

nmpp

Создано системой Doxygen 1.8.13



# Оглавление

|         |                              |    |
|---------|------------------------------|----|
| 1       | Ошибки                       | 1  |
| 2       | Алфавитный указатель групп   | 3  |
| 2.1     | nmppr                        | 3  |
| 3       | Иерархический список классов | 11 |
| 3.1     | Иерархия классов             | 11 |
| 4       | Алфавитный указатель классов | 13 |
| 4.1     | Классы                       | 13 |
| 5       | Список файлов                | 15 |
| 5.1     | Файлы                        | 15 |
| 6       | Группы                       | 25 |
| 6.1     | FFTFwdInitAlloc              | 25 |
| 6.1.1   | Подробное описание           | 25 |
| 6.2     | FFTInvInitAlloc              | 26 |
| 6.2.1   | Подробное описание           | 26 |
| 6.3     | DFT-8                        | 27 |
| 6.3.1   | Подробное описание           | 27 |
| 6.3.2   | Функции                      | 27 |
| 6.3.2.1 | nmppsDFT8Fwd_32fcr()         | 27 |
| 6.4     | FFT-16                       | 28 |
| 6.4.1   | Подробное описание           | 28 |
| 6.5     | FFT-32                       | 29 |

|          |                                   |    |
|----------|-----------------------------------|----|
| 6.5.1    | Подробное описание . . . . .      | 29 |
| 6.5.2    | Функции . . . . .                 | 29 |
| 6.5.2.1  | nmppsFFT32Fwd_32fcr() . . . . .   | 29 |
| 6.6      | FFT-64 . . . . .                  | 32 |
| 6.6.1    | Подробное описание . . . . .      | 32 |
| 6.6.2    | Функции . . . . .                 | 32 |
| 6.6.2.1  | nmppsFFT64Fwd_32fcr() . . . . .   | 32 |
| 6.7      | FFT-128 . . . . .                 | 35 |
| 6.7.1    | Подробное описание . . . . .      | 35 |
| 6.7.2    | Функции . . . . .                 | 35 |
| 6.7.2.1  | nmppsFFT128Fwd_32fcr() . . . . .  | 35 |
| 6.8      | FFT-256 . . . . .                 | 38 |
| 6.8.1    | Подробное описание . . . . .      | 38 |
| 6.8.2    | Функции . . . . .                 | 38 |
| 6.8.2.1  | nmppsFFT256Fwd_32fcr() . . . . .  | 38 |
| 6.9      | FFT-512 . . . . .                 | 41 |
| 6.9.1    | Подробное описание . . . . .      | 41 |
| 6.9.2    | Функции . . . . .                 | 41 |
| 6.9.2.1  | nmppsFFT512Fwd_32fcr() . . . . .  | 41 |
| 6.10     | FFT-1024 . . . . .                | 44 |
| 6.10.1   | Подробное описание . . . . .      | 44 |
| 6.10.2   | Функции . . . . .                 | 44 |
| 6.10.2.1 | nmppsFFT1024Fwd_32fcr() . . . . . | 44 |
| 6.11     | FFT-2048 . . . . .                | 47 |
| 6.11.1   | Подробное описание . . . . .      | 47 |
| 6.11.2   | Функции . . . . .                 | 47 |
| 6.11.2.1 | nmppsFFT2048Fwd_32fcr() . . . . . | 47 |
| 6.12     | FFT-4096 . . . . .                | 50 |
| 6.12.1   | Подробное описание . . . . .      | 50 |
| 6.12.2   | Функции . . . . .                 | 50 |

|                                  |    |
|----------------------------------|----|
| 6.12.2.1 nmppsFFT4096Fwd_32fcr() | 50 |
| 6.13 IDFT-8                      | 52 |
| 6.13.1 Подробное описание        | 52 |
| 6.14 IFFT-16                     | 53 |
| 6.14.1 Подробное описание        | 53 |
| 6.15 IFFT-32                     | 54 |
| 6.15.1 Подробное описание        | 54 |
| 6.15.2 Функции                   | 54 |
| 6.15.2.1 nmppsFFT32Inv_32fcr()   | 54 |
| 6.16 IFFT-64                     | 57 |
| 6.16.1 Подробное описание        | 57 |
| 6.16.2 Функции                   | 57 |
| 6.16.2.1 nmppsFFT64Inv_32fcr()   | 57 |
| 6.17 IFFT-128                    | 60 |
| 6.17.1 Подробное описание        | 60 |
| 6.17.2 Функции                   | 60 |
| 6.17.2.1 nmppsFFT128Inv_32fcr()  | 60 |
| 6.18 IFFT-256                    | 63 |
| 6.18.1 Подробное описание        | 63 |
| 6.18.2 Функции                   | 63 |
| 6.18.2.1 nmppsFFT256Inv_32fcr()  | 63 |
| 6.19 IFFT-512                    | 66 |
| 6.19.1 Подробное описание        | 66 |
| 6.19.2 Функции                   | 66 |
| 6.19.2.1 nmppsFFT512Inv_32fcr()  | 66 |
| 6.20 IFFT-1024                   | 69 |
| 6.20.1 Подробное описание        | 69 |
| 6.20.2 Функции                   | 69 |
| 6.20.2.1 nmppsFFT1024Inv_32fcr() | 69 |
| 6.21 IFFT-2048                   | 72 |

|          |                                   |    |
|----------|-----------------------------------|----|
| 6.21.1   | Подробное описание . . . . .      | 72 |
| 6.21.2   | Функции . . . . .                 | 72 |
| 6.21.2.1 | nmppsFFT2048Inv_32fcr() . . . . . | 72 |
| 6.22     | IFFT-4096 . . . . .               | 75 |
| 6.22.1   | Подробное описание . . . . .      | 75 |
| 6.22.2   | Функции . . . . .                 | 75 |
| 6.22.2.1 | nmppsFFT4096Inv_32fcr() . . . . . | 75 |
| 6.23     | FFT-Common . . . . .              | 77 |
| 6.23.1   | Подробное описание . . . . .      | 77 |
| 6.24     | IFFT-Common . . . . .             | 78 |
| 6.25     | nmppsMalloc . . . . .             | 79 |
| 6.25.1   | Подробное описание . . . . .      | 79 |
| 6.26     | nmppsFree . . . . .               | 80 |
| 6.26.1   | Подробное описание . . . . .      | 80 |
| 6.27     | BLASS-LEVEL3 . . . . .            | 81 |
| 6.27.1   | Подробное описание . . . . .      | 81 |
| 6.28     | BLASS-LEVEL1 . . . . .            | 82 |
| 6.28.1   | Подробное описание . . . . .      | 83 |
| 6.29     | nmblas_dasum . . . . .            | 84 |
| 6.29.1   | Подробное описание . . . . .      | 84 |
| 6.29.2   | Функции . . . . .                 | 84 |
| 6.29.2.1 | nmblas_dasum() . . . . .          | 84 |
| 6.30     | nmblas_daxpy . . . . .            | 86 |
| 6.30.1   | Подробное описание . . . . .      | 86 |
| 6.30.2   | Функции . . . . .                 | 86 |
| 6.30.2.1 | nmblas_daxpy() . . . . .          | 86 |
| 6.31     | nmblas_dcopy . . . . .            | 88 |
| 6.31.1   | Подробное описание . . . . .      | 88 |
| 6.31.2   | Функции . . . . .                 | 88 |
| 6.31.2.1 | nmblas_dcopy() . . . . .          | 88 |

|          |                             |     |
|----------|-----------------------------|-----|
| 6.32     | <code>nmblas_ddot</code>    | 90  |
| 6.32.1   | Подробное описание          | 90  |
| 6.32.2   | Функции                     | 90  |
| 6.32.2.1 | <code>nmblas_ddot()</code>  | 90  |
| 6.33     | <code>nmblas_dnrm2</code>   | 92  |
| 6.33.1   | Подробное описание          | 92  |
| 6.33.2   | Функции                     | 92  |
| 6.33.2.1 | <code>nmblas_dnrm2()</code> | 92  |
| 6.34     | <code>nmblas_drot</code>    | 94  |
| 6.34.1   | Подробное описание          | 94  |
| 6.34.2   | Функции                     | 94  |
| 6.34.2.1 | <code>nmblas_drot()</code>  | 94  |
| 6.35     | <code>nmblas_drotg</code>   | 96  |
| 6.35.1   | Подробное описание          | 96  |
| 6.36     | <code>nmblas_drotm</code>   | 97  |
| 6.36.1   | Подробное описание          | 97  |
| 6.37     | <code>nmblas_dscal</code>   | 98  |
| 6.37.1   | Подробное описание          | 98  |
| 6.37.2   | Функции                     | 98  |
| 6.37.2.1 | <code>nmblas_dscal()</code> | 98  |
| 6.38     | <code>nmblas_dsdot</code>   | 100 |
| 6.38.1   | Подробное описание          | 100 |
| 6.38.2   | Функции                     | 100 |
| 6.38.2.1 | <code>nmblas_dsdot()</code> | 100 |
| 6.39     | <code>nmblas_dswap</code>   | 102 |
| 6.39.1   | Подробное описание          | 102 |
| 6.40     | <code>nmblas_dznrm2</code>  | 103 |
| 6.40.1   | Подробное описание          | 103 |
| 6.41     | <code>nmblas_idamax</code>  | 104 |
| 6.41.1   | Подробное описание          | 104 |

|          |                              |     |
|----------|------------------------------|-----|
| 6.41.2   | Функции                      | 104 |
| 6.41.2.1 | <code>nmblas_idamax()</code> | 104 |
| 6.42     | <code>nmblas_isamax</code>   | 106 |
| 6.42.1   | Подробное описание           | 106 |
| 6.42.2   | Функции                      | 106 |
| 6.42.2.1 | <code>nmblas_isamax()</code> | 106 |
| 6.43     | <code>nmblas_sasum</code>    | 108 |
| 6.43.1   | Подробное описание           | 108 |
| 6.43.2   | Функции                      | 108 |
| 6.43.2.1 | <code>nmblas_sasum()</code>  | 108 |
| 6.44     | <code>nmblas_saxpy</code>    | 110 |
| 6.44.1   | Подробное описание           | 110 |
| 6.44.2   | Функции                      | 110 |
| 6.44.2.1 | <code>nmblas_saxpy()</code>  | 110 |
| 6.45     | <code>nmblas_scnrm2</code>   | 112 |
| 6.45.1   | Подробное описание           | 112 |
| 6.45.2   | Функции                      | 112 |
| 6.45.2.1 | <code>nmblas_scnrm2()</code> | 112 |
| 6.46     | <code>nmblas_scopy</code>    | 114 |
| 6.46.1   | Подробное описание           | 114 |
| 6.46.2   | Функции                      | 114 |
| 6.46.2.1 | <code>nmblas_scopy()</code>  | 114 |
| 6.47     | <code>nmblas_sdot</code>     | 116 |
| 6.47.1   | Подробное описание           | 116 |
| 6.47.2   | Функции                      | 116 |
| 6.47.2.1 | <code>nmblas_sdot()</code>   | 116 |
| 6.48     | <code>nmblas_sdsdot</code>   | 118 |
| 6.48.1   | Подробное описание           | 118 |
| 6.49     | <code>nmblas_snrm2</code>    | 119 |
| 6.49.1   | Подробное описание           | 119 |



|          |                    |     |
|----------|--------------------|-----|
| 6.49.2   | Функции            | 119 |
| 6.49.2.1 | nmblas_snrm2()     | 119 |
| 6.50     | nmblas_srot        | 121 |
| 6.50.1   | Подробное описание | 121 |
| 6.51     | nmblas_srotm       | 122 |
| 6.51.1   | Подробное описание | 122 |
| 6.52     | nmblas_sscal       | 123 |
| 6.52.1   | Подробное описание | 123 |
| 6.52.2   | Функции            | 123 |
| 6.52.2.1 | nmblas_sscal()     | 123 |
| 6.53     | nmblas_sswap       | 125 |
| 6.53.1   | Подробное описание | 125 |
| 6.53.2   | Функции            | 125 |
| 6.53.2.1 | nmblas_sswap()     | 125 |
| 6.54     | nmblas_sgemv       | 127 |
| 6.54.1   | Подробное описание | 127 |
| 6.55     | nmblas_dgemv       | 129 |
| 6.55.1   | Подробное описание | 129 |
| 6.56     | nmppcDivC          | 131 |
| 6.56.1   | Подробное описание | 131 |
| 6.57     | nmppcProdC         | 132 |
| 6.57.1   | Подробное описание | 132 |
| 6.58     | nmppcFixExp32      | 133 |
| 6.58.1   | Подробное описание | 133 |
| 6.58.2   | Функции            | 133 |
| 6.58.2.1 | nmppcFixExp32()    | 133 |
| 6.59     | nmppcFixSinCos32   | 135 |
| 6.59.1   | Подробное описание | 135 |
| 6.59.2   | Функции            | 135 |
| 6.59.2.1 | nmppcFixSinCos32() | 135 |

|          |                                    |     |
|----------|------------------------------------|-----|
| 6.60     | <code>nmppcFixArcTan32</code>      | 137 |
| 6.60.1   | Подробное описание                 | 137 |
| 6.60.2   | Функции                            | 137 |
| 6.60.2.1 | <code>nmppcFixArcTan32()</code>    | 137 |
| 6.61     | <code>nmppcDoubleToFix32</code>    | 139 |
| 6.61.1   | Подробное описание                 | 139 |
| 6.62     | <code>nmppcFix32ToDouble</code>    | 140 |
| 6.62.1   | Подробное описание                 | 140 |
| 6.62.2   | Функции                            | 140 |
| 6.62.2.1 | <code>nmppcFix32ToDouble()</code>  | 140 |
| 6.63     | <code>nmppcFixSqrt32</code>        | 142 |
| 6.63.1   | Подробное описание                 | 142 |
| 6.64     | <code>nmppcFixMul32</code>         | 143 |
| 6.64.1   | Подробное описание                 | 143 |
| 6.64.2   | Функции                            | 143 |
| 6.64.2.1 | <code>nmppcFixDiv32()</code>       | 144 |
| 6.64.2.2 | <code>nmppcFixMul32()</code>       | 146 |
| 6.65     | <code>nmppcFixInv32</code>         | 148 |
| 6.65.1   | Подробное описание                 | 148 |
| 6.65.2   | Функции                            | 148 |
| 6.65.2.1 | <code>nmppcFixInv32()</code>       | 148 |
| 6.66     | <code>nmppcTblFixArcSin32</code>   | 150 |
| 6.66.1   | Подробное описание                 | 150 |
| 6.66.2   | Функции                            | 150 |
| 6.66.2.1 | <code>nmppcTblFixArcSin32()</code> | 150 |
| 6.67     | <code>nmppcTblFixArcCos32</code>   | 152 |
| 6.67.1   | Подробное описание                 | 152 |
| 6.67.2   | Функции                            | 152 |
| 6.67.2.1 | <code>nmppcTblFixArcCos32()</code> | 152 |
| 6.68     | <code>nmppcTblFixCos32</code>      | 154 |

|          |                                |     |
|----------|--------------------------------|-----|
| 6.68.1   | Подробное описание . . . . .   | 154 |
| 6.68.2   | Функции . . . . .              | 154 |
| 6.68.2.1 | nmppcTblFixCos32() . . . . .   | 154 |
| 6.69     | nmppcTblFixSin32 . . . . .     | 156 |
| 6.69.1   | Подробное описание . . . . .   | 156 |
| 6.69.2   | Функции . . . . .              | 156 |
| 6.69.2.1 | nmppcTblFixSin32() . . . . .   | 156 |
| 6.70     | nmppcFixDivMod32 . . . . .     | 158 |
| 6.70.1   | Подробное описание . . . . .   | 158 |
| 6.70.2   | Функции . . . . .              | 158 |
| 6.70.2.1 | nmppcFixDivMod32() . . . . .   | 158 |
| 6.71     | nmppcFixSqrt64 . . . . .       | 160 |
| 6.71.1   | Подробное описание . . . . .   | 160 |
| 6.72     | nmppcDoubleToFix64 . . . . .   | 161 |
| 6.72.1   | Подробное описание . . . . .   | 161 |
| 6.73     | nmppcFix64ToDouble . . . . .   | 162 |
| 6.73.1   | Подробное описание . . . . .   | 162 |
| 6.73.2   | Функции . . . . .              | 162 |
| 6.73.2.1 | nmppcFix64ToDouble() . . . . . | 162 |
| 6.74     | nmppcFixDiv64 . . . . .        | 164 |
| 6.74.1   | Подробное описание . . . . .   | 164 |
| 6.74.2   | Функции . . . . .              | 164 |
| 6.74.2.1 | nmppcFixDiv64() . . . . .      | 164 |
| 6.75     | nmppcFixSinCos64 . . . . .     | 166 |
| 6.75.1   | Подробное описание . . . . .   | 166 |
| 6.75.2   | Функции . . . . .              | 166 |
| 6.75.2.1 | nmppcFixSinCos64() . . . . .   | 166 |
| 6.76     | nmppcFixArcTan64 . . . . .     | 168 |
| 6.76.1   | Подробное описание . . . . .   | 168 |
| 6.76.2   | Функции . . . . .              | 168 |

|          |                                      |     |
|----------|--------------------------------------|-----|
| 6.76.2.1 | <code>nmppcFixArcTan64()</code>      | 168 |
| 6.77     | <code>nmppcFix64Exp01</code>         | 169 |
| 6.77.1   | Подробное описание                   | 169 |
| 6.77.2   | Функции                              | 169 |
| 6.77.2.1 | <code>nmppcFix64Exp01()</code>       | 169 |
| 6.78     | <code>nmppsRand</code>               | 171 |
| 6.78.1   | Подробное описание                   | 171 |
| 6.78.2   | Функции                              | 171 |
| 6.78.2.1 | <code>nmppcRandMinMaxDiv()</code>    | 171 |
| 6.79     | <code>nmppsSqrt</code>               | 173 |
| 6.79.1   | Подробное описание                   | 173 |
| 6.80     | Инициализация                        | 174 |
| 6.80.1   | Подробное описание                   | 174 |
| 6.81     | Integer operations                   | 175 |
| 6.81.1   | Подробное описание                   | 175 |
| 6.82     | Fix-point 64                         | 176 |
| 6.82.1   | Подробное описание                   | 176 |
| 6.83     | Fix-point 32                         | 177 |
| 6.83.1   | Подробное описание                   | 177 |
| 6.84     | Арифметические операции              | 178 |
| 6.84.1   | Подробное описание                   | 178 |
| 6.85     | Функции деинтерлейсинга              | 179 |
| 6.85.1   | Подробное описание                   | 179 |
| 6.85.2   | Функции                              | 179 |
| 6.85.2.1 | <code>nmppiDeinterlaceBlend()</code> | 179 |
| 6.85.2.2 | <code>nmppiDeinterlaceSplit()</code> | 179 |
| 6.86     | КИХ-фильтрация                       | 181 |
| 6.87     | Floodfill                            | 182 |
| 6.87.1   | Подробное описание                   | 182 |
| 6.87.2   | Функции                              | 182 |

|                                          |     |
|------------------------------------------|-----|
| 6.87.2.1 FloodFill8()                    | 183 |
| 6.88 nmppiConvert                        | 188 |
| 6.88.1 Подробное описание                | 188 |
| 6.89 nmppiRGB32ToGray                    | 189 |
| 6.89.1 Подробное описание                | 189 |
| 6.90 Переупорядочивание изображений      | 190 |
| 6.90.1 Подробное описание                | 190 |
| 6.91 Блочное переупорядочивание          | 191 |
| 6.91.1 Подробное описание                | 191 |
| 6.92 nmppiSplitIntoBlocks                | 192 |
| 6.92.1 Подробное описание                | 192 |
| 6.92.2 Функции                           | 192 |
| 6.92.2.1 nmppiSplitIntoBlocks8x8()       | 192 |
| 6.93 nmppiMergeFromBlocks                | 194 |
| 6.93.1 Подробное описание                | 194 |
| 6.93.2 Функции                           | 194 |
| 6.93.2.1 nmppiMergeFromBlocks8x8()       | 194 |
| 6.94 nmppiFree                           | 196 |
| 6.94.1 Подробное описание                | 196 |
| 6.95 nmppiRelease                        | 197 |
| 6.95.1 Подробное описание                | 197 |
| 6.96 Арифметические действия             | 198 |
| 6.97 Масочная фильтрация                 | 199 |
| 6.97.1 Подробное описание                | 199 |
| 6.98 Инициализация и копирование         | 200 |
| 6.98.1 Подробное описание                | 200 |
| 6.99 Функции поддержки                   | 201 |
| 6.99.1 Подробное описание                | 201 |
| 6.100 Функции графического вывода текста | 202 |
| 6.100.1 Подробное описание               | 202 |

|                                             |     |
|---------------------------------------------|-----|
| 6.100.2 Функции . . . . .                   | 202 |
| 6.100.2.1 hex2ascii() . . . . .             | 202 |
| 6.100.2.2 nmppiPrint8x15() . . . . .        | 202 |
| 6.101 Инициализация и копирование . . . . . | 204 |
| 6.101.1 Подробное описание . . . . .        | 204 |
| 6.102 nmppmCopyua . . . . .                 | 205 |
| 6.102.1 Подробное описание . . . . .        | 205 |
| 6.102.2 Функции . . . . .                   | 205 |
| 6.102.2.1 nmppmCopyua_16s() . . . . .       | 206 |
| 6.102.2.2 nmppmCopyua_32s() . . . . .       | 208 |
| 6.102.2.3 nmppmCopyua_8s() . . . . .        | 210 |
| 6.103 MTR_Copyua . . . . .                  | 213 |
| 6.103.1 Подробное описание . . . . .        | 213 |
| 6.103.2 Функции . . . . .                   | 214 |
| 6.103.2.1 nmppmCopyua_8s() . . . . .        | 214 |
| 6.104 MTR_Copy . . . . .                    | 216 |
| 6.104.1 Подробное описание . . . . .        | 216 |
| 6.104.2 Функции . . . . .                   | 217 |
| 6.104.2.1 nmppmCopy_1() . . . . .           | 217 |
| 6.104.2.2 nmppmCopy_16s() . . . . .         | 218 |
| 6.104.2.3 nmppmCopy_2s() . . . . .          | 219 |
| 6.104.2.4 nmppmCopy_32s() . . . . .         | 220 |
| 6.104.2.5 nmppmCopy_4s() . . . . .          | 222 |
| 6.104.2.6 nmppmCopy_64s() . . . . .         | 223 |
| 6.104.2.7 nmppmCopy_8s() . . . . .          | 224 |
| 6.105 MTR_fpResolve_Gauss . . . . .         | 226 |
| 6.105.1 Подробное описание . . . . .        | 226 |
| 6.106 MTR_fpResolve_PivotGauss . . . . .    | 227 |
| 6.106.1 Подробное описание . . . . .        | 227 |
| 6.107 nmppmLUDecomp . . . . .               | 228 |

