFFPACK_SIZEOF_INT 4
- #define __FFLASFFPACK_SIZEOF_LONG 8
- #define __FFLASFFPACK_SIZEOF_LONG_LONG 8
- #define __FFLASFFPACK_SIZEOF_SHORT 2
- #define __FFLASFFPACK_SIZEOF___INT64_T 8
- #define __FFLASFFPACK_STDC_HEADERS 1
- #define __FFLASFFPACK_USE_OPENMP 1
- #define __FFLASFFPACK_VERSION "2.5.0"

13.60.1 Macro Definition Documentation

13.60.1.1 __FFLASFFPACK_HAVE_BLAS

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

13.60.1.2 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

13.60.1.3 __FFLASFFPACK_HAVE_CXX11

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

13.60.1.4 __FFLASFFPACK_HAVE_DLFCN_H

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

13.60.1.5 __FFLASFFPACK_HAVE_FLOAT_H

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

13.60.1.6 __FFLASFFPACK_HAVE_INT128

```
#define __FFLASFFPACK_HAVE_INT128 1
```

13.60.1.7 __FFLASFFPACK_HAVE_INTTYPES_H

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

13.60.1.8 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.60.1.9 __FFLASFFPACK_HAVE_LIMITS_H

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

13.60.1.10 __FFLASFFPACK_HAVE_LITTLE_ENDIAN

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

13.60.1.11 __FFLASFFPACK_HAVE_PTHREAD_H

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

13.60.1.12 __FFLASFFPACK_HAVE_STDDEF_H

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

13.60.1.13 __FFLASFFPACK_HAVE_STDINT_H

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

13.60.1.14 __FFLASFFPACK_HAVE_STDIO_H

```
#define __FFLASFFPACK_HAVE_STDIO_H 1
```

13.60.1.15 __FFLASFFPACK_HAVE_STDLIB_H

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

13.60.1.16 __FFLASFFPACK_HAVE_STRINGS_H

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

13.60.1.17 __FFLASFFPACK_HAVE_STRING_H

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

13.60.1.18 __FFLASFFPACK_HAVE_SYS_STAT_H

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

13.60.1.19 __FFLASFFPACK_HAVE_SYS_TIME_H

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

13.60.1.20 __FFLASFFPACK_HAVE_SYS_TYPES_H

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

13.60.1.21 __FFLASFFPACK_HAVE_UNISTD_H

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

13.60.1.22 __FFLASFFPACK_LT_OBJDIR

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

13.60.1.23 __FFLASFFPACK_OPENBLAS_NUM_THREADS

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

13.60.1.24 __FFLASFFPACK_PACKAGE

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

13.60.1.25 __FFLASFFPACK_PACKAGE_BUGREPORT

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

13.60.1.26 __FFLASFFPACK_PACKAGE_NAME

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

13.60.1.27 __FFLASFFPACK_PACKAGE_STRING

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

13.60.1.28 __FFLASFFPACK_PACKAGE_TARNAME

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

13.60.1.29 __FFLASFFPACK_PACKAGE_URL

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

13.60.1.30 __FFLASFFPACK_PACKAGE_VERSION

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"
```

13.60.1.31 __FFLASFFPACK_SIZEOF_CHAR

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

13.60.1.32 __FFLASFFPACK_SIZEOF_INT

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

13.60.1.33 __FFLASFFPACK_SIZEOF_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG 8
```

13.60.1.34 __FFLASFFPACK_SIZEOF_LONG_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

13.60.1.35 __FFLASFFPACK_SIZEOF_SHORT

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

13.60.1.36 __FFLASFFPACK_SIZEOF___INT64_T

```
#define __FFLASFFPACK_SIZEOF___INT64_T 8
```

13.60.1.37 __FFLASFFPACK_STDC_HEADERS

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

13.60.1.38 __FFLASFFPACK_USE_OPENMP

```
#define __FFLASFFPACK_USE_OPENMP 1
```

13.60.1.39 __FFLASFFPACK_VERSION

```
#define __FFLASFFPACK_VERSION "2.5.0"
```

13.61 fflas-ffpack-config.h File Reference

Defaults for optimised values.

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/fflas-ffpack-thresholds.h"
#include "fflas-ffpack/fflas-ffpack-default-thresholds.h"
#include "givaro/givconfig.h"
```

Macros

- #define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)

13.61.1 Detailed Description

Defaults for optimised values.
While `fflas-ffpack-optimise.h` is created by `configure` script, (either left blank or filled by optimiser), this file produces the defaults for the optimised values. If `fflas-ffpack-optimise.h` is not empty, then its values preceeds the defaults here.

13.61.2 Macro Definition Documentation

13.61.2.1 GCC_VERSION

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

13.62 fflas-ffpack-default-thresholds.h File Reference

Macros

- #define __FFLASFFPACK_WINOTHRESHOLD 1000
- #define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
- #define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
- #define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
- #define __FFLASFFPACK_PLUQ_THRESHOLD 256
- #define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
- #define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
- #define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
- #define __FFLASFFPACK_FTRTRI_THRESHOLD 32
- #define __FFLASFFPACK_FSYTRF_THRESHOLD 64
- #define __FFLASFFPACK_FSYRK_THRESHOLD 3000

13.62.1 Macro Definition Documentation

13.62.1.1 __FFLASFFPACK_WINOTHRESHOLD

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

13.62.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

13.62.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

13.62.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

13.62.1.5 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

13.62.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

13.62.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

13.62.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

13.62.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

13.62.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

13.62.1.11 __FFLASFFPACK_FSYRK_THRESHOLD

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

13.63 fflas-ffpack-thresholds.h File Reference

13.64 fflas-ffpack.doxy File Reference

13.65 fflas-ffpack.h File Reference

Includes FFLAS and FFPACK.

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

13.65.1 Detailed Description

Includes FFLAS and FFPACK.

13.66 fflas.doxy File Reference

13.67 fflas.h File Reference

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include <cmath>
#include <cstring>
#include <float.h>
#include <algorithm>
#include "fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas_level1.inl"
#include "fflas_level2.inl"
#include "fflas_level3.inl"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas_freduce.h"
#include "fflas_fadd.h"
```

```
#include "fflas_fscal.h"
#include "fflas_fassign.h"
#include "fflas_fgemm.inl"
#include "fflas_pfgemm.inl"
#include "fflas_fgemv.inl"
#include "fflas-ffpack/paladin/pfgemv.inl"
#include "fflas_freivalds.inl"
#include "fflas_fger.inl"
#include "fflas_fsyrk.inl"
#include "fflas_fsyrk_strassen.inl"
#include "fflas_fsyr2k.inl"
#include "fflas_ftrsm.inl"
#include "fflas_pftrsm.inl"
#include "fflas_ftrmm.inl"
#include "fflas_ftrsv.inl"
#include "fflas_faxpy.inl"
#include "fflas_fdot.inl"
#include "fflas-ffpack/field/rns.h"
#include "fflas_fscal_mp.inl"
#include "fflas_freduce_mp.inl"
#include "fflas-ffpack/fflas/fflas_fger_mp.inl"
#include "fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas_ftrsm_mp.inl"
#include "fflas_fgemv_mp.inl"
#include "fflas-ffpack/field/rns.inl"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas_sparse.h"
#include "fflas-ffpack/checkers/checkers_fflas.inl"
```

Macros

- #define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
- #define DOUBLE_TO_FLOAT_CROSSOVER 800

  Thresholds determining which floating point representation to use, depending on the cardinality of the finite field.

13.67.1 Detailed Description

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

Author

Clément Pernet.

13.67.2 Macro Definition Documentation

13.67.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

13.67.2.2 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 800
```

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field. This is only used when the element representation is not a floating point type.

Bug to be benchmarked.

13.68 fflas_bounds.inl File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/flimits.h"
#include <givaro/udl.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_bounds_INL
- #define FFLAS_INT_TYPE uint64_t

Functions

- template<class Field>
  double computeFactorClassic (const Field &F)
- template<> double computeFactorClassic (const Givaro::ModularBalanced< double > &F)
- template<> double computeFactorClassic (const Givaro::ModularBalanced< float > &F)
- template<class Field>
  size_t DotProdBoundClassic (const Field &F, const typename Field::Element &beta)
- Givaro::Integer InfNorm (const size_t M, const size_t N, const Givaro::Integer *A, const size_t lda)
- template<class Field>
  size_t TRSMBound (const Field &)

  TRSMBound.
- template<class Element>
  size_t TRSMBound (const Givaro::Modular< Element > &F)

  Specialization for positive modular representation over double Computes nmax s.t.
- template<class Element>
  size_t TRSMBound (const Givaro::ModularBalanced< Element > &F)

  Specialization for balanced modular representation over double.

13.68.1 Macro Definition Documentation

13.68.1.1 __FFLASFFPACK_fflas_bounds_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

13.68.1.2 FFLAS_INT_TYPE

```
#define FFLAS_INT_TYPE uint64_t
```

13.69 fflas_enum.h File Reference

```
#include <algorithm>
```

Data Structures

- class AreEqual< X, Y >
- class AreEqual< X, X >

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Enumerations

- enum FFLAS_ORDER { FflasRowMajor =101 , FflasColMajor =102 }

  Storage by row or col ?
- enum FFLAS_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }

  Is matrix transposed ?
- enum FFLAS_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasLeftTri = 123 , FflasRightTri = 124 }

  Is triangular matrix's shape upper ?
- enum FFLAS_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }

  Is the triangular matrix implicitly unit diagonal ?
- enum FFLAS_SIDE { FflasLeft = 141 , FflasRight = 142 }

  On what side ?
- enum FFLAS_BASE { FflasDouble = 151 , FflasFloat = 152 , FflasGeneric = 153 }

  `FFLAS_BASE` determines the type of the element representation for Matrix Mult kernel.

Functions

- template<class T>
  const T & min3 (const T &m, const T &n, const T &k)
- template<class T>
  const T & max3 (const T &m, const T &n, const T &k)
- template<class T>
  const T & min4 (const T &m, const T &n, const T &k, const T &l)
- template<class T>
  const T & max4 (const T &m, const T &n, const T &k, const T &l)

13.70 fflas_fadd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas_fadd.inl"
```

Data Structures

- struct support_simd_add< T >

Namespaces

- namespace FFLAS

Functions

- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)

- template<class Field>
  void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)
- template<class Field>
  void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)
- template<class Field>
  void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition.
- template<class Field>
  void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsub : matrix subtraction.
- template<class Field>
  void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  faddin
- template<class Field>
  void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadding for symmetric matrices
- template<class Field>
  void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsubin C = C - B
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition with scaling.

13.71 fflas_fadd.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::vectorised
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_fadd_INL

Functions

- template<class SimdT, class Element, bool positive>
  std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)
- template<class SimdT, class Element, bool positive>
  std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)
- template<class Element>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class Element>
  std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class Field, bool ADD>
  std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)
- template<class Field, bool ADD>
  std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)
- template<class Field, bool ADD>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::GenericTag)
- template<class Field, bool ADD>
  std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)
- template<class Field, bool ADD>
  std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)

13.71.1 Macro Definition Documentation

13.71.1.1 __FFLASFFPACK_fadd_INL

```
#define __FFLASFFPACK_fadd_INL
```

13.72 fflas_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

13.73 fflas_fassign.inl File Reference

```
#include <string.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/debug.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fassign_INL

Functions

- template<class Field>
  void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  fassign : $x \leftarrow y$.
- template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<class Field>
  void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  fassign : $A \leftarrow B$.

13.73.1 Macro Definition Documentation

13.73.1.1 __FFLASFFPACK_fassign_INL

```
#define __FFLASFFPACK_fassign_INL
```

13.74 fflas_faxpy.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::vectorised
- namespace FFLAS::vectorised::unswitch
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_faxpy_INL

Functions

- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)
- template<class Field, class FC>
  void faxpy (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)
- template<class Field>
  void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<class Field>
  void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)

  faxpy : $y \leftarrow \alpha \cdot x + y$.

13.74.1 Macro Definition Documentation

13.74.1.1 __FFLASFFPACK_faxpy_INL

```
#define __FFLASFFPACK_faxpy_INL
```

13.75 fflas_fdot.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fdot_INL

Functions

- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)
- template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)
- template<class Field, class T>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fdot: dot product $x^T y$.

13.75.1 Macro Definition Documentation

13.75.1.1 __FFLASFFPACK_fdot_INL

```
#define __FFLASFFPACK_fdot_INL
```

13.76 fflas_fgemm.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/debug.h"
#include "fflas_fgemm/fgemm_classical.inl"
#include "fflas_fgemm/fgemm_winograd.inl"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fgemm_INL

Functions

- template<class NewField, class Field, class FieldMode>
  Field::Element_ptr fgemm_convert (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template<typename Field, class Cut, class Param>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)

- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  *fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*
- template<typename Field, class ModeT, class ParSeq>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > &H)
- template<class Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field>
  Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsquare: Squares a matrix.
- template<class Field>
  Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)
- template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)
- template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)
- template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)
- template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)

13.76.1 Macro Definition Documentation

13.76.1.1 __FFLASFFPACK_fgemm_INL

```
#define __FFLASFFPACK_fgemm_INL
```

13.77 fgemm_classical.inl File Reference

13.78 fgemm_classical_mp.inl File Reference

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFPACK_fgemm_classical_INL

Functions

- template<typename RNS, typename ParSeqTrait>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)
- template<typename RNS>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)
- template<typename RNS, typename ParSeqTrait>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)
- template<typename RNS, typename Cut, typename Param>
  FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)
- template<class ParSeq>
  Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<typename RNS, class ModeT>
  RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha,

typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)

- template<typename RNS>
  RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)
- Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<class ParSeq>
  Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<size_t K1, size_t K2, class ParSeq>
  RecInt::ruint< K1 > * fgemm (const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const RecInt::ruint< K1 > alpha, const RecInt::ruint< K1 > *A, const size_t lda, const RecInt::ruint< K1 > *B, const size_t ldb, RecInt::ruint< K1 > beta, RecInt::ruint< K1 > *C, const size_t ldc, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

13.78.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

13.78.2 Macro Definition Documentation

13.78.2.1 __FFPACK_fgemm_classical_INL

```
#define __FFPACK_fgemm_classical_INL
```

13.79 fgemm_winograd.inl File Reference

```
#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemm_classical.inl"
#include "schedule_winograd.inl"
#include "schedule_winograd_acc.inl"
#include "schedule_winograd_acc_ip.inl"
#include "schedule_winograd_ip.inl"
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
- #define NEWWINO

Functions

- template<class Field>
  int WinogradThreshold (const Field &F)

  Computes the number of recursive levels to perform.
- template<> int WinogradThreshold (const Givaro::Modular< float > &F)
- template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)
- template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)
- template<class Field>
  int WinogradSteps (const Field &F, const size_t &m)

  Computes the number of recursive levels to perform.
- template<class Field, class FieldMode>
  void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field, class FieldMode>
  void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field, class FieldMode>
  void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)
- template<class Field, class ModeT>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)
- template<class Field, class ModeT, class Cut, class Param>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)

13.79.1 Macro Definition Documentation

13.79.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

13.79.1.2 NEWWINO

```
#define NEWWINO
```

13.80 matmul.doxy File Reference

13.81 schedule_bini.inl File Reference

Bini implementation.

Namespaces

- namespace FFLAS
- namespace FFLAS::BLAS3

Macros

- #define __FFLASFFPACK_fgemm_bini_INL

Functions

- template<class Field>
  void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE base, const size_t rec_level)

13.81.1 Detailed Description

Bini implementation.

13.81.2 Macro Definition Documentation

13.81.2.1 __FFLASFFPACK_fgemm_bini_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

13.82 schedule_winograd.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::BLAS3

Macros

- #define __FFLASFFPACK_fgemm_winograd_INL

Functions

- template<class Field, class FieldTrait, class Strat, class Param>
  Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)

- template<class Field, class FieldTrait>
  void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Field↩ Trait > &WH)

13.82.1 Macro Definition Documentation

13.82.1.1 __FFLASFFPACK_fgemm_winograd_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

13.83 schedule_winograd_acc.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::BLAS3

Macros

- #define __FFLASFFPACK_fgemm_winograd_acc_INL

Functions

- template<class Field, class FieldTrait>
  void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

13.83.1 Macro Definition Documentation

13.83.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

13.84 schedule_winograd_acc_ip.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::BLAS3

Macros

- #define __FFLASFFPACK_fgemm_winograd_acc_ip_INL

Functions

- template<class Field, class FieldTrait>
  void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field↩ Trait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

13.84.1 Macro Definition Documentation

13.84.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

13.85 schedule_winograd_ip.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::BLAS3

Macros

- #define __FFLASFFPACK_fgemm_winograd_ip_INL

Functions

- template<class Field, class FieldTrait>
  void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field↩ Trait > &WH)

- template<class Field, class FieldTrait>
  void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field, class FieldTrait>
  void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

13.85.1 Macro Definition Documentation

13.85.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

13.86 fflas_fgemv.inl File Reference

```
#include <givaro/zring.h>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fgemv_INL

Functions

- template<typename FloatElement, class Field>
  Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)

- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)

  finite prime Field GEneral Matrix Vector multiplication.
- Givaro::ZRing< int64_t >::Element_ptr fgemv (const Givaro::ZRing< int64_t > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *X, const size_t incX, const int64_t beta, int64_t *Y, const size_t incY, MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::DoubleDomain::Element_ptr fgemv (const Givaro::DoubleDomain &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::ConstElement_ptr X, const size_t incX, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fgemv (const Givaro::FloatDomain &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::ConstElement_ptr X, const size_t incX, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field, class Cut, class Param>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Parallel< Cut, Param > &parH)
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Sequential &seqH)

13.86.1 Macro Definition Documentation

13.86.1.1 __FFLASFFPACK_fgemv_INL

```
#define __FFLASFFPACK_fgemv_INL
```

13.87 fflas_fgemv_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fgemv_mp_INL

Functions

- FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::Integer * fgemv (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- Givaro::Integer * fgemv (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<size_t K1, size_t K2, class ParSeq>
  RecInt::ruint< K1 > * fgemv (const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const RecInt::ruint< K1 > alpha, const RecInt::ruint< K1 > *A, const size_t lda, const RecInt::ruint< K1 > *X, const size_t incx, RecInt::ruint< K1 > beta, RecInt::ruint< K1 > *Y, const size_t incy, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

13.87.1 Macro Definition Documentation

13.87.1.1 __FFLASFFPACK_fgemv_mp_INL

```
#define __FFLASFFPACK_fgemv_mp_INL
```

13.88 fflas_fger.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fger_INL

Functions

- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)

fger: rank one update of a general matrix

- template<class FloatElement, class Field>
  void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)
- template<class Field, class AnyTag>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)
- void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)

13.88.1 Macro Definition Documentation

13.88.1.1 __FFLASFFPACK_fger_INL

```
#define __FFLASFFPACK_fger_INL
```

13.89 fflas_fger_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFPACK_fger_mp_INL

Functions

- void fger (const Givaro::Modular< Givaro::Integer > &F, const size_t M, const size_t N, const typename Givaro::Integer alpha, typename Givaro::Integer *x, const size_t incx, typename Givaro::Integer *y, const size_t incy, typename Givaro::Integer *A, const size_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename RNS>
  void fger (const FFPACK::RNSInteger< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSInteger< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSInteger< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename RNS>
  void fger (const FFPACK::RNSIntegerMod< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSIntegerMod< RNS >::Element alpha, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > &H)

13.89.1 Macro Definition Documentation

13.89.1.1 __FFPACK_fger_mp_INL

```
#define __FFPACK_fger_mp_INL
```

13.90 fflas_freduce.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/utils/cast.h"
#include "fflas-ffpack/fflas/fflas_freduce.inl"
```

Data Structures

- struct support_simd_mod< T >
- struct support_fast_mod< T >
- struct support_fast_mod< float >
- struct support_fast_mod< double >
- struct support_fast_mod< int64_t >

Namespaces

- namespace FFLAS

Functions

- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow y mod F$.
- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow x mod F$.

- template<class Field>
  void freduce_constoverride (const Field &F, const size_t m, typename Field::ConstElement_ptr A, const size_t incX)
- template<class Field, class ConstOtherElement_ptr>
  void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)
- template<class Field>
  void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  finit Initializes X in F.
- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow A mod F$.
- template<class Field>
  void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

  freduce for square symmetric matrices
- template<class Field>
  void pfreduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const size_t numths)
- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow B mod F$.
- template<class Field>
  void freduce_constoverride (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  finit $A \leftarrow B mod F$.
- template<class Field>
  void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

13.91 fflas_freduce.inl File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

Data Structures

- struct HelperMod< Field, ElementCategories::MachineIntTag >
- struct HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >
- struct HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >

Namespaces

- namespace FFLAS
- namespace FFLAS::vectorised
- namespace FFLAS::vectorised::unswitch
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_fflas_freduce_INL
- #define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCMARK LATER */

Functions

- template<class T>
  std::enable_if<!std::is_integral< T >::value, T >::type reduce (T A, T B)
- template<class T>
  std::enable_if< std::is_integral< T >::value, T >::type reduce (T A, T B)
- template<> Givaro::Integer reduce (Givaro::Integer A, Givaro::Integer B)
- float reduce (float A, float B, float invB, float min, float max)
- double reduce (double A, double B, double invB, double min, double max)
- int64_t reduce (int64_t A, int64_t p, double invp, double min, double max, int64_t pow50rem)
- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineIntTag > &H)
- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineFloatTag > &H)
- template<class Field>
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > &H)
- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)
- template<class Field, class FC>
  void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)
- template<class Field, class FC>
  void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename Field::Element_ptr A, const size_t incX, FC)

13.91.1 Macro Definition Documentation

13.91.1.1 __FFLASFFPACK_fflas_freduce_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

13.91.1.2 FFLASFFPACK_COPY_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCMARK LATER */
```

13.92 fflas_freduce_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_freduce_mp_INL

Functions

- template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, size_t inc)
- template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, FFPACK::rns_double::Element_ptr A, size_t lda)

13.92.1 Macro Definition Documentation

13.92.1.1 __FFLASFFPACK_fflas_freduce_mp_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

13.93 fflas_freivalds.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_freivalds_INL

Functions

- template<class Field>
  bool freivalds (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::ConstElement_ptr C, const size_t ldc)

  *freivalds: **F**reivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

13.93.1 Macro Definition Documentation

13.93.1.1 __FFLASFFPACK_freivalds_INL

```
#define __FFLASFFPACK_freivalds_INL
```

13.94 fflas_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

13.95 fflas_fscal.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::vectorised
- namespace FFLAS::vectorised::unswitch
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_fscal_INL

Functions

- template<class Field>
  std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, const size_t &incX, const size_t &incY)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin (const Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX, FieldCategories::ModularTag)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)
- template<class Field, class FC>
  void fscalin (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX, FC)
- template<class Field, class FC>
  void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)
- template<class Field>
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

fscalin $x \leftarrow \alpha \cdot x$.

- template<class Field>
  void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  fscal $y \leftarrow \alpha \cdot x$.
- template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element↩ _ptr y, const size_t incy)
- template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<class Field>
  void fscalin (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)

  fscalin $A \leftarrow a \cdot A$.
- template<class Field>
  void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  fscal $B \leftarrow a \cdot A$.

13.95.1 Macro Definition Documentation

13.95.1.1 __FFLASFFPACK_fscal_INL

```
#define __FFLASFFPACK_fscal_INL
```

13.96 fflas_fscal_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas_fscal.h"
#include "fflas_fgemm.inl"
#include "fflas-ffpack/fflas/fflas_freduce_mp.inl"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fscal_mp_INL

Functions

- template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t inc)
- template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)
- template<> void fscalin (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)

- template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)
- template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)
- template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)
- template<> void fscalin (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)
- template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)

13.96.1 Macro Definition Documentation

13.96.1.1 __FFLASFFPACK_fscal_mp_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

13.97 fflas_fsyr2k.inl File Reference

```
#include <fflas-ffpack/utils/fflas_io.h>
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_fsyr2k_INL

Functions

- template<class Field>
  Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyr2k: Symmetric Rank 2K update

13.97.1 Macro Definition Documentation

13.97.1.1 __FFLASFFPACK_fflas_fsyr2k_INL

```
#define __FFLASFFPACK_fflas_fsyr2k_INL
```

13.98 fflas_fsyrk.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fsyrk_INL

Functions

- template<class NewField, class Field, class FieldMode>
  Field::Element_ptr fsyrk_convert (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, FieldMode > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyrk: Symmetric Rank K update
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const FFLAS_UPLO UpLo, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field, typename Mode>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fsyrk (const Givaro::FloatDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::FloatDomain::Element alpha, Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)

- Givaro::DoubleDomain::Element_ptr fsyrk (const Givaro::DoubleDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::DoubleDomain::Element alpha, Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::↩ Element beta, Givaro::DoubleDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq, const size_t threshold)
- template<class Field, class Cut, class Param>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &two↩ Block, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling

13.98.1 Macro Definition Documentation

13.98.1.1 __FFLASFFPACK_fflas_fsyrk_INL

```
#define __FFLASFFPACK_fflas_fsyrk_INL
```

13.99 fflas_fsyrk_strassen.inl File Reference

```
#include <givaro/givintsqrootmod.h>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fsyrk_strassen_INL

Functions

- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)

- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class FieldTrait>
  void computeS1S2 (const Field &F, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element x, const typename Field::Element y, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr S, const size_t lds, typename Field::Element_ptr T, const size_t ldt, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field, class Mode>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Mode > &H)
- template<class Field, class FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO uplo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

13.99.1 Macro Definition Documentation

13.99.1.1 __FFLASFFPACK_fflas_fsyrk_strassen_INL

```
#define __FFLASFFPACK_fflas_fsyrk_strassen_INL
```

13.100 fflas_ftrmm.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_ftrmm_INL

Functions

- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  *ftrmm: **TR**iangular **M**atrix **M**ultiply.*
- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

*ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A) B + beta C$ *or* $C \leftarrow \alpha B \mathrm{op}(A) + beta C$.

13.100.1 Macro Definition Documentation

13.100.1.1 __FFLASFFPACK_ftrmm_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

13.101 fflas_ftrsm.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_ftrsm_INL

Functions

- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)
- template<class Field, class Cut, class Param>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)
- template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)

13.101.1 Macro Definition Documentation

13.101.1.1 __FFLASFFPACK_ftrsm_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

13.102 fflas_ftrsm_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

```
#include <cmath>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFPACK_ftrsm_mp_INL

Functions

- void ftrsm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, Givaro::Integer *B, const size_t ldb)
- void cblas_imptrsm (const enum FFLAS_ORDER Order, const enum FFLAS_SIDE Side, const enum FFLAS_UPLO Uplo, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_DIAG Diag, const int M, const int N, const FFPACK::rns_double_elt alpha, FFPACK::rns_double_elt_cstptr A, const int lda, FFPACK::rns_double_elt_ptr B, const int ldb)

13.102.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

13.102.2 Macro Definition Documentation

13.102.2.1 __FFPACK_ftrsm_mp_INL

```
#define __FFPACK_ftrsm_mp_INL
```

13.103 fflas_ftrsv.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_ftrsv_INL

Functions

- template<class Field>
  void ftrsv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

13.103.1 Macro Definition Documentation

13.103.1.1 __FFLASFFPACK_ftrsv_INL

```
#define __FFLASFFPACK_ftrsv_INL
```

13.104 fflas_helpers.inl File Reference

```
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/flimits.h"
#include <algorithm>
```

Data Structures

- struct Auto
- struct Classic
- struct DivideAndConquer
- struct Winograd
- struct WinogradPar
- struct Bini
- struct AlgoChooser< ModeT, ParSeq >
- struct AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >
- struct MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >
- struct Recursive
- struct Iterative
- struct Hybrid
- struct TRSMHelper< RecIterTrait, ParSeqTrait >

  TRSM Helper.

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected
- namespace FFLAS::MMHelperAlgo
- namespace FFLAS::StructureHelper

  StructureHelper for ftrsm.

Macros

- #define __FFLASFFPACK_fflas_fflas_mmhelper_INL

Functions

- template<class Field>
  int WinogradSteps (const Field &F, const size_t &m)

  Computes the number of recursive levels to perform.
- template<class DFE>
  size_t min_types (const DFE &k)
- template<> size_t min_types (const RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::rint< 10 > &k)
- template<> size_t min_types (const Givaro::Integer &k)
- template<class T>
  bool unfit (T x)
- template<> bool unfit (int64_t x)
- template<size_t K>
  bool unfit (RecInt::rint< K > x)
- template<> bool unfit (RecInt::rint< 6 > x)

13.104.1 Macro Definition Documentation

13.104.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

13.105 igemm.doxy File Reference

13.106 igemm.h File Reference

```
#include "igemm_kernels.h"
#include "igemm_tools.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm.inl"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Enumerations

- enum number_kind { zero =0 , one =1 , mone =-1 , other =2 }

Functions

- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)
- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)
- void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)
- void igemm_ (const enum FFLAS_ORDER Order, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, const size_t M, const size_t N, const size_t K, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *B, const size_t ldb, const int64_t beta, int64_t *C, const size_t ldc)

13.107 igemm.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_igemm_igemm_INL

Functions

- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)
- template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>
  void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)

- void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)
- void igemm_ (const enum FFLAS_ORDER Order, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, const size_t M, const size_t N, const size_t K, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *B, const size_t ldb, const int64_t beta, int64_t *C, const size_t ldc)

13.107.1 Macro Definition Documentation

13.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

13.108 igemm_kernels.h File Reference

```
#include "igemm_kernels.inl"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::details

Functions

- template<enum number_kind K>
  void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)

13.109 igemm_kernels.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm_tools.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL

Functions

- template<enum number_kind K>
  void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t *blB, int64_t *C, size_t ldc)
- template<enum number_kind K>
  void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)

13.109.1 Macro Definition Documentation

13.109.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

13.110 igemm_tools.h File Reference

```
#include "igemm_tools.inl"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::details

Functions

- template<size_t k, bool transpose>
  void pack_lhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- template<size_t k, bool transpose>
  void pack_rhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- void gebp (size_t rows, size_t cols, size_t depth, int64_t *C, size_t ldc, const int64_t *blockA, size_t lda, const int64_t *BlockB, size_t ldb, int64_t *BlockW)
- void BlockingFactor (size_t &m, size_t &n, size_t &k)

13.111 igemm_tools.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_fflas_igemm_igemm_tools_INL

Functions

- template<size_t k, bool transpose>
  void pack_rhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- template<size_t k, bool transpose>
  void pack_lhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)
- void BlockingFactor (size_t &m, size_t &n, size_t &k)

13.111.1 Macro Definition Documentation

13.111.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

13.112 fflas_level1.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_fflas_level1_INL

Functions

- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow xmodF$.
- template<class Field>
  void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  freduce $x \leftarrow ymodF$.
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  finit $x \leftarrow ymodF$.
- template<class Field>
  void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  finit Initializes X *in* $F\$$.
- template<class Field, class OtherElement_ptr>
  void fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fconvert $x \leftarrow ymodF$.
- template<class Field>
  void fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  fnegin $x \leftarrow -x$.

- template<class Field>
  void fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  fneg $x \leftarrow -y$.
- template<class Field>
  void fzero (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

  fzero : $A \leftarrow 0$.
- template<class Field, class RandIter>
  void frand (const Field &F, RandIter &G, const size_t n, typename Field::Element_ptr X, const size_t incX)

  frand : $A \leftarrow random$.
- template<class Field>
  bool fiszero (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX)

  fiszero : test $X = 0$.
- template<class Field>
  bool fequal (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fequal : test $X = Y$.
- template<class Field>
  void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

  fassign : $x \leftarrow y$.
- template<class Field>
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- template<class Field>
  void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  fscal $y \leftarrow \alpha \cdot x$.
- template<class Field>
  void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template<class Field>
  void faxpby (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

  fdot: dot product $x^T y$.
- template<class Field>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)
- template<typename Field, class Cut, class Param>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field>
  void fswap (const Field &F, const size_t N, typename Field::Element_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)

  fswap: $X \leftrightarrow Y$.

- template<class Field>
  void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)
- template<class Field>
  void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)
- template<class Field>
  void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)
- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)
- template<class Field>
  void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)

13.112.1 Macro Definition Documentation

13.112.1.1 __FFLASFFPACK_fflas_fflas_level1_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

13.113 fflas_level2.inl File Reference

```
#include "givaro/zring.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_fflas_level2_INL

Functions

- template<class Field>
  void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  fassign : $A \leftarrow B$.
- template<class Field>
  void fzero (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

fzero : $A \leftarrow 0$.

- template<class Field>
  void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fzero : $A \leftarrow 0$ *for a triangular matrix.*
- template<class Field, class RandIter>
  void frand (const Field &F, RandIter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  frand : $A \leftarrow random$.
- template<class Field>
  bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)

  fequal : test $A = B$.
- template<class Field>
  bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)

  fiszero : test $A = 0$.
- template<class Field>
  void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const typename Field::Element &d)

  creates a diagonal matrix
- template<class Field>
  void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  creates a diagonal matrix
- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow AmodF$.
- template<class Field>
  void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

  freduce for square symmetric matrices
- template<class Field>
  void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  freduce $A \leftarrow BmodF$.
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  finit $A \leftarrow BmodF$.
- template<class Field, class OtherElement_ptr>
  void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  finit Initializes A *in* F.
- template<class Field, class OtherElement_ptr>
  void fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)

  fconvert $A \leftarrow BmodF$.
- template<class Field>
  void fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fnegin $A \leftarrow -A$.
- template<class Field>
  void fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)

  fneg $A \leftarrow -B$.

- template<class Field>
  void fscalin (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)

  fscalin $A \leftarrow a \cdot A$.
- template<class Field>
  void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  fscal $B \leftarrow a \cdot A$.
- template<class Field>
  void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template<class Field>
  void faxpby (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t ldy)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template<class Field>
  void fmove (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  fmove : $A \leftarrow B$ *and* $B \leftarrow 0$.
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition.
- template<class Field>
  void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsub : matrix subtraction.
- template<class Field>
  void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fsubin C = C - B
- template<class Field>
  void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadd : matrix addition with scaling.
- template<class Field>
  void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  faddin
- template<class Field>
  void faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadding for symmetric matrices
- template<class Field>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)

  finite prime Field GEneral Matrix Vector multiplication.
- template<class Field>
  void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)

fger: rank one update of a general matrix

- template<class Field>
  void ftrsv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$
- template<class Field>
  size_t bitsize (const Field &F, size_t M, size_t N, const typename Field::ConstElement_ptr A, size_t lda)

  bitsize: Computes the largest bitsize of the matrix' coefficients.
- template<> size_t bitsize< Givaro::ZRing< Givaro::Integer > > (const Givaro::ZRing< Givaro::Integer > &F, size_t M, size_t N, const Givaro::Integer *A, size_t lda)
- template<class Field>
  void ftrmv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)

  ftrsm: TRiangular Matrix Vector prodcut Computes $X \leftarrow \mathrm{op}(A)X$

13.113.1 Macro Definition Documentation

13.113.1.1 __FFLASFFPACK_fflas_fflas_level2_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

13.114 fflas_level3.inl File Reference

```
#include "fflas_bounds.inl"
#include "fflas_helpers.inl"
#include "fflas-ffpack/paladin/parallel.h"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fflas_level3_INL
- #define __FFLAS__TRSM_READONLY

Functions

- template<class Field>
  void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)
- template<class Field>
  void MatF2MatFl_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)
- template<class Field>
  void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

  *ftrsm: **TR**iangular **S**ystem solve with **M**atrix.*
- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

*ftrmm: **TR**iangular **M**atrix **M**ultiply.*

- template<class Field>
  void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  *ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha \mathrm{op}(A) B + beta C$ or $C \leftarrow \alpha B \mathrm{op}(A) + beta C$.*
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyrk: Symmetric Rank K update
- template<class Field, typename FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &H)
- template<class Field>
  Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsyr2k: Symmetric Rank 2K update
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq, const size_t threshold)
- template<class Field, class Cut, class Param>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)
- template<class Field>
  Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &two↩
  Block, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)

  fsyrk: Symmetric Rank K update with diagonal scaling
- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  *fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*
- template<typename Field>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE

tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)

- template<typename Field, class Cut, class Param>
  Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)
- template<typename Field>
  Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t numthreads=0)
- template<class Field>
  Field::Element * pfgemm_1D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)
- template<class Field>
  Field::Element * pfgemm_2D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)
- template<class Field>
  Field::Element * pfgemm_3D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)
- template<class Field>
  Field::Element_ptr pfgemm_3D_rec2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)
- template<class Field>
  Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)

  fsquare: Squares a matrix.

13.114.1 Macro Definition Documentation

13.114.1.1 __FFLASFFPACK_fflas_fflas_level3_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

13.114.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.115 fflas_pfgemm.inl File Reference

```
#include "fflas-ffpack/paladin/blockcuts.inl"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/paladin/pfgemm_variants.inl"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_pfgemm_INL
- #define __FFLASFFPACK_SEQPARTHRESHOLD 220
- #define __FFLASFFPACK_DIMKPENALTY 1

Functions

- template<class Field, class ModeTrait, class Strat, class Param>
  std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > >::value, typenameField::Element_ptr >::type fgemm (const Field &F, const FFLAS::FFLAS_TRANSPOSE ta, const FFLAS::FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > > &H)

13.115.1 Macro Definition Documentation

13.115.1.1 __FFLASFFPACK_fflas_pfgemm_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

13.115.1.2 __FFLASFFPACK_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

13.115.1.3 __FFLASFFPACK_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

13.116 fflas_pftrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_pftrsm_INL
- #define PTRSM_HYBRID_THRESHOLD 256

Functions

- template<class Field, class Cut, class Param>
  Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &H)

- template<class Field, class Cut, class Param>
  Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)

13.116.1 Macro Definition Documentation

13.116.1.1 __FFLASFFPACK_fflas_pftrsm_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

13.116.1.2 PTRSM_HYBRID_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

13.117 fflas_simd.h File Reference

```
#include "fflas-ffpack/utils/fflas_intrinsic.h"
#include <iostream>
#include <type_traits>
#include <limits>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include "givaro/givtypestring.h"
#include <fflas-ffpack/fflas/fflas_simd/simd_modular.inl>
```

Data Structures

- struct support_simd< T >
- struct is_simd< T >
- struct NoSimd< T >
- struct SimdChooser< T, bool, bool >
- struct SimdChooser< T, false, b >
- struct SimdChooser< T, true, false >
- struct SimdChooser< T, true, true >

Namespaces

- namespace FFLAS

Macros

- #define SIMD_INT 1
- #define INLINE inline
- #define CONST
- #define PURE
- #define NORML_MOD(C, P, NEGP, MIN, MAX, Q, T)
- #define FLOAT_MOD(C, P, INVP, Q)

Typedefs

- template<class T>
  using Simd = typename SimdChooser<T>::value

13.117.1 Macro Definition Documentation

13.117.1.1 SIMD_INT

```
#define SIMD_INT 1
```

13.117.1.2 INLINE

```
#define INLINE inline
```

13.117.1.3 CONST

```
#define CONST
```

13.117.1.4 PURE

```
#define PURE
```

13.117.1.5 NORML_MOD

```
#define NORML_MOD(
                C,
                P,
                NEGP,
                MIN,
                MAX,
                Q,
                T)
```

Value:

```
{
      \
    Q = greater(C, MAX);
      \
    T = lesser(C, MIN);
      \
    Q = vand(Q, NEGP);
      \
    T = vand(T, P);
      \
    Q = vor(Q, T);
      \
    C = add(C, Q);
      \
}
```

13.117.1.6 FLOAT_MOD

```
#define FLOAT_MOD(
                C,
                P,
                INVP,
                Q)
```

Value:

```
{
      \
    Q = mul(C, INVP);
      \
    Q = floor(Q);
      \
    C = fnmadd(C, Q, P);
      \
}
```

13.117.2 Typedef Documentation

13.117.2.1 Simd

```
template<class T>
```

```
using Simd = typename SimdChooser<T>::value
```

13.118 simd.doxy File Reference

13.119 simd128.inl File Reference

```
#include "simd128_float.inl"
#include "simd128_double.inl"
```

Data Structures

- struct Simd128i_base

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL

Typedefs

- template<class T>
  using Simd128

13.119.1 Macro Definition Documentation

13.119.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

13.119.2 Typedef Documentation

13.119.2.1 Simd128

```
template<class T>
using Simd128
```

Initial value:

```
Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.120 simd128_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

13.120.1 Macro Definition Documentation

13.120.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

13.121 simd128_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

13.121.1 Macro Definition Documentation

13.121.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

13.122 simd128_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, true, true, 2 >
- union Simd128_impl< true, true, true, 2 >::Converter
- struct Simd128_impl< true, true, false, 2 >
- union Simd128_impl< true, true, false, 2 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL

13.122.1 Macro Definition Documentation

13.122.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

13.123 simd128_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
```

```
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, true, true, 4 >
- union Simd128_impl< true, true, true, 4 >::Converter
- struct Simd128_impl< true, true, false, 4 >
- union Simd128_impl< true, true, false, 4 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL

13.123.1 Macro Definition Documentation

13.123.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

13.124 simd128_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, true, true, 8 >
- union Simd128_impl< true, true, true, 8 >::Converter
- struct Simd128_impl< true, true, false, 8 >
- union Simd128_impl< true, true, false, 8 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
- #define vect_t Simd128_impl<true,true,true,8>::vect_t

13.124.1 Macro Definition Documentation

13.124.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

13.124.1.2 vect_t

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

13.125 simd256.inl File Reference

```
#include "simd256_float.inl"
#include "simd256_double.inl"
```

Data Structures

- struct Simd256fp_base
- struct Simd256i_base

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL

Typedefs

- template<class T>
  using Simd256

13.125.1 Macro Definition Documentation

13.125.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

13.125.2 Typedef Documentation

13.125.2.1 Simd256

```
template<class T>
using Simd256
```

Initial value:

```
Simd256_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.126 simd256_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, false, true, 8 >
- union Simd256_impl< true, false, true, 8 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

13.126.1 Macro Definition Documentation

13.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

13.127 simd256_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

13.127.1 Macro Definition Documentation

13.127.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

13.128 simd256_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, true, true, 2 >
- union Simd256_impl< true, true, true, 2 >::Converter
- struct Simd256_impl< true, true, false, 2 >
- union Simd256_impl< true, true, false, 2 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL

13.128.1 Macro Definition Documentation

13.128.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

13.129 simd256_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, true, true, 4 >
- union Simd256_impl< true, true, true, 4 >::Converter
- struct Simd256_impl< true, true, false, 4 >
- union Simd256_impl< true, true, false, 4 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL

13.129.1 Macro Definition Documentation

13.129.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

13.130 simd256_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, true, true, 8 >
- union Simd256_impl< true, true, true, 8 >::Converter
- struct Simd256_impl< true, true, false, 8 >
- union Simd256_impl< true, true, false, 8 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
- #define vect_t Simd256_impl<true, true, true, 8>::vect_t

13.130.1 Macro Definition Documentation

13.130.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

13.130.1.2 vect_t

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

13.131 simd512.inl File Reference

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

Data Structures

- struct Simd512i_base

Macros

- #define __FFLASFFPACK_simd512_INL

Typedefs

- template<class T>
  using Simd512

13.131.1 Macro Definition Documentation

13.131.1.1 __FFLASFFPACK_simd512_INL

```
#define __FFLASFFPACK_simd512_INL
```

13.131.2 Typedef Documentation

13.131.2.1 Simd512

```
template<class T>
using Simd512
```

Initial value:

```
Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.132 simd512_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd512_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_simd512_double_INL

13.132.1 Macro Definition Documentation

13.132.1.1 __FFLASFFPACK_simd512_double_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

13.133 simd512_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd512_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_simd512_float_INL

13.133.1 Macro Definition Documentation

13.133.1.1 __FFLASFFPACK_simd512_float_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

13.134 simd512_int32.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, true, true, 4 >
- union Simd256_impl< true, true, true, 4 >::Converter
- struct Simd256_impl< true, true, false, 4 >
- union Simd256_impl< true, true, false, 4 >::Converter

Macros

- #define __FFLASFFPACK_simd512_int32_INL

13.134.1 Macro Definition Documentation

13.134.1.1 __FFLASFFPACK_simd512_int32_INL

```
#define __FFLASFFPACK_simd512_int32_INL
```

13.135 simd512_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd512_impl< true, true, true, 8 >
- union Simd512_impl< true, true, true, 8 >::Converter
- struct Simd512_impl< true, true, false, 8 >
- union Simd512_impl< true, true, false, 8 >::Converter

Macros

- #define _simd512_int64_INL
- #define vect_t Simd512_impl<true, true, true, 8>::vect_t

13.135.1 Macro Definition Documentation

13.135.1.1 _simd512_int64_INL

```
#define _simd512_int64_INL
```

13.135.1.2 vect_t

```
#define vect_t Simd512_impl<true, true, true, 8>::vect_t
```

13.136 simd_modular.inl File Reference

Data Structures

- class FieldSimd< _Field >

13.137 fflas_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/paladin/parallel.h"
#include <recint/recint.h>
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <type_traits>
#include <vector>
#include <iostream>
#include "fflas-ffpack/fflas/fflas_sparse/sparse_matrix_traits.h"
#include "fflas-ffpack/fflas/fflas_sparse/utils.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr.h"
#include "fflas-ffpack/fflas/fflas_sparse/coo.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell.h"
#include "fflas-ffpack/fflas/fflas_sparse/sell.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd.h"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo.h"
#include "fflas-ffpack/fflas/fflas_sparse.inl"
#include "fflas-ffpack/fflas/fflas_sparse/read_sparse.h"
```

Data Structures

- struct HelperFlag
- struct CsrMat< Field >
- struct CooMat< Field >
- struct EllMat< Field >
- struct SpMat< Field, flag >

Namespaces

- namespace MKL_CONFIG
- namespace FFLAS
- namespace FFLAS::sparse_details

Macros

- #define index_t uint32_t
- #define ROUND_DOWN(x, s)
- #define __FFLASFFPACK_CACHE_LINE_SIZE 64
- #define assume_aligned(pout, pin, v)
- #define DENSE_THRESHOLD 0.5

Enumerations

- enum class SparseMatrix_t {
  CSR , CSR_ZO , CSC , CSC_ZO ,
  COO , COO_ZO , ELL , ELL_ZO ,
  SELL , SELL_ZO , ELL_simd , ELL_simd_ZO ,
  CSR_HYB , HYB_ZO }

Functions

- template<class Field>
  void init_y (const Field &F, const size_t m, const typename Field::Element b, typename Field::Element_ptr y)
- template<class Field>
  void init_y (const Field &F, const size_t m, const size_t n, const typename Field::Element b, typename Field::Element_ptr y, const int ldy)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloa
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value)>::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, type-
  name Field::Element_ptr y, FC fc, MZO mzo)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value >::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, type-
  name Field::Element_ptr y, FC fc, MZO mzo)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr
  y, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag,
  NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag,
  NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag,
  NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag,
  NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr
  y, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag,
  ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM
  &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag,
  ZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr
  y, FieldCategories::ModularTag, std::true_type)

- template<class Field, class SM, class FCat, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloa
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr
  x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr
  x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,
  typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,
  typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM
  &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
  FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,
  typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloa
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value)>::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename
  Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename
  Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,
  typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)

- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)
- template<class Field, class SM>
  void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)

13.137.1 Macro Definition Documentation

13.137.1.1 index_t

```
#define index_t uint32_t
```

13.137.1.2 ROUND_DOWN

```
#define ROUND_DOWN(
              x,
              s)
```

Value:

```
((x) & ~((s)-1))
```

13.137.1.3 __FFLASFFPACK_CACHE_LINE_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

13.137.1.4 assume_aligned

```
#define assume_aligned(
              pout,
              pin,
              v)
```

Value:

```
decltype(pin) pout = pin;
```

13.137.1.5 DENSE_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

13.138 fflas_sparse.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details

Macros

- #define __FFLASFFPACK_fflas_fflas_sparse_INL

Functions

- template<class Field>
  void init_y (const Field &F, const size_t m, const typename Field::Element b, typename Field::Element_ptr y)
- template<class Field>
  void init_y (const Field &F, const size_t m, const size_t n, const typename Field::Element b, typename Field::Element_ptr y, const int ldy)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloa
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value)>::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, type-
  name Field::Element_ptr y, FC fc, MZO mzo)
- template<class Field, class SM, class FC, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat
  >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag
  >::value >::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, type-
  name Field::Element_ptr y, FC fc, MZO mzo)

- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloa >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM, class FCat, class MZO>
  std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)

- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)
- template<class Field, class SM>
  void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)
- template<class Field, class SM>
  void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)

13.138.1 Macro Definition Documentation

13.138.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

13.139 coo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmm.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::COO >
- struct Sparse< _Field, SparseMatrix_t::COO_ZO >

Namespaces

- namespace FFLAS

Functions

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)

13.140 coo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_coo_spmm_INL

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t block↩ Size, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_↩ t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_↩ t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_↩ t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)

13.140.1 Macro Definition Documentation

13.140.1.1 __FFLASFFPACK_fflas_sparse_coo_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmm_INL
```

13.141 coo_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_coo_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.141.1 Macro Definition Documentation

13.141.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

13.142 coo_utils.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_sparse_coo_utils_INL

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.142.1 Macro Definition Documentation

13.142.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

13.143 csr.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmm.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::CSR >
- struct Sparse< _Field, SparseMatrix_t::CSR_ZO >

Namespaces

- namespace FFLAS

Functions

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)

13.144 csr_pspmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL

Functions

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)

13.144.1 Macro Definition Documentation

13.144.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

13.145 csr_pspmv.inl File Reference

```
#include <thread>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.145.1 Macro Definition Documentation

13.145.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

13.146 csr_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)

- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)

13.146.1 Macro Definition Documentation

13.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL
```

13.147 csr_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)

- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.147.1 Macro Definition Documentation

13.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

13.148 csr_utils.inl File Reference

Namespaces

- namespace FFLAS

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)
- template<class Field>
  std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)
- template<class IndexT>
  void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT>
  void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT, size_t RECINT_SIZE>
  void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class IndexT, size_t RECINT_SIZE>
  void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.149 csr_hyb.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmm.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::CSR_HYB >

Namespaces

- namespace FFLAS

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.150 csr_hyb_pspmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL

Functions

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)

13.150.1 Macro Definition Documentation

13.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

13.151 csr_hyb_pspmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)

13.151.1 Macro Definition Documentation

13.151.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

13.152 csr_hyb_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)

13.152.1 Macro Definition Documentation

13.152.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL
```

13.153 csr_hyb_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)

13.153.1 Macro Definition Documentation

13.153.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

13.154 csr_hyb_utils.inl File Reference

Data Structures

- struct Info
- struct Coo< ValT, IdxT >

Namespaces

- namespace FFLAS
- namespace FFLAS::csr_hyb_details

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.154.1 Macro Definition Documentation

13.154.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

13.155 ell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmm.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::ELL >
- struct Sparse< _Field, SparseMatrix_t::ELL_ZO >

Namespaces

- namespace FFLAS

Functions

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)

13.156 ell_pspmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL

Functions

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)
- template<class Field, class Func>
  void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)
- template<class Field, class Func>
  void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)

13.156.1 Macro Definition Documentation

13.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

13.157 ell_pspmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.157.1 Macro Definition Documentation

13.157.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

13.158 ell_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_spmm_INL

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)

- template<class Field>
  void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)

13.158.1 Macro Definition Documentation

13.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmm_INL
```

13.159 ell_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.159.1 Macro Definition Documentation

13.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

13.160 ell_utils.inl File Reference

```
#include <vector>
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_utils_INL

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.160.1 Macro Definition Documentation

13.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

13.161 ell_simd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_spmv.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::ELL_simd >
- struct Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >

Namespaces

- namespace FFLAS

Functions

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)

13.162 ell_simd_pspmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.162.1 Macro Definition Documentation

13.162.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

13.163 ell_simd_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)

- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.163.1 Macro Definition Documentation

13.163.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

13.164 ell_simd_utils.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)
- template<class Field>
  void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.164.1 Macro Definition Documentation

13.164.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

13.165 hyb_zo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmm.inl"
```

Namespaces

- namespace FFLAS

13.166 hyb_zo_pspmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL

Functions

- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)

13.166.1 Macro Definition Documentation

13.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

13.167 hyb_zo_pspmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)

13.167.1 Macro Definition Documentation

13.167.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

13.168 hyb_zo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL

Functions

- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)

13.168.1 Macro Definition Documentation

13.168.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL
```

13.169 hyb_zo_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)

13.169.1 Macro Definition Documentation

13.169.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

13.170 hyb_zo_utils.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL

Functions

- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
- template<typename _Field>
  std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)

13.170.1 Macro Definition Documentation

13.170.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

13.171 read_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <fstream>
#include <string>
#include <cstdlib>
#include <iterator>
```

Data Structures

- struct Coo< Field >
- struct readMyMachineType< Field, T >
- struct readMyMachineType< Field, mpz_t >

Namespaces

- namespace FFLAS
- namespace FFLAS::details_spmv

Macros

- #define DNS_BIN_VER 0
- #define mask_t uint64_t

Functions

- template<class Field, bool sorted = true, bool read_integer = false>
  void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)
- template<class Field>
  void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)
- template<class T>
  std::enable_if< std::is_integral< T >::value, int > getDataType ()
- template<class T>
  std::enable_if< std::is_floating_point< T >::value, int > getDataType ()
- template<class T>
  std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()
- template<class T>
  int getDataType ()
- template<class Field>
  void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)
- template<class Field>
  void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)
- template<class Field>
  void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t ldA)

13.171.1 Macro Definition Documentation

13.171.1.1 DNS_BIN_VER

```
#define DNS_BIN_VER 0
```

13.171.1.2 mask_t

```
#define mask_t uint64_t
```

13.172 sell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_spmv.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::SELL >
- struct Sparse< _Field, SparseMatrix_t::SELL_ZO >

Namespaces

- namespace FFLAS

13.173 sell_pspmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL

Functions

- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.173.1 Macro Definition Documentation

13.173.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

13.174 sell_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_sell_spmv_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
- template<class Field>
  void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
- template<class Field>
  void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)

13.174.1 Macro Definition Documentation

13.174.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

13.175 sell_utils.inl File Reference

Data Structures

- struct Info
- struct Coo< ValT, IdxT >

Namespaces

- namespace FFLAS
- namespace FFLAS::sell_details

Macros

- #define __FFLASFFPACK_fflas_sparse_sell_utils_INL

Functions

- template<class Field>
  void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)
- template<class Field>
  void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)
- template<class Field>
  void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)
- template<class Field, class IndexT>
  void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.175.1 Macro Definition Documentation

13.175.1.1 __FFLASFFPACK_fflas_sparse_sell_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

13.176 sparse_matrix_traits.h File Reference

```
#include <type_traits>
```

Data Structures

- struct isSparseMatrix< Field, M >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >
- struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >
- struct isZOSparseMatrix< F, M >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >
- struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >
- struct isSparseMatrixSimdFormat< F, M >
- struct isSparseMatrixMKLFormat< F, M >

- struct tfn_plus
- struct tfn_mul
- struct tfn_mul_eq
- struct tfn_minus
- struct tfn_plus_eq
- struct tfn_minus_eq
- struct has_plus_impl< C >
- struct has_mul_impl< C >
- struct has_mul_eq_impl< C >
- struct has_plus_eq_impl< C >
- struct has_minus_eq_impl< C >
- struct has_minus_impl< C >
- struct has_operation< T >

Namespaces

- namespace FFLAS

Typedefs

- using ZOSparseMatrix = std::true_type
- using NotZOSparseMatrix = std::false_type
- using SimdSparseMatrix = std::true_type
- using NoSimdSparseMatrix = std::false_type
- using MKLSparseMatrixFormat = std::true_type
- using NotMKLSparseMatrixFormat = std::false_type
- template<class T>
  using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_plus_impl<T>>↩
  ::type
- template<class T>
  using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_minus_impl<T>>↩
  ::type
- template<class T>
  using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, std::is_copy_↩
  assignable<T>>::type
- template<class T>
  using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_plus_eq_impl<T>>↩
  ::type
- template<class T>
  using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_minus_eq_impl<T>>↩
  ::type
- template<class T>
  using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>>↩
  ::type
- template<class T>
  using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_eq_impl<T>>↩
  ::type

13.177 utils.h File Reference

```
#include <algorithm>
#include <numeric>
#include <vector>
```

Data Structures

- struct StatsMatrix

Namespaces

- namespace FFLAS

Functions

- template<class It>
  double computeDeviation (It begin, It end)
- template<class Field>
  StatsMatrix getStat (const Field &F, const index_t *row, const index_t *col, typename Field::ConstElement_ptr val, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

13.178 fflas_transpose.h File Reference

transpose the storage of the matrix (switch between row and col major mode)

```
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
```

Data Structures

- struct BlockTransposeSIMD< Field, Simd, >

Namespaces

- namespace FFLAS
- namespace FFLAS::_ftranspose_impl

Macros

- #define FFLAS_TRANSPOSE_BLOCKSIZE 32
- #define LD(i)
- #define ST(i)

Functions

- template<size_t bs, typename Field, typename BTSimd>
  void not_inplace (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<size_t bs, typename Field, typename BTSimd>
  void square_inplace (const Field &F, const BTSimd &BTS, const size_t m, typename Field::Element_ptr A, const size_t lda)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v1 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v2 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value >::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type * = nullptr>
  Field::Element_ptr ftranspose (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

13.178.1 Detailed Description

transpose the storage of the matrix (switch between row and col major mode)

13.178.2 Macro Definition Documentation

13.178.2.1 FFLAS_TRANSPOSE_BLOCKSIZE

```
#define FFLAS_TRANSPOSE_BLOCKSIZE 32
```

13.178.2.2 LD

```
#define LD(
              i)
```

Value:

```
R##i=Simd::loadu(A+lda*i)
```

13.178.2.3 ST

```
#define ST(
              i)
```

Value:

```
Simd::storeu(B+ldb*i,R##i)
```

13.179 ffpack.doxy File Reference

13.180 ffpack.h File Reference

Set of elimination based routines for dense linear algebra.