|                                          |     |
|------------------------------------------|-----|
| 6.107.1 Подробное описание . . . . .     | 228 |
| 6.108nmppmLUResolve . . . . .            | 229 |
| 6.108.1 Подробное описание . . . . .     | 229 |
| 6.109nmppmMul_mm . . . . .               | 230 |
| 6.109.1 Подробное описание . . . . .     | 230 |
| 6.109.2 Функции . . . . .                | 231 |
| 6.109.2.1 nmppmMul_mm_16s32s() . . . . . | 231 |
| 6.109.2.2 nmppmMul_mm_16s64s() . . . . . | 233 |
| 6.109.2.3 nmppmMul_mm_2s64s() . . . . .  | 235 |
| 6.109.2.4 nmppmMul_mm_32s32s() . . . . . | 237 |
| 6.109.2.5 nmppmMul_mm_32s64s() . . . . . | 239 |
| 6.109.2.6 nmppmMul_mm_4s64s() . . . . .  | 242 |
| 6.109.2.7 nmppmMul_mm_64s64s() . . . . . | 244 |
| 6.109.2.8 nmppmMul_mm_8s64s() . . . . .  | 246 |
| 6.110nmppmMul_mv_ . . . . .              | 249 |
| 6.110.1 Подробное описание . . . . .     | 249 |
| 6.110.2 Функции . . . . .                | 249 |
| 6.110.2.1 nmppmMul_mv_16s64s() . . . . . | 250 |
| 6.110.2.2 nmppmMul_mv_32s64s() . . . . . | 251 |
| 6.110.2.3 nmppmMul_mv_8s64s() . . . . .  | 252 |
| 6.111nmppmMul_mv_8xH . . . . .           | 254 |
| 6.111.1 Подробное описание . . . . .     | 254 |
| 6.112nmppmMul_mv__AddC . . . . .         | 255 |
| 6.112.1 Подробное описание . . . . .     | 255 |
| 6.113MTR_ProdUnitV . . . . .             | 256 |
| 6.113.1 Подробное описание . . . . .     | 256 |
| 6.114MTR_Malloc . . . . .                | 257 |
| 6.115MTR_Free . . . . .                  | 258 |
| 6.115.1 Подробное описание . . . . .     | 258 |
| 6.116MTR_Addr . . . . .                  | 259 |

|                                       |     |
|---------------------------------------|-----|
| 6.116.1 Подробное описание . . . . .  | 259 |
| 6.116.2 Функции . . . . .             | 259 |
| 6.116.2.1 MTR_Addr_1() . . . . .      | 260 |
| 6.116.2.2 MTR_Addr_16s() . . . . .    | 260 |
| 6.116.2.3 MTR_Addr_16u() . . . . .    | 261 |
| 6.116.2.4 MTR_Addr_2s() . . . . .     | 261 |
| 6.116.2.5 MTR_Addr_2u() . . . . .     | 262 |
| 6.116.2.6 MTR_Addr_32s() . . . . .    | 262 |
| 6.116.2.7 MTR_Addr_32u() . . . . .    | 263 |
| 6.116.2.8 MTR_Addr_4s() . . . . .     | 264 |
| 6.116.2.9 MTR_Addr_4u() . . . . .     | 264 |
| 6.116.2.10 MTR_Addr_64s() . . . . .   | 265 |
| 6.116.2.11 MTR_Addr_64u() . . . . .   | 265 |
| 6.116.2.12 MTR_Addr_8s() . . . . .    | 266 |
| 6.116.2.13 MTR_Addr_8u() . . . . .    | 267 |
| 6.117 MTR_SetVal . . . . .            | 268 |
| 6.117.1 Подробное описание . . . . .  | 268 |
| 6.117.2 Функции . . . . .             | 268 |
| 6.117.2.1 MTR_SetVal_1() . . . . .    | 269 |
| 6.117.2.2 MTR_SetVal_16s() . . . . .  | 269 |
| 6.117.2.3 MTR_SetVal_16u() . . . . .  | 270 |
| 6.117.2.4 MTR_SetVal_2s() . . . . .   | 270 |
| 6.117.2.5 MTR_SetVal_2u() . . . . .   | 271 |
| 6.117.2.6 MTR_SetVal_32s() . . . . .  | 272 |
| 6.117.2.7 MTR_SetVal_32u() . . . . .  | 272 |
| 6.117.2.8 MTR_SetVal_4s() . . . . .   | 273 |
| 6.117.2.9 MTR_SetVal_4u() . . . . .   | 273 |
| 6.117.2.10 MTR_SetVal_64s() . . . . . | 274 |
| 6.117.2.11 MTR_SetVal_64u() . . . . . | 275 |
| 6.117.2.12 MTR_SetVal_8s() . . . . .  | 275 |



|                                  |     |
|----------------------------------|-----|
| 6.117.2.13MTR_SetVal_8u()        | 276 |
| 6.118MTR_GetVal                  | 277 |
| 6.118.1 Подробное описание       | 277 |
| 6.118.2 Функции                  | 277 |
| 6.118.2.1 MTR_GetVal_1()         | 278 |
| 6.118.2.2 MTR_GetVal_16s()       | 278 |
| 6.118.2.3 MTR_GetVal_16u()       | 279 |
| 6.118.2.4 MTR_GetVal_2s()        | 279 |
| 6.118.2.5 MTR_GetVal_2u()        | 280 |
| 6.118.2.6 MTR_GetVal_32s()       | 281 |
| 6.118.2.7 MTR_GetVal_32u()       | 281 |
| 6.118.2.8 MTR_GetVal_4s()        | 282 |
| 6.118.2.9 MTR_GetVal_4u()        | 282 |
| 6.118.2.10MTR_GetVal_64s()       | 283 |
| 6.118.2.11MTR_GetVal_64u()       | 284 |
| 6.118.2.12MTR_GetVal_8s()        | 284 |
| 6.118.2.13MTR_GetVal_8u()        | 285 |
| 6.119Функции поддержки           | 286 |
| 6.119.1 Подробное описание       | 286 |
| 6.120Векторно-матричные операции | 287 |
| 6.120.1 Подробное описание       | 287 |
| 6.121Обращение матрицы           | 288 |
| 6.121.1 Подробное описание       | 288 |
| 6.122FFT-256                     | 289 |
| 6.122.1 Подробное описание       | 289 |
| 6.122.2 Функции                  | 289 |
| 6.122.2.1 FFT_Fwd256()           | 289 |
| 6.123IFFT-256                    | 291 |
| 6.123.1 Подробное описание       | 291 |
| 6.123.2 Функции                  | 291 |

|                            |     |
|----------------------------|-----|
| 6.123.2.1 FFT_Inv256()     | 291 |
| 6.124FFT-512               | 294 |
| 6.124.1 Подробное описание | 294 |
| 6.124.2 Функции            | 294 |
| 6.124.2.1 FFT_Fwd512()     | 294 |
| 6.125IFFT-512              | 296 |
| 6.125.1 Подробное описание | 296 |
| 6.125.2 Функции            | 296 |
| 6.125.2.1 FFT_Inv512()     | 296 |
| 6.126FFT-1024              | 299 |
| 6.126.1 Подробное описание | 299 |
| 6.126.2 Функции            | 299 |
| 6.126.2.1 FFT_Fwd1024()    | 299 |
| 6.127IFFT-1024             | 301 |
| 6.127.1 Подробное описание | 301 |
| 6.127.2 Функции            | 301 |
| 6.127.2.1 FFT_Inv1024()    | 301 |
| 6.128FFT-2048              | 304 |
| 6.128.1 Подробное описание | 304 |
| 6.128.2 Функции            | 304 |
| 6.128.2.1 FFT_Fwd2048()    | 304 |
| 6.129IFFT-2048             | 306 |
| 6.129.1 Подробное описание | 306 |
| 6.129.2 Функции            | 306 |
| 6.129.2.1 FFT_Inv2048()    | 306 |
| 6.130FFT-4096              | 309 |
| 6.130.1 Подробное описание | 309 |
| 6.130.2 Функции            | 309 |
| 6.130.2.1 FFT_Fwd4096()    | 309 |
| 6.131IFFT-4096             | 311 |

|                                             |     |
|---------------------------------------------|-----|
| 6.131.1 Подробное описание . . . . .        | 311 |
| 6.131.2 Функции . . . . .                   | 311 |
| 6.131.2.1 FFT_Inv4096() . . . . .           | 311 |
| 6.132FFT-8192 . . . . .                     | 313 |
| 6.132.1 Подробное описание . . . . .        | 313 |
| 6.132.2 Функции . . . . .                   | 313 |
| 6.132.2.1 FFT_Fwd8192() . . . . .           | 313 |
| 6.133IFFT-8192 . . . . .                    | 315 |
| 6.133.1 Подробное описание . . . . .        | 315 |
| 6.133.2 Функции . . . . .                   | 315 |
| 6.133.2.1 FFT_Inv8192() . . . . .           | 315 |
| 6.134Свертка . . . . .                      | 317 |
| 6.134.1 Подробное описание . . . . .        | 317 |
| 6.135Масочная фильтрация . . . . .          | 318 |
| 6.135.1 Подробное описание . . . . .        | 318 |
| 6.136Изменение размеров . . . . .           | 319 |
| 6.136.1 Подробное описание . . . . .        | 319 |
| 6.137Быстрое преобразование Фурье . . . . . | 320 |
| 6.137.1 Подробное описание . . . . .        | 320 |
| 6.138Быстрое преобразование Фурье . . . . . | 321 |
| 6.138.1 Подробное описание . . . . .        | 322 |
| 6.138.2 Функции . . . . .                   | 322 |
| 6.138.2.1 nmppsFFTFree_32fcr() . . . . .    | 322 |
| 6.139SIG_XCorr . . . . .                    | 324 |
| 6.139.1 Подробное описание . . . . .        | 324 |
| 6.140SIG_Median3 . . . . .                  | 326 |
| 6.140.1 Подробное описание . . . . .        | 326 |
| 6.141КИХ-фильтрация . . . . .               | 327 |
| 6.141.1 Подробное описание . . . . .        | 327 |
| 6.142nmppsFIR_Xs . . . . .                  | 328 |

|                                        |     |
|----------------------------------------|-----|
| 6.142.1 Подробное описание . . . . .   | 328 |
| 6.143nmppsFIRInit_Xs . . . . .         | 329 |
| 6.143.1 Подробное описание . . . . .   | 329 |
| 6.144nmppsFIRInitAlloc_Xs . . . . .    | 330 |
| 6.144.1 Подробное описание . . . . .   | 330 |
| 6.145nmppsFIRGetStateSize_Xs . . . . . | 331 |
| 6.145.1 Подробное описание . . . . .   | 331 |
| 6.146nmppsFIRFree . . . . .            | 332 |
| 6.146.1 Подробное описание . . . . .   | 332 |
| 6.147SIG_ResampleDown2 . . . . .       | 333 |
| 6.147.1 Подробное описание . . . . .   | 333 |
| 6.148SIG_ResampleUp3Down2 . . . . .    | 334 |
| 6.148.1 Подробное описание . . . . .   | 334 |
| 6.149SIG_CreateResample . . . . .      | 335 |
| 6.149.1 Подробное описание . . . . .   | 335 |
| 6.150SIG_SetResample . . . . .         | 336 |
| 6.150.1 Подробное описание . . . . .   | 336 |
| 6.151SIG_Resample_perf . . . . .       | 337 |
| 6.151.1 Подробное описание . . . . .   | 337 |
| 6.152Типы векторных данных . . . . .   | 338 |
| 6.152.1 Подробное описание . . . . .   | 339 |
| 6.152.2 Типы . . . . .                 | 340 |
| 6.152.2.1 nm1 . . . . .                | 340 |
| 6.152.2.2 nm16s . . . . .              | 340 |
| 6.152.2.3 nm16s15b . . . . .           | 341 |
| 6.152.2.4 nm16u . . . . .              | 341 |
| 6.152.2.5 nm16u15b . . . . .           | 341 |
| 6.152.2.6 nm2s . . . . .               | 342 |
| 6.152.2.7 nm2u . . . . .               | 342 |
| 6.152.2.8 nm32s . . . . .              | 342 |

|                              |     |
|------------------------------|-----|
| 6.152.2.9 nm32s30b . . . . . | 343 |
| 6.152.2.10nm32s31b . . . . . | 343 |
| 6.152.2.11nm32u . . . . .    | 343 |
| 6.152.2.12nm32u31b . . . . . | 344 |
| 6.152.2.13nm4s . . . . .     | 344 |
| 6.152.2.14nm4u . . . . .     | 344 |
| 6.152.2.15nm4u3b . . . . .   | 345 |
| 6.152.2.16nm64s . . . . .    | 345 |
| 6.152.2.17nm64s63b . . . . . | 345 |
| 6.152.2.18nm64u . . . . .    | 346 |
| 6.152.2.19nm8s . . . . .     | 346 |
| 6.152.2.20nm8s7b . . . . .   | 346 |
| 6.152.2.21nm8u . . . . .     | 347 |
| 6.152.2.22nm8u7b . . . . .   | 347 |
| 6.152.2.2316nm16s . . . . .  | 347 |
| 6.152.2.2416nm16u . . . . .  | 347 |
| 6.152.2.2516nm32s . . . . .  | 347 |
| 6.152.2.2616nm32u . . . . .  | 348 |
| 6.152.2.2716nm4b3u . . . . . | 348 |
| 6.152.2.2816nm4u . . . . .   | 348 |
| 6.152.2.2916nm8s . . . . .   | 348 |
| 6.152.2.3016nm8s7b . . . . . | 348 |
| 6.152.2.3116nm8u . . . . .   | 348 |
| 6.152.2.322nm32s . . . . .   | 349 |
| 6.152.2.332nm32u . . . . .   | 349 |
| 6.152.2.344nm16s . . . . .   | 349 |
| 6.152.2.354nm16u . . . . .   | 349 |
| 6.152.2.364nm32s . . . . .   | 349 |
| 6.152.2.374nm32u . . . . .   | 349 |
| 6.152.2.384nm8u . . . . .    | 349 |

|            |                       |     |
|------------|-----------------------|-----|
| 6.152.2.39 | 8nm16s                | 349 |
| 6.152.2.40 | 8nm16u                | 350 |
| 6.152.2.41 | 8nm32s                | 350 |
| 6.152.2.42 | 8nm32u                | 350 |
| 6.152.2.43 | 8nm8s                 | 350 |
| 6.152.2.44 | 8nm8u                 | 350 |
| 6.153      | Типы скалярных данных | 351 |
| 6.153.1    | Подробное описание    | 351 |
| 6.153.2    | Типы                  | 351 |
| 6.153.2.1  | int15b                | 352 |
| 6.153.2.2  | int16b                | 352 |
| 6.153.2.3  | int1b                 | 352 |
| 6.153.2.4  | int2b                 | 352 |
| 6.153.2.5  | int30b                | 353 |
| 6.153.2.6  | int31b                | 353 |
| 6.153.2.7  | int32b                | 353 |
| 6.153.2.8  | int3b                 | 353 |
| 6.153.2.9  | int4b                 | 354 |
| 6.153.2.10 | nt63b                 | 354 |
| 6.153.2.11 | nt64b                 | 354 |
| 6.153.2.12 | nt7b                  | 354 |
| 6.153.2.13 | nt8b                  | 355 |
| 6.153.2.14 | int15b                | 355 |
| 6.153.2.15 | int16b                | 355 |
| 6.153.2.16 | int1b                 | 355 |
| 6.153.2.17 | int2b                 | 356 |
| 6.153.2.18 | int31b                | 356 |
| 6.153.2.19 | int32b                | 356 |
| 6.153.2.20 | int3b                 | 356 |
| 6.153.2.21 | int4b                 | 357 |

|            |                                               |     |
|------------|-----------------------------------------------|-----|
| 6.153.2.22 | uint63b . . . . .                             | 357 |
| 6.153.2.23 | uint64b . . . . .                             | 357 |
| 6.153.2.24 | uint7b . . . . .                              | 357 |
| 6.153.2.25 | uint8b . . . . .                              | 357 |
| 6.154      | Функции поддержки . . . . .                   | 358 |
| 6.154.1    | Подробное описание . . . . .                  | 358 |
| 6.155      | Инициализация и копирование . . . . .         | 359 |
| 6.155.1    | Подробное описание . . . . .                  | 359 |
| 6.156      | Инициализация и копирование . . . . .         | 360 |
| 6.156.1    | Подробное описание . . . . .                  | 360 |
| 6.157      | Арифметические операции . . . . .             | 361 |
| 6.157.1    | Подробное описание . . . . .                  | 361 |
| 6.158      | Арифметические операции . . . . .             | 362 |
| 6.158.1    | Подробное описание . . . . .                  | 362 |
| 6.159      | Логические и бинарные операции . . . . .      | 363 |
| 6.159.1    | Подробное описание . . . . .                  | 363 |
| 6.160      | Операции сравнения . . . . .                  | 364 |
| 6.160.1    | Подробное описание . . . . .                  | 365 |
| 6.161      | Операции сравнения . . . . .                  | 366 |
| 6.161.1    | Подробное описание . . . . .                  | 366 |
| 6.162      | Переупорядочивание и сортировка . . . . .     | 367 |
| 6.162.1    | Подробное описание . . . . .                  | 367 |
| 6.163      | Элементарные математические функции . . . . . | 368 |
| 6.163.1    | Подробное описание . . . . .                  | 368 |
| 6.164      | Тригонометрические функции . . . . .          | 369 |
| 6.164.1    | Подробное описание . . . . .                  | 369 |
| 6.165      | Функция деления векторов . . . . .            | 370 |
| 6.165.1    | Подробное описание . . . . .                  | 370 |
| 6.166      | Экспонента . . . . .                          | 371 |
| 6.166.1    | Подробное описание . . . . .                  | 371 |

|           |                      |     |
|-----------|----------------------|-----|
| 6.167     | Натуральный логарифм | 372 |
| 6.167.1   | Подробное описание   | 372 |
| 6.168     | Возведение в степень | 373 |
| 6.168.1   | Подробное описание   | 373 |
| 6.169     | nmppsAbs             | 374 |
| 6.169.1   | Подробное описание   | 374 |
| 6.169.2   | Функции              | 374 |
| 6.169.2.1 | nmppsAbs_16s()       | 375 |
| 6.169.2.2 | nmppsAbs_32s()       | 376 |
| 6.169.2.3 | nmppsAbs_4s()        | 377 |
| 6.169.2.4 | nmppsAbs_64s()       | 378 |
| 6.169.2.5 | nmppsAbs_8s()        | 379 |
| 6.170     | nmppsAbs1            | 381 |
| 6.170.1   | Подробное описание   | 381 |
| 6.170.2   | Функции              | 381 |
| 6.170.2.1 | nmppsAbs1_16s()      | 382 |
| 6.170.2.2 | nmppsAbs1_32s()      | 383 |
| 6.170.2.3 | nmppsAbs1_64s()      | 384 |
| 6.170.2.4 | nmppsAbs1_8s()       | 385 |
| 6.171     | nmppsNeg             | 387 |
| 6.171.1   | Подробное описание   | 387 |
| 6.171.2   | Функции              | 387 |
| 6.171.2.1 | nmppsNeg_16s()       | 387 |
| 6.171.2.2 | nmppsNeg_32s()       | 388 |
| 6.171.2.3 | nmppsNeg_64s()       | 390 |
| 6.171.2.4 | nmppsNeg_8s()        | 391 |
| 6.172     | nmppsAddC            | 393 |
| 6.172.1   | Подробное описание   | 393 |
| 6.172.2   | Функции              | 393 |
| 6.172.2.1 | nmppsAddC_16s()      | 394 |



|                               |     |
|-------------------------------|-----|
| 6.172.2.2 nmppsAddC_32s()     | 395 |
| 6.172.2.3 nmppsAddC_64s()     | 396 |
| 6.172.2.4 nmppsAddC_8s()      | 397 |
| 6.173nmppsAddC_p              | 399 |
| 6.173.1 Подробное описание    | 399 |
| 6.173.2 Функции               | 399 |
| 6.173.2.1 nmppsAddC_p64s()    | 399 |
| 6.174nmppsAddC                | 401 |
| 6.174.1 Подробное описание    | 401 |
| 6.174.2 Функции               | 401 |
| 6.174.2.1 nmppsAddC_32fcr()   | 401 |
| 6.175nmppsAdd                 | 403 |
| 6.175.1 Подробное описание    | 403 |
| 6.175.2 Функции               | 403 |
| 6.175.2.1 nmppsAdd_16s()      | 404 |
| 6.175.2.2 nmppsAdd_32s()      | 405 |
| 6.175.2.3 nmppsAdd_4s()       | 406 |
| 6.175.2.4 nmppsAdd_64s()      | 407 |
| 6.175.2.5 nmppsAdd_8s()       | 409 |
| 6.176nmppsAdd                 | 411 |
| 6.176.1 Подробное описание    | 411 |
| 6.176.2 Функции               | 411 |
| 6.176.2.1 nmppsAdd_32f()      | 411 |
| 6.177nmppsAdd_AddC            | 414 |
| 6.177.1 Подробное описание    | 414 |
| 6.177.2 Функции               | 414 |
| 6.177.2.1 nmppsAdd_AddC_32s() | 414 |
| 6.178nmppsSubC                | 417 |
| 6.178.1 Подробное описание    | 417 |
| 6.178.2 Функции               | 417 |

|                              |     |
|------------------------------|-----|
| 6.178.2.1 nmppsSubC_16s()    | 418 |
| 6.178.2.2 nmppsSubC_32s()    | 419 |
| 6.178.2.3 nmppsSubC_4s()     | 420 |
| 6.178.2.4 nmppsSubC_64s()    | 421 |
| 6.178.2.5 nmppsSubC_8s()     | 422 |
| 6.179nmppsSubCRev            | 424 |
| 6.179.1 Подробное описание   | 424 |
| 6.179.2 Функции              | 424 |
| 6.179.2.1 nmppsSubCRev_16s() | 425 |
| 6.179.2.2 nmppsSubCRev_32s() | 426 |
| 6.179.2.3 nmppsSubCRev_64s() | 427 |
| 6.179.2.4 nmppsSubCRev_8s()  | 428 |
| 6.180nmppsSub                | 430 |
| 6.180.1 Подробное описание   | 430 |
| 6.180.2 Функции              | 430 |
| 6.180.2.1 nmppsSub_16s()     | 431 |
| 6.180.2.2 nmppsSub_32s()     | 432 |
| 6.180.2.3 nmppsSub_64s()     | 433 |
| 6.180.2.4 nmppsSub_8s()      | 434 |
| 6.181nmppsSub                | 436 |
| 6.181.1 Подробное описание   | 436 |
| 6.181.2 Функции              | 436 |
| 6.181.2.1 nmppsSub_32f()     | 436 |
| 6.182nmppsAbsDiff            | 439 |
| 6.182.1 Подробное описание   | 439 |
| 6.182.2 Функции              | 439 |
| 6.182.2.1 nmppsAbsDiff_16s() | 440 |
| 6.182.2.2 nmppsAbsDiff_32s() | 441 |
| 6.182.2.3 nmppsAbsDiff_64s() | 442 |
| 6.182.2.4 nmppsAbsDiff_8s()  | 443 |

|                                              |     |
|----------------------------------------------|-----|
| 6.183nmppsAbsDiff1 . . . . .                 | 445 |
| 6.183.1 Подробное описание . . . . .         | 445 |
| 6.183.2 Функции . . . . .                    | 445 |
| 6.183.2.1 nmppsAbsDiff1_8s() . . . . .       | 446 |
| 6.184nmppsMulC . . . . .                     | 448 |
| 6.184.1 Подробное описание . . . . .         | 448 |
| 6.184.2 Функции . . . . .                    | 448 |
| 6.184.2.1 nmppsMulC_16s() . . . . .          | 449 |
| 6.184.2.2 nmppsMulC_16s32s() . . . . .       | 450 |
| 6.184.2.3 nmppsMulC_32s() . . . . .          | 451 |
| 6.184.2.4 nmppsMulC_32s64s() . . . . .       | 452 |
| 6.184.2.5 nmppsMulC_64s() . . . . .          | 453 |
| 6.184.2.6 nmppsMulC_8s() . . . . .           | 454 |
| 6.184.2.7 nmppsMulC_8s16s() . . . . .        | 455 |
| 6.185nmppsMulC . . . . .                     | 457 |
| 6.185.1 Подробное описание . . . . .         | 457 |
| 6.185.2 Функции . . . . .                    | 457 |
| 6.185.2.1 nmppsMulC_32f() . . . . .          | 457 |
| 6.186nmppsMul_Mul_Add . . . . .              | 459 |
| 6.186.1 Подробное описание . . . . .         | 459 |
| 6.186.2 Функции . . . . .                    | 459 |
| 6.186.2.1 nmppsMul_Mul_Add_32f() . . . . .   | 459 |
| 6.186.2.2 nmppsMul_Mul_Add_32fcr() . . . . . | 461 |
| 6.186.2.3 nmppsMul_Mul_Add_64f() . . . . .   | 462 |
| 6.187nmppsMul_Add . . . . .                  | 464 |
| 6.187.1 Подробное описание . . . . .         | 464 |
| 6.187.2 Функции . . . . .                    | 464 |
| 6.187.2.1 nmppsMul_Add_32f() . . . . .       | 464 |
| 6.187.2.2 nmppsMul_Add_32fcr() . . . . .     | 465 |
| 6.187.2.3 nmppsMul_Add_64f() . . . . .       | 467 |

|                                                  |     |
|--------------------------------------------------|-----|
| 6.188nmppsMul_ConjMul_Add . . . . .              | 469 |
| 6.188.1 Подробное описание . . . . .             | 469 |
| 6.188.2 Функции . . . . .                        | 469 |
| 6.188.2.1 nmppsMul_ConjMul_Add_32fcr() . . . . . | 469 |
| 6.189nmppsMulC_AddC . . . . .                    | 472 |
| 6.189.1 Подробное описание . . . . .             | 472 |
| 6.189.2 Функции . . . . .                        | 472 |
| 6.189.2.1 nmppsMulC_AddC_32f() . . . . .         | 472 |
| 6.190nmppsMul . . . . .                          | 474 |
| 6.190.1 Подробное описание . . . . .             | 474 |
| 6.190.2 Функции . . . . .                        | 474 |
| 6.190.2.1 nmppsMul_32fcr() . . . . .             | 474 |
| 6.191nmppsConjMul . . . . .                      | 477 |
| 6.191.1 Подробное описание . . . . .             | 477 |
| 6.191.2 Функции . . . . .                        | 477 |
| 6.191.2.1 nmppsConjMul_32fcr() . . . . .         | 477 |
| 6.192nmppsMul_Mul_Sub . . . . .                  | 480 |
| 6.192.1 Подробное описание . . . . .             | 480 |
| 6.192.2 Функции . . . . .                        | 480 |
| 6.192.2.1 nmppsMul_Mul_Sub_32f() . . . . .       | 480 |
| 6.193nmppsMulC_Add . . . . .                     | 483 |
| 6.193.1 Подробное описание . . . . .             | 483 |
| 6.193.2 Функции . . . . .                        | 483 |
| 6.193.2.1 nmppsMulC_Add_32fcr() . . . . .        | 483 |
| 6.194nmppsMul_AddC . . . . .                     | 485 |
| 6.194.1 Подробное описание . . . . .             | 485 |
| 6.195nmppsMul_AddC . . . . .                     | 486 |
| 6.195.1 Подробное описание . . . . .             | 486 |
| 6.195.2 Функции . . . . .                        | 486 |
| 6.195.2.1 nmppsMul_AddC_32f() . . . . .          | 486 |

|                                               |     |
|-----------------------------------------------|-----|
| 6.196nmppsMulC_AddC . . . . .                 | 489 |
| 6.196.1 Подробное описание . . . . .          | 489 |
| 6.196.2 Функции . . . . .                     | 489 |
| 6.196.2.1 nmppsMulC_AddC_2x32s() . . . . .    | 490 |
| 6.197nmppsRShiftC_MulC_AddC . . . . .         | 491 |
| 6.197.1 Подробное описание . . . . .          | 491 |
| 6.198nmppsMulC_AddV_AddC . . . . .            | 492 |
| 6.198.1 Подробное описание . . . . .          | 492 |
| 6.198.2 Функции . . . . .                     | 492 |
| 6.198.2.1 nmppsMulC_AddV_AddC_32s() . . . . . | 493 |
| 6.199nmppsSumN . . . . .                      | 495 |
| 6.199.1 Подробное описание . . . . .          | 495 |
| 6.200nmppsDivC . . . . .                      | 496 |
| 6.200.1 Подробное описание . . . . .          | 496 |
| 6.200.2 Функции . . . . .                     | 497 |
| 6.200.2.1 nmppsDivC_32s() . . . . .           | 497 |
| 6.201nmppsSum . . . . .                       | 499 |
| 6.201.1 Подробное описание . . . . .          | 499 |
| 6.201.2 Функции . . . . .                     | 499 |
| 6.201.2.1 nmppsSum_16s() . . . . .            | 499 |
| 6.201.2.2 nmppsSum_32s() . . . . .            | 500 |
| 6.201.2.3 nmppsSum_64s() . . . . .            | 501 |
| 6.201.2.4 nmppsSum_8s() . . . . .             | 502 |
| 6.202nmppsSum_1 . . . . .                     | 504 |
| 6.202.1 Подробное описание . . . . .          | 504 |
| 6.203nmppsDotProd_sm . . . . .                | 505 |
| 6.203.1 Подробное описание . . . . .          | 505 |
| 6.203.2 Функции . . . . .                     | 505 |
| 6.203.2.1 nmppsDotProd_16s16sm() . . . . .    | 506 |
| 6.203.2.2 nmppsDotProd_16s32sm() . . . . .    | 507 |

|                                     |     |
|-------------------------------------|-----|
| 6.203.2.3 nmppsDotProd_32s32sm()    | 508 |
| 6.203.2.4 nmppsDotProd_8s16sm()     | 509 |
| 6.203.2.5 nmppsDotProd_8s32sm()     | 511 |
| 6.203.2.6 nmppsDotProd_8s8sm()      | 512 |
| 6.204nmppsDotProd                   | 514 |
| 6.204.1 Подробное описание          | 514 |
| 6.204.2 Функции                     | 514 |
| 6.204.2.1 nmppsDotProd_16s64s()     | 515 |
| 6.204.2.2 nmppsDotProd_32s64s()     | 516 |
| 6.204.2.3 nmppsDotProd_64s64s()     | 517 |
| 6.204.2.4 nmppsDotProd_8s64s()      | 518 |
| 6.205nmppsWeightedSum               | 520 |
| 6.205.1 Подробное описание          | 520 |
| 6.205.2 Функции                     | 521 |
| 6.205.2.1 nmppsWeightedSum_16s32s() | 521 |
| 6.205.2.2 nmppsWeightedSum_32s64s() | 522 |
| 6.205.2.3 nmppsWeightedSum_8s16s()  | 524 |
| 6.206nmppsNot_                      | 527 |
| 6.206.1 Подробное описание          | 527 |
| 6.206.2 Функции                     | 527 |
| 6.206.2.1 nmppsNot_16u()            | 528 |
| 6.206.2.2 nmppsNot_2u()             | 529 |
| 6.206.2.3 nmppsNot_32u()            | 530 |
| 6.206.2.4 nmppsNot_4u()             | 531 |
| 6.206.2.5 nmppsNot_64u()            | 532 |
| 6.206.2.6 nmppsNot_8u()             | 533 |
| 6.207nmppsAndC                      | 535 |
| 6.207.1 Подробное описание          | 535 |
| 6.208nmppsAnd                       | 536 |
| 6.208.1 Подробное описание          | 536 |

|                                        |     |
|----------------------------------------|-----|
| 6.208.2 Функции . . . . .              | 536 |
| 6.208.2.1 nmppsAnd_1() . . . . .       | 537 |
| 6.208.2.2 nmppsAnd_16u() . . . . .     | 538 |
| 6.208.2.3 nmppsAnd_2u() . . . . .      | 539 |
| 6.208.2.4 nmppsAnd_32u() . . . . .     | 540 |
| 6.208.2.5 nmppsAnd_4u() . . . . .      | 542 |
| 6.208.2.6 nmppsAnd_64u() . . . . .     | 543 |
| 6.208.2.7 nmppsAnd_8u() . . . . .      | 544 |
| 6.209nmppsAnd4V_ . . . . .             | 546 |
| 6.209.1 Подробное описание . . . . .   | 546 |
| 6.210nmppsAndNotV_ . . . . .           | 547 |
| 6.210.1 Подробное описание . . . . .   | 547 |
| 6.210.2 Функции . . . . .              | 547 |
| 6.210.2.1 nmppsAndNotV_64u() . . . . . | 547 |
| 6.211nmppsOrC . . . . .                | 550 |
| 6.211.1 Подробное описание . . . . .   | 550 |
| 6.212nmppsOr . . . . .                 | 551 |
| 6.212.1 Подробное описание . . . . .   | 551 |
| 6.212.2 Функции . . . . .              | 551 |
| 6.212.2.1 nmppsOr_1() . . . . .        | 552 |
| 6.212.2.2 nmppsOr_16u() . . . . .      | 553 |
| 6.212.2.3 nmppsOr_2u() . . . . .       | 554 |
| 6.212.2.4 nmppsOr_32u() . . . . .      | 555 |
| 6.212.2.5 nmppsOr_4u() . . . . .       | 557 |
| 6.212.2.6 nmppsOr_64u() . . . . .      | 558 |
| 6.212.2.7 nmppsOr_8u() . . . . .       | 559 |
| 6.213nmppsOr3V_ . . . . .              | 561 |
| 6.213.1 Подробное описание . . . . .   | 561 |
| 6.214nmppsOr4V_ . . . . .              | 562 |
| 6.214.1 Подробное описание . . . . .   | 562 |

|                                        |     |
|----------------------------------------|-----|
| 6.215nmppsXorC . . . . .               | 563 |
| 6.215.1 Подробное описание . . . . .   | 563 |
| 6.215.2 Функции . . . . .              | 563 |
| 6.215.2.1 nmppsXorC_16u() . . . . .    | 564 |
| 6.215.2.2 nmppsXorC_32u() . . . . .    | 565 |
| 6.215.2.3 nmppsXorC_64u() . . . . .    | 566 |
| 6.215.2.4 nmppsXorC_8u() . . . . .     | 567 |
| 6.216nmppsXor . . . . .                | 569 |
| 6.216.1 Подробное описание . . . . .   | 569 |
| 6.216.2 Функции . . . . .              | 569 |
| 6.216.2.1 nmppsXor_16u() . . . . .     | 570 |
| 6.216.2.2 nmppsXor_32u() . . . . .     | 571 |
| 6.216.2.3 nmppsXor_4u() . . . . .      | 572 |
| 6.216.2.4 nmppsXor_64u() . . . . .     | 573 |
| 6.216.2.5 nmppsXor_8u() . . . . .      | 575 |
| 6.217nmppsMaskV_ . . . . .             | 577 |
| 6.217.1 Подробное описание . . . . .   | 577 |
| 6.217.2 Функции . . . . .              | 577 |
| 6.217.2.1 nmppsMaskV_64u() . . . . .   | 577 |
| 6.218nmppsRShiftC . . . . .            | 580 |
| 6.218.1 Подробное описание . . . . .   | 580 |
| 6.218.2 Функции . . . . .              | 580 |
| 6.218.2.1 nmppsRShiftC_16s() . . . . . | 581 |
| 6.218.2.2 nmppsRShiftC_32s() . . . . . | 582 |
| 6.218.2.3 nmppsRShiftC_64s() . . . . . | 584 |
| 6.218.2.4 nmppsRShiftC_8s() . . . . .  | 585 |
| 6.219nmppsRShiftC_ . . . . .           | 588 |
| 6.219.1 Подробное описание . . . . .   | 588 |
| 6.219.2 Функции . . . . .              | 588 |
| 6.219.2.1 nmppsRShiftC_16u() . . . . . | 589 |



|                                    |     |
|------------------------------------|-----|
| 6.219.2.2 nmppsRShiftC_32u()       | 590 |
| 6.219.2.3 nmppsRShiftC_4u()        | 591 |
| 6.219.2.4 nmppsRShiftC_64u()       | 593 |
| 6.219.2.5 nmppsRShiftC_8u()        | 594 |
| 6.220nmppsRShiftC_AddC_            | 597 |
| 6.221nmppsDisplaceBits             | 598 |
| 6.221.1 Подробное описание         | 598 |
| 6.221.2 Функции                    | 599 |
| 6.221.2.1 nmppsFwdShiftBitstream() | 599 |
| 6.222nmppsSet-инициализация        | 602 |
| 6.222.1 Подробное описание         | 602 |
| 6.222.2 Функции                    | 602 |
| 6.222.2.1 nmppsSet_16s()           | 603 |
| 6.222.2.2 nmppsSet_16u()           | 604 |
| 6.222.2.3 nmppsSet_32s()           | 605 |
| 6.222.2.4 nmppsSet_32u()           | 606 |
| 6.222.2.5 nmppsSet_64s()           | 607 |
| 6.222.2.6 nmppsSet_64sp()          | 608 |
| 6.222.2.7 nmppsSet_64u()           | 609 |
| 6.222.2.8 nmppsSet_64up()          | 610 |
| 6.222.2.9 nmppsSet_8s()            | 611 |
| 6.222.2.10nmppsSet_8u()            | 612 |
| 6.223nmppsRandUniform              | 614 |
| 6.223.1 Подробное описание         | 614 |
| 6.223.2 Функции                    | 614 |
| 6.223.2.1 nmppsRandUniform_16s()   | 614 |
| 6.223.2.2 nmppsRandUniform_16u()   | 615 |
| 6.223.2.3 nmppsRandUniform_32s()   | 616 |
| 6.223.2.4 nmppsRandUniform_32u()   | 617 |
| 6.223.2.5 nmppsRandUniform_64s()   | 618 |

|                                          |     |
|------------------------------------------|-----|
| 6.223.2.6 nmppsRandUniform_64u()         | 619 |
| 6.223.2.7 nmppsRandUniform_8s()          | 621 |
| 6.223.2.8 nmppsRandUniform_8u()          | 622 |
| 6.224nmppsRandUniform                    | 624 |
| 6.224.1 Подробное описание               | 624 |
| 6.224.2 Функции                          | 624 |
| 6.224.2.1 nmppsRand_32f()                | 624 |
| 6.224.2.2 nmppsRandUniform_32f_integer() | 625 |
| 6.224.2.3 nmppsRandUniform_64f()         | 626 |
| 6.225nmppsRandUniform_                   | 628 |
| 6.225.1 Подробное описание               | 628 |
| 6.226nmppsRamp_                          | 629 |
| 6.226.1 Подробное описание               | 629 |
| 6.226.2 Функции                          | 629 |
| 6.226.2.1 nmppsRamp_16s()                | 630 |
| 6.226.2.2 nmppsRamp_32s()                | 631 |
| 6.226.2.3 nmppsRamp_64s()                | 633 |
| 6.226.2.4 nmppsRamp_8s()                 | 635 |
| 6.227nmppsConvert                        | 638 |
| 6.227.1 Подробное описание               | 638 |
| 6.227.2 Функции                          | 639 |
| 6.227.2.1 nmppsConvert_16s32s()          | 639 |
| 6.227.2.2 nmppsConvert_16s4s()           | 640 |
| 6.227.2.3 nmppsConvert_16s64s()          | 641 |
| 6.227.2.4 nmppsConvert_16s8s()           | 642 |
| 6.227.2.5 nmppsConvert_16u32u()          | 643 |
| 6.227.2.6 nmppsConvert_1s2s()            | 645 |
| 6.227.2.7 nmppsConvert_1u2u()            | 647 |
| 6.227.2.8 nmppsConvert_1u4u()            | 648 |
| 6.227.2.9 nmppsConvert_2s1s()            | 649 |

|            |                                       |     |
|------------|---------------------------------------|-----|
| 6.227.2.10 | <code>nmppsConvert_2s4s()</code>      | 650 |
| 6.227.2.11 | <code>nmppsConvert_2u4u()</code>      | 651 |
| 6.227.2.12 | <code>nmppsConvert_32s16s()</code>    | 652 |
| 6.227.2.13 | <code>nmppsConvert_32s64s()</code>    | 653 |
| 6.227.2.14 | <code>nmppsConvert_32s8s()</code>     | 655 |
| 6.227.2.15 | <code>nmppsConvert_32u64u()</code>    | 657 |
| 6.227.2.16 | <code>nmppsConvert_4s1s()</code>      | 658 |
| 6.227.2.17 | <code>nmppsConvert_4s2s()</code>      | 659 |
| 6.227.2.18 | <code>nmppsConvert_4s8s()</code>      | 660 |
| 6.227.2.19 | <code>nmppsConvert_4u8u()</code>      | 661 |
| 6.227.2.20 | <code>nmppsConvert_64s16s()</code>    | 662 |
| 6.227.2.21 | <code>nmppsConvert_64s32s()</code>    | 663 |
| 6.227.2.22 | <code>nmppsConvert_8s16s()</code>     | 664 |
| 6.227.2.23 | <code>nmppsConvert_8s32s()</code>     | 666 |
| 6.227.2.24 | <code>nmppsConvert_8s4s()</code>      | 668 |
| 6.227.2.25 | <code>nmppsConvert_8s64s()</code>     | 669 |
| 6.227.2.26 | <code>nmppsConvert_8u16u()</code>     | 670 |
| 6.227.2.27 | <code>nmppsConvert_8u32u()</code>     | 671 |
| 6.227.2.28 | <code>nmppsConvert_8u64u()</code>     | 672 |
| 6.228      | <code>nmppsConvert</code>             | 674 |
| 6.228.1    | Подробное описание                    | 674 |
| 6.228.2    | Функции                               | 674 |
| 6.228.2.1  | <code>nmppsConvert_32f32fcr()</code>  | 674 |
| 6.228.2.2  | <code>nmppsConvert_32f64f()</code>    | 675 |
| 6.228.2.3  | <code>nmppsConvert_32s32f()</code>    | 677 |
| 6.228.2.4  | <code>nmppsConvert_32s32fcr()</code>  | 678 |
| 6.228.2.5  | <code>nmppsConvert_32sc32fcr()</code> | 679 |
| 6.228.2.6  | <code>nmppsConvert_32u32fcr()</code>  | 680 |
| 6.228.2.7  | <code>nmppsConvert_64f32f()</code>    | 681 |
| 6.229      | <code>nmppsConvert_rounding</code>    | 683 |

|                                                    |     |
|----------------------------------------------------|-----|
| 6.229.1 Подробное описание . . . . .               | 683 |
| 6.229.2 Функции . . . . .                          | 683 |
| 6.229.2.1 nmppsConvert_32f32s_ceiling() . . . . .  | 683 |
| 6.229.2.2 nmppsConvert_32f32s_floor() . . . . .    | 684 |
| 6.229.2.3 nmppsConvert_32f32s_rounding() . . . . . | 686 |
| 6.229.2.4 nmppsConvert_32f32s_truncate() . . . . . | 687 |
| 6.230nmppsMerge . . . . .                          | 689 |
| 6.230.1 Подробное описание . . . . .               | 689 |
| 6.230.2 Функции . . . . .                          | 689 |
| 6.230.2.1 nmppsMerge_32f() . . . . .               | 689 |
| 6.231nmppsConvertRisc . . . . .                    | 692 |
| 6.231.1 Подробное описание . . . . .               | 692 |
| 6.231.2 Функции . . . . .                          | 692 |
| 6.231.2.1 nmppsConvertRisc_32u8u() . . . . .       | 692 |
| 6.231.2.2 nmppsConvertRisc_8u32u() . . . . .       | 693 |
| 6.232nmppsCopy_ . . . . .                          | 695 |
| 6.232.1 Подробное описание . . . . .               | 695 |
| 6.232.2 Функции . . . . .                          | 695 |
| 6.232.2.1 nmppsCopy_16s() . . . . .                | 696 |
| 6.232.2.2 nmppsCopy_16u() . . . . .                | 697 |
| 6.232.2.3 nmppsCopy_2s() . . . . .                 | 698 |
| 6.232.2.4 nmppsCopy_32s() . . . . .                | 699 |
| 6.232.2.5 nmppsCopy_32u() . . . . .                | 700 |
| 6.232.2.6 nmppsCopy_4s() . . . . .                 | 701 |
| 6.232.2.7 nmppsCopy_4u() . . . . .                 | 702 |
| 6.232.2.8 nmppsCopy_64s() . . . . .                | 703 |
| 6.232.2.9 nmppsCopy_64sc() . . . . .               | 705 |
| 6.232.2.10nmppsCopy_64u() . . . . .                | 706 |
| 6.232.2.11nmppsCopy_8s() . . . . .                 | 707 |
| 6.232.2.12nmppsCopy_8u() . . . . .                 | 708 |