```
#include "givaro/givpoly1.h"
#include <fflas-ffpack/fflas-ffpack-config.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include <list>
#include <vector>
#include <iostream>
#include <algorithm>
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "ffpack_fgesv.inl"
#include "ffpack_fgetrs.inl"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
#include "ffpack_pluq.inl"
#include "ffpack_pluq_mp.inl"
#include "ffpack_ppluq.inl"
#include "ffpack_ludivine.inl"
#include "ffpack_ludivine_mp.inl"
#include "ffpack_echelonforms.inl"
#include "ffpack_fsytrf.inl"
#include "ffpack_invert.inl"
#include "ffpack_ftrtr.inl"
#include "ffpack_ftrstr.inl"
#include "ffpack_ftrssyr2k.inl"
#include "ffpack_charpoly_kglu.inl"
#include "ffpack_charpoly_kgfast.inl"
#include "ffpack_charpoly_kgfastgeneralized.inl"
#include "ffpack_charpoly_danilevski.inl"
#include "ffpack_charpoly.inl"
#include "ffpack_frobenius.inl"
```

```
#include "ffpack_minpoly.inl"
#include "ffpack_krylovelim.inl"
#include "ffpack_permutation.inl"
#include "ffpack_rankprofiles.inl"
#include "ffpack_det_mp.inl"
#include "ffpack_bruhatgen.inl"
#include "ffpack.inl"
```

Data Structures

- class CharpolyFailed

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_FTRSTR_THRESHOLD 64
- #define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64

Functions

- void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)

  Conversion of a permutation from LAPACK format to Math format.
- void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)

  Conversion of a permutation from Maths format to LAPACK format.
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field, class Cut, class Param>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field>
  void MonotonicApplyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t R)

  Apply a R-monotonically increasing permutation P, to the matrix A.
- template<class Field>
  void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)

  Solve the system $AX = B$ or $XA = B$.

- template<class Field>
  Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Solve the system A X = B or X A = B.
- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)

  Square system solver.
- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Rectangular system solver.
- template<class Field>
  void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)

  Compute the inverse of a triangular matrix.
- template<class Field>
  void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)
- template<class Field>
  void ftrtrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Compute the product of two triangular matrices of opposite shape.
- template<class Field>
  void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)

  Solve a triangular system with a triangular right hand side of the same shape.
- template<class Field>
  void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)

  Solve a triangular system in a symmetric sum: find B upper/lower triangular such that A^T B + B^T A = C where C is symmetric.
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)

  Triangular factorization of symmetric matrices.
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOL
- template<class Field, class Cut, class Param>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_
- template<class Field>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)

  Triangular factorization of symmetric matrices.
- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

Compute a PLUQ factorization of the given matrix.

- template<class Field>
  size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)
- template<class Field, class Cut, class Param>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)
- template<class Field>
  size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE↩_THRESHOLD)

  Compute the CUP or PLE factorization of the given matrix.
- template<class Field>
  size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FfpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)
- template<class Field>
  size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG Lu↩Tag=FfpackSlabRecursive)

  Compute the Column Echelon form of the input matrix in-place.
- template<class Field>
  size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↩Tag=FfpackSlabRecursive)

  Compute the Row Echelon form of the input matrix in-place.
- template<class Field>
  size_t pRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_↩LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_↩LU_TAG LuTag=FfpackSlabRecursive)

  Compute the Reduced Column Echelon form of the input matrix in-place.
- template<class Field>
  size_t pReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename

Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)

- template<class Field, class PSHelper>
  size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Compute the Reduced Row Echelon form of the input matrix in-place.
- template<class Field>
  size_t pReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class Field>
  size_t GaussJordan (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)

  Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.
- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, int &nullity)

  Invert the given matrix in place or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert2 (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.
- template<class PolRing>
  std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class PolRing>
  PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class PolRing>
  PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class Field, class Polynomial>
  std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

- template<class Field, class Polynomial>
  int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)
- template<class Field, class Polynomial>
  std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class Field>
  void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)
- template<class Field, class Polynomial, class RandIter>
  std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)
- template<class Field, class Polynomial>
  std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class PolRing>
  void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completed↩ Factors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_↩ t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)
- template<class PolRing>
  std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::↩ Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)
- template<class Field, class Polynomial>
  std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)
- template<class Field, class Polynomial>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial, class RandIter>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, RandIter &G)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr v, const size_t incv)

  Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, ..., A^N v)$.
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)
- template<class Field, class Polynomial>
  Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_↩ mb=0, const size_t kg_j=0)
- template<class Field>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Computes the rank of the given matrix using a PLUQ factorization.
- template<class Field>
  size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0)

- template<class Field, class PSHelper>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH)
- template<class Field>
  bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Returns true if the given matrix is singular.
- template<class Field>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P=NULL, size_t *Q=NULL)

  Returns the determinant of the given square matrix.
- template<class Field>
  Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)
- template<class Field, class PSHelper>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)
- template<class Field>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)

  Solves a linear system AX = b using PLUQ factorization.
- template<class Field, class PSHelper>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)
- template<class Field>
  Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)
- template<class Field>
  *void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)

  Solve L X = B or X L = B in place.
- template<class Field>
  size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &NS, size_t &ldn, size_t &NSdim)

  Computes a basis of the Left/Right nullspace of the matrix A.
- template<class Field>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the row rank profile of A.
- template<class Field>
  size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- template<class Field>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the column rank profile of A.
- template<class Field>
  size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)

- template<class Field, class PSHelper>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_↩ LU_TAG LuTag)

  Recovers the column/row rank profile from the permutation of an LU decomposition.
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)

  Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.
- template<class Field>
  size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  RowRankProfileSubmatrixIndices.
- template<class Field>
  size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  *Computes the indices of the submatrix r*r X of A whose columns correspond to the column rank profile of A.*
- template<class Field>
  size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  *Computes the r*r submatrix X of A, by picking the row rank profile rows of A.*
- template<class Field>
  size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.
- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)

  Extracts a triangular matrix from a compact storage A=L\U of rank R.
- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)

  Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by RowEchelonForm or Column↩ EchelonForm.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Cleans up a compact storage A=L\U obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.
- template<class Field>
  void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK↩ _LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.

- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG Lu↩ Tag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_↩ LU_TAG LuTag=FfpackSlabRecursive)

  Cleans up a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelon↩ Form with transform = true.

- template<class Field>
  void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.

- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

  Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

- template<class Field>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.

- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.

- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.

- size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.

- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size↩ _t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.

- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size↩ _t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.

- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.

- size_t * TInverter (size_t *T, size_t r)

- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, size_t *MU, size_t *ML)
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, const typename Field::Element_ptr Xu, size_t ldu, size_t NbBlocksU, size_t *Ku, size_t *Tu, size_t *MU, const typename Field::Element_ptr Xl, size_t ldl, size_t NbBlocksL, size_t *Kl, size_t *Tl, size_t *ML, typename Field::Element_ptr B, size_t t, size_t ldb, typename Field::Element_ptr C, size_t ldc)

  productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix
- template<class Field>
  Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)

  LQUPtoInverseOfFullRankMinor.

13.180.1 Detailed Description

Set of elimination based routines for dense linear algebra.
Matrices are supposed over finite prime field of characteristic less than 2^26.

13.180.2 Macro Definition Documentation

13.180.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

13.180.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

13.181 ffpack.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_INL

Functions

- template<class Field>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Computes the rank of the given matrix using a PLUQ factorization.
- template<class Field>
  size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0)
- template<class Field, class PSHelper>
  size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH)
- template<class Field>
  bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Returns true if the given matrix is singular.

- template<class Field>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P=NULL, size_t *Q=NULL)

  Returns the determinant of the given square matrix.
- template<class Field>
  Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)
- template<class Field, class PSHelper>
  Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)
- template<class Field>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)

  Solves a linear system AX = b using PLUQ factorization.
- template<class Field, class PSHelper>
  Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)
- template<class Field>
  Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)
- template<class Field>
  void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)

  Solve L X = B or X L = B in place.
- template<class Field>
  size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &NS, size_t &ldn, size_t &NSdim)

  Computes a basis of the Left/Right nullspace of the matrix A.
- template<class Field>
  void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)
- template<class Field>
  void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)

13.181.1 Macro Definition Documentation

13.181.1.1 __FFLASFFPACK_ffpack_INL

```
#define __FFLASFFPACK_ffpack_INL
```

13.182 ffpack_bruhatgen.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_bruhatgen_inl

Functions

- template<class Field>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.
- size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.
- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.
- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.
- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.
- size_t * TInverter (const size_t *T, size_t r)
- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, const size_t *MU, const size_t *ML)
- template<class Field>
  Field::Element_ptr expandLCRE (const Field &Fi, size_t N, size_t s, size_t r, size_t *R, size_t i, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tuinv, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tlinv, typename Field::Element_ptr CRE, size_t ldcre)

  Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

  Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.

13.182.1 Macro Definition Documentation

13.182.1.1 __FFLASFFPACK_ffpack_bruhatgen_inl

```
#define __FFLASFFPACK_ffpack_bruhatgen_inl
```

13.183 ffpack_charpoly.inl File Reference

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "ffpack_charpoly_mp.inl"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_charpoly_INL

Functions

- template<class PolRing>
  std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::↩ Element > &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_↩ t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class PolRing>
  PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

  Compute the characteristic polynomial of the matrix A.
- template<class Field, class Polynomial, class RandIter>
  std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)
- template<class Field, class Polynomial>
  std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)

13.183.1 Macro Definition Documentation

13.183.1.1 __FFLASFFPACK_charpoly_INL

```
#define __FFLASFFPACK_charpoly_INL
```

13.184 ffpack_charpoly_danilevski.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_charpoly_danilveski_INL

Functions

- template<class Field, class Polynomial>
  std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)

13.184.1 Macro Definition Documentation

13.184.1.1 __FFLASFFPACK_ffpack_charpoly_danilveski_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

13.185 ffpack_charpoly_kgfast.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_ffpack_charpoly_kgfast_INL

Functions

- template<class Field, class Polynomial>
  int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)
- template<class Field>
  void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)

13.185.1 Macro Definition Documentation

13.185.1.1 __FFLASFFPACK_ffpack_charpoly_kgfast_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

13.186 ffpack_charpoly_kgfastgeneralized.inl File Reference

```
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL

Functions

- template<class Field>
  Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)
- template<class Field, class Polynomial>
  std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)

13.186.1 Macro Definition Documentation

13.186.1.1 __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

13.187 ffpack_charpoly_kglu.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_ffpack_charpoly_kglu_INL

Functions

- template<class Field>
  size_t updateD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > > &minpt)
- template<class Field>
  size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)
- template<class Field, class Polynomial>
  std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

13.187.1 Macro Definition Documentation

13.187.1.1 __FFLASFFPACK_ffpack_charpoly_kglu_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

13.188 ffpack_charpoly_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "givaro/givpoly1.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFPACK_charpoly_mp_INL

Functions

- FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)
- template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)

13.188.1 Macro Definition Documentation

13.188.1.1 __FFPACK_charpoly_mp_INL

```
#define __FFPACK_charpoly_mp_INL
```

13.189 ffpack_det_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFPACK_det_mp_INL

Functions

- template<class PSHelper>
  FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)
- template<class PSHelper>
  Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)

13.189.1 Macro Definition Documentation

13.189.1.1 __FFPACK_det_mp_INL

```
#define __FFPACK_det_mp_INL
```

13.190 ffpack_echelonforms.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_echelon_forms_INL
- #define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256

Functions

- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)

  Extracts a triangular matrix from a compact storage A=L\U of rank R.
- template<class Field>
  void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)

  Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.
- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

  Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by RowEchelonForm or Column↩ EchelonForm.
- template<class Field>
  void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Cleans up a compact storage A=L\U obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.
- template<class Field>
  void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK↩ _LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.
- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG Lu↩ Tag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.
- template<class Field>
  void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_↩ LU_TAG LuTag=FfpackSlabRecursive)

Cleans up a compact storage A=L\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelon↩ Form with transform = true.

- template<class Field>
  void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by RowEchelon↩ Form or ColumnEchelonForm.

13.190.1 Macro Definition Documentation

13.190.1.1 __FFLASFFPACK_ffpack_echelon_forms_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

13.190.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

13.191 ffpack_fgesv.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_fgesv_INL

Functions

- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)

  Square system solver.
- template<class Field>
  size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Rectangular system solver.

13.191.1 Macro Definition Documentation

13.191.1.1 __FFLASFFPACK_ffpack_fgesv_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

13.192 ffpack_fgetrs.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_fgetrs_INL

Functions

- template<class Field>
  void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)

  Solve the system $AX = B$ or $XA = B$.
- template<class Field>
  Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Solve the system A X = B or X A = B.

13.192.1 Macro Definition Documentation

13.192.1.1 __FFLASFFPACK_ffpack_fgetrs_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

13.193 ffpack_frobenius.inl File Reference

```
#include <givaro/givranditer.h>
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Functions

- template<class Field>
  void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)
- template<class Field>
  void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)
- template<class Field>
  void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class Field>
  void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)
- template<class PolRing>
  void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completed↩ Factors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_↩ t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

- template<class PolRing>
  std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::↩ Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)

13.194 ffpack_fsytrf.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_fsytrf_INL

Functions

- template<class Field>
  bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)
- template<class Field>
  size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)
- template<class Field>
  size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)
- template<class Field>
  size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)
- template<class Field>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Sequential seq, size_t threshold)
- template<class Field, class Cut, class Param>
  bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)

  Triangular factorization of symmetric matrices.
- template<class Field>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOL
- template<class Field, class Cut, class Param>
  bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_

- template<class Field>
  size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)
- template<class Field>
  void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)

13.194.1 Macro Definition Documentation

13.194.1.1 __FFLASFFPACK_ffpack_fsytrf_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

13.195 ffpack_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_ftrssyr2k_INL

Functions

- template<class Field>
  void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)

  Solve a triangular system in a symmetric sum: find B upper/lower triangular such that A^T B + B^T A = C where C is symmetric.

13.195.1 Macro Definition Documentation

13.195.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

13.196 ffpack_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_ftrstr_INL

Functions

- template<class Field>
  void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)

  Solve a triangular system with a triangular right hand side of the same shape.

13.196.1 Macro Definition Documentation

13.196.1.1 __FFLASFFPACK_ffpack_ftrstr_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

13.197 ffpack_ftrtr.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define ENABLE_ALL_CHECKINGS 1
- #define __FFLASFFPACK_ffpack_ftrtr_INL

Functions

- template<class Field>
  void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)

  Compute the inverse of a triangular matrix.
- template<class Field>
  void ftrtrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)

  Compute the product of two triangular matrices of opposite shape.
- template<class Field>
  void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)

13.197.1 Macro Definition Documentation

13.197.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.197.1.2 __FFLASFFPACK_ffpack_ftrtr_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

13.198 ffpack_invert.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_invert_INL

Functions

- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, int &nullity)

  Invert the given matrix in place or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert (const Field &F, const size_t M, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.
- template<class Field>
  Field::Element_ptr Invert2 (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, int &nullity)

  Invert the given matrix or computes its nullity if it is singular.

13.198.1 Macro Definition Documentation

13.198.1.1 __FFLASFFPACK_ffpack_invert_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

13.199 ffpack_krylovelim.inl File Reference

Macros

- #define __FFLASFFPACK_ffpack_krylovelim_INL

13.199.1 Macro Definition Documentation

13.199.1.1 __FFLASFFPACK_ffpack_krylovelim_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

13.200 ffpack_ludivine.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

Data Structures

- class callLUdivine_small< Element >
- class callLUdivine_small< double >
- class callLUdivine_small< float >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_ffpack_ludivine_INL

Functions

- template<class Field>
  size_t LUdivine_gauss (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- template<class Field>
  size_t LUdivine_small (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
- template<class Field>
  size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)
- template<class Field>
  size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK_MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)

13.200.1 Macro Definition Documentation

13.200.1.1 __FFLASFFPACK_ffpack_ludivine_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

13.201 ffpack_ludivine_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack_ludivine.inl"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFPACK_ludivine_mp_INL

Functions

- template<> size_t LUdivine (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)

13.201.1 Macro Definition Documentation

13.201.1.1 __FFPACK_ludivine_mp_INL

```
#define __FFPACK_ludivine_mp_INL
```

13.202 ffpack_minpoly.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFPACK::Protected

Macros

- #define __FFLASFFPACK_ffpack_minpoly_INL

Functions

- template<class Field, class Polynomial>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial, class RandIter>
  Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, RandIter &G)

  Compute the minimal polynomial of the matrix A.
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr v, const size_t incv)

  Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, ..., A^N v)$.
- template<class Field, class Polynomial>
  Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)
- template<class Field, class Polynomial>
  Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_↩ mb=0, const size_t kg_j=0)

13.202.1 Macro Definition Documentation

13.202.1.1 __FFLASFFPACK_ffpack_minpoly_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

13.203 ffpack_permutation.inl File Reference

```
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_permutation_INL
- #define FFLASFFPACK_PERM_BKSIZE 32

Functions

- template<class Field>
  void MonotonicApplyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t R)

  Apply a R-monotonically increasing permutation P, to the matrix A.
- template<class Field>
  void MonotonicCompress (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)
- template<class Field>
  void MonotonicCompressMorePivots (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t rowstomove, const size_t lenP)
- template<class Field>
  void MonotonicCompressCycles (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t lenP)
- template<class Field>
  void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)
- template<class Field>
  void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field>
  void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field, class Cut, class Param>
  void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class T>
  void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- template<class Field>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)

- template<class Field, class Cut, class Param>
  void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class T>
  void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)

  Conversion of a permutation from LAPACK format to Math format.
- void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)

  Conversion of a permutation from Maths format to LAPACK format.
- void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.
- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in MathP as a MathPermutation format.
- void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)

  Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.
- void cyclic_shift_mathPerm (size_t *P, const size_t s)
- template<class Field>
  void cyclic_shift_row_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  void cyclic_shift_row (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<typename T>
  void cyclic_shift_row (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  void cyclic_shift_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)
- template<typename T>
  void cyclic_shift_col (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class Field>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.
- template<class Field, class Cut, class Param>
  void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)

13.203.1 Macro Definition Documentation

13.203.1.1 __FFLASFFPACK_ffpack_permutation_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

13.203.1.2 FFLASFFPACK_PERM_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

13.204 ffpack_pluq.inl File Reference

Namespaces

- namespace FFPACK

  Finite Field PACK Set of elimination based routines for dense linear algebra.

Macros

- #define __FFLASFFPACK_ffpack_pluq_INL
- #define CROUT

Functions

- template<class Field>
  size_t PLUQ_basecaseV3 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t PLUQ_basecaseCrout (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)
- template<class Field>
  size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)
- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)
- template<class Field>
  size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  Compute a PLUQ factorization of the given matrix.

13.204.1 Macro Definition Documentation

13.204.1.1 __FFLASFFPACK_ffpack_pluq_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

13.204.1.2 CROUT

```
#define CROUT
```

13.205 ffpack_pluq_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "givaro/givinteger.h"
#include "givaro/modular-integer.h"
```

Namespaces

- namespace FFPACK

  Finite Field PACK Set of elimination based routines for dense linear algebra.

Macros

- #define __FFPACK_pluq_mp_INL

Functions

- template<class Cut, class Param>
  size_t PLUQ (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)

13.205.1 Macro Definition Documentation

13.205.1.1 __FFPACK_pluq_mp_INL

```
#define __FFPACK_pluq_mp_INL
```

13.206 ffpack_ppluq.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_ppluq_INL
- #define __FFLAS__TRSM_READONLY
- #define PBASECASE_K 256

Functions

- template<class Field>
  void threads_fgemm (const size_t m, const size_t n, const size_t r, int nbthreads, size_t *W1, size_t *W2, size_t *W3, size_t gamma)
- template<class Field>
  void threads_ftrsm (const size_t m, const size_t n, int nbthreads, size_t *t1, size_t *t2)
- template<class Field>
  size_t PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &PSHelper)
- template<class Field>
  size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

13.206.1 Macro Definition Documentation

13.206.1.1 __FFLASFFPACK_ffpack_ppluq_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

13.206.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.206.1.3 PBASECASE_K

```
#define PBASECASE_K 256
```

13.207 ffpack_rankprofiles.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_rank_profiles_INL

Functions

- template<class Field>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the row rank profile of A.
- template<class Field>
  size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- template<class Field>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)

  Computes the column rank profile of A.
- template<class Field>
  size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)
- template<class Field, class PSHelper>
  size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_↩
  LU_TAG LuTag)

  Recovers the column/row rank profile from the permutation of an LU decomposition.
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)

  Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.
- template<class Field>
  size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  RowRankProfileSubmatrixIndices.
- template<class Field>
  size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)

  *Computes the indices of the submatrix r*r X of A whose columns correspond to the column rank profile of A.*
- template<class Field>
  size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  *Computes the r*r submatrix X of A, by picking the row rank profile rows of A.*
- template<class Field>
  size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)

  Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.

- template<class Field>
  Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)

  LQUPtoInverseOfFullRankMinor.

13.207.1 Macro Definition Documentation

13.207.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

13.208 field-traits.h File Reference

Field Traits.

```
#include <type_traits>
#include "fflas-ffpack/field/rns-double-elt.h"
#include "recint/rmint.h"
#include "givaro/modular-general.h"
#include "givaro/zring.h"
```

Data Structures

- struct GenericTag

  generic ring.
- struct ModularTag

  This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
- struct UnparametricTag

  If the field uses a representation with infix operators.
- struct DefaultTag

  No specific mode of action: use standard field operations.
- struct DefaultBoundedTag

  Use standard field operations, but keeps track of bounds on input and output.
- struct ConvertTo< T >

  Force conversion to appropriate element type of ElementCategory T.
- struct DelayedTag

  Performs field operations with delayed mod reductions. Ensures result is reduced.
- struct LazyTag

  Performs field operations with delayed mod only when necessary. Result may not be reduced.
- struct GenericTag

  default is generic
- struct MachineFloatTag

  float or double
- struct MachineIntTag

  short, int, long, long long, and unsigned variants
- struct FixedPrecIntTag

  Fixed precision integers above machine precision: Givaro::recInt.
- struct ArbitraryPrecIntTag

  Arbitrary precision integers: GMP.
- struct RNSElementTag

  Representation in a Residue Number System.
- struct ElementTraits< Element >

ElementTraits.

- struct ElementTraits< float >
- struct ElementTraits< double >
- struct ElementTraits< int8_t >
- struct ElementTraits< int16_t >
- struct ElementTraits< int32_t >
- struct ElementTraits< int64_t >
- struct ElementTraits< uint8_t >
- struct ElementTraits< uint16_t >
- struct ElementTraits< uint32_t >
- struct ElementTraits< uint64_t >
- struct ElementTraits< Givaro::Integer >
- struct ElementTraits< RecInt::rint< K > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< RecInt::rmint< K, MG > >
- struct ElementTraits< FFPACK::rns_double_elt >
- struct ModeTraits< Field >

ModeTraits.

- struct ModeTraits< Givaro::Modular< Element, Compute > >
- struct ModeTraits< Givaro::Modular< int64_t, uint64_t > >
- struct ModeTraits< Givaro::Modular< int8_t, Compute > >
- struct ModeTraits< Givaro::Modular< int16_t, Compute > >
- struct ModeTraits< Givaro::Modular< int32_t, Compute > >
- struct ModeTraits< Givaro::Modular< uint8_t, Compute > >
- struct ModeTraits< Givaro::Modular< uint16_t, Compute > >
- struct ModeTraits< Givaro::Modular< uint32_t, Compute > >
- struct ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >
- struct ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >
- struct ModeTraits< Givaro::ModularBalanced< Element > >
- struct ModeTraits< Givaro::ModularBalanced< int8_t > >
- struct ModeTraits< Givaro::ModularBalanced< int16_t > >
- struct ModeTraits< Givaro::ModularBalanced< int32_t > >
- struct ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >
- struct ModeTraits< Givaro::ZRing< Givaro::Integer > >
- struct ModeTraits< Givaro::ZRing< float > >
- struct ModeTraits< Givaro::ZRing< double > >
- struct ModeTraits< Givaro::Montgomery< T > >
- struct FieldTraits< Field >

FieldTrait.

- struct FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >
- struct FieldTraits< Givaro::Modular< Element > >
- struct FieldTraits< Givaro::ModularBalanced< Element > >
- struct FieldTraits< Givaro::ZRing< double > >
- struct FieldTraits< Givaro::ZRing< float > >
- struct FieldTraits< Givaro::ZRing< int16_t > >
- struct FieldTraits< Givaro::ZRing< uint16_t > >
- struct FieldTraits< Givaro::ZRing< int32_t > >
- struct FieldTraits< Givaro::ZRing< uint32_t > >
- struct FieldTraits< Givaro::ZRing< int64_t > >
- struct FieldTraits< Givaro::ZRing< uint64_t > >
- struct FieldTraits< Givaro::ZRing< Givaro::Integer > >
- struct FieldTraits< FFPACK::RNSInteger< T > >
- struct FieldTraits< FFPACK::RNSIntegerMod< T > >
- struct associatedDelayedField< Field >
- struct associatedDelayedField< const Givaro::Modular< T, X > >
- struct associatedDelayedField< const Givaro::ModularBalanced< T > >
- struct associatedDelayedField< const Givaro::ZRing< T > >
- struct associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >

Namespaces

- namespace RecInt
- namespace Givaro
- namespace FFPACK

  Finite Field PACK Set of elimination based routines for dense linear algebra.
- namespace FFLAS
- namespace FFLAS::FieldCategories

  Traits and categories will need to be placed in a proper file later.
- namespace FFLAS::ModeCategories

  Specifies the mode of action for an algorithm w.r.t.
- namespace FFLAS::ElementCategories

Functions

- template<class Field, class enable = void>
  Field::Residu_t maxCardinality ()
- template<> uint64_t maxCardinality< Givaro::Modular< int64_t > > ()
- template<> uint32_t maxCardinality< Givaro::Modular< int32_t > > ()
- template<class Field>
  Field::Residu_t minCardinality ()

13.208.1 Detailed Description

Field Traits.

13.209 field.doxy File Reference

13.210 rns-double-elt.h File Reference

rns elt structure with double support

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/cast.h"
```

Data Structures

- struct rns_double_elt
- struct rns_double_elt_ptr
- struct rns_double_elt_cstptr

Namespaces

- namespace FFPACK

  Finite Field PACK Set of elimination based routines for dense linear algebra.

Functions

- template<> rns_double_elt_ptr fflas_const_cast (rns_double_elt_cstptr x)
- template<> rns_double_elt_cstptr fflas_const_cast (rns_double_elt_ptr x)

13.210.1 Detailed Description

rns elt structure with double support

13.211 rns-double-recint.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_field_rns_double_recint_INL

13.211.1 Macro Definition Documentation

13.211.1.1 __FFLASFFPACK_field_rns_double_recint_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

13.212 rns-double.h File Reference

rns structure with double support

```
#include <iterator>
#include <vector>
#include <givaro/modular-floating.h>
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include "givaro/modular-extended.h"
#include <recint/ruint.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/field/rns-double-elt.h"
#include "rns-double.inl"
#include "rns-double-recint.inl"
```

Data Structures

- struct rns_double
- struct rns_double_extended
- class rnsRandIter< RNS >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFLAS

Macros

- #define ROUND_DOWN(x, s)

Functions

- template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)
- template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)

13.212.1 Detailed Description

rns structure with double support

13.212.2 Macro Definition Documentation

13.212.2.1 ROUND_DOWN

```
#define ROUND_DOWN(
              x,
              s)
```

Value:

```
((x) & ~((s)-1))
```

13.213 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_field_rns_double_INL

13.213.1 Macro Definition Documentation

13.213.1.1 __FFLASFFPACK_field_rns_double_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

13.214 rns-integer-mod.h File Reference

representation of Z/pZ using RNS representation (note: fixed precision)

```
#include <vector>
#include <cmath>
#include <recint/recint.h>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include <givaro/udl.h>
#include "givaro/modular-extended.h"
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas_level1.inl"
#include "fflas-ffpack/fflas/fflas_level2.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/fflas/fflas_fscal_mp.inl"
```

Data Structures

- class RNSIntegerMod< RNS >
- class RNSIntegerMod< RNS >::RandIter

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFLAS

Functions

- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const Alignment align)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)
- template<typename RNS>
  void finit_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)
- template<typename RNS>
  void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)
- template<typename RNS>
  void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::↩ Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)
- template<typename RNS>
  void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)

13.214.1 Detailed Description

representation of Z/pZ using RNS representation (note: fixed precision)

13.215 rns-integer.h File Reference

representation of Z using RNS representation (note: fixed precision)

```
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-double.h"
```

Data Structures

- class RNSInteger< RNS >
- class RNSInteger< RNS >::RandIter

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*
- namespace FFLAS

Functions

- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const Alignment align)
- template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)
- template<typename RNS>
  void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)
- template<typename RNS>
  void fconvert_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)

13.215.1 Detailed Description

representation of Z using RNS representation (note: fixed precision)

13.216 rns.h File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

13.217 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

Macros

- #define __FFLASFFPACK_field_rns_INL

13.217.1 Macro Definition Documentation

13.217.1.1 __FFLASFFPACK_field_rns_INL

```
#define __FFLASFFPACK_field_rns_INL
```

13.218 interfaces.doxy File Reference

13.219 fflas_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

Macros

- #define FFLAS_COMPILED

Enumerations

- enum FFLAS_C_ORDER { FflasRowMajor =101 , FflasColMajor =102 }

  Storage by row or col ?
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }

  Is matrix transposed ?
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 }

  Is triangular matrix's shape upper ?
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }

  Is the triangular matrix implicitly unit diagonal ?
- enum FFLAS_C_SIDE { FflasLeft = 141 , FflasRight = 142 }

  On what side ?
- enum FFLAS_C_BASE { FflasDouble = 151 , FflasFloat = 152 , FflasGeneric = 153 }

  FFLAS_C_BASE determines the type of the element representation for Matrix Mult kernel.

Functions

- void freducein_1_modular_double (const double p, const size_t n, double ∗X, const size_t incX, bool positive)
- void freduce_1_modular_double (const double F, const size_t n, const double ∗Y, const size_t incY, double ∗X, const size_t incX, bool positive)
- void fnegin_1_modular_double (const double F, const size_t n, double ∗X, const size_t incX, bool positive)
- void fneg_1_modular_double (const double p, const size_t n, const double ∗Y, const size_t incY, double ∗X, const size_t incX, bool positive)
- void fzero_1_modular_double (const double p, const size_t n, double ∗X, const size_t incX, bool positive)
- bool fiszero_1_modular_double (const double p, const size_t n, const double ∗X, const size_t incX, bool positive)
- bool fequal_1_modular_double (const double p, const size_t n, const double ∗X, const size_t incX, const double ∗Y, const size_t incY, bool positive)
- void fassign_1_modular_double (const double p, const size_t n, const double ∗Y, const size_t incY, double ∗X, const size_t incX, bool positive)
- void fscalin_1_modular_double (const double p, const size_t n, const double alpha, double ∗X, const size_t incX, bool positive)
- void fscal_1_modular_double (const double p, const size_t n, const double alpha, const double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- void faxpy_1_modular_double (const double p, const size_t n, const double alpha, const double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- double fdot_1_modular_double (const double p, const size_t n, const double ∗X, const size_t incX, const double ∗Y, const size_t incY, bool positive)
- void fswap_1_modular_double (const double p, const size_t n, double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- void fadd_1_modular_double (const double p, const size_t n, const double ∗A, const size_t incA, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void fsub_1_modular_double (const double p, const size_t n, const double ∗A, const size_t incA, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void faddin_1_modular_double (const double p, const size_t n, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void fsubin_1_modular_double (const double p, const size_t n, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void fassign_2_modular_double (const double p, const size_t m, const size_t n, const double ∗B, const size_t ldB, double ∗A, const size_t ldA, bool positive)
- void fzero_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t ldA, bool positive)
- bool fequal_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t ldA, const double ∗B, const size_t ldB, bool positive)
- bool fiszero_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t ldA, bool positive)
- void fidentity_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t ldA, const double d, bool positive)
- void freducein_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t ldA, bool positive)
- void freduce_2_modular_double (const double p, const size_t m, const size_t n, const double ∗B, const size_t ldB, double ∗A, const size_t ldA, bool positive)
- void fnegin_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t ldA, bool positive)
- void fneg_2_modular_double (const double p, const size_t m, const size_t n, const double ∗B, const size_t ldB, double ∗A, const size_t ldA, bool positive)
- void fscalin_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, double ∗A, const size_t ldA, bool positive)
- void fscal_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double ∗A, const size_t ldA, double ∗B, const size_t ldB, bool positive)
- void faxpy_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double ∗X, const size_t ldX, double ∗Y, const size_t ldY, bool positive)

- void fmove_2_modular_double (const double p, const size_t m, const size_t n, double *A, const size_t ldA, double *B, const size_t ldB, bool positive)
- void fadd_2_modular_double (const double p, const size_t m, const size_t n, const double *A, const size_t ldA, const double *B, const size_t ldB, double *C, const size_t ldC, bool positive)
- void fsub_2_modular_double (const double p, const size_t m, const size_t n, const double *A, const size_t ldA, const double *B, const size_t ldB, double *C, const size_t ldC, bool positive)
- void fsubin_2_modular_double (const double p, const size_t m, const size_t n, const double *B, const size_t ldB, double *C, const size_t ldC, bool positive)
- void faddin_2_modular_double (const double p, const size_t m, const size_t n, const double *B, const size_t ldB, double *C, const size_t ldC, bool positive)
- double * fgemv_2_modular_double (const double p, const enum FFLAS_C_TRANSPOSE TransA, const size_t m, const size_t n, const double alpha, const double *A, const size_t ldA, const double *X, const size_t incX, const double betA, double *Y, const size_t incY, bool positive)
- void fger_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double *x, const size_t incX, const double *y, const size_t incY, double *A, const size_t ldA, bool positive)
- void ftrsv_2_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE TransA, const enum FFLAS_C_DIAG Diag, const size_t n, const double *A, const size_t ldA, double *X, int incX, bool positive)
- void ftrsm_3_modular_double (const double p, const enum FFLAS_C_SIDE Side, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE TransA, const enum FFLAS_C_DIAG Diag, const size_t m, const size_t n, const double alpha, const double *A, const size_t ldA, double *B, const size_t ldB, bool positive)
- void ftrmm_3_modular_double (const double p, const enum FFLAS_C_SIDE Side, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE TransA, const enum FFLAS_C_DIAG Diag, const size_t m, const size_t n, const double alpha, double *A, const size_t ldA, double *B, const size_t ldB, bool positive)
- double * fgemm_3_modular_double (const double p, const enum FFLAS_C_TRANSPOSE tA, const enum FFLAS_C_TRANSPOSE tB, const size_t m, const size_t n, const size_t k, const double alpha, const double *A, const size_t ldA, const double *B, const size_t ldB, const double betA, double *C, const size_t ldC, bool positive)
- double * fsquare_3_modular_double (const double p, const enum FFLAS_C_TRANSPOSE tA, const size_t n, const double alpha, const double *A, const size_t ldA, const double betA, double *C, const size_t ldC, bool positive)

13.219.1 Macro Definition Documentation

13.219.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.219.2 Enumeration Type Documentation

13.219.2.1 FFLAS_C_ORDER

enum FFLAS_C_ORDER
Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

13.219.2.2 FFLAS_C_TRANSPOSE

enum FFLAS_C_TRANSPOSE
Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

13.219.2.3 FFLAS_C_UPLO

`enum FFLAS_C_UPLO`
Is triangular matrix's shape upper ?

Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$)
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$)

13.219.2.4 FFLAS_C_DIAG

`enum FFLAS_C_DIAG`
Is the triangular matrix implicitly unit diagonal ?

Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ($T_{i,i} = 1$)

13.219.2.5 FFLAS_C_SIDE

`enum FFLAS_C_SIDE`
On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the rigth.

13.219.2.6 FFLAS_C_BASE

`enum FFLAS_C_BASE`
`FFLAS_C_BASE` determines the type of the element representation for Matrix Mult kernel. (deprecated, should not be used)

Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

13.219.3 Function Documentation

13.219.3.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
             const double p,
             const size_t n,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.2 freduce_1_modular_double()

```
void freduce_1_modular_double (
             const double F,
             const size_t n,
             const double * Y,
             const size_t incY,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
             const double F,
             const size_t n,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.4 fneg_1_modular_double()

```
void fneg_1_modular_double (
             const double p,
             const size_t n,
             const double * Y,
             const size_t incY,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
             const double p,
             const size_t n,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.6 fiszero_1_modular_double()

```
bool fiszero_1_modular_double (
             const double p,
             const size_t n,
             const double * X,
             const size_t incX,
             bool positive)
```

13.219.3.7 fequal_1_modular_double()

```
bool fequal_1_modular_double (
             const double p,
             const size_t n,
             const double * X,
             const size_t incX,
             const double * Y,
             const size_t incY,
             bool positive)
```

13.219.3.8 fassign_1_modular_double()

```
void fassign_1_modular_double (
             const double p,
             const size_t n,
             const double * Y,
             const size_t incY,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
             const double p,
             const size_t n,
             const double alpha,
             double * X,
             const size_t incX,
             bool positive)
```

13.219.3.10 fscal_1_modular_double()

```
void fscal_1_modular_double (
             const double p,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t incX,
             double * Y,
             const size_t incY,
             bool positive)
```

13.219.3.11 faxpy_1_modular_double()

```
void faxpy_1_modular_double (
             const double p,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t incX,
             double * Y,
             const size_t incY,
             bool positive)
```

13.219.3.12 fdot_1_modular_double()

```
double fdot_1_modular_double (
             const double p,
```

```
            const size_t n,
            const double * X,
            const size_t incX,
            const double * Y,
            const size_t incY,
            bool positive)
```

13.219.3.13 fswap_1_modular_double()

```
void fswap_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            double * Y,
            const size_t incY,
            bool positive)
```

13.219.3.14 fadd_1_modular_double()

```
void fadd_1_modular_double (
            const double p,
            const size_t n,
            const double * A,
            const size_t incA,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.219.3.15 fsub_1_modular_double()

```
void fsub_1_modular_double (
            const double p,
            const size_t n,
            const double * A,
            const size_t incA,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.219.3.16 faddin_1_modular_double()

```
void faddin_1_modular_double (
            const double p,
            const size_t n,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.219.3.17 fsubin_1_modular_double()

```
void fsubin_1_modular_double (
            const double p,
```

```
            const size_t n,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.219.3.18 fassign_2_modular_double()

```
void fassign_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldB,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.19 fzero_2_modular_double()

```
void fzero_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.20 fequal_2_modular_double()

```
bool fequal_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t ldA,
            const double * B,
            const size_t ldB,
            bool positive)
```

13.219.3.21 fiszero_2_modular_double()

```
bool fiszero_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.22 fidentity_2_modular_double()

```
void fidentity_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t ldA,
```

```
            const double d,
            bool positive)
```

13.219.3.23 freducein_2_modular_double()

```
void freducein_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.24 freduce_2_modular_double()

```
void freduce_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldB,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.25 fnegin_2_modular_double()

```
void fnegin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.26 fneg_2_modular_double()

```
void fneg_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldB,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.27 fscalin_2_modular_double()

```
void fscalin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double alpha,
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.28 fscal_2_modular_double()

```
void fscal_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double alpha,
             const double * A,
             const size_t ldA,
             double * B,
             const size_t ldB,
             bool positive)
```

13.219.3.29 faxpy_2_modular_double()

```
void faxpy_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t ldX,
             double * Y,
             const size_t ldY,
             bool positive)
```

13.219.3.30 fmove_2_modular_double()

```
void fmove_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             double * A,
             const size_t ldA,
             double * B,
             const size_t ldB,
             bool positive)
```

13.219.3.31 fadd_2_modular_double()

```
void fadd_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double * A,
             const size_t ldA,
             const double * B,
             const size_t ldB,
             double * C,
             const size_t ldC,
             bool positive)
```

13.219.3.32 fsub_2_modular_double()

```
void fsub_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double * A,
```

```
            const size_t ldA,
            const double * B,
            const size_t ldB,
            double * C,
            const size_t ldC,
            bool positive)
```

13.219.3.33 fsubin_2_modular_double()

```
void fsubin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldB,
            double * C,
            const size_t ldC,
            bool positive)
```

13.219.3.34 faddin_2_modular_double()

```
void faddin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldB,
            double * C,
            const size_t ldC,
            bool positive)
```

13.219.3.35 fgemv_2_modular_double()

```
double * fgemv_2_modular_double (
            const double p,
            const enum FFLAS_C_TRANSPOSE TransA,
            const size_t m,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t ldA,
            const double * X,
            const size_t incX,
            const double betA,
            double * Y,
            const size_t incY,
            bool positive)
```

13.219.3.36 fger_2_modular_double()

```
void fger_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double alpha,
            const double * x,
            const size_t incX,
            const double * y,
            const size_t incY,
```

```
            double * A,
            const size_t ldA,
            bool positive)
```

13.219.3.37 ftrsv_2_modular_double()

```
void ftrsv_2_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_TRANSPOSE TransA,
            const enum FFLAS_C_DIAG Diag,
            const size_t n,
            const double * A,
            const size_t ldA,
            double * X,
            int incX,
            bool positive)
```

13.219.3.38 ftrsm_3_modular_double()

```
void ftrsm_3_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_TRANSPOSE TransA,
            const enum FFLAS_C_DIAG Diag,
            const size_t m,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t ldA,
            double * B,
            const size_t ldB,
            bool positive)
```

13.219.3.39 ftrmm_3_modular_double()

```
void ftrmm_3_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_TRANSPOSE TransA,
            const enum FFLAS_C_DIAG Diag,
            const size_t m,
            const size_t n,
            const double alpha,
            double * A,
            const size_t ldA,
            double * B,
            const size_t ldB,
            bool positive)
```

13.219.3.40 fgemm_3_modular_double()

```
double * fgemm_3_modular_double (
            const double p,
            const enum FFLAS_C_TRANSPOSE tA,
            const enum FFLAS_C_TRANSPOSE tB,
            const size_t m,
```

```
            const size_t n,
            const size_t k,
            const double alpha,
            const double * A,
            const size_t ldA,
            const double * B,
            const size_t ldB,
            const double betA,
            double * C,
            const size_t ldC,
            bool positive)
```

13.219.3.41 fsquare_3_modular_double()

```
double * fsquare_3_modular_double (
            const double p,
            const enum FFLAS_C_TRANSPOSE tA,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t ldA,
            const double betA,
            double * C,
            const size_t ldC,
            bool positive)
```

13.220 fflas_L1_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

Macros

- #define __FFLAS_L1_INST_C
- #define INST_OR_DECL
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.220.1 Macro Definition Documentation

13.220.1.1 __FFLAS_L1_INST_C

```
#define __FFLAS_L1_INST_C
```

13.220.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.220.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.220.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.220.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.220.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.220.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.220.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.220.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.220.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.221 fflas_L1_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

Macros

- #define INST_OR_DECL <>
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.221.1 Macro Definition Documentation

13.221.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.221.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.221.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.221.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.221.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.221.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.221.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.221.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.221.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.222 fflas_L1_inst_implem.inl File Reference

Namespaces

- namespace FFLAS

Functions

- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  freduce $x \leftarrow xmodF$.
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  freduce $x \leftarrow ymodF$.
- template INST_OR_DECL void finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  finit $x \leftarrow ymodF$.
- template INST_OR_DECL void fconvert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  fconvert $x \leftarrow ymodF$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  fnegin $x \leftarrow -x$.
- template INST_OR_DECL void fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  fneg $x \leftarrow -y$.

- template INST_OR_DECL void fzero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  fzero : $A \leftarrow 0$.
- template INST_OR_DECL bool fiszero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX)

  fiszero : test $X = 0$.
- template INST_OR_DECL bool fequal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  fequal : test $X = Y$.
- template INST_OR_DECL void fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

  fassign : $x \leftarrow y$.
- template INST_OR_DECL void fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- template INST_OR_DECL void fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  fscal $y \leftarrow \alpha \cdot x$.
- template INST_OR_DECL void faxpy (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template INST_OR_DECL FFLAS_ELT fdot (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template INST_OR_DECL void fswap (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

  fswap: $X \leftrightarrow Y$.
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void fsub (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void faddin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *A, const size_t inca, const FFLAS_ELT alpha, const FFLAS_ELT *B, const size_t incb, FFLAS_ELT *C, const size_t incc)

13.223 fflas_L2_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

Macros

- #define __FFLAS_L2_INST_C
- #define INST_OR_DECL
- #define FFLAS_FIELD Givaro::ModularBalanced

- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.223.1 Macro Definition Documentation

13.223.1.1 __FFLAS_L2_INST_C

```
#define __FFLAS_L2_INST_C
```

13.223.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.223.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.223.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.223.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.223.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.223.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.223.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.223.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.223.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.224 fflas_L2_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

Macros

- #define INST_OR_DECL <>
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.224.1 Macro Definition Documentation

13.224.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.224.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.224.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.224.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.224.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.224.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.224.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.224.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.224.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.225 fflas_L2_inst_implem.inl File Reference

Namespaces

- namespace FFLAS

Functions

- template INST_OR_DECL void fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  fassign : $A \leftarrow B$.
- template INST_OR_DECL void fzero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  fzero : $A \leftarrow 0$.
- template INST_OR_DECL bool fequal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb)

  fequal : test $A = B$.
- template INST_OR_DECL bool fiszero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *A, const size_t lda)

  fiszero : test $A = 0$.
- template INST_OR_DECL void fidentity (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda, const FFLAS_ELT &d)

  creates a diagonal matrix
- template INST_OR_DECL void fidentity (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  creates a diagonal matrix
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  freduce $A \leftarrow A mod F$.
- template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  freduce $A \leftarrow B mod F$.
- template INST_OR_DECL void finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  finit $A \leftarrow B mod F$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda)

  fnegin $A \leftarrow -A$.
- template INST_OR_DECL void fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *A, const size_t lda)

  fneg $A \leftarrow -B$.
- template INST_OR_DECL void fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *A, const size_t lda)

  fscalin $A \leftarrow a \cdot A$.
- template INST_OR_DECL void fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  fscal $B \leftarrow a \cdot A$.
- template INST_OR_DECL void faxpy (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t ldx, FFLAS_ELT *Y, const size_t ldy)

  faxpy : $y \leftarrow \alpha \cdot x + y$.
- template INST_OR_DECL void fmove (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t m, const size_t n, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fadd : matrix addition.
- template INST_OR_DECL void fsub (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

fsub : matrix subtraction.

- template INST_OR_DECL void fsubin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fsubin C = C - B

- template INST_OR_DECL void fadd (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT alpha, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  fadd : matrix addition with scaling.

- template INST_OR_DECL void faddin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT *B, const size_t ldb, FFLAS_ELT *C, const size_t ldc)

  faddin

- template INST_OR_DECL FFLAS_ELT * fgemv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT beta, FFLAS_ELT *Y, const size_t incY)

  finite prime FFLAS_FIELD<FFLAS_ELT> GEneral Matrix Vector multiplication.

- template INST_OR_DECL void fger (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *x, const size_t incx, const FFLAS_ELT *y, const size_t incy, FFLAS_ELT *A, const size_t lda)

  fger: rank one update of a general matrix

- template INST_OR_DECL void ftrsv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

13.226 fflas_L3_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

Macros

- #define __FFLAS_L3_INST_C
- #define INST_OR_DECL
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.226.1 Macro Definition Documentation

13.226.1.1 __FFLAS_L3_INST_C

```
#define __FFLAS_L3_INST_C
```

13.226.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.226.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.226.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.226.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.226.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.226.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.226.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.226.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.226.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.227 fflas_L3_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

Macros

- #define INST_OR_DECL <>
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.227.1 Macro Definition Documentation

13.227.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.227.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.227.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.227.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.227.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.227.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.227.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.227.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.227.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.228 fflas_L3_inst_implem.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLAS__TRSM_READONLY

Functions

- template INST_OR_DECL void ftrsm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  *ftrsm: **TR**iangular **S**ystem solve with **M**atrix.*
- template INST_OR_DECL void ftrmm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)

  *ftrmm: **TR**iangular **M**atrix **M**ultiply.*

- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc)

  *fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Sequential seq)
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > par)
- template INST_OR_DECL FFLAS_ELT * fgemm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *B, const size_t ldb, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > par)
- template INST_OR_DECL FFLAS_ELT * fsquare (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_TRANSPOSE ta, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT beta, FFLAS_ELT *C, const size_t ldc)

  fsquare: Squares a matrix.

13.228.1 Macro Definition Documentation

13.228.1.1 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.229 fflas_lvl1.C File Reference

C functions calls for level 1 FFLAS in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

Functions

- void freducein_1_modular_double (const double p, const size_t n, double *X, const size_t incX, bool positive)
- void freduce_1_modular_double (const double p, const size_t n, const double *Y, const size_t incY, double *X, const size_t incX, bool positive)
- void fnegin_1_modular_double (const double p, const size_t n, double *X, const size_t incX, bool positive)
- void fneg_1_modular_double (const double p, const size_t n, const double *Y, const size_t incY, double *X, const size_t incX, bool positive)
- void fzero_1_modular_double (const double p, const size_t n, double *X, const size_t incX, bool positive)
- bool fiszero_1_modular_double (const double p, const size_t n, const double *X, const size_t incX, bool positive)
- bool fequal_1_modular_double (const double p, const size_t n, const double *X, const size_t incX, const double *Y, const size_t incY, bool positive)
- void fassign_1_modular_double (const double p, const size_t n, const double *Y, const size_t incY, double *X, const size_t incX, bool positive)
- void fscalin_1_modular_double (const double p, const size_t n, const double alpha, double *X, const size_t incX, bool positive)

- void fscal_1_modular_double (const double p, const size_t n, const double alpha, const double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- void faxpy_1_modular_double (const double p, const size_t n, const double alpha, const double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- double fdot_1_modular_double (const double p, const size_t n, const double ∗X, const size_t incX, const double ∗Y, const size_t incY, bool positive)
- void fswap_1_modular_double (const double p, const size_t n, double ∗X, const size_t incX, double ∗Y, const size_t incY, bool positive)
- void fadd_1_modular_double (const double p, const size_t n, const double ∗A, const size_t incA, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void fsub_1_modular_double (const double p, const size_t n, const double ∗A, const size_t incA, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void faddin_1_modular_double (const double p, const size_t n, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)
- void fsubin_1_modular_double (const double p, const size_t n, const double ∗B, const size_t incB, double ∗C, const size_t incC, bool positive)

13.229.1 Detailed Description

C functions calls for level 1 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level1.inl

13.229.2 Function Documentation

13.229.2.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
             const double p,
             const size_t n,
             double * X,
             const size_t incX,
             bool positive)
```

13.229.2.2 freduce_1_modular_double()

```
void freduce_1_modular_double (
             const double p,
             const size_t n,
             const double * Y,
             const size_t incY,
             double * X,
             const size_t incX,
             bool positive)
```

13.229.2.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
             const double p,
             const size_t n,
             double * X,
             const size_t incX,
             bool positive)
```

13.229.2.4 fneg_1_modular_double()

```
void fneg_1_modular_double (
            const double p,
            const size_t n,
            const double * Y,
            const size_t incY,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.6 fiszero_1_modular_double()

```
bool fiszero_1_modular_double (
            const double p,
            const size_t n,
            const double * X,
            const size_t incX,
            bool positive)
```

13.229.2.7 fequal_1_modular_double()

```
bool fequal_1_modular_double (
            const double p,
            const size_t n,
            const double * X,
            const size_t incX,
            const double * Y,
            const size_t incY,
            bool positive)
```

13.229.2.8 fassign_1_modular_double()

```
void fassign_1_modular_double (
            const double p,
            const size_t n,
            const double * Y,
            const size_t incY,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
            const double p,
            const size_t n,
            const double alpha,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.10 fscal_1_modular_double()

```
void fscal_1_modular_double (
             const double p,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t incX,
             double * Y,
             const size_t incY,
             bool positive)
```

13.229.2.11 faxpy_1_modular_double()

```
void faxpy_1_modular_double (
             const double p,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t incX,
             double * Y,
             const size_t incY,
             bool positive)
```

13.229.2.12 fdot_1_modular_double()

```
double fdot_1_modular_double (
             const double p,
             const size_t n,
             const double * X,
             const size_t incX,
             const double * Y,
             const size_t incY,
             bool positive)
```

13.229.2.13 fswap_1_modular_double()

```
void fswap_1_modular_double (
             const double p,
             const size_t n,
             double * X,
             const size_t incX,
             double * Y,
             const size_t incY,
             bool positive)
```

13.229.2.14 fadd_1_modular_double()

```
void fadd_1_modular_double (
             const double p,
             const size_t n,
             const double * A,
             const size_t incA,
             const double * B,
             const size_t incB,
             double * C,
             const size_t incC,
             bool positive)
```

13.229.2.15 fsub_1_modular_double()

```
void fsub_1_modular_double (
            const double p,
            const size_t n,
            const double * A,
            const size_t incA,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.229.2.16 faddin_1_modular_double()

```
void faddin_1_modular_double (
            const double p,
            const size_t n,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.229.2.17 fsubin_1_modular_double()

```
void fsubin_1_modular_double (
            const double p,
            const size_t n,
            const double * B,
            const size_t incB,
            double * C,
            const size_t incC,
            bool positive)
```

13.230 fflas_lvl2.C File Reference

C functions calls for level 2 FFLAS in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

Functions

- void fassign_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void fzero_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t lda, bool positive)
- bool fequal_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, const double ∗B, const size_t ldb, bool positive)
- bool fiszero_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, bool positive)
- void fidentity_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t lda, const double d, bool positive)
- void freducein_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t lda, bool positive)

- void freduce_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void fnegin_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t lda, bool positive)
- void fneg_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void fscalin_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, double ∗A, const size_t lda, bool positive)
- void fscal_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void faxpy_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void fmove_2_modular_double (const double p, const size_t m, const size_t n, double ∗A, const size_t lda, double ∗B, const size_t ldb, bool positive)
- void fadd_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, const double ∗B, const size_t ldb, double ∗C, const size_t ldc, bool positive)
- void fsub_2_modular_double (const double p, const size_t m, const size_t n, const double ∗A, const size_t lda, const double ∗B, const size_t ldb, double ∗C, const size_t ldc, bool positive)
- void fsubin_2_modular_double (const double p, const size_t m, const size_t n, const double ∗B, const size_t ldb, double ∗C, const size_t ldc, bool positive)
- void faddin_2_modular_double (const double p, const size_t m, const size_t n, const double ∗B, const size_t ldb, double ∗C, const size_t ldc, bool positive)
- double ∗ fgemv_2_modular_double (const double p, const enum FFLAS_C_TRANSPOSE TransA, const size_t m, const size_t n, const double alpha, const double ∗A, const size_t lda, const double ∗X, const size_t incX, const double beta, double ∗Y, const size_t incY, bool positive)
- void fger_2_modular_double (const double p, const size_t m, const size_t n, const double alpha, const double ∗X, const size_t incX, const double ∗Y, const size_t incY, double ∗A, const size_t lda, bool positive)
- void ftrsv_2_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE TransA, const enum FFLAS_C_DIAG Diag, const size_t n, const double ∗A, const size_t lda, double ∗X, int incX, bool positive)

13.230.1 Detailed Description

C functions calls for level 2 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level2.inl

13.230.2 Function Documentation

13.230.2.1 fassign_2_modular_double()

```
void fassign_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.2 fzero_2_modular_double()

```
void fzero_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t lda,
            bool positive)
```

13.230.2.3 fequal_2_modular_double()

```
bool fequal_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            const double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.4 fiszero_2_modular_double()

```
bool fiszero_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            bool positive)
```

13.230.2.5 fidentity_2_modular_double()

```
void fidentity_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t lda,
            const double d,
            bool positive)
```

13.230.2.6 freducein_2_modular_double()

```
void freducein_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t lda,
            bool positive)
```

13.230.2.7 freduce_2_modular_double()

```
void freduce_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
```

```
            const double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.8 fnegin_2_modular_double()

```
void fnegin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t lda,
            bool positive)
```

13.230.2.9 fneg_2_modular_double()

```
void fneg_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.10 fscalin_2_modular_double()

```
void fscalin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double alpha,
            double * A,
            const size_t lda,
            bool positive)
```

13.230.2.11 fscal_2_modular_double()

```
void fscal_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.12 faxpy_2_modular_double()

```
void faxpy_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double alpha,
```

```
            const double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.13 fmove_2_modular_double()

```
void fmove_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            bool positive)
```

13.230.2.14 fadd_2_modular_double()

```
void fadd_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            const double * B,
            const size_t ldb,
            double * C,
            const size_t ldc,
            bool positive)
```

13.230.2.15 fsub_2_modular_double()

```
void fsub_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * A,
            const size_t lda,
            const double * B,
            const size_t ldb,
            double * C,
            const size_t ldc,
            bool positive)
```

13.230.2.16 fsubin_2_modular_double()

```
void fsubin_2_modular_double (
            const double p,
            const size_t m,
            const size_t n,
            const double * B,
            const size_t ldb,
            double * C,
            const size_t ldc,
            bool positive)
```

13.230.2.17 faddin_2_modular_double()

```
void faddin_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double * B,
             const size_t ldb,
             double * C,
             const size_t ldc,
             bool positive)
```

13.230.2.18 fgemv_2_modular_double()

```
double * fgemv_2_modular_double (
             const double p,
             const enum FFLAS_C_TRANSPOSE TransA,
             const size_t m,
             const size_t n,
             const double alpha,
             const double * A,
             const size_t lda,
             const double * X,
             const size_t incX,
             const double beta,
             double * Y,
             const size_t incY,
             bool positive)
```

13.230.2.19 fger_2_modular_double()

```
void fger_2_modular_double (
             const double p,
             const size_t m,
             const size_t n,
             const double alpha,
             const double * X,
             const size_t incX,
             const double * Y,
             const size_t incY,
             double * A,
             const size_t lda,
             bool positive)
```

13.230.2.20 ftrsv_2_modular_double()

```
void ftrsv_2_modular_double (
             const double p,
             const enum FFLAS_C_UPLO Uplo,
             const enum FFLAS_C_TRANSPOSE TransA,
             const enum FFLAS_C_DIAG Diag,
             const size_t n,
             const double * A,
             const size_t lda,
             double * X,
             int incX,
             bool positive)
```

13.231 fflas_lvl3.C File Reference

C functions calls for level 3 FFLAS in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

Functions

- void ftrsm_3_modular_double (const double p, const enum FFLAS_C_SIDE Side, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE tA, const enum FFLAS_C_DIAG Diag, const size_t m, const size_t n, const double alpha, const double *A, const size_t ldA, double *B, const size_t ldB, bool positive)
- void ftrmm_3_modular_double (const double p, const enum FFLAS_C_SIDE Side, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_TRANSPOSE tA, const enum FFLAS_C_DIAG Diag, const size_t m, const size_t n, const double alpha, double *A, const size_t ldA, double *B, const size_t ldB, bool positive)
- double * fgemm_3_modular_double (const double p, const enum FFLAS_C_TRANSPOSE tA, const enum FFLAS_C_TRANSPOSE tB, const size_t m, const size_t n, const size_t k, const double alpha, const double *A, const size_t ldA, const double *B, const size_t ldB, const double betA, double *C, const size_t ldC, bool positive)
- double * fsquare_3_modular_double (const double p, const enum FFLAS_C_TRANSPOSE tA, const size_t n, const double alpha, const double *A, const size_t ldA, const double betA, double *C, const size_t ldC, bool positive)

13.231.1 Detailed Description

C functions calls for level 3 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level3.inl

13.231.2 Function Documentation

13.231.2.1 ftrsm_3_modular_double()

```
void ftrsm_3_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_TRANSPOSE tA,
            const enum FFLAS_C_DIAG Diag,
            const size_t m,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t ldA,
            double * B,
            const size_t ldB,
            bool positive)
```

13.231.2.2 ftrmm_3_modular_double()

```
void ftrmm_3_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_TRANSPOSE tA,
            const enum FFLAS_C_DIAG Diag,
            const size_t m,
            const size_t n,
            const double alpha,
            double * A,
            const size_t ldA,
            double * B,
            const size_t ldB,
            bool positive)
```

13.231.2.3 fgemm_3_modular_double()

```
double * fgemm_3_modular_double (
            const double p,
            const enum FFLAS_C_TRANSPOSE tA,
            const enum FFLAS_C_TRANSPOSE tB,
            const size_t m,
            const size_t n,
            const size_t k,
            const double alpha,
            const double * A,
            const size_t ldA,
            const double * B,
            const size_t ldB,
            const double betA,
            double * C,
            const size_t ldC,
            bool positive)
```

13.231.2.4 fsquare_3_modular_double()

```
double * fsquare_3_modular_double (
            const double p,
            const enum FFLAS_C_TRANSPOSE tA,
            const size_t n,
            const double alpha,
            const double * A,
            const size_t ldA,
            const double betA,
            double * C,
            const size_t ldC,
            bool positive)
```

13.232 fflas_sparse.C File Reference

C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h.