|                                        |     |
|----------------------------------------|-----|
| 6.233nmppsCoryua_ . . . . .            | 710 |
| 6.233.1 Подробное описание . . . . .   | 710 |
| 6.233.2 Функции . . . . .              | 710 |
| 6.233.2.1 nmppsCoryua_8s() . . . . .   | 711 |
| 6.234nmppsSwap_ . . . . .              | 713 |
| 6.234.1 Подробное описание . . . . .   | 713 |
| 6.234.2 Функции . . . . .              | 713 |
| 6.234.2.1 nmppsSwap_64s() . . . . .    | 713 |
| 6.235nmppsMax_ . . . . .               | 715 |
| 6.235.1 Подробное описание . . . . .   | 715 |
| 6.235.2 Функции . . . . .              | 715 |
| 6.235.2.1 nmppsMax_16s15b() . . . . .  | 716 |
| 6.235.2.2 nmppsMax_32s31b() . . . . .  | 716 |
| 6.235.2.3 nmppsMax_64s63b() . . . . .  | 717 |
| 6.235.2.4 nmppsMax_8s7b() . . . . .    | 718 |
| 6.236nmppsMin . . . . .                | 720 |
| 6.236.1 Подробное описание . . . . .   | 720 |
| 6.236.2 Функции . . . . .              | 720 |
| 6.236.2.1 nmppsMin_16s15b() . . . . .  | 721 |
| 6.236.2.2 nmppsMin_32s31b() . . . . .  | 721 |
| 6.236.2.3 nmppsMin_64s63b() . . . . .  | 722 |
| 6.236.2.4 nmppsMin_8s7b() . . . . .    | 723 |
| 6.237nmppsMaxIndx_ . . . . .           | 725 |
| 6.237.1 Подробное описание . . . . .   | 725 |
| 6.237.2 Функции . . . . .              | 726 |
| 6.237.2.1 nmppsMaxIndx_32s() . . . . . | 726 |
| 6.237.2.2 nmppsMaxIndx_8s() . . . . .  | 727 |
| 6.238nmppsMinIndx_ . . . . .           | 729 |
| 6.238.1 Подробное описание . . . . .   | 729 |
| 6.238.2 Функции . . . . .              | 730 |

|                                       |     |
|---------------------------------------|-----|
| 6.238.2.1 nmppsMinIndx_32s()          | 730 |
| 6.238.2.2 nmppsMinIndx_8s()           | 731 |
| 6.239nmppsMinIndxVN_                  | 733 |
| 6.239.1 Подробное описание            | 733 |
| 6.239.2 Функции                       | 733 |
| 6.239.2.1 nmppsMinIndxV1024_32s()     | 733 |
| 6.239.2.2 nmppsMinIndxV16_32s()       | 734 |
| 6.239.2.3 nmppsMinIndxV256_32s()      | 735 |
| 6.239.2.4 nmppsMinIndxV25_32s()       | 735 |
| 6.239.2.5 nmppsMinIndxV9_32s()        | 736 |
| 6.240nmppsFirstZeroIndx               | 737 |
| 6.240.1 Подробное описание            | 737 |
| 6.240.2 Функции                       | 737 |
| 6.240.2.1 nmppsFirstZeroIndx_32s()    | 737 |
| 6.241nmppsFirstNonZeroIndx            | 739 |
| 6.241.1 Подробное описание            | 739 |
| 6.241.2 Функции                       | 739 |
| 6.241.2.1 nmppsFirstNonZeroIndx_32s() | 739 |
| 6.242nmppsLastZeroIndx                | 741 |
| 6.242.1 Подробное описание            | 741 |
| 6.243nmppsLastNonZeroIndx             | 742 |
| 6.243.1 Подробное описание            | 742 |
| 6.243.2 Функции                       | 742 |
| 6.243.2.1 nmppsLastNonZeroIndx_32s()  | 742 |
| 6.244nmppsMinEvery_                   | 744 |
| 6.244.1 Подробное описание            | 744 |
| 6.244.2 Функции                       | 745 |
| 6.244.2.1 nmppsMinEvery_16s()         | 745 |
| 6.244.2.2 nmppsMinEvery_32s()         | 746 |
| 6.244.2.3 nmppsMinEvery_64s()         | 747 |

|                                |     |
|--------------------------------|-----|
| 6.244.2.4 nmppsMinEvery_8s()   | 749 |
| 6.245nmppsMaxEvery_            | 751 |
| 6.245.1 Подробное описание     | 751 |
| 6.245.2 Функции                | 752 |
| 6.245.2.1 nmppsMaxEvery_16s()  | 752 |
| 6.245.2.2 nmppsMaxEvery_32s()  | 753 |
| 6.245.2.3 nmppsMaxEvery_64s()  | 754 |
| 6.245.2.4 nmppsMaxEvery_8s()   | 756 |
| 6.246nmppsMinCmpLtV_           | 758 |
| 6.246.1 Подробное описание     | 758 |
| 6.246.2 Функции                | 759 |
| 6.246.2.1 nmppsMinCmpLtV_16s() | 759 |
| 6.247nmppsCmpLt0               | 761 |
| 6.247.1 Подробное описание     | 761 |
| 6.247.2 Функции                | 761 |
| 6.247.2.1 nmppsCmpLt0_16s()    | 762 |
| 6.247.2.2 nmppsCmpLt0_32s()    | 763 |
| 6.247.2.3 nmppsCmpLt0_64s()    | 764 |
| 6.247.2.4 nmppsCmpLt0_8s()     | 765 |
| 6.248nmppsCmpLteC              | 767 |
| 6.248.1 Подробное описание     | 767 |
| 6.249nmppsCmpLtC               | 768 |
| 6.249.1 Подробное описание     | 768 |
| 6.250nmppsCmpEq0               | 769 |
| 6.250.1 Подробное описание     | 769 |
| 6.250.2 Функции                | 770 |
| 6.250.2.1 nmppsCmpEq0_16u15b() | 770 |
| 6.250.2.2 nmppsCmpEq0_32u31b() | 770 |
| 6.250.2.3 nmppsCmpEq0_8u7b()   | 771 |
| 6.251nmppsCmpMinMaxV_          | 772 |

|                                                        |     |
|--------------------------------------------------------|-----|
| 6.251.1 Подробное описание . . . . .                   | 772 |
| 6.251.2 Функции . . . . .                              | 773 |
| 6.251.2.1 nmppsCmpMinMaxV_16s() . . . . .              | 773 |
| 6.251.2.2 nmppsCmpMinMaxV_32s() . . . . .              | 774 |
| 6.251.2.3 nmppsCmpMinMaxV_8s() . . . . .               | 775 |
| 6.252nmppsClipPowC_ . . . . .                          | 777 |
| 6.252.1 Подробное описание . . . . .                   | 777 |
| 6.252.2 Функции . . . . .                              | 777 |
| 6.252.2.1 nmppsClipPowC_16s() . . . . .                | 778 |
| 6.252.2.2 nmppsClipPowC_32s() . . . . .                | 779 |
| 6.252.2.3 nmppsClipPowC_64s() . . . . .                | 781 |
| 6.252.2.4 nmppsClipPowC_8s() . . . . .                 | 782 |
| 6.253nmppsClipCC_ . . . . .                            | 785 |
| 6.253.1 Подробное описание . . . . .                   | 785 |
| 6.254nmppsClipRShiftConvert_AddC_ . . . . .            | 786 |
| 6.254.1 Подробное описание . . . . .                   | 786 |
| 6.254.2 Функции . . . . .                              | 786 |
| 6.254.2.1 nmppsClipRShiftConvertAddC_16s8s() . . . . . | 787 |
| 6.254.2.2 nmppsClipRShiftConvertAddC_32s8s() . . . . . | 789 |
| 6.255nmppsClipConvert_AddC_ . . . . .                  | 792 |
| 6.255.1 Подробное описание . . . . .                   | 792 |
| 6.256nmppsCmpEqC . . . . .                             | 794 |
| 6.256.1 Подробное описание . . . . .                   | 794 |
| 6.256.2 Функции . . . . .                              | 794 |
| 6.256.2.1 nmppsCmpEqC_16u15b() . . . . .               | 795 |
| 6.256.2.2 nmppsCmpEqC_4u3b() . . . . .                 | 795 |
| 6.256.2.3 nmppsCmpEqC_8u7b() . . . . .                 | 796 |
| 6.257nmppsCmpNe0 . . . . .                             | 797 |
| 6.257.1 Подробное описание . . . . .                   | 797 |
| 6.257.2 Функции . . . . .                              | 797 |



|                                |     |
|--------------------------------|-----|
| 6.257.2.1 nmppsCmpNe0_16s()    | 797 |
| 6.257.2.2 nmppsCmpNe0_32s()    | 799 |
| 6.257.2.3 nmppsCmpNe0_64s()    | 800 |
| 6.257.2.4 nmppsCmpNe0_8s()     | 801 |
| 6.258nmppsCmpNeC               | 803 |
| 6.258.1 Подробное описание     | 803 |
| 6.258.2 Функции                | 803 |
| 6.258.2.1 nmppsCmpNeC_16s()    | 803 |
| 6.258.2.2 nmppsCmpNeC_32s()    | 805 |
| 6.258.2.3 nmppsCmpNeC_64s()    | 806 |
| 6.258.2.4 nmppsCmpNeC_8s()     | 807 |
| 6.259nmppsCmpNeC_flag          | 809 |
| 6.259.1 Подробное описание     | 809 |
| 6.260nmppsCmpLtC               | 810 |
| 6.260.1 Подробное описание     | 810 |
| 6.260.2 Функции                | 810 |
| 6.260.2.1 nmppsCmpLtC_16s15b() | 811 |
| 6.260.2.2 nmppsCmpLtC_32s31b() | 812 |
| 6.260.2.3 nmppsCmpLtC_64s63b() | 813 |
| 6.260.2.4 nmppsCmpLtC_8s7b()   | 813 |
| 6.261nmppsCmpGtC               | 815 |
| 6.261.1 Подробное описание     | 815 |
| 6.262nmppsCmpGteC              | 817 |
| 6.262.1 Подробное описание     | 817 |
| 6.263nmppsCmpEqV_              | 818 |
| 6.263.1 Подробное описание     | 818 |
| 6.264nmppsCmpNe                | 819 |
| 6.264.1 Подробное описание     | 819 |
| 6.264.2 Функции                | 819 |
| 6.264.2.1 nmppsCmpNe_16s()     | 819 |

|                                 |     |
|---------------------------------|-----|
| 6.264.2.2 nmppsCmpNe_32s()      | 821 |
| 6.264.2.3 nmppsCmpNe_64s()      | 821 |
| 6.264.2.4 nmppsCmpNe_8s()       | 822 |
| 6.265nmppsCmpLt                 | 823 |
| 6.265.1 Подробное описание      | 823 |
| 6.266nmppsCmpNeV_               | 824 |
| 6.266.1 Подробное описание      | 824 |
| 6.266.2 Функции                 | 824 |
| 6.266.2.1 nmppsCmpNeV_16u()     | 825 |
| 6.266.2.2 nmppsCmpNeV_8u()      | 825 |
| 6.267Vec_ClipRShiftConvert_AddC | 827 |
| 6.267.1 Подробное описание      | 827 |
| 6.268nmppsAddr_                 | 828 |
| 6.268.1 Подробное описание      | 828 |
| 6.268.2 Функции                 | 828 |
| 6.268.2.1 nmppsAddr_1()         | 829 |
| 6.268.2.2 nmppsAddr_16s()       | 829 |
| 6.268.2.3 nmppsAddr_16u()       | 830 |
| 6.268.2.4 nmppsAddr_2s()        | 830 |
| 6.268.2.5 nmppsAddr_2u()        | 831 |
| 6.268.2.6 nmppsAddr_32f()       | 831 |
| 6.268.2.7 nmppsAddr_32s()       | 832 |
| 6.268.2.8 nmppsAddr_32u()       | 833 |
| 6.268.2.9 nmppsAddr_4s()        | 833 |
| 6.268.2.10nmppsAddr_4u()        | 834 |
| 6.268.2.11nmppsAddr_64f()       | 834 |
| 6.268.2.12nmppsAddr_64s()       | 835 |
| 6.268.2.13nmppsAddr_64u()       | 835 |
| 6.268.2.14nmppsAddr_8s()        | 836 |
| 6.268.2.15nmppsAddr_8u()        | 837 |

|                                       |     |
|---------------------------------------|-----|
| 6.269nmppsSetVal_ . . . . .           | 838 |
| 6.269.1 Подробное описание . . . . .  | 838 |
| 6.269.2 Функции . . . . .             | 838 |
| 6.269.2.1 nmppsPut_1() . . . . .      | 839 |
| 6.269.2.2 nmppsPut_16s() . . . . .    | 840 |
| 6.269.2.3 nmppsPut_16u() . . . . .    | 841 |
| 6.269.2.4 nmppsPut_2s() . . . . .     | 841 |
| 6.269.2.5 nmppsPut_2u() . . . . .     | 842 |
| 6.269.2.6 nmppsPut_32s() . . . . .    | 842 |
| 6.269.2.7 nmppsPut_32u() . . . . .    | 843 |
| 6.269.2.8 nmppsPut_4s() . . . . .     | 843 |
| 6.269.2.9 nmppsPut_4u() . . . . .     | 844 |
| 6.269.2.10nmppsPut_64s() . . . . .    | 845 |
| 6.269.2.11nmppsPut_64u() . . . . .    | 845 |
| 6.269.2.12nmppsPut_8s() . . . . .     | 846 |
| 6.269.2.13nmppsPut_8u() . . . . .     | 846 |
| 6.270nmppsGetVal_ . . . . .           | 849 |
| 6.270.1 Подробное описание . . . . .  | 849 |
| 6.270.2 Функции . . . . .             | 849 |
| 6.270.2.1 nmppsGetVal_1() . . . . .   | 850 |
| 6.270.2.2 nmppsGetVal_16s() . . . . . | 851 |
| 6.270.2.3 nmppsGetVal_16u() . . . . . | 852 |
| 6.270.2.4 nmppsGetVal_2s() . . . . .  | 852 |
| 6.270.2.5 nmppsGetVal_2u() . . . . .  | 853 |
| 6.270.2.6 nmppsGetVal_32s() . . . . . | 853 |
| 6.270.2.7 nmppsGetVal_32u() . . . . . | 854 |
| 6.270.2.8 nmppsGetVal_4s() . . . . .  | 854 |
| 6.270.2.9 nmppsGetVal_4u() . . . . .  | 855 |
| 6.270.2.10nmppsGetVal_64s() . . . . . | 856 |
| 6.270.2.11nmppsGetVal_64u() . . . . . | 856 |

|            |                      |     |
|------------|----------------------|-----|
| 6.270.2.12 | nmppsGetVal_8s()     | 857 |
| 6.270.2.13 | nmppsGetVal_8u()     | 857 |
| 6.271      | nmppsGetVal_(return) | 859 |
| 6.271.1    | Подробное описание   | 859 |
| 6.271.2    | Функции              | 859 |
| 6.271.2.1  | nmppsGet_1()         | 859 |
| 6.271.2.2  | nmppsGet_16s()       | 860 |
| 6.271.2.3  | nmppsGet_16u()       | 861 |
| 6.271.2.4  | nmppsGet_2s()        | 861 |
| 6.271.2.5  | nmppsGet_2u()        | 862 |
| 6.271.2.6  | nmppsGet_32s()       | 862 |
| 6.271.2.7  | nmppsGet_32u()       | 863 |
| 6.271.2.8  | nmppsGet_4s()        | 863 |
| 6.271.2.9  | nmppsGet_4u()        | 864 |
| 6.271.2.10 | nmppsGet_8s()        | 865 |
| 6.271.2.11 | nmppsGet_8u()        | 865 |
| 6.272      | VEC_QSort            | 867 |
| 6.272.1    | Подробное описание   | 867 |
| 6.272.2    | Функции              | 867 |
| 6.272.2.1  | nmppsQSort_32s()     | 867 |
| 6.273      | nmppsRemap_          | 869 |
| 6.273.1    | Подробное описание   | 869 |
| 6.274      | nmppSplitTmp         | 871 |
| 6.274.1    | Подробное описание   | 871 |
| 6.275      | nmppSplit            | 872 |
| 6.275.1    | Подробное описание   | 872 |
| 6.276      | nmppMerge            | 873 |
| 6.276.1    | Подробное описание   | 873 |
| 6.277      | nmppSplit_32fcr      | 874 |
| 6.277.1    | Подробное описание   | 874 |

|                                                       |     |
|-------------------------------------------------------|-----|
| 6.278nmppsDecimate . . . . .                          | 875 |
| 6.278.1 Подробное описание . . . . .                  | 875 |
| 6.279Целочисленные функции . . . . .                  | 876 |
| 6.279.1 Подробное описание . . . . .                  | 876 |
| 6.280Функции с плавающей точкой . . . . .             | 877 |
| 6.280.1 Подробное описание . . . . .                  | 877 |
| 6.281Типы данных . . . . .                            | 878 |
| 6.281.1 Подробное описание . . . . .                  | 878 |
| 6.282Векторные функции . . . . .                      | 879 |
| 6.282.1 Подробное описание . . . . .                  | 879 |
| 6.283Векторные функции . . . . .                      | 880 |
| 6.283.1 Подробное описание . . . . .                  | 880 |
| 6.284Матричные функции . . . . .                      | 881 |
| 6.284.1 Подробное описание . . . . .                  | 881 |
| 6.285Матричные функции . . . . .                      | 882 |
| 6.285.1 Подробное описание . . . . .                  | 882 |
| 6.286Функции обработки сигналов . . . . .             | 883 |
| 6.286.1 Подробное описание . . . . .                  | 883 |
| 6.287Функции обработки сигналов . . . . .             | 884 |
| 6.287.1 Подробное описание . . . . .                  | 884 |
| 6.288Функции обработки изображений . . . . .          | 885 |
| 6.288.1 Подробное описание . . . . .                  | 885 |
| 6.289Скалярные функции . . . . .                      | 886 |
| 6.289.1 Подробное описание . . . . .                  | 886 |
| 6.289.1.1 Введение . . . . .                          | 886 |
| 6.290Базовые регистровые функции библиотеки . . . . . | 888 |
| 6.290.1 Подробное описание . . . . .                  | 888 |
| 6.291nmblas . . . . .                                 | 889 |
| 6.292контроль переполнения . . . . .                  | 890 |
| 6.292.1 Подробное описание . . . . .                  | 890 |

|                                                  |     |
|--------------------------------------------------|-----|
| 6.292.2 Макросы . . . . .                        | 890 |
| 6.292.2.1 GetVec . . . . .                       | 890 |
| 6.292.3 Функции . . . . .                        | 891 |
| 6.292.3.1 operator<<() . . . . .                 | 891 |
| 6.293 Элементарные функции . . . . .             | 892 |
| 6.293.1 Подробное описание . . . . .             | 892 |
| 6.294 функции взвешенного суммирования . . . . . | 893 |
| 6.294.1 Подробное описание . . . . .             | 893 |
| 6.295 Целевые функции . . . . .                  | 894 |
| 6.295.1 Подробное описание . . . . .             | 894 |
| 6.296 Vec_0_sub_data . . . . .                   | 895 |
| 6.296.1 Подробное описание . . . . .             | 895 |
| 6.297 Vec_activate_data . . . . .                | 896 |
| 6.297.1 Подробное описание . . . . .             | 896 |
| 6.298 Vec_activate_data_add_0 . . . . .          | 897 |
| 6.298.1 Подробное описание . . . . .             | 897 |
| 6.299 Vec_activate_data_xor_data . . . . .       | 898 |
| 6.299.1 Подробное описание . . . . .             | 898 |
| 6.300 Vec_activate_data_add_ram . . . . .        | 899 |
| 6.300.1 Подробное описание . . . . .             | 899 |
| 6.301 Vec_Add_VV_shift . . . . .                 | 900 |
| 6.301.1 Подробное описание . . . . .             | 900 |
| 6.302 Vec_afifo . . . . .                        | 901 |
| 6.302.1 Подробное описание . . . . .             | 901 |
| 6.303 Vec_data . . . . .                         | 902 |
| 6.303.1 Подробное описание . . . . .             | 902 |
| 6.304 Vec_data_add_afifo . . . . .               | 903 |
| 6.304.1 Подробное описание . . . . .             | 903 |
| 6.305 Vec_data_add_ram . . . . .                 | 904 |
| 6.305.1 Подробное описание . . . . .             | 904 |

|                                       |     |
|---------------------------------------|-----|
| 6.306Vec_data_and_ram . . . . .       | 905 |
| 6.306.1 Подробное описание . . . . .  | 905 |
| 6.307Vec_data_or_ram . . . . .        | 906 |
| 6.307.1 Подробное описание . . . . .  | 906 |
| 6.308Vec_data_sub_ram . . . . .       | 907 |
| 6.308.1 Подробное описание . . . . .  | 907 |
| 6.309Vec_data_xor_ram . . . . .       | 908 |
| 6.309.1 Подробное описание . . . . .  | 908 |
| 6.310Vec_FilterCoreRow2 . . . . .     | 909 |
| 6.310.1 Подробное описание . . . . .  | 909 |
| 6.311Vec_FilterCoreRow4 . . . . .     | 910 |
| 6.311.1 Подробное описание . . . . .  | 910 |
| 6.312Vec_FilterCoreRow8 . . . . .     | 911 |
| 6.312.1 Подробное описание . . . . .  | 911 |
| 6.313Vec_And . . . . .                | 912 |
| 6.313.1 Подробное описание . . . . .  | 912 |
| 6.314Vec_Mask . . . . .               | 913 |
| 6.314.1 Подробное описание . . . . .  | 913 |
| 6.315Vec_Or . . . . .                 | 914 |
| 6.315.1 Подробное описание . . . . .  | 914 |
| 6.316Vec_Xor . . . . .                | 915 |
| 6.316.1 Подробное описание . . . . .  | 915 |
| 6.317Vec_Abs . . . . .                | 916 |
| 6.317.1 Подробное описание . . . . .  | 916 |
| 6.318Vec_Add . . . . .                | 917 |
| 6.318.1 Подробное описание . . . . .  | 917 |
| 6.319Vec_ClipExt . . . . .            | 918 |
| 6.319.1 Подробное описание . . . . .  | 918 |
| 6.320Vec_ClipMul2D2W8_AddVr . . . . . | 919 |
| 6.320.1 Подробное описание . . . . .  | 919 |