13.232.1 Detailed Description

C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_sparse.h

13.233 ffpack.C File Reference

C functions calls for FFPACK in ffpack-c.h.

```
#include "fflas-ffpack/interfaces/libs/ffpack_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

Functions

- void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)
- void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)
- void MatrixApplyS_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)
- void PermApplyS_double (double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- void MatrixApplyT_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)
- void PermApplyT_double (double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)
- void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)
- void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)
- void cyclic_shift_mathPerm (size_t *P, const size_t s)
- void cyclic_shift_row_modular_double (const double p, double *A, size_t m, size_t n, size_t lda, bool positive)
- void cyclic_shift_col_modular_double (const double p, double *A, size_t m, size_t n, size_t lda, bool positive)
- void applyP_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const enum FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, double *A, const size_t lda, const size_t *P, bool positive)
- void fgetrsin_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, double *A, const size_t lda, const size_t *P, const size_t *Q, double *B, const size_t ldb, int *info, bool positive)
- double * fgetrsv_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, double *A, const size_t lda, const size_t *P, const size_t *Q, double *X, const size_t ldx, const double *B, const size_t ldb, int *info, bool positive)
- size_t fgesvin_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double *B, const size_t ldb, int *info, bool positive)
- size_t fgesv_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, double *A, const size_t lda, double *X, const size_t ldx, const double *B, const size_t ldb, int *info, bool positive)
- void ftrtri_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t N, double *A, const size_t lda, bool positive)
- void trinv_left_modular_double (const double p, const size_t N, const double *L, const size_t ldl, double *X, const size_t ldx, bool positive)
- void ftrtrm_modular_double (const double p, const FFLAS::FFLAS_SIDE side, const enum FFLAS::FFLAS_DIAG Diag, const size_t N, double *A, const size_t lda, bool positive)

- size_t PLUQ_modular_double (const double p, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, bool positive)
- size_t LUdivine_modular_double (const double p, const enum FFLAS::FFLAS_DIAG Diag, const enum FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const enum FFPACK_C_LU_TAG LuTag, const size_t cutoff, bool positive)
- size_t ColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pRowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pReducedColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pReducedRowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pRowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pReducedColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pReducedRowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pRowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)

- size_t pReducedColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t pReducedRowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- double * Invertin_modular_double (const double p, const size_t M, double *A, const size_t lda, int *nullity, bool positive)
- double * Invert_modular_double (const double p, const size_t M, const double *A, const size_t lda, double *X, const size_t ldx, int *nullity, bool positive)
- double * Invert2_modular_double (const double p, const size_t M, double *A, const size_t lda, double *X, const size_t ldx, int *nullity, bool positive)
- size_t KrylovElim_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, const size_t deg, size_t *iterates, size_t *inviterates, const size_t maxit, size_t virt, bool positive)
- size_t SpecRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, const size_t deg, size_t *rankProfile, bool positive)
- size_t Rank_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, bool positive)
- bool IsSingular_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, bool positive)
- double Det_modular_double (const double p, const size_t N, double *A, const size_t lda, bool positive)
- double * Solve_modular_double (const double p, const size_t M, double *A, const size_t lda, double *x, const int incx, const double *b, const int incb, bool positive)
- void solveLB_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, double *L, const size_t ldl, const size_t *Q, double *B, const size_t ldb, bool positive)
- void solveLB2_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, double *L, const size_t ldl, const size_t *Q, double *B, const size_t ldb, bool positive)
- void RandomNullSpaceVector_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double *X, const size_t incX, bool positive)
- size_t NullSpaceBasis_modular_double (const double p, const enum FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double **NS, size_t *ldn, size_t *NSdim, bool positive)
- size_t RowRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rkprofile, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rkprofile, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const enum FFPACK_C_LU_TAG LuTag)
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)
- size_t RowRankProfileSubmatrixIndices_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rowindices, size_t **colindices, size_t *R, bool positive)
- size_t ColRankProfileSubmatrixIndices_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rowindices, size_t **colindices, size_t *R, bool positive)
- size_t RowRankProfileSubmatrix_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, double **X, size_t *R, bool positive)
- size_t ColRankProfileSubmatrix_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, double **X, size_t *R, bool positive)
- void getTriangular_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, const size_t R, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, bool positive)
- void getTriangularin_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, const size_t R, double *A, const size_t lda, bool positive)

- void getEchelonForm_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, const size_t R, const size_t *P, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getEchelonFormin_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, const size_t R, const size_t *P, double *A, const size_t lda, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getEchelonTransform_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const enum FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const double *A, const size_t lda, double *T, const size_t ldt, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonForm_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonFormin_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, double *A, const size_t lda, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonTransform_modular_double (const double p, const enum FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const double *A, const size_t lda, double *T, const size_t ldt, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

13.233.1 Detailed Description

C functions calls for FFPACK in ffpack-c.h.

Author

Brice Boyer

See also

ffpack/ffpack.h

13.233.2 Function Documentation

13.233.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N)
```

13.233.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N)
```

13.233.2.3 MatrixApplyS_modular_double()

```
void MatrixApplyS_modular_double (
            const double p,
            double * A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
```

```
            const size_t R3,
            const size_t R4,
            bool positive)
```

13.233.2.4 PermApplyS_double()

```
void PermApplyS_double (
            double * A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)
```

13.233.2.5 MatrixApplyT_modular_double()

```
void MatrixApplyT_modular_double (
            const double p,
            double * A,
            const size_t lda,
            const size_t width,
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4,
            bool positive)
```

13.233.2.6 PermApplyT_double()

```
void PermApplyT_double (
            double * A,
            const size_t lda,
            const size_t width,
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)
```

13.233.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)
```

13.233.2.11 cyclic_shift_row_modular_double()

```
void cyclic_shift_row_modular_double (
            const double p,
            double * A,
            size_t m,
            size_t n,
            size_t lda,
            bool positive)
```

13.233.2.12 cyclic_shift_col_modular_double()

```
void cyclic_shift_col_modular_double (
            const double p,
            double * A,
            size_t m,
            size_t n,
            size_t lda,
            bool positive)
```

13.233.2.13 applyP_modular_double()

```
void applyP_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const enum FFLAS::FFLAS_TRANSPOSE Trans,
            const size_t M,
            const size_t ibeg,
            const size_t iend,
            double * A,
            const size_t lda,
            const size_t * P,
            bool positive)
```

13.233.2.14 fgetrsin_modular_double()

```
void fgetrsin_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * A,
            const size_t lda,
            const size_t * P,
```

```
            const size_t * Q,
            double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.233.2.15 fgetrsv_modular_double()

```
double * fgetrsv_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            const size_t R,
            double * A,
            const size_t lda,
            const size_t * P,
            const size_t * Q,
            double * X,
            const size_t ldx,
            const double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.233.2.16 fgesvin_modular_double()

```
size_t fgesvin_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.233.2.17 fgesv_modular_double()

```
size_t fgesv_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            double * A,
            const size_t lda,
            double * X,
            const size_t ldx,
            const double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.233.2.18 ftrtri_modular_double()

```
void ftrtri_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.19 trinv_left_modular_double()

```
void trinv_left_modular_double (
            const double p,
            const size_t N,
            const double * L,
            const size_t ldl,
            double * X,
            const size_t ldx,
            bool positive)
```

13.233.2.20 ftrtrm_modular_double()

```
void ftrtrm_modular_double (
            const double p,
            const FFLAS::FFLAS_SIDE side,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.21 PLUQ_modular_double()

```
size_t PLUQ_modular_double (
            const double p,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            bool positive)
```

13.233.2.22 LUdivine_modular_double()

```
size_t LUdivine_modular_double (
            const double p,
            const enum FFLAS::FFLAS_DIAG Diag,
            const enum FFLAS::FFLAS_TRANSPOSE Trans,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const enum FFPACK_C_LU_TAG LuTag,
```

```
            const size_t cutoff,
            bool positive)
```

13.233.2.23 ColumnEchelonForm_modular_double()

```
size_t ColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.24 RowEchelonForm_modular_double()

```
size_t RowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.25 ReducedColumnEchelonForm_modular_double()

```
size_t ReducedColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.26 ReducedRowEchelonForm_modular_double()

```
size_t ReducedRowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.27 ColumnEchelonForm_modular_float()

```
size_t ColumnEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.28 RowEchelonForm_modular_float()

```
size_t RowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.29 ReducedColumnEchelonForm_modular_float()

```
size_t ReducedColumnEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.30 ReducedRowEchelonForm_modular_float()

```
size_t ReducedRowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.31 ColumnEchelonForm_modular_int32_t()

```
size_t ColumnEchelonForm_modular_int32_t (
            const int32_t p,
```

```
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.32 RowEchelonForm_modular_int32_t()

```
size_t RowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.33 ReducedColumnEchelonForm_modular_int32_t()

```
size_t ReducedColumnEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.34 ReducedRowEchelonForm_modular_int32_t()

```
size_t ReducedRowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.35 pColumnEchelonForm_modular_double()

```
size_t pColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
```

```
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.36 pRowEchelonForm_modular_double()

```
size_t pRowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.37 pReducedColumnEchelonForm_modular_double()

```
size_t pReducedColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.38 pReducedRowEchelonForm_modular_double()

```
size_t pReducedRowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.39 pColumnEchelonForm_modular_float()

```
size_t pColumnEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
```

```
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.40 pRowEchelonForm_modular_float()

```
size_t pRowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.41 pReducedColumnEchelonForm_modular_float()

```
size_t pReducedColumnEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.42 pReducedRowEchelonForm_modular_float()

```
size_t pReducedRowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.43 pColumnEchelonForm_modular_int32_t()

```
size_t pColumnEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.44 pRowEchelonForm_modular_int32_t()

```
size_t pRowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.45 pReducedColumnEchelonForm_modular_int32_t()

```
size_t pReducedColumnEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.46 pReducedRowEchelonForm_modular_int32_t()

```
size_t pReducedRowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.47 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive)
```

13.233.2.48 Invert_modular_double()

```
double * Invert_modular_double (
            const double p,
            const size_t M,
            const double * A,
            const size_t lda,
```

```
            double * X,
            const size_t ldx,
            int * nullity,
            bool positive)
```

13.233.2.49 Invert2_modular_double()

```
double * Invert2_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            double * X,
            const size_t ldx,
            int * nullity,
            bool positive)
```

13.233.2.50 KrylovElim_modular_double()

```
size_t KrylovElim_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const size_t deg,
            size_t * iterates,
            size_t * inviterates,
            const size_t maxit,
            size_t virt,
            bool positive)
```

13.233.2.51 SpecRankProfile_modular_double()

```
size_t SpecRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            const size_t deg,
            size_t * rankProfile,
            bool positive)
```

13.233.2.52 Rank_modular_double()

```
size_t Rank_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.53 IsSingular_modular_double()

```
bool IsSingular_modular_double (
```

```
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.54 Det_modular_double()

```
double Det_modular_double (
            const double p,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.55 Solve_modular_double()

```
double * Solve_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            double * x,
            const int incx,
            const double * b,
            const int incb,
            bool positive)
```

13.233.2.56 solveLB_modular_double()

```
void solveLB_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * L,
            const size_t ldl,
            const size_t * Q,
            double * B,
            const size_t ldb,
            bool positive)
```

13.233.2.57 solveLB2_modular_double()

```
void solveLB2_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * L,
            const size_t ldl,
            const size_t * Q,
            double * B,
            const size_t ldb,
            bool positive)
```

13.233.2.58 RandomNullSpaceVector_modular_double()

```
void RandomNullSpaceVector_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double * X,
            const size_t incX,
            bool positive)
```

13.233.2.59 NullSpaceBasis_modular_double()

```
size_t NullSpaceBasis_modular_double (
            const double p,
            const enum FFLAS::FFLAS_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double ** NS,
            size_t * ldn,
            size_t * NSdim,
            bool positive)
```

13.233.2.60 RowRankProfile_modular_double()

```
size_t RowRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rkprofile,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.61 ColumnRankProfile_modular_double()

```
size_t ColumnRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rkprofile,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.62 RankProfileFromLU()

```
void RankProfileFromLU (
            const size_t * P,
            const size_t N,
            const size_t R,
            size_t * rkprofile,
            const enum FFPACK_C_LU_TAG LuTag)
```

13.233.2.63 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t LSm,
            const size_t LSn,
            const size_t * P,
            const size_t * Q,
            size_t * RRP,
            size_t * CRP)
```

13.233.2.64 RowRankProfileSubmatrixIndices_modular_double()

```
size_t RowRankProfileSubmatrixIndices_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rowindices,
            size_t ** colindices,
            size_t * R,
            bool positive)
```

13.233.2.65 ColRankProfileSubmatrixIndices_modular_double()

```
size_t ColRankProfileSubmatrixIndices_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rowindices,
            size_t ** colindices,
            size_t * R,
            bool positive)
```

13.233.2.66 RowRankProfileSubmatrix_modular_double()

```
size_t RowRankProfileSubmatrix_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double ** X,
            size_t * R,
            bool positive)
```

13.233.2.67 ColRankProfileSubmatrix_modular_double()

```
size_t ColRankProfileSubmatrix_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
```

```
            double ** X,
            size_t * R,
            bool positive)
```

13.233.2.68 getTriangular_modular_double()

```
void getTriangular_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            bool positive)
```

13.233.2.69 getTriangularin_modular_double()

```
void getTriangularin_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            const size_t R,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.70 getEchelonForm_modular_double()

```
void getEchelonForm_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.71 getEchelonFormin_modular_double()

```
void getEchelonFormin_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
```

```
            const size_t R,
            const size_t * P,
            double * A,
            const size_t lda,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.72 getEchelonTransform_modular_double()

```
void getEchelonTransform_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const enum FFLAS::FFLAS_DIAG Diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.73 getReducedEchelonForm_modular_double()

```
void getReducedEchelonForm_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.74 getReducedEchelonFormin_modular_double()

```
void getReducedEchelonFormin_modular_double (
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            double * A,
            const size_t lda,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.75 getReducedEchelonTransform_modular_double()

```
void getReducedEchelonTransform_modular_double (
```

```
            const double p,
            const enum FFLAS::FFLAS_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.233.2.76 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
            const size_t * P,
            size_t * outPerm)
```

13.234 ffpack_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

Macros

- #define FFPACK_COMPILED

Enumerations

- enum FFLAS_C_ORDER { FflasRowMajor =101 , FflasColMajor =102 }
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 }
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }
- enum FFLAS_C_SIDE { FflasLeft = 141 , FflasRight = 142 }
- enum FFPACK_C_LU_TAG { FfpackSlabRecursive = 1 , FfpackTileRecursive = 2 , FfpackSingular = 3 }
- enum FFPACK_C_CHARPOLY_TAG {
  FfpackLUK =1 , FfpackKG =2 , FfpackHybrid =3 , FfpackKGFast =4 ,
  FfpackDanilevski =5 , FfpackArithProg =6 , FfpackKGFastG =7 }
- enum FFPACK_C_MINPOLY_TAG { FfpackDense =1 , FfpackKGF =2 }

Functions

- void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)
- void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)
- void MatrixApplyS_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)
- void PermApplyS_double (double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)
- void MatrixApplyT_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)
- void PermApplyT_double (double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)

- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)
- void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)
- void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)
- void cyclic_shift_mathPerm (size_t *P, const size_t s)
- void cyclic_shift_row_modular_double (const double p, double *A, size_t m, size_t n, size_t lda, bool positive)
- void cyclic_shift_col_modular_double (const double p, double *A, size_t m, size_t n, size_t lda, bool positive)
- void applyP_modular_double (const double p, const enum FFLAS_C_SIDE Side, const enum FFLAS_C_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, double *A, const size_t lda, const size_t *P, bool positive)
- void fgetrsin_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, const size_t R, double *A, const size_t lda, const size_t *P, const size_t *Q, double *B, const size_t ldb, int *info, bool positive)
- double * fgetrs_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, double *A, const size_t lda, const size_t *P, const size_t *Q, double *X, const size_t ldx, const double *B, const size_t ldb, int *info, bool positive)
- size_t fgesvin_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double *B, const size_t ldb, int *info, bool positive)
- size_t fgesv_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, const size_t NRHS, double *A, const size_t lda, double *X, const size_t ldx, const double *B, const size_t ldb, int *info)
- void ftrtri_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG Diag, const size_t N, double *A, const size_t lda, bool positive)
- void trinv_left_modular_double (const double p, const size_t N, const double *L, const size_t ldl, double *X, const size_t ldx, bool positive)
- void ftrtrm_modular_double (const double p, const enum FFLAS_C_DIAG diag, const size_t N, double *A, const size_t lda, bool positive)
- size_t PLUQ_modular_double (const double p, const enum FFLAS_C_DIAG Diag, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, bool positive)
- size_t LUdivine_modular_double (const double p, const enum FFLAS_C_DIAG Diag, const enum FFLAS_C_TRANSPOSE trans, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const enum FFPACK_C_LU_TAG LuTag, const size_t cutoff, bool positive)
- size_t LUdivine_small_modular_double (const double p, const enum FFLAS_C_DIAG Diag, const enum FFLAS_C_TRANSPOSE trans, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t LUdivine_gauss_modular_double (const double p, const enum FFLAS_C_DIAG Diag, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t RowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedColumnEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)

- size_t ReducedColumnEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_float (const float p, const size_t M, const size_t N, float *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedColumnEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm_modular_int32_t (const int32_t p, const size_t M, const size_t N, int32_t *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ReducedRowEchelonForm2_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, bool positive)
- size_t REF_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *Qt, size_t *P, bool positive)
- double * Invertin_modular_double (const double p, const size_t M, double *A, const size_t lda, int *nullity, bool positive)
- double * Invert_modular_double (const double p, const size_t M, const double *A, const size_t lda, double *X, const size_t ldx, int *nullity, bool positive)
- double * Invert2_modular_double (const double p, const size_t M, double *A, const size_t lda, double *X, const size_t ldx, int *nullity, bool positive)
- size_t KrylovElim_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t *P, size_t *Q, const size_t deg, size_t *iterates, size_t *inviterates, const size_t maxit, size_t virt, bool positive)
- size_t SpecRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, const size_t deg, size_t *rankProfile, bool positive)
- size_t Rank_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, bool positive)
- bool IsSingular_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, bool positive)
- double Det_modular_double (const double p, const size_t N, double *A, const size_t lda, bool positive)
- double * Solve_modular_double (const double p, const size_t M, double *A, const size_t lda, double *x, const int incx, const double *b, const int incb, bool positive)
- void solveLB_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, const size_t R, double *L, const size_t ldl, const size_t *Q, double *B, const size_t ldb)
- void solveLB2_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, const size_t R, double *L, const size_t ldl, const size_t *Q, double *B, const size_t ldb, bool positive)
- void RandomNullSpaceVector_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double *X, const size_t incX, bool positive)
- size_t NullSpaceBasis_modular_double (const double p, const enum FFLAS_C_SIDE Side, const size_t M, const size_t N, double *A, const size_t lda, double **NS, size_t *ldn, size_t *NSdim, bool positive)
- size_t RowRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rkprofile, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- size_t ColumnRankProfile_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rkprofile, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const enum FFPACK_C_LU_TAG LuTag)
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)
- size_t RowRankProfileSubmatrixIndices_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rowindices, size_t **colindices, size_t *R, bool positive)
- size_t ColRankProfileSubmatrixIndices_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, size_t **rowindices, size_t **colindices, size_t *R, bool positive)
- size_t RowRankProfileSubmatrix_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, double **X, size_t *R, bool positive)
- size_t ColRankProfileSubmatrix_modular_double (const double p, const size_t M, const size_t N, double *A, const size_t lda, double **X, size_t *R, bool positive)

- void getTriangular_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG diag, const size_t M, const size_t N, const size_t R, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, bool positive)
- void getTriangularin_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG diag, const size_t M, const size_t N, const size_t R, double *A, const size_t lda, bool positive)
- void getEchelonForm_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getEchelonFormin_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, double *A, const size_t lda, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getEchelonTransform_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const enum FFLAS_C_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const double *A, const size_t lda, double *T, const size_t ldt, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonForm_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const double *A, const size_t lda, double *T, const size_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonFormin_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, double *A, const size_t lda, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void getReducedEchelonTransform_modular_double (const double p, const enum FFLAS_C_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const double *A, const size_t lda, double *T, const size_t ldt, const enum FFPACK_C_LU_TAG LuTag, bool positive)
- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

13.234.1 Macro Definition Documentation

13.234.1.1 FFPACK_COMPILED

```
#define FFPACK_COMPILED
```

13.234.2 Enumeration Type Documentation

13.234.2.1 FFLAS_C_ORDER

```
enum FFLAS_C_ORDER
```

Enumerator

FflasRowMajor	
FflasColMajor	

13.234.2.2 FFLAS_C_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Enumerator

FflasNoTrans	
FflasTrans	

13.234.2.3 FFLAS_C_UPLO

```
enum FFLAS_C_UPLO
```

Enumerator

FflasUpper	
FflasLower	

13.234.2.4 FFLAS_C_DIAG

`enum FFLAS_C_DIAG`

Enumerator

FflasNonUnit	
FflasUnit	

13.234.2.5 FFLAS_C_SIDE

`enum FFLAS_C_SIDE`

Enumerator

FflasLeft	
FflasRight	

13.234.2.6 FFPACK_C_LU_TAG

`enum FFPACK_C_LU_TAG`

Enumerator

FfpackSlabRecursive	
FfpackTileRecursive	
FfpackSingular	

13.234.2.7 FFPACK_C_CHARPOLY_TAG

`enum FFPACK_C_CHARPOLY_TAG`

Enumerator

FfpackLUK	
FfpackKG	
FfpackHybrid	
FfpackKGFast	
FfpackDanilevski	
FfpackArithProg	
FfpackKGFastG	

13.234.2.8 FFPACK_C_MINPOLY_TAG

`enum FFPACK_C_MINPOLY_TAG`

Enumerator

FfpackDense	
FfpackKGF	

13.234.3 Function Documentation

13.234.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N)
```

13.234.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N)
```

13.234.3.3 MatrixApplyS_modular_double()

```
void MatrixApplyS_modular_double (
            const double p,
            double * A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4,
            bool positive)
```

13.234.3.4 PermApplyS_double()

```
void PermApplyS_double (
            double * A,
            const size_t lda,
            const size_t width,
            const size_t M2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)
```

13.234.3.5 MatrixApplyT_modular_double()

```
void MatrixApplyT_modular_double (
            const double p,
            double * A,
            const size_t lda,
            const size_t width,
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
```

```
            const size_t R4,
            bool positive)
```

13.234.3.6 PermApplyT_double()

```
void PermApplyT_double (
            double * A,
            const size_t lda,
            const size_t width,
            const size_t N2,
            const size_t R1,
            const size_t R2,
            const size_t R3,
            const size_t R4)
```

13.234.3.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)
```

13.234.3.11 cyclic_shift_row_modular_double()

```
void cyclic_shift_row_modular_double (
            const double p,
            double * A,
            size_t m,
            size_t n,
            size_t lda,
            bool positive)
```

13.234.3.12 cyclic_shift_col_modular_double()

```
void cyclic_shift_col_modular_double (
            const double p,
```

```
            double * A,
            size_t m,
            size_t n,
            size_t lda,
            bool positive)
```

13.234.3.13 applyP_modular_double()

```
void applyP_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const enum FFLAS_C_TRANSPOSE Trans,
            const size_t M,
            const size_t ibeg,
            const size_t iend,
            double * A,
            const size_t lda,
            const size_t * P,
            bool positive)
```

13.234.3.14 fgetrsin_modular_double()

```
void fgetrsin_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * A,
            const size_t lda,
            const size_t * P,
            const size_t * Q,
            double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.234.3.15 fgetrs_modular_double()

```
double * fgetrs_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            const size_t R,
            double * A,
            const size_t lda,
            const size_t * P,
            const size_t * Q,
            double * X,
            const size_t ldx,
            const double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.234.3.16 fgesvin_modular_double()

```
size_t fgesvin_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double * B,
            const size_t ldb,
            int * info,
            bool positive)
```

13.234.3.17 fgesv_modular_double()

```
size_t fgesv_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t NRHS,
            double * A,
            const size_t lda,
            double * X,
            const size_t ldx,
            const double * B,
            const size_t ldb,
            int * info)
```

13.234.3.18 ftrtri_modular_double()

```
void ftrtri_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG Diag,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.19 trinv_left_modular_double()

```
void trinv_left_modular_double (
            const double p,
            const size_t N,
            const double * L,
            const size_t ldl,
            double * X,
            const size_t ldx,
            bool positive)
```

13.234.3.20 ftrtrm_modular_double()

```
void ftrtrm_modular_double (
            const double p,
            const enum FFLAS_C_DIAG diag,
            const size_t N,
            double * A,
```

```
            const size_t lda,
            bool positive)
```

13.234.3.21 PLUQ_modular_double()

```
size_t PLUQ_modular_double (
            const double p,
            const enum FFLAS_C_DIAG Diag,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            bool positive)
```

13.234.3.22 LUdivine_modular_double()

```
size_t LUdivine_modular_double (
            const double p,
            const enum FFLAS_C_DIAG Diag,
            const enum FFLAS_C_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const enum FFPACK_C_LU_TAG LuTag,
            const size_t cutoff,
            bool positive)
```

13.234.3.23 LUdivine_small_modular_double()

```
size_t LUdivine_small_modular_double (
            const double p,
            const enum FFLAS_C_DIAG Diag,
            const enum FFLAS_C_TRANSPOSE trans,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.24 LUdivine_gauss_modular_double()

```
size_t LUdivine_gauss_modular_double (
            const double p,
            const enum FFLAS_C_DIAG Diag,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
```

```
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.25 ColumnEchelonForm_modular_double()

```
size_t ColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.26 RowEchelonForm_modular_double()

```
size_t RowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.27 ColumnEchelonForm_modular_float()

```
size_t ColumnEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.28 RowEchelonForm_modular_float()

```
size_t RowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.29 ColumnEchelonForm_modular_int32_t()

```
size_t ColumnEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.30 RowEchelonForm_modular_int32_t()

```
size_t RowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.31 ReducedColumnEchelonForm_modular_double()

```
size_t ReducedColumnEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.32 ReducedRowEchelonForm_modular_double()

```
size_t ReducedRowEchelonForm_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.33 ReducedColumnEchelonForm_modular_float()

```
size_t ReducedColumnEchelonForm_modular_float (
            const float p,
```

```
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.34 ReducedRowEchelonForm_modular_float()

```
size_t ReducedRowEchelonForm_modular_float (
            const float p,
            const size_t M,
            const size_t N,
            float * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.35 ReducedColumnEchelonForm_modular_int32_t()

```
size_t ReducedColumnEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.36 ReducedRowEchelonForm_modular_int32_t()

```
size_t ReducedRowEchelonForm_modular_int32_t (
            const int32_t p,
            const size_t M,
            const size_t N,
            int32_t * A,
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.37 ReducedRowEchelonForm2_modular_double()

```
size_t ReducedRowEchelonForm2_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
```

```
            const size_t lda,
            size_t * P,
            size_t * Qt,
            const bool transform,
            bool positive)
```

13.234.3.38 REF_modular_double()

```
size_t REF_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            const size_t colbeg,
            const size_t rowbeg,
            const size_t colsize,
            size_t * Qt,
            size_t * P,
            bool positive)
```

13.234.3.39 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive)
```

13.234.3.40 Invert_modular_double()

```
double * Invert_modular_double (
            const double p,
            const size_t M,
            const double * A,
            const size_t lda,
            double * X,
            const size_t ldx,
            int * nullity,
            bool positive)
```

13.234.3.41 Invert2_modular_double()

```
double * Invert2_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            double * X,
            const size_t ldx,
            int * nullity,
            bool positive)
```

13.234.3.42 KrylovElim_modular_double()

```
size_t KrylovElim_modular_double (
            const double p,
```

```
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t * P,
            size_t * Q,
            const size_t deg,
            size_t * iterates,
            size_t * inviterates,
            const size_t maxit,
            size_t virt,
            bool positive)
```

13.234.3.43 SpecRankProfile_modular_double()

```
size_t SpecRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            const size_t deg,
            size_t * rankProfile,
            bool positive)
```

13.234.3.44 Rank_modular_double()

```
size_t Rank_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.45 IsSingular_modular_double()

```
bool IsSingular_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.46 Det_modular_double()

```
double Det_modular_double (
            const double p,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.47 Solve_modular_double()

```
double * Solve_modular_double (
            const double p,
            const size_t M,
```

```
            double * A,
            const size_t lda,
            double * x,
            const int incx,
            const double * b,
            const int incb,
            bool positive)
```

13.234.3.48 solveLB_modular_double()

```
void solveLB_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * L,
            const size_t ldl,
            const size_t * Q,
            double * B,
            const size_t ldb)
```

13.234.3.49 solveLB2_modular_double()

```
void solveLB2_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            const size_t R,
            double * L,
            const size_t ldl,
            const size_t * Q,
            double * B,
            const size_t ldb,
            bool positive)
```

13.234.3.50 RandomNullSpaceVector_modular_double()

```
void RandomNullSpaceVector_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double * X,
            const size_t incX,
            bool positive)
```

13.234.3.51 NullSpaceBasis_modular_double()

```
size_t NullSpaceBasis_modular_double (
            const double p,
            const enum FFLAS_C_SIDE Side,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
```

```
            double ** NS,
            size_t * ldn,
            size_t * NSdim,
            bool positive)
```

13.234.3.52 RowRankProfile_modular_double()

```
size_t RowRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rkprofile,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.53 ColumnRankProfile_modular_double()

```
size_t ColumnRankProfile_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rkprofile,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.54 RankProfileFromLU()

```
void RankProfileFromLU (
            const size_t * P,
            const size_t N,
            const size_t R,
            size_t * rkprofile,
            const enum FFPACK_C_LU_TAG LuTag)
```

13.234.3.55 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t LSm,
            const size_t LSn,
            const size_t * P,
            const size_t * Q,
            size_t * RRP,
            size_t * CRP)
```

13.234.3.56 RowRankProfileSubmatrixIndices_modular_double()

```
size_t RowRankProfileSubmatrixIndices_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
```

```
            size_t ** rowindices,
            size_t ** colindices,
            size_t * R,
            bool positive)
```

13.234.3.57 ColRankProfileSubmatrixIndices_modular_double()

```
size_t ColRankProfileSubmatrixIndices_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            size_t ** rowindices,
            size_t ** colindices,
            size_t * R,
            bool positive)
```

13.234.3.58 RowRankProfileSubmatrix_modular_double()

```
size_t RowRankProfileSubmatrix_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double ** X,
            size_t * R,
            bool positive)
```

13.234.3.59 ColRankProfileSubmatrix_modular_double()

```
size_t ColRankProfileSubmatrix_modular_double (
            const double p,
            const size_t M,
            const size_t N,
            double * A,
            const size_t lda,
            double ** X,
            size_t * R,
            bool positive)
```

13.234.3.60 getTriangular_modular_double()

```
void getTriangular_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            bool positive)
```

13.234.3.61 getTriangularin_modular_double()

```
void getTriangularin_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.62 getEchelonForm_modular_double()

```
void getEchelonForm_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.63 getEchelonFormin_modular_double()

```
void getEchelonFormin_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            double * A,
            const size_t lda,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.64 getEchelonTransform_modular_double()

```
void getEchelonTransform_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const enum FFLAS_C_DIAG diag,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const double * A,
            const size_t lda,
```

```
            double * T,
            const size_t ldt,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.65 getReducedEchelonForm_modular_double()

```
void getReducedEchelonForm_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const bool OnlyNonZeroVectors,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.66 getReducedEchelonFormin_modular_double()

```
void getReducedEchelonFormin_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            double * A,
            const size_t lda,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.67 getReducedEchelonTransform_modular_double()

```
void getReducedEchelonTransform_modular_double (
            const double p,
            const enum FFLAS_C_UPLO Uplo,
            const size_t M,
            const size_t N,
            const size_t R,
            const size_t * P,
            const size_t * Q,
            const double * A,
            const size_t lda,
            double * T,
            const size_t ldt,
            const enum FFPACK_C_LU_TAG LuTag,
            bool positive)
```

13.234.3.68 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
```

```
            const size_t * P,
            size_t * outPerm)
```

13.235 ffpack_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

Macros

- #define __FFPACK_INST_C
- #define FFLAS_COMPILED
- #define INST_OR_DECL
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.235.1 Macro Definition Documentation

13.235.1.1 __FFPACK_INST_C

```
#define __FFPACK_INST_C
```

13.235.1.2 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.235.1.3 INST_OR_DECL

```
#define INST_OR_DECL
```

13.235.1.4 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.235.1.5 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.235.1.6 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.235.1.7 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.235.1.8 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.235.1.9 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.235.1.10 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.235.1.11 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.236 ffpack_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

Macros

- #define FFLAS_COMPILED
- #define INST_OR_DECL <>
- #define FFLAS_FIELD Givaro::ModularBalanced
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t
- #define FFLAS_FIELD Givaro::Modular
- #define FFLAS_ELT double
- #define FFLAS_ELT float
- #define FFLAS_ELT int64_t

13.236.1 Macro Definition Documentation

13.236.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.236.1.2 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.236.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.236.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.236.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.236.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.236.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.236.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.236.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.236.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.237 ffpack_inst_implem.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in MathP as a MathPermutation format.
- void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.
- void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)

  Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.
- void cyclic_shift_mathPerm (size_t *P, const size_t s)
- template<typename Base_t>
  void cyclic_shift_row_col (Base_t *A, size_t m, size_t n, size_t lda)
- template INST_OR_DECL void cyclic_shift_row (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)
- template INST_OR_DECL void cyclic_shift_col (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)
- template INST_OR_DECL void applyP (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, FFLAS_ELT *A, const size_t lda, const size_t *P)
- template INST_OR_DECL void fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL FFLAS_ELT * fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb, int *info)
- template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)

- template INST_OR_DECL void ftrtri (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t threshold)
- template INST_OR_DECL void trinv_left (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *L, const size_t ldl, FFLAS_ELT *X, const size_t ldx)
- template INST_OR_DECL void ftrtrm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL size_t PLUQ (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)
- template INST_OR_DECL size_t LUdivine (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag, const size_t cutoff)
- template INST_OR_DECL size_t LUdivine_small (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t LUdivine_gauss (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t RowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ReducedRowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ColumnEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ReducedColumnEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL FFLAS_ELT * Invert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, int &nullity)
- template INST_OR_DECL FFLAS_ELT * Invert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, int &nullity)
- template INST_OR_DECL FFLAS_ELT * Invert2 (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, int &nullity)
- template INST_OR_DECL std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &charp, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, const size_t degree)
- template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, const size_t degree)
- template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R, Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp, const size_t N, FFLAS_ELT *A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag, const size_t degree)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda, FFLAS_FIELD< FFLAS_ELT >::RandIter &G)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL std::vector< FFLAS_ELT > & MatVecMinPoly (const FFLAS_FIELD< FFLAS_ELT > &F, std::vector< FFLAS_ELT > &minP, const size_t N, const FFLAS_ELT *A, const size_t lda, const FFLAS_ELT *V, const size_t incv)

- template INST_OR_DECL size_t KrylovElim (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const size_t deg, size_t *iterates, size_t *inviterates, const size_t maxit, size_t virt)
- template INST_OR_DECL size_t SpecRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t deg, size_t *rankProfile)
- template INST_OR_DECL size_t Rank (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL bool IsSingular (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda)
- template INST_OR_DECL FFLAS_ELT & Det (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT &det, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)
- template INST_OR_DECL FFLAS_ELT & Det (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT &det, const size_t N, FFLAS_ELT *A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &parH, size_t *P, size_t *Q)
- template INST_OR_DECL FFLAS_ELT * Solve (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *x, const int incx, const FFLAS_ELT *b, const int incb)
- template INST_OR_DECL void solveLB (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *L, const size_t ldl, const size_t *Q, FFLAS_ELT *B, const size_t ldb)
- template INST_OR_DECL void solveLB2 (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *L, const size_t ldl, const size_t *Q, FFLAS_ELT *B, const size_t ldb)
- template INST_OR_DECL void RandomNullSpaceVector (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t incX)
- template INST_OR_DECL size_t NullSpaceBasis (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&NS, size_t &ldn, size_t &NSdim)
- template INST_OR_DECL size_t RowRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL size_t ColumnRankProfile (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag)
- void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_LU_TAG LuTag)

  Recovers the column/row rank profile from the permutation of an LU decomposition.
- size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)

  Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.
- template INST_OR_DECL size_t RowRankProfileSubmatrixIndices (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
- template INST_OR_DECL size_t ColRankProfileSubmatrixIndices (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
- template INST_OR_DECL size_t RowRankProfileSubmatrix (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&X, size_t &R)
- template INST_OR_DECL size_t ColRankProfileSubmatrix (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *&X, size_t &R)
- template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors)
- template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda)

- template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, FFLAS_ELT *A, const size_t lda, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const bool OnlyNonZeroVectors, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, FFLAS_ELT *A, const size_t lda, const FFPACK_LU_TAG LuTag)
- template INST_OR_DECL void getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *T, const size_t ldt, const FFPACK_LU_TAG LuTag)
- void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)

  Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.
- template INST_OR_DECL FFLAS_ELT * LQUPtoInverseOfFullRankMinor (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t rank, FFLAS_ELT *A_factors, const size_t lda, const size_t *QtPointer, FFLAS_ELT *X, const size_t ldx)

13.238 blockcuts.inl File Reference

```
#include <fflas-ffpack/fflas/fflas_enum.h>
#include <math.h>
#include <cassert>
```

Data Structures

- struct Single
- struct Row
- struct Column
- struct Block
- struct Recursive
- struct Fixed
- struct Threads
- struct Grain
- struct TwoD
- struct TwoDAdaptive
- struct ThreeD
- struct ThreeDInPlace
- struct ThreeDAdaptive
- struct Parallel< C, P >
- struct Sequential
- struct Compose< H1, H2 >
- struct ForStrategy1D< blocksize_t, Cut, Param >
- struct ForStrategy2D< blocksize_t, Cut, Param >

Namespaces

- namespace FFLAS
- namespace FFLAS::CuttingStrategy
- namespace FFLAS::StrategyParameter
- namespace FFLAS::ParSeqHelper

  ParSeqHelper for both fgemm and ftrsm.

Macros

- #define __FFLASFFPACK_fflas_blockcuts_INL
- #define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)

Typedefs

- typedef Row RNSModulus

Functions

- template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>
  void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)
- template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>
  void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)

13.238.1 Macro Definition Documentation

13.238.1.1 __FFLASFFPACK_fflas_blockcuts_INL

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

13.238.1.2 __FFLASFFPACK_MINBLOCKCUTS

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

13.239 fflas_plevel1.h File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

Namespaces

- namespace FFLAS

Functions

- template<class Field>
  void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)
- template<class Field, class RandIter>
  void pfrand (const Field &F, RandIter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)
- template<class Field, class Cut, class Param>
  Field::Element & fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)
- template<typename Field, class Cut, class Param>
  Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)

13.240 kaapi_routines.inl File Reference

Macros

- #define __FFLASFFPACK_KAAPI_ROUTINES_INL

13.240.1 Macro Definition Documentation

13.240.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

13.241 parallel.h File Reference

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/paladin/blockcuts.inl"
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define index_t size_t
- #define TASK(M, I)
- #define WAIT
- #define CHECK_DEPENDENCIES
- #define BARRIER
- #define PAR_BLOCK
- #define SYNCH_GROUP(Args...)
- #define THREAD_INDEX 0
- #define NUM_THREADS 1
- #define SET_THREADS(num_threads)
- #define MAX_THREADS 1
- #define READ(Args...)
- #define WRITE(Args...)

- #define READWRITE(Args...)
- #define CONSTREFERENCE(...)
- #define VALUE(...)
- #define BEGIN_PARALLEL_MAIN(Args...)
- #define END_PARALLEL_MAIN(void)
- #define FORBLOCK1D(iter, m, Helper, Args...)
- #define FOR1D(i, m, Helper, Args...)
- #define PARFORBLOCK1D(iter, m, Helper, Args...)
- #define PARFOR1D(iter, m, Helper, Args...)
- #define FORBLOCK2D(iter, m, n, Helper, Args...)
- #define FOR2D(i, j, m, n, Helper, Args...)
- #define PARFORBLOCK2D(iter, m, n, Helper, Args...)
- #define PARFOR2D(i, j, m, n, Helper, Args...)
- #define COMMA ,
- #define MODE(...)
- #define RETURNPARAM(f, P1, Args...)
- #define NUMARGS(...)
- #define PP_NARG_(...)
- #define PP_ARG_N(_1, _2, _3, _4, _5, _6, _7, _8, _9, _10, _11, _12, _13, _14, _15, _16, _17, _18, _19, _20, _21, _22, _23, _24, _25, _26, _27, _28, _29, _30, _31, _32, _33, _34, _35, _36, _37, _38, _39, _40, _41, _42, _43, _44, _45, _46, _47, _48, _49, _50, _51, _52, _53, _54, _55, _56, _57, _58, _59, _60, _61, _62, _63, N, ...)
- #define PP_RSEQ_N()
- #define NOSPLIT()
- #define splitting_0()
- #define splitting_1(a)
- #define splitting_2(a, c)
- #define splitting_3(a, b, c)
- #define splitt(_1, _2, _3, NAME, ...)
- #define SPLITTER(...)

13.241.1 Macro Definition Documentation

13.241.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.241.1.2 index_t

```
#define index_t size_t
```

13.241.1.3 TASK

```
#define TASK(
             M,
             I)
```

Value:

```
{I;}
```

13.241.1.4 WAIT

```
#define WAIT
```

13.241.1.5 CHECK_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

13.241.1.6 BARRIER

```
#define BARRIER
```

13.241.1.7 PAR_BLOCK

```
#define PAR_BLOCK
```

13.241.1.8 SYNCH_GROUP

```
#define SYNCH_GROUP(
              Args...)
```

Value:

```
{{Args};}
```

13.241.1.9 THREAD_INDEX

```
#define THREAD_INDEX 0
```

13.241.1.10 NUM_THREADS

```
#define NUM_THREADS 1
```

13.241.1.11 SET_THREADS

```
#define SET_THREADS(
              num_threads)
```

Value:

```
{}
```

13.241.1.12 MAX_THREADS

```
#define MAX_THREADS 1
```

13.241.1.13 READ

```
#define READ(
              Args...)
```

13.241.1.14 WRITE

```
#define WRITE(
              Args...)
```

13.241.1.15 READWRITE

```
#define READWRITE(
              Args...)
```

13.241.1.16 CONSTREFERENCE

```
#define CONSTREFERENCE(
              ...)
```

13.241.1.17 VALUE

```
#define VALUE(
              ...)
```

13.241.1.18 BEGIN_PARALLEL_MAIN

```
#define BEGIN_PARALLEL_MAIN(
              Args...)
```

Value:

```
int main(Args)  {
```

13.241.1.19 END_PARALLEL_MAIN

```
#define END_PARALLEL_MAIN(
              void)
```

Value:

```
return 0; }
```

13.241.1.20 FORBLOCK1D

```
#define FORBLOCK1D(
              iter,
              m,
              Helper,
              Args...)
```

Value:

```
{ FFLAS::ForStrategy1D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
      decltype(Helper)::Param> iter(m, Helper); \
    for(iter.initialize(); !iter.isTerminated(); ++iter)            \
    {Args;} }
```

13.241.1.21 FOR1D

```
#define FOR1D(
              i,
              m,
              Helper,
              Args...)
```

Value:

```
FORBLOCK1D(_internal_iterator, m, Helper,                           \
           for(auto i=_internal_iterator.begin(); i!=_internal_iterator.end(); ++i) \
           { Args; })
```

13.241.1.22 PARFORBLOCK1D

```
#define PARFORBLOCK1D(
              iter,
              m,
              Helper,
              Args...)
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)     \
{ Args; }
```

13.241.1.23 PARFOR1D

```
#define PARFOR1D(
              iter,
              m,
              Helper,
              Args...)
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)     \
{ Args; }
```

13.241.1.24 FORBLOCK2D

```
#define FORBLOCK2D(
                iter,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
{ FFLAS::ForStrategy2D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
      decltype(Helper)::Param> iter(m,n,Helper); \
    for(iter.initialize(); !iter.isTerminated(); ++iter)             \
    { Args; } }
```

13.241.1.25 FOR2D

```
#define FOR2D(
                i,
                j,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
FORBLOCK2D(_internal_iterator, m, n, Helper,                          \
           for(auto i=_internal_iterator.ibegin(); i!=_internal_iterator.iend(); ++i) \
           for(auto j=_internal_iterator.jbegin(); j!=_internal_iterator.jend(); ++j) \
           { Args; })
```

13.241.1.26 PARFORBLOCK2D

```
#define PARFORBLOCK2D(
                iter,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
FORBLOCK2D(iter, m, n, Helper, Args)
```

13.241.1.27 PARFOR2D

```
#define PARFOR2D(
                i,
                j,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
FOR2D(i, j, m, n, Helper, Args)
```

13.241.1.28 COMMA

```
#define COMMA ,
```

13.241.1.29 MODE

```
#define MODE(
                ...)
```

Value:

```
__VA_ARGS__
```

13.241.1.30 RETURNPARAM

```
#define RETURNPARAM(
              f,
              P1,
              Args...)
```

Value:

```
P1=f(Args)
```

13.241.1.31 NUMARGS

```
#define NUMARGS(
              ...)
```

Value:

```
PP_NARG_(__VA_ARGS__,PP_RSEQ_N())
```

13.241.1.32 PP_NARG_

```
#define PP_NARG_(
              ...)
```

Value:

```
PP_ARG_N(__VA_ARGS__)
```

13.241.1.33 PP_ARG_N

```
#define PP_ARG_N(
              _1,
              _2,
              _3,
              _4,
              _5,
              _6,
              _7,
              _8,
              _9,
              _10,
              _11,
              _12,
              _13,
              _14,
              _15,
              _16,
              _17,
              _18,
              _19,
              _20,
              _21,
              _22,
              _23,
              _24,
              _25,
              _26,
              _27,
              _28,
              _29,
              _30,
              _31,
              _32,
              _33,
```

```
                _34,
                _35,
                _36,
                _37,
                _38,
                _39,
                _40,
                _41,
                _42,
                _43,
                _44,
                _45,
                _46,
                _47,
                _48,
                _49,
                _50,
                _51,
                _52,
                _53,
                _54,
                _55,
                _56,
                _57,
                _58,
                _59,
                _60,
                _61,
                _62,
                _63,
                N,
                ...)
```

Value:

```
N
```

13.241.1.34 PP_RSEQ_N

```
#define PP_RSEQ_N()
```

Value:

```
63,62,61,60,           \
59,58,57,56,55,54,53,52,51,50, \
49,48,47,46,45,44,43,42,41,40, \
39,38,37,36,35,34,33,32,31,30, \
29,28,27,26,25,24,23,22,21,20, \
19,18,17,16,15,14,13,12,11,10, \
9,8,7,6,5,4,3,2,1,0
```

13.241.1.35 NOSPLIT

```
#define NOSPLIT()
```

Value:

```
FFLAS::ParSeqHelper::Sequential()
```

13.241.1.36 splitting_0

```
#define splitting_0()
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>()
```

13.241.1.37 splitting_1

```
#define splitting_1(
              a)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>(a)
```

13.241.1.38 splitting_2

```
#define splitting_2(
              a,
              c)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)
```

13.241.1.39 splitting_3

```
#define splitting_3(
              a,
              b,
              c)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<b,c>(a)
```

13.241.1.40 splitt

```
#define splitt(
              _1,
              _2,
              _3,
              NAME,
              ...)
```

Value:

```
NAME
```

13.241.1.41 SPLITTER

```
#define SPLITTER(
              ...)
```

Value:

```
splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)
```

13.242 pfgemm_variants.inl File Reference

Namespaces

- namespace FFLAS

Functions

- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename

Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_↩ t ldb, const typename Field::Element beta, typename Field::Element ∗C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive > > &H)

- template<class Field, class AlgoT, class FieldTrait>
  Field::Element ∗ pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_↩ t ldb, const typename Field::Element beta, typename Field::Element ∗C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element ∗ pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_↩ t ldb, const typename Field::Element beta, typename Field::Element ∗C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)
- template<class Field, class AlgoT, class FieldTrait>
  Field::Element ∗ pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace > > &H)

13.243 pfgemv.inl File Reference

Namespaces

- namespace FFLAS

Functions

- template<class Field, class AlgoT, class FieldTrait>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_↩ t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)
- template<class Field, class AlgoT, class FieldTrait, class Cut>
  Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_↩ t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)

13.244 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

13.245 args-parser.h File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/givinteger.h>
#include <givaro/givprint.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cstring>
#include <list>
#include <stdlib.h>
```

Data Structures

- struct Argument

Namespaces

- namespace FFLAS

Macros

- #define TYPE_BOOL TYPE_NONE
- #define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
- #define type_integer long int

Enumerations

- enum ArgumentType {
  TYPE_NONE , TYPE_INT , TYPE_UINT64 , TYPE_LONGLONG ,
  TYPE_INTEGER , TYPE_DOUBLE , TYPE_INTLIST , TYPE_STR }

Functions

- void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)
- void printHelpMessage (const char *program, Argument *args, bool printDefaults=false)
- Argument * findArgument (Argument *args, char c)
- int getListArgs (std::list< int > &outlist, std::string &instring)

  transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}
- char * getArgumentValue (int argc, char **argv, int i)

  Get the value of an argument and avoid core dump when no value was given after an argument.
- std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)

  writes the values of all arguments, preceded by the programName

13.245.1 Macro Definition Documentation

13.245.1.1 TYPE_BOOL

```
#define TYPE_BOOL TYPE_NONE
```

13.245.1.2 END_OF_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

13.245.1.3 type_integer

```
#define type_integer long int
```

13.245.2 Enumeration Type Documentation

13.245.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

TYPE_NONE	
TYPE_INT	
TYPE_UINT64	
TYPE_LONGLONG	
TYPE_INTEGER	
TYPE_DOUBLE	
TYPE_INTLIST	
TYPE_STR	

13.245.3 Function Documentation

13.245.3.1 printHelpMessage()

```
void printHelpMessage (
            const char * program,
            Argument * args,
            bool printDefaults = false)
```

13.245.3.2 findArgument()

```
Argument * findArgument (
            Argument * args,
            char c)
```

13.245.3.3 getListArgs()

```
int getListArgs (
            std::list< int > & outlist,
            std::string & instring)
```

transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}

Parameters

outlist	list once converted
instring	list to be converted

Returns

status message.

13.246 bit_manipulation.h File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Macros

- #define __has_builtin(x)

Functions

- int32_t clz (uint64_t val)
- int32_t clz (uint32_t val)
- int32_t ctz (uint32_t val)
- int32_t ctz (uint64_t val)

13.246.1 Macro Definition Documentation

13.246.1.1 __has_builtin

```
#define __has_builtin(
              x)
```

Value:

```
0
```

13.246.2 Function Documentation

13.246.2.1 clz() [1/2]

```
int32_t clz (
             uint64_t val)  [inline]
```

13.246.2.2 clz() [2/2]

```
int32_t clz (
             uint32_t val)  [inline]
```

13.246.2.3 ctz() [1/2]

```
int32_t ctz (
             uint32_t val)  [inline]
```

13.246.2.4 ctz() [2/2]

```
int32_t ctz (
             uint64_t val)  [inline]
```

13.247 cast.h File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- template<class T, class CT = const T>
  T fflas_const_cast (CT x)

13.248 debug.h File Reference

Various utilities for debugging.