|                                        |     |
|----------------------------------------|-----|
| 6.321 Vec_ClipMulNDNW2_AddVr . . . . . | 920 |
| 6.321.1 Подробное описание . . . . .   | 920 |
| 6.322 Vec_ClipMulNDNW4_AddVr . . . . . | 921 |
| 6.322.1 Подробное описание . . . . .   | 921 |
| 6.323 Vec_ClipMulNDNW8_AddVr . . . . . | 922 |
| 6.323.1 Подробное описание . . . . .   | 922 |
| 6.324 Vec_IncNeg . . . . .             | 923 |
| 6.324.1 Подробное описание . . . . .   | 923 |
| 6.325 Vec_Mul2D2W1_AddVr . . . . .     | 924 |
| 6.325.1 Подробное описание . . . . .   | 924 |
| 6.326 Vec_Mul2D2W2_AddVr . . . . .     | 925 |
| 6.326.1 Подробное описание . . . . .   | 925 |
| 6.327 Vec_Mul2D2W4_AddVr . . . . .     | 926 |
| 6.327.1 Подробное описание . . . . .   | 926 |
| 6.328 Vec_Mul2D2W8_AddVr . . . . .     | 927 |
| 6.328.1 Подробное описание . . . . .   | 927 |
| 6.329 Vec_Mul3D3W2_AddVr . . . . .     | 928 |
| 6.329.1 Подробное описание . . . . .   | 928 |
| 6.330 Vec_Mul3D3W8_AddVr . . . . .     | 929 |
| 6.330.1 Подробное описание . . . . .   | 929 |
| 6.331 Vec_Mul4D4W2_AddVr . . . . .     | 930 |
| 6.331.1 Подробное описание . . . . .   | 930 |
| 6.332 Vec_MulVN_AddVN . . . . .        | 931 |
| 6.332.1 Подробное описание . . . . .   | 931 |
| 6.333 Vec_Sub . . . . .                | 932 |
| 6.333.1 Подробное описание . . . . .   | 932 |
| 6.334 Vec_SubAbs . . . . .             | 933 |
| 6.334.1 Подробное описание . . . . .   | 933 |
| 6.335 Vec_SubVN_Abs . . . . .          | 934 |
| 6.335.1 Подробное описание . . . . .   | 934 |



|                                          |     |
|------------------------------------------|-----|
| 6.336Vec_Swap . . . . .                  | 935 |
| 6.336.1 Подробное описание . . . . .     | 935 |
| 6.337Vec_MUL_2V4toW8_shift . . . . .     | 936 |
| 6.337.1 Подробное описание . . . . .     | 936 |
| 6.338Vec_MUL_2V8toW16_shift . . . . .    | 937 |
| 6.338.1 Подробное описание . . . . .     | 937 |
| 6.339Vec_not_data . . . . .              | 938 |
| 6.339.1 Подробное описание . . . . .     | 938 |
| 6.340Vec_ram . . . . .                   | 939 |
| 6.340.1 Подробное описание . . . . .     | 939 |
| 6.341Vec_ram_sub_data . . . . .          | 940 |
| 6.341.1 Подробное описание . . . . .     | 940 |
| 6.342Vec_vsum_activate_data_0 . . . . .  | 941 |
| 6.342.1 Подробное описание . . . . .     | 941 |
| 6.343Vec_vsum_data_0 . . . . .           | 942 |
| 6.343.1 Подробное описание . . . . .     | 942 |
| 6.344Vec_vsum_data_affo . . . . .        | 943 |
| 6.344.1 Подробное описание . . . . .     | 943 |
| 6.345Vec_vsum_data_vr . . . . .          | 944 |
| 6.345.1 Подробное описание . . . . .     | 944 |
| 6.346Vec_vsum_shift_data_0 . . . . .     | 945 |
| 6.346.1 Подробное описание . . . . .     | 945 |
| 6.347Vec_vsum_shift_data_vr . . . . .    | 946 |
| 6.347.1 Подробное описание . . . . .     | 946 |
| 6.348Vec_vsum_shift_data_afifo . . . . . | 947 |
| 6.348.1 Подробное описание . . . . .     | 947 |
| 6.349Vec_CompareMinV . . . . .           | 948 |
| 6.349.1 Подробное описание . . . . .     | 948 |
| 6.350Vec_CompareMaxV . . . . .           | 949 |
| 6.350.1 Подробное описание . . . . .     | 949 |

|                                        |     |
|----------------------------------------|-----|
| 6.351 Vec_DupValueInVector8 . . . . .  | 950 |
| 6.351.1 Подробное описание . . . . .   | 950 |
| 6.352 Vec_DupValueInVector16 . . . . . | 951 |
| 6.352.1 Подробное описание . . . . .   | 951 |
| 6.353 Vec_BuildDiagWeights8 . . . . .  | 952 |
| 6.353.1 Подробное описание . . . . .   | 952 |
| 6.354 Vec_BuildDiagWeights16 . . . . . | 953 |
| 6.354.1 Подробное описание . . . . .   | 953 |
| 6.355 Vec_MaxVal_v8nm8s . . . . .      | 954 |
| 6.355.1 Подробное описание . . . . .   | 954 |
| 6.356 Vec_MaxVal_v4nm16s . . . . .     | 955 |
| 6.356.1 Подробное описание . . . . .   | 955 |
| 6.357 Vec_MaxVal . . . . .             | 956 |
| 6.357.1 Подробное описание . . . . .   | 956 |
| 6.358 Vec_MinVal_v8nm8s . . . . .      | 957 |
| 6.358.1 Подробное описание . . . . .   | 957 |
| 6.359 Vec_MinVal_v4nm16s . . . . .     | 958 |
| 6.359.1 Подробное описание . . . . .   | 958 |
| 6.360 Vec_MinVal . . . . .             | 959 |
| 6.360.1 Подробное описание . . . . .   | 959 |
| 6.361 Vec_AccMul1D1W32_AddVr . . . . . | 960 |
| 6.361.1 Подробное описание . . . . .   | 960 |

|         |                                                          |     |
|---------|----------------------------------------------------------|-----|
| 7       | Классы                                                   | 961 |
| 7.1     | Класс <code>C_2DSubPixelMinPosition</code>               | 961 |
| 7.1.1   | Подробное описание                                       | 961 |
| 7.2     | Класс <code>C_2DTrigSubPixelMinPosition</code>           | 961 |
| 7.2.1   | Подробное описание                                       | 962 |
| 7.3     | Класс <code>C_Allocator32</code>                         | 962 |
| 7.3.1   | Подробное описание                                       | 962 |
| 7.4     | Шаблон класса <code>C_BoxImg&lt; T &gt;</code>           | 963 |
| 7.4.1   | Подробное описание                                       | 963 |
| 7.5     | Шаблон класса <code>C_BoxVec&lt; T &gt;</code>           | 963 |
| 7.5.1   | Подробное описание                                       | 964 |
| 7.6     | Класс <code>C_Heap</code>                                | 964 |
| 7.6.1   | Подробное описание                                       | 965 |
| 7.6.2   | Методы                                                   | 965 |
| 7.6.2.1 | <code>AllocateMaxAvail()</code>                          | 965 |
| 7.7     | Шаблон класса <code>C_Img&lt; T &gt;</code>              | 965 |
| 7.7.1   | Подробное описание                                       | 966 |
| 7.8     | Класс <code>C_MultiHeap</code>                           | 966 |
| 7.8.1   | Подробное описание                                       | 967 |
| 7.9     | Класс <code>C_PlessyCornerDetector</code>                | 967 |
| 7.9.1   | Подробное описание                                       | 968 |
| 7.10    | Класс <code>C_PlessyCornerDetector_16s</code>            | 968 |
| 7.10.1  | Подробное описание                                       | 969 |
| 7.11    | Класс <code>C_PlessyCornerDetector_32f</code>            | 969 |
| 7.11.1  | Подробное описание                                       | 970 |
| 7.12    | Шаблон класса <code>C_RingBufferRemote&lt; T &gt;</code> | 970 |
| 7.12.1  | Подробное описание                                       | 971 |
| 7.13    | Структура <code>ds_struct</code>                         | 971 |
| 7.13.1  | Подробное описание                                       | 972 |
| 7.14    | Класс <code>EnterHardMode</code>                         | 972 |

|        |                                                             |     |
|--------|-------------------------------------------------------------|-----|
| 7.14.1 | Подробное описание . . . . .                                | 972 |
| 7.15   | Класс <code>I_2DSubPixelMinPosition</code> . . . . .        | 972 |
| 7.15.1 | Подробное описание . . . . .                                | 972 |
| 7.16   | Класс <code>I_PlessyCornerDetector</code> . . . . .         | 973 |
| 7.16.1 | Подробное описание . . . . .                                | 973 |
| 7.17   | Структура <code>int15in16x4</code> . . . . .                | 973 |
| 7.17.1 | Подробное описание . . . . .                                | 973 |
| 7.18   | Структура <code>int30in32x2</code> . . . . .                | 973 |
| 7.18.1 | Подробное описание . . . . .                                | 974 |
| 7.19   | Структура <code>int31in32x2</code> . . . . .                | 974 |
| 7.19.1 | Подробное описание . . . . .                                | 974 |
| 7.20   | Шаблон класса <code>mtr&lt; T &gt;</code> . . . . .         | 974 |
| 7.20.1 | Подробное описание . . . . .                                | 976 |
| 7.21   | Структура <code>nm16sc</code> . . . . .                     | 976 |
| 7.21.1 | Подробное описание . . . . .                                | 976 |
| 7.22   | Класс <code>nmchar</code> . . . . .                         | 976 |
| 7.22.1 | Подробное описание . . . . .                                | 977 |
| 7.23   | Шаблон класса <code>nmchar1D&lt; N &gt;</code> . . . . .    | 977 |
| 7.23.1 | Подробное описание . . . . .                                | 977 |
| 7.24   | Шаблон класса <code>nmchar2D&lt; Y, X &gt;</code> . . . . . | 977 |
| 7.24.1 | Подробное описание . . . . .                                | 978 |
| 7.25   | Шаблон класса <code>nmintpack&lt; T &gt;</code> . . . . .   | 978 |
| 7.25.1 | Подробное описание . . . . .                                | 978 |
| 7.26   | Шаблон класса <code>nmtr&lt; T &gt;</code> . . . . .        | 978 |
| 7.26.1 | Подробное описание . . . . .                                | 980 |
| 7.27   | Структура <code>NmppsFFTSpec</code> . . . . .               | 980 |
| 7.27.1 | Подробное описание . . . . .                                | 980 |
| 7.28   | Структура <code>NmppsFFTSpec_32fcr</code> . . . . .         | 980 |
| 7.28.1 | Подробное описание . . . . .                                | 981 |
| 7.29   | Структура <code>NmppsFrame_16s</code> . . . . .             | 981 |

|        |                                                              |     |
|--------|--------------------------------------------------------------|-----|
| 7.29.1 | Подробное описание . . . . .                                 | 981 |
| 7.30   | Структура <code>NmrpsFrame_16u</code> . . . . .              | 981 |
| 7.30.1 | Подробное описание . . . . .                                 | 981 |
| 7.31   | Структура <code>NmrpsFrame_32s</code> . . . . .              | 981 |
| 7.31.1 | Подробное описание . . . . .                                 | 982 |
| 7.32   | Структура <code>NmrpsFrame_32u</code> . . . . .              | 982 |
| 7.32.1 | Подробное описание . . . . .                                 | 982 |
| 7.33   | Структура <code>NmrpsFrame_64s</code> . . . . .              | 982 |
| 7.33.1 | Подробное описание . . . . .                                 | 982 |
| 7.34   | Структура <code>NmrpsFrame_64u</code> . . . . .              | 982 |
| 7.34.1 | Подробное описание . . . . .                                 | 983 |
| 7.35   | Структура <code>NmrpsFrame_8s</code> . . . . .               | 983 |
| 7.35.1 | Подробное описание . . . . .                                 | 983 |
| 7.36   | Структура <code>NmrpsFrame_8u</code> . . . . .               | 983 |
| 7.36.1 | Подробное описание . . . . .                                 | 983 |
| 7.37   | Структура <code>NmrpsMallocSpec</code> . . . . .             | 984 |
| 7.37.1 | Подробное описание . . . . .                                 | 984 |
| 7.38   | Структура <code>NmrpsTmpSpec</code> . . . . .                | 984 |
| 7.38.1 | Подробное описание . . . . .                                 | 984 |
| 7.39   | Структура <code>nmreg</code> . . . . .                       | 984 |
| 7.39.1 | Подробное описание . . . . .                                 | 985 |
| 7.40   | Класс <code>nmshort</code> . . . . .                         | 985 |
| 7.40.1 | Подробное описание . . . . .                                 | 985 |
| 7.41   | Шаблон класса <code>nmshort2D&lt; Y, X &gt;</code> . . . . . | 985 |
| 7.41.1 | Подробное описание . . . . .                                 | 986 |
| 7.42   | Шаблон класса <code>nmvespack&lt; T &gt;</code> . . . . .    | 986 |
| 7.42.1 | Подробное описание . . . . .                                 | 987 |
| 7.43   | Структура <code>RGB32_nm10s</code> . . . . .                 | 988 |
| 7.43.1 | Подробное описание . . . . .                                 | 988 |
| 7.44   | Структура <code>RGB32_nm10u</code> . . . . .                 | 988 |

|          |                                                |     |
|----------|------------------------------------------------|-----|
| 7.44.1   | Подробное описание . . . . .                   | 988 |
| 7.45     | Структура RGB32_nm8s . . . . .                 | 988 |
| 7.45.1   | Подробное описание . . . . .                   | 989 |
| 7.46     | Структура RGB32_nm8u . . . . .                 | 989 |
| 7.46.1   | Подробное описание . . . . .                   | 989 |
| 7.47     | Структура RGB64_nm16u . . . . .                | 989 |
| 7.47.1   | Подробное описание . . . . .                   | 989 |
| 7.48     | Структура RoundShift . . . . .                 | 990 |
| 7.48.1   | Подробное описание . . . . .                   | 990 |
| 7.49     | Класс RPoint . . . . .                         | 990 |
| 7.49.1   | Подробное описание . . . . .                   | 990 |
| 7.50     | Структура S_BufferInfo . . . . .               | 990 |
| 7.50.1   | Подробное описание . . . . .                   | 991 |
| 7.51     | Структура s_int32x2 . . . . .                  | 991 |
| 7.51.1   | Подробное описание . . . . .                   | 991 |
| 7.52     | Структура s_nm32fc . . . . .                   | 992 |
| 7.52.1   | Подробное описание . . . . .                   | 992 |
| 7.53     | Структура s_nm32fcr . . . . .                  | 992 |
| 7.53.1   | Подробное описание . . . . .                   | 992 |
| 7.54     | Структура s_nm32sc . . . . .                   | 992 |
| 7.54.1   | Подробное описание . . . . .                   | 992 |
| 7.55     | Структура s_nm64sc . . . . .                   | 993 |
| 7.55.1   | Подробное описание . . . . .                   | 993 |
| 7.55.2   | Данные класса . . . . .                        | 993 |
| 7.55.2.1 | im . . . . .                                   | 993 |
| 7.55.2.2 | re . . . . .                                   | 993 |
| 7.56     | Структура S_nmppiFilterKernel . . . . .        | 993 |
| 7.56.1   | Подробное описание . . . . .                   | 994 |
| 7.57     | Структура S_nmppiFilterKernel_32s32s . . . . . | 994 |
| 7.57.1   | Подробное описание . . . . .                   | 994 |

|                                     |      |
|-------------------------------------|------|
| 7.58 Структура s_v16nm16s . . . . . | 994  |
| 7.58.1 Подробное описание . . . . . | 994  |
| 7.59 Структура s_v16nm16u . . . . . | 995  |
| 7.59.1 Подробное описание . . . . . | 995  |
| 7.60 Структура s_v16nm32s . . . . . | 995  |
| 7.60.1 Подробное описание . . . . . | 995  |
| 7.61 Структура s_v16nm32u . . . . . | 995  |
| 7.61.1 Подробное описание . . . . . | 996  |
| 7.62 Структура s_v16nm4u . . . . .  | 996  |
| 7.62.1 Подробное описание . . . . . | 996  |
| 7.63 Структура s_v16nm8s . . . . .  | 996  |
| 7.63.1 Подробное описание . . . . . | 996  |
| 7.64 Структура s_v16nm8u . . . . .  | 997  |
| 7.64.1 Подробное описание . . . . . | 997  |
| 7.65 Структура s_v2nm32f . . . . .  | 997  |
| 7.65.1 Подробное описание . . . . . | 997  |
| 7.66 Структура s_v2nm32s . . . . .  | 997  |
| 7.66.1 Подробное описание . . . . . | 998  |
| 7.67 Структура s_v2nm32u . . . . .  | 998  |
| 7.67.1 Подробное описание . . . . . | 998  |
| 7.68 Структура s_v4nm16s . . . . .  | 998  |
| 7.68.1 Подробное описание . . . . . | 998  |
| 7.69 Структура s_v4nm16u . . . . .  | 999  |
| 7.69.1 Подробное описание . . . . . | 999  |
| 7.70 Структура s_v4nm32f . . . . .  | 999  |
| 7.70.1 Подробное описание . . . . . | 999  |
| 7.71 Структура s_v4nm32s . . . . .  | 999  |
| 7.71.1 Подробное описание . . . . . | 1000 |
| 7.72 Структура s_v4nm32u . . . . .  | 1000 |
| 7.72.1 Подробное описание . . . . . | 1000 |

|                                                                       |      |
|-----------------------------------------------------------------------|------|
| 7.73 Структура <code>s_v4nm64f</code> . . . . .                       | 1000 |
| 7.73.1 Подробное описание . . . . .                                   | 1000 |
| 7.74 Структура <code>s_v4nm8u</code> . . . . .                        | 1000 |
| 7.74.1 Подробное описание . . . . .                                   | 1001 |
| 7.75 Структура <code>s_v8nm16s</code> . . . . .                       | 1001 |
| 7.75.1 Подробное описание . . . . .                                   | 1001 |
| 7.76 Структура <code>s_v8nm16u</code> . . . . .                       | 1001 |
| 7.76.1 Подробное описание . . . . .                                   | 1002 |
| 7.77 Структура <code>s_v8nm32s</code> . . . . .                       | 1002 |
| 7.77.1 Подробное описание . . . . .                                   | 1002 |
| 7.78 Структура <code>s_v8nm32u</code> . . . . .                       | 1002 |
| 7.78.1 Подробное описание . . . . .                                   | 1002 |
| 7.79 Структура <code>s_v8nm8s</code> . . . . .                        | 1003 |
| 7.79.1 Подробное описание . . . . .                                   | 1003 |
| 7.80 Структура <code>s_v8nm8u</code> . . . . .                        | 1003 |
| 7.80.1 Подробное описание . . . . .                                   | 1003 |
| 7.81 Структура <code>SpecTmp1</code> . . . . .                        | 1003 |
| 7.81.1 Подробное описание . . . . .                                   | 1004 |
| 7.82 Структура <code>spot_struct</code> . . . . .                     | 1004 |
| 7.82.1 Подробное описание . . . . .                                   | 1004 |
| 7.83 Структура <code>tagSegmentInfo</code> . . . . .                  | 1004 |
| 7.83.1 Подробное описание . . . . .                                   | 1005 |
| 7.84 Шаблон класса <code>tfixpoint&lt; T, point &gt;</code> . . . . . | 1005 |
| 7.84.1 Подробное описание . . . . .                                   | 1006 |
| 7.85 Структура <code>Tmp2BuffSpec</code> . . . . .                    | 1006 |
| 7.85.1 Подробное описание . . . . .                                   | 1006 |
| 7.86 Класс <code>uint16ptr</code> . . . . .                           | 1006 |
| 7.86.1 Подробное описание . . . . .                                   | 1007 |
| 7.87 Класс <code>uint8ptr</code> . . . . .                            | 1007 |
| 7.87.1 Подробное описание . . . . .                                   | 1008 |
| 7.88 Структура <code>v16nm4s</code> . . . . .                         | 1008 |
| 7.88.1 Подробное описание . . . . .                                   | 1008 |
| 7.89 Структура <code>v4nm8s</code> . . . . .                          | 1008 |
| 7.89.1 Подробное описание . . . . .                                   | 1009 |
| 7.90 Шаблон класса <code>vec&lt; T &gt;</code> . . . . .              | 1009 |
| 7.90.1 Подробное описание . . . . .                                   | 1010 |



|         |                                                |      |
|---------|------------------------------------------------|------|
| 8       | Файлы                                          | 1011 |
| 8.1     | Файл D:/GIT/nmpp/include/nmblas.h              | 1011 |
| 8.1.1   | Подробное описание                             | 1012 |
| 8.2     | Файл D:/GIT/nmpp/include/nmblas/nmblas_sgemv.h | 1012 |
| 8.2.1   | Подробное описание                             | 1013 |
| 8.2.2   | Макросы                                        | 1014 |
| 8.2.2.1 | ALL_FPU                                        | 1014 |
| 8.3     | Файл D:/GIT/nmpp/include/nmtype.h              | 1014 |
| 8.3.1   | Подробное описание                             | 1017 |
| 8.3.2   | Макросы                                        | 1017 |
| 8.3.2.1 | CAPACITY_nm64s                                 | 1017 |
| 8.3.2.2 | NM16Sx4                                        | 1018 |
| 8.3.2.3 | NM32Sx2                                        | 1018 |
| 8.3.2.4 | VEC_NM16S                                      | 1018 |
| 8.3.2.5 | VEC_NM16U                                      | 1019 |
| 8.3.2.6 | VEC_NM32S                                      | 1019 |
| 8.3.2.7 | VEC_NM4U                                       | 1019 |
| 8.3.2.8 | VEC_NM8S                                       | 1019 |
|         | Алфавитный указатель                           | 1021 |