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <iostream>
#include <sstream>
#include <cmath>
#include <stdexcept>
```

Data Structures

- class Failure

  A precondtion failed.

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define FFLASFFPACK_check(check)
- #define FFLASFFPACK_abort(msg)

Functions

- Failure & failure ()
- template<class T>
  bool isOdd (const T &a)
- bool isOdd (const float &a)
- bool isOdd (const double &a)

13.248.1 Detailed Description

Various utilities for debugging.

Todo we should put vector printing elsewhere.

13.248.2 Macro Definition Documentation

13.248.2.1 FFLASFFPACK_check

```
#define FFLASFFPACK_check(
                check)
```

Value:

```
if (!(check)) {\
    FFPACK::failure()(__func__, __FILE__, __LINE__, #check); \
    throw std::runtime_error(#check); \
}
```

13.248.2.2 FFLASFFPACK_abort

```
#define FFLASFFPACK_abort(
                msg)
```

Value:

```
{\
    FFPACK::failure()(__func__, __FILE__, __LINE__, msg); \
    throw std::runtime_error(msg); \
}
```

13.249 fflas_intrinsic.h File Reference

13.250 fflas_io.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include <fstream>
#include "fflas-ffpack/fflas/fflas.h"
```

```
#include "fflas_memory.h"
```

Namespaces

- namespace FFLAS

Enumerations

- enum FFLAS_FORMAT {
  FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,
  FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }

Functions

- template<class Field>
  std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format, bool column_major)

  WriteMatrix: write a matrix to an output stream.
- void preamble (std::ifstream &ifs, FFLAS_FORMAT &format)
- template<class Field>
  Field::Element_ptr ReadMatrix (std::ifstream &ifs, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)

  ReadMatrix: read a matrix from an input stream.
- template<class Field>
  Field::Element_ptr ReadMatrix (const std::string &matrix_file, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)

  ReadMatrix: read a matrix from a file.
- template<class Field>
  void WriteMatrix (std::string &matrix_file, const Field &F, int m, int n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format=FflasDense, bool column_major=false)

  WriteMatrix: write a matrix to a file.
- std::ostream & WritePermutation (std::ostream &c, const size_t *P, size_t N)

  WritePermutation: write a permutation matrix to an output stream.

13.251 fflas_memory.h File Reference

```
#include "fflas-ffpack/utils/align-allocator.h"
#include <givaro/givinteger.h>
```

Namespaces

- namespace FFLAS

Functions

- template<class Element>
  bool alignable ()
- template<> bool alignable< Givaro::Integer * > ()
- template<class Field>
  Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)
- template<class Field>
  Field::Element_ptr fflas_new (const Field &F, const size_t m, const size_t n, const Alignment align=Alignment::DEFAULT)
- template<class Element>
  Element * fflas_new (const size_t m, const Alignment align=Alignment::DEFAULT)

- template<class Element_ptr>
  void fflas_delete (Element_ptr A)
- template<class Ptr, class ... Args>
  void fflas_delete (Ptr p, Args ... args)
- void prefetch (const int64_t *)
- void getTLBSize (int &tlb)
- void queryCacheSizes (int &l1, int &l2, int &l3)
- int queryL1CacheSize ()
- int queryTopLevelCacheSize ()

13.252 fflas_randommatrix.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- template<class Field, class RandIter>
  Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random non-zero Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)

  Random non-zero Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix.
- template<class Field>
  Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda)

  Random Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Triangular Matrix.
- template<class Field>
  Field::Element_ptr RandomTriangularMatrix (const Field &F, size_t m, size_t n, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_DIAG Diag, bool nonsingular, typename Field::Element_ptr A, size_t lda)

  Random Triangular Matrix.
- size_t RandInt (size_t a, size_t b)

- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrix (const Field &F, size_t n, bool nonsingular, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Symmetric Matrix.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed rank.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRank (const Field &F, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed rank.
- size_t * RandomIndexSubset (size_t N, size_t R, size_t *P)

  Pick uniformly at random a sequence of R distinct elements from the set $\{0, \ldots, N-1\}$ using Knuth's shuffle.
- size_t * RandomPermutation (size_t N, size_t *P)

  Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.
- void RandomRankProfileMatrix (size_t M, size_t N, size_t R, size_t *rows, size_t *cols)

  Pick uniformly at random an R-subpermutation of dimension M x N : a matrix with only R non-zeros equal to one, in a random rook placement.
- void swapval (size_t k, size_t N, size_t *P, size_t val)
- void RandomSymmetricRankProfileMatrix (size_t N, size_t R, size_t *rows, size_t *cols)

  Pick uniformly at random a symmetric R-subpermutation of dimension N x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.
- void RandomLTQSRankProfileMatrix (size_t n, size_t r, size_t t, size_t *rows, size_t *cols)
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)

  Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)

  Random Matrix with prescribed rank and rank profile matrix Creates an m x n matrix with random entries and rank r.
- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP, RandIter &G)

  Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an n x n symmetric matrix with random entries and rank r.
- template<class Field>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, const size_t *RRP, const size_t *CRP)

  Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an n x n symmetric matrix with random entries and rank r.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field>
  Field::Element_ptr RandomMatrixWithRankandRandomRPM (const Field &F, size_t M, size_t N, size_t R, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed rank, with random rank profile matrix Creates an m x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field, class RandIter>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda, RandIter &G)

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

- template<class Field>
  Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda)

  Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x n matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- template<class Field>
  Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (Field &F, size_t n, size_t r, size_t t, typename Field::Element_ptr A, size_t lda, RandIter &G)

13.253 flimits.h File Reference

```
#include <climits>
#include <limits>
#include <type_traits>
#include <givaro/givinteger.h>
```

Data Structures

- struct limits< unsigned char >
- struct limits< signed char >
- struct limits< char >
- struct limits< unsigned short int >
- struct limits< short int >
- struct limits< unsigned int >
- struct limits< int >
- struct limits< unsigned long >
- struct limits< long >
- struct limits< unsigned long long >
- struct limits< long long >
- struct limits< float >
- struct limits< double >
- struct limits< Givaro::Integer >
- struct limits< RecInt::ruint< K > >
- struct limits< RecInt::rint< K > >

Functions

- template<class T, class E>
  std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_range (E e)
- template<class T, class E>
  std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type in_range (E e)
- template<class T, class E>
  std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type in_range (E e)

13.253.1 Function Documentation

13.253.1.1 in_range() [1/3]

```
template<class T, class E>
std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_↩
range (
            E e)
```

13.253.1.2 in_range() [2/3]

```
template<class T, class E>
std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type
in_range (
            E e)
```

13.253.1.3 in_range() [3/3]

```
template<class T, class E>
std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type
in_range (
            E e)
```

13.254 Matio.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include "fflas_memory.h"
```

Functions

- template<class Field>
  Field::Element_ptr read_field (const Field &F, const char *mat_file, size_t *tni, size_t *tnj)
- template<class Field>
  std::ostream & write_field (const Field &F, std::ostream &c, typename Field::ConstElement_ptr E, int n, int m, int id, bool mapleFormat=false, bool column_major=false)

13.254.1 Function Documentation

13.254.1.1 read_field()

```
template<class Field>
Field::Element_ptr read_field (
            const Field & F,
            const char * mat_file,
            size_t * tni,
            size_t * tnj)
```

13.254.1.2 write_field()

```
template<class Field>
std::ostream & write_field (
            const Field & F,
            std::ostream & c,
            typename Field::ConstElement_ptr E,
            int n,
            int m,
```

```
            int id,
            bool mapleFormat = false,
            bool column_major = false)
```

13.255 test-utils.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include <givaro/givtimer.h>
#include <random>
#include <functional>
```

Namespaces

- namespace FFLAS
- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- uint64_t getSeed ()
- template<typename Field>
  Field * chooseField (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (Givaro::Integer q, uint64_t b, uint64_t seed)
- template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (Givaro::Integer q, uint64_t b, uint64_t seed)

13.256 timer.h File Reference

```
#include <time.h>
#include <givaro/givtimer.h>
```

Namespaces

- namespace FFLAS

Typedefs

- typedef Givaro::Timer Timer
- typedef Givaro::BaseTimer BaseTimer
- typedef Givaro::UserTimer UserTimer
- typedef Givaro::SysTimer SysTimer

13.257 cblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_CBLAS 1

Functions

- int main ()

13.257.1 Macro Definition Documentation

13.257.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.257.1.2 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

13.257.2 Function Documentation

13.257.2.1 main()

```
int main (
            void )
```

13.258 clapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_LAPACK 1
- #define __FFLASFFPACK_HAVE_CLAPACK 1

Functions

- int main ()

13.258.1 Macro Definition Documentation

13.258.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.258.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.258.1.3 __FFLASFFPACK_HAVE_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

13.258.2 Function Documentation

13.258.2.1 main()

```
int main (
            void )
```

13.259 cuda.C File Reference

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

Functions

- int main ()

13.259.1 Function Documentation

13.259.1.1 main()

```
int main (
            void )
```

13.260 fblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION

Functions

- void dgemm_ (const char *, const char *, const int *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)
- int main ()

13.260.1 Macro Definition Documentation

13.260.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.260.2 Function Documentation

13.260.2.1 dgemm_()

```
void dgemm_ (
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
```

```
            const double * ,
            const int * ,
            const double * ,
            double * ,
            const int * )
```

13.260.2.2 main()

```
int main (
            void )
```

13.261 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_LAPACK 1

Functions

- int main ()

13.261.1 Macro Definition Documentation

13.261.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.261.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.261.2 Function Documentation

13.261.2.1 main()

```
int main (
            void )
```

13.262 regression-check.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- bool check1 ()
- bool check2 ()
- bool check3 ()
- bool check4 ()
- bool checkZeroDimCharpoly ()
- bool checkZeroDimMinPoly ()
- bool gf2ModularBalanced ()
- int main ()

13.262.1 Function Documentation

13.262.1.1 check1()

```
bool check1 ()
```

13.262.1.2 check2()

```
bool check2 ()
```

13.262.1.3 check3()

```
bool check3 ()
```

13.262.1.4 check4()

```
bool check4 ()
```

13.262.1.5 checkZeroDimCharpoly()

```
bool checkZeroDimCharpoly ()
```

13.262.1.6 checkZeroDimMinPoly()

```
bool checkZeroDimMinPoly ()
```

13.262.1.7 gf2ModularBalanced()

```
bool gf2ModularBalanced ()
```

13.262.1.8 main()

```
int main (
            void )
```

13.263 test-charpoly-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Macros

- #define ENABLE_CHECKER_charpoly 1
- #define TIME_CHECKER_CHARPOLY 1

Functions

- template<class Field, class Polynomial>
  void printPolynomial (const Field &F, Polynomial &v)
- int main (int argc, char **argv)

13.263.1 Macro Definition Documentation

13.263.1.1 ENABLE_CHECKER_charpoly

```
#define ENABLE_CHECKER_charpoly 1
```

13.263.1.2 TIME_CHECKER_CHARPOLY

```
#define TIME_CHECKER_CHARPOLY 1
```

13.263.2 Function Documentation

13.263.2.1 printPolynomial()

```
template<class Field, class Polynomial>
void printPolynomial (
            const Field & F,
            Polynomial & v)
```

13.263.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.264 test-charpoly.C File Reference

```
#include <iostream>
#include <iomanip>
#include "givaro/modular.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
#include <chrono>
```

Functions

- template<class Field, class RandIter>
  bool launch_test (const Field &F, size_t n, typename Field::Element *A, size_t lda, size_t nbit, RandIter &G, FFPACK::FFPACK_CHARPOLY_TAG CT)
- template<class Field>
  bool run_with_field (const Givaro::Integer p, uint64_t bits, size_t n, std::string file, int variant, size_t iter, uint64_t seed)
- int main (int argc, char **argv)

13.264.1 Function Documentation

13.264.1.1 launch_test()

```
template<class Field, class RandIter>
bool launch_test (
            const Field & F,
            size_t n,
            typename Field::Element * A,
            size_t lda,
```

```
            size_t nbit,
            RandIter & G,
            FFPACK::FFPACK_CHARPOLY_TAG CT)
```

13.264.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            const Givaro::Integer p,
            uint64_t bits,
            size_t n,
            std::string file,
            int variant,
            size_t iter,
            uint64_t seed)
```

13.264.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.265 test-compressQ.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <list>
#include <vector>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Typedefs

- typedef Givaro::Modular< double > Field

Functions

- template<class T>
  std::ostream & printvect (std::ostream &o, vector< T > &vect)
- int main (int argc, char **argv)

13.265.1 Typedef Documentation

13.265.1.1 Field

```
typedef Givaro::Modular<double> Field
```

13.265.2 Function Documentation

13.265.2.1 printvect()

```
template<class T>
std::ostream & printvect (
            std::ostream & o,
            vector< T > & vect)
```

Bug does not belong here

13.265.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.266 test-det-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

Macros

- #define ENABLE_CHECKER_Det 1
- #define TIME_CHECKER_Det 1

Functions

- int main (int argc, char **argv)

13.266.1 Macro Definition Documentation

13.266.1.1 ENABLE_CHECKER_Det

```
#define ENABLE_CHECKER_Det 1
```

13.266.1.2 TIME_CHECKER_Det

```
#define TIME_CHECKER_Det 1
```

13.266.2 Function Documentation

13.266.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.267 test-det.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

Functions

- template<class Field, class RandIter>
  bool test_det (Field &F, size_t n, int iter, RandIter &G)
- int main (int argc, char **argv)

13.267.1 Function Documentation

13.267.1.1 test_det()

```
template<class Field, class RandIter>
bool test_det (
             Field & F,
             size_t n,
             int iter,
             RandIter & G)
```

Todo test with stride

13.267.1.2 main()

```
int main (
             int argc,
             char ** argv)
```

13.268 test-echelon.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <iomanip>
#include <givaro/modular-balanced.h>
#include <givaro/udl.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include <random>
#include <chrono>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
- #define __FFLASFFPACK_PLUQ_THRESHOLD 25

Functions

- template<class Field, class RandIter>
  bool test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)
- template<class Field, class RandIter>
  bool test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)
- template<class Field, class RandIter>
  bool test_redcolechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

- template<class Field, class RandIter>
  bool test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG Lu↩
  Tag, RandIter &G, bool par)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.268.1 Macro Definition Documentation

13.268.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.268.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

13.268.1.3 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

13.268.2 Function Documentation

13.268.2.1 test_colechelon()

```
template<class Field, class RandIter>
bool test_colechelon (
             Field & F,
             size_t m,
             size_t n,
             size_t r,
             size_t iters,
             FFPACK::FFPACK_LU_TAG LuTag,
             RandIter & G,
             bool par)
```

Todo check lda

13.268.2.2 test_rowechelon()

```
template<class Field, class RandIter>
bool test_rowechelon (
             Field & F,
             size_t m,
             size_t n,
             size_t r,
             size_t iters,
             FFPACK::FFPACK_LU_TAG LuTag,
             RandIter & G,
             bool par)
```

Todo check lda

13.268.2.3 test_redcolechelon()

```
template<class Field, class RandIter>
bool test_redcolechelon (
             Field & F,
```

```
            size_t m,
            size_t n,
            size_t r,
            size_t iters,
            FFPACK::FFPACK_LU_TAG LuTag,
            RandIter & G,
            bool par)
```

Todo check lda

13.268.2.4 test_redrowechelon()

```
template<class Field, class RandIter>
bool test_redrowechelon (
            Field & F,
            size_t m,
            size_t n,
            size_t r,
            size_t iters,
            FFPACK::FFPACK_LU_TAG LuTag,
            RandIter & G,
            bool par)
```

Todo check lda

13.268.2.5 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t m,
            size_t n,
            size_t r,
            size_t iters,
            uint64_t seed)
```

13.268.2.6 main()

```
int main (
            int argc,
            char ** argv)
```

13.269 test-fadd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

Functions

- template<class Field>
  bool test_fadd (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- template<class Field>
  bool test_faddin (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- template<class Field>
  bool test_fsub (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- template<class Field>
  bool test_fsubin (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- int main (int ac, char **av)

13.269.1 Function Documentation

13.269.1.1 test_fadd()

```
template<class Field>
bool test_fadd (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.269.1.2 test_faddin()

```
template<class Field>
bool test_faddin (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.269.1.3 test_fsub()

```
template<class Field>
bool test_fsub (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.269.1.4 test_fsubin()

```
template<class Field>
bool test_fsubin (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.269.1.5 main()

```
int main (
            int ac,
            char ** av)
```

13.270 test-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include <givaro/zring.h>
#include <givaro/modular.h>
#include <random>
#include <chrono>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field>
  bool check_fdot (const Field &F, size_t n, typename Field::ConstElement_ptr a, size_t inca, typename Field::ConstElement_ptr b, size_t incb)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t BS, size_t n, size_t iters, uint64_t seed)
- bool run_with_Integer (size_t BS, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.270.1 Macro Definition Documentation

13.270.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.270.2 Function Documentation

13.270.2.1 check_fdot()

```
template<typename Field>
bool check_fdot (
            const Field & F,
            size_t n,
            typename Field::ConstElement_ptr a,
            size_t inca,
            typename Field::ConstElement_ptr b,
            size_t incb)
```

13.270.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
```

```
            Givaro::Integer q,
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.270.2.3 run_with_Integer()

```
bool run_with_Integer (
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.270.2.4 main()

```
int main (
            int argc,
            char ** argv)
```

13.271 test-fgemm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<class Field, class RandIter>
  bool launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, int m, int n, int k, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.271.1 Macro Definition Documentation

13.271.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.271.2 Function Documentation

13.271.2.1 launch_MM_dispatch()

```
template<class Field, class RandIter>
bool launch_MM_dispatch (
            const Field & F,
            const int mm,
            const int nn,
            const int kk,
```

```
            const typename Field::Element alpha,
            const typename Field::Element beta,
            const size_t iters,
            RandIter & G)
```

Bug test for ldX equal

Bug

Bug test for transpo

Bug

Todo does nbw actually do nbw recursive calls and then call blas (check ?) ?

13.271.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            int m,
            int n,
            int k,
            size_t iters,
            uint64_t seed)
```

13.271.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.272 test-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
```

Macros

- #define ENABLE_CHECKER_fgemm 1

Functions

- template<class Field>
  bool check_MM (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, enum FFLAS_TRANSPOSE &tb, const size_t m, const size_t n, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr B, size_t ldb, const typename Field::Element &beta, const typename Field::Element_ptr C, size_t ldc)

- template<class Field, class RandIter>
  bool launch_MM (const Field &F, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t ldc, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t ldb, enum FFLAS_TRANSPOSE tb, size_t iters, int nbw, bool par, RandIter &G)
- template<class Field, class RandIter>
  bool launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const int nbw, const bool par, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, int m, int n, int k, int nbw, size_t iters, bool par, size_t seed)
- int main (int argc, char **argv)

13.272.1 Macro Definition Documentation

13.272.1.1 ENABLE_CHECKER_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

13.272.2 Function Documentation

13.272.2.1 check_MM()

```
template<class Field>
bool check_MM (
             const Field & F,
             const typename Field::Element_ptr Cd,
             enum FFLAS_TRANSPOSE & ta,
             enum FFLAS_TRANSPOSE & tb,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element & alpha,
             const typename Field::Element_ptr A,
             size_t lda,
             const typename Field::Element_ptr B,
             size_t ldb,
             const typename Field::Element & beta,
             const typename Field::Element_ptr C,
             size_t ldc)
```

13.272.2.2 launch_MM()

```
template<class Field, class RandIter>
bool launch_MM (
             const Field & F,
             const size_t m,
             const size_t n,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::Element beta,
             const size_t ldc,
             const size_t lda,
             enum FFLAS_TRANSPOSE ta,
             const size_t ldb,
             enum FFLAS_TRANSPOSE tb,
             size_t iters,
             int nbw,
             bool par,
             RandIter & G)
```

13.272.2.3 launch_MM_dispatch()

```
template<class Field, class RandIter>
bool launch_MM_dispatch (
             const Field & F,
             const int mm,
             const int nn,
             const int kk,
             const typename Field::Element alpha,
             const typename Field::Element beta,
             const size_t iters,
             const int nbw,
             const bool par,
             RandIter & G)
```

Bug test for ldX equal

Bug

Bug test for transpo

Bug

Todo does nbw actually do nbw recursive calls and then call blas (check ?) ?

13.272.2.4 run_with_field()

```
template<class Field>
bool run_with_field (
             Givaro::Integer q,
             uint64_t b,
             int m,
             int n,
             int k,
             int nbw,
             size_t iters,
             bool par,
             size_t seed)
```

13.272.2.5 main()

```
int main (
             int argc,
             char ** argv)
```

13.273 test-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

Functions

- template<class Field>
  bool check_MV (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, const size_t m, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr X, size_t incX, const typename Field::Element &beta, const typename Field::Element_ptr Y, size_t incY)
- template<class Field, class RandIter>
  bool launch_MV (const Field &F, const size_t m, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t incX, const size_t incY, size_t iters, bool par, RandIter &G)
- template<class Field, class RandIter>
  bool launch_MV_dispatch (const Field &F, const int mm, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const bool par, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, int m, int k, size_t iters, bool par, uint64_t seed)
- int main (int argc, char **argv)

13.273.1 Function Documentation

13.273.1.1 check_MV()

```
template<class Field>
bool check_MV (
             const Field & F,
             const typename Field::Element_ptr Cd,
             enum FFLAS_TRANSPOSE & ta,
             const size_t m,
             const size_t k,
             const typename Field::Element & alpha,
             const typename Field::Element_ptr A,
             size_t lda,
             const typename Field::Element_ptr X,
             size_t incX,
             const typename Field::Element & beta,
             const typename Field::Element_ptr Y,
             size_t incY)
```

13.273.1.2 launch_MV()

```
template<class Field, class RandIter>
bool launch_MV (
             const Field & F,
             const size_t m,
             const size_t k,
             const typename Field::Element alpha,
             const typename Field::Element beta,
             const size_t lda,
             enum FFLAS_TRANSPOSE ta,
             const size_t incX,
             const size_t incY,
             size_t iters,
             bool par,
             RandIter & G)
```

13.273.1.3 launch_MV_dispatch()

```
template<class Field, class RandIter>
bool launch_MV_dispatch (
```

```
            const Field & F,
            const int mm,
            const int kk,
            const typename Field::Element alpha,
            const typename Field::Element beta,
            const size_t iters,
            const bool par,
            RandIter & G)
```

13.273.1.4 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            int m,
            int k,
            size_t iters,
            bool par,
            uint64_t seed)
```

13.273.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.274 test-fger.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-integral.h>
#include <givaro/modular-balanced.h>
#include <givaro/givintprime.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Macros

- #define TIME 1

Functions

- template<class Field>
  bool check_fger (const Field &F, const typename Field::Element_ptr Cd, const size_t m, const size_t n, const typename Field::Element &alpha, const typename Field::Element_ptr x, const size_t incx, const typename Field::Element_ptr y, const size_t incy, const typename Field::Element_ptr C, const size_t ldc)
- template<class Field, class RandIter>
  bool launch_fger (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, const size_t ldc, const size_t inca, const size_t incb, size_t iters, RandIter &G)
- template<class Field, class RandIter>
  bool launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const size_t iters, RandIter &G)

- template<class Field>
  bool run_with_field (int64_t q, uint64_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.274.1 Macro Definition Documentation

13.274.1.1 TIME

```
#define TIME 1
```

13.274.2 Function Documentation

13.274.2.1 check_fger()

```
template<class Field>
bool check_fger (
              const Field & F,
              const typename Field::Element_ptr Cd,
              const size_t m,
              const size_t n,
              const typename Field::Element & alpha,
              const typename Field::Element_ptr x,
              const size_t incx,
              const typename Field::Element_ptr y,
              const size_t incy,
              const typename Field::Element_ptr C,
              const size_t ldc)
```

13.274.2.2 launch_fger()

```
template<class Field, class RandIter>
bool launch_fger (
              const Field & F,
              const size_t m,
              const size_t n,
              const typename Field::Element alpha,
              const size_t ldc,
              const size_t inca,
              const size_t incb,
              size_t iters,
              RandIter & G)
```

13.274.2.3 launch_fger_dispatch()

```
template<class Field, class RandIter>
bool launch_fger_dispatch (
              const Field & F,
              const size_t nn,
              const typename Field::Element alpha,
              const size_t iters,
              RandIter & G)
```

Bug test for incx equal

Bug

Bug test for transpo

Bug

Todo does nbw actually do nbw recursive calls and then call blas (check ?) ?

13.274.2.4 run_with_field()

```
template<class Field>
bool run_with_field (
            int64_t q,
            uint64_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.274.2.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.275 test-fgesv.C File Reference

```
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

Functions

- template<class Field, class RandIter>
  bool test_square_fgesv (Field &F, FFLAS_SIDE side, string fileA, string fileB, size_t m, size_t k, size_t r, RandIter &G)
- template<class Field, class RandIter>
  bool test_rect_fgesv (Field &F, FFLAS_SIDE side, string fileA, string fileB, size_t m, size_t n, size_t k, size_t r, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t k, size_t r, size_t iters, string fileA, string fileB, uint64_t &seed)
- int main (int argc, char **argv)

13.275.1 Function Documentation

13.275.1.1 test_square_fgesv()

```
template<class Field, class RandIter>
bool test_square_fgesv (
            Field & F,
            FFLAS_SIDE side,
            string fileA,
            string fileB,
            size_t m,
            size_t k,
            size_t r,
            RandIter & G)
```

13.275.1.2 test_rect_fgesv()

```
template<class Field, class RandIter>
bool test_rect_fgesv (
```

```
            Field & F,
            FFLAS_SIDE side,
            string fileA,
            string fileB,
            size_t m,
            size_t n,
            size_t k,
            size_t r,
            RandIter & G)
```

13.275.1.3 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t m,
            size_t n,
            size_t k,
            size_t r,
            size_t iters,
            string fileA,
            string fileB,
            uint64_t & seed)
```

13.275.1.4 main()

```
int main (
            int argc,
            char ** argv)
```

13.276 test-finit.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
#include <random>
#include <chrono>
```

Functions

- template<class Field>
  bool test_freduce (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t k, size_t n, size_t iters, bool timing, uint64_t seed)
- int main (int ac, char **av)

13.276.1 Function Documentation

13.276.1.1 test_freduce()

```
template<class Field>
bool test_freduce (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.276.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t k,
            size_t n,
            size_t iters,
            bool timing,
            uint64_t seed)
```

13.276.1.3 main()

```
int main (
            int ac,
            char ** av)
```

13.277 test-fscal.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

Functions

- template<class Field, class RandIter>
  bool test_fscal (const Field &F, const typename Field::Element &alpha, size_t m, size_t k, size_t n, bool timing, RandIter &G)
- template<class Field>
  bool test_fscal (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- template<class Field, class RandIter>
  bool test_fscalin (const Field &F, const typename Field::Element &alpha, size_t m, size_t k, size_t n, bool timing, RandIter &G)
- template<class Field>
  bool test_fscalin (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)
- int main (int ac, char **av)

- template<class Field, class RandIter>
  Field::Element_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element_ptr A, size_t lda, RandIter &G)

  Random Matrix.

13.277.1 Function Documentation

13.277.1.1 test_fscal() [1/2]

```
template<class Field, class RandIter>
bool test_fscal (
             const Field & F,
             const typename Field::Element & alpha,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             RandIter & G)
```

13.277.1.2 test_fscal() [2/2]

```
template<class Field>
bool test_fscal (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.277.1.3 test_fscalin() [1/2]

```
template<class Field, class RandIter>
bool test_fscalin (
             const Field & F,
             const typename Field::Element & alpha,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             RandIter & G)
```

13.277.1.4 test_fscalin() [2/2]

```
template<class Field>
bool test_fscalin (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.277.1.5 main()

```
int main (
             int ac,
             char ** av)
```

13.277.1.6 RandomMatrix()

```
template<class Field, class RandIter>
Field::Element_ptr RandomMatrix (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

Random Matrix.
Creates a m x n matrix with random entries.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	*A*	the matrix (preallocated to at least m x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

13.278 test-fsyr2k.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_fsyr2k (const Field &F, size_t n, size_t k, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t k, int a, int c, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.278.1 Macro Definition Documentation

13.278.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.278.2 Function Documentation

13.278.2.1 check_fsyr2k()

```
template<typename Field, class RandIter>
bool check_fsyr2k (
            const Field & F,
            size_t n,
            size_t k,
            const typename Field::Element & alpha,
            const typename Field::Element & beta,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_TRANSPOSE trans,
            RandIter & Rand)
```

13.278.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t k,
            int a,
            int c,
            size_t iters,
            uint64_t seed)
```

13.278.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.279 test-fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_fsyrk (const Field &F, size_t n, size_t k, size_t w, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, RandIter &Rand)

- template<typename Field, class RandIter>
  bool check_fsyrk_diag (const Field &F, size_t n, size_t k, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, RandIter &Rand)
- template<typename Field, class RandIter>
  bool check_fsyrk_bkdiag (const Field &F, size_t n, size_t k, const typename Field::Element &alpha, const typename Field::Element &beta, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, RandIter &Rand)
- template<class Field, class RandIter>
  bool check_computeS1S2 (const Field &F, size_t N, size_t K, FFLAS_TRANSPOSE trans, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t k, size_t w, int a, int c, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.279.1 Macro Definition Documentation

13.279.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.279.2 Function Documentation

13.279.2.1 check_fsyrk()

```
template<typename Field, class RandIter>
bool check_fsyrk (
             const Field & F,
             size_t n,
             size_t k,
             size_t w,
             const typename Field::Element & alpha,
             const typename Field::Element & beta,
             FFLAS::FFLAS_UPLO uplo,
             FFLAS::FFLAS_TRANSPOSE trans,
             RandIter & Rand)
```

13.279.2.2 check_fsyrk_diag()

```
template<typename Field, class RandIter>
bool check_fsyrk_diag (
             const Field & F,
             size_t n,
             size_t k,
             const typename Field::Element & alpha,
             const typename Field::Element & beta,
             FFLAS::FFLAS_UPLO uplo,
             FFLAS::FFLAS_TRANSPOSE trans,
             RandIter & Rand)
```

13.279.2.3 check_fsyrk_bkdiag()

```
template<typename Field, class RandIter>
bool check_fsyrk_bkdiag (
             const Field & F,
             size_t n,
             size_t k,
             const typename Field::Element & alpha,
             const typename Field::Element & beta,
             FFLAS_UPLO uplo,
```

```
            FFLAS_TRANSPOSE trans,
            RandIter & Rand)
```

13.279.2.4 check_computeS1S2()

```
template<class Field, class RandIter>
bool check_computeS1S2 (
            const Field & F,
            size_t N,
            size_t K,
            FFLAS_TRANSPOSE trans,
            RandIter & G)
```

13.279.2.5 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t k,
            size_t w,
            int a,
            int c,
            size_t iters,
            uint64_t seed)
```

13.279.2.6 main()

```
int main (
            int argc,
            char ** argv)
```

13.280 test-fsytrf.C File Reference

```
#include <iostream>
#include <iterator>
#include <vector>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/utils/test-utils.h"
```

Functions

- template<typename T>
  std::ostream & operator<< (std::ostream &os, const std::vector< T > &x)
- template<class Field, class RandIter>
  bool test_RPM_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, size_t r, RandIter &G, size_t threshold)
- template<class Field, class RandIter>
  bool test_generic_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, RandIter &G, size_t threshold)

- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t iters, string file, size_t threshold, uint64_t &seed)
- int main (int argc, char **argv)

13.280.1 Function Documentation

13.280.1.1 operator<<()

```
template<typename T>
std::ostream & operator<< (
             std::ostream & os,
             const std::vector< T > & x)
```

13.280.1.2 test_RPM_fsytrf()

```
template<class Field, class RandIter>
bool test_RPM_fsytrf (
             Field & F,
             FFLAS_UPLO uplo,
             string file,
             size_t n,
             size_t r,
             RandIter & G,
             size_t threshold)
```

13.280.1.3 test_generic_fsytrf()

```
template<class Field, class RandIter>
bool test_generic_fsytrf (
             Field & F,
             FFLAS_UPLO uplo,
             string file,
             size_t n,
             RandIter & G,
             size_t threshold)
```

13.280.1.4 run_with_field()

```
template<class Field>
bool run_with_field (
             Givaro::Integer q,
             uint64_t b,
             size_t n,
             size_t r,
             size_t iters,
             string file,
             size_t threshold,
             uint64_t & seed)
```

13.280.1.5 main()

```
int main (
             int argc,
             char ** argv)
```

13.281 test-ftrmm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL

Functions

- template<typename Field, class RandIter>
  bool check_ftrmm (const Field &F, size_t m, size_t n, const typename Field::Element &alpha, FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, FFLAS::FFLAS_DIAG diag, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t n, uint64_t a, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.281.1 Macro Definition Documentation

13.281.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.281.2 Function Documentation

13.281.2.1 check_ftrmm()

```
template<typename Field, class RandIter>
bool check_ftrmm (
            const Field & F,
            size_t m,
            size_t n,
            const typename Field::Element & alpha,
            FFLAS::FFLAS_SIDE side,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_TRANSPOSE trans,
            FFLAS::FFLAS_DIAG diag,
            RandIter & Rand)
```

13.281.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t n,
            uint64_t a,
            size_t iters,
            uint64_t seed)
```

13.281.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.282 test-ftrmv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <chrono>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrmv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.282.1 Macro Definition Documentation

13.282.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.282.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.282.2 Function Documentation

13.282.2.1 check_ftrmv()

```
template<typename Field, class RandIter>
bool check_ftrmv (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand)
```

13.282.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.282.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.283 test-ftrsm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- int main (int argc, char **argv)

13.283.1 Macro Definition Documentation

13.283.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.283.2 Function Documentation

13.283.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.284 test-ftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrsm (const Field &F, size_t m, size_t n, const typename Field::Element &alpha, FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, FFLAS::FFLAS_DIAG diag, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t n, uint64_t a, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.284.1 Macro Definition Documentation

13.284.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.284.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.284.2 Function Documentation

13.284.2.1 check_ftrsm()

```
template<typename Field, class RandIter>
bool check_ftrsm (
            const Field & F,
            size_t m,
            size_t n,
            const typename Field::Element & alpha,
            FFLAS::FFLAS_SIDE side,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_TRANSPOSE trans,
            FFLAS::FFLAS_DIAG diag,
            RandIter & Rand)
```

13.284.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t n,
            uint64_t a,
            size_t iters,
            uint64_t seed)
```

13.284.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.285 test-ftrssyr2k.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrssyr2k (const Field &F, size_t n, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_DIAG diagA, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.285.1 Macro Definition Documentation

13.285.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.285.2 Function Documentation

13.285.2.1 check_ftrssyr2k()

```
template<typename Field, class RandIter>
bool check_ftrssyr2k (
            const Field & F,
            size_t n,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_DIAG diagA,
            RandIter & Rand)
```

13.285.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
```

```
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.285.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.286 test-ftrstr.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrstr (const Field &F, size_t n, FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_DIAG diagA, FFLAS::FFLAS_DIAG diagB, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.286.1 Macro Definition Documentation

13.286.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.286.2 Function Documentation

13.286.2.1 check_ftrstr()

```
template<typename Field, class RandIter>
bool check_ftrstr (
            const Field & F,
            size_t n,
            FFLAS::FFLAS_SIDE side,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_DIAG diagA,
            FFLAS::FFLAS_DIAG diagB,
            RandIter & Rand)
```

13.286.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.286.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.287 test-ftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrsv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.287.1 Macro Definition Documentation

13.287.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.287.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.287.2 Function Documentation

13.287.2.1 check_ftrsv()

```
template<typename Field, class RandIter>
bool check_ftrsv (
```

```
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand)
```

13.287.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.287.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.288 test-ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL
- #define ENABLE_ALL_CHECKINGS 1

Functions

- template<typename Field, class RandIter>
  bool check_ftrtri (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_DIAG diag, RandIter &Rand)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.288.1 Macro Definition Documentation

13.288.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.288.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.288.2 Function Documentation

13.288.2.1 check_ftrtri()

```
template<typename Field, class RandIter>
bool check_ftrtri (
             const Field & F,
             size_t n,
             FFLAS_UPLO uplo,
             FFLAS_DIAG diag,
             RandIter & Rand)
```

13.288.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
             Givaro::Integer q,
             size_t b,
             size_t n,
             size_t iters,
             uint64_t seed)
```

13.288.2.3 main()

```
int main (
             int argc,
             char ** argv)
```

13.289 test-interfaces-c.c File Reference

```
#include <fflas-ffpack/interfaces/libs/fflas_c.h>
#include <fflas-ffpack/interfaces/libs/ffpack_c.h>
#include <stdlib.h>
#include <stdio.h>
```

Functions

- int main ()

13.289.1 Function Documentation

13.289.1.1 main()

```
int main (
             void )
```

13.290 test-invert-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- int main (int argc, char **argv)

13.290.1 Macro Definition Documentation

13.290.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.290.2 Function Documentation

13.290.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.291 test-io.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <random>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Data Structures

- struct CompactElement< Element >
- struct CompactElement< double >
- struct CompactElement< float >
- struct CompactElement< int64_t >
- struct CompactElement< int32_t >
- struct CompactElement< int16_t >

Functions

- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.291.1 Function Documentation

13.291.1.1 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
```

```
            uint64_t b,
            size_t m,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.291.1.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.292 test-lu.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

Macros

- #define BASECASE_K 37
- #define __FFLASFFPACK_SEQUENTIAL
- #define __LUDIVINE_CUTOFF 1

Functions

- template<class Field, FFLAS::FFLAS_DIAG diag, FFLAS_TRANSPOSE trans>
  bool test_LUdivine (const Field &F, typename Field::ConstElement_ptr A, size_t lda, size_t r, size_t m, size_t n)

  Tests the LUdivine routine.
- template<class Field, FFLAS_DIAG diag>
  bool verifPLUQ (const Field &F, typename Field::ConstElement_ptr A, size_t lda, typename Field::Element_ptr PLUQ, size_t ldpluq, size_t *P, size_t *Q, size_t m, size_t n, size_t R)

  Verifies that B = PLUQ where A stores [L\U].
- template<class Field, FFLAS_DIAG diag, class RandIter>
  bool test_pluq (const Field &F, typename Field::ConstElement_ptr A, size_t r, size_t m, size_t n, size_t lda, RandIter &G)

  Tests the LUdivine routine.
- template<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class RandIter>
  bool launch_test (const Field &F, size_t r, size_t m, size_t n, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

Variables

- Givaro::Timer tperm
- Givaro::Timer tgemm
- Givaro::Timer tBC

- Givaro::Timer ttrsm
- Givaro::Timer trest
- Givaro::Timer timtot
- size_t mvcnt = 0

13.292.1 Macro Definition Documentation

13.292.1.1 BASECASE_K

```
#define BASECASE_K 37
```

13.292.1.2 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.292.1.3 __LUDIVINE_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

13.292.2 Function Documentation

13.292.2.1 test_LUdivine()

```
template<class Field, FFLAS::FFLAS_DIAG diag, FFLAS_TRANSPOSE trans>
bool test_LUdivine (
             const Field & F,
             typename Field::ConstElement_ptr A,
             size_t lda,
             size_t r,
             size_t m,
             size_t n)
```

Tests the LUdivine routine.

Template Parameters

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.2 verifPLUQ()

```
template<class Field, FFLAS_DIAG diag>
bool verifPLUQ (
             const Field & F,
             typename Field::ConstElement_ptr A,
             size_t lda,
             typename Field::Element_ptr PLUQ,
             size_t ldpluq,
             size_t * P,
             size_t * Q,
             size_t m,
             size_t n,
             size_t R)
```

Verifies that B = PLUQ where A stores [L\U].

Template Parameters

Field	Field
Diag	Unit diagonal in U

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.3 test_pluq()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
bool test_pluq (
             const Field & F,
             typename Field::ConstElement_ptr A,
             size_t r,
             size_t m,
             size_t n,
             size_t lda,
             RandIter & G)
```

Tests the LUdivine routine.

Template Parameters

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.4 launch_test()

```
template<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class RandIter>
bool launch_test (
              const Field & F,
              size_t r,
              size_t m,
              size_t n,
              RandIter & G)
```

13.292.2.5 run_with_field()

```
template<class Field>
bool run_with_field (
              Givaro::Integer q,
              uint64_t b,
              size_t m,
              size_t n,
              size_t r,
              size_t iters,
              uint64_t seed)
```

13.292.2.6 main()

```
int main (
              int argc,
              char ** argv)
```

13.292.3 Variable Documentation

13.292.3.1 tperm

```
Givaro::Timer tperm
```

13.292.3.2 tgemm

```
Givaro::Timer tgemm
```

13.292.3.3 tBC

```
Givaro::Timer tBC
```

13.292.3.4 ttrsm

```
Givaro::Timer ttrsm
```

13.292.3.5 trest

```
Givaro::Timer trest
```

13.292.3.6 timtot

```
Givaro::Timer timtot
```

13.292.3.7 mvcnt

```
size_t mvcnt = 0
```

13.293 test-maxdelayeddim.C File Reference

```
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include <stdlib.h>
#include <stdio.h>
```

Macros

- #define MAX_WITH_SIZE_T(x)

Functions

- template<class Field>
  bool test (Givaro::Integer p, size_t kmax)
- int main ()

13.293.1 Macro Definition Documentation

13.293.1.1 MAX_WITH_SIZE_T

```
#define MAX_WITH_SIZE_T(
              x)
```

Value:

```
( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) <  x)? std::numeric_limits<size_t>::max() : x )
```

13.293.2 Function Documentation

13.293.2.1 test()

```
template<class Field>
bool test (
             Givaro::Integer p,
             size_t kmax)
```

13.293.2.2 main()

```
int main (
             void )
```

13.294 test-minpoly.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
```

```
#include <chrono>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/modular-integer.h>
#include <givaro/givpoly1factor.h>
#include <givaro/givpoly1.h>
```

Functions

- template<typename Field, class RandIter>
  bool check_minpoly (const Field &F, size_t n, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.294.1 Function Documentation

13.294.1.1 check_minpoly()

```
template<typename Field, class RandIter>
bool check_minpoly (
            const Field & F,
            size_t n,
            RandIter & G)
```

13.294.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.294.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.295 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

13.296 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (void)

13.296.1 Function Documentation

13.296.1.1 main()

```
int main (
            void )
```

13.297 test-nullspace.C File Reference

```
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/timer.h"
```

Functions

- template<class Field>
  std::string checkingMessage (const Field &F)
- template<class Field>
  Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (const Field &F, std::string file, size_t &m, size_t &n, size_t &lda, size_t &r, uint64_t seed)

  If file is not empty, read it and set m, n, lda and r.
- template<class Field>
  bool test_nullspace (Field &F, FFLAS::FFLAS_SIDE side, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, std::string file, uint64_t &seed)
- int main (int argc, char **argv)

13.297.1 Function Documentation

13.297.1.1 checkingMessage()

```
template<class Field>
std::string checkingMessage (
            const Field & F)
```

13.297.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
template<class Field>
Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (
            const Field & F,
            std::string file,
            size_t & m,
            size_t & n,
            size_t & lda,
            size_t & r,
            uint64_t seed)
```

If file is not empty, read it and set m, n, lda and r.

Otherwise, generate a random matrix of size m x n with random lda.

13.297.1.3 test_nullspace()

```
template<class Field>
bool test_nullspace (
            Field & F,
            FFLAS::FFLAS_SIDE side,
            size_t m,
            size_t n,
            size_t r,
            typename Field::Element_ptr A,
            size_t lda)
```

13.297.1.4 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t m,
            size_t n,
            size_t r,
            size_t iters,
            std::string file,
            uint64_t & seed)
```

13.297.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.298 test-permutations.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

Functions

- bool checkMonotonicApplyP (FFLAS_SIDE Side, FFLAS_TRANSPOSE trans, size_t *P, size_t N, size_t R)
- int main ()

Variables

- Givaro::Timer tperm
- Givaro::Timer tgemm
- Givaro::Timer tBC
- Givaro::Timer ttrsm
- Givaro::Timer trest
- Givaro::Timer timtot

13.298.1 Function Documentation

13.298.1.1 checkMonotonicApplyP()

```
bool checkMonotonicApplyP (
            FFLAS_SIDE Side,
            FFLAS_TRANSPOSE trans,
            size_t * P,
            size_t N,
            size_t R)
```

13.298.1.2 main()

```
int main (
            void )
```

13.298.2 Variable Documentation

13.298.2.1 tperm

```
Givaro::Timer tperm
```

13.298.2.2 tgemm

```
Givaro::Timer tgemm
```

13.298.2.3 tBC

```
Givaro::Timer tBC
```

13.298.2.4 ttrsm

```
Givaro::Timer ttrsm
```

13.298.2.5 trest

```
Givaro::Timer trest
```

13.298.2.6 timtot

```
Givaro::Timer timtot
```

13.299 test-pluq-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

Macros

- #define ENABLE_ALL_CHECKINGS 1

Functions

- int main (int argc, char **argv)

13.299.1 Macro Definition Documentation

13.299.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.299.2 Function Documentation

13.299.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.300 test-quasisep.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

Functions

- template<class Field, FFLAS_DIAG diag, class RandIter>
  bool test_BruhatGenerator (const Field &F, size_t n, size_t r, size_t t, typename Field::ConstElement_ptr A, size_t lda, typename Field::Element_ptr TS, size_t l, RandIter &G)
- template<class Field, FFLAS_DIAG diag, class RandIter>
  bool launch_test (const Field &F, size_t n, size_t r, size_t t, size_t l, RandIter &G)
- template<class Field, class RandGen>
  bool testLTQSRPM (const Field &F, size_t n, size_t r, size_t t, RandGen &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t t, size_t l, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.300.1 Function Documentation

13.300.1.1 test_BruhatGenerator()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
bool test_BruhatGenerator (
            const Field & F,
            size_t n,
            size_t r,
            size_t t,
            typename Field::ConstElement_ptr A,
            size_t lda,
            typename Field::Element_ptr TS,
            size_t l,
            RandIter & G)
```

13.300.1.2 launch_test()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
bool launch_test (
             const Field & F,
             size_t n,
             size_t r,
             size_t t,
             size_t l,
             RandIter & G)
```

13.300.1.3 testLTQSRPM()

```
template<class Field, class RandGen>
bool testLTQSRPM (
             const Field & F,
             size_t n,
             size_t r,
             size_t t,
             RandGen & G)
```

13.300.1.4 run_with_field()

```
template<class Field>
bool run_with_field (
             Givaro::Integer q,
             uint64_t b,
             size_t n,
             size_t r,
             size_t t,
             size_t l,
             size_t iters,
             uint64_t seed)
```

13.300.1.5 main()

```
int main (
             int argc,
             char ** argv)
```

13.301 test-rankprofiles.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <chrono>
```

Macros

- #define __FFLASFFPACK_SEQUENTIAL

Functions

- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed, bool par)
- int main (int argc, char **argv)

13.301.1 Macro Definition Documentation

13.301.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.301.2 Function Documentation

13.301.2.1 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t m,
            size_t n,
            size_t r,
            size_t iters,
            uint64_t seed,
            bool par)
```

13.301.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.302 test-rpm.C File Reference

```
#include <iostream>
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Functions

- bool checkRPM (size_t M, size_t N, size_t R)
- bool checkSymmetricRPM (size_t N, size_t R)
- int main (int argc, char **argv)

13.302.1 Function Documentation

13.302.1.1 checkRPM()

```
bool checkRPM (
            size_t M,
            size_t N,
            size_t R)
```

13.302.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
            size_t N,
            size_t R)
```

13.302.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.303 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <array>
#include <vector>
#include <random>
#include <string>
#include <functional>
#include <limits>
#include <type_traits>
#include <algorithm>
```

Data Structures

- struct ALL< true, v... >
- struct ALL< false, v... >
- struct ALL<>
- struct count_nonconst_lvalue_reference< T, O... >
- struct count_nonconst_lvalue_reference< T &, O... >
- struct count_nonconst_lvalue_reference< const T &, O... >
- struct count_nonconst_lvalue_reference<>
- struct is_all_same< T, Args... >
- struct is_all_same<>
- struct width< T >
- struct width< float >
- struct width< double >
- class TestOneMethod< Simd >
- struct ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >
- class ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestD
- struct ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >
- struct ScalFunctions< Element >
- class string

  STL class.
- class vector< T >

  STL class.
- class array< T >

  STL class.

Macros

- #define _TEST_ONE(K, f1, f2, r, n)
- #define TEST_ONE_OP(f)
- #define TEST_ONE_OP_WZ(f)
- #define TEST_IMPL(SIZE, Elt)

Functions

- template<typename Element>
  enable_if< is_integral< Element >::value, bool >::type check_eq (Element x, Element y)
- template<typename Element>
  enable_if< is_floating_point< Element >::value, bool >::type check_eq (Element x, Element y)
- template<typename Element>
  bool cmp (vector< Element > out_scal, vector< Element > out_simd)
- template<typename Ret, typename T>
  Ret eval_func_on_array (function< Ret()> f, array< T, 0 > &arr)
- template<typename T, typename... TArgs>
  void eval_func_on_array (function< void(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- template<typename Ret, typename T, typename... TArgs>
  Ret eval_func_on_array (function< Ret(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- template<typename E>
  std::ostream & operator<< (std::ostream &o, const vector< E > &V)
- template<typename Simd, typename Element>
  enable_if< is_floating_point< Element >::value, bool >::type test_impl_base ()
- template<typename Simd, typename Element>
  enable_if< is_integral< Element >::value, bool >::type test_impl_base ()
- template<typename Simd, typename Element>
  bool test_impl ()
- int main (int argc, char *argv[])

13.303.1 Macro Definition Documentation

13.303.1.1 _TEST_ONE

```
#define _TEST_ONE(
                K,
                f1,
                f2,
                r,
                n)
```

Value:

```
        do {     \
        K T(f1, f2, r, n);                  \
        bool b = T.writeResultLine();       \
        if (b == false)                     \
            T.writeDebugData();             \
        btest &= b;                         \
    } while (0)
```

13.303.1.2 TEST_ONE_OP

```
#define TEST_ONE_OP(
                f)
```

Value:

```
        _TEST_ONE(TestOneMethod<Simd>,              \
        function<decltype(Simd::f)>(Simd::f),                       \
        function<decltype(Scal::f)>(Scal::f),                       \
        function<decltype(Scal::genInputs)>(Scal::genInputs), #f)
```

13.303.1.3 TEST_ONE_OP_WZ

```
#define TEST_ONE_OP_WZ(
              f)
```

Value:

```
        _TEST_ONE(TestOneMethod<Simd>,                      \
        function<decltype(Simd::f)>(Simd::f),                                \
        function<decltype(Scal::f)>(Scal::f),                                \
        function<decltype(Scal::genInputsWithZero)>(Scal::genInputsWithZero),\
        #f " test with zero")
```

13.303.1.4 TEST_IMPL

```
#define TEST_IMPL(
              SIZE,
              Elt)
```

Value:

```
        do {       \
        pass &= test_impl<Simd##SIZE<Elt>, Elt>(); \
        cout « endl; \
    } while (0)
```

13.303.2 Function Documentation

13.303.2.1 check_eq() [1/2]

```
template<typename Element>
enable_if< is_integral< Element >::value, bool >::type check_eq (
              Element x,
              Element y)
```

13.303.2.2 check_eq() [2/2]

```
template<typename Element>
enable_if< is_floating_point< Element >::value, bool >::type check_eq (
              Element x,
              Element y)
```

13.303.2.3 cmp()

```
template<typename Element>
bool cmp (
              vector< Element > out_scal,
              vector< Element > out_simd)
```

13.303.2.4 eval_func_on_array() [1/3]

```
template<typename Ret, typename T>
Ret eval_func_on_array (
              function< Ret()> f,
              array< T, 0 > & arr)
```

13.303.2.5 eval_func_on_array() [2/3]

```
template<typename T, typename...  TArgs>
void eval_func_on_array (
              function< void(T, TArgs...)> f,
              array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr)
```

13.303.2.6 eval_func_on_array() [3/3]

```
template<typename Ret, typename T, typename...  TArgs>
Ret eval_func_on_array (
```

```
            function< Ret(T, TArgs...)> f,
            array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr)
```

13.303.2.7 operator<<()

```
template<typename E>
std::ostream & operator<< (
            std::ostream & o,
            const vector< E > & V)
```

13.303.2.8 test_impl_base() [1/2]

```
template<typename Simd, typename Element>
enable_if< is_floating_point< Element >::value, bool >::type test_impl_base ()
```

13.303.2.9 test_impl_base() [2/2]

```
template<typename Simd, typename Element>
enable_if< is_integral< Element >::value, bool >::type test_impl_base ()
```

13.303.2.10 test_impl()

```
template<typename Simd, typename Element>
bool test_impl ()
```

13.303.2.11 main()

```
int main (
            int argc,
            char * argv[])
```

13.304 test-solve.C File Reference

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

Functions

- template<typename Field, class RandIter>
  bool check_solve (const Field &F, size_t m, RandIter &Rand, bool isParallel)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.304.1 Function Documentation

13.304.1.1 check_solve()

```
template<typename Field, class RandIter>
bool check_solve (
            const Field & F,
```

```
            size_t m,
            RandIter & Rand,
            bool isParallel)
```

13.304.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t iters,
            uint64_t seed)
```

13.304.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.305 test-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

Data Structures

- class Test< Elt >

Functions

- int main (int argc, char **argv)

13.305.1 Function Documentation

13.305.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.306 101-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
```

```
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.306.1 Function Documentation

13.306.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.307 2x2-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.307.1 Function Documentation

13.307.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.308 2x2-ftrsv.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
#include <array>
```

Functions

- int main (int argc, char **argv)

13.308.1 Function Documentation

13.308.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.309 2x2-pluq.C File Reference

```
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Functions

- int main (int argc, char **argv)

13.309.1 Function Documentation

13.309.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.310 fflas-101_1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.310.1 Function Documentation