## Глава 1

## Ошибки

Файл [nmblas.h](#)

Файл [nmblas\\_sgemv.h](#)



## Глава 2

# Алфавитный указатель групп

### 2.1 nmpp

Полный список групп.

|                                 |     |
|---------------------------------|-----|
| BLASS-LEVEL3 . . . . .          | 81  |
| BLASS-LEVEL1 . . . . .          | 82  |
| nmblas_dasum . . . . .          | 84  |
| nmblas_daxpy . . . . .          | 86  |
| nmblas_dcopy . . . . .          | 88  |
| nmblas_ddot . . . . .           | 90  |
| nmblas_dnrm2 . . . . .          | 92  |
| nmblas_drot . . . . .           | 94  |
| nmblas_drotg . . . . .          | 96  |
| nmblas_drotm . . . . .          | 97  |
| nmblas_dscal . . . . .          | 98  |
| nmblas_dsdot . . . . .          | 100 |
| nmblas_dswap . . . . .          | 102 |
| nmblas_dznrm2 . . . . .         | 103 |
| nmblas_idamax . . . . .         | 104 |
| nmblas_isamax . . . . .         | 106 |
| nmblas_sasum . . . . .          | 108 |
| nmblas_saxpy . . . . .          | 110 |
| nmblas_scnrm2 . . . . .         | 112 |
| nmblas_scopy . . . . .          | 114 |
| nmblas_sdot . . . . .           | 116 |
| nmblas_sdsdot . . . . .         | 118 |
| nmblas_snrm2 . . . . .          | 119 |
| nmblas_srot . . . . .           | 121 |
| nmblas_srotm . . . . .          | 122 |
| nmblas_sscal . . . . .          | 123 |
| nmblas_sswap . . . . .          | 125 |
| nmblas_sgemv . . . . .          | 127 |
| nmblas_dgemv . . . . .          | 129 |
| Целочисленные функции . . . . . | 876 |
| Векторные функции . . . . .     | 879 |
| Функции поддержки . . . . .     | 358 |
| nmppsMalloc . . . . .           | 79  |
| nmppsFree . . . . .             | 80  |
| nmppsAddr_ . . . . .            | 828 |

|                                          |     |
|------------------------------------------|-----|
| nmppsSetVal_ . . . . .                   | 838 |
| nmppsGetVal_ . . . . .                   | 849 |
| nmppsGetVal_(return) . . . . .           | 859 |
| Инициализация и копирование . . . . .    | 359 |
| nmppsSet-инициализация . . . . .         | 602 |
| nmppsRandUniform . . . . .               | 614 |
| nmppsRandUniform_ . . . . .              | 628 |
| nmppsRamp_ . . . . .                     | 629 |
| nmppsConvert . . . . .                   | 638 |
| nmppsCopy_ . . . . .                     | 695 |
| nmppsCopyua_ . . . . .                   | 710 |
| nmppsSwap_ . . . . .                     | 713 |
| Vec_ClipRShiftConvert_AddC . . . . .     | 827 |
| Арифметические операции . . . . .        | 361 |
| nmppsAbs . . . . .                       | 374 |
| nmppsAbs1 . . . . .                      | 381 |
| nmppsNeg . . . . .                       | 387 |
| nmppsAddC . . . . .                      | 393 |
| nmppsAddC_p . . . . .                    | 399 |
| nmppsAdd . . . . .                       | 403 |
| nmppsAdd_AddC . . . . .                  | 414 |
| nmppsSubC . . . . .                      | 417 |
| nmppsSubCRev . . . . .                   | 424 |
| nmppsSub . . . . .                       | 430 |
| nmppsAbsDiff . . . . .                   | 439 |
| nmppsAbsDiff1 . . . . .                  | 445 |
| nmppsMulC . . . . .                      | 448 |
| nmppsMul_AddC . . . . .                  | 485 |
| nmppsMulC_AddC . . . . .                 | 489 |
| nmppsRShiftC_MulC_AddC . . . . .         | 491 |
| nmppsMulC_AddV_AddC . . . . .            | 492 |
| nmppsSumN . . . . .                      | 495 |
| nmppsDivC . . . . .                      | 496 |
| nmppsSum . . . . .                       | 499 |
| nmppsSum_1 . . . . .                     | 504 |
| nmppsDotProd_sm . . . . .                | 505 |
| nmppsDotProd . . . . .                   | 514 |
| nmppsWeightedSum . . . . .               | 520 |
| Логические и бинарные операции . . . . . | 363 |
| nmppsNot_ . . . . .                      | 527 |
| nmppsAndC . . . . .                      | 535 |
| nmppsAnd . . . . .                       | 536 |
| nmppsAnd4V_ . . . . .                    | 546 |
| nmppsAndNotV_ . . . . .                  | 547 |
| nmppsOrC . . . . .                       | 550 |
| nmppsOr . . . . .                        | 551 |
| nmppsOr3V_ . . . . .                     | 561 |
| nmppsOr4V_ . . . . .                     | 562 |
| nmppsXorC . . . . .                      | 563 |
| nmppsXor . . . . .                       | 569 |
| nmppsMaskV_ . . . . .                    | 577 |
| nmppsRShiftC . . . . .                   | 580 |
| nmppsRShiftC_ . . . . .                  | 588 |
| nmppsRShiftC_AddC_ . . . . .             | 597 |
| nmppsDisplaceBits . . . . .              | 598 |
| Операции сравнения . . . . .             | 364 |
| nmppsMax_ . . . . .                      | 715 |
| nmppsMin . . . . .                       | 720 |

|                                           |     |
|-------------------------------------------|-----|
| nmppsMaxIndx_ . . . . .                   | 725 |
| nmppsMinIndx_ . . . . .                   | 729 |
| nmppsMinIndxVN_ . . . . .                 | 733 |
| nmppsFirstZeroIndx . . . . .              | 737 |
| nmppsFirstNonZeroIndx . . . . .           | 739 |
| nmppsLastZeroIndx . . . . .               | 741 |
| nmppsLastNonZeroIndx . . . . .            | 742 |
| nmppsMinEvery_ . . . . .                  | 744 |
| nmppsMaxEvery_ . . . . .                  | 751 |
| nmppsMinCmpLtV_ . . . . .                 | 758 |
| nmppsCmpLt0 . . . . .                     | 761 |
| nmppsCmpEq0 . . . . .                     | 769 |
| nmppsCmpMinMaxV_ . . . . .                | 772 |
| nmppsClipPowC_ . . . . .                  | 777 |
| nmppsClipCC_ . . . . .                    | 785 |
| nmppsClipRShiftConvert_AddC_ . . . . .    | 786 |
| nmppsClipConvert_AddC_ . . . . .          | 792 |
| nmppsCmpEqC . . . . .                     | 794 |
| nmppsCmpNe0 . . . . .                     | 797 |
| nmppsCmpNeC . . . . .                     | 803 |
| nmppsCmpNeC_flag . . . . .                | 809 |
| nmppsCmpLtC . . . . .                     | 810 |
| nmppsCmpGtC . . . . .                     | 815 |
| nmppsCmpEqV_ . . . . .                    | 818 |
| nmppsCmpNe . . . . .                      | 819 |
| nmppsCmpLt . . . . .                      | 823 |
| nmppsCmpNeV_ . . . . .                    | 824 |
| Переупорядочивание и сортировка . . . . . | 367 |
| VEC_ QSort . . . . .                      | 867 |
| nmppsRemap_ . . . . .                     | 869 |
| nmppSplitTmp . . . . .                    | 871 |
| nmppSplit . . . . .                       | 872 |
| nmppMerge . . . . .                       | 873 |
| nmppSplit_32fcr . . . . .                 | 874 |
| nmppsDecimate . . . . .                   | 875 |
| Матричные функции . . . . .               | 881 |
| Инициализация и копирование . . . . .     | 204 |
| nmpprmCopyua . . . . .                    | 205 |
| MTR_Copyau . . . . .                      | 213 |
| MTR_Copy . . . . .                        | 216 |
| Функции поддержки . . . . .               | 286 |
| MTR_Malloc . . . . .                      | 257 |
| MTR_Free . . . . .                        | 258 |
| MTR_Addr . . . . .                        | 259 |
| MTR_SetVal . . . . .                      | 268 |
| MTR_GetVal . . . . .                      | 277 |
| Векторно-матричные операции . . . . .     | 287 |
| nmpprmMul_mm . . . . .                    | 230 |
| nmpprmMul_mv_ . . . . .                   | 249 |
| nmpprmMul_mv_8xH . . . . .                | 254 |
| nmpprmMul_mv__AddC . . . . .              | 255 |
| MTR_ProdUnitV . . . . .                   | 256 |
| Функции обработки сигналов . . . . .      | 883 |
| Свертка . . . . .                         | 317 |
| SIG_XCorr . . . . .                       | 324 |
| Масочная фильтрация . . . . .             | 318 |
| SIG_Median3 . . . . .                     | 326 |
| КИХ-фильтрация . . . . .                  | 327 |

|                                              |     |
|----------------------------------------------|-----|
| nmppsFIR_Xs . . . . .                        | 328 |
| nmppsFIRInit_Xs . . . . .                    | 329 |
| nmppsFIRInitAlloc_Xs . . . . .               | 330 |
| nmppsFIRGetStateSize_Xs . . . . .            | 331 |
| nmppsFIRFree . . . . .                       | 332 |
| Изменение размеров . . . . .                 | 319 |
| SIG_ResampleDown2 . . . . .                  | 333 |
| SIG_ResampleUp3Down2 . . . . .               | 334 |
| SIG_CreateResample . . . . .                 | 335 |
| SIG_SetResample . . . . .                    | 336 |
| SIG_Resample_perf . . . . .                  | 337 |
| Быстрое преобразование Фурье . . . . .       | 320 |
| FFT-256 . . . . .                            | 289 |
| IFFT-256 . . . . .                           | 291 |
| FFT-512 . . . . .                            | 294 |
| IFFT-512 . . . . .                           | 296 |
| FFT-1024 . . . . .                           | 299 |
| IFFT-1024 . . . . .                          | 301 |
| FFT-2048 . . . . .                           | 304 |
| IFFT-2048 . . . . .                          | 306 |
| FFT-4096 . . . . .                           | 309 |
| IFFT-4096 . . . . .                          | 311 |
| FFT-8192 . . . . .                           | 313 |
| IFFT-8192 . . . . .                          | 315 |
| Функции обработки изображений . . . . .      | 885 |
| Floodfill . . . . .                          | 182 |
| Переупорядочивание изображений . . . . .     | 190 |
| Блочное переупорядочивание . . . . .         | 191 |
| nmppiSplitIntoBlocks . . . . .               | 192 |
| nmppiMergeFromBlocks . . . . .               | 194 |
| Арифметические действия . . . . .            | 198 |
| Масочная фильтрация . . . . .                | 199 |
| КИХ-фильтрация . . . . .                     | 181 |
| Инициализация и копирование . . . . .        | 200 |
| nmppiConvert . . . . .                       | 188 |
| nmppiRGB32ToGray . . . . .                   | 189 |
| Функции поддержки . . . . .                  | 201 |
| nmppiFree . . . . .                          | 196 |
| nmppiRelease . . . . .                       | 197 |
| Функции графического вывода текста . . . . . | 202 |
| Скалярные функции . . . . .                  | 886 |
| Инициализация . . . . .                      | 174 |
| nmppsRand . . . . .                          | 171 |
| Integer operations . . . . .                 | 175 |
| nmppcSqrt . . . . .                          | 173 |
| Fix-point 64 . . . . .                       | 176 |
| nmppcFixSqrt64 . . . . .                     | 160 |
| nmppcDoubleToFix64 . . . . .                 | 161 |
| nmppcFix64ToDouble . . . . .                 | 162 |
| nmppcFixDiv64 . . . . .                      | 164 |
| nmppcFixSinCos64 . . . . .                   | 166 |
| nmppcFixArcTan64 . . . . .                   | 168 |
| Fix-point 32 . . . . .                       | 177 |
| nmppcFixExp32 . . . . .                      | 133 |
| nmppcFixSinCos32 . . . . .                   | 135 |
| nmppcFixArcTan32 . . . . .                   | 137 |
| nmppcDoubleToFix32 . . . . .                 | 139 |



|                                                  |     |
|--------------------------------------------------|-----|
| nmppcFix32ToDouble . . . . .                     | 140 |
| nmppcFixSqrt32 . . . . .                         | 142 |
| nmppcFixMul32 . . . . .                          | 143 |
| nmppcFixInv32 . . . . .                          | 148 |
| nmppcTblFixArcSin32 . . . . .                    | 150 |
| nmppcTblFixArcCos32 . . . . .                    | 152 |
| nmppcTblFixCos32 . . . . .                       | 154 |
| nmppcTblFixSin32 . . . . .                       | 156 |
| nmppcFixDivMod32 . . . . .                       | 158 |
| nmppcFix64Exp01 . . . . .                        | 169 |
| Арифметические операции . . . . .                | 178 |
| nmppcDivC . . . . .                              | 131 |
| nmppcProdC . . . . .                             | 132 |
| Функции деинтерлейсинга . . . . .                | 179 |
| Базовые регистровые функции библиотеки . . . . . | 888 |
| Элементарные функции . . . . .                   | 892 |
| Vec_0_sub_data . . . . .                         | 895 |
| Vec_activate_data . . . . .                      | 896 |
| Vec_activate_data_add_0 . . . . .                | 897 |
| Vec_activate_data_xor_data . . . . .             | 898 |
| Vec_activate_data_add_ram . . . . .              | 899 |
| Vec_affo . . . . .                               | 901 |
| Vec_data . . . . .                               | 902 |
| Vec_data_add_ram . . . . .                       | 904 |
| Vec_data_and_ram . . . . .                       | 905 |
| Vec_data_or_ram . . . . .                        | 906 |
| Vec_data_sub_ram . . . . .                       | 907 |
| Vec_data_xor_ram . . . . .                       | 908 |
| Vec_And . . . . .                                | 912 |
| Vec_Mask . . . . .                               | 913 |
| Vec_Or . . . . .                                 | 914 |
| Vec_Xor . . . . .                                | 915 |
| Vec_Add . . . . .                                | 917 |
| Vec_Sub . . . . .                                | 932 |
| Vec_not_data . . . . .                           | 938 |
| Vec_ram . . . . .                                | 939 |
| Vec_ram_sub_data . . . . .                       | 940 |
| Vec_vsum_activate_data_0 . . . . .               | 941 |
| функции взвешенного суммирования . . . . .       | 893 |
| Vec_ClipMul2D2W8_AddVr . . . . .                 | 919 |
| Vec_ClipMulNDNW2_AddVr . . . . .                 | 920 |
| Vec_ClipMulNDNW4_AddVr . . . . .                 | 921 |
| Vec_ClipMulNDNW8_AddVr . . . . .                 | 922 |
| Vec_Mul2D2W1_AddVr . . . . .                     | 924 |
| Vec_Mul2D2W2_AddVr . . . . .                     | 925 |
| Vec_Mul2D2W4_AddVr . . . . .                     | 926 |
| Vec_Mul2D2W8_AddVr . . . . .                     | 927 |
| Vec_Mul3D3W2_AddVr . . . . .                     | 928 |
| Vec_Mul3D3W8_AddVr . . . . .                     | 929 |
| Vec_Mul4D4W2_AddVr . . . . .                     | 930 |
| Vec_MulVN_AddVN . . . . .                        | 931 |
| Vec_vsum_data_0 . . . . .                        | 942 |
| Vec_vsum_data_vr . . . . .                       | 944 |
| Vec_vsum_shift_data_0 . . . . .                  | 945 |
| Vec_vsum_shift_data_vr . . . . .                 | 946 |
| Vec_vsum_shift_data_affo . . . . .               | 947 |
| Целевые функции . . . . .                        | 894 |
| Vec_Add_VV_shift . . . . .                       | 900 |

|                                               |     |
|-----------------------------------------------|-----|
| Vec_data_add_afifo . . . . .                  | 903 |
| Vec_FilterCoreRow2 . . . . .                  | 909 |
| Vec_FilterCoreRow4 . . . . .                  | 910 |
| Vec_FilterCoreRow8 . . . . .                  | 911 |
| Vec_Abs . . . . .                             | 916 |
| Vec_ClipExt . . . . .                         | 918 |
| Vec_IncNeg . . . . .                          | 923 |
| Vec_SubAbs . . . . .                          | 933 |
| Vec_SubVN_Abs . . . . .                       | 934 |
| Vec_Swap . . . . .                            | 935 |
| Vec_MUL_2V4toW8_shift . . . . .               | 936 |
| Vec_MUL_2V8toW16_shift . . . . .              | 937 |
| Vec_vsum_data_afifo . . . . .                 | 943 |
| Vec_CompareMinV . . . . .                     | 948 |
| Vec_CompareMaxV . . . . .                     | 949 |
| Vec_DupValueInVector8 . . . . .               | 950 |
| Vec_DupValueInVector16 . . . . .              | 951 |
| Vec_BuildDiagWeights8 . . . . .               | 952 |
| Vec_BuildDiagWeights16 . . . . .              | 953 |
| Vec_MaxVal_v8nm8s . . . . .                   | 954 |
| Vec_MaxVal_v4nm16s . . . . .                  | 955 |
| Vec_MaxVal . . . . .                          | 956 |
| Vec_MinVal_v8nm8s . . . . .                   | 957 |
| Vec_MinVal_v4nm16s . . . . .                  | 958 |
| Vec_MinVal . . . . .                          | 959 |
| Vec_AccMul1D1W32_AddVr . . . . .              | 960 |
| Функции с плавающей точкой . . . . .          | 877 |
| Векторные функции . . . . .                   | 880 |
| Инициализация и копирование . . . . .         | 360 |
| nmppsRandUniform . . . . .                    | 624 |
| nmppsConvert . . . . .                        | 674 |
| nmppsConvert_rounding . . . . .               | 683 |
| nmppsMerge . . . . .                          | 689 |
| nmppsConvertRisc . . . . .                    | 692 |
| Арифметические операции . . . . .             | 362 |
| nmppsAddC . . . . .                           | 401 |
| nmppsAdd . . . . .                            | 411 |
| nmppsSub . . . . .                            | 436 |
| nmppsMulC . . . . .                           | 457 |
| nmppsMul_Mul_Add . . . . .                    | 459 |
| nmppsMul_Add . . . . .                        | 464 |
| nmppsMul_ConjMul_Add . . . . .                | 469 |
| nmppsMulC_AddC . . . . .                      | 472 |
| nmppsMul . . . . .                            | 474 |
| nmppsConjMul . . . . .                        | 477 |
| nmppsMul_Mul_Sub . . . . .                    | 480 |
| nmppsMulC_Add . . . . .                       | 483 |
| nmppsMul_AddC . . . . .                       | 486 |
| Операции сравнения . . . . .                  | 366 |
| nmppsCmpLteC . . . . .                        | 767 |
| nmppsCmpLtC . . . . .                         | 768 |
| nmppsCmpGteC . . . . .                        | 817 |
| nmppsCmpGtC . . . . .                         | 815 |
| Элементарные математические функции . . . . . | 368 |
| Тригонометрические функции . . . . .          | 369 |
| Функция деления векторов . . . . .            | 370 |
| Экспонента . . . . .                          | 371 |
| Натуральный логарифм . . . . .                | 372 |

|                                        |     |
|----------------------------------------|-----|
| Возведение в степень . . . . .         | 373 |
| Матричные функции . . . . .            | 882 |
| Обращение матрицы . . . . .            | 288 |
| MTR_fpResolve_Gauss . . . . .          | 226 |
| MTR_fpResolve_PivotGauss . . . . .     | 227 |
| nmppmLUDecomp . . . . .                | 228 |
| nmppmLUResolve . . . . .               | 229 |
| Функции обработки сигналов . . . . .   | 884 |
| Быстрое преобразование Фурье . . . . . | 321 |
| FFTFwdInitAlloc . . . . .              | 25  |
| FFTInvInitAlloc . . . . .              | 26  |
| DFT-8 . . . . .                        | 27  |
| FFT-16 . . . . .                       | 28  |
| FFT-32 . . . . .                       | 29  |
| FFT-64 . . . . .                       | 32  |
| FFT-128 . . . . .                      | 35  |
| FFT-256 . . . . .                      | 38  |
| FFT-512 . . . . .                      | 41  |
| FFT-1024 . . . . .                     | 44  |
| FFT-2048 . . . . .                     | 47  |
| FFT-4096 . . . . .                     | 50  |
| IDFT-8 . . . . .                       | 52  |
| IFFT-16 . . . . .                      | 53  |
| IFFT-32 . . . . .                      | 54  |
| IFFT-64 . . . . .                      | 57  |
| IFFT-128 . . . . .                     | 60  |
| IFFT-256 . . . . .                     | 63  |
| IFFT-512 . . . . .                     | 66  |
| IFFT-1024 . . . . .                    | 69  |
| IFFT-2048 . . . . .                    | 72  |
| IFFT-4096 . . . . .                    | 75  |
| FFT-Common . . . . .                   | 77  |
| IFFT-Common . . . . .                  | 78  |
| nmblas . . . . .                       | 889 |
| Типы данных . . . . .                  | 878 |
| Типы векторных данных . . . . .        | 338 |
| Типы скалярных данных . . . . .        | 351 |
| контроль переполнения . . . . .        | 890 |



## Глава 3

# Иерархический список классов

### 3.1 Иерархия классов

Иерархия классов.