13.310.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.311 fflas-101_3.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

```
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.311.1 Function Documentation

13.311.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.312 fflas_101.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.312.1 Function Documentation

13.312.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.313 fflas_101_lvl1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

Functions

- int main (int argc, char **argv)

13.313.1 Function Documentation

13.313.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.314 ffpack-fgesv.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.314.1 Function Documentation

13.314.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.315 ffpack-solve.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

Functions

- int main (int argc, char **argv)

13.315.1 Function Documentation

13.315.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

PS: the function Solve will modify the matrix A so here we used a duplicate matrix A2 otherwise A*x will not be equal to b for the later verification stage

Index

_F
RNSIntegerMod< RNS >, 548
_M
rns_double, 527
rns_double_extended, 538
_MAX_SIZE_MATRICES
benchmark-checkers.C, 779
_MMi
rns_double, 527
rns_double_extended, 538
_Mi
rns_double, 527
rns_double_extended, 538
_Mi_modp_rns
RNSIntegerMod< RNS >, 548
_NR_TESTS
benchmark-checkers.C, 779
_PLUQ
FFPACK, 361
_RNSdelayed
RNSIntegerMod< RNS >, 548
_TEST_ONE
test-simd.C, 1099
__FFLASFFPACK_ARITHPROG_THRESHOLD
fflas-ffpack-default-thresholds.h, 823
__FFLASFFPACK_CACHE_LINE_SIZE
fflas_sparse.h, 883
__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_CONFIGURATION
cblas.C, 1048
clapack.C, 1048
fblas.C, 1049
lapack.C, 1050
__FFLASFFPACK_DIMKPENALTY
fflas_pfgemm.inl, 868
__FFLASFFPACK_FORCE_SEQ
benchmark-charpoly-mp.C, 777
__FFLASFFPACK_FSYRK_THRESHOLD
fflas-ffpack-default-thresholds.h, 823
__FFLASFFPACK_FSYTRF_THRESHOLD
fflas-ffpack-default-thresholds.h, 823
__FFLASFFPACK_FTRSSYR2K_THRESHOLD
ffpack.h, 921
__FFLASFFPACK_FTRSTR_THRESHOLD
ffpack.h, 921
__FFLASFFPACK_FTRTRI_THRESHOLD
fflas-ffpack-default-thresholds.h, 823
__FFLASFFPACK_GAUSSJORDAN_BASECASE
ffpack_echelonforms.inl, 929
test-echelon.C, 1056
__FFLASFFPACK_HAVE_BLAS
config.h, 819
__FFLASFFPACK_HAVE_CBLAS
cblas.C, 1048
config.h, 819
__FFLASFFPACK_HAVE_CLAPACK
clapack.C, 1048
__FFLASFFPACK_HAVE_CXX11
config.h, 819
__FFLASFFPACK_HAVE_DGETRF
benchmark-dgetrf.C, 781
__FFLASFFPACK_HAVE_DLFCN_H
config.h, 819
__FFLASFFPACK_HAVE_DTRTRI
benchmark-dtrtri.C, 783
__FFLASFFPACK_HAVE_FLOAT_H
config.h, 819
__FFLASFFPACK_HAVE_INT128
config.h, 819
__FFLASFFPACK_HAVE_INTTYPES_H
config.h, 819
__FFLASFFPACK_HAVE_LAPACK
clapack.C, 1048
config.h, 819
lapack.C, 1050
__FFLASFFPACK_HAVE_LIMITS_H
config.h, 819
__FFLASFFPACK_HAVE_LITTLE_ENDIAN
config.h, 819
__FFLASFFPACK_HAVE_PTHREAD_H
config.h, 819
__FFLASFFPACK_HAVE_STDDEF_H
config.h, 819
__FFLASFFPACK_HAVE_STDINT_H
config.h, 820
__FFLASFFPACK_HAVE_STDIO_H
config.h, 820
__FFLASFFPACK_HAVE_STDLIB_H
config.h, 820
__FFLASFFPACK_HAVE_STRINGS_H
config.h, 820
__FFLASFFPACK_HAVE_STRING_H
config.h, 820
__FFLASFFPACK_HAVE_SYS_STAT_H
config.h, 820

__FFLASFFPACK_HAVE_SYS_TIME_H
config.h, 820
__FFLASFFPACK_HAVE_SYS_TYPES_H
config.h, 820
__FFLASFFPACK_HAVE_UNISTD_H
config.h, 820
__FFLASFFPACK_KAAPI_ROUTINES_INL
kaapi_routines.inl, 1029
__FFLASFFPACK_LT_OBJDIR
config.h, 820
__FFLASFFPACK_MINBLOCKCUTS
blockcuts.inl, 1028
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET
benchmark-charpoly.C, 778
benchmark-fadd-lvl2.C, 783
benchmark-fdot.C, 784
benchmark-fgemm-mp.C, 785
benchmark-fgemm-rns.C, 786
benchmark-fgemv-mp.C, 788
benchmark-fgemv.C, 789
benchmark-fgesv.C, 792
benchmark-fsyrk.C, 793
benchmark-fsytrf.C, 794
benchmark-ftrsm-mp.C, 794
benchmark-ftrsm.C, 795
benchmark-ftrsv.C, 796
benchmark-ftrtri.C, 796
benchmark-pluq.C, 799
benchmark-quasisep.C, 800
__FFLASFFPACK_OPENBLAS_NUM_THREADS
config.h, 820
__FFLASFFPACK_PACKAGE
config.h, 820
__FFLASFFPACK_PACKAGE_BUGREPORT
config.h, 820
__FFLASFFPACK_PACKAGE_NAME
config.h, 820
__FFLASFFPACK_PACKAGE_STRING
config.h, 820
__FFLASFFPACK_PACKAGE_TARNAME
config.h, 821
__FFLASFFPACK_PACKAGE_URL
config.h, 821
__FFLASFFPACK_PACKAGE_VERSION
config.h, 821
__FFLASFFPACK_PLUQ_THRESHOLD
fflas-ffpack-default-thresholds.h, 822
test-echelon.C, 1056
__FFLASFFPACK_SEQPARTHRESHOLD
fflas_pfgemm.inl, 868
__FFLASFFPACK_SEQUENTIAL
parallel.h, 1030
test-echelon.C, 1056
test-ftrmm.C, 1076
test-ftrmv.C, 1077
test-ftrsm.C, 1079
test-ftrsv.C, 1082
test-ftrtri.C, 1083
test-lu.C, 1087
test-rankprofiles.C, 1097
__FFLASFFPACK_SIZEOF_CHAR
config.h, 821
__FFLASFFPACK_SIZEOF_INT
config.h, 821
__FFLASFFPACK_SIZEOF_LONG
config.h, 821
__FFLASFFPACK_SIZEOF_LONG_LONG
config.h, 821
__FFLASFFPACK_SIZEOF_SHORT
config.h, 821
__FFLASFFPACK_SIZEOF___INT64_T
config.h, 821
__FFLASFFPACK_STDC_HEADERS
config.h, 821
__FFLASFFPACK_USE_OPENMP
config.h, 821
__FFLASFFPACK_VERSION
config.h, 821
__FFLASFFPACK_WINOTHRESHOLD
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_WINOTHRESHOLD_BAL
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_WINOTHRESHOLD_FLT
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_charpoly_INL
ffpack_charpoly.inl, 924
__FFLASFFPACK_checker_charpoly_INL
checker_charpoly.inl, 804
__FFLASFFPACK_checker_det_INL
checker_det.inl, 804
__FFLASFFPACK_checker_fgemm_INL
checker_fgemm.inl, 805
__FFLASFFPACK_checker_ftrsm_INL
checker_ftrsm.inl, 805
__FFLASFFPACK_checker_invert_INL
checker_invert.inl, 805
__FFLASFFPACK_checker_pluq_INL
checker_pluq.inl, 806
__FFLASFFPACK_fadd_INL
fflas_fadd.inl, 828
__FFLASFFPACK_fassign_INL
fflas_fassign.inl, 829
__FFLASFFPACK_faxpy_INL
fflas_faxpy.inl, 830
__FFLASFFPACK_fdot_INL
fflas_fdot.inl, 831
__FFLASFFPACK_fflas_blockcuts_INL
blockcuts.inl, 1028
__FFLASFFPACK_fflas_bounds_INL
fflas_bounds.inl, 825
__FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
fgemm_winograd.inl, 836
__FFLASFFPACK_fflas_fflas_level1_INL
fflas_level1.inl, 862

__FFLASFFPACK_fflas_fflas_level2_INL
fflas_level2.inl, 865
__FFLASFFPACK_fflas_fflas_level3_INL
fflas_level3.inl, 867
__FFLASFFPACK_fflas_fflas_mmhelper_INL
fflas_helpers.inl, 856
__FFLASFFPACK_fflas_fflas_sparse_INL
fflas_sparse.inl, 885
__FFLASFFPACK_fflas_ffpack_utils_simd128_INL
simd128.inl, 871
__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
simd128_double.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
simd128_float.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
simd128_int16.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
simd128_int32.inl, 873
__FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
simd128_int64.inl, 873
__FFLASFFPACK_fflas_ffpack_utils_simd256_INL
simd256.inl, 874
__FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
simd256_double.inl, 874
__FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
simd256_float.inl, 875
__FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
simd256_int16.inl, 875
__FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
simd256_int32.inl, 876
__FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
simd256_int64.inl, 876
__FFLASFFPACK_fflas_freduce_INL
fflas_freduce.inl, 846
__FFLASFFPACK_fflas_freduce_mp_INL
fflas_freduce_mp.inl, 847
__FFLASFFPACK_fflas_fsyr2k_INL
fflas_fsyr2k.inl, 850
__FFLASFFPACK_fflas_fsyrk_INL
fflas_fsyrk.inl, 852
__FFLASFFPACK_fflas_fsyrk_strassen_INL
fflas_fsyrk_strassen.inl, 853
__FFLASFFPACK_fflas_igemm_igemm_INL
igemm.inl, 858
__FFLASFFPACK_fflas_igemm_igemm_kernels_INL
igemm_kernels.inl, 859
__FFLASFFPACK_fflas_igemm_igemm_tools_INL
igemm_tools.inl, 860
__FFLASFFPACK_fflas_pfgemm_INL
fflas_pfgemm.inl, 868
__FFLASFFPACK_fflas_pftrsm_INL
fflas_pftrsm.inl, 869
__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
csr_hyb_pspmm.inl, 894
__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
csr_hyb_pspmv.inl, 894
__FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL
csr_hyb_spmm.inl, 895
__FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
csr_hyb_spmv.inl, 895
__FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
csr_hyb_utils.inl, 896
__FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
csr_pspmm.inl, 889
__FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
csr_pspmv.inl, 890
__FFLASFFPACK_fflas_sparse_CSR_spmm_INL
csr_spmm.inl, 891
__FFLASFFPACK_fflas_sparse_CSR_spmv_INL
csr_spmv.inl, 892
__FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
ell_pspmm.inl, 897
__FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
ell_pspmv.inl, 898
__FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
ell_simd_pspmv.inl, 901
__FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
ell_simd_spmv.inl, 902
__FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
ell_simd_utils.inl, 903
__FFLASFFPACK_fflas_sparse_ELL_spmm_INL
ell_spmm.inl, 899
__FFLASFFPACK_fflas_sparse_ELL_spmv_INL
ell_spmv.inl, 899
__FFLASFFPACK_fflas_sparse_ELL_utils_INL
ell_utils.inl, 900
__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
hyb_zo_pspmm.inl, 903
__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
hyb_zo_pspmv.inl, 904
__FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL
hyb_zo_spmm.inl, 904
__FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
hyb_zo_spmv.inl, 905
__FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
hyb_zo_utils.inl, 905
__FFLASFFPACK_fflas_sparse_coo_spmm_INL
coo_spmm.inl, 886
__FFLASFFPACK_fflas_sparse_coo_spmv_INL
coo_spmv.inl, 887
__FFLASFFPACK_fflas_sparse_coo_utils_INL
coo_utils.inl, 888
__FFLASFFPACK_fflas_sparse_sell_pspmv_INL
sell_pspmv.inl, 907
__FFLASFFPACK_fflas_sparse_sell_spmv_INL
sell_spmv.inl, 908
__FFLASFFPACK_fflas_sparse_sell_utils_INL
sell_utils.inl, 909
__FFLASFFPACK_ffpack_INL
ffpack.inl, 922
__FFLASFFPACK_ffpack_bruhatgen_inl
ffpack_bruhatgen.inl, 924
__FFLASFFPACK_ffpack_charpoly_danilveski_INL
ffpack_charpoly_danilevski.inl, 925
__FFLASFFPACK_ffpack_charpoly_kgfast_INL
ffpack_charpoly_kgfast.inl, 925

__FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
ffpack_charpoly_kgfastgeneralized.inl, 926
__FFLASFFPACK_ffpack_charpoly_kglu_INL
ffpack_charpoly_kglu.inl, 926
__FFLASFFPACK_ffpack_echelon_forms_INL
ffpack_echelonforms.inl, 929
__FFLASFFPACK_ffpack_fgesv_INL
ffpack_fgesv.inl, 929
__FFLASFFPACK_ffpack_fgetrs_INL
ffpack_fgetrs.inl, 930
__FFLASFFPACK_ffpack_fsytrf_INL
ffpack_fsytrf.inl, 932
__FFLASFFPACK_ffpack_ftrssyr2k_INL
ffpack_ftrssyr2k.inl, 932
__FFLASFFPACK_ffpack_ftrstr_INL
ffpack_ftrstr.inl, 933
__FFLASFFPACK_ffpack_ftrtr_INL
ffpack_ftrtr.inl, 933
__FFLASFFPACK_ffpack_invert_INL
ffpack_invert.inl, 934
__FFLASFFPACK_ffpack_krylovelim_INL
ffpack_krylovelim.inl, 934
__FFLASFFPACK_ffpack_ludivine_INL
ffpack_ludivine.inl, 935
__FFLASFFPACK_ffpack_minpoly_INL
ffpack_minpoly.inl, 936
__FFLASFFPACK_ffpack_permutation_INL
ffpack_permutation.inl, 938
__FFLASFFPACK_ffpack_pluq_INL
ffpack_pluq.inl, 939
__FFLASFFPACK_ffpack_ppluq_INL
ffpack_ppluq.inl, 940
__FFLASFFPACK_ffpack_rank_profiles_INL
ffpack_rankprofiles.inl, 942
__FFLASFFPACK_fgemm_INL
fflas_fgemm.inl, 833
__FFLASFFPACK_fgemm_bini_INL
schedule_bini.inl, 837
__FFLASFFPACK_fgemm_winograd_INL
schedule_winograd.inl, 838
__FFLASFFPACK_fgemm_winograd_acc_INL
schedule_winograd_acc.inl, 838
__FFLASFFPACK_fgemm_winograd_acc_ip_INL
schedule_winograd_acc_ip.inl, 839
__FFLASFFPACK_fgemm_winograd_ip_INL
schedule_winograd_ip.inl, 840
__FFLASFFPACK_fgemv_INL
fflas_fgemv.inl, 841
__FFLASFFPACK_fgemv_mp_INL
fflas_fgemv_mp.inl, 842
__FFLASFFPACK_fger_INL
fflas_fger.inl, 843
__FFLASFFPACK_field_rns_INL
rns.inl, 948
__FFLASFFPACK_field_rns_double_INL
rns-double.inl, 946
__FFLASFFPACK_field_rns_double_recint_INL
rns-double-recint.inl, 945
__FFLASFFPACK_freivalds_INL
fflas_freivalds.inl, 847
__FFLASFFPACK_fscal_INL
fflas_fscal.inl, 849
__FFLASFFPACK_fscal_mp_INL
fflas_fscal_mp.inl, 850
__FFLASFFPACK_ftrmm_INL
fflas_ftrmm.inl, 854
__FFLASFFPACK_ftrsm_INL
fflas_ftrsm.inl, 854
__FFLASFFPACK_ftrsv_INL
fflas_ftrsv.inl, 855
__FFLASFFPACK_simd512_INL
simd512.inl, 877
__FFLASFFPACK_simd512_double_INL
simd512_double.inl, 877
__FFLASFFPACK_simd512_float_INL
simd512_float.inl, 877
__FFLASFFPACK_simd512_int32_INL
simd512_int32.inl, 878
__FFLAS_L1_INST_C
fflas_L1_inst.C, 960
__FFLAS_L2_INST_C
fflas_L2_inst.C, 964
__FFLAS_L3_INST_C
fflas_L3_inst.C, 967
__FFLAS__TRSM_READONLY
fflas_L3_inst_implem.inl, 970
fflas_level3.inl, 867
ffpack_ppluq.inl, 940
__FFPACK_FSYTRF_BC_CROUT
benchmark-fsytrf.C, 794
__FFPACK_INST_C
ffpack_inst.C, 1022
__FFPACK_charpoly_mp_INL
ffpack_charpoly_mp.inl, 927
__FFPACK_det_mp_INL
ffpack_det_mp.inl, 927
__FFPACK_fgemm_classical_INL
fgemm_classical_mp.inl, 835
__FFPACK_fger_mp_INL
fflas_fger_mp.inl, 844
__FFPACK_ftrsm_mp_INL
fflas_ftrsm_mp.inl, 855
__FFPACK_ludivine_mp_INL
ffpack_ludivine_mp.inl, 935
__FFPACK_pluq_mp_INL
ffpack_pluq_mp.inl, 940
__LUDIVINE_CUTOFF
test-lu.C, 1087
__has_builtin
bit_manipulation.h, 1040
_alloc
rns_double_elt, 529
rns_double_elt_cstptr, 532
rns_double_elt_ptr, 534
_basis
rns_double, 527

rns_double_extended, 537
_basisMax
rns_double, 527
rns_double_extended, 537
_coo
SpMat< Field, flag >, 752
_coo16
CooMat< Field >, 434
_coo16_zo
CooMat< Field >, 434
_coo32
CooMat< Field >, 434
_coo32_zo
CooMat< Field >, 434
_coo64
CooMat< Field >, 434
_coo64_zo
CooMat< Field >, 434
_crt_in
rns_double, 527
rns_double_extended, 538
_crt_out
rns_double, 527
rns_double_extended, 538
_csr
SpMat< Field, flag >, 752
_csr16
CsrMat< Field >, 436
_csr16_zo
CsrMat< Field >, 436
_csr32
CsrMat< Field >, 436
_csr32_zo
CsrMat< Field >, 436
_csr64
CsrMat< Field >, 436
_csr64_zo
CsrMat< Field >, 436
_ell
SpMat< Field, flag >, 752
_ell16
EllMat< Field >, 444
_ell16_zo
EllMat< Field >, 444
_ell32
EllMat< Field >, 444
_ell32_zo
EllMat< Field >, 444
_ell64
EllMat< Field >, 444
_ell64_zo
EllMat< Field >, 444
_errorStream
Failure, 446
_field_rns
rns_double, 527
rns_double_extended, 538
_iM_modp_rns
RNSIntegerMod< RNS >, 548
_ibeg
ForStrategy2D< blocksize_t, Cut, Param >, 468
_iend
ForStrategy2D< blocksize_t, Cut, Param >, 468
_invbasis
rns_double, 527
rns_double_extended, 537
_jbeg
ForStrategy2D< blocksize_t, Cut, Param >, 468
_jend
ForStrategy2D< blocksize_t, Cut, Param >, 468
_ldm
rns_double, 528
rns_double_extended, 538
_mi_sum
rns_double, 528
_mm
Test< Elt >, 759
_negbasis
rns_double, 527
rns_double_extended, 537
_nn
Test< Elt >, 759
_p
RNSIntegerMod< RNS >, 548
_pbits
rns_double, 528
rns_double_extended, 538
_ptr
rns_double_elt, 529
rns_double_elt_cstptr, 532
rns_double_elt_ptr, 534
_rns
RNSInteger< RNS >, 542
RNSIntegerMod< RNS >, 548
_simd512_int64_INL
simd512_int64.inl, 878
_size
rns_double, 527
rns_double_extended, 538
_stride
rns_double_elt, 529
rns_double_elt_cstptr, 532
rns_double_elt_ptr, 534
_zero
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 557
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
~CheckerImplem_Det
CheckerImplem_Det< Field >, 414
~CheckerImplem_PLUQ
CheckerImplem_PLUQ< Field >, 418
~CheckerImplem_charpoly
CheckerImplem_charpoly< Field, Polynomial >, 411

CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 413
~CheckerImplem_fgemm
CheckerImplem_fgemm< Field >, 415
~CheckerImplem_ftrsm
CheckerImplem_ftrsm< Field >, 416
~CheckerImplem_invert
CheckerImplem_invert< Field >, 417
~rns_double_elt
rns_double_elt, 528
101-fgemm.C, 1102
main, 1103
2x2-fgemm.C, 1103
main, 1103
2x2-ftrsv.C, 1103
main, 1104
2x2-pluq.C, 1104
main, 1104

add
FFLAS::vectorised, 280
FieldSimd< _Field >, 449
RNSIntegerMod< RNS >, 546
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 652
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
add_r
FieldSimd< _Field >, 449
addin
FieldSimd< _Field >, 449
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 653
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
addin_r
FieldSimd< _Field >, 449
addp
FFLAS::vectorised, 279
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, 397
value, 397
AlgoChooser< ModeT, ParSeq >, 397
value, 397
align-allocator.h, 1037
alignable
FFLAS, 188
alignable< Givaro::Integer * >
FFLAS, 188
aligned_allocator
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 573
Simd128_impl< true, true, false, 8 >, 583
Simd128_impl< true, true, true, 2 >, 593
Simd128_impl< true, true, true, 4 >, 602
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 630
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 679
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 723
aligned_vector
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 573
Simd128_impl< true, true, false, 8 >, 583
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 630
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 679, 680
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 723
alignment
FieldSimd< _Field >, 452
NoSimd< T >, 517
Simd128_impl< true, true, false, 2 >, 571
Simd128_impl< true, true, false, 4 >, 581

Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 619
Simd256_impl< true, false, true, 8 >, 628
Simd256_impl< true, true, false, 2 >, 638
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 692
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, false, true, 8 >, 709
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 730
ALL< false, v... >, 398
value, 398
ALL< true, v... >, 398
value, 398
ALL< v >, 397
ALL<>, 398
value, 398
Amax
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 505
Amin
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 505
applyP
FFPACK, 304, 305, 363
applyP_block
FFPACK, 355
applyP_modular_double
ffpack.C, 987
ffpack_c.h, 1009
ArbitraryPrecIntTag, 398
Architecture of the library., 2
areEqual
RNSIntegerMod< RNS >, 547
AreEqual< X, X >, 399
value, 399
AreEqual< X, Y >, 399
value, 399
args-parser.h, 1038
ArgumentType, 1039
END_OF_ARGUMENTS, 1038
findArgument, 1039
getListArgs, 1039
printHelpMessage, 1039
TYPE_BOOL, 1038
TYPE_DOUBLE, 1039
TYPE_INT, 1039
TYPE_INTEGER, 1039
type_integer, 1038
TYPE_INTLIST, 1039
TYPE_LONGLONG, 1039
TYPE_NONE, 1039
TYPE_STR, 1039
TYPE_UINT64, 1039
Argument, 399
c, 399
data, 400
example, 399
helpString, 400
type, 400
ArgumentType
args-parser.h, 1039
ArithProg
FFPACK::Protected, 393
arithprog.C, 771
main, 771
TTimer, 771
array< T >, 400
elements, 400
array< T >::const_iterator, 423
array< T >::const_reverse_iterator, 423
array< T >::iterator, 490
array< T >::reverse_iterator, 522
assign
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 546
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, 401
field, 401
type, 401
associatedDelayedField< const Givaro::Modular< T, X > >, 401
field, 402
type, 402
associatedDelayedField< const Givaro::ModularBalanced< T > >, 402
field, 402
type, 402
associatedDelayedField< const Givaro::ZRing< T > >, 403
field, 403
type, 403
associatedDelayedField< Field >, 400
field, 401
type, 401
assume_aligned
fflas_sparse.h, 883
AtlasConj
config-blas.h, 810
Aunfit
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 504
aut
HelperFlag, 477
Auto, 403
averageCol
StatsMatrix, 753
averageColDifference
StatsMatrix, 754
averageRow
StatsMatrix, 753
averageRowDifference

StatsMatrix, 754
axpy
FieldSimd< _Field >, 451
axpy_r
FieldSimd< _Field >, 452
axpyin
FieldSimd< _Field >, 451
RNSIntegerMod< RNS >, 546
axpyin_r
FieldSimd< _Field >, 452
axpyp
FFLAS::vectorised, 280
FFLAS::vectorised::unswitch, 284

balanced
FieldTraits< FFPACK::RNSInteger< T > >, 454
FieldTraits< FFPACK::RNSIntegerMod< T > >, 454
FieldTraits< Field >, 453
FieldTraits< Givaro::Modular< Element > >, 455
FieldTraits< Givaro::ModularBalanced< Element > >, 456
FieldTraits< Givaro::ZRing< double > >, 456
FieldTraits< Givaro::ZRing< float > >, 457
FieldTraits< Givaro::ZRing< Givaro::Integer > >, 458
FieldTraits< Givaro::ZRing< int16_t > >, 458
FieldTraits< Givaro::ZRing< int32_t > >, 459
FieldTraits< Givaro::ZRing< int64_t > >, 460
FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >, 460
FieldTraits< Givaro::ZRing< uint16_t > >, 461
FieldTraits< Givaro::ZRing< uint32_t > >, 462
FieldTraits< Givaro::ZRing< uint64_t > >, 462
winograd.C, 777
BARRIER
parallel.h, 1030
BASECASE_K
test-lu.C, 1087
BaseTimer
FFLAS, 74
BasisElement
rns_double, 524
rns_double_extended, 535
RNSInteger< RNS >, 539
RNSIntegerMod< RNS >, 544
begin
ForStrategy1D< blocksize_t, Cut, Param >, 464
Info, 482, 483
BEGIN_PARALLEL_MAIN
parallel.h, 1031
Bench
Bench< Elt >, 405
Bench< Elt >, 403
Bench, 405
cardinality, 405
doBenchs, 405
Elt_ptr, 404
enable_if_no_simd_t, 405
enable_if_simd128_t, 405
enable_if_simd256_t, 405
enable_if_simd512_t, 405
enable_if_t, 404
F, 406
Field, 404
inplace, 406
is_same_element, 404
iters, 406
m, 406
n, 406
Residu, 404
run, 405
benchmark-charpoly-mp.C, 777
__FFLASFFPACK_FORCE_SEQ, 777
main, 777
benchmark-charpoly.C, 778
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 778
main, 778
run_with_field, 778
benchmark-checkers.C, 778
_MAX_SIZE_MATRICES, 779
_NR_TESTS, 779
CUBE, 779
ENABLE_ALL_CHECKINGS, 779
main, 779
benchmark-dgemm.C, 779
CBLAS_GEMM, 780
Floats, 780
main, 780
TTimer, 780
benchmark-dgetrf.C, 780
__FFLASFFPACK_HAVE_DGETRF, 781
main, 781
benchmark-dgetri.C, 781
main, 781
benchmark-dsytrf.C, 781
EFFGFF, 782
main, 782
benchmark-dtrsm.C, 782
main, 782
benchmark-dtrtri.C, 782
__FFLASFFPACK_HAVE_DTRTRI, 783
main, 783
benchmark-fadd-lvl2.C, 783
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 783
main, 783
benchmark-fdot.C, 783
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 784
main, 784
run_with_field, 784
benchmark-fgemm-mp.C, 784
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 785
main, 785

MG_DEFAULT, 785
STD_RECINT_SIZE, 785
tmain, 785
benchmark-fgemm-rns.C, 785
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 786
ConstElement_ptr, 786
Element_ptr, 786
Field, 786
GRAIN, 786
main, 787
PSeq, 787
RNS, 786
THREADS, 786
THREED, 786
THREEDA, 786
THREEDIP, 786
TWOD, 786
TWODA, 786
benchmark-fgemm.C, 787
CLASSIC_HYBRID, 787
main, 787
benchmark-fgemv-mp.C, 787
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 788
main, 788
MG_DEFAULT, 788
STD_RECINT_SIZE, 788
tmain, 788
write_matrix, 788
benchmark-fgemv.C, 789
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 789
benchmark_disp, 791
benchmark_in_Field, 791
benchmark_with_field, 791
benchmark_with_timer, 790
check_result, 790
fill_value, 790
genData, 790
main, 792
benchmark-fgesv.C, 792
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 792
main, 792
benchmark-fsyr2k.C, 792
main, 793
benchmark-fsyrk.C, 793
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 793
main, 793
benchmark-fsytrf.C, 793
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 794
__FFPACK_FSYTRF_BC_CROUT, 794
CUBE, 794
main, 794
benchmark-ftrsm-mp.C, 794
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 794
main, 795
benchmark-ftrsm.C, 795
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 795
main, 795
benchmark-ftrsv.C, 795
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 796
main, 796
benchmark-ftrtri.C, 796
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 796
CUBE, 796
main, 796
benchmark-inverse.C, 796
CUBE, 797
main, 797
benchmark-lqup-mp.C, 797
main, 797
benchmark-lqup.C, 798
CUBE, 798
main, 798
benchmark-pluq.C, 798
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 799
CUBE, 799
Field, 799
main, 799
Rec_Initialize, 799
verification_PLUQ, 799
benchmark-quasisep.C, 799
__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET, 800
main, 800
run_with_field, 800
benchmark-storage-transpose.C, 800
main, 801
benchmark-wino.C, 801
CUBE, 801
launch_wino, 801
main, 802
benchmark_disp
benchmark-fgemv.C, 791
benchmark_in_Field
benchmark-fgemv.C, 791
benchmark_with_field
benchmark-fgemv.C, 791
benchmark_with_timer
benchmark-fgemv.C, 790
Bibliography, 7
Bini, 406
FFLAS::BLAS3, 192
bit_manipulation.h, 1039
__has_builtin, 1040
clz, 1040
ctz, 1040

bitsize
FFLAS, 140
bitsize< Givaro::ZRing< Givaro::Integer > >
FFLAS, 140
blas_enum
config-blas.h, 809
blend
ScalFunctions< Element >, 555
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 652
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 682, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
blendv
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
Simd256_impl< true, false, true, 8 >, 625
Simd512_impl< true, false, true, 8 >, 707
Block, 406
BlockCuts
FFLAS, 179, 181
BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >
FFLAS, 180
BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >
FFLAS, 179
BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >
FFLAS, 181
BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >
FFLAS, 180
BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >
FFLAS, 180
BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >
FFLAS, 180
BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >
FFLAS, 179
BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >
FFLAS, 179
BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >
FFLAS, 180
BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >
FFLAS, 179
blockcuts.inl, 1027
__FFLASFFPACK_MINBLOCKCUTS, 1028
__FFLASFFPACK_fflas_blockcuts_INL, 1028
blockindex
ForStrategy1D< blocksize_t, Cut, Param >, 465
ForStrategy2D< blocksize_t, Cut, Param >, 468
BlockingFactor
FFLAS::details, 206
BLOCKS
ForStrategy2D< blocksize_t, Cut, Param >, 469
BlockTransposeSIMD< Field, Simd, >, 406
info, 407
size, 407
transpose, 407, 408
Bmax
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
Bmin
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
Bruhat2EchelonPermutation
FFPACK, 343
Bug List, 3
build
ForStrategy1D< blocksize_t, Cut, Param >, 464
buildMatrix
FFPACK, 349
Bunfit
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 504

c
Argument, 399
callLUdivine_small< double >, 409
operator(), 409
callLUdivine_small< Element >, 408
operator(), 408
callLUdivine_small< float >, 410
operator(), 410
cardinality
Bench< Elt >, 405
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 545
Test< Elt >, 759
cast.h, 1040
category
FieldTraits< FFPACK::RNSInteger< T > >, 454
FieldTraits< FFPACK::RNSIntegerMod< T > >, 454
FieldTraits< Field >, 453
FieldTraits< Givaro::Modular< Element > >, 455
FieldTraits< Givaro::ModularBalanced< Element > >, 456

FieldTraits< Givaro::ZRing< double > >, 456
FieldTraits< Givaro::ZRing< float > >, 457
FieldTraits< Givaro::ZRing< Givaro::Integer > >, 458
FieldTraits< Givaro::ZRing< int16_t > >, 458
FieldTraits< Givaro::ZRing< int32_t > >, 459
FieldTraits< Givaro::ZRing< int64_t > >, 459
FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >, 460
FieldTraits< Givaro::ZRing< uint16_t > >, 461
FieldTraits< Givaro::ZRing< uint32_t > >, 461
FieldTraits< Givaro::ZRing< uint64_t > >, 462
cblas.C, 1048
__FFLASFFPACK_CONFIGURATION, 1048
__FFLASFFPACK_HAVE_CBLAS, 1048
main, 1048
CBLAS_DIAG
config-blas.h, 810
cblas_dsyrk
config-blas.h, 814
CBLAS_ENUM_DEFINED_H
config-blas.h, 809
CBLAS_EXTERNALS
config-blas.h, 809
CBLAS_GEMM
benchmark-dgemm.C, 780
cblas_imptrsm
FFLAS, 127
CBLAS_INT
config-blas.h, 809
CBLAS_ORDER
config-blas.h, 809
CBLAS_SIDE
config-blas.h, 810
CBLAS_TRANSPOSE
config-blas.h, 809
CBLAS_UPLO
config-blas.h, 810
CblasColMajor
config-blas.h, 809
CblasConjTrans
config-blas.h, 810
CblasLeft
config-blas.h, 810
CblasLower
config-blas.h, 810
CblasNonUnit
config-blas.h, 810
CblasNoTrans
config-blas.h, 810
CblasRight
config-blas.h, 810
CblasRowMajor
config-blas.h, 809
CblasTrans
config-blas.h, 810
CblasUnit
config-blas.h, 810
CblasUpper
config-blas.h, 810
ceil
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
Simd256_impl< true, false, true, 8 >, 627
Simd512_impl< true, false, true, 8 >, 709
changeBS
ForStrategy1D< blocksize_t, Cut, Param >, 465
changeCBS
ForStrategy2D< blocksize_t, Cut, Param >, 469
changeRBS
ForStrategy2D< blocksize_t, Cut, Param >, 469
characteristic
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 545
CharPoly
FFPACK, 321, 322, 349, 367, 368
charpoly.C, 771, 772
CUBE, 772
GFOPS, 772
main, 772
TTimer, 772
CharpolyFailed, 410
check
Checker_Empty< Field >, 411
CheckerImplem_charpoly< Field, Polynomial >, 412
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 413
CheckerImplem_Det< Field >, 414
CheckerImplem_fgemm< Field >, 415
CheckerImplem_ftrsm< Field >, 416
CheckerImplem_invert< Field >, 417
CheckerImplem_PLUQ< Field >, 418
check1
regression-check.C, 1051
check2
regression-check.C, 1051
check3
regression-check.C, 1051
check4
regression-check.C, 1051
check_computeS1S2
test-fsyrk.C, 1074
CHECK_DEPENDENCIES
parallel.h, 1030
check_eq
test-simd.C, 1100
check_fdot
test-fdot.C, 1059
check_fger
test-fger.C, 1066
check_fsyr2k
test-fsyr2k.C, 1072
check_fsyrk
test-fsyrk.C, 1073

check_fsyrk_bkdiag
test-fsyrk.C, 1073
check_fsyrk_diag
test-fsyrk.C, 1073
check_ftrmm
test-ftrmm.C, 1076
check_ftrmv
test-ftrmv.C, 1077
check_ftrsm
test-ftrsm.C, 1079
check_ftrssyr2k
test-ftrssyr2k.C, 1080
check_ftrstr
test-ftrstr.C, 1081
check_ftrsv
test-ftrsv.C, 1082
check_ftrtri
test-ftrtri.C, 1084
check_minpoly
test-minpoly.C, 1091
check_MM
test-fgemm.C, 1062
check_MV
test-fgemv.C, 1064
check_result
benchmark-fgemv.C, 790
check_solve
test-solve.C, 1101
checkA
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 504
checkB
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 504
CHECKER, 41
Checker_charpoly
FFPACK, 303
checker_charpoly.inl, 803
__FFLASFFPACK_checker_charpoly_INL, 804
Checker_Det
FFPACK, 302
checker_det.inl, 804
__FFLASFFPACK_checker_det_INL, 804
Checker_Empty
Checker_Empty< Field >, 411
Checker_Empty< Field >, 410
check, 411
Checker_Empty, 411
checker_empty.h, 804
Checker_fgemm
FFLAS, 72
checker_fgemm.inl, 804
__FFLASFFPACK_checker_fgemm_INL, 805
Checker_ftrsm
FFLAS, 72
checker_ftrsm.inl, 805
__FFLASFFPACK_checker_ftrsm_INL, 805
Checker_invert
FFPACK, 303
checker_invert.inl, 805
__FFLASFFPACK_checker_invert_INL, 805
Checker_PLUQ
FFPACK, 302
checker_pluq.inl, 805
__FFLASFFPACK_checker_pluq_INL, 806
CheckerImplem_charpoly
CheckerImplem_charpoly< Field, Polynomial >, 411
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 412, 413
CheckerImplem_charpoly< Field, Polynomial >, 411
~CheckerImplem_charpoly, 411
check, 412
CheckerImplem_charpoly, 411
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 412
~CheckerImplem_charpoly, 413
check, 413
CheckerImplem_charpoly, 412, 413
Ring, 412
CheckerImplem_Det
CheckerImplem_Det< Field >, 414
CheckerImplem_Det< Field >, 413
~CheckerImplem_Det, 414
check, 414
CheckerImplem_Det, 414
CheckerImplem_fgemm
CheckerImplem_fgemm< Field >, 415
CheckerImplem_fgemm< Field >, 414
~CheckerImplem_fgemm, 415
check, 415
CheckerImplem_fgemm, 415
CheckerImplem_ftrsm
CheckerImplem_ftrsm< Field >, 416
CheckerImplem_ftrsm< Field >, 416
~CheckerImplem_ftrsm, 416
check, 416
CheckerImplem_ftrsm, 416
CheckerImplem_invert
CheckerImplem_invert< Field >, 417
CheckerImplem_invert< Field >, 417
~CheckerImplem_invert, 417
check, 417
CheckerImplem_invert, 417
CheckerImplem_PLUQ
CheckerImplem_PLUQ< Field >, 418
CheckerImplem_PLUQ< Field >, 417
~CheckerImplem_PLUQ, 418
check, 418
CheckerImplem_PLUQ, 418
checkers.doxy, 806
checkers_fflas.h, 806
checkers_fflas.inl, 806
FFLASFFPACK_checkers_fflas_inl_H, 807
checkers_ffpack.h, 807
checkers_ffpack.inl, 807

FFLASFFPACK_checkers_ffpack_inl_H, 808
checkingMessage
test-nullspace.C, 1092
checkMonotonicApplyP
test-permutations.C, 1094
checkOut
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 504
checkRPM
test-rpm.C, 1097
checkSymmetricRPM
test-rpm.C, 1097
checkZeroDimCharpoly
regression-check.C, 1051
checkZeroDimMinPoly
regression-check.C, 1051
chooseField
FFPACK, 388
chooseField< Givaro::ZRing< double > >
FFPACK, 389
chooseField< Givaro::ZRing< float > >
FFPACK, 388
chooseField< Givaro::ZRing< int32_t > >
FFPACK, 388
chooseField< Givaro::ZRing< int64_t > >
FFPACK, 388
chunk
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
chunkSize
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
clapack.C, 1048
__FFLASFFPACK_CONFIGURATION, 1048
__FFLASFFPACK_HAVE_CLAPACK, 1048
__FFLASFFPACK_HAVE_LAPACK, 1048
main, 1049
Classic, 418
CLASSIC_HYBRID
benchmark-fgemm.C, 787
clz
bit_manipulation.h, 1040
Cmax
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 505
Cmin
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 505
cmp
test-simd.C, 1100
cmp_false
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 557
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 560
cmp_true
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 557
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
col
Coo< Field >, 432
Coo< ValT, IdxT >, 430, 433
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 744
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
colblockindex
ForStrategy2D< blocksize_t, Cut, Param >, 468
colBlockSize
ForStrategy2D< blocksize_t, Cut, Param >, 469
coldim
StatsMatrix, 753
colnumblocks
ForStrategy2D< blocksize_t, Cut, Param >, 467
ColRankProfileSubmatrix
FFPACK, 334, 372
ColRankProfileSubmatrix_modular_double
ffpack.C, 999
ffpack_c.h, 1019
ColRankProfileSubmatrixIndices
FFPACK, 333, 372
ColRankProfileSubmatrixIndices_modular_double
ffpack.C, 999
ffpack_c.h, 1019
Column, 419
ColumnEchelonForm
FFPACK, 315, 316, 366
ColumnEchelonForm_modular_double
ffpack.C, 990
ffpack_c.h, 1012
ColumnEchelonForm_modular_float
ffpack.C, 990
ffpack_c.h, 1012
ColumnEchelonForm_modular_int32_t
ffpack.C, 991
ffpack_c.h, 1012
ColumnRankProfile
FFPACK, 330, 331, 371
ColumnRankProfile_modular_double
ffpack.C, 998

ffpack_c.h, 1018
COMMA
parallel.h, 1033
CompactElement< double >, 419
type, 419
CompactElement< Element >, 419
type, 419
CompactElement< float >, 419
type, 419
CompactElement< int16_t >, 420
type, 420
CompactElement< int32_t >, 420
type, 420
CompactElement< int64_t >, 420
type, 420
compatible_data_type< Field >, 421
value, 421
compatible_data_type< Givaro::ZRing< double > >, 421
value, 421
compatible_data_type< Givaro::ZRing< float > >, 421
value, 421
compliant
NoSimd< T >, 517
Simd128_impl< true, true, false, 2 >, 566
Simd128_impl< true, true, false, 4 >, 576
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 648
Simd256_impl< true, true, false, 8 >, 662
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 715
Simd512_impl< true, true, true, 8 >, 723
Compose
Compose< H1, H2 >, 422
Compose< H1, H2 >, 422
Compose, 422
first_component, 422
operator<<, 423
second_component, 422
composePermutationsLLL
FFPACK, 358
ffpack.C, 986
ffpack_c.h, 1008
composePermutationsLLM
FFPACK, 358
ffpack.C, 986
ffpack_c.h, 1008
composePermutationsMLM
FFPACK, 359
ffpack.C, 986
ffpack_c.h, 1008
CompressRows
FFPACK::Protected, 394
CompressRowsQA
FFPACK::Protected, 395
CompressRowsQK
FFPACK::Protected, 395
CompressToBlockBiDiagonal
FFPACK, 342
computeDeviation
FFLAS, 153
computeFactorClassic
FFLAS::Protected, 211
ComputeRPermutation
FFPACK, 344, 347
computeS1S2
FFLAS, 123
config-blas.h, 808
AtlasConj, 810
blas_enum, 809
CBLAS_DIAG, 810
cblas_dsyrk, 814
CBLAS_ENUM_DEFINED_H, 809
CBLAS_EXTERNALS, 809
CBLAS_INT, 809
CBLAS_ORDER, 809
CBLAS_SIDE, 810
CBLAS_TRANSPOSE, 809
CBLAS_UPLO, 810
CblasColMajor, 809
CblasConjTrans, 810
CblasLeft, 810
CblasLower, 810
CblasNonUnit, 810
CblasNoTrans, 810
CblasRight, 810
CblasRowMajor, 809
CblasTrans, 810
CblasUnit, 810
CblasUpper, 810
dasum_, 811
daxpy_, 810
dcopy_, 812
ddot_, 810
dgemm_, 814
dgemv_, 811
dger_, 812
dnrm2_, 811
dscal_, 812
dtrmm_, 813
dtrsm_, 813
idamax_, 811
saxpy_, 810
scopy_, 812
sdot_, 811
sgemm_, 814
sgemv_, 811
sger_, 812

sscal_, 813
strmm_, 813
strsm_, 813
config.h, 815, 818
__FFLASFFPACK_HAVE_BLAS, 819
__FFLASFFPACK_HAVE_CBLAS, 819
__FFLASFFPACK_HAVE_CXX11, 819
__FFLASFFPACK_HAVE_DLFCN_H, 819
__FFLASFFPACK_HAVE_FLOAT_H, 819
__FFLASFFPACK_HAVE_INT128, 819
__FFLASFFPACK_HAVE_INTTYPES_H, 819
__FFLASFFPACK_HAVE_LAPACK, 819
__FFLASFFPACK_HAVE_LIMITS_H, 819
__FFLASFFPACK_HAVE_LITTLE_ENDIAN, 819
__FFLASFFPACK_HAVE_PTHREAD_H, 819
__FFLASFFPACK_HAVE_STDDEF_H, 819
__FFLASFFPACK_HAVE_STDINT_H, 820
__FFLASFFPACK_HAVE_STDIO_H, 820
__FFLASFFPACK_HAVE_STDLIB_H, 820
__FFLASFFPACK_HAVE_STRINGS_H, 820
__FFLASFFPACK_HAVE_STRING_H, 820
__FFLASFFPACK_HAVE_SYS_STAT_H, 820
__FFLASFFPACK_HAVE_SYS_TIME_H, 820
__FFLASFFPACK_HAVE_SYS_TYPES_H, 820
__FFLASFFPACK_HAVE_UNISTD_H, 820
__FFLASFFPACK_LT_OBJDIR, 820
__FFLASFFPACK_OPENBLAS_NUM_THREADS, 820
__FFLASFFPACK_PACKAGE, 820
__FFLASFFPACK_PACKAGE_BUGREPORT, 820
__FFLASFFPACK_PACKAGE_NAME, 820
__FFLASFFPACK_PACKAGE_STRING, 820
__FFLASFFPACK_PACKAGE_TARNAME, 821
__FFLASFFPACK_PACKAGE_URL, 821
__FFLASFFPACK_PACKAGE_VERSION, 821
__FFLASFFPACK_SIZEOF_CHAR, 821
__FFLASFFPACK_SIZEOF_INT, 821
__FFLASFFPACK_SIZEOF_LONG, 821
__FFLASFFPACK_SIZEOF_LONG_LONG, 821
__FFLASFFPACK_SIZEOF_SHORT, 821
__FFLASFFPACK_SIZEOF___INT64_T, 821
__FFLASFFPACK_STDC_HEADERS, 821
__FFLASFFPACK_USE_OPENMP, 821
__FFLASFFPACK_VERSION, 821
HAVE_BLAS, 815
HAVE_CBLAS, 815
HAVE_CXX11, 815
HAVE_DLFCN_H, 816
HAVE_FLOAT_H, 816
HAVE_INT128, 816
HAVE_INTTYPES_H, 816
HAVE_LAPACK, 816
HAVE_LIMITS_H, 816
HAVE_LITTLE_ENDIAN, 816
HAVE_PTHREAD_H, 816
HAVE_STDDEF_H, 816
HAVE_STDINT_H, 816
HAVE_STDIO_H, 816
HAVE_STDLIB_H, 816
HAVE_STRING_H, 816
HAVE_STRINGS_H, 816
HAVE_SYS_STAT_H, 816
HAVE_SYS_TIME_H, 817
HAVE_SYS_TYPES_H, 817
HAVE_UNISTD_H, 817
LT_OBJDIR, 817
OPENBLAS_NUM_THREADS, 817
PACKAGE, 817
PACKAGE_BUGREPORT, 817
PACKAGE_NAME, 817
PACKAGE_STRING, 817
PACKAGE_TARNAME, 817
PACKAGE_URL, 817
PACKAGE_VERSION, 817
SIZEOF___INT64_T, 818
SIZEOF_CHAR, 817
SIZEOF_INT, 817
SIZEOF_LONG, 817
SIZEOF_LONG_LONG, 818
SIZEOF_SHORT, 818
STDC_HEADERS, 818
USE_OPENMP, 818
VERSION, 818
Configuring and Installing FFLAS-FFPACK, 2
CONST
fflas_simd.h, 870
ConstElement_ptr
benchmark-fgemm-rns.C, 786
rns_double, 524
rns_double_extended, 536
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 543
CONSTREFERENCE
parallel.h, 1031
convert
rns_double, 526, 527
rns_double_extended, 537
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 546
convert_transpose
rns_double, 526
ConvertTo< T >, 429
COO
FFLAS, 76
Coo
Coo< Field >, 431
Coo< ValT, IdxT >, 430, 433
coo
HelperFlag, 477
Coo< Field >, 431
col, 432
Coo, 431
deleted, 432
operator=, 432
row, 432
val, 432

Coo< ValT, IdxT >, 429, 432
col, 430, 433
Coo, 430, 433
operator=, 430, 433
row, 430, 433
Self, 430, 433
val, 430, 433
coo.h, 885
coo_spmm.inl, 886
__FFLASFFPACK_fflas_sparse_coo_spmm_INL, 886
coo_spmv.inl, 887
__FFLASFFPACK_fflas_sparse_coo_spmv_INL, 887
coo_utils.inl, 887
__FFLASFFPACK_fflas_sparse_coo_utils_INL, 888
COO_ZO
FFLAS, 76
CooMat< Field >, 434
_coo16, 434
_coo16_zo, 434
_coo32, 434
_coo32_zo, 434
_coo64, 434
_coo64_zo, 434
Copying and Licence, 2
count_nonconst_lvalue_reference< const T &, O... >, 435
n, 435
count_nonconst_lvalue_reference< T >, 435
count_nonconst_lvalue_reference< T &, O... >, 435
n, 435
count_nonconst_lvalue_reference< T, O... >, 435
n, 435
count_nonconst_lvalue_reference<>, 436
n, 436
CROUT
ffpack_pluq.inl, 939
CSC
FFLAS, 76
CSC_ZO
FFLAS, 76
CSR
FFLAS, 76
csr
HelperFlag, 477
csr.h, 888
CSR_HYB
FFLAS, 76
csr_hyb.h, 893
csr_hyb_pspmm.inl, 893
__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL, 894
csr_hyb_pspmv.inl, 894
__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL, 894
csr_hyb_spmm.inl, 894
__FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL, 895
csr_hyb_spmv.inl, 895
__FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL, 895
csr_hyb_utils.inl, 895
__FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL, 896
csr_pspmm.inl, 888
__FFLASFFPACK_fflas_sparse_CSR_pspmm_INL, 889
csr_pspmv.inl, 889
__FFLASFFPACK_fflas_sparse_CSR_pspmv_INL, 890
csr_spmm.inl, 890
__FFLASFFPACK_fflas_sparse_CSR_spmm_INL, 891
csr_spmv.inl, 891
__FFLASFFPACK_fflas_sparse_CSR_spmv_INL, 892
csr_utils.inl, 892
CSR_ZO
FFLAS, 76
CsrMat< Field >, 436
_csr16, 436
_csr16_zo, 436
_csr32, 436
_csr32_zo, 436
_csr64, 436
_csr64_zo, 436
cst
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 744
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
ctz
bit_manipulation.h, 1040
CUBE
benchmark-checkers.C, 779
benchmark-fsytrf.C, 794
benchmark-ftrtri.C, 796
benchmark-inverse.C, 797
benchmark-lqup.C, 798
benchmark-pluq.C, 799
benchmark-wino.C, 801
charpoly.C, 772
fsyrk.C, 773
fsytrf.C, 774
ftrtri.C, 774
pluq.C, 775
tuna.C, 1049
main, 1049
current
ForStrategy1D< blocksize_t, Cut, Param >, 465
ForStrategy2D< blocksize_t, Cut, Param >, 469
Cut

Parallel< C, P >, 517
cyclic_shift_col
FFPACK, 360, 363
cyclic_shift_col_modular_double
ffpack.C, 987
ffpack_c.h, 1008
cyclic_shift_mathPerm
FFPACK, 359
ffpack.C, 987
ffpack_c.h, 1008
cyclic_shift_row
FFPACK, 359, 362
cyclic_shift_row_col
FFPACK, 359, 362
cyclic_shift_row_modular_double
ffpack.C, 987
ffpack_c.h, 1008

Danilevski
FFPACK, 349
FFPACK::Protected, 393
dasum_
config-blas.h, 811
dat
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 741
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 744
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
data
Argument, 400
daxpy_
config-blas.h, 810
dcopy_
config-blas.h, 812
ddot_
config-blas.h, 810
debug.h, 1040
FFLASFFPACK_abort, 1041
FFLASFFPACK_check, 1041
DeCompressRows
FFPACK::Protected, 395
DeCompressRowsQA
FFPACK::Protected, 396
DeCompressRowsQK
FFPACK::Protected, 395
DefaultBoundedTag, 437
DefaultTag, 437
delayed
RNSIntegerMod< RNS >, 544
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 741
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 744
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
DelayedField
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502
delayedField
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 506
DelayedField_t
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502
DelayedTag, 437
deleted
Coo< Field >, 432
DENSE_THRESHOLD
fflas_sparse.h, 883
denseCols
StatsMatrix, 754
denseRows
StatsMatrix, 754
Det
FFPACK, 326, 327, 350, 369, 370
det.C, 802
main, 802
Det_modular_double
ffpack.C, 997
ffpack_c.h, 1016
deviationCol
StatsMatrix, 753
deviationColDifference
StatsMatrix, 754
deviationRow
StatsMatrix, 753
deviationRowDifference
StatsMatrix, 754
DFElt
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502
dgemm_
config-blas.h, 814
fblas.C, 1049
dgemv_
config-blas.h, 811
dger_
config-blas.h, 812
digits
limits< char >, 491
limits< double >, 492

limits< float >, 492
limits< int >, 494
limits< long >, 494
limits< long long >, 495
limits< short int >, 497
limits< signed char >, 497
limits< unsigned char >, 498
limits< unsigned int >, 499
limits< unsigned long >, 499
limits< unsigned long long >, 500
limits< unsigned short int >, 500
div
ScalFunctions< Element >, 553
Simd256_impl< true, false, true, 8 >, 625
Simd512_impl< true, false, true, 8 >, 707
DivideAndConquer, 437
dnrm2_
config-blas.h, 811
DNS_BIN_VER
read_sparse.h, 906
doApplyS
FFPACK, 355
doApplyT
FFPACK, 356
doBenchs
Bench< Elt >, 405
doTests
Test< Elt >, 759
DotProdBoundClassic
FFLAS::Protected, 212
DOUBLE_TO_FLOAT_CROSSOVER
fflas.h, 824
winograd.C, 776
dscal_
config-blas.h, 812
dtrmm_
config-blas.h, 813
dtrsm_
config-blas.h, 813
DynamicPeeling
FFLAS::Protected, 216
DynamicPeeling2
FFLAS::Protected, 216

EFFGFF
benchmark-dsytrf.C, 782
Element
FieldSimd< _Field >, 448
readMyMachineType< Field, mpz_t >, 521, 522
readMyMachineType< Field, T >, 521
rns_double, 524
rns_double_extended, 536
RNSInteger< RNS >, 539
RNSIntegerMod< RNS >, 543
TestOneMethod< Simd >, 761
Element_ptr
benchmark-fgemm-rns.C, 786
readMyMachineType< Field, mpz_t >, 521, 522
readMyMachineType< Field, T >, 521
rns_double, 524
rns_double_extended, 536
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 543
elements
array< T >, 400
vector< T >, 768
ElementTraits< double >, 438
value, 438
ElementTraits< Element >, 437
value, 438
ElementTraits< FFPACK::rns_double_elt >, 438
value, 438
ElementTraits< float >, 438
value, 439
ElementTraits< Givaro::Integer >, 439
value, 439
ElementTraits< int16_t >, 439
value, 439
ElementTraits< int32_t >, 440
value, 440
ElementTraits< int64_t >, 440
value, 440
ElementTraits< int8_t >, 440
value, 440
ElementTraits< RecInt::rint< K > >, 441
value, 441
ElementTraits< RecInt::rmint< K, MG > >, 441
value, 441
ElementTraits< RecInt::ruint< K > >, 441
value, 442
ElementTraits< uint16_t >, 442
value, 442
ElementTraits< uint32_t >, 442
value, 442
ElementTraits< uint64_t >, 443
value, 443
ElementTraits< uint8_t >, 443
value, 443
ELL
FFLAS, 76
ell
HelperFlag, 477
ell.h, 896
ell_pspmm.inl, 896
__FFLASFFPACK_fflas_sparse_ELL_pspmm_INL, 897
ell_pspmv.inl, 897
__FFLASFFPACK_fflas_sparse_ELL_pspmv_INL, 898
ELL_simd
FFLAS, 76
ell_simd.h, 900
ell_simd_pspmv.inl, 901
__FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL, 901
ell_simd_spmv.inl, 901

__FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL, 902
ell_simd_utils.inl, 902
__FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL, 903
ELL_simd_ZO
FFLAS, 76
ell_spmm.inl, 898
__FFLASFFPACK_fflas_sparse_ELL_spmm_INL, 899
ell_spmv.inl, 899
__FFLASFFPACK_fflas_sparse_ELL_spmv_INL, 899
ell_utils.inl, 899
__FFLASFFPACK_fflas_sparse_ELL_utils_INL, 900
ELL_ZO
FFLAS, 76
EllMat< Field >, 443
_ell16, 444
_ell16_zo, 444
_ell32, 444
_ell32_zo, 444
_ell64, 444
_ell64_zo, 444
Elt_ptr
Bench< Elt >, 404
Test< Elt >, 758
ENABLE_ALL_CHECKINGS
benchmark-checkers.C, 779
ffpack_ftrtr.inl, 933
test-fdot.C, 1059
test-fgemm-check.C, 1060
test-fsyr2k.C, 1071
test-fsyrk.C, 1073
test-ftrmv.C, 1077
test-ftrsm-check.C, 1078
test-ftrsm.C, 1079
test-ftrssyr2k.C, 1080
test-ftrstr.C, 1081
test-ftrsv.C, 1082
test-ftrtri.C, 1083
test-invert-check.C, 1085
test-pluq-check.C, 1095
ENABLE_CHECKER_charpoly
test-charpoly-check.C, 1052
ENABLE_CHECKER_Det
test-det-check.C, 1054
ENABLE_CHECKER_fgemm
test-fgemm.C, 1062
enable_if_no_simd_t
Bench< Elt >, 405
Test< Elt >, 758
enable_if_simd128_t
Bench< Elt >, 405
Test< Elt >, 758
enable_if_simd256_t
Bench< Elt >, 405
Test< Elt >, 758
enable_if_simd512_t
Bench< Elt >, 405
Test< Elt >, 758
enable_if_t
Bench< Elt >, 404
Test< Elt >, 758
TestOneMethod< Simd >, 761
end
ForStrategy1D< blocksize_t, Cut, Param >, 465
END_OF_ARGUMENTS
args-parser.h, 1038
END_PARALLEL_MAIN
parallel.h, 1032
eq
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 655
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 728
eval_func_on_array
test-simd.C, 1100
evaluate_scalar_method
TestOneMethod< Simd >, 761
evaluate_simd_method
TestOneMethod< Simd >, 762
example
Argument, 399
ExpandBlockBiDiagonalToBruhat
FFPACK, 343
expandLCRE
FFPACK, 347

F
Bench< Elt >, 406
Test< Elt >, 759
fadd
FFLAS, 77–79, 81, 164, 171, 172
FFLAS::details, 199, 200
fadd_1_modular_double
fflas_c.h, 954
fflas_lvl1.C, 973
fadd_2_modular_double
fflas_c.h, 957
fflas_lvl2.C, 978
faddin
FFLAS, 77, 81, 164, 172

faddin_1_modular_double
fflas_c.h, 954
fflas_lvl1.C, 974
faddin_2_modular_double
fflas_c.h, 958
fflas_lvl2.C, 978
Failure, 444
_errorStream, 446
Failure, 445
operator(), 445
print, 445
setErrorStream, 445
failure
FFPACK, 375
FailureCharpolyCheck, 446
FailureDetCheck, 446
FailureFgemmCheck, 446
FailureInvertCheck, 446
FailurePLUQCheck, 446
FailureTrsmCheck, 446
fassign
FFLAS, 82–84, 160, 164
fassign_1_modular_double
fflas_c.h, 953
fflas_lvl1.C, 972
fassign_2_modular_double
fflas_c.h, 955
fflas_lvl2.C, 975
faxpby
FFLAS, 132, 139
faxpy
FFLAS, 84, 85, 162, 169
FFLAS::details, 201
faxpy_1_modular_double
fflas_c.h, 953
fflas_lvl1.C, 973
faxpy_2_modular_double
fflas_c.h, 957
fflas_lvl2.C, 977
fblas.C, 1049
__FFLASFFPACK_CONFIGURATION, 1049
dgemm_, 1049
main, 1050
fconvert
FFLAS, 129, 137, 158
fconvert_rns
FFLAS, 156, 157
fconvert_trans_rns
FFLAS, 156
fdot
FFLAS, 86, 87, 133, 162, 181
fdot_1_modular_double
fflas_c.h, 953
fflas_lvl1.C, 973
fequal
FFLAS, 132, 136, 160, 165
fequal_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 972
fequal_2_modular_double
fflas_c.h, 955
fflas_lvl2.C, 976
FFLAS, 42, 45
alignable, 188
alignable< Givaro::Integer * >, 188
BaseTimer, 74
bitsize, 140
bitsize< Givaro::ZRing< Givaro::Integer > >, 140
BlockCuts, 179, 181
BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >, 180
BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >, 179
BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >, 181
BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >, 180
BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >, 180
BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >, 180
BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >, 179
BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >, 179
BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >, 180
BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >, 179
cblas_imptrsm, 127
Checker_fgemm, 72
Checker_ftrsm, 72
computeDeviation, 153
computeS1S2, 123
COO, 76
COO_ZO, 76
CSC, 76
CSC_ZO, 76
CSR, 76
CSR_HYB, 76
CSR_ZO, 76
ELL, 76
ELL_simd, 76
ELL_simd_ZO, 76
ELL_ZO, 76
fadd, 77–79, 81, 164, 171, 172
faddin, 77, 81, 164, 172
fassign, 82–84, 160, 164
faxpby, 132, 139
faxpy, 84, 85, 162, 169
fconvert, 129, 137, 158
fconvert_rns, 156, 157
fconvert_trans_rns, 156
fdot, 86, 87, 133, 162, 181
fequal, 132, 136, 160, 165
FFLAS_BASE, 75

fflas_delete, 155, 188
FFLAS_DIAG, 75
FFLAS_FORMAT, 76
fflas_new, 155, 156, 188
FFLAS_ORDER, 74
FFLAS_SIDE, 75
FFLAS_TRANSPOSE, 74
FFLAS_UPLO, 74
FflasAuto, 76
FflasBinary, 76
FflasColMajor, 74
FflasDense, 76
FflasDouble, 75
FflasFloat, 75
FflasGeneric, 75
FflasLeft, 75
FflasLeftTri, 75
FflasLower, 75
FflasMaple, 76
FflasMath, 76
FflasNonUnit, 75
FflasNoTrans, 74
FflasRight, 75
FflasRightTri, 75
FflasRowMajor, 74
FflasSageMath, 76
FflasSMS, 76
FflasTrans, 74
FflasUnit, 75
FflasUpper, 75
fgemm, 88–90, 92–97, 144, 176–178
fgemv, 97–102, 173, 184
fger, 103–106, 173
fidentity, 136, 137, 166
finit, 108, 110, 129, 137, 158, 167
finit_rns, 155, 156
finit_trans_rns, 155
fiszero, 131, 136, 160, 166
fmove, 139, 170
fneg, 130, 138, 159, 168
fnegin, 130, 138, 159, 168
ForceCheck_fgemm, 72
ForceCheck_ftrsm, 72
frand, 131, 135
freduce, 107–111, 157, 167
freduce_constoverride, 108, 110
freivalds, 111
fscal, 112–116, 161, 169
fscalin, 111, 113–116, 161, 169
fspmm, 145
fspmv, 145, 152
fsquare, 91, 92, 178
fsub, 78, 80, 164, 171
fsubin, 78, 81, 172
fswap, 133, 163
fsyr2k, 116
fsyrk, 117–124
fsyrk_strassen, 124, 142
ftranspose, 154
ftrmm, 124, 125, 175
ftrmv, 140
ftrsm, 126, 127, 141, 144, 145, 175
ftrsv, 128, 174
fzero, 130, 133, 135, 159, 165
getArgumentValue, 185
getDataType, 151, 152
getSeed, 189
getStat, 154
getTLBSize, 189
has_equal, 73
has_minus, 73
has_minus_eq, 73
has_mul, 73
has_mul_eq, 73
has_plus, 73
has_plus_eq, 73
HYB_ZO, 76
igemm_, 128
InfNorm, 76
max3, 77
max4, 77
maxCardinality, 154
maxCardinality< Givaro::Modular< int32_t > >, 154
maxCardinality< Givaro::Modular< int64_t > >, 154
min3, 76
min4, 77
minCardinality, 154
MKLSparseMatrixFormat, 72
mone, 76
NoSimdSparseMatrix, 72
NotMKLSparseMatrixFormat, 72
NotZOSparseMatrix, 72
number_kind, 75
one, 76
operator<<, 151
other, 76
parseArguments, 185
pfadd, 78
pfaddin, 79
pfgemm, 142, 182–184
pfgemm_1D_rec, 142
pfgemm_2D_rec, 143
pfgemm_3D_rec, 143
pfgemm_3D_rec2, 144
pfrand, 181
pfreduce, 109
pfsub, 79
pfsubin, 79
pfzero, 181
preamble, 186
prefetch, 189
queryCacheSizes, 189
queryL1CacheSize, 189
queryTopLevelCacheSize, 189

readDnsFormat, 152
readMachineType, 152
ReadMatrix, 186
readSmsFormat, 151
readSprFormat, 151
SELL, 76
SELL_ZO, 76
SimdSparseMatrix, 72
sparse_delete, 146–150, 153
sparse_init, 145–150, 153
sparse_print, 147, 150, 153
SparseMatrix_t, 76
SysTimer, 74
Timer, 73
UserTimer, 74
writeCommandString, 185
writeDnsFormat, 152
WriteMatrix, 185, 187
WritePermutation, 187
zero, 76
ZOSparseMatrix, 72
fflas-101_1.C, 1104
main, 1104
fflas-101_3.C, 1104
main, 1105
FFLAS-FFPACK, 41
FFLAS-FFPACK Documentation., 1
FFLAS-FFPACK fields, 43
fflas-ffpack-config.h, 821
GCC_VERSION, 822
fflas-ffpack-default-thresholds.h, 822
__FFLASFFPACK_ARITHPROG_THRESHOLD, 823
__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD, 822
__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD, 822
__FFLASFFPACK_FSYRK_THRESHOLD, 823
__FFLASFFPACK_FSYTRF_THRESHOLD, 823
__FFLASFFPACK_FTRTRI_THRESHOLD, 823
__FFLASFFPACK_PLUQ_THRESHOLD, 822
__FFLASFFPACK_WINOTHRESHOLD, 822
__FFLASFFPACK_WINOTHRESHOLD_BAL, 822
__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT, 822
__FFLASFFPACK_WINOTHRESHOLD_FLT, 822
fflas-ffpack-thresholds.h, 823
fflas-ffpack.doxy, 823
fflas-ffpack.h, 823
fflas.doxy, 823
fflas.h, 823
DOUBLE_TO_FLOAT_CROSSOVER, 824
WINOTHRESHOLD, 824
FFLAS::_ftranspose_impl, 190
nonsquare_inplace_v1, 190
nonsquare_inplace_v2, 190
not_inplace, 190
square_inplace, 190
FFLAS::BLAS3, 191
Bini, 192
Winograd, 193
Winograd_L_S, 196
Winograd_LR_S, 196
Winograd_R_S, 197
WinogradAcc_2_24, 194
WinogradAcc_2_27, 194
WinogradAcc_3_21, 194
WinogradAcc_3_23, 193
WinogradAcc_L_S, 196
WinogradAcc_LR, 195
WinogradAcc_R_S, 195
WinoPar, 193
FFLAS::csr_hyb_details, 197
FFLAS::CuttingStrategy, 197
RNSModulus, 198
FFLAS::details, 198
BlockingFactor, 206
fadd, 199, 200
faxpy, 201
freduce, 201, 202
fscal, 202, 203
fscalin, 202, 203
gebp, 205
igebb11, 204
igebb14, 204
igebb21, 204
igebb24, 203
igebb41, 204
igebb44, 203
igebp, 205
pack_lhs, 205
pack_rhs, 205
FFLAS::details_spmv, 206
FFLAS::ElementCategories, 206
FFLAS::FieldCategories, 206
FFLAS::MMHelperAlgo, 207
FFLAS::ModeCategories, 207
FFLAS::ParSeqHelper, 208
FFLAS::Protected, 208
computeFactorClassic, 211
DotProdBoundClassic, 212
DynamicPeeling, 216
DynamicPeeling2, 216
fgemm_convert, 212
fgemv_convert, 217
fger_convert, 218
fsquareCommon, 215
fsyrk_convert, 218
igemm, 221
igemm_colmajor, 221
MatF2MatD_Triangular, 222
MatF2MatFl_Triangular, 222
min_types, 219, 220
NeedDoublePreAddReduction, 214
NeedPreAddReduction, 213
NeedPreAxpyReduction, 219

NeedPreScalReduction, 218, 219
NeedPreSubReduction, 213
ScalAndReduce, 214, 218
TRSMBound, 212
unfit, 220, 221
WinogradCalc, 217
WinogradSteps, 216
WinogradThreshold, 215
FFLAS::sell_details, 222
FFLAS::sparse_details, 223
fspmm, 229–231
fspmm_dispatch, 228, 229
fspmv, 226–228, 236
fspmv_dispatch, 226
init_y, 226
pfspmm, 232–234
pfspmm_dispatch, 231, 232
pfspmv, 234, 235
FFLAS::sparse_details_impl, 237
fspmm, 245, 252, 253, 258, 259, 263, 272
fspmm_mone, 246, 254, 264
fspmm_mone_simd_aligned, 247, 254, 265
fspmm_mone_simd_unaligned, 247, 255, 265
fspmm_one, 246, 253, 264
fspmm_one_simd_aligned, 247, 254, 265
fspmm_one_simd_unaligned, 247, 254, 265
fspmm_simd_aligned, 246, 253
fspmm_simd_unaligned, 246, 253
fspmv, 248, 255, 259, 266, 268, 269, 273, 275
fspmv_mone, 248, 249, 256, 266, 267, 270, 276, 277
fspmv_mone_simd, 270, 276
fspmv_one, 248, 249, 255, 256, 266, 267, 269, 270, 276
fspmv_one_simd, 270, 276
fspmv_simd, 269, 275
pfspmm, 249, 256, 257, 259–261, 271
pfspmm_mone, 250
pfspmm_one, 250
pfspmm_zo, 261
pfspmv, 251, 258, 261, 262, 267, 271–274
pfspmv_mone, 252, 262, 263, 268, 274
pfspmv_one, 251, 252, 262, 268, 274
pfspmv_task, 251
FFLAS::StrategyParameter, 277
FFLAS::StructureHelper, 277
FFLAS::vectorised, 278
add, 280
addp, 279
axpyp, 280
modp, 282
reduce, 281, 282
scalp, 282, 283
sub, 280
subp, 280
VEC_ADD, 279
VEC_SUB, 279
FFLAS::vectorised::unswitch, 283
axpyp, 284
modp, 284, 285
scalp, 285
fflas_101.C, 1105
main, 1105
fflas_101_lvl1.C, 1105
main, 1105
FFLAS_BASE
FFLAS, 75
fflas_bounds.inl, 825
__FFLASFFPACK_fflas_bounds_INL, 825
FFLAS_INT_TYPE, 825
fflas_c.h, 948
fadd_1_modular_double, 954
fadd_2_modular_double, 957
faddin_1_modular_double, 954
faddin_2_modular_double, 958
fassign_1_modular_double, 953
fassign_2_modular_double, 955
faxpy_1_modular_double, 953
faxpy_2_modular_double, 957
fdot_1_modular_double, 953
fequal_1_modular_double, 952
fequal_2_modular_double, 955
FFLAS_C_BASE, 951
FFLAS_C_DIAG, 951
FFLAS_C_ORDER, 950
FFLAS_C_SIDE, 951
FFLAS_C_TRANSPOSE, 950
FFLAS_C_UPLO, 951
FFLAS_COMPILED, 950
FflasColMajor, 950
FflasDouble, 951
FflasFloat, 951
FflasGeneric, 951
FflasLeft, 951
FflasLower, 951
FflasNonUnit, 951
FflasNoTrans, 951
FflasRight, 951
FflasRowMajor, 950
FflasTrans, 951
FflasUnit, 951
FflasUpper, 951
fgemm_3_modular_double, 959
fgemv_2_modular_double, 958
fger_2_modular_double, 958
fidentity_2_modular_double, 955
fiszero_1_modular_double, 952
fiszero_2_modular_double, 955
fmove_2_modular_double, 957
fneg_1_modular_double, 952
fneg_2_modular_double, 956
fnegin_1_modular_double, 952
fnegin_2_modular_double, 956
freduce_1_modular_double, 952
freduce_2_modular_double, 956
freducein_1_modular_double, 952

freducein_2_modular_double, 956
fscal_1_modular_double, 953
fscal_2_modular_double, 956
fscalin_1_modular_double, 953
fscalin_2_modular_double, 956
fsquare_3_modular_double, 960
fsub_1_modular_double, 954
fsub_2_modular_double, 957
fsubin_1_modular_double, 954
fsubin_2_modular_double, 958
fswap_1_modular_double, 954
ftrmm_3_modular_double, 959
ftrsm_3_modular_double, 959
ftrsv_2_modular_double, 959
fzero_1_modular_double, 952
fzero_2_modular_double, 955
FFLAS_C_BASE
fflas_c.h, 951
FFLAS_C_DIAG
fflas_c.h, 951
ffpack_c.h, 1006
FFLAS_C_ORDER
fflas_c.h, 950
ffpack_c.h, 1005
FFLAS_C_SIDE
fflas_c.h, 951
ffpack_c.h, 1006
FFLAS_C_TRANSPOSE
fflas_c.h, 950
ffpack_c.h, 1005
FFLAS_C_UPLO
fflas_c.h, 951
ffpack_c.h, 1005
FFLAS_COMPILED
fflas_c.h, 950
ffpack_inst.C, 1022
ffpack_inst.h, 1023
fflas_const_cast
FFPACK, 362, 375
fflas_delete
FFLAS, 155, 188
FFLAS_DIAG
FFLAS, 75
FFLAS_ELT
fflas_L1_inst.C, 961
fflas_L1_inst.h, 962
fflas_L2_inst.C, 964
fflas_L2_inst.h, 965
fflas_L3_inst.C, 968
fflas_L3_inst.h, 969
ffpack_inst.C, 1022, 1023
ffpack_inst.h, 1023, 1024
fflas_enum.h, 825
fflas_fadd.h, 826
fflas_fadd.inl, 827
__FFLASFFPACK_fadd_INL, 828
fflas_fassign.h, 828
fflas_fassign.inl, 829
__FFLASFFPACK_fassign_INL, 829
fflas_faxpy.inl, 829
__FFLASFFPACK_faxpy_INL, 830
fflas_fdot.inl, 830
__FFLASFFPACK_fdot_INL, 831
fflas_fgemm.inl, 831
__FFLASFFPACK_fgemm_INL, 833
fflas_fgemv.inl, 840
__FFLASFFPACK_fgemv_INL, 841
fflas_fgemv_mp.inl, 841
__FFLASFFPACK_fgemv_mp_INL, 842
fflas_fger.inl, 842
__FFLASFFPACK_fger_INL, 843
fflas_fger_mp.inl, 843
__FFPACK_fger_mp_INL, 844
FFLAS_FIELD
fflas_L1_inst.C, 960, 961
fflas_L1_inst.h, 961, 962
fflas_L2_inst.C, 964
fflas_L2_inst.h, 965
fflas_L3_inst.C, 968
fflas_L3_inst.h, 969
ffpack_inst.C, 1022
ffpack_inst.h, 1023
FFLAS_FORMAT
FFLAS, 76
fflas_freduce.h, 844
fflas_freduce.inl, 845
__FFLASFFPACK_fflas_freduce_INL, 846
FFLASFFPACK_COPY_REDUCE, 846
fflas_freduce_mp.inl, 847
__FFLASFFPACK_fflas_freduce_mp_INL, 847
fflas_freivalds.inl, 847
__FFLASFFPACK_freivalds_INL, 847
fflas_fscal.h, 847
fflas_fscal.inl, 848
__FFLASFFPACK_fscal_INL, 849
fflas_fscal_mp.inl, 849
__FFLASFFPACK_fscal_mp_INL, 850
fflas_fsyr2k.inl, 850
__FFLASFFPACK_fflas_fsyr2k_INL, 850
fflas_fsyrk.inl, 850
__FFLASFFPACK_fflas_fsyrk_INL, 852
fflas_fsyrk_strassen.inl, 852
__FFLASFFPACK_fflas_fsyrk_strassen_INL, 853
fflas_ftrmm.inl, 853
__FFLASFFPACK_ftrmm_INL, 854
fflas_ftrsm.inl, 854
__FFLASFFPACK_ftrsm_INL, 854
fflas_ftrsm_mp.inl, 854
__FFPACK_ftrsm_mp_INL, 855
fflas_ftrsv.inl, 855
__FFLASFFPACK_ftrsv_INL, 855
fflas_helpers.inl, 855
__FFLASFFPACK_fflas_fflas_mmhelper_INL, 856
FFLAS_INT_TYPE
fflas_bounds.inl, 825
fflas_intrinsic.h, 1041

fflas_io.h, 1041
fflas_L1_inst.C, 960
__FFLAS_L1_INST_C, 960
FFLAS_ELT, 961
FFLAS_FIELD, 960, 961
INST_OR_DECL, 960
fflas_L1_inst.h, 961
FFLAS_ELT, 962
FFLAS_FIELD, 961, 962
INST_OR_DECL, 961
fflas_L1_inst_implem.inl, 962
fflas_L2_inst.C, 963
__FFLAS_L2_INST_C, 964
FFLAS_ELT, 964
FFLAS_FIELD, 964
INST_OR_DECL, 964
fflas_L2_inst.h, 964
FFLAS_ELT, 965
FFLAS_FIELD, 965
INST_OR_DECL, 965
fflas_L2_inst_implem.inl, 965
fflas_L3_inst.C, 967
__FFLAS_L3_INST_C, 967
FFLAS_ELT, 968
FFLAS_FIELD, 968
INST_OR_DECL, 967
fflas_L3_inst.h, 968
FFLAS_ELT, 969
FFLAS_FIELD, 969
INST_OR_DECL, 969
fflas_L3_inst_implem.inl, 969
__FFLAS__TRSM_READONLY, 970
fflas_level1.inl, 860
__FFLASFFPACK_fflas_fflas_level1_INL, 862
fflas_level2.inl, 862
__FFLASFFPACK_fflas_fflas_level2_INL, 865
fflas_level3.inl, 865
__FFLASFFPACK_fflas_fflas_level3_INL, 867
__FFLAS__TRSM_READONLY, 867
fflas_lvl1.C, 970
fadd_1_modular_double, 973
faddin_1_modular_double, 974
fassign_1_modular_double, 972
faxpy_1_modular_double, 973
fdot_1_modular_double, 973
fequal_1_modular_double, 972
fiszero_1_modular_double, 972
fneg_1_modular_double, 971
fnegin_1_modular_double, 971
freduce_1_modular_double, 971
freducein_1_modular_double, 971
fscal_1_modular_double, 972
fscalin_1_modular_double, 972
fsub_1_modular_double, 973
fsubin_1_modular_double, 974
fswap_1_modular_double, 973
fzero_1_modular_double, 972
fflas_lvl2.C, 974
fadd_2_modular_double, 978
faddin_2_modular_double, 978
fassign_2_modular_double, 975
faxpy_2_modular_double, 977
fequal_2_modular_double, 976
fgemv_2_modular_double, 979
fger_2_modular_double, 979
fidentity_2_modular_double, 976
fiszero_2_modular_double, 976
fmove_2_modular_double, 978
fneg_2_modular_double, 977
fnegin_2_modular_double, 977
freduce_2_modular_double, 976
freducein_2_modular_double, 976
fscal_2_modular_double, 977
fscalin_2_modular_double, 977
fsub_2_modular_double, 978
fsubin_2_modular_double, 978
ftrsv_2_modular_double, 979
fzero_2_modular_double, 975
fflas_lvl3.C, 980
fgemm_3_modular_double, 981
fsquare_3_modular_double, 981
ftrmm_3_modular_double, 980
ftrsm_3_modular_double, 980
fflas_memory.h, 1042
fflas_new
FFLAS, 155, 156, 188
FFLAS_ORDER
FFLAS, 74
fflas_pfgemm.inl, 867
__FFLASFFPACK_DIMKPENALTY, 868
__FFLASFFPACK_SEQPARTHRESHOLD, 868
__FFLASFFPACK_fflas_pfgemm_INL, 868
fflas_pftrsm.inl, 868
__FFLASFFPACK_fflas_pftrsm_INL, 869
PTRSM_HYBRID_THRESHOLD, 869
fflas_plevel1.h, 1029
fflas_randommatrix.h, 1043
FFLAS_SIDE
FFLAS, 75
fflas_simd.h, 869
CONST, 870
FLOAT_MOD, 870
INLINE, 870
NORML_MOD, 870
PURE, 870
Simd, 870
SIMD_INT, 870
fflas_sparse.C, 981
fflas_sparse.h, 879
__FFLASFFPACK_CACHE_LINE_SIZE, 883
assume_aligned, 883
DENSE_THRESHOLD, 883
index_t, 883
ROUND_DOWN, 883
fflas_sparse.inl, 883
__FFLASFFPACK_fflas_fflas_sparse_INL, 885

FFLAS_TRANSPOSE
FFLAS, 74
fflas_transpose.h, 911
FFLAS_TRANSPOSE_BLOCKSIZE, 912
LD, 912
ST, 912
FFLAS_TRANSPOSE_BLOCKSIZE
fflas_transpose.h, 912
FFLAS_UPLO
FFLAS, 74
FflasAuto
FFLAS, 76
FflasBinary
FFLAS, 76
FflasColMajor
FFLAS, 74
fflas_c.h, 950
ffpack_c.h, 1005
FflasDense
FFLAS, 76
FflasDouble
FFLAS, 75
fflas_c.h, 951
FFLASFFPACK_abort
debug.h, 1041
FFLASFFPACK_check
debug.h, 1041
FFLASFFPACK_checkers_fflas_inl_H
checkers_fflas.inl, 807
FFLASFFPACK_checkers_ffpack_inl_H
checkers_ffpack.inl, 808
FFLASFFPACK_COPY_REDUCE
fflas_freduce.inl, 846
FFLASFFPACK_PERM_BKSIZE
ffpack_permutation.inl, 938
FflasFloat
FFLAS, 75
fflas_c.h, 951
FflasGeneric
FFLAS, 75
fflas_c.h, 951
FflasLeft
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FflasLeftTri
FFLAS, 75
FflasLower
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FflasMaple
FFLAS, 76
FflasMath
FFLAS, 76
FflasNonUnit
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FflasNoTrans
FFLAS, 74
fflas_c.h, 951
ffpack_c.h, 1005
FflasRight
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FflasRightTri
FFLAS, 75
FflasRowMajor
FFLAS, 74
fflas_c.h, 950
ffpack_c.h, 1005
FflasSageMath
FFLAS, 76
FflasSMS
FFLAS, 76
FflasTrans
FFLAS, 74
fflas_c.h, 951
ffpack_c.h, 1005
FflasUnit
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FflasUpper
FFLAS, 75
fflas_c.h, 951
ffpack_c.h, 1006
FFPACK, 43, 286
_PLUQ, 361
applyP, 304, 305, 363
applyP_block, 355
Bruhat2EchelonPermutation, 343
buildMatrix, 349
CharPoly, 321, 322, 349, 367, 368
Checker_charpoly, 303
Checker_Det, 302
Checker_invert, 303
Checker_PLUQ, 302
chooseField, 388
chooseField< Givaro::ZRing< double > >, 389
chooseField< Givaro::ZRing< float > >, 388
chooseField< Givaro::ZRing< int32_t > >, 388
chooseField< Givaro::ZRing< int64_t > >, 388
ColRankProfileSubmatrix, 334, 372
ColRankProfileSubmatrixIndices, 333, 372
ColumnEchelonForm, 315, 316, 366
ColumnRankProfile, 330, 331, 371
composePermutationsLLL, 358
composePermutationsLLM, 358
composePermutationsMLM, 359
CompressToBlockBiDiagonal, 342
ComputeRPermutation, 344, 347
cyclic_shift_col, 360, 363
cyclic_shift_mathPerm, 359

cyclic_shift_row, 359, 362
cyclic_shift_row_col, 359, 362
Danilevski, 349
Det, 326, 327, 350, 369, 370
doApplyS, 355
doApplyT, 356
ExpandBlockBiDiagonalToBruhat, 343
expandLCRE, 347
failure, 375
fflas_const_cast, 362, 375
fgesv, 307, 308, 364
fgetrs, 306, 363
ForceCheck_charpoly, 303
ForceCheck_Det, 303
ForceCheck_invert, 303
ForceCheck_PLUQ, 303
fsytrf, 311, 312
fsytrf_BC_Crout, 350
fsytrf_BC_RL, 350
fsytrf_LOW_RPM_BC_Crout, 351
fsytrf_nonunit, 312, 352
fsytrf_RPM, 352
fsytrf_UP_RPM, 351
fsytrf_UP_RPM_BC_Crout, 351
fsytrf_UP_RPM_BC_RL, 351
ftrssyr2k, 310
ftrstr, 310
ftrtri, 309, 364
ftrtrm, 309, 365
getEchelonForm, 336
getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >, 373
getEchelonTransform, 337
getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >, 373
getLTBruhatGen, 341
getReducedEchelonForm, 338, 339
getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >, 374
getReducedEchelonTransform, 339
getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >, 374
getTriangular, 335
getTriangular< FFLAS_FIELD< FFLAS_ELT > >, 372, 373
getTridiagonal, 352
Invert, 320, 367
Invert2, 321, 367
isOdd, 375
IsSingular, 325, 369
KrylovElim, 369
LAPACKPerm2MathPerm, 303
LeadingSubmatrixRankProfiles, 331
LQUPtoInverseOfFullRankMinor, 345, 375
LTBruhatGen, 340
LTQSorder, 342
LUdivine, 314, 353, 354, 365
LUdivine_gauss, 353, 366
LUdivine_small, 353, 365
MathPerm2LAPACKPerm, 303
MatrixApplyS, 355, 356
MatrixApplyT, 357
MatVecMinPoly, 324, 368
MinPoly, 323, 368
MonotonicApplyP, 305
MonotonicCompress, 354
MonotonicCompressCycles, 354
MonotonicCompressMorePivots, 354
MonotonicExpand, 355
NonZeroRandomMatrix, 375, 376
NullSpaceBasis, 328, 371
pColumnEchelonForm, 316
pColumnRankProfile, 331
pDet, 326
PermApplyS, 356
PermApplyT, 358
PLUQ, 313, 314, 361, 362, 365
PLUQ_basecaseCrout, 360
PLUQ_basecaseV2, 360
PLUQ_basecaseV3, 360
PLUQtoEchelonPermutation, 340
pPLUQ, 313
pRank, 325
pReducedColumnEchelonForm, 318
pReducedRowEchelonForm, 319
productBruhatxTS, 344, 347
pRowEchelonForm, 317
pRowRankProfile, 330
pSolve, 328
RandInt, 379
RandomIndexSubset, 380
RandomLTQSMatrixWithRankandQSorder, 388
RandomLTQSRankProfileMatrix, 382
RandomMatrix, 376, 377
RandomMatrixWithDet, 387
RandomMatrixWithRank, 379, 380
RandomMatrixWithRankandRandomRPM, 385
RandomMatrixWithRankandRPM, 382, 383
RandomNullSpaceVector, 328, 345, 371
RandomPermutation, 381
RandomRankProfileMatrix, 381
RandomSymmetricMatrix, 379
RandomSymmetricMatrixWithRankandRandomRPM, 386
RandomSymmetricMatrixWithRankandRPM, 383, 384
RandomSymmetricRankProfileMatrix, 382
RandomTriangularMatrix, 377, 378
Rank, 324, 325, 369
RankProfileFromLU, 331
ReducedColumnEchelonForm, 317, 318, 367
ReducedRowEchelonForm, 319, 366
RowEchelonForm, 316, 317, 366
RowRankProfile, 329, 330, 371
RowRankProfileSubmatrix, 333, 372
RowRankProfileSubmatrixIndices, 332, 372

Solve, 327, 370
solveLB, 346, 370
solveLB2, 346, 370
SpecRankProfile, 369
swapval, 381
threads_fgemm, 361
threads_ftrsm, 361
TInverter, 344, 347
trinv_left, 309, 365
ffpack-fgesv.C, 1106
main, 1106
ffpack-solve.C, 1106
main, 1106
ffpack.C, 982
applyP_modular_double, 987
ColRankProfileSubmatrix_modular_double, 999
ColRankProfileSubmatrixIndices_modular_double, 999
ColumnEchelonForm_modular_double, 990
ColumnEchelonForm_modular_float, 990
ColumnEchelonForm_modular_int32_t, 991
ColumnRankProfile_modular_double, 998
composePermutationsLLL, 986
composePermutationsLLM, 986
composePermutationsMLM, 986
cyclic_shift_col_modular_double, 987
cyclic_shift_mathPerm, 987
cyclic_shift_row_modular_double, 987
Det_modular_double, 997
fgesv_modular_double, 988
fgesvin_modular_double, 988
fgetrsin_modular_double, 987
fgetrsv_modular_double, 988
ftrtri_modular_double, 988
ftrtrm_modular_double, 989
getEchelonForm_modular_double, 1000
getEchelonFormin_modular_double, 1000
getEchelonTransform_modular_double, 1001
getReducedEchelonForm_modular_double, 1001
getReducedEchelonFormin_modular_double, 1001
getReducedEchelonTransform_modular_double, 1001
getTriangular_modular_double, 1000
getTriangularin_modular_double, 1000
Invert2_modular_double, 996
Invert_modular_double, 995
Invertin_modular_double, 995
IsSingular_modular_double, 996
KrylovElim_modular_double, 996
LAPACKPerm2MathPerm, 985
LeadingSubmatrixRankProfiles, 998
LUdivine_modular_double, 989
MathPerm2LAPACKPerm, 985
MatrixApplyS_modular_double, 985
MatrixApplyT_modular_double, 986
NullSpaceBasis_modular_double, 998
pColumnEchelonForm_modular_double, 992
pColumnEchelonForm_modular_float, 993
pColumnEchelonForm_modular_int32_t, 994
PermApplyS_double, 986
PermApplyT_double, 986
PLUQ_modular_double, 989
PLUQtoEchelonPermutation, 1002
pReducedColumnEchelonForm_modular_double, 993
pReducedColumnEchelonForm_modular_float, 994
pReducedColumnEchelonForm_modular_int32_t, 995
pReducedRowEchelonForm_modular_double, 993
pReducedRowEchelonForm_modular_float, 994
pReducedRowEchelonForm_modular_int32_t, 995
pRowEchelonForm_modular_double, 993
pRowEchelonForm_modular_float, 994
pRowEchelonForm_modular_int32_t, 994
RandomNullSpaceVector_modular_double, 997
Rank_modular_double, 996
RankProfileFromLU, 998
ReducedColumnEchelonForm_modular_double, 990
ReducedColumnEchelonForm_modular_float, 991
ReducedColumnEchelonForm_modular_int32_t, 992
ReducedRowEchelonForm_modular_double, 990
ReducedRowEchelonForm_modular_float, 991
ReducedRowEchelonForm_modular_int32_t, 992
RowEchelonForm_modular_double, 990
RowEchelonForm_modular_float, 991
RowEchelonForm_modular_int32_t, 992
RowRankProfile_modular_double, 998
RowRankProfileSubmatrix_modular_double, 999
RowRankProfileSubmatrixIndices_modular_double, 999
Solve_modular_double, 997
solveLB2_modular_double, 997
solveLB_modular_double, 997
SpecRankProfile_modular_double, 996
trinv_left_modular_double, 989
ffpack.doxy, 912
ffpack.h, 912
__FFLASFFPACK_FTRSSYR2K_THRESHOLD, 921
__FFLASFFPACK_FTRSTR_THRESHOLD, 921
ffpack.inl, 921
__FFLASFFPACK_ffpack_INL, 922
FFPACK::Protected, 389
ArithProg, 393
CompressRows, 394
CompressRowsQA, 395
CompressRowsQK, 395
Danilevski, 393
DeCompressRows, 395
DeCompressRowsQA, 396
DeCompressRowsQK, 395
fgemv_kgf, 392

GaussJordan, 391
Hybrid_KGF_LUK_MinPoly, 394
KellerGehrig, 391
KGFast, 392
KGFast_generalized, 392
LUdivine_construct, 390, 396
LUKrylov, 392
LUKrylov_KGFast, 393
MatVecMinPoly, 393
newD, 394
RandomKrylovPrecond, 393
updateD, 394
ffpack_bruhatgen.inl, 922
__FFLASFFPACK_ffpack_bruhatgen_inl, 924
ffpack_c.h, 1002
applyP_modular_double, 1009
ColRankProfileSubmatrix_modular_double, 1019
ColRankProfileSubmatrixIndices_modular_double, 1019
ColumnEchelonForm_modular_double, 1012
ColumnEchelonForm_modular_float, 1012
ColumnEchelonForm_modular_int32_t, 1012
ColumnRankProfile_modular_double, 1018
composePermutationsLLL, 1008
composePermutationsLLM, 1008
composePermutationsMLM, 1008
cyclic_shift_col_modular_double, 1008
cyclic_shift_mathPerm, 1008
cyclic_shift_row_modular_double, 1008
Det_modular_double, 1016
FFLAS_C_DIAG, 1006
FFLAS_C_ORDER, 1005
FFLAS_C_SIDE, 1006
FFLAS_C_TRANSPOSE, 1005
FFLAS_C_UPLO, 1005
FflasColMajor, 1005
FflasLeft, 1006
FflasLower, 1006
FflasNonUnit, 1006
FflasNoTrans, 1005
FflasRight, 1006
FflasRowMajor, 1005
FflasTrans, 1005
FflasUnit, 1006
FflasUpper, 1006
FFPACK_C_CHARPOLY_TAG, 1006
FFPACK_C_LU_TAG, 1006
FFPACK_C_MINPOLY_TAG, 1006
FFPACK_COMPILED, 1005
FfpackArithProg, 1006
FfpackDanilevski, 1006
FfpackDense, 1007
FfpackHybrid, 1006
FfpackKG, 1006
FfpackKGF, 1007
FfpackKGFast, 1006
FfpackKGFastG, 1006
FfpackLUK, 1006
FfpackSingular, 1006
FfpackSlabRecursive, 1006
FfpackTileRecursive, 1006
fgesv_modular_double, 1010
fgesvin_modular_double, 1009
fgetrs_modular_double, 1009
fgetrsin_modular_double, 1009
ftrtri_modular_double, 1010
ftrtrm_modular_double, 1010
getEchelonForm_modular_double, 1020
getEchelonFormin_modular_double, 1020
getEchelonTransform_modular_double, 1020
getReducedEchelonForm_modular_double, 1021
getReducedEchelonFormin_modular_double, 1021
getReducedEchelonTransform_modular_double, 1021
getTriangular_modular_double, 1019
getTriangularin_modular_double, 1019
Invert2_modular_double, 1015
Invert_modular_double, 1015
Invertin_modular_double, 1015
IsSingular_modular_double, 1016
KrylovElim_modular_double, 1015
LAPACKPerm2MathPerm, 1007
LeadingSubmatrixRankProfiles, 1018
LUdivine_gauss_modular_double, 1011
LUdivine_modular_double, 1011
LUdivine_small_modular_double, 1011
MathPerm2LAPACKPerm, 1007
MatrixApplyS_modular_double, 1007
MatrixApplyT_modular_double, 1007
NullSpaceBasis_modular_double, 1017
PermApplyS_double, 1007
PermApplyT_double, 1008
PLUQ_modular_double, 1011
PLUQtoEchelonPermutation, 1021
RandomNullSpaceVector_modular_double, 1017
Rank_modular_double, 1016
RankProfileFromLU, 1018
ReducedColumnEchelonForm_modular_double, 1013
ReducedColumnEchelonForm_modular_float, 1013
ReducedColumnEchelonForm_modular_int32_t, 1014
ReducedRowEchelonForm2_modular_double, 1014
ReducedRowEchelonForm_modular_double, 1013
ReducedRowEchelonForm_modular_float, 1014
ReducedRowEchelonForm_modular_int32_t, 1014
REF_modular_double, 1015
RowEchelonForm_modular_double, 1012
RowEchelonForm_modular_float, 1012
RowEchelonForm_modular_int32_t, 1013
RowRankProfile_modular_double, 1018
RowRankProfileSubmatrix_modular_double, 1019

RowRankProfileSubmatrixIndices_modular_double, 1018
Solve_modular_double, 1016
solveLB2_modular_double, 1017
solveLB_modular_double, 1017
SpecRankProfile_modular_double, 1016
trinv_left_modular_double, 1010
FFPACK_C_CHARPOLY_TAG
ffpack_c.h, 1006
FFPACK_C_LU_TAG
ffpack_c.h, 1006
FFPACK_C_MINPOLY_TAG
ffpack_c.h, 1006
ffpack_charpoly.inl, 924
__FFLASFFPACK_charpoly_INL, 924
ffpack_charpoly_danilevski.inl, 924
__FFLASFFPACK_ffpack_charpoly_danilveski_INL, 925
ffpack_charpoly_kgfast.inl, 925
__FFLASFFPACK_ffpack_charpoly_kgfast_INL, 925
ffpack_charpoly_kgfastgeneralized.inl, 925
__FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL, 926
ffpack_charpoly_kglu.inl, 926
__FFLASFFPACK_ffpack_charpoly_kglu_INL, 926
ffpack_charpoly_mp.inl, 926
__FFPACK_charpoly_mp_INL, 927
FFPACK_COMPILED
ffpack_c.h, 1005
ffpack_det_mp.inl, 927
__FFPACK_det_mp_INL, 927
ffpack_echelonforms.inl, 928
__FFLASFFPACK_GAUSSJORDAN_BASECASE, 929
__FFLASFFPACK_ffpack_echelon_forms_INL, 929
ffpack_fgesv.inl, 929
__FFLASFFPACK_ffpack_fgesv_INL, 929
ffpack_fgetrs.inl, 929
__FFLASFFPACK_ffpack_fgetrs_INL, 930
ffpack_frobenius.inl, 930
ffpack_fsytrf.inl, 931
__FFLASFFPACK_ffpack_fsytrf_INL, 932
ffpack_ftrssyr2k.inl, 932
__FFLASFFPACK_ffpack_ftrssyr2k_INL, 932
ffpack_ftrstr.inl, 932
__FFLASFFPACK_ffpack_ftrstr_INL, 933
ffpack_ftrtr.inl, 933
__FFLASFFPACK_ffpack_ftrtr_INL, 933
ENABLE_ALL_CHECKINGS, 933
ffpack_inst.C, 1022
__FFPACK_INST_C, 1022
FFLAS_COMPILED, 1022
FFLAS_ELT, 1022, 1023
FFLAS_FIELD, 1022
INST_OR_DECL, 1022
ffpack_inst.h, 1023
FFLAS_COMPILED, 1023
FFLAS_ELT, 1023, 1024
FFLAS_FIELD, 1023
INST_OR_DECL, 1023
ffpack_inst_implem.inl, 1024
ffpack_invert.inl, 933
__FFLASFFPACK_ffpack_invert_INL, 934
ffpack_krylovelim.inl, 934
__FFLASFFPACK_ffpack_krylovelim_INL, 934
ffpack_ludivine.inl, 934
__FFLASFFPACK_ffpack_ludivine_INL, 935
ffpack_ludivine_mp.inl, 935
__FFPACK_ludivine_mp_INL, 935
ffpack_minpoly.inl, 936
__FFLASFFPACK_ffpack_minpoly_INL, 936
ffpack_permutation.inl, 936
__FFLASFFPACK_ffpack_permutation_INL, 938
FFLASFFPACK_PERM_BKSIZE, 938
ffpack_pluq.inl, 939
__FFLASFFPACK_ffpack_pluq_INL, 939
CROUT, 939
ffpack_pluq_mp.inl, 939
__FFPACK_pluq_mp_INL, 940
ffpack_ppluq.inl, 940
__FFLASFFPACK_ffpack_ppluq_INL, 940
__FFLAS__TRSM_READONLY, 940
PBASECASE_K, 940
ffpack_rankprofiles.inl, 941
__FFLASFFPACK_ffpack_rank_profiles_INL, 942
FfpackArithProg
ffpack_c.h, 1006
FfpackDanilevski
ffpack_c.h, 1006
FfpackDense
ffpack_c.h, 1007
FfpackHybrid
ffpack_c.h, 1006
FfpackKG
ffpack_c.h, 1006
FfpackKGF
ffpack_c.h, 1007
FfpackKGFast
ffpack_c.h, 1006
FfpackKGFastG
ffpack_c.h, 1006
FfpackLUK
ffpack_c.h, 1006
FfpackSingular
ffpack_c.h, 1006
FfpackSlabRecursive
ffpack_c.h, 1006
FfpackTileRecursive
ffpack_c.h, 1006
fgemm
FFLAS, 88–90, 92–97, 144, 176–178
fgemm_3_modular_double
fflas_c.h, 959
fflas_lvl3.C, 981

fgemm_classical.inl, 833
fgemm_classical_mp.inl, 833
__FFPACK_fgemm_classical_INL, 835
fgemm_convert
FFLAS::Protected, 212
fgemm_winograd.inl, 835
__FFLASFFPACK_fflas_fflas_fgemm_winograd_INL, 836
NEWWINO, 836
fgemv
FFLAS, 97–102, 173, 184
fgemv_2_modular_double
fflas_c.h, 958
fflas_lvl2.C, 979
fgemv_convert
FFLAS::Protected, 217
fgemv_kgf
FFPACK::Protected, 392
fger
FFLAS, 103–106, 173
fger_2_modular_double
fflas_c.h, 958
fflas_lvl2.C, 979
fger_convert
FFLAS::Protected, 218
fgesv
FFPACK, 307, 308, 364
fgesv_modular_double
ffpack.C, 988
ffpack_c.h, 1010
fgesvin_modular_double
ffpack.C, 988
ffpack_c.h, 1009
fgetrs
FFPACK, 306, 363
fgetrs_modular_double
ffpack_c.h, 1009
fgetrsin_modular_double
ffpack.C, 987
ffpack_c.h, 1009
fgetrsv_modular_double
ffpack.C, 988
fidentity
FFLAS, 136, 137, 166
fidentity_2_modular_double
fflas_c.h, 955
fflas_lvl2.C, 976
Field
Bench< Elt >, 404
benchmark-fgemm-rns.C, 786
benchmark-pluq.C, 799
FieldSimd< _Field >, 448
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 734
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 741
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
Test< Elt >, 758
test-compressQ.C, 1053
field
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, 401
associatedDelayedField< const Givaro::Modular< T, X > >, 402
associatedDelayedField< const Givaro::ModularBalanced< T > >, 402
associatedDelayedField< const Givaro::ZRing< T > >, 403
associatedDelayedField< Field >, 401
field-traits.h, 942
field.doxy, 944
FieldMax
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
FieldMin
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
FieldSimd
FieldSimd< _Field >, 448
FieldSimd< _Field >, 447
add, 449
add_r, 449
addin, 449
addin_r, 449
alignment, 452
axpy, 451
axpy_r, 452
axpyin, 451
axpyin_r, 452
Element, 448
Field, 448
FieldSimd, 448
init, 448, 449
maxpy, 452
maxpyin, 452
mod, 450
mul, 450, 451
mul_r, 451
mulin, 451
operator=, 448
scalar_t, 448
simd, 448
sub, 449, 450
sub_r, 450
subin, 450
subin_r, 450
vect_size, 452
vect_t, 448
zero, 450
FieldTraits< FFPACK::RNSInteger< T > >, 453
balanced, 454

category, 454
FieldTraits< FFPACK::RNSIntegerMod< T > >, 454
balanced, 454
category, 454
FieldTraits< Field >, 453
balanced, 453
category, 453
FieldTraits< Givaro::Modular< Element > >, 455
balanced, 455
category, 455
FieldTraits< Givaro::ModularBalanced< Element > >, 455
balanced, 456
category, 456
FieldTraits< Givaro::ZRing< double > >, 456
balanced, 456
category, 456
FieldTraits< Givaro::ZRing< float > >, 457
balanced, 457
category, 457
FieldTraits< Givaro::ZRing< Givaro::Integer > >, 457
balanced, 458
category, 458
FieldTraits< Givaro::ZRing< int16_t > >, 458
balanced, 458
category, 458
FieldTraits< Givaro::ZRing< int32_t > >, 459
balanced, 459
category, 459
FieldTraits< Givaro::ZRing< int64_t > >, 459
balanced, 460
category, 459
FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >, 460
balanced, 460
category, 460
FieldTraits< Givaro::ZRing< uint16_t > >, 460
balanced, 461
category, 461
FieldTraits< Givaro::ZRing< uint32_t > >, 461
balanced, 462
category, 461
FieldTraits< Givaro::ZRing< uint64_t > >, 462
balanced, 462
category, 462
fill_value
benchmark-fgemv.C, 790
findArgument
args-parser.h, 1039
finit
FFLAS, 108, 110, 129, 137, 158, 167
finit_rns
FFLAS, 155, 156
finit_trans_rns
FFLAS, 155
first_component
Compose< H1, H2 >, 422
firstBlockSize
ForStrategy1D< blocksize_t, Cut, Param >, 465
fiszero
FFLAS, 131, 136, 160, 166
fiszero_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 972
fiszero_2_modular_double
fflas_c.h, 955
fflas_lvl2.C, 976
Fixed, 462
FixedPrecIntTag, 462
flimits.h, 1045
in_range, 1046
FLOAT_MOD
fflas_simd.h, 870
FloatingPointTestDistribution
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution, 463
Floats
benchmark-dgemm.C, 780
floor
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
Simd256_impl< true, false, true, 8 >, 627
Simd512_impl< true, false, true, 8 >, 709
fma
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
fmadd
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 654
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 683, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 727
fmaddin
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606

Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 654
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 683, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 727
fmaddx
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 727
fmaddxin
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 644, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 727
fmove
FFLAS, 139, 170
fmove_2_modular_double
fflas_c.h, 957
fflas_lvl2.C, 978
fmsub
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 655
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 684, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 728
fmsubin
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 655
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 684, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 728
fmsubx
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 576
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 645, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 728
fmsubxin
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 566
Simd128_impl< true, true, false, 4 >, 576
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 645, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 674

Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, true, false, 8 >, 715
Simd512_impl< true, true, true, 8 >, 728
fneg
FFLAS, 130, 138, 159, 168
fneg_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 971
fneg_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 977
fnegin
FFLAS, 130, 138, 159, 168
fnegin_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 971
fnegin_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 977
fnmadd
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 654
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 727
fnmaddin
ScalFunctions< Element >, 553
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 654
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 727
fnmaddx
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 644, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 727
fnmaddxin
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 598
Simd128_impl< true, true, true, 4 >, 607
Simd128_impl< true, true, true, 8 >, 616
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 645, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 684, 690
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 727
FOR1D
parallel.h, 1032
FOR2D
parallel.h, 1033
FORBLOCK1D
parallel.h, 1032
FORBLOCK2D
parallel.h, 1032
ForceCheck_charpoly
FFPACK, 303
ForceCheck_Det
FFPACK, 303
ForceCheck_fgemm
FFLAS, 72
ForceCheck_ftrsm
FFLAS, 72
ForceCheck_invert
FFPACK, 303
ForceCheck_PLUQ
FFPACK, 303
ForStrategy1D
ForStrategy1D< blocksize_t, Cut, Param >, 464
ForStrategy1D< blocksize_t, Cut, Param >, 463
begin, 464
blockindex, 465

build, 464
changeBS, 465
current, 465
end, 465
firstBlockSize, 465
ForStrategy1D, 464
ibeg, 465
iend, 465
initialize, 464
isTerminated, 464
lastBlockSize, 465
numBlock, 466
numblocks, 465
operator++, 465
ForStrategy2D
ForStrategy2D< blocksize_t, Cut, Param >, 467
ForStrategy2D< blocksize_t, Cut, Param >, 466
_ibeg, 468
_iend, 468
_jbeg, 468
_jend, 468
blockindex, 468
BLOCKS, 469
changeCBS, 469
changeRBS, 469
colblockindex, 468
colBlockSize, 469
colnumblocks, 467
current, 469
ForStrategy2D, 467
ibegin, 467
iend, 467
initialize, 467
isTerminated, 467
jbegin, 467
jend, 467
lastCBS, 469
lastRBS, 469
numColBlock, 469
numRowBlock, 469
operator<<, 468
operator++, 467
rowblockindex, 468
rowBlockSize, 468
rownumblocks, 467
frand
FFLAS, 131, 135
freduce
FFLAS, 107–111, 157, 167
FFLAS::details, 201, 202
freduce_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 971
freduce_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 976
freduce_constoverride
FFLAS, 108, 110
freducein_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 971
freducein_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 976
freivalds
FFLAS, 111
fscal
FFLAS, 112–116, 161, 169
FFLAS::details, 202, 203
fscal_1_modular_double
fflas_c.h, 953
fflas_lvl1.C, 972
fscal_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 977
fscalin
FFLAS, 111, 113–116, 161, 169
FFLAS::details, 202, 203
fscalin_1_modular_double
fflas_c.h, 953
fflas_lvl1.C, 972
fscalin_2_modular_double
fflas_c.h, 956
fflas_lvl2.C, 977
fspmm
FFLAS, 145
FFLAS::sparse_details, 229–231
FFLAS::sparse_details_impl, 245, 252, 253, 258, 259, 263, 272
fspmm_dispatch
FFLAS::sparse_details, 228, 229
fspmm_mone
FFLAS::sparse_details_impl, 246, 254, 264
fspmm_mone_simd_aligned
FFLAS::sparse_details_impl, 247, 254, 265
fspmm_mone_simd_unaligned
FFLAS::sparse_details_impl, 247, 255, 265
fspmm_one
FFLAS::sparse_details_impl, 246, 253, 264
fspmm_one_simd_aligned
FFLAS::sparse_details_impl, 247, 254, 265
fspmm_one_simd_unaligned
FFLAS::sparse_details_impl, 247, 254, 265
fspmm_simd_aligned
FFLAS::sparse_details_impl, 246, 253
fspmm_simd_unaligned
FFLAS::sparse_details_impl, 246, 253
fspmv
FFLAS, 145, 152
FFLAS::sparse_details, 226–228, 236
FFLAS::sparse_details_impl, 248, 255, 259, 266, 268, 269, 273, 275
fspmv_dispatch
FFLAS::sparse_details, 226
fspmv_mone

FFLAS::sparse_details_impl, 248, 249, 256, 266, 267, 270, 276, 277
fspmv_mone_simd
FFLAS::sparse_details_impl, 270, 276
fspmv_one
FFLAS::sparse_details_impl, 248, 249, 255, 256, 266, 267, 269, 270, 276
fspmv_one_simd
FFLAS::sparse_details_impl, 270, 276
fspmv_simd
FFLAS::sparse_details_impl, 269, 275
fsquare
FFLAS, 91, 92, 178
fsquare_3_modular_double
fflas_c.h, 960
fflas_lvl3.C, 981
fsquareCommon
FFLAS::Protected, 215
fsub
FFLAS, 78, 80, 164, 171
fsub_1_modular_double
fflas_c.h, 954
fflas_lvl1.C, 973
fsub_2_modular_double
fflas_c.h, 957
fflas_lvl2.C, 978
fsubin
FFLAS, 78, 81, 172
fsubin_1_modular_double
fflas_c.h, 954
fflas_lvl1.C, 974
fsubin_2_modular_double
fflas_c.h, 958
fflas_lvl2.C, 978
fswap
FFLAS, 133, 163
fswap_1_modular_double
fflas_c.h, 954
fflas_lvl1.C, 973
fsyr2k
FFLAS, 116
fsyrk
FFLAS, 117–124
fsyrk.C, 773
CUBE, 773
GFOPS, 773
main, 773
fsyrk_convert
FFLAS::Protected, 218
fsyrk_strassen
FFLAS, 124, 142
fsytrf
FFPACK, 311, 312
fsytrf.C, 773
CUBE, 774
GFOPS, 774
main, 774
fsytrf_BC_Crout
FFPACK, 350
fsytrf_BC_RL
FFPACK, 350
fsytrf_LOW_RPM_BC_Crout
FFPACK, 351
fsytrf_nonunit
FFPACK, 312, 352
fsytrf_RPM
FFPACK, 352
fsytrf_UP_RPM
FFPACK, 351
fsytrf_UP_RPM_BC_Crout
FFPACK, 351
fsytrf_UP_RPM_BC_RL
FFPACK, 351
ftranspose
FFLAS, 154
ftrmm
FFLAS, 124, 125, 175
ftrmm_3_modular_double
fflas_c.h, 959
fflas_lvl3.C, 980
ftrmmLeftLowerNoTransNonUnit< Element >, 470
ftrmmLeftLowerNoTransUnit< Element >, 470
ftrmmLeftLowerTransNonUnit< Element >, 470
ftrmmLeftLowerTransUnit< Element >, 470
ftrmmLeftUpperNoTransNonUnit< Element >, 470
ftrmmLeftUpperNoTransUnit< Element >, 470
ftrmmLeftUpperTransNonUnit< Element >, 470
ftrmmLeftUpperTransUnit< Element >, 471
ftrmmRightLowerNoTransNonUnit< Element >, 471
ftrmmRightLowerNoTransUnit< Element >, 471
ftrmmRightLowerTransNonUnit< Element >, 471
ftrmmRightLowerTransUnit< Element >, 471
ftrmmRightUpperNoTransNonUnit< Element >, 471
ftrmmRightUpperNoTransUnit< Element >, 471
ftrmmRightUpperTransNonUnit< Element >, 471
ftrmmRightUpperTransUnit< Element >, 472
ftrmv
FFLAS, 140
ftrsm
FFLAS, 126, 127, 141, 144, 145, 175
ftrsm_3_modular_double
fflas_c.h, 959
fflas_lvl3.C, 980
ftrsmLeftLowerNoTransNonUnit< Element >, 472
ftrsmLeftLowerNoTransUnit< Element >, 472
ftrsmLeftLowerTransNonUnit< Element >, 472
ftrsmLeftLowerTransUnit< Element >, 472
ftrsmLeftUpperNoTransNonUnit< Element >, 472
ftrsmLeftUpperNoTransUnit< Element >, 473
ftrsmLeftUpperTransNonUnit< Element >, 473
ftrsmLeftUpperTransUnit< Element >, 473
ftrsmRightLowerNoTransNonUnit< Element >, 473
ftrsmRightLowerNoTransUnit< Element >, 473
ftrsmRightLowerTransNonUnit< Element >, 473
ftrsmRightLowerTransUnit< Element >, 473
ftrsmRightUpperNoTransNonUnit< Element >, 473

ftrsmRightUpperNoTransUnit< Element >, 474
ftrsmRightUpperTransNonUnit< Element >, 474
ftrsmRightUpperTransUnit< Element >, 474
ftrssyr2k
FFPACK, 310
ftrstr
FFPACK, 310
ftrsv
FFLAS, 128, 174
ftrsv_2_modular_double
fflas_c.h, 959
fflas_lvl2.C, 979
ftrtri
FFPACK, 309, 364
ftrtri.C, 774
CUBE, 774
GFOPS, 774
main, 775
ftrtri_modular_double
ffpack.C, 988
ffpack_c.h, 1010
ftrtrm
FFPACK, 309, 365
ftrtrm_modular_double
ffpack.C, 989
ffpack_c.h, 1010
fzero
FFLAS, 130, 133, 135, 159, 165
fzero_1_modular_double
fflas_c.h, 952
fflas_lvl1.C, 972
fzero_2_modular_double
fflas_c.h, 955
fflas_lvl2.C, 975

gather
Simd128_impl< true, true, false, 2 >, 564, 566
Simd128_impl< true, true, false, 4 >, 574, 576
Simd128_impl< true, true, false, 8 >, 584, 586
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 634
Simd256_impl< true, true, false, 4 >, 643, 645, 648
Simd256_impl< true, true, false, 8 >, 659, 662
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 712, 715
Simd512_impl< true, true, true, 8 >, 723
GaussJordan
FFPACK::Protected, 391
GCC_VERSION
fflas-ffpack-config.h, 822
gebp
FFLAS::details, 205

genData
benchmark-fgemv.C, 790
GenericTag, 474
genInputs
ScalFunctions< Element >, 551
genInputsWithZero
ScalFunctions< Element >, 551
get
Simd128_impl< true, true, false, 8 >, 587
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, true, false, 8 >, 662
Simd256_impl< true, true, true, 8 >, 696
get_default_random_generator
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
getArgumentValue
FFLAS, 185
getDataType
FFLAS, 151, 152
getEchelonForm
FFPACK, 336
getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >
FFPACK, 373
getEchelonForm_modular_double
ffpack.C, 1000
ffpack_c.h, 1020
getEchelonFormin_modular_double
ffpack.C, 1000
ffpack_c.h, 1020
getEchelonTransform
FFPACK, 337
getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >
FFPACK, 373
getEchelonTransform_modular_double
ffpack.C, 1001
ffpack_c.h, 1020
getListArgs
args-parser.h, 1039
getLTBruhatGen
FFPACK, 341
getReducedEchelonForm
FFPACK, 338, 339
getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >
FFPACK, 374
getReducedEchelonForm_modular_double
ffpack.C, 1001
ffpack_c.h, 1021
getReducedEchelonFormin_modular_double
ffpack.C, 1001
ffpack_c.h, 1021
getReducedEchelonTransform
FFPACK, 339

getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >
FFPACK, 374
getReducedEchelonTransform_modular_double
ffpack.C, 1001
ffpack_c.h, 1021
getSeed
FFLAS, 189
getStat
FFLAS, 154
getStatus
TestOneMethod< Simd >, 762
getTestName
TestOneMethod< Simd >, 762
getTLBSize
FFLAS, 189
getTriangular
FFPACK, 335
getTriangular< FFLAS_FIELD< FFLAS_ELT > >
FFPACK, 372, 373
getTriangular_modular_double
ffpack.C, 1000
ffpack_c.h, 1019
getTriangularin_modular_double
ffpack.C, 1000
ffpack_c.h, 1019
getTridiagonal
FFPACK, 352
gf2ModularBalanced
regression-check.C, 1051
GFOPS
charpoly.C, 772
fsyrk.C, 773
fsytrf.C, 774
ftrtri.C, 774
pluq.C, 775
winograd.C, 776
Givaro, 396
GRAIN
benchmark-fgemm-rns.C, 786