|                                       |     |
|---------------------------------------|-----|
| C_Allocator32 . . . . .               | 962 |
| C_MultiHeap . . . . .                 | 966 |
| C_BoxImg< T > . . . . .               | 963 |
| C_BoxVec< T > . . . . .               | 963 |
| C_Heap . . . . .                      | 964 |
| C_Img< T > . . . . .                  | 965 |
| C_RingBufferRemote< T > . . . . .     | 970 |
| ds_struct . . . . .                   | 971 |
| EnterHardMode . . . . .               | 972 |
| I_2DSubPixelMinPosition . . . . .     | 972 |
| C_2DSubPixelMinPosition . . . . .     | 961 |
| C_2DTrigSubPixelMinPosition . . . . . | 961 |
| I_PlessyCornerDetector . . . . .      | 973 |
| C_PlessyCornerDetector . . . . .      | 967 |
| C_PlessyCornerDetector_16s . . . . .  | 968 |
| C_PlessyCornerDetector_32f . . . . .  | 969 |
| int15in16x4 . . . . .                 | 973 |
| int30in32x2 . . . . .                 | 973 |
| int31in32x2 . . . . .                 | 974 |
| mtr< T > . . . . .                    | 974 |
| nm16sc . . . . .                      | 976 |
| nmchar . . . . .                      | 976 |
| nmchar1D< N > . . . . .               | 977 |
| nmchar2D< Y, X > . . . . .            | 977 |
| nmintpack< T > . . . . .              | 978 |
| nmnmtr< T > . . . . .                 | 978 |
| NmppsFFTSpec . . . . .                | 980 |
| NmppsFFTSpec_32fcr . . . . .          | 980 |
| NmppsFrame_16s . . . . .              | 981 |
| NmppsFrame_16u . . . . .              | 981 |
| NmppsFrame_32s . . . . .              | 981 |
| NmppsFrame_32u . . . . .              | 982 |
| NmppsFrame_64s . . . . .              | 982 |
| NmppsFrame_64u . . . . .              | 982 |

|                                      |      |
|--------------------------------------|------|
| NmppsFrame_8s . . . . .              | 983  |
| NmppsFrame_8u . . . . .              | 983  |
| NmppsMallocSpec . . . . .            | 984  |
| NmppsTmpSpec . . . . .               | 984  |
| nmreg . . . . .                      | 984  |
| nmshort . . . . .                    | 985  |
| nmshort2D< Y, X > . . . . .          | 985  |
| nmvecpack< T > . . . . .             | 986  |
| RGB32_nm10s . . . . .                | 988  |
| RGB32_nm10u . . . . .                | 988  |
| RGB32_nm8s . . . . .                 | 988  |
| RGB32_nm8u . . . . .                 | 989  |
| RGB64_nm16u . . . . .                | 989  |
| RoundShift . . . . .                 | 990  |
| RPoint . . . . .                     | 990  |
| S_BufferInfo . . . . .               | 990  |
| s_int32x2 . . . . .                  | 991  |
| s_nm32fc . . . . .                   | 992  |
| s_nm32fcr . . . . .                  | 992  |
| s_nm32sc . . . . .                   | 992  |
| s_nm64sc . . . . .                   | 993  |
| S_nmppiFilterKernel . . . . .        | 993  |
| S_nmppiFilterKernel_32s32s . . . . . | 994  |
| s_v16nm16s . . . . .                 | 994  |
| s_v16nm16u . . . . .                 | 995  |
| s_v16nm32s . . . . .                 | 995  |
| s_v16nm32u . . . . .                 | 995  |
| s_v16nm4u . . . . .                  | 996  |
| s_v16nm8s . . . . .                  | 996  |
| s_v16nm8u . . . . .                  | 997  |
| s_v2nm32f . . . . .                  | 997  |
| s_v2nm32s . . . . .                  | 997  |
| s_v2nm32u . . . . .                  | 998  |
| s_v4nm16s . . . . .                  | 998  |
| s_v4nm16u . . . . .                  | 999  |
| s_v4nm32f . . . . .                  | 999  |
| s_v4nm32s . . . . .                  | 999  |
| s_v4nm32u . . . . .                  | 1000 |
| s_v4nm64f . . . . .                  | 1000 |
| s_v4nm8u . . . . .                   | 1000 |
| s_v8nm16s . . . . .                  | 1001 |
| s_v8nm16u . . . . .                  | 1001 |
| s_v8nm32s . . . . .                  | 1002 |
| s_v8nm32u . . . . .                  | 1002 |
| s_v8nm8s . . . . .                   | 1003 |
| s_v8nm8u . . . . .                   | 1003 |
| SpecTmp1 . . . . .                   | 1003 |
| spot_struct . . . . .                | 1004 |
| tagSegmentInfo . . . . .             | 1004 |
| tfixpoint< T, point > . . . . .      | 1005 |
| Tmp2BuffSpec . . . . .               | 1006 |
| uint16ptr . . . . .                  | 1006 |
| uint8ptr . . . . .                   | 1007 |
| v16nm4s . . . . .                    | 1008 |
| v4nm8s . . . . .                     | 1008 |
| vec< T > . . . . .                   | 1009 |

## Глава 4

# Алфавитный указатель классов

### 4.1 Классы

Классы с их кратким описанием.

|                             |     |
|-----------------------------|-----|
| C_2DSubPixelMinPosition     | 961 |
| C_2DTrigSubPixelMinPosition | 961 |
| C_Allocator32               | 962 |
| C_BoxImg< T >               | 963 |
| C_BoxVec< T >               | 963 |
| C_Heap                      |     |
| класс - куча                | 964 |
| C_Img< T >                  | 965 |
| C_MultiHeap                 | 966 |
| C_PlessyCornerDetector      | 967 |
| C_PlessyCornerDetector_16s  | 968 |
| C_PlessyCornerDetector_32f  | 969 |
| C_RingBufferRemote< T >     | 970 |
| ds_struct                   | 971 |
| EnterHardMode               | 972 |
| I_2DSubPixelMinPosition     | 972 |
| I_PlessyCornerDetector      | 973 |
| int15in16x4                 | 973 |
| int30in32x2                 | 973 |
| int31in32x2                 | 974 |
| mtr< T >                    | 974 |
| nm16sc                      | 976 |
| nmchar                      | 976 |
| nmchar1D< N >               | 977 |
| nmchar2D< Y, X >            | 977 |
| nmintpack< T >              | 978 |
| nmctr< T >                  | 978 |
| NmppsFFTSpec                | 980 |
| NmppsFFTSpec_32fcr          | 980 |
| NmppsFrame_16s              | 981 |
| NmppsFrame_16u              | 981 |
| NmppsFrame_32s              | 981 |
| NmppsFrame_32u              | 982 |
| NmppsFrame_64s              | 982 |
| NmppsFrame_64u              | 982 |

|                                                                 |      |
|-----------------------------------------------------------------|------|
| NmppsFrame_8s                                                   | 983  |
| NmppsFrame_8u                                                   | 983  |
| NmppsMallocSpec                                                 | 984  |
| NmppsTmpSpec                                                    | 984  |
| nmreg                                                           | 984  |
| nmshort                                                         | 985  |
| nmshort2D< Y, X >                                               | 985  |
| nmvecpack< T >                                                  | 986  |
| RGB32_nm10s                                                     | 988  |
| RGB32_nm10u                                                     | 988  |
| RGB32_nm8s                                                      | 988  |
| RGB32_nm8u                                                      | 989  |
| RGB64_nm16u                                                     | 989  |
| RoundShift                                                      | 990  |
| RPoint                                                          | 990  |
| S_BufferInfo                                                    |      |
| класс буфер - заголовок в начале выделяемой динамической памяти | 990  |
| s_int32x2                                                       | 991  |
| s_nm32fc                                                        | 992  |
| s_nm32fcr                                                       | 992  |
| s_nm32sc                                                        | 992  |
| s_nm64sc                                                        | 993  |
| S_nmppiFilterKernel                                             | 993  |
| S_nmppiFilterKernel_32s32s                                      | 994  |
| s_v16nm16s                                                      | 994  |
| s_v16nm16u                                                      | 995  |
| s_v16nm32s                                                      | 995  |
| s_v16nm32u                                                      | 995  |
| s_v16nm4u                                                       | 996  |
| s_v16nm8s                                                       | 996  |
| s_v16nm8u                                                       | 997  |
| s_v2nm32f                                                       | 997  |
| s_v2nm32s                                                       | 997  |
| s_v2nm32u                                                       | 998  |
| s_v4nm16s                                                       | 998  |
| s_v4nm16u                                                       | 999  |
| s_v4nm32f                                                       | 999  |
| s_v4nm32s                                                       | 999  |
| s_v4nm32u                                                       | 1000 |
| s_v4nm64f                                                       | 1000 |
| s_v4nm8u                                                        | 1000 |
| s_v8nm16s                                                       | 1001 |
| s_v8nm16u                                                       | 1001 |
| s_v8nm32s                                                       | 1002 |
| s_v8nm32u                                                       | 1002 |
| s_v8nm8s                                                        | 1003 |
| s_v8nm8u                                                        | 1003 |
| SpecTmp1                                                        | 1003 |
| spot_struct                                                     | 1004 |
| tagSegmentInfo                                                  | 1004 |
| tfixpoint< T, point >                                           | 1005 |
| Tmp2BuffSpec                                                    | 1006 |
| uint16ptr                                                       | 1006 |
| uint8ptr                                                        | 1007 |
| v16nm4s                                                         | 1008 |
| v4nm8s                                                          | 1008 |
| vec< T >                                                        | 1009 |



## Глава 5

# Список файлов

### 5.1 Файлы

Полный список документированных файлов.

|                                              |    |
|----------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_1.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_16s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_16u.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_2s.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_2u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_32s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_32u.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_4s.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_4u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_64s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_64u.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_8s.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_Addr_8u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_1.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_16s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_16u.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_2s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_2u.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_32s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_32u.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_4s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_4u.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_64s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_64u.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_8s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_GetVal_8u.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_1.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_16s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_16u.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_2s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_2u.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_32s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_32u.h | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_4s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_4u.h  | ?? |

|                                                       |    |
|-------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_64s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_64u.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_8s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/MTR_SetVal_8u.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_dasum.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_daxpy.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_dcopy.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_ddot.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_dnrm2.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_drot.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_dscal.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_dsdot.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_idamax.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_isamax.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_sasum.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_saxpy.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_scnrm2.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_scopy.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_sdot.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_snrm2.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_sscal.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmblas_sswap.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFix32ToDouble.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFix64Exp01.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFix64ToDouble.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixArcTan32.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixArcTan64.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixDiv32.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixDiv64.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixDivMod32.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixExp32.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixInv32.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixMul32.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixSinCos32.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcFixSinCos64.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcRandMinMaxDiv.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcTblFixArcCos32.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcTblFixArcSin32.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcTblFixCos32.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppcTblFixSin32.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppiMergeFromBlocks8x8.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppiSplitIntoBlocks8x8.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_1.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_16s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_2s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_32s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_4s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_64s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopy_8s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopyau_8s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopyua_16s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopyua_32s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmCopyua_8s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_16s32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_16s64s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_2s64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_32s32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_32s64s.h      | ?? |

|                                                  |    |
|--------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_4s64s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_64s64s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mm_8s64s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mv_16s64s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mv_32s64s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppmMul_mv_8s64s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs1_16s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs1_32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs1_64s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs1_8s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs_16s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs_32s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs_4s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs_64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbs_8s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbsDiff1_8s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbsDiff_16s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbsDiff_32s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbsDiff_64s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAbsDiff_8s.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_16s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_32f.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_32s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_4s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_8s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAdd_AddC_32s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_16s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_32fcr.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_64s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_8s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddC_p64s.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_1.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_16s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_16u.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_2s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_2u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_32f.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_32u.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_4s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_4u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_64f.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_64s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_64u.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_8s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAddr_8u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_1.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_16u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_2u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_32u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_4u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_64u.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAnd_8u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsAndNotV_64u.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsClipPowC_16s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsClipPowC_32s.h  | ?? |

|                                                                |    |
|----------------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppsClipPowC_64s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsClipPowC_8s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsClipRShiftConvertAddC_16s8s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsClipRShiftConvertAddC_32s8s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpEqC_16u15b.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpEqC_4u3b.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpEqC_8u7b.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLt0_16s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLt0_32s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLt0_64s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLt0_8s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLtC_16s15b.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLtC_32s31b.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLtC_64s63b.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpLtC_8s7b.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpMinMaxV_16s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpMinMaxV_32s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpMinMaxV_8s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe0_16s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe0_32s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe0_64s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe0_8s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe_16s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe_32s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe_64s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNe_8s.h                    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeC_16s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeC_32s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeC_64s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeC_8s.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeV_16u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCmpNeV_8u.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConjMul_32fcr.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_16s32s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_16s4s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_16s64s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_16s8s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_16u32u.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_1s2s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_1u2u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_1u4u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_2s1s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_2s4s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_2u4u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f32fcr.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f32s_ceil.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f32s_floor.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f32s_rounding.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f32s_truncate.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32f64f.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32s16s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32s32f.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32s32fcr.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32s64s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32s8s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32sc32fcr.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32u32fcr.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_32u64u.h              | ?? |

|                                                         |    |
|---------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_4s1s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_4s2s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_4s8s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_4u8u.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_64f32f.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_64s16s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_64s32s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8s16s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8s32s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8s4s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8s64s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8u16u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8u32u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvert_8u64u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvertRisc_32u8u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsConvertRisc_8u32u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_16s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_16u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_2s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_32s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_32u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_4s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_4u.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_64s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_64sc.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_64u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_8s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopy_8u.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsCopyua_8s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDivC_32s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_16s16sm.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_16s32sm.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_16s64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_32s32sm.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_32s64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_64s64s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_8s16sm.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_8s32sm.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_8s64s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsDotProd_8s8sm.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT1024Fwd_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT1024Inv_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT128Fwd_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT128Inv_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT2048Fwd_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT2048Inv_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT256Fwd_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT256Inv_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT32Fwd_32fcr.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT32Inv_32fcr.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT4096Fwd_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT4096Inv_32fcr.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT512Fwd_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT512Inv_32fcr.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT64Fwd_32fcr.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFFT64Inv_32fcr.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFirstNonZeroIndx_32s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsFirstZeroIndx_32s.h    | ?? |

|                                                          |    |
|----------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppsFwdShiftBitstream.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_1.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_16s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_16u.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_2s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_2u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_32s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_32u.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_4s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_4u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_8s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGet_8u.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_1.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_16s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_16u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_2s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_2u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_32s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_32u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_4s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_4u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_64s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_64u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_8s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsGetVal_8u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsLastNonZeroIndx_32s.h   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaskV_64u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMax_32s31b.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMax_64s63b.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMax_8s7b.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxEvery_16s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxEvery_32s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxEvery_64s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxEvery_8s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxIndx_32s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMaxIndx_8s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMerge_32f.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMin_32s31b.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMin_64s63b.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMin_8s7b.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinCmpLtV_16s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinEvery_16s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinEvery_32s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinEvery_64s.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinEvery_8s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndx_32s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndx_8s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndxV1024_32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndxV16_32s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndxV256_32s.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndxV25_32s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMinIndxV9_32s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_32fcr.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Add_32f.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Add_32fcr.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Add_64f.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_AddC_32f.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_ConjMul_Add_32fcr.h | ?? |

|                                                            |    |
|------------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Mul_Add_32f.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Mul_Add_32fcr.h       | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Mul_Add_64f.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMul_Mul_Sub_32f.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_16s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_16s32s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_32f.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_32s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_32s64s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_64s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_8s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_8s16s.h              | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_Add_32fcr.h          | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_AddC_32f.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsMulC_AddV_AddC_32s.h      | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNeg_16s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNeg_32s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNeg_64s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNeg_8s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_16u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_2u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_32u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_4u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_64u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsNot_8u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_1.h                    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_16u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_2u.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_32u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_4u.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_64u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsOr_8u.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_1.h                   | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_16s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_16u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_2s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_2u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_32s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_32u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_4s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_4u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_64s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_64u.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_8s.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsPut_8u.h                  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsQSort_32s.h               | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRamp_16s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRamp_32s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRamp_64s.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRamp_8s.h                 | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRand_32f.h                | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_16s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_16u.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_32f_integer.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_32s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_32u.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_64f.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_64s.h         | ?? |

|                                                       |    |
|-------------------------------------------------------|----|
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_64u.h    | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_8s.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRandUniform_8u.h     | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_16s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_16u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_32s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_32u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_4u.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_64s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_64u.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_8s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsRShiftC_8u.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_16s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_16u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_32s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_32u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_64s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_64sp.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_64u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_64up.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_8s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSet_8u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSub_16s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSub_32f.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSub_32s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSub_64s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSub_8s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubC_16s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubC_32s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubC_4s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubC_64s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubC_8s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubCRev_16s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubCRev_32s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubCRev_64s.h        | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSubCRev_8s.h         | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSum_16s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSum_32s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSum_64s.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSum_8s.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsSwap_64s.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsWeightedSum_16s32s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsWeightedSum_32s64s.h | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsWeightedSum_8s16s.h  | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXor_16u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXor_32u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXor_4u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXor_64u.h            | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXor_8u.h             | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXorC_16u.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXorC_32u.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXorC_64u.h           | ?? |
| d:/GIT/nmpp/app/perf/tables/nmppsXorC_8u.h            | ?? |
| D:/GIT/nmpp/include/crtdbg2.h                         | ?? |
| D:/GIT/nmpp/include/fft.h                             | ?? |
| D:/GIT/nmpp/include/fft_32fcr.h                       | ?? |
| D:/GIT/nmpp/include/fftexp.h                          | ?? |
| D:/GIT/nmpp/include/macros_fpu.h                      | ?? |



|                                                                  |      |
|------------------------------------------------------------------|------|
| D:/GIT/nmpp/include/malloc32.h . . . . .                         | ??   |
| D:/GIT/nmpp/include/metric.h . . . . .                           | ??   |
| D:/GIT/nmpp/include/minrep.h . . . . .                           | ??   |
| D:/GIT/nmpp/include/multiheap.h . . . . .                        | ??   |
| D:/GIT/nmpp/include/nmbblas.h                                    |      |
| Nmblas (NeuroMatrix Basic Linear Algebra Subroutines) . . . . .  | 1011 |
| D:/GIT/nmpp/include/nmchar.h . . . . .                           | ??   |
| D:/GIT/nmpp/include/nmdef.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmplc.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmpli.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmplm.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmpls.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmplv.h . . . . .                            | ??   |
| D:/GIT/nmpp/include/nmpp-cpp.h . . . . .                         | ??   |
| D:/GIT/nmpp/include/nmpp.h . . . . .                             | ??   |
| D:/GIT/nmpp/include/nmshort.h . . . . .                          | ??   |
| D:/GIT/nmpp/include/nmtl.h . . . . .                             | ??   |
| D:/GIT/nmpp/include/nmtype.h . . . . .                           | 1014 |
| D:/GIT/nmpp/include/ringremote.h . . . . .                       | ??   |
| D:/GIT/nmpp/include/test.h . . . . .                             | ??   |
| D:/GIT/nmpp/include/tfixpoint.h . . . . .                        | ??   |
| D:/GIT/nmpp/include/nmbblas/nmbblas_sgemm.h                      |      |
| Part of nmbblas library, sgemm function implementation . . . . . | 1012 |
| D:/GIT/nmpp/include/nmplc/cArithmetic.h . . . . .                | ??   |
| D:/GIT/nmpp/include/nmplc/cfixpnt32.h . . . . .                  | ??   |
| D:/GIT/nmpp/include/nmplc/cfixpnt64.h . . . . .                  | ??   |
| D:/GIT/nmpp/include/nmplc/cInit.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmplc/cInteger.h . . . . .                   | ??   |
| D:/GIT/nmpp/include/nmplc/nmplc.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmpli/filter.h . . . . .                     | ??   |
| D:/GIT/nmpp/include/nmpli/iArithmetics.h . . . . .               | ??   |
| D:/GIT/nmpp/include/nmpli/iCellTexture.h . . . . .               | ??   |
| D:/GIT/nmpp/include/nmpli/iDef.h . . . . .                       | ??   |
| D:/GIT/nmpp/include/nmpli/iDeinterlace.h . . . . .               | ??   |
| D:/GIT/nmpp/include/nmpli/iFilter.h . . . . .                    | ??   |
| D:/GIT/nmpp/include/nmpli/iFiltration.h . . . . .                | ??   |
| D:/GIT/nmpp/include/nmpli/iFloodFill.h . . . . .                 | ??   |
| D:/GIT/nmpp/include/nmpli/iInit.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmpli/iPlessy.h . . . . .                    | ??   |
| D:/GIT/nmpp/include/nmpli/iPlessyDetector.h . . . . .            | ??   |
| D:/GIT/nmpp/include/nmpli/iPrint.h . . . . .                     | ??   |
| D:/GIT/nmpp/include/nmpli/iReodering.h . . . . .                 | ??   |
| D:/GIT/nmpp/include/nmpli/iResample.h . . . . .                  | ??   |
| D:/GIT/nmpp/include/nmpli/iSelect.h . . . . .                    | ??   |
| D:/GIT/nmpp/include/nmpli/isubpixel2d.h . . . . .                | ??   |
| D:/GIT/nmpp/include/nmpli/isubpixel2dimpl.h . . . . .            | ??   |
| D:/GIT/nmpp/include/nmpli/iSupport.h . . . . .                   | ??   |
| D:/GIT/nmpp/include/nmpli/nmpli.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmpli/warpimg.h . . . . .                    | ??   |
| D:/GIT/nmpp/include/nmplm/mInit.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmplm/mInverse.h . . . . .                   | ??   |
| D:/GIT/nmpp/include/nmplm/mMatrixVector.h . . . . .              | ??   |
| D:/GIT/nmpp/include/nmplm/mMatrixVectorDev.h . . . . .           | ??   |
| D:/GIT/nmpp/include/nmplm/mSupport.h . . . . .                   | ??   |
| D:/GIT/nmpp/include/nmplm/nmplm.h . . . . .                      | ??   |
| D:/GIT/nmpp/include/nmpls/fft.h . . . . .                        | ??   |
| D:/GIT/nmpp/include/nmpls/fft_old.h . . . . .                    | ??   |
| D:/GIT/nmpp/include/nmpls/fftext.h . . . . .                     | ??   |

|                                                       |    |
|-------------------------------------------------------|----|
| D:/GIT/nmpp/include/nmpls/fftref.h . . . . .          | ?? |
| D:/GIT/nmpp/include/nmpls/nmpls.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmpls/sCorrelation.h . . . . .    | ?? |
| D:/GIT/nmpp/include/nmpls/sFiltration.h . . . . .     | ?? |
| D:/GIT/nmpp/include/nmpls/sfir.h . . . . .            | ?? |
| D:/GIT/nmpp/include/nmpls/sResample.h . . . . .       | ?? |
| D:/GIT/nmpp/include/nmplv/nmplv.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmplv/nmtl.h . . . . .            | ?? |
| D:/GIT/nmpp/include/nmplv/sElementary.h . . . . .     | ?? |
| D:/GIT/nmpp/include/nmplv/vArithmetics.h . . . . .    | ?? |
| D:/GIT/nmpp/include/nmplv/vArithmeticsDev.h . . . . . | ?? |
| D:/GIT/nmpp/include/nmplv/vBitwise.h . . . . .        | ?? |
| D:/GIT/nmpp/include/nmplv/vInit.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmplv/vSelect.h . . . . .         | ?? |
| D:/GIT/nmpp/include/nmplv/vStat.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmplv/vSupport.h . . . . .        | ?? |
| D:/GIT/nmpp/include/nmplv/vTransform.h . . . . .      | ?? |
| D:/GIT/nmpp/include/nmtl/nmtl.h . . . . .             | ?? |
| D:/GIT/nmpp/include/nmtl/nmtlio.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tcmplx.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tcmplx_spec.h . . . . .      | ?? |
| D:/GIT/nmpp/include/nmtl/tfixpoint.h . . . . .        | ?? |
| D:/GIT/nmpp/include/nmtl/tfixpointmath.h . . . . .    | ?? |
| D:/GIT/nmpp/include/nmtl/tmatrix.h . . . . .          | ?? |
| D:/GIT/nmpp/include/nmtl/tmatrix_spec.h . . . . .     | ?? |
| D:/GIT/nmpp/include/nmtl/tnmvec.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tnmint.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tnmmtr.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tnmvec.h . . . . .           | ?? |
| D:/GIT/nmpp/include/nmtl/tnmvecpack.h . . . . .       | ?? |
| D:/GIT/nmpp/include/nmtl/tvector.h . . . . .          | ?? |
| D:/GIT/nmpp/include/nmvcore/vCore.h . . . . .         | ?? |
| D:/GIT/nmpp/include/rpc/aura-exports.c . . . . .      | ?? |
| D:/GIT/nmpp/include/rpc/rpc-host.h . . . . .          | ?? |
| D:/GIT/nmpp/include/rpc/rpc-nmc-func.h . . . . .      | ?? |
| D:/GIT/nmpp/include/rpc/rpc-nmc.h . . . . .           | ?? |