Grain, 474
greater
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 564
Simd128_impl< true, true, false, 4 >, 574
Simd128_impl< true, true, false, 8 >, 584
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 646
Simd256_impl< true, true, false, 8 >, 660
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 713
Simd512_impl< true, true, true, 8 >, 728
greater_eq
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 646
Simd256_impl< true, true, false, 8 >, 660
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 709
Simd512_impl< true, true, false, 8 >, 713
Simd512_impl< true, true, true, 8 >, 728
hadd
Simd256_impl< true, false, true, 8 >, 627
Simd512_impl< true, false, true, 8 >, 709
hadd_to_scal
Simd128_impl< true, true, false, 2 >, 566
Simd128_impl< true, true, false, 4 >, 576
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 628
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 645, 647
Simd256_impl< true, true, false, 8 >, 662
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, false, true, 8 >, 709
Simd512_impl< true, true, false, 8 >, 715
Simd512_impl< true, true, true, 8 >, 728
half_t
Simd256_impl< true, true, false, 2 >, 631
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 668
Simd256_impl< true, true, true, 4 >, 679
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 722
has_equal
FFLAS, 73
has_minus
FFLAS, 73
has_minus_eq
FFLAS, 73
has_minus_eq_impl< C >, 474
value, 475
has_minus_impl< C >, 475
value, 475

has_mul
FFLAS, 73
has_mul_eq
FFLAS, 73
has_mul_eq_impl< C >, 475
value, 475
has_mul_impl< C >, 475
value, 476
has_operation< T >, 476
value, 476
has_plus
FFLAS, 73
has_plus_eq
FFLAS, 73
has_plus_eq_impl< C >, 476
value, 476
has_plus_impl< C >, 476
value, 477
HAVE_BLAS
config.h, 815
HAVE_CBLAS
config.h, 815
HAVE_CXX11
config.h, 815
HAVE_DLFCN_H
config.h, 816
HAVE_FLOAT_H
config.h, 816
HAVE_INT128
config.h, 816
HAVE_INTTYPES_H
config.h, 816
HAVE_LAPACK
config.h, 816
HAVE_LIMITS_H
config.h, 816
HAVE_LITTLE_ENDIAN
config.h, 816
HAVE_PTHREAD_H
config.h, 816
HAVE_STDDEF_H
config.h, 816
HAVE_STDINT_H
config.h, 816
HAVE_STDIO_H
config.h, 816
HAVE_STDLIB_H
config.h, 816
HAVE_STRING_H
config.h, 816
HAVE_STRINGS_H
config.h, 816
HAVE_SYS_STAT_H
config.h, 816
HAVE_SYS_TIME_H
config.h, 817
HAVE_SYS_TYPES_H
config.h, 817
HAVE_UNISTD_H
config.h, 817
HelperFlag, 477
aut, 477
coo, 477
csr, 477
ell, 477
none, 477
pm1, 477
HelperMod
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, 479
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, 479
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 480
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod, 478
invp, 478
max, 478
min, 478
p, 478
pow50rem, 478
HelperMod< Field, ElementTraits >, 477
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, 479
HelperMod, 479
p, 479
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, 479
HelperMod, 479
p, 480
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 480
HelperMod, 480
invp, 480
max, 480
min, 480
p, 480
helpString
Argument, 400
HYB_ZO
FFLAS, 76
hyb_zo.h, 903
hyb_zo_pspmm.inl, 903
__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL, 903
hyb_zo_pspmv.inl, 903
__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL, 904
hyb_zo_spmm.inl, 904
__FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL, 904
hyb_zo_spmv.inl, 904
__FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL,

905
hyb_zo_utils.inl, 905
__FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL, 905
Hybrid, 481
Hybrid_KGF_LUK_MinPoly
FFPACK::Protected, 394

ibeg
ForStrategy1D< blocksize_t, Cut, Param >, 465
ibegin
ForStrategy2D< blocksize_t, Cut, Param >, 467
idamax_
config-blas.h, 811
iend
ForStrategy1D< blocksize_t, Cut, Param >, 465
ForStrategy2D< blocksize_t, Cut, Param >, 467
igebb11
FFLAS::details, 204
igebb14
FFLAS::details, 204
igebb21
FFLAS::details, 204
igebb24
FFLAS::details, 203
igebb41
FFLAS::details, 204
igebb44
FFLAS::details, 203
igebp
FFLAS::details, 205
igemm
FFLAS::Protected, 221
igemm.doxy, 857
igemm.h, 857
igemm.inl, 857
__FFLASFFPACK_fflas_igemm_igemm_INL, 858
igemm_
FFLAS, 128
igemm_colmajor
FFLAS::Protected, 221
igemm_kernels.h, 858
igemm_kernels.inl, 858
__FFLASFFPACK_fflas_igemm_igemm_kernels_INL, 859
igemm_tools.h, 859
igemm_tools.inl, 859
__FFLASFFPACK_fflas_igemm_igemm_tools_INL, 860
in_range
flimits.h, 1046
index_t
fflas_sparse.h, 883
parallel.h, 1030
InfNorm
FFLAS, 76
Info, 481, 482
begin, 482, 483
Info, 481, 482
operator=, 481, 483
perm, 482, 483
size, 482, 483
info
BlockTransposeSIMD< Field, Simd, >, 407
init
FieldSimd< _Field >, 448, 449
rns_double, 525, 526
rns_double_extended, 536, 537
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 545, 546
init_transpose
rns_double, 526
init_y
FFLAS::sparse_details, 226
initA
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 503
initB
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 503
initC
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 503
initialize
ForStrategy1D< blocksize_t, Cut, Param >, 464
ForStrategy2D< blocksize_t, Cut, Param >, 467
initOut
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 503
INLINE
fflas_simd.h, 870
inplace
Bench< Elt >, 406
inputs
TestOneMethod< Simd >, 762
INST_OR_DECL
fflas_L1_inst.C, 960
fflas_L1_inst.h, 961
fflas_L2_inst.C, 964
fflas_L2_inst.h, 965
fflas_L3_inst.C, 967
fflas_L3_inst.h, 969
ffpack_inst.C, 1022
ffpack_inst.h, 1023
integer
rns_double, 524
rns_double_extended, 535
RNSInteger< RNS >, 539
RNSIntegerMod< RNS >, 544
Interfaces, 42
interfaces.doxy, 948
IntType
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution, 463
inv
RNSIntegerMod< RNS >, 547

Invert
FFPACK, 320, 367
Invert2
FFPACK, 321, 367
Invert2_modular_double
ffpack.C, 996
ffpack_c.h, 1015
Invert_modular_double
ffpack.C, 995
ffpack_c.h, 1015
Invertin_modular_double
ffpack.C, 995
ffpack_c.h, 1015
invp
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod< Field, FFLAS::ElementCategories::MachineIntTag >, 480
is_all_same< Args >, 483
is_all_same< T, Args... >, 483
value, 483
is_all_same<>, 483
value, 484
is_same_element
Bench< Elt >, 404
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 573
Simd128_impl< true, true, false, 8 >, 583
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 630
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 679, 680
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 723
Test< Elt >, 758
is_simd< T >, 484
type, 484
value, 484
isMOne
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 544
isOdd
FFPACK, 375
isOne
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 544
IsSingular
FFPACK, 325, 369
IsSingular_modular_double
ffpack.C, 996
ffpack_c.h, 1016
isSparseMatrix< Field, M >, 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >, 484
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >, 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >, 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >, 485
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >, 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >, 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >, 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >, 486
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >, 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >, 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >, 487
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >, 487
isSparseMatrixMKLFormat< F, M >, 488
isSparseMatrixSimdFormat< F, M >, 488
isTerminated
ForStrategy1D< blocksize_t, Cut, Param >, 464
ForStrategy2D< blocksize_t, Cut, Param >, 467
isZero
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 544
isZOSparseMatrix< F, M >, 488
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >, 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >, 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >, 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >, 489
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >, 490
Iterative, 490
iters
Bench< Elt >, 406

jbegin
ForStrategy2D< blocksize_t, Cut, Param >, 467
jend
ForStrategy2D< blocksize_t, Cut, Param >, 467

kaapi_routines.inl, 1029
__FFLASFFPACK_KAAPI_ROUTINES_INL, 1029
KellerGehrig
FFPACK::Protected, 391
KGFast

FFPACK::Protected, 392
KGFast_generalized
FFPACK::Protected, 392
kmax
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 741
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
KrylovElim
FFPACK, 369
KrylovElim_modular_double
ffpack.C, 996
ffpack_c.h, 1015

lapack.C, 1050
__FFLASFFPACK_CONFIGURATION, 1050
__FFLASFFPACK_HAVE_LAPACK, 1050
main, 1050
LAPACKPerm2MathPerm
FFPACK, 303
ffpack.C, 985
ffpack_c.h, 1007
lastBlockSize
ForStrategy1D< blocksize_t, Cut, Param >, 465
lastCBS
ForStrategy2D< blocksize_t, Cut, Param >, 469
lastRBS
ForStrategy2D< blocksize_t, Cut, Param >, 469
launch_fger
test-fger.C, 1066
launch_fger_dispatch
test-fger.C, 1066
launch_MM
test-fgemm.C, 1062
launch_MM_dispatch
test-fgemm-check.C, 1060
test-fgemm.C, 1062
launch_MV
test-fgemv.C, 1064
launch_MV_dispatch
test-fgemv.C, 1064
launch_test
test-charpoly.C, 1052
test-lu.C, 1089
test-quasisep.C, 1095
launch_wino
benchmark-wino.C, 801
LazyTag, 490
LD
fflas_transpose.h, 912
ld
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
LeadingSubmatrixRankProfiles
FFPACK, 331
ffpack.C, 998
ffpack_c.h, 1018
lesser
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 564
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 646
Simd256_impl< true, true, false, 8 >, 660
Simd256_impl< true, true, true, 2 >, 674
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 713
Simd512_impl< true, true, true, 8 >, 728
lesser_eq
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 626
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 646
Simd256_impl< true, true, false, 8 >, 660
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 685, 691
Simd256_impl< true, true, true, 8 >, 700
Simd512_impl< true, false, true, 8 >, 708
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 728
limits< char >, 491
digits, 491
max, 491
min, 491
T, 491
limits< double >, 491
digits, 492
max, 492
min, 492
T, 492
limits< float >, 492
digits, 492

max, 492
min, 492
T, 492
limits< Givaro::Integer >, 493
max, 493
min, 493
T, 493
limits< int >, 493
digits, 494
max, 494
min, 494
T, 493
limits< long >, 494
digits, 494
max, 494
min, 494
T, 494
limits< long long >, 494
digits, 495
max, 495
min, 495
T, 495
limits< RecInt::rint< K > >, 495
max, 495
min, 495
T, 495
limits< RecInt::ruint< K > >, 496
max, 496
min, 496
T, 496
limits< short int >, 496
digits, 497
max, 497
min, 497
T, 497
limits< signed char >, 497
digits, 497
max, 497
min, 497
T, 497
limits< T >, 491
limits< unsigned char >, 498
digits, 498
max, 498
min, 498
T, 498
limits< unsigned int >, 498
digits, 499
max, 499
min, 499
T, 498
limits< unsigned long >, 499
digits, 499
max, 499
min, 499
T, 499
limits< unsigned long long >, 499
digits, 500
max, 500
min, 500
T, 500
limits< unsigned short int >, 500
digits, 500
max, 500
min, 500
T, 500
load
Simd128_impl< true, true, false, 2 >, 564, 566
Simd128_impl< true, true, false, 4 >, 574, 576
Simd128_impl< true, true, false, 8 >, 584, 587
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 634
Simd256_impl< true, true, false, 4 >, 643, 645, 649
Simd256_impl< true, true, false, 8 >, 660, 662
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
loadu
Simd128_impl< true, true, false, 2 >, 564, 566
Simd128_impl< true, true, false, 4 >, 574, 577
Simd128_impl< true, true, false, 8 >, 584, 587
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 634
Simd256_impl< true, true, false, 4 >, 643, 646, 649
Simd256_impl< true, true, false, 8 >, 660, 662
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 680, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
LQUPtoInverseOfFullRankMinor
FFPACK, 345, 375
LT_OBJDIR
config.h, 817
LTBruhatGen
FFPACK, 340
LTQSorder
FFPACK, 342
LUdivine
FFPACK, 314, 353, 354, 365
LUdivine_construct
FFPACK::Protected, 390, 396
LUdivine_gauss
FFPACK, 353, 366

LUdivine_gauss_modular_double
ffpack_c.h, 1011
LUdivine_modular_double
ffpack.C, 989
ffpack_c.h, 1011
LUdivine_small
FFPACK, 353, 365
LUdivine_small_modular_double
ffpack_c.h, 1011
LUKrylov
FFPACK::Protected, 392
LUKrylov_KGFast
FFPACK::Protected, 393

m
Bench< Elt >, 406
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
MachineFloatTag, 501
MachineIntTag, 501
main
101-fgemm.C, 1103
2x2-fgemm.C, 1103
2x2-ftrsv.C, 1104
2x2-pluq.C, 1104
arithprog.C, 771
benchmark-charpoly-mp.C, 777
benchmark-charpoly.C, 778
benchmark-checkers.C, 779
benchmark-dgemm.C, 780
benchmark-dgetrf.C, 781
benchmark-dgetri.C, 781
benchmark-dsytrf.C, 782
benchmark-dtrsm.C, 782
benchmark-dtrtri.C, 783
benchmark-fadd-lvl2.C, 783
benchmark-fdot.C, 784
benchmark-fgemm-mp.C, 785
benchmark-fgemm-rns.C, 787
benchmark-fgemm.C, 787
benchmark-fgemv-mp.C, 788
benchmark-fgemv.C, 792
benchmark-fgesv.C, 792
benchmark-fsyr2k.C, 793
benchmark-fsyrk.C, 793
benchmark-fsytrf.C, 794
benchmark-ftrsm-mp.C, 795
benchmark-ftrsm.C, 795
benchmark-ftrsv.C, 796
benchmark-ftrtri.C, 796
benchmark-inverse.C, 797
benchmark-lqup-mp.C, 797
benchmark-lqup.C, 798
benchmark-pluq.C, 799
benchmark-quasisep.C, 800
benchmark-storage-transpose.C, 801
benchmark-wino.C, 802
cblas.C, 1048
charpoly.C, 772
clapack.C, 1049
cuda.C, 1049
det.C, 802
fblas.C, 1050
fflas-101_1.C, 1104
fflas-101_3.C, 1105
fflas_101.C, 1105
fflas_101_lvl1.C, 1105
ffpack-fgesv.C, 1106
ffpack-solve.C, 1106
fsyrk.C, 773
fsytrf.C, 774
ftrtri.C, 775
lapack.C, 1050
matmul.C, 802
pluq.C, 776
rank.C, 803
regression-check.C, 1051
solve.C, 803
test-charpoly-check.C, 1052
test-charpoly.C, 1053
test-compressQ.C, 1054
test-det-check.C, 1054
test-det.C, 1055
test-echelon.C, 1057
test-fadd.C, 1058
test-fdot.C, 1060
test-fgemm-check.C, 1061
test-fgemm.C, 1063
test-fgemv.C, 1065
test-fger.C, 1067
test-fgesv.C, 1068
test-finit.C, 1069
test-fscal.C, 1070
test-fsyr2k.C, 1072
test-fsyrk.C, 1074
test-fsytrf.C, 1075
test-ftrmm.C, 1076
test-ftrmv.C, 1078
test-ftrsm-check.C, 1078
test-ftrsm.C, 1079
test-ftrssyr2k.C, 1081
test-ftrstr.C, 1082
test-ftrsv.C, 1083
test-ftrtri.C, 1084
test-interfaces-c.c, 1084
test-invert-check.C, 1085

test-io.C, 1086
test-lu.C, 1089
test-maxdelayeddim.C, 1090
test-minpoly.C, 1091
test-multifile2.C, 1092
test-nullspace.C, 1093
test-permutations.C, 1094
test-pluq-check.C, 1095
test-quasisep.C, 1096
test-rankprofiles.C, 1097
test-rpm.C, 1098
test-simd.C, 1101
test-solve.C, 1102
test-storage-transpose.C, 1102
winograd.C, 777
mainpage.doxy, 802
mask_high
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 729
mask_t
read_sparse.h, 906
maskstore
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
MatF2MatD_Triangular
FFLAS::Protected, 222
MatF2MatFl_Triangular
FFLAS::Protected, 222
MathPerm2LAPACKPerm
FFPACK, 303
ffpack.C, 985
ffpack_c.h, 1007
Matio.h, 1046
read_field, 1046
write_field, 1046
matmul.C, 802
main, 802
matmul.doxy, 837
Matrix Multiplication Algorithms, 42
MatrixApplyS
FFPACK, 355, 356
MatrixApplyS_modular_double
ffpack.C, 985
ffpack_c.h, 1007
MatrixApplyT
FFPACK, 357
MatrixApplyT_modular_double
ffpack.C, 986
ffpack_c.h, 1007
MatVecMinPoly
FFPACK, 324, 368
FFPACK::Protected, 393
max
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 480
limits< char >, 491
limits< double >, 492
limits< float >, 492
limits< Givaro::Integer >, 493
limits< int >, 494
limits< long >, 494
limits< long long >, 495
limits< RecInt::rint< K > >, 495
limits< RecInt::ruint< K > >, 496
limits< short int >, 497
limits< signed char >, 497
limits< unsigned char >, 498
limits< unsigned int >, 499
limits< unsigned long >, 499
limits< unsigned long long >, 500
limits< unsigned short int >, 500
max3
FFLAS, 77
max4
FFLAS, 77
MAX_THREADS
parallel.h, 1031
MAX_WITH_SIZE_T
test-maxdelayeddim.C, 1090
maxCardinality
FFLAS, 154
maxCardinality< Givaro::Modular< int32_t > >
FFLAS, 154
maxCardinality< Givaro::Modular< int64_t > >
FFLAS, 154
maxCol
StatsMatrix, 753
maxColDifference
StatsMatrix, 754
MaxDelayedDim
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 503
maxElement
RNSIntegerMod< RNS >, 545
maxpy
FieldSimd< _Field >, 452
maxpyin
FieldSimd< _Field >, 452
maxRow
StatsMatrix, 753
maxrow
Sparse< _Field, SparseMatrix_t::COO >, 734
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743

Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
maxRowDifference
StatsMatrix, 754
MaxStorableValue
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 506
MG_DEFAULT
benchmark-fgemm-mp.C, 785
benchmark-fgemv-mp.C, 788
min
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 480
limits< char >, 491
limits< double >, 492
limits< float >, 492
limits< Givaro::Integer >, 493
limits< int >, 494
limits< long >, 494
limits< long long >, 495
limits< RecInt::rint< K > >, 495
limits< RecInt::ruint< K > >, 496
limits< short int >, 497
limits< signed char >, 497
limits< unsigned char >, 498
limits< unsigned int >, 499
limits< unsigned long >, 499
limits< unsigned long long >, 500
limits< unsigned short int >, 500
min3
FFLAS, 76
min4
FFLAS, 77
min_types
FFLAS::Protected, 219, 220
minCardinality
FFLAS, 154
minCol
StatsMatrix, 753
minColDifference
StatsMatrix, 754
minElement
RNSIntegerMod< RNS >, 545
MinPoly
FFPACK, 323, 368
minRow
StatsMatrix, 753
minRowDifference
StatsMatrix, 754
MKL_CONFIG, 396
MKLSparseMatrixFormat
FFLAS, 72
MMHelper
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502, 503
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 501
Amax, 505
Amin, 505
Aunfit, 504
Bmax, 505
Bmin, 505
Bunfit, 504
checkA, 504
checkB, 504
checkOut, 504
Cmax, 505
Cmin, 505
DelayedField, 502
DelayedField, 506
DelayedField_t, 502
DFElt, 502
FieldMax, 505
FieldMin, 505
initA, 503
initB, 503
initC, 503
initOut, 503
MaxDelayedDim, 503
MaxStorableValue, 506
MMHelper, 502, 503
operator<<, 505
Outmax, 506
Outmin, 506
parseq, 506
recLevel, 505
Self_t, 502
setOutBounds, 504
mod
FieldSimd< _Field >, 450
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 628
Simd256_impl< true, true, false, 2 >, 638
Simd256_impl< true, true, false, 4 >, 655, 656
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 685, 692
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 729
MODE
parallel.h, 1033
ModeTraits< Field >, 506
value, 506
ModeTraits< Givaro::Modular< Element, Compute > >, 507

value, 507
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, 507
value, 507
ModeTraits< Givaro::Modular< int16_t, Compute > >, 507
value, 508
ModeTraits< Givaro::Modular< int32_t, Compute > >, 508
value, 508
ModeTraits< Givaro::Modular< int64_t, uint64_t > >, 508
value, 508
ModeTraits< Givaro::Modular< int8_t, Compute > >, 509
value, 509
ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >, 509
value, 509
ModeTraits< Givaro::Modular< uint16_t, Compute > >, 509
value, 510
ModeTraits< Givaro::Modular< uint32_t, Compute > >, 510
value, 510
ModeTraits< Givaro::Modular< uint8_t, Compute > >, 510
value, 510
ModeTraits< Givaro::ModularBalanced< Element > >, 511
value, 511
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, 511
value, 511
ModeTraits< Givaro::ModularBalanced< int16_t > >, 511
value, 512
ModeTraits< Givaro::ModularBalanced< int32_t > >, 512
value, 512
ModeTraits< Givaro::ModularBalanced< int8_t > >, 512
value, 512
ModeTraits< Givaro::Montgomery< T > >, 513
value, 513
ModeTraits< Givaro::ZRing< double > >, 513
value, 513
ModeTraits< Givaro::ZRing< float > >, 513
value, 513
ModeTraits< Givaro::ZRing< Givaro::Integer > >, 514
value, 514
ModField
rns_double, 524
rns_double_extended, 535
RNSIntegerMod< RNS >, 544
modp
FFLAS::vectorised, 282
FFLAS::vectorised::unswitch, 284, 285
ModularBalanced< T >, 514
ModularTag, 514
mOne
RNSInteger< RNS >, 542
RNSIntegerMod< RNS >, 548
mone
FFLAS, 76
MonotonicApplyP
FFPACK, 305
MonotonicCompress
FFPACK, 354
MonotonicCompressCycles
FFPACK, 354
MonotonicCompressMorePivots
FFPACK, 354
MonotonicExpand
FFPACK, 355
Montgomery< T >, 514
mul
FieldSimd< _Field >, 450, 451
RNSIntegerMod< RNS >, 546
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 653
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
mul_r
FieldSimd< _Field >, 451
mulhi
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 646
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 726
mulhi_fast

Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 729
mulin
FieldSimd< _Field >, 451
ScalFunctions< Element >, 553
Simd256_impl< true, false, true, 8 >, 625
Simd512_impl< true, false, true, 8 >, 707
mullo
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, true, false, 2 >, 637
Simd256_impl< true, true, false, 4 >, 653
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 726
mulx
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 565
Simd128_impl< true, true, false, 4 >, 575
Simd128_impl< true, true, false, 8 >, 585
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 644, 647
Simd256_impl< true, true, false, 8 >, 661
Simd256_impl< true, true, true, 2 >, 673
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 699
Simd512_impl< true, true, false, 8 >, 714
Simd512_impl< true, true, true, 8 >, 726
mvcnt
test-lu.C, 1090

n
Bench< Elt >, 406
count_nonconst_lvalue_reference< const T &, O... >, 435
count_nonconst_lvalue_reference< T &, O... >, 435
count_nonconst_lvalue_reference< T, O... >, 435
count_nonconst_lvalue_reference<>, 436
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 750
name
TestOneMethod< Simd >, 762
nb_lref
TestOneMethod< Simd >, 762
nChunks
Sparse< _Field, SparseMatrix_t::ELL_simd >, 744
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
nDenseCols
StatsMatrix, 754
nDenseRows
StatsMatrix, 754
need_field_characteristic< Field >, 515
value, 515
need_field_characteristic< Givaro::Modular< Field > >, 515
value, 515
need_field_characteristic< Givaro::ModularBalanced< Field > >, 515
value, 515
NeedDoublePreAddReduction
FFLAS::Protected, 214
NeedPreAddReduction
FFLAS::Protected, 213
NeedPreAxpyReduction
FFLAS::Protected, 219
NeedPreScalReduction
FFLAS::Protected, 218, 219
NeedPreSubReduction
FFLAS::Protected, 213
neg
RNSIntegerMod< RNS >, 546
nElements
Sparse< _Field, SparseMatrix_t::COO >, 734
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751

nEmptyCols
StatsMatrix, 754
nEmptyColsEnd
StatsMatrix, 754
nEmptyRows
StatsMatrix, 754
newD
FFPACK::Protected, 394
NEWWINO
fgemm_winograd.inl, 836
nMOnes
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 739
StatsMatrix, 753
nnz
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::ELL >, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 747
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
StatsMatrix, 753
none
HelperFlag, 477
nOnes
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 739
StatsMatrix, 753
nonsquare_inplace_v1
FFLAS::_ftranspose_impl, 190
nonsquare_inplace_v2
FFLAS::_ftranspose_impl, 190
NonZeroRandomMatrix
FFPACK, 375, 376
NORML_MOD
fflas_simd.h, 870
NoSimd< T >, 516
aligned_allocator, 516
aligned_vector, 516
alignment, 517
compliant, 517
is_same_element, 516
scalar_t, 516
type_string, 516
valid, 516
vect_size, 517
vect_t, 516
NoSimdSparseMatrix
FFLAS, 72
NOSPLIT
parallel.h, 1035
not_inplace
FFLAS::_ftranspose_impl, 190
nOthers
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 739
StatsMatrix, 753
NotMKLSparseMatrixFormat
FFLAS, 72
NotZOSparseMatrix
FFLAS, 72
NullSpaceBasis
FFPACK, 328, 371
NullSpaceBasis_modular_double
ffpack.C, 998
ffpack_c.h, 1017
NUM_THREADS
parallel.h, 1031
NUMARGS
parallel.h, 1034
number_kind
FFLAS, 75
numBlock
ForStrategy1D< blocksize_t, Cut, Param >, 466
numblocks
ForStrategy1D< blocksize_t, Cut, Param >, 465
numColBlock
ForStrategy2D< blocksize_t, Cut, Param >, 469
numRowBlock
ForStrategy2D< blocksize_t, Cut, Param >, 469
numthreads
Parallel< C, P >, 518
Sequential, 560
one
FFLAS, 76
RNSInteger< RNS >, 542
RNSIntegerMod< RNS >, 548
OPENBLAS_NUM_THREADS
config.h, 817
operator!=
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator<
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator<<
Compose< H1, H2 >, 423
FFLAS, 151
ForStrategy2D< blocksize_t, Cut, Param >, 468
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
Parallel< C, P >, 518
Sequential, 560
test-fsytrf.C, 1075
test-simd.C, 1101
operator()
callLUdivine_small< double >, 409
callLUdivine_small< Element >, 408
callLUdivine_small< float >, 410

Failure, 445
readMyMachineType< Field, mpz_t >, 522
readMyMachineType< Field, T >, 521
RNSInteger< RNS >::RandIter, 519
RNSIntegerMod< RNS >::RandIter, 520
rnsRandIter< RNS >, 549, 550
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution, 463
tfn_minus, 763
tfn_minus_eq, 763
tfn_mul, 764
tfn_mul_eq, 764
tfn_plus, 764
tfn_plus_eq, 765
operator+
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 533
operator++
ForStrategy1D< blocksize_t, Cut, Param >, 465
ForStrategy2D< blocksize_t, Cut, Param >, 467
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 533
operator+=
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator-
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator--
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 533
operator-=
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator=
Coo< Field >, 432
Coo< ValT, IdxT >, 430, 433
FieldSimd< _Field >, 448
Info, 481, 483
rns_double_elt_cstptr, 531
rns_double_elt_ptr, 534
operator&
rns_double_elt, 529
rns_double_elt_cstptr, 530, 531
rns_double_elt_ptr, 533, 534
operator[]
rns_double_elt_cstptr, 530, 531
rns_double_elt_ptr, 533
operator*
rns_double_elt_cstptr, 530
rns_double_elt_ptr, 533
other
FFLAS, 76
rns_double_elt_cstptr, 532
rns_double_elt_ptr, 534
Outmax
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 506
Outmin
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 506
outputs_scalar
TestOneMethod< Simd >, 763
outputs_simd
TestOneMethod< Simd >, 762

p
HelperMod< Field, ElementCategories::MachineIntTag >, 478
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, 479
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, 480
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 480
pack
ScalFunctions< Element >, 555
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 651
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
pack_even
ScalFunctions< Element >, 555
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 651
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
pack_lhs
FFLAS::details, 205
pack_odd
ScalFunctions< Element >, 555

Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 651
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
pack_rhs
FFLAS::details, 205
PACKAGE
config.h, 817
PACKAGE_BUGREPORT
config.h, 817
PACKAGE_NAME
config.h, 817
PACKAGE_STRING
config.h, 817
PACKAGE_TARNAME
config.h, 817
PACKAGE_URL
config.h, 817
PACKAGE_VERSION
config.h, 817
PAR_BLOCK
parallel.h, 1031
Parallel
Parallel< C, P >, 518
Parallel< C, P >, 517
Cut, 517
numthreads, 518
operator<<, 518
Parallel, 518
Param, 517
set_numthreads, 518
parallel.h, 1029
__FFLASFFPACK_SEQUENTIAL, 1030
BARRIER, 1030
BEGIN_PARALLEL_MAIN, 1031
CHECK_DEPENDENCIES, 1030
COMMA, 1033
CONSTREFERENCE, 1031
END_PARALLEL_MAIN, 1032
FOR1D, 1032
FOR2D, 1033
FORBLOCK1D, 1032
FORBLOCK2D, 1032
index_t, 1030
MAX_THREADS, 1031
MODE, 1033
NOSPLIT, 1035
NUM_THREADS, 1031
NUMARGS, 1034
PAR_BLOCK, 1031
PARFOR1D, 1032
PARFOR2D, 1033
PARFORBLOCK1D, 1032
PARFORBLOCK2D, 1033
PP_ARG_N, 1034
PP_NARG_, 1034
PP_RSEQ_N, 1035
READ, 1031
READWRITE, 1031
RETURNPARAM, 1033
SET_THREADS, 1031
splitt, 1036
SPLITTER, 1036
splitting_0, 1035
splitting_1, 1035
splitting_2, 1036
splitting_3, 1036
SYNCH_GROUP, 1031
TASK, 1030
THREAD_INDEX, 1031
VALUE, 1031
WAIT, 1030
WRITE, 1031
Param
Parallel< C, P >, 517
PARFOR1D
parallel.h, 1032
PARFOR2D
parallel.h, 1033
PARFORBLOCK1D
parallel.h, 1032
PARFORBLOCK2D
parallel.h, 1033
parseArguments
FFLAS, 185
parseq
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 506
TRSMHelper< RecIterTrait, ParSeqTrait >, 767
PBASECASE_K
ffpack_ppluq.inl, 940
pColumnEchelonForm
FFPACK, 316
pColumnEchelonForm_modular_double
ffpack.C, 992
pColumnEchelonForm_modular_float
ffpack.C, 993
pColumnEchelonForm_modular_int32_t
ffpack.C, 994
pColumnRankProfile
FFPACK, 331
pDet
FFPACK, 326
perm

Info, 482, 483
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
PermApplyS
FFPACK, 356
PermApplyS_double
ffpack.C, 986
ffpack_c.h, 1007
PermApplyT
FFPACK, 358
PermApplyT_double
ffpack.C, 986
ffpack_c.h, 1008
pfadd
FFLAS, 78
pfaddin
FFLAS, 79
pfgemm
FFLAS, 142, 182–184
pfgemm_1D_rec
FFLAS, 142
pfgemm_2D_rec
FFLAS, 143
pfgemm_3D_rec
FFLAS, 143
pfgemm_3D_rec2
FFLAS, 144
pfgemm_variants.inl, 1036
pfgemv.inl, 1037
pfrand
FFLAS, 181
pfreduce
FFLAS, 109
pfspmm
FFLAS::sparse_details, 232–234
FFLAS::sparse_details_impl, 249, 256, 257, 259–261, 271
pfspmm_dispatch
FFLAS::sparse_details, 231, 232
pfspmm_mone
FFLAS::sparse_details_impl, 250
pfspmm_one
FFLAS::sparse_details_impl, 250
pfspmm_zo
FFLAS::sparse_details_impl, 261
pfspmv
FFLAS::sparse_details, 234, 235
FFLAS::sparse_details_impl, 251, 258, 261, 262, 267, 271–274
pfspmv_mone
FFLAS::sparse_details_impl, 252, 262, 263, 268, 274
pfspmv_one
FFLAS::sparse_details_impl, 251, 252, 262, 268, 274
pfspmv_task
FFLAS::sparse_details_impl, 251
pfsub
FFLAS, 79
pfsubin
FFLAS, 79
pfzero
FFLAS, 181
PLUQ
FFPACK, 313, 314, 361, 362, 365
pluq.C, 775, 776
CUBE, 775
GFOPS, 775
main, 776
PLUQ_basecaseCrout
FFPACK, 360
PLUQ_basecaseV2
FFPACK, 360
PLUQ_basecaseV3
FFPACK, 360
PLUQ_modular_double
ffpack.C, 989
ffpack_c.h, 1011
PLUQtoEchelonPermutation
FFPACK, 340
ffpack.C, 1002
ffpack_c.h, 1021
pm1
HelperFlag, 477
pMMH
TRSMHelper< RecIterTrait, ParSeqTrait >, 766
pow50rem
HelperMod< Field, ElementCategories::MachineIntTag >, 478
PP_ARG_N
parallel.h, 1034
PP_NARG_
parallel.h, 1034
PP_RSEQ_N
parallel.h, 1035
pPLUQ
FFPACK, 313
pRank
FFPACK, 325
preamble
FFLAS, 186
precompute_cst
rns_double, 525
rns_double_extended, 536
pReducedColumnEchelonForm
FFPACK, 318
pReducedColumnEchelonForm_modular_double
ffpack.C, 993
pReducedColumnEchelonForm_modular_float
ffpack.C, 994
pReducedColumnEchelonForm_modular_int32_t
ffpack.C, 995
pReducedRowEchelonForm
FFPACK, 319
pReducedRowEchelonForm_modular_double
ffpack.C, 993

pReducedRowEchelonForm_modular_float
ffpack.C, 994
pReducedRowEchelonForm_modular_int32_t
ffpack.C, 995
prefetch
FFLAS, 189
print
Failure, 445
printHelpMessage
args-parser.h, 1039
printPolynomial
test-charpoly-check.C, 1052
printvect
test-compressQ.C, 1053
productBruhatxTS
FFPACK, 344, 347
pRowEchelonForm
FFPACK, 317
pRowEchelonForm_modular_double
ffpack.C, 993
pRowEchelonForm_modular_float
ffpack.C, 994
pRowEchelonForm_modular_int32_t
ffpack.C, 994
pRowRankProfile
FFPACK, 330
PSeq
benchmark-fgemm-rns.C, 787
pSolve
FFPACK, 328
PTRSM_HYBRID_THRESHOLD
fflas_pftrsm.inl, 869
PURE
fflas_simd.h, 870

queryCacheSizes
FFLAS, 189
queryL1CacheSize
FFLAS, 189
queryTopLevelCacheSize
FFLAS, 189

RandInt
FFPACK, 379
RandIter
RNSInteger< RNS >::RandIter, 519
RNSIntegerMod< RNS >::RandIter, 520
random
RNSInteger< RNS >::RandIter, 519
RNSIntegerMod< RNS >::RandIter, 520
rnsRandIter< RNS >, 549, 550
RandomIndexSubset
FFPACK, 380
RandomKrylovPrecond
FFPACK::Protected, 393
RandomLTQSMatrixWithRankandQSorder
FFPACK, 388
RandomLTQSRankProfileMatrix
FFPACK, 382
RandomMatrix
FFPACK, 376, 377
test-fscal.C, 1070
RandomMatrixWithDet
FFPACK, 387
RandomMatrixWithRank
FFPACK, 379, 380
RandomMatrixWithRankandRandomRPM
FFPACK, 385
RandomMatrixWithRankandRPM
FFPACK, 382, 383
RandomNullSpaceVector
FFPACK, 328, 345, 371
RandomNullSpaceVector_modular_double
ffpack.C, 997
ffpack_c.h, 1017
RandomPermutation
FFPACK, 381
RandomRankProfileMatrix
FFPACK, 381
RandomSymmetricMatrix
FFPACK, 379
RandomSymmetricMatrixWithRankandRandomRPM
FFPACK, 386
RandomSymmetricMatrixWithRankandRPM
FFPACK, 383, 384
RandomSymmetricRankProfileMatrix
FFPACK, 382
RandomTriangularMatrix
FFPACK, 377, 378
Rank
FFPACK, 324, 325, 369
rank.C, 803
main, 803
Rank_modular_double
ffpack.C, 996
ffpack_c.h, 1016
RankProfileFromLU
FFPACK, 331
ffpack.C, 998
ffpack_c.h, 1018
READ
parallel.h, 1031
read_field
Matio.h, 1046
read_sparse.h, 905
DNS_BIN_VER, 906
mask_t, 906
readDnsFormat
FFLAS, 152
readMachineType
FFLAS, 152
ReadMatrix
FFLAS, 186
readMyMachineType< Field, mpz_t >, 521
Element, 521, 522
Element_ptr, 521, 522
operator(), 522

readMyMachineType< Field, T >, 520
Element, 521
Element_ptr, 521
operator(), 521
readOrRandomMatrixWithRankAndRandomRPM
test-nullspace.C, 1092
readSmsFormat
FFLAS, 151
readSprFormat
FFLAS, 151
READWRITE
parallel.h, 1031
Rec_Initialize
benchmark-pluq.C, 799
RecInt, 396
recLevel
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 505
Recursive, 522
reduce
FFLAS::vectorised, 281, 282
rns_double, 526
rns_double_extended, 537
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 545
reduce_modp
RNSIntegerMod< RNS >, 547, 548
reduce_modp_rnsmajor
RNSIntegerMod< RNS >, 548
ReducedColumnEchelonForm
FFPACK, 317, 318, 367
ReducedColumnEchelonForm_modular_double
ffpack.C, 990
ffpack_c.h, 1013
ReducedColumnEchelonForm_modular_float
ffpack.C, 991
ffpack_c.h, 1013
ReducedColumnEchelonForm_modular_int32_t
ffpack.C, 992
ffpack_c.h, 1014
ReducedRowEchelonForm
FFPACK, 319, 366
ReducedRowEchelonForm2_modular_double
ffpack_c.h, 1014
ReducedRowEchelonForm_modular_double
ffpack.C, 990
ffpack_c.h, 1013
ReducedRowEchelonForm_modular_float
ffpack.C, 991
ffpack_c.h, 1014
ReducedRowEchelonForm_modular_int32_t
ffpack.C, 992
ffpack_c.h, 1014
REF_modular_double
ffpack_c.h, 1015
regression-check.C, 1050
check1, 1051
check2, 1051
check3, 1051
check4, 1051
checkZeroDimCharpoly, 1051
checkZeroDimMinPoly, 1051
gf2ModularBalanced, 1051
main, 1051
Residu
Bench< Elt >, 404
Test< Elt >, 758
RETURNPARAM
parallel.h, 1033
Ring
CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >, 412
ring
RNSInteger< RNS >::RandIter, 519
RNSIntegerMod< RNS >::RandIter, 520
rnsRandIter< RNS >, 550
rint< K >, 523
RNS, 43
benchmark-fgemm-rns.C, 786
rns
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 544
rns-double-elt.h, 944
rns-double-recint.inl, 945
__FFLASFFPACK_field_rns_double_recint_INL, 945
rns-double.h, 945
ROUND_DOWN, 946
rns-double.inl, 946
__FFLASFFPACK_field_rns_double_INL, 946
rns-integer-mod.h, 946
rns-integer.h, 947
rns.h, 948
rns.inl, 948
__FFLASFFPACK_field_rns_INL, 948
rns_double, 523
_M, 527
_MMi, 527
_Mi, 527
_basis, 527
_basisMax, 527
_crt_in, 527
_crt_out, 527
_field_rns, 527
_invbasis, 527
_ldm, 528
_mi_sum, 528
_negbasis, 527
_pbits, 528
_size, 527
BasisElement, 524
ConstElement_ptr, 524
convert, 526, 527
convert_transpose, 526
Element, 524
Element_ptr, 524

init, 525, 526
init_transpose, 526
integer, 524
ModField, 524
precompute_cst, 525
reduce, 526
rns_double, 525
rns_double_elt, 528
_alloc, 529
_ptr, 529
_stride, 529
~rns_double_elt, 528
operator&, 529
rns_double_elt, 528, 529
rns_double_elt_cstptr, 529
_alloc, 532
_ptr, 532
_stride, 532
operator!=, 531
operator<, 531
operator+, 531
operator++, 531
operator+=, 531
operator-, 531
operator--, 531
operator-=, 531
operator=, 531
operator&, 530, 531
operator[], 530, 531
operator*, 530
other, 532
rns_double_elt_cstptr, 530
rns_double_elt_ptr, 532
_alloc, 534
_ptr, 534
_stride, 534
operator!=, 534
operator<, 534
operator+, 533
operator++, 533
operator+=, 534
operator-, 534
operator--, 533
operator-=, 534
operator=, 534
operator&, 533, 534
operator[], 533
operator*, 533
other, 534
rns_double_elt_ptr, 533
rns_double_extended, 535
_M, 538
_MMi, 538
_Mi, 538
_basis, 537
_basisMax, 537
_crt_in, 538
_crt_out, 538
_field_rns, 538
_invbasis, 537
_ldm, 538
_negbasis, 537
_pbits, 538
_size, 538
BasisElement, 535
ConstElement_ptr, 536
convert, 537
Element, 536
Element_ptr, 536
init, 536, 537
integer, 535
ModField, 535
precompute_cst, 536
reduce, 537
rns_double_extended, 536
RNSElementTag, 538
RNSInteger
RNSInteger< RNS >, 540
RNSInteger< RNS >, 538
_rns, 542
assign, 541
BasisElement, 539
cardinality, 541
characteristic, 540
ConstElement_ptr, 540
convert, 541
Element, 539
Element_ptr, 540
init, 541
integer, 539
isMOne, 540
isOne, 540
isZero, 540
mOne, 542
one, 542
reduce, 541
rns, 540
RNSInteger, 540
size, 540
write, 541
zero, 542
RNSInteger< RNS >::RandIter, 518
operator(), 519
RandIter, 519
random, 519
ring, 519
RNSIntegerMod
RNSIntegerMod< RNS >, 544
RNSIntegerMod< RNS >, 542
_F, 548
_Mi_modp_rns, 548
_RNSdelayed, 548
_iM_modp_rns, 548
_p, 548
_rns, 548
add, 546

areEqual, 547
assign, 546
axpyin, 546
BasisElement, 544
cardinality, 545
characteristic, 545
ConstElement_ptr, 543
convert, 546
delayed, 544
Element, 543
Element_ptr, 543
init, 545, 546
integer, 544
inv, 547
isMOne, 544
isOne, 544
isZero, 544
maxElement, 545
minElement, 545
ModField, 544
mOne, 548
mul, 546
neg, 546
one, 548
reduce, 545
reduce_modp, 547, 548
reduce_modp_rnsmajor, 548
rns, 544
RNSIntegerMod, 544
size, 544
sub, 546
write, 547
write_matrix, 547
write_matrix_long, 547
zero, 549
RNSIntegerMod< RNS >::RandIter, 519
operator(), 520
RandIter, 520
random, 520
ring, 520
RNSModulus
FFLAS::CuttingStrategy, 198
rnsRandIter
rnsRandIter< RNS >, 549
rnsRandIter< RNS >, 549
operator(), 549, 550
random, 549, 550
ring, 550
rnsRandIter, 549
round
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 558
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 599
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 617
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 2 >, 638
Simd256_impl< true, true, false, 4 >, 655
Simd256_impl< true, true, false, 8 >, 665
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 685, 692
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, false, true, 8 >, 709
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 729
ROUND_DOWN
fflas_sparse.h, 883
rns-double.h, 946
Row, 550
row
Coo< Field >, 432
Coo< ValT, IdxT >, 430, 433
Sparse< _Field, SparseMatrix_t::COO >, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 735
rowblockindex
ForStrategy2D< blocksize_t, Cut, Param >, 468
rowBlockSize
ForStrategy2D< blocksize_t, Cut, Param >, 468
rowdim
StatsMatrix, 753
RowEchelonForm
FFPACK, 316, 317, 366
RowEchelonForm_modular_double
ffpack.C, 990
ffpack_c.h, 1012
RowEchelonForm_modular_float
ffpack.C, 991
ffpack_c.h, 1012
RowEchelonForm_modular_int32_t
ffpack.C, 992
ffpack_c.h, 1013
rownumblocks
ForStrategy2D< blocksize_t, Cut, Param >, 467
RowRankProfile
FFPACK, 329, 330, 371
RowRankProfile_modular_double
ffpack.C, 998
ffpack_c.h, 1018
RowRankProfileSubmatrix
FFPACK, 333, 372
RowRankProfileSubmatrix_modular_double
ffpack.C, 999
ffpack_c.h, 1019
RowRankProfileSubmatrixIndices
FFPACK, 332, 372
RowRankProfileSubmatrixIndices_modular_double
ffpack.C, 999
ffpack_c.h, 1018
ruint< K >, 550
run

Bench< Elt >, 405
Test< Elt >, 759
run_with_field
benchmark-charpoly.C, 778
benchmark-fdot.C, 784
benchmark-quasisep.C, 800
test-charpoly.C, 1053
test-echelon.C, 1057
test-fdot.C, 1059
test-fgemm-check.C, 1061
test-fgemm.C, 1063
test-fgemv.C, 1065
test-fger.C, 1066
test-fgesv.C, 1068
test-finit.C, 1069
test-fsyr2k.C, 1072
test-fsyrk.C, 1074
test-fsytrf.C, 1075
test-ftrmm.C, 1076
test-ftrmv.C, 1077
test-ftrsm.C, 1079
test-ftrssyr2k.C, 1080
test-ftrstr.C, 1081
test-ftrsv.C, 1083
test-ftrtri.C, 1084
test-io.C, 1085
test-lu.C, 1089
test-minpoly.C, 1091
test-nullspace.C, 1093
test-quasisep.C, 1096
test-rankprofiles.C, 1097
test-solve.C, 1102
run_with_Integer
test-fdot.C, 1060

saxpy_
config-blas.h, 810
ScalAndReduce
FFLAS::Protected, 214, 218
scalar_t
FieldSimd< _Field >, 448
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 573
Simd128_impl< true, true, false, 8 >, 583
Simd128_impl< true, true, true, 2 >, 593
Simd128_impl< true, true, true, 4 >, 602
Simd128_impl< true, true, true, 8 >, 611
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 630
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 668
Simd256_impl< true, true, true, 4 >, 679
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 722
ScalFunctions< Element >, 550
add, 552
addin, 552
blend, 555
div, 553
eq, 554
fmadd, 553
fmaddin, 553
fmsub, 553
fmsubin, 553
fnmadd, 553
fnmaddin, 553
genInputs, 551
genInputsWithZero, 551
greater, 554
greater_eq, 554
lesser, 554
lesser_eq, 554
mul, 552
mulin, 553
pack, 555
pack_even, 555
pack_odd, 555
sub, 552
subin, 552
unpackhi, 554
unpacklo, 554
unpacklohi, 554
vand, 551
vandnot, 552
vectElt, 551
vor, 552
vxor, 552
zero, 551
ScalFunctionsBase< Element, Enable >, 555
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >, 556
_zero, 557
blendv, 556
ceil, 556
cmp_false, 557
cmp_true, 557
floor, 556
fma, 556
get_default_random_generator, 556
round, 556
ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution, 463
FloatingPointTestDistribution, 463
IntType, 463
operator(), 463
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 557
_zero, 559
cmp_false, 560
cmp_true, 559
fma, 558

fmaddx, 558
fmaddxin, 558
fmsubx, 558
fmsubxin, 559
fnmaddx, 559
fnmaddxin, 559
get_default_random_generator, 558
mulhi, 558
mullo, 558
mulx, 558
round, 558
sll, 559
sra, 559
srl, 559
scalp
FFLAS::vectorised, 282, 283
FFLAS::vectorised::unswitch, 285
schedule_bini.inl, 837
__FFLASFFPACK_fgemm_bini_INL, 837
schedule_winograd.inl, 837
__FFLASFFPACK_fgemm_winograd_INL, 838
schedule_winograd_acc.inl, 838
__FFLASFFPACK_fgemm_winograd_acc_INL, 838
schedule_winograd_acc_ip.inl, 839
__FFLASFFPACK_fgemm_winograd_acc_ip_INL, 839
schedule_winograd_ip.inl, 839
__FFLASFFPACK_fgemm_winograd_ip_INL, 840
scopy_
config-blas.h, 812
sdot_
config-blas.h, 811
second_component
Compose< H1, H2 >, 422
Self
Coo< ValT, IdxT >, 430, 433
Self_t
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502
SELL
FFLAS, 76
sell.h, 906
sell_pspmv.inl, 907
__FFLASFFPACK_fflas_sparse_sell_pspmv_INL, 907
sell_spmv.inl, 907
__FFLASFFPACK_fflas_sparse_sell_spmv_INL, 908
sell_utils.inl, 908
__FFLASFFPACK_fflas_sparse_sell_utils_INL, 909
SELL_ZO
FFLAS, 76
Sequential, 560
numthreads, 560
operator<<, 560
Sequential, 560
set
Simd128_impl< true, true, false, 2 >, 563, 566
Simd128_impl< true, true, false, 4 >, 574, 576
Simd128_impl< true, true, false, 8 >, 584, 586
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 634
Simd256_impl< true, true, false, 4 >, 643, 645, 648
Simd256_impl< true, true, false, 8 >, 659, 662
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 712, 715
Simd512_impl< true, true, true, 8 >, 723
set1
Simd128_impl< true, true, false, 2 >, 563, 566
Simd128_impl< true, true, false, 4 >, 574, 576
Simd128_impl< true, true, false, 8 >, 584, 586
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 633
Simd256_impl< true, true, false, 4 >, 643, 645, 648
Simd256_impl< true, true, false, 8 >, 659, 662
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 712, 715
Simd512_impl< true, true, true, 8 >, 723
set_numthreads
Parallel< C, P >, 518
SET_THREADS
parallel.h, 1031
setErrorStream
Failure, 445
setOutBounds
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 504
sgemm_
config-blas.h, 814
sgemv_
config-blas.h, 811
sger_
config-blas.h, 812
shuffle
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613

Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 650
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 716
Simd512_impl< true, true, true, 8 >, 724
shuffle_twice
Simd256_impl< true, true, false, 4 >, 649, 650
Simd256_impl< true, true, true, 4 >, 681, 687
sigma
Sparse< _Field, SparseMatrix_t::SELL >, 748
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
signbits
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 8 >, 618
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, true, false, 8 >, 719
Simd512_impl< true, true, true, 8 >, 729
Simd
fflas_simd.h, 870
simd
FieldSimd< _Field >, 448
SIMD wrapper, 42
simd.doxy, 871
Simd128
simd128.inl, 871
simd128.inl, 871
__FFLASFFPACK_fflas_ffpack_utils_simd128_INL, 871
Simd128, 871
simd128_double.inl, 871
__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL, 872
simd128_float.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL, 872
Simd128_impl< ArithType, Int, Signed, Size >, 561
Simd128_impl< true, false, true, 4 >, 561
Simd128_impl< true, false, true, 8 >, 561
Simd128_impl< true, true, false, 2 >, 561
add, 568
addin, 568
aligned_allocator, 563
aligned_vector, 563
alignment, 571
blend, 568
compliant, 566
eq, 570
fmadd, 569
fmaddin, 569
fmaddx, 565
fmaddxin, 565
fmsub, 569
fmsubin, 570
fmsubx, 565
fmsubxin, 566
fnmadd, 569
fnmaddin, 569
fnmaddx, 565
fnmaddxin, 565
gather, 564, 566
greater, 564
greater_eq, 565
hadd_to_scal, 566
is_same_element, 563
lesser, 564
lesser_eq, 565
load, 564, 566
loadu, 564, 566
mod, 570
mul, 569
mulhi, 565
mullo, 569
mulx, 565
pack, 568
pack_even, 568
pack_odd, 568
round, 570
scalar_t, 563
set, 563, 566
set1, 563, 566
shuffle, 567
sll, 567
sll128, 570
sra, 564
srl, 567
srl128, 570
store, 564, 567
storeu, 564, 567
stream, 564, 567
sub, 569
subin, 569
transpose, 568
type_string, 563
unpackhi, 567
unpackhi_intrinsic, 567
unpacklo, 567
unpacklo_intrinsic, 567
unpacklohi, 568
valid, 566
vand, 570
vandnot, 571
vect_size, 571
vect_t, 563
vor, 570
vxor, 571
zero, 570
Simd128_impl< true, true, false, 2 >::Converter, 424
t, 424
v, 424
Simd128_impl< true, true, false, 4 >, 571
add, 578

addin, 578
aligned_allocator, 573
aligned_vector, 573
alignment, 581
blend, 578
compliant, 576
eq, 580
fmadd, 579
fmaddin, 579
fmaddx, 575
fmaddxin, 575
fmsub, 579
fmsubin, 580
fmsubx, 576
fmsubxin, 576
fnmadd, 579
fnmaddin, 579
fnmaddx, 575
fnmaddxin, 575
gather, 574, 576
greater, 574
greater_eq, 575
hadd_to_scal, 576
is_same_element, 573
lesser, 575
lesser_eq, 575
load, 574, 576
loadu, 574, 577
mod, 580
mul, 579
mulhi, 575
mullo, 579
mulx, 575
pack, 578
pack_even, 578
pack_odd, 578
round, 580
scalar_t, 573
set, 574, 576
set1, 574, 576
shuffle, 577
sll, 577
sll128, 580
sra, 574
srl, 577
srl128, 580
store, 574, 577
storeu, 574, 577
stream, 574, 577
sub, 579
subin, 579
transpose, 578
type_string, 574
unpackhi, 578
unpackhi_intrinsic, 577
unpacklo, 577
unpacklo_intrinsic, 577
unpacklohi, 578
valid, 576
vand, 580
vandnot, 581
vect_size, 581
vect_t, 573
vor, 580
vxor, 581
zero, 580
Simd128_impl< true, true, false, 4 >::Converter, 424
t, 424
v, 424
Simd128_impl< true, true, false, 8 >, 581
add, 589
addin, 589
aligned_allocator, 583
aligned_vector, 583
alignment, 591
blend, 588
compliant, 586
eq, 590
fmadd, 589
fmaddin, 589
fmaddx, 585
fmaddxin, 585
fmsub, 590
fmsubin, 590
fmsubx, 586
fmsubxin, 586
fnmadd, 589
fnmaddin, 589
fnmaddx, 585
fnmaddxin, 586
gather, 584, 586
get, 587
greater, 584
greater_eq, 585
hadd_to_scal, 586
is_same_element, 583
lesser, 585
lesser_eq, 585
load, 584, 587
loadu, 584, 587
mask_high, 590
mod, 590
mul, 589
mulhi, 585
mulhi_fast, 590
mullo, 585
mulx, 585
pack, 588
pack_even, 588
pack_odd, 588
round, 590
scalar_t, 583
set, 584, 586
set1, 584, 586
shuffle, 587
signbits, 590

sll, 587
sll128, 591
sra, 584
srl, 587
srl128, 591
store, 584, 587
storeu, 584, 587
stream, 584, 587
sub, 589
subin, 589
transpose, 588
type_string, 584
unpackhi, 588
unpackhi_intrinsic, 588
unpacklo, 588
unpacklo_intrinsic, 587
unpacklohi, 588
valid, 586
vand, 591
vandnot, 591
vect_size, 591
vect_t, 583
vor, 591
vxor, 591
zero, 590
Simd128_impl< true, true, false, 8 >::Converter, 424
t, 425
v, 425
Simd128_impl< true, true, true, 2 >, 591
add, 596
addin, 597
aligned_allocator, 593
aligned_vector, 594
alignment, 600
blend, 596
compliant, 594
eq, 599
fmadd, 597
fmaddin, 597
fmaddx, 597
fmaddxin, 598
fmsub, 598
fmsubin, 598
fmsubx, 598
fmsubxin, 599
fnmadd, 598
fnmaddin, 598
fnmaddx, 598
fnmaddxin, 598
gather, 594
greater, 599
greater_eq, 599
hadd_to_scal, 599
is_same_element, 594
lesser, 599
lesser_eq, 599
load, 594
loadu, 594
mod, 599
mul, 597
mulhi, 597
mullo, 597
mulx, 597
pack, 596
pack_even, 596
pack_odd, 596
round, 599
scalar_t, 593
set, 594
set1, 594
shuffle, 595
sll, 595
sll128, 600
sra, 595
srl, 595
srl128, 600
store, 595
storeu, 595
stream, 595
sub, 597
subin, 597
transpose, 596
type_string, 594
unpackhi, 596
unpackhi_intrinsic, 595
unpacklo, 595
unpacklo_intrinsic, 595
unpacklohi, 596
valid, 594
vand, 600
vandnot, 600
vect_size, 600
vect_t, 593
vor, 600
vxor, 600
zero, 600
Simd128_impl< true, true, true, 2 >::Converter, 425
t, 425
v, 425
Simd128_impl< true, true, true, 4 >, 601
add, 605
addin, 605
aligned_allocator, 602
aligned_vector, 603
alignment, 609
blend, 605
compliant, 603
eq, 608
fmadd, 606
fmaddin, 606
fmaddx, 606
fmaddxin, 606
fmsub, 607
fmsubin, 607
fmsubx, 607
fmsubxin, 607

fnmadd, 607
fnmaddin, 607
fnmaddx, 607
fnmaddxin, 607
gather, 603
greater, 608
greater_eq, 608
hadd_to_scal, 608
is_same_element, 603
lesser, 608
lesser_eq, 608
load, 603
loadu, 603
mod, 608
mul, 606
mulhi, 606
mullo, 606
mulx, 606
pack, 605
pack_even, 605
pack_odd, 605
round, 608
scalar_t, 602
set, 603
set1, 603
shuffle, 604
sll, 604
sll128, 608
sra, 604
srl, 604
srl128, 609
store, 603
storeu, 604
stream, 604
sub, 605
subin, 606
transpose, 605
type_string, 603
unpackhi, 604
unpackhi_intrinsic, 604
unpacklo, 604
unpacklo_intrinsic, 604
unpacklohi, 605
valid, 603
vand, 609
vandnot, 609
vect_size, 609
vect_t, 602
vor, 609
vxor, 609
zero, 608
Simd128_impl< true, true, true, 4 >::Converter, 425
t, 425
v, 425
Simd128_impl< true, true, true, 8 >, 609
add, 614
addin, 615
aligned_allocator, 612
aligned_vector, 612
alignment, 619
blend, 614
compliant, 612
eq, 617
fmadd, 615
fmaddin, 615
fmaddx, 615
fmaddxin, 616
fmsub, 616
fmsubin, 616
fmsubx, 616
fmsubxin, 617
fnmadd, 616
fnmaddin, 616
fnmaddx, 616
fnmaddxin, 616
gather, 612
get, 612
greater, 617
greater_eq, 617
hadd_to_scal, 617
is_same_element, 612
lesser, 617
lesser_eq, 617
load, 612
loadu, 613
mask_high, 617
mod, 617
mul, 615
mulhi, 615
mulhi_fast, 617
mullo, 615
mulx, 615
pack, 614
pack_even, 614
pack_odd, 614
round, 617
scalar_t, 611
set, 612
set1, 612
shuffle, 613
signbits, 618
sll, 613
sll128, 618
sra, 613
srl, 613
srl128, 618
store, 613
storeu, 613
stream, 613
sub, 615
subin, 615
transpose, 614
type_string, 612
unpackhi, 614
unpackhi_intrinsic, 613
unpacklo, 614

unpacklo_intrinsic, 613
unpacklohi, 614
valid, 612
vand, 618
vandnot, 618
vect_size, 619
vect_t, 611
vor, 618
vxor, 618
zero, 618
Simd128_impl< true, true, true, 8 >::Converter, 425
t, 426
v, 426
simd128_int16.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL, 872
simd128_int32.inl, 872
__FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL, 873
simd128_int64.inl, 873
__FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL, 873
vect_t, 873
Simd128i_base, 619
sll128, 619
srl128, 619
vand, 620
vandnot, 620
vect_t, 619
vor, 620
vxor, 620
zero, 619
Simd256
simd256.inl, 874
simd256.inl, 873
__FFLASFFPACK_fflas_ffpack_utils_simd256_INL, 874
Simd256, 874
simd256_double.inl, 874
__FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL, 874
simd256_float.inl, 874
__FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL, 875
Simd256_impl< ArithType, Int, Signed, Size >, 620
Simd256_impl< true, false, true, 4 >, 620
Simd256_impl< true, false, true, 8 >, 621
add, 625
addin, 625
aligned_allocator, 622
aligned_vector, 622
alignment, 628
blend, 625
blendv, 625
ceil, 627
compliant, 623
div, 625
eq, 626
floor, 627
fmadd, 626
fmaddin, 626
fmsub, 626
fmsubin, 626
fnmadd, 626
fnmaddin, 626
gather, 623
greater, 627
greater_eq, 627
hadd, 627
hadd_to_scal, 628
is_same_element, 622
lesser, 626
lesser_eq, 626
load, 623
loadu, 623
mod, 628
mul, 625
mulin, 625
pack, 624
pack_even, 624
pack_odd, 624
round, 627
scalar_t, 622
set, 623
set1, 623
store, 623
storeu, 623
stream, 623
sub, 625
subin, 625
transpose, 624
type_string, 622
unpackhi, 624
unpackhi_intrinsic, 624
unpacklo, 624
unpacklo_intrinsic, 624
unpacklohi, 624
valid, 622
vand, 627
vandnot, 627
vect_size, 628
vect_t, 622
vor, 627
vxor, 627
zero, 623
Simd256_impl< true, false, true, 8 >::Converter, 426
t, 426
v, 426
Simd256_impl< true, true, false, 2 >, 628
add, 636
addin, 636
aligned_allocator, 630
aligned_vector, 630
alignment, 638
blend, 636
compliant, 633

eq, 637
fmadd, 637
fmaddin, 637
fmaddx, 632
fmaddxin, 633
fmsub, 637
fmsubin, 637
fmsubx, 633
fmsubxin, 633
fnmadd, 637
fnmaddin, 637
fnmaddx, 633
fnmaddxin, 633
gather, 631, 634
greater, 632
greater_eq, 632
hadd_to_scal, 633
half_t, 631
is_same_element, 630
lesser, 632
lesser_eq, 632
load, 631, 634
loadu, 631, 634
mod, 638
mul, 637
mulhi, 632
mullo, 637
mulx, 632
pack, 635
pack_even, 635
pack_odd, 635
round, 638
scalar_t, 630
set, 631, 634
set1, 631, 633
shuffle, 635
simdHalf, 630
sll, 634
sra, 632
srl, 635
store, 631, 634
storeu, 632, 634
stream, 632, 634
sub, 636
subin, 636
transpose, 636
type_string, 631
unpackhi, 635
unpackhi_intrinsic, 635
unpacklo, 635
unpacklo_intrinsic, 635
unpacklohi, 635
valid, 633
vect_size, 638
vect_t, 631
zero, 638
Simd256_impl< true, true, false, 2 >::Converter, 426
t, 426
v, 426
Simd256_impl< true, true, false, 4 >, 638
add, 652
addin, 653
aligned_allocator, 642
aligned_vector, 642
alignment, 656
blend, 652
compliant, 648
eq, 655
fmadd, 654
fmaddin, 654
fmaddx, 644, 647
fmaddxin, 644, 647
fmsub, 655
fmsubin, 655
fmsubx, 645, 647
fmsubxin, 645, 647
fnmadd, 654
fnmaddin, 654
fnmaddx, 644, 647
fnmaddxin, 645, 647
gather, 643, 645, 648
greater, 644, 646
greater_eq, 644, 646
hadd_to_scal, 645, 647
half_t, 642
is_same_element, 642
lesser, 644, 646
lesser_eq, 644, 646
load, 643, 645, 649
loadu, 643, 646, 649
mod, 655, 656
mul, 653
mulhi, 644, 646
mullo, 653
mulx, 644, 647
pack, 651
pack_even, 651
pack_odd, 651
round, 655
scalar_t, 642
set, 643, 645, 648
set1, 643, 645, 648
shuffle, 650
shuffle_twice, 649, 650
simdHalf, 642
sll, 649
sra, 643, 646
srl, 649
store, 643, 646, 649
storeu, 643, 646, 649
stream, 643, 646, 649
sub, 653
subin, 653
transpose, 652
type_string, 643, 645
unpackhi, 650, 651

unpackhi_intrinsic, 650
unpacklo, 650
unpacklo_intrinsic, 650
unpacklohi, 651
valid, 647, 648
vand, 656
vandnot, 656
vect_size, 656
vect_t, 642
vor, 656
vxor, 656
zero, 656
Simd256_impl< true, true, false, 4 >::Converter, 426
t, 427
v, 427
Simd256_impl< true, true, false, 8 >, 657
add, 664
addin, 664
aligned_allocator, 659
aligned_vector, 659
alignment, 666
blend, 664
compliant, 662
eq, 665
fmadd, 665
fmaddin, 665
fmaddx, 661
fmaddxin, 661
fmsub, 665
fmsubin, 665
fmsubx, 661
fmsubxin, 661
fnmadd, 665
fnmaddin, 665
fnmaddx, 661
fnmaddxin, 661
gather, 659, 662
get, 662
greater, 660
greater_eq, 660
hadd_to_scal, 662
half_t, 659
is_same_element, 659
lesser, 660
lesser_eq, 660
load, 660, 662
loadu, 660, 662
mask_high, 666
mod, 666
mul, 665
mulhi, 661
mulhi_fast, 666
mullo, 661
mulx, 661
pack, 664
pack_even, 664
pack_odd, 664
round, 665
scalar_t, 659
set, 659, 662
set1, 659, 662
shuffle, 663
signbits, 666
simdHalf, 659
sll, 663
sra, 660
srl, 663
store, 660, 662
storeu, 660, 663
stream, 660, 663
sub, 664
subin, 664
transpose, 664
type_string, 659
unpackhi, 663
unpackhi_intrinsic, 663
unpacklo, 663
unpacklo_intrinsic, 663
unpacklohi, 663
valid, 662
vect_size, 666
vect_t, 659
zero, 666
Simd256_impl< true, true, false, 8 >::Converter, 427
t, 427
v, 427
Simd256_impl< true, true, true, 2 >, 666
add, 672
addin, 672
aligned_allocator, 669
aligned_vector, 669
alignment, 675
blend, 672
compliant, 669
eq, 674
fmadd, 673
fmaddin, 673
fmaddx, 673
fmaddxin, 673
fmsub, 674
fmsubin, 674
fmsubx, 674
fmsubxin, 674
fnmadd, 673
fnmaddin, 673
fnmaddx, 673
fnmaddxin, 674
gather, 669
greater, 674
greater_eq, 674
hadd_to_scal, 675
half_t, 668
is_same_element, 669
lesser, 674
lesser_eq, 675
load, 670

loadu, 670
mod, 675
mul, 672
mulhi, 673
mullo, 672
mulx, 673
pack, 671
pack_even, 671
pack_odd, 671
round, 675
scalar_t, 668
set, 669
set1, 669
shuffle, 670
simdHalf, 668
sll, 670
sra, 670
srl, 670
store, 670
storeu, 670
stream, 670
sub, 672
subin, 672
transpose, 671
type_string, 669
unpackhi, 671
unpackhi_intrinsic, 671
unpacklo, 671
unpacklo_intrinsic, 670
unpacklohi, 671
valid, 669
vect_size, 675
vect_t, 668
zero, 675
Simd256_impl< true, true, true, 2 >::Converter, 427
t, 427
v, 427
Simd256_impl< true, true, true, 4 >, 675
add, 683, 689
addin, 683, 689
aligned_allocator, 679
aligned_vector, 679, 680
alignment, 692
blend, 682, 689
compliant, 680, 686
eq, 685, 691
fmadd, 683, 690
fmaddin, 683, 690
fmaddx, 684, 690
fmaddxin, 684, 690
fmsub, 684, 691
fmsubin, 684, 691
fmsubx, 685, 691
fmsubxin, 685, 691
fnmadd, 684, 690
fnmaddin, 684, 690
fnmaddx, 684, 690
fnmaddxin, 684, 690
gather, 680, 686
greater, 685, 691
greater_eq, 685, 691
hadd_to_scal, 685, 691
half_t, 679
is_same_element, 679, 680
lesser, 685, 691
lesser_eq, 685, 691
load, 680, 686
loadu, 680, 687
mod, 685, 692
mul, 683, 689
mulhi, 683, 689
mullo, 683, 689
mulx, 683, 689
pack, 682, 688
pack_even, 682, 688
pack_odd, 682, 688
round, 685, 692
scalar_t, 679
set, 680, 686
set1, 680, 686
shuffle, 681, 687
shuffle_twice, 681, 687
simdHalf, 679
sll, 681, 687
sra, 681, 687
srl, 681, 687
store, 681, 687
storeu, 681, 687
stream, 681, 687
sub, 683, 689
subin, 683, 689
transpose, 682, 688
type_string, 680, 686
unpackhi, 682, 688
unpackhi_intrinsic, 681, 688
unpacklo, 682, 688
unpacklo_intrinsic, 681, 687
unpacklohi, 682, 688
valid, 680, 686
vand, 692
vandnot, 692
vect_size, 692
vect_t, 679
vor, 692
vxor, 692
zero, 692
Simd256_impl< true, true, true, 4 >::Converter, 427
t, 428
v, 428
Simd256_impl< true, true, true, 8 >, 693
add, 698
addin, 698
aligned_allocator, 695
aligned_vector, 695
alignment, 701
blend, 698

compliant, 695
eq, 700
fmadd, 699
fmaddin, 699
fmaddx, 699
fmaddxin, 699
fmsub, 700
fmsubin, 700
fmsubx, 700
fmsubxin, 700
fnmadd, 699
fnmaddin, 699
fnmaddx, 699
fnmaddxin, 699
gather, 696
get, 696
greater, 700
greater_eq, 700
hadd_to_scal, 701
half_t, 695
is_same_element, 695
lesser, 700
lesser_eq, 700
load, 696
loadu, 696
mask_high, 701
mod, 701
mul, 698
mulhi, 698
mulhi_fast, 701
mullo, 698
mulx, 699
pack, 697
pack_even, 697
pack_odd, 697
round, 701
scalar_t, 695
set, 695
set1, 695
shuffle, 697
signbits, 701
simdHalf, 695
sll, 696
sra, 696
srl, 696
store, 696
storeu, 696
stream, 696
sub, 698
subin, 698
transpose, 698
type_string, 695
unpackhi, 697
unpackhi_intrinsic, 697
unpacklo, 697
unpacklo_intrinsic, 697
unpacklohi, 697
valid, 695
vect_size, 701
vect_t, 695
zero, 701
Simd256_impl< true, true, true, 8 >::Converter, 428
t, 428
v, 428
simd256_int16.inl, 875
__FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL, 875
simd256_int32.inl, 875
__FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL, 876
simd256_int64.inl, 876
__FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL, 876
vect_t, 876
Simd256fp_base, 702
Simd256i_base, 702
vect_t, 702
zero, 702
Simd512
simd512.inl, 877
simd512.inl, 876
__FFLASFFPACK_simd512_INL, 877
Simd512, 877
simd512_double.inl, 877
__FFLASFFPACK_simd512_double_INL, 877
simd512_float.inl, 877
__FFLASFFPACK_simd512_float_INL, 877
Simd512_impl< ArithType, Int, Signed, Size >, 702
Simd512_impl< true, false, true, 4 >, 702
Simd512_impl< true, false, true, 8 >, 703
add, 707
addin, 707
aligned_allocator, 704
aligned_vector, 704
alignment, 709
blend, 707
blendv, 707
ceil, 709
compliant, 704
div, 707
eq, 708
floor, 709
fmadd, 707
fmaddin, 708
fmsub, 708
fmsubin, 708
fnmadd, 708
fnmaddin, 708
gather, 705
greater, 708
greater_eq, 709
hadd, 709
hadd_to_scal, 709
is_same_element, 704
lesser, 708
lesser_eq, 708

load, 705
loadu, 705
mul, 707
mulin, 707
pack, 706
pack_even, 706
pack_odd, 706
round, 709
scalar_t, 704
set, 705
set1, 705
shuffle, 705
store, 705
storeu, 705
stream, 705
sub, 707
subin, 707
transpose, 706
type_string, 704
unpackhi, 706
unpackhi_intrinsic, 706
unpacklo, 706
unpacklo_intrinsic, 705
unpacklohi, 706
valid, 704
vect_size, 709
vect_t, 704
zero, 704
Simd512_impl< true, true, false, 8 >, 709
add, 718
addin, 718
aligned_allocator, 712
aligned_vector, 712
alignment, 720
blend, 718
compliant, 715
eq, 719
fmadd, 718
fmaddin, 718
fmaddx, 714
fmaddxin, 714
fmsub, 719
fmsubin, 719
fmsubx, 714
fmsubxin, 715
fnmadd, 718
fnmaddin, 719
fnmaddx, 714
fnmaddxin, 714
gather, 712, 715
greater, 713
greater_eq, 713
hadd_to_scal, 715
half_t, 712
is_same_element, 712
lesser, 713
lesser_eq, 714
load, 713, 716
loadu, 713, 716
mask_high, 719
maskstore, 713, 716
mod, 719
mul, 718
mulhi, 714
mulhi_fast, 719
mullo, 714
mulx, 714
pack, 717
pack_even, 717
pack_odd, 717
round, 719
scalar_t, 712
set, 712, 715
set1, 712, 715
shuffle, 716
signbits, 719
simdHalf, 712
sll, 716
sra, 713
srl, 716
store, 713, 716
storeu, 713, 716
stream, 713, 716
sub, 718
subin, 718
transpose, 717
type_string, 712
unpackhi, 717
unpackhi_intrinsic, 717
unpacklo, 717
unpacklo_intrinsic, 716
unpacklohi, 717
valid, 715
vand, 720
vandnot, 720
vect_size, 720
vect_t, 712
vor, 720
vxor, 720
zero, 720
Simd512_impl< true, true, false, 8 >::Converter, 428
t, 428
v, 428
Simd512_impl< true, true, true, 8 >, 720
add, 726
addin, 726
aligned_allocator, 723
aligned_vector, 723
alignment, 730
blend, 726
compliant, 723
eq, 728
fmadd, 727
fmaddin, 727
fmaddx, 727
fmaddxin, 727

fmsub, 728
fmsubin, 728
fmsubx, 728
fmsubxin, 728
fnmadd, 727
fnmaddin, 727
fnmaddx, 727
fnmaddxin, 727
gather, 723
greater, 728
greater_eq, 728
hadd_to_scal, 728
half_t, 722
is_same_element, 723
lesser, 728
lesser_eq, 728
load, 724
loadu, 724
mask_high, 729
maskstore, 724
mod, 729
mul, 726
mulhi, 726
mulhi_fast, 729
mullo, 726
mulx, 726
pack, 725
pack_even, 725
pack_odd, 725
round, 729
scalar_t, 722
set, 723
set1, 723
shuffle, 724
signbits, 729
simdHalf, 722
sll, 724
sra, 724
srl, 724
store, 724
storeu, 724
stream, 724
sub, 726
subin, 726
transpose, 725
type_string, 723
unpackhi, 725
unpackhi_intrinsic, 725
unpacklo, 725
unpacklo_intrinsic, 725
unpacklohi, 725
valid, 723
vand, 729
vandnot, 729
vect_size, 730
vect_t, 722
vor, 729
vxor, 729
zero, 729
Simd512_impl< true, true, true, 8 >::Converter, 428
t, 429
v, 429
simd512_int32.inl, 878
__FFLASFFPACK_simd512_int32_INL, 878
simd512_int64.inl, 878
_simd512_int64_INL, 878
vect_t, 878
Simd512i_base, 730
vand, 731
vandnot, 731
vect_t, 730
vor, 730
vxor, 730
zero, 730
SIMD_INT
fflas_simd.h, 870
simd_modular.inl, 879
SimdChooser< T, bool, bool >, 731
SimdChooser< T, false, b >, 731
value, 731
SimdChooser< T, true, false >, 731
value, 732
SimdChooser< T, true, true >, 732
value, 732
simdHalf
Simd256_impl< true, true, false, 2 >, 630
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 668
Simd256_impl< true, true, true, 4 >, 679
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 722
SimdSparseMatrix
FFLAS, 72
simdToType< T >, 732
Single, 732
size
BlockTransposeSIMD< Field, Simd, >, 407
Info, 482, 483
RNSInteger< RNS >, 540
RNSIntegerMod< RNS >, 544
SIZEOF___INT64_T
config.h, 818
SIZEOF_CHAR
config.h, 817
SIZEOF_INT
config.h, 817
SIZEOF_LONG
config.h, 817
SIZEOF_LONG_LONG
config.h, 818
SIZEOF_SHORT
config.h, 818
sll

ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, true, false, 2 >, 634
Simd256_impl< true, true, false, 4 >, 649
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, true, false, 8 >, 716
Simd512_impl< true, true, true, 8 >, 724
sll128
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 619
Solve
FFPACK, 327, 370
solve.C, 803
main, 803
Solve_modular_double
ffpack.C, 997
ffpack_c.h, 1016
solveLB
FFPACK, 346, 370
solveLB2
FFPACK, 346, 370
solveLB2_modular_double
ffpack.C, 997
ffpack_c.h, 1017
solveLB_modular_double
ffpack.C, 997
ffpack_c.h, 1017
Sparse< _Field, SparseMatrix_t::COO >, 732
col, 733
dat, 733
delayed, 733
Field, 733
kmax, 733
m, 733
maxrow, 734
n, 733
nElements, 734
nnz, 733
row, 733
Sparse< _Field, SparseMatrix_t::COO_ZO >, 734
col, 735
cst, 735
dat, 735
delayed, 735
Field, 734
kmax, 735
m, 735
maxrow, 735
n, 735
nElements, 735
nnz, 735
row, 735
Sparse< _Field, SparseMatrix_t::CSR >, 736
col, 737
dat, 737
delayed, 736
Field, 736
kmax, 736
m, 736
maxrow, 737
n, 736
nElements, 737
nnz, 736
st, 737
stend, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 737
col, 738
dat, 738
delayed, 738
Field, 738
kmax, 738
m, 738
maxrow, 738
n, 738
nElements, 738
nMOnes, 739
nnz, 738
nOnes, 739
nOthers, 739
st, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 739
col, 740
cst, 740
dat, 741
delayed, 740
Field, 740
kmax, 740
m, 740
maxrow, 740
n, 740
nElements, 740
nnz, 740
st, 740
stend, 740
Sparse< _Field, SparseMatrix_t::ELL >, 741
col, 742
dat, 742
delayed, 741
Field, 741
kmax, 741
ld, 742
m, 742

maxrow, 742
n, 742
nElements, 742
nnz, 742
Sparse< _Field, SparseMatrix_t::ELL_simd >, 742
chunk, 743
col, 744
dat, 744
delayed, 743
kmax, 743
ld, 743
m, 743
maxrow, 743
n, 743
nChunks, 744
nElements, 743
nnz, 743
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >, 744
chunk, 745
col, 745
cst, 744
dat, 745
delayed, 744
kmax, 745
ld, 745
m, 745
maxrow, 745
n, 745
nChunks, 745
nElements, 745
nnz, 745
Sparse< _Field, SparseMatrix_t::ELL_ZO >, 746
col, 747
cst, 746
dat, 747
delayed, 746
Field, 746
kmax, 746
ld, 747
m, 746
maxrow, 747
n, 746
nElements, 747
nnz, 747
Sparse< _Field, SparseMatrix_t::SELL >, 747
chunk, 748
chunkSize, 749
col, 749
dat, 749
delayed, 748
Field, 748
kmax, 748
m, 748
maxrow, 748
n, 748
nChunks, 749
nElements, 749
nnz, 749
perm, 749
sigma, 748
st, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 749
chunk, 750
chunkSize, 751
col, 751
cst, 750
dat, 751
delayed, 750
Field, 750
kmax, 750
m, 750
maxrow, 751
n, 750
nChunks, 751
nElements, 751
nnz, 751
perm, 751
sigma, 751
st, 751
Sparse< Field, SparseMatrix_t, IdxT, PtrT >, 732
sparse_delete
FFLAS, 146–150, 153
sparse_init
FFLAS, 145–150, 153
sparse_matrix_traits.h, 909
sparse_print
FFLAS, 147, 150, 153
SparseMatrix_t
FFLAS, 76
SpecRankProfile
FFPACK, 369
SpecRankProfile_modular_double
ffpack.C, 996
ffpack_c.h, 1016
splitt
parallel.h, 1036
SPLITTER
parallel.h, 1036
splitting_0
parallel.h, 1035
splitting_1
parallel.h, 1035
splitting_2
parallel.h, 1036
splitting_3
parallel.h, 1036
SpMat< Field, flag >, 751
_coo, 752
_csr, 752
_ell, 752
square_inplace
FFLAS::_ftranspose_impl, 190
sra
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 564

Simd128_impl< true, true, false, 4 >, 574
Simd128_impl< true, true, false, 8 >, 584
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, true, false, 2 >, 632
Simd256_impl< true, true, false, 4 >, 643, 646
Simd256_impl< true, true, false, 8 >, 660
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, true, false, 8 >, 713
Simd512_impl< true, true, true, 8 >, 724
srl
ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >, 559
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 649
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, true, false, 8 >, 716
Simd512_impl< true, true, true, 8 >, 724
srl128
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 619
sscal_
config-blas.h, 813
ST
fflas_transpose.h, 912
st
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_HYB >, 738
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
Sparse< _Field, SparseMatrix_t::SELL >, 749
Sparse< _Field, SparseMatrix_t::SELL_ZO >, 751
StatsMatrix, 752
averageCol, 753
averageColDifference, 754
averageRow, 753
averageRowDifference, 754
coldim, 753
denseCols, 754
denseRows, 754
deviationCol, 753
deviationColDifference, 754
deviationRow, 753
deviationRowDifference, 754
maxCol, 753
maxColDifference, 754
maxRow, 753
maxRowDifference, 754
minCol, 753
minColDifference, 754
minRow, 753
minRowDifference, 754
nDenseCols, 754
nDenseRows, 754
nEmptyCols, 754
nEmptyColsEnd, 754
nEmptyRows, 754
nMOnes, 753
nnz, 753
nOnes, 753
nOthers, 753
rowdim, 753
STD_RECINT_SIZE
benchmark-fgemm-mp.C, 785
benchmark-fgemv-mp.C, 788
STDC_HEADERS
config.h, 818
stend
Sparse< _Field, SparseMatrix_t::CSR >, 737
Sparse< _Field, SparseMatrix_t::CSR_ZO >, 740
store
Simd128_impl< true, true, false, 2 >, 564, 567
Simd128_impl< true, true, false, 4 >, 574, 577
Simd128_impl< true, true, false, 8 >, 584, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 631, 634
Simd256_impl< true, true, false, 4 >, 643, 646, 649
Simd256_impl< true, true, false, 8 >, 660, 662
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
storeu
Simd128_impl< true, true, false, 2 >, 564, 567
Simd128_impl< true, true, false, 4 >, 574, 577
Simd128_impl< true, true, false, 8 >, 584, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 632, 634
Simd256_impl< true, true, false, 4 >, 643, 646, 649

Simd256_impl< true, true, false, 8 >, 660, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
stream
Simd128_impl< true, true, false, 2 >, 564, 567
Simd128_impl< true, true, false, 4 >, 574, 577
Simd128_impl< true, true, false, 8 >, 584, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 632, 634
Simd256_impl< true, true, false, 4 >, 643, 646, 649
Simd256_impl< true, true, false, 8 >, 660, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 696
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 713, 716
Simd512_impl< true, true, true, 8 >, 724
string, 755
string::const_iterator, 423
string::const_reverse_iterator, 423
string::iterator, 490
string::reverse_iterator, 523
strmm_
config-blas.h, 813
strsm_
config-blas.h, 813
sub
FFLAS::vectorised, 280
FieldSimd< _Field >, 449, 450
RNSIntegerMod< RNS >, 546
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 653
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
sub_r
FieldSimd< _Field >, 450
subin
FieldSimd< _Field >, 450
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 569
Simd128_impl< true, true, false, 4 >, 579
Simd128_impl< true, true, false, 8 >, 589
Simd128_impl< true, true, true, 2 >, 597
Simd128_impl< true, true, true, 4 >, 606
Simd128_impl< true, true, true, 8 >, 615
Simd256_impl< true, false, true, 8 >, 625
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 653
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 672
Simd256_impl< true, true, true, 4 >, 683, 689
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 707
Simd512_impl< true, true, false, 8 >, 718
Simd512_impl< true, true, true, 8 >, 726
subin_r
FieldSimd< _Field >, 450
subp
FFLAS::vectorised, 280
support_fast_mod< double >, 755
support_fast_mod< float >, 756
support_fast_mod< int64_t >, 756
support_fast_mod< T >, 755
support_simd< T >, 756
support_simd_add< T >, 756
support_simd_mod< T >, 757
swapval
FFPACK, 381
SYNCH_GROUP
parallel.h, 1031
SysTimer
FFLAS, 74

T
limits< char >, 491
limits< double >, 492
limits< float >, 492
limits< Givaro::Integer >, 493
limits< int >, 493
limits< long >, 494
limits< long long >, 495
limits< RecInt::rint< K > >, 495
limits< RecInt::ruint< K > >, 496
limits< short int >, 497
limits< signed char >, 497
limits< unsigned char >, 498
limits< unsigned int >, 498
limits< unsigned long >, 499
limits< unsigned long long >, 500
limits< unsigned short int >, 500
t
Simd128_impl< true, true, false, 2 >::Converter, 424
Simd128_impl< true, true, false, 4 >::Converter, 424

Simd128_impl< true, true, false, 8 >::Converter, 425
Simd128_impl< true, true, true, 2 >::Converter, 425
Simd128_impl< true, true, true, 4 >::Converter, 425
Simd128_impl< true, true, true, 8 >::Converter, 426
Simd256_impl< true, false, true, 8 >::Converter, 426
Simd256_impl< true, true, false, 2 >::Converter, 426
Simd256_impl< true, true, false, 4 >::Converter, 427
Simd256_impl< true, true, false, 8 >::Converter, 427
Simd256_impl< true, true, true, 2 >::Converter, 427
Simd256_impl< true, true, true, 4 >::Converter, 428
Simd256_impl< true, true, true, 8 >::Converter, 428
Simd512_impl< true, true, false, 8 >::Converter, 428
Simd512_impl< true, true, true, 8 >::Converter, 429
TASK
parallel.h, 1030
tBC
test-lu.C, 1089
test-permutations.C, 1094
Test
Test< Elt >, 759
test
test-maxdelayeddim.C, 1090
Test< Elt >, 757
_mm, 759
_nn, 759
cardinality, 759
doTests, 759
Elt_ptr, 758
enable_if_no_simd_t, 758
enable_if_simd128_t, 758
enable_if_simd256_t, 758
enable_if_simd512_t, 758
enable_if_t, 758
F, 759
Field, 758
is_same_element, 758
Residu, 758
run, 759
Test, 759
test_ftranspose, 759
test-charpoly-check.C, 1051
ENABLE_CHECKER_charpoly, 1052
main, 1052
printPolynomial, 1052
TIME_CHECKER_CHARPOLY, 1052
test-charpoly.C, 1052
launch_test, 1052
main, 1053
run_with_field, 1053
test-compressQ.C, 1053
Field, 1053
main, 1054
printvect, 1053
test-det-check.C, 1054
ENABLE_CHECKER_Det, 1054
main, 1054
TIME_CHECKER_Det, 1054
test-det.C, 1054
main, 1055
test_det, 1055
test-echelon.C, 1055
__FFLASFFPACK_GAUSSJORDAN_BASECASE, 1056
__FFLASFFPACK_PLUQ_THRESHOLD, 1056
__FFLASFFPACK_SEQUENTIAL, 1056
main, 1057
run_with_field, 1057
test_colechelon, 1056
test_redcolechelon, 1056
test_redrowechelon, 1057
test_rowechelon, 1056
test-fadd.C, 1057
main, 1058
test_fadd, 1058
test_faddin, 1058
test_fsub, 1058
test_fsubin, 1058
test-fdot.C, 1059
check_fdot, 1059
ENABLE_ALL_CHECKINGS, 1059
main, 1060
run_with_field, 1059
run_with_Integer, 1060
test-fgemm-check.C, 1060
ENABLE_ALL_CHECKINGS, 1060
launch_MM_dispatch, 1060
main, 1061
run_with_field, 1061
test-fgemm.C, 1061
check_MM, 1062
ENABLE_CHECKER_fgemm, 1062
launch_MM, 1062
launch_MM_dispatch, 1062
main, 1063
run_with_field, 1063
test-fgemv.C, 1063
check_MV, 1064
launch_MV, 1064
launch_MV_dispatch, 1064
main, 1065
run_with_field, 1065
test-fger.C, 1065
check_fger, 1066

launch_fger, 1066
launch_fger_dispatch, 1066
main, 1067
run_with_field, 1066
TIME, 1066
test-fgesv.C, 1067
main, 1068
run_with_field, 1068
test_rect_fgesv, 1067
test_square_fgesv, 1067
test-finit.C, 1068
main, 1069
run_with_field, 1069
test_freduce, 1069
test-fscal.C, 1069
main, 1070
RandomMatrix, 1070
test_fscal, 1070
test_fscalin, 1070
test-fsyr2k.C, 1071
check_fsyr2k, 1072
ENABLE_ALL_CHECKINGS, 1071
main, 1072
run_with_field, 1072
test-fsyrk.C, 1072
check_computeS1S2, 1074
check_fsyrk, 1073
check_fsyrk_bkdiag, 1073
check_fsyrk_diag, 1073
ENABLE_ALL_CHECKINGS, 1073
main, 1074
run_with_field, 1074
test-fsytrf.C, 1074
main, 1075
operator<<, 1075
run_with_field, 1075
test_generic_fsytrf, 1075
test_RPM_fsytrf, 1075
test-ftrmm.C, 1075
__FFLASFFPACK_SEQUENTIAL, 1076
check_ftrmm, 1076
main, 1076
run_with_field, 1076
test-ftrmv.C, 1077
__FFLASFFPACK_SEQUENTIAL, 1077
check_ftrmv, 1077
ENABLE_ALL_CHECKINGS, 1077
main, 1078
run_with_field, 1077
test-ftrsm-check.C, 1078
ENABLE_ALL_CHECKINGS, 1078
main, 1078
test-ftrsm.C, 1078
__FFLASFFPACK_SEQUENTIAL, 1079
check_ftrsm, 1079
ENABLE_ALL_CHECKINGS, 1079
main, 1079
run_with_field, 1079
test-ftrssyr2k.C, 1080
check_ftrssyr2k, 1080
ENABLE_ALL_CHECKINGS, 1080
main, 1081
run_with_field, 1080
test-ftrstr.C, 1081
check_ftrstr, 1081
ENABLE_ALL_CHECKINGS, 1081
main, 1082
run_with_field, 1081
test-ftrsv.C, 1082
__FFLASFFPACK_SEQUENTIAL, 1082
check_ftrsv, 1082
ENABLE_ALL_CHECKINGS, 1082
main, 1083
run_with_field, 1083
test-ftrtri.C, 1083
__FFLASFFPACK_SEQUENTIAL, 1083
check_ftrtri, 1084
ENABLE_ALL_CHECKINGS, 1083
main, 1084
run_with_field, 1084
test-interfaces-c.c, 1084
main, 1084
test-invert-check.C, 1084
ENABLE_ALL_CHECKINGS, 1085
main, 1085
test-io.C, 1085
main, 1086
run_with_field, 1085
test-lu.C, 1086
__FFLASFFPACK_SEQUENTIAL, 1087
__LUDIVINE_CUTOFF, 1087
BASECASE_K, 1087
launch_test, 1089
main, 1089
mvcnt, 1090
run_with_field, 1089
tBC, 1089
test_LUdivine, 1087
test_pluq, 1088
tgemm, 1089
timtot, 1090
tperm, 1089
trest, 1089
ttrsm, 1089
verifPLUQ, 1087
test-maxdelayeddim.C, 1090
main, 1090
MAX_WITH_SIZE_T, 1090
test, 1090
test-minpoly.C, 1090
check_minpoly, 1091
main, 1091
run_with_field, 1091
test-multifile1.C, 1091
test-multifile2.C, 1091
main, 1092

test-nullspace.C, 1092
checkingMessage, 1092
main, 1093
readOrRandomMatrixWithRankAndRandomRPM, 1092
run_with_field, 1093
test_nullspace, 1093
test-permutations.C, 1093
checkMonotonicApplyP, 1094
main, 1094
tBC, 1094
tgemm, 1094
timtot, 1094
tperm, 1094
trest, 1094
ttrsm, 1094
test-pluq-check.C, 1094
ENABLE_ALL_CHECKINGS, 1095
main, 1095
test-quasisep.C, 1095
launch_test, 1095
main, 1096
run_with_field, 1096
test_BruhatGenerator, 1095
testLTQSRPM, 1096
test-rankprofiles.C, 1096
__FFLASFFPACK_SEQUENTIAL, 1097
main, 1097
run_with_field, 1097
test-rpm.C, 1097
checkRPM, 1097
checkSymmetricRPM, 1097
main, 1098
test-simd.C, 1098
_TEST_ONE, 1099
check_eq, 1100
cmp, 1100
eval_func_on_array, 1100
main, 1101
operator<<, 1101
TEST_IMPL, 1100
test_impl, 1101
test_impl_base, 1101
TEST_ONE_OP, 1099
TEST_ONE_OP_WZ, 1099
test-solve.C, 1101
check_solve, 1101
main, 1102
run_with_field, 1102
test-storage-transpose.C, 1102
main, 1102
test-utils.h, 1047
test_BruhatGenerator
test-quasisep.C, 1095
test_colechelon
test-echelon.C, 1056
test_det
test-det.C, 1055
test_fadd
test-fadd.C, 1058
test_faddin
test-fadd.C, 1058
test_freduce
test-finit.C, 1069
test_fscal
test-fscal.C, 1070
test_fscalin
test-fscal.C, 1070
test_fsub
test-fadd.C, 1058
test_fsubin
test-fadd.C, 1058
test_ftranspose
Test< Elt >, 759
test_generic_fsytrf
test-fsytrf.C, 1075
TEST_IMPL
test-simd.C, 1100
test_impl
test-simd.C, 1101
test_impl_base
test-simd.C, 1101
test_LUdivine
test-lu.C, 1087
test_nullspace
test-nullspace.C, 1093
TEST_ONE_OP
test-simd.C, 1099
TEST_ONE_OP_WZ
test-simd.C, 1099
test_pluq
test-lu.C, 1088
test_rect_fgesv
test-fgesv.C, 1067
test_redcolechelon
test-echelon.C, 1056
test_redrowechelon
test-echelon.C, 1057
test_rowechelon
test-echelon.C, 1056
test_RPM_fsytrf
test-fsytrf.C, 1075
test_square_fgesv
test-fgesv.C, 1067
testLTQSRPM
test-quasisep.C, 1096
TestOneMethod
TestOneMethod< Simd >, 761
TestOneMethod< Simd >, 760
Element, 761
enable_if_t, 761
evaluate_scalar_method, 761
evaluate_simd_method, 762
getStatus, 762
getTestName, 762
inputs, 762

name, 762
nb_lref, 762
outputs_scalar, 763
outputs_simd, 762
TestOneMethod, 761
vect_size, 762
vect_t, 761
vectElt, 761
writeDebugData, 762
writeResultLine, 762
tfn_minus, 763
operator(), 763
tfn_minus_eq, 763
operator(), 763
tfn_mul, 763
operator(), 764
tfn_mul_eq, 764
operator(), 764
tfn_plus, 764
operator(), 764
tfn_plus_eq, 765
operator(), 765
tgemm
test-lu.C, 1089
test-permutations.C, 1094
THREAD_INDEX
parallel.h, 1031
THREADS
benchmark-fgemm-rns.C, 786
Threads, 765
threads_fgemm
FFPACK, 361
threads_ftrsm
FFPACK, 361
THREED
benchmark-fgemm-rns.C, 786
ThreeD, 765
THREEDA
benchmark-fgemm-rns.C, 786
ThreeDAdaptive, 765
ThreeDInPlace, 765
THREEDIP
benchmark-fgemm-rns.C, 786
TIME
test-fger.C, 1066
TIME_CHECKER_CHARPOLY
test-charpoly-check.C, 1052
TIME_CHECKER_Det
test-det-check.C, 1054
Timer
FFLAS, 73
timer.h, 1047
timtot
test-lu.C, 1090
test-permutations.C, 1094
TInverter
FFPACK, 344, 347
tmain
benchmark-fgemm-mp.C, 785
benchmark-fgemv-mp.C, 788
Todo List, 9
tperm
test-lu.C, 1089
test-permutations.C, 1094
transpose
BlockTransposeSIMD< Field, Simd, >, 407, 408
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 636
Simd256_impl< true, true, false, 4 >, 652
Simd256_impl< true, true, false, 8 >, 664
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 698
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
trest
test-lu.C, 1089
test-permutations.C, 1094
trinv_left
FFPACK, 309, 365
trinv_left_modular_double
ffpack.C, 989
ffpack_c.h, 1010
TRSMBound
FFLAS::Protected, 212
TRSMHelper
TRSMHelper< RecIterTrait, ParSeqTrait >, 766
TRSMHelper< RecIterTrait, ParSeqTrait >, 765
parseq, 767
pMMH, 766
TRSMHelper, 766
TTimer
arithprog.C, 771
benchmark-dgemm.C, 780
charpoly.C, 772
ttrsm
test-lu.C, 1089
test-permutations.C, 1094
Tutorial, 2
TWOD
benchmark-fgemm-rns.C, 786
TwoD, 767
TWODA
benchmark-fgemm-rns.C, 786
TwoDAdaptive, 767
type
Argument, 400
associatedDelayedField< const FFPACK::RNSIntegerMod<
RNS > >, 401

associatedDelayedField< const Givaro::Modular< T, X > >, 402
associatedDelayedField< const Givaro::ModularBalanced< T > >, 402
associatedDelayedField< const Givaro::ZRing< T > >, 403
associatedDelayedField< Field >, 401
CompactElement< double >, 419
CompactElement< Element >, 419
CompactElement< float >, 419
CompactElement< int16_t >, 420
CompactElement< int32_t >, 420
CompactElement< int64_t >, 420
is_simd< T >, 484
TYPE_BOOL
args-parser.h, 1038
TYPE_DOUBLE
args-parser.h, 1039
TYPE_INT
args-parser.h, 1039
TYPE_INTEGER
args-parser.h, 1039
type_integer
args-parser.h, 1038
TYPE_INTLIST
args-parser.h, 1039
TYPE_LONGLONG
args-parser.h, 1039
TYPE_NONE
args-parser.h, 1039
TYPE_STR
args-parser.h, 1039
type_string
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 574
Simd128_impl< true, true, false, 8 >, 584
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 631
Simd256_impl< true, true, false, 4 >, 643, 645
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 723
TYPE_UINT64
args-parser.h, 1039

unfit
FFLAS::Protected, 220, 221
unpackhi
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 650, 651
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
unpackhi_intrinsic
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 650
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 681, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
unpacklo
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 650
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
unpacklo_intrinsic
Simd128_impl< true, true, false, 2 >, 567
Simd128_impl< true, true, false, 4 >, 577
Simd128_impl< true, true, false, 8 >, 587
Simd128_impl< true, true, true, 2 >, 595
Simd128_impl< true, true, true, 4 >, 604
Simd128_impl< true, true, true, 8 >, 613
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635

Simd256_impl< true, true, false, 4 >, 650
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 670
Simd256_impl< true, true, true, 4 >, 681, 687
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 705
Simd512_impl< true, true, false, 8 >, 716
Simd512_impl< true, true, true, 8 >, 725
unpacklohi
ScalFunctions< Element >, 554
Simd128_impl< true, true, false, 2 >, 568
Simd128_impl< true, true, false, 4 >, 578
Simd128_impl< true, true, false, 8 >, 588
Simd128_impl< true, true, true, 2 >, 596
Simd128_impl< true, true, true, 4 >, 605
Simd128_impl< true, true, true, 8 >, 614
Simd256_impl< true, false, true, 8 >, 624
Simd256_impl< true, true, false, 2 >, 635
Simd256_impl< true, true, false, 4 >, 651
Simd256_impl< true, true, false, 8 >, 663
Simd256_impl< true, true, true, 2 >, 671
Simd256_impl< true, true, true, 4 >, 682, 688
Simd256_impl< true, true, true, 8 >, 697
Simd512_impl< true, false, true, 8 >, 706
Simd512_impl< true, true, false, 8 >, 717
Simd512_impl< true, true, true, 8 >, 725
UnparametricTag, 767
updateD
FFPACK::Protected, 394
USE_OPENMP
config.h, 818
UserTimer
FFLAS, 74
utils.h, 910

v
Simd128_impl< true, true, false, 2 >::Converter, 424
Simd128_impl< true, true, false, 4 >::Converter, 424
Simd128_impl< true, true, false, 8 >::Converter, 425
Simd128_impl< true, true, true, 2 >::Converter, 425
Simd128_impl< true, true, true, 4 >::Converter, 425
Simd128_impl< true, true, true, 8 >::Converter, 426
Simd256_impl< true, false, true, 8 >::Converter, 426
Simd256_impl< true, true, false, 2 >::Converter, 426
Simd256_impl< true, true, false, 4 >::Converter, 427
Simd256_impl< true, true, false, 8 >::Converter, 427
Simd256_impl< true, true, true, 2 >::Converter, 427
Simd256_impl< true, true, true, 4 >::Converter, 428
Simd256_impl< true, true, true, 8 >::Converter, 428
Simd512_impl< true, true, false, 8 >::Converter, 428
Simd512_impl< true, true, true, 8 >::Converter, 429
val
Coo< Field >, 432
Coo< ValT, IdxT >, 430, 433
valid
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 566
Simd128_impl< true, true, false, 4 >, 576
Simd128_impl< true, true, false, 8 >, 586
Simd128_impl< true, true, true, 2 >, 594
Simd128_impl< true, true, true, 4 >, 603
Simd128_impl< true, true, true, 8 >, 612
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 633
Simd256_impl< true, true, false, 4 >, 647, 648
Simd256_impl< true, true, false, 8 >, 662
Simd256_impl< true, true, true, 2 >, 669
Simd256_impl< true, true, true, 4 >, 680, 686
Simd256_impl< true, true, true, 8 >, 695
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 715
Simd512_impl< true, true, true, 8 >, 723
VALUE
parallel.h, 1031
value
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, 397
AlgoChooser< ModeT, ParSeq >, 397
ALL< false, v... >, 398
ALL< true, v... >, 398
ALL<>, 398
AreEqual< X, X >, 399
AreEqual< X, Y >, 399
compatible_data_type< Field >, 421
compatible_data_type< Givaro::ZRing< double > >, 421
compatible_data_type< Givaro::ZRing< float > >, 421
ElementTraits< double >, 438
ElementTraits< Element >, 438
ElementTraits< FFPACK::rns_double_elt >, 438
ElementTraits< float >, 439
ElementTraits< Givaro::Integer >, 439
ElementTraits< int16_t >, 439
ElementTraits< int32_t >, 440
ElementTraits< int64_t >, 440
ElementTraits< int8_t >, 440
ElementTraits< RecInt::rint< K > >, 441
ElementTraits< RecInt::rmint< K, MG > >, 441
ElementTraits< RecInt::ruint< K > >, 442

ElementTraits< uint16_t >, 442
ElementTraits< uint32_t >, 442
ElementTraits< uint64_t >, 443
ElementTraits< uint8_t >, 443
has_minus_eq_impl< C >, 475
has_minus_impl< C >, 475
has_mul_eq_impl< C >, 475
has_mul_impl< C >, 476
has_operation< T >, 476
has_plus_eq_impl< C >, 476
has_plus_impl< C >, 477
is_all_same< T, Args... >, 483
is_all_same<>, 484
is_simd< T >, 484
ModeTraits< Field >, 506
ModeTraits< Givaro::Modular< Element, Compute > >, 507
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, 507
ModeTraits< Givaro::Modular< int16_t, Compute > >, 508
ModeTraits< Givaro::Modular< int32_t, Compute > >, 508
ModeTraits< Givaro::Modular< int64_t, uint64_t > >, 508
ModeTraits< Givaro::Modular< int8_t, Compute > >, 509
ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > >, 509
ModeTraits< Givaro::Modular< uint16_t, Compute > >, 510
ModeTraits< Givaro::Modular< uint32_t, Compute > >, 510
ModeTraits< Givaro::Modular< uint8_t, Compute > >, 510
ModeTraits< Givaro::ModularBalanced< Element > >, 511
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, 511
ModeTraits< Givaro::ModularBalanced< int16_t > >, 512
ModeTraits< Givaro::ModularBalanced< int32_t > >, 512
ModeTraits< Givaro::ModularBalanced< int8_t > >, 512
ModeTraits< Givaro::Montgomery< T > >, 513
ModeTraits< Givaro::ZRing< double > >, 513
ModeTraits< Givaro::ZRing< float > >, 513
ModeTraits< Givaro::ZRing< Givaro::Integer > >, 514
need_field_characteristic< Field >, 515
need_field_characteristic< Givaro::Modular< Field > >, 515
need_field_characteristic< Givaro::ModularBalanced< Field > >, 515
SimdChooser< T, false, b >, 731
SimdChooser< T, true, false >, 732
SimdChooser< T, true, true >, 732
width< double >, 768
width< float >, 769
width< T >, 768
vand
ScalFunctions< Element >, 551
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 620
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, true, 4 >, 692
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 729
Simd512i_base, 731
vandnot
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 571
Simd128_impl< true, true, false, 4 >, 581
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 620
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, true, 4 >, 692
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 729
Simd512i_base, 731
VEC_ADD
FFLAS::vectorised, 279
VEC_SUB
FFLAS::vectorised, 279
vect_size
FieldSimd< _Field >, 452
NoSimd< T >, 517
Simd128_impl< true, true, false, 2 >, 571
Simd128_impl< true, true, false, 4 >, 581
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 619
Simd256_impl< true, false, true, 8 >, 628
Simd256_impl< true, true, false, 2 >, 638
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 692
Simd256_impl< true, true, true, 8 >, 701
Simd512_impl< true, false, true, 8 >, 709
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 730
TestOneMethod< Simd >, 762
vect_t

FieldSimd< _Field >, 448
NoSimd< T >, 516
Simd128_impl< true, true, false, 2 >, 563
Simd128_impl< true, true, false, 4 >, 573
Simd128_impl< true, true, false, 8 >, 583
Simd128_impl< true, true, true, 2 >, 593
Simd128_impl< true, true, true, 4 >, 602
Simd128_impl< true, true, true, 8 >, 611
simd128_int64.inl, 873
Simd128i_base, 619
Simd256_impl< true, false, true, 8 >, 622
Simd256_impl< true, true, false, 2 >, 631
Simd256_impl< true, true, false, 4 >, 642
Simd256_impl< true, true, false, 8 >, 659
Simd256_impl< true, true, true, 2 >, 668
Simd256_impl< true, true, true, 4 >, 679
Simd256_impl< true, true, true, 8 >, 695
simd256_int64.inl, 876
Simd256i_base, 702
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 712
Simd512_impl< true, true, true, 8 >, 722
simd512_int64.inl, 878
Simd512i_base, 730
TestOneMethod< Simd >, 761
vectElt
ScalFunctions< Element >, 551
TestOneMethod< Simd >, 761
vector< T >, 767
elements, 768
vector< T >::const_iterator, 423
vector< T >::const_reverse_iterator, 424
vector< T >::iterator, 490
vector< T >::reverse_iterator, 523
verification_PLUQ
benchmark-pluq.C, 799
verifPLUQ
test-lu.C, 1087
VERSION
config.h, 818
vor
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 620
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, true, 4 >, 692
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 729
Simd512i_base, 730
vxor
ScalFunctions< Element >, 552
Simd128_impl< true, true, false, 2 >, 571
Simd128_impl< true, true, false, 4 >, 581
Simd128_impl< true, true, false, 8 >, 591
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 609
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 620
Simd256_impl< true, false, true, 8 >, 627
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, true, 4 >, 692
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 729
Simd512i_base, 730

WAIT
parallel.h, 1030
width< double >, 768
value, 768
width< float >, 769
value, 769
width< T >, 768
value, 768
Winograd, 769
FFLAS::BLAS3, 193
winograd.C, 776
balanced, 777
DOUBLE_TO_FLOAT_CROSSOVER, 776
GFOPS, 776
main, 777
Winograd_L_S
FFLAS::BLAS3, 196
Winograd_LR_S
FFLAS::BLAS3, 196
Winograd_R_S
FFLAS::BLAS3, 197
WinogradAcc_2_24
FFLAS::BLAS3, 194
WinogradAcc_2_27
FFLAS::BLAS3, 194
WinogradAcc_3_21
FFLAS::BLAS3, 194
WinogradAcc_3_23
FFLAS::BLAS3, 193
WinogradAcc_L_S
FFLAS::BLAS3, 196
WinogradAcc_LR
FFLAS::BLAS3, 195
WinogradAcc_R_S
FFLAS::BLAS3, 195
WinogradCalc
FFLAS::Protected, 217
WinogradPar, 769
WinogradSteps
FFLAS::Protected, 216
WinogradThreshold
FFLAS::Protected, 215
WinoPar
FFLAS::BLAS3, 193
WINOTHRESHOLD
fflas.h, 824

WRITE
parallel.h, 1031
write
RNSInteger< RNS >, 541
RNSIntegerMod< RNS >, 547
write_field
Matio.h, 1046
write_matrix
benchmark-fgemv-mp.C, 788
RNSIntegerMod< RNS >, 547
write_matrix_long
RNSIntegerMod< RNS >, 547
writeCommandString
FFLAS, 185
writeDebugData
TestOneMethod< Simd >, 762
writeDnsFormat
FFLAS, 152
WriteMatrix
FFLAS, 185, 187
WritePermutation
FFLAS, 187
writeResultLine
TestOneMethod< Simd >, 762

zero
FFLAS, 76
FieldSimd< _Field >, 450
RNSInteger< RNS >, 542
RNSIntegerMod< RNS >, 549
ScalFunctions< Element >, 551
Simd128_impl< true, true, false, 2 >, 570
Simd128_impl< true, true, false, 4 >, 580
Simd128_impl< true, true, false, 8 >, 590
Simd128_impl< true, true, true, 2 >, 600
Simd128_impl< true, true, true, 4 >, 608
Simd128_impl< true, true, true, 8 >, 618
Simd128i_base, 619
Simd256_impl< true, false, true, 8 >, 623
Simd256_impl< true, true, false, 2 >, 638
Simd256_impl< true, true, false, 4 >, 656
Simd256_impl< true, true, false, 8 >, 666
Simd256_impl< true, true, true, 2 >, 675
Simd256_impl< true, true, true, 4 >, 692
Simd256_impl< true, true, true, 8 >, 701
Simd256i_base, 702
Simd512_impl< true, false, true, 8 >, 704
Simd512_impl< true, true, false, 8 >, 720
Simd512_impl< true, true, true, 8 >, 729
Simd512i_base, 730
ZOSparseMatrix
FFLAS, 72