## Глава 6

# Группы

### 6.1 FFTFwdInitAlloc

Функции инициализации структур коэффициентов, необходимых для вычисления прямого БПФ

#### Функции

- `int nmppsFFT16FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT32FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT64FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT128FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT256FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT512FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT1024FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT2048FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT4096FwdInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`

#### 6.1.1 Подробное описание

Функции инициализации структур коэффициентов, необходимых для вычисления прямого БПФ

#### Аргументы

|    |      |                                              |
|----|------|----------------------------------------------|
| in | addr | двойной указатель на структуру коэффициентов |
|----|------|----------------------------------------------|

#### Возвращает

Функции возвращают 0 в случае успешной инициализации и число, отличное от нуля, в случае ошибок

## 6.2 FFTInvInitAlloc

Функции инициализации структур коэффициентов, необходимых для вычисления обратного БПФ

Функции

- `int nmppsFFT16InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT32InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT64InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT128InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT256InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT512InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT1024InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT2048InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`
- `int nmppsFFT4096InvInitAlloc_32fcr (NmppsFFTSpec_32fcr **addr)`

### 6.2.1 Подробное описание

Функции инициализации структур коэффициентов, необходимых для вычисления обратного БПФ

Аргументы

|    |      |                                              |
|----|------|----------------------------------------------|
| in | addr | двойной указатель на структуру коэффициентов |
|----|------|----------------------------------------------|

Возвращает

Функции возвращают 0 в случае успешной инициализации и число, отличное от нуля, в случае ошибок

## 6.3 DFT-8

Функция для вычисления прямого ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

Функции

- void `nmppsDFT8Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

Функция для вычисления прямого ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

### 6.3.1 Подробное описание

Функция для вычисления прямого ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

### 6.3.2 Функции

#### 6.3.2.1 nmppsDFT8Fwd\_32fcr()

```
void nmppsDFT8Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Функция для вычисления прямого ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

## 6.4 FFT-16

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

Функции

- void nmppsFFT16Fwd\_32fcr (const [nm32fcr](#) \*pSrcVec, [nm32fcr](#) \*pDstVec, [NmppsFFTSpec\\_32fcr](#) \*spec)

### 6.4.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

## 6.5 FFT-32

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

Функции

- void `nmppsFFT32Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.5.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

### 6.5.2 Функции

#### 6.5.2.1 nmppsFFT32Fwd\_32fcr()

```
void nmppsFFT32Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 333   | 10.40      |
| im1     | im1     | 333   | 10.40      |
| im1     | im2     | 333   | 10.40      |
| im1     | im2     | 333   | 10.40      |
| im1     | im3     | 333   | 10.40      |
| im1     | im3     | 333   | 10.40      |
| im1     | im4     | 333   | 10.40      |
| im1     | im4     | 333   | 10.40      |
| im1     | im5     | 333   | 10.40      |
| im1     | im5     | 333   | 10.40      |
| im2     | im1     | 333   | 10.40      |
| im2     | im1     | 333   | 10.40      |
| im2     | im2     | 333   | 10.40      |
| im2     | im2     | 333   | 10.40      |
| im2     | im3     | 333   | 10.40      |
| im2     | im3     | 333   | 10.40      |
| im2     | im4     | 333   | 10.40      |
| im2     | im4     | 333   | 10.40      |
| im2     | im5     | 333   | 10.40      |
| im2     | im5     | 333   | 10.40      |
| im3     | im1     | 389   | 12.15      |
| im3     | im1     | 389   | 12.15      |
| im3     | im2     | 389   | 12.15      |
| im3     | im2     | 389   | 12.15      |
| im3     | im3     | 389   | 12.15      |
| im3     | im3     | 389   | 12.15      |
| im3     | im4     | 389   | 12.15      |
| im3     | im4     | 389   | 12.15      |
| im3     | im5     | 389   | 12.15      |
| im3     | im5     | 389   | 12.15      |
| im4     | im1     | 360   | 11.25      |
| im4     | im1     | 360   | 11.25      |
| im4     | im2     | 360   | 11.25      |
| im4     | im2     | 360   | 11.25      |
| im4     | im3     | 360   | 11.25      |
| im4     | im3     | 360   | 11.25      |
| im4     | im4     | 360   | 11.25      |
| im4     | im4     | 360   | 11.25      |
| im4     | im5     | 360   | 11.25      |
| im4     | im5     | 360   | 11.25      |
| im5     | im1     | 333   | 10.40      |
| im5     | im1     | 333   | 10.40      |
| im5     | im2     | 333   | 10.40      |
| im5     | im2     | 333   | 10.40      |
| im5     | im3     | 333   | 10.40      |
| im5     | im3     | 333   | 10.40      |
| im5     | im4     | 333   | 10.40      |
| im5     | im4     | 333   | 10.40      |
| im5     | im5     | 333   | 10.40      |
| im5     | im5     | 333   | 10.40      |

The best configuration:



| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 333   | 10.40      |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im1     | 389   | 12.15      |

## 6.6 FFT-64

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

Функции

- void `nmppsFFT64Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.6.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

### 6.6.2 Функции

#### 6.6.2.1 nmppsFFT64Fwd\_32fcr()

```
void nmppsFFT64Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 520   | 8.12       |
| im1     | im1     | 520   | 8.12       |
| im1     | im2     | 520   | 8.12       |
| im1     | im2     | 520   | 8.12       |
| im1     | im3     | 520   | 8.12       |
| im1     | im3     | 520   | 8.12       |
| im1     | im4     | 520   | 8.12       |
| im1     | im4     | 520   | 8.12       |
| im1     | im5     | 520   | 8.12       |
| im1     | im5     | 520   | 8.12       |
| im2     | im1     | 520   | 8.12       |
| im2     | im1     | 520   | 8.12       |
| im2     | im2     | 520   | 8.12       |
| im2     | im2     | 520   | 8.12       |
| im2     | im3     | 520   | 8.12       |
| im2     | im3     | 520   | 8.12       |
| im2     | im4     | 520   | 8.12       |
| im2     | im4     | 520   | 8.12       |
| im2     | im5     | 520   | 8.12       |
| im2     | im5     | 520   | 8.12       |
| im3     | im1     | 630   | 9.84       |
| im3     | im1     | 630   | 9.84       |
| im3     | im2     | 630   | 9.84       |
| im3     | im2     | 630   | 9.84       |
| im3     | im3     | 630   | 9.84       |
| im3     | im3     | 630   | 9.84       |
| im3     | im4     | 630   | 9.84       |
| im3     | im4     | 630   | 9.84       |
| im3     | im5     | 630   | 9.84       |
| im3     | im5     | 630   | 9.84       |
| im4     | im1     | 601   | 9.39       |
| im4     | im1     | 601   | 9.39       |
| im4     | im2     | 601   | 9.39       |
| im4     | im2     | 601   | 9.39       |
| im4     | im3     | 601   | 9.39       |
| im4     | im3     | 601   | 9.39       |
| im4     | im4     | 601   | 9.39       |
| im4     | im4     | 601   | 9.39       |
| im4     | im5     | 601   | 9.39       |
| im4     | im5     | 601   | 9.39       |
| im5     | im1     | 520   | 8.12       |
| im5     | im1     | 520   | 8.12       |
| im5     | im2     | 520   | 8.12       |
| im5     | im2     | 520   | 8.12       |
| im5     | im3     | 520   | 8.12       |
| im5     | im3     | 520   | 8.12       |
| im5     | im4     | 520   | 8.12       |
| im5     | im4     | 520   | 8.12       |
| im5     | im5     | 520   | 8.12       |
| im5     | im5     | 520   | 8.12       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 520   | 8.12       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im1     | 630   | 9.84       |

## 6.7 FFT-128

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

Функции

- void `nmppsFFT128Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.7.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

### 6.7.2 Функции

#### 6.7.2.1 nmppsFFT128Fwd\_32fcr()

```
void nmppsFFT128Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 1091  | 8.52       |
| im1     | im1     | 1091  | 8.52       |
| im1     | im2     | 1091  | 8.52       |
| im1     | im2     | 1091  | 8.52       |
| im1     | im3     | 1091  | 8.52       |
| im1     | im3     | 1091  | 8.52       |
| im1     | im4     | 1091  | 8.52       |
| im1     | im4     | 1091  | 8.52       |
| im1     | im5     | 1091  | 8.52       |
| im1     | im5     | 1091  | 8.52       |
| im2     | im1     | 1091  | 8.52       |
| im2     | im1     | 1091  | 8.52       |
| im2     | im2     | 1091  | 8.52       |
| im2     | im2     | 1091  | 8.52       |
| im2     | im3     | 1091  | 8.52       |
| im2     | im3     | 1091  | 8.52       |
| im2     | im4     | 1091  | 8.52       |
| im2     | im4     | 1091  | 8.52       |
| im2     | im5     | 1091  | 8.52       |
| im2     | im5     | 1091  | 8.52       |
| im3     | im1     | 1309  | 10.22      |
| im3     | im1     | 1309  | 10.22      |
| im3     | im2     | 1309  | 10.22      |
| im3     | im2     | 1309  | 10.22      |
| im3     | im3     | 1309  | 10.22      |
| im3     | im3     | 1309  | 10.22      |
| im3     | im4     | 1309  | 10.22      |
| im3     | im4     | 1309  | 10.22      |
| im3     | im5     | 1309  | 10.22      |
| im3     | im5     | 1309  | 10.22      |
| im4     | im1     | 1280  | 10.00      |
| im4     | im1     | 1280  | 10.00      |
| im4     | im2     | 1280  | 10.00      |
| im4     | im2     | 1280  | 10.00      |
| im4     | im3     | 1280  | 10.00      |
| im4     | im3     | 1280  | 10.00      |
| im4     | im4     | 1280  | 10.00      |
| im4     | im4     | 1280  | 10.00      |
| im4     | im5     | 1280  | 10.00      |
| im4     | im5     | 1280  | 10.00      |
| im5     | im1     | 1091  | 8.52       |
| im5     | im1     | 1091  | 8.52       |
| im5     | im2     | 1091  | 8.52       |
| im5     | im2     | 1091  | 8.52       |
| im5     | im3     | 1091  | 8.52       |
| im5     | im3     | 1091  | 8.52       |
| im5     | im4     | 1091  | 8.52       |
| im5     | im4     | 1091  | 8.52       |
| im5     | im5     | 1091  | 8.52       |
| im5     | im5     | 1091  | 8.52       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 1091  | 8.52       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im1     | 1309  | 10.22      |

## 6.8 FFT-256

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

Функции

- void `nmppsFFT256Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.8.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (1763 такта) необходимо положить входной вектор в 1-ый банк, выходной вектор в 5-ый банк

### 6.8.2 Функции

#### 6.8.2.1 nmppsFFT256Fwd\_32fcr()

```
void nmppsFFT256Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0



| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 1819  | 7.10       |
| im1     | im1     | 1819  | 7.10       |
| im1     | im2     | 1763  | 6.88       |
| im1     | im2     | 1763  | 6.88       |
| im1     | im3     | 1763  | 6.88       |
| im1     | im3     | 1763  | 6.88       |
| im1     | im4     | 1763  | 6.88       |
| im1     | im4     | 1763  | 6.88       |
| im1     | im5     | 1763  | 6.88       |
| im1     | im5     | 1763  | 6.88       |
| im2     | im1     | 1910  | 7.46       |
| im2     | im1     | 1910  | 7.46       |
| im2     | im2     | 1854  | 7.24       |
| im2     | im2     | 1854  | 7.24       |
| im2     | im3     | 1854  | 7.24       |
| im2     | im3     | 1854  | 7.24       |
| im2     | im4     | 1854  | 7.24       |
| im2     | im4     | 1854  | 7.24       |
| im2     | im5     | 1854  | 7.24       |
| im2     | im5     | 1854  | 7.24       |
| im3     | im1     | 1906  | 7.44       |
| im3     | im1     | 1906  | 7.44       |
| im3     | im2     | 1850  | 7.22       |
| im3     | im2     | 1850  | 7.22       |
| im3     | im3     | 1850  | 7.22       |
| im3     | im3     | 1850  | 7.22       |
| im3     | im4     | 1850  | 7.22       |
| im3     | im4     | 1850  | 7.22       |
| im3     | im5     | 1850  | 7.22       |
| im3     | im5     | 1850  | 7.22       |
| im4     | im1     | 1896  | 7.40       |
| im4     | im1     | 1896  | 7.40       |
| im4     | im2     | 1840  | 7.18       |
| im4     | im2     | 1840  | 7.18       |
| im4     | im3     | 1840  | 7.18       |
| im4     | im3     | 1840  | 7.18       |
| im4     | im4     | 1840  | 7.18       |
| im4     | im4     | 1840  | 7.18       |
| im4     | im5     | 1840  | 7.18       |
| im4     | im5     | 1840  | 7.18       |
| im5     | im1     | 1819  | 7.10       |
| im5     | im1     | 1819  | 7.10       |
| im5     | im2     | 1763  | 6.88       |
| im5     | im2     | 1763  | 6.88       |
| im5     | im3     | 1763  | 6.88       |
| im5     | im3     | 1763  | 6.88       |
| im5     | im4     | 1763  | 6.88       |
| im5     | im4     | 1763  | 6.88       |
| im5     | im5     | 1763  | 6.88       |
| im5     | im5     | 1763  | 6.88       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im2     | 1763  | 6.88       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im1     | 1910  | 7.46       |

## 6.9 FFT-512

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

### Функции

- void `nmppsFFT512Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.9.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

#### Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

#### Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

#### Аргументы

|    |      |                                                                                                 |
|----|------|-------------------------------------------------------------------------------------------------|
| in | spec | структра, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|-------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (3675 такта) необходимо положить входной вектор в 2-ый банк, выходной вектор в 5-ый банк

### 6.9.2 Функции

#### 6.9.2.1 nmppsFFT512Fwd\_32fcr()

```
void nmppsFFT512Fwd_32fcr (  
    const nm32fcr * pSrcVec,  
    nm32fcr * pDstVec,  
    NmppsFFTSpec_32fcr * spec )
```

Perfomance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 3858  | 7.53       |
| im1     | im1     | 3858  | 7.53       |
| im1     | im2     | 3976  | 7.76       |
| im1     | im2     | 3976  | 7.76       |
| im1     | im3     | 3858  | 7.53       |
| im1     | im3     | 3858  | 7.53       |
| im1     | im4     | 3858  | 7.53       |
| im1     | im4     | 3858  | 7.53       |
| im1     | im5     | 3858  | 7.53       |
| im1     | im5     | 3858  | 7.53       |
| im2     | im1     | 3675  | 7.17       |
| im2     | im1     | 3675  | 7.17       |
| im2     | im2     | 3793  | 7.40       |
| im2     | im2     | 3793  | 7.40       |
| im2     | im3     | 3675  | 7.17       |
| im2     | im3     | 3675  | 7.17       |
| im2     | im4     | 3675  | 7.17       |
| im2     | im4     | 3675  | 7.17       |
| im2     | im5     | 3675  | 7.17       |
| im2     | im5     | 3675  | 7.17       |
| im3     | im1     | 3858  | 7.53       |
| im3     | im1     | 3858  | 7.53       |
| im3     | im2     | 3976  | 7.76       |
| im3     | im2     | 3976  | 7.76       |
| im3     | im3     | 3858  | 7.53       |
| im3     | im3     | 3858  | 7.53       |
| im3     | im4     | 3858  | 7.53       |
| im3     | im4     | 3858  | 7.53       |
| im3     | im5     | 3858  | 7.53       |
| im3     | im5     | 3858  | 7.53       |
| im4     | im1     | 3848  | 7.51       |
| im4     | im1     | 3848  | 7.51       |
| im4     | im2     | 3966  | 7.74       |
| im4     | im2     | 3966  | 7.74       |
| im4     | im3     | 3848  | 7.51       |
| im4     | im3     | 3848  | 7.51       |
| im4     | im4     | 3848  | 7.51       |
| im4     | im4     | 3848  | 7.51       |
| im4     | im5     | 3848  | 7.51       |
| im4     | im5     | 3848  | 7.51       |
| im5     | im1     | 3675  | 7.17       |
| im5     | im1     | 3675  | 7.17       |
| im5     | im2     | 3793  | 7.40       |
| im5     | im2     | 3793  | 7.40       |
| im5     | im3     | 3675  | 7.17       |
| im5     | im3     | 3675  | 7.17       |
| im5     | im4     | 3675  | 7.17       |
| im5     | im4     | 3675  | 7.17       |
| im5     | im5     | 3675  | 7.17       |
| im5     | im5     | 3675  | 7.17       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im1     | 3675  | 7.17       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im2     | 3976  | 7.76       |

## 6.10 FFT-1024

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 1024 комплексных чисел

Функции

- void `nmppsFFT1024Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.10.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 1024 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (8655 такта) необходимо положить входной вектор в 6-ый банк, выходной вектор в 5-ый банк

### 6.10.2 Функции

#### 6.10.2.1 nmppsFFT1024Fwd\_32fcr()

```
void nmppsFFT1024Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im1      | 10749 | 10.49      |
| im1      | im1      | 10749 | 10.49      |
| im1      | im2      | 9687  | 9.45       |
| im1      | im2      | 9687  | 9.45       |
| im1      | im3      | 10800 | 10.54      |
| im1      | im3      | 10800 | 10.54      |
| im1      | im4      | 11110 | 10.84      |
| im1      | im4      | 11110 | 10.84      |
| im1      | im5      | 8988  | 8.77       |
| im1      | im5      | 8988  | 8.77       |
| im2      | im1      | 10687 | 10.43      |
| im2      | im1      | 10687 | 10.43      |
| im2      | im2      | 9625  | 9.39       |
| im2      | im2      | 9625  | 9.39       |
| im2      | im3      | 10738 | 10.48      |
| im2      | im3      | 10738 | 10.48      |
| im2      | im4      | 11048 | 10.78      |
| im2      | im4      | 11048 | 10.78      |
| im2      | im5      | 8926  | 8.71       |
| im2      | im5      | 8926  | 8.71       |
| im3      | im1      | 11214 | 10.95      |
| im3      | im1      | 11214 | 10.95      |
| im3      | im2      | 10152 | 9.91       |
| im3      | im2      | 10152 | 9.91       |
| im3      | im3      | 11265 | 11.00      |
| im3      | im3      | 11265 | 11.00      |
| im3      | im4      | 11575 | 11.30      |
| im3      | im4      | 11575 | 11.30      |
| im3      | im5      | 9453  | 9.23       |
| im3      | im5      | 9453  | 9.23       |
| im4      | im1      | 10408 | 10.16      |
| im4      | im1      | 10408 | 10.16      |
| im4      | im2      | 9346  | 9.12       |
| im4      | im2      | 9346  | 9.12       |
| im4      | im3      | 10459 | 10.21      |
| im4      | im3      | 10459 | 10.21      |
| im4      | im4      | 10769 | 10.51      |
| im4      | im4      | 10769 | 10.51      |
| im4      | im5      | 8647  | 8.44       |
| im4      | im5      | 8647  | 8.44       |
| im5      | im1      | 10408 | 10.16      |
| im5      | im1      | 10408 | 10.16      |
| im5      | im2      | 9346  | 9.12       |
| im5      | im2      | 9346  | 9.12       |
| im5      | im3      | 10459 | 10.21      |
| im5      | im3      | 10459 | 10.21      |
| im5      | im4      | 10769 | 10.51      |
| im5      | im4      | 10769 | 10.51      |
| im5      | im5      | 8647  | 8.44       |
| im5      | im5      | 8647  | 8.44       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im4     | im5     | 8647  | 8.44       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im4     | 11575 | 11.30      |



## 6.11 FFT-2048

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

### Функции

- void `nmppsFFT2048Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

#### 6.11.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

##### Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

##### Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

##### Аргументы

|    |      |                                                                                                 |
|----|------|-------------------------------------------------------------------------------------------------|
| in | spec | структра, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|-------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (19504 такта) необходимо положить входной вектор в 5-ый банк, выходной вектор в 5-ый банк

#### 6.11.2 Функции

##### 6.11.2.1 nmppsFFT2048Fwd\_32fcr()

```
void nmppsFFT2048Fwd_32fcr (  
    const nm32fcr * pSrcVec,  
    nm32fcr * pDstVec,  
    NmppsFFTSpec_32fcr * spec )
```

Perfomance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im1      | 21217 | 10.35      |
| im1      | im1      | 21217 | 10.35      |
| im1      | im2      | 24180 | 11.80      |
| im1      | im2      | 24180 | 11.80      |
| im1      | im3      | 24237 | 11.83      |
| im1      | im3      | 24237 | 11.83      |
| im1      | im4      | 24602 | 12.01      |
| im1      | im4      | 24602 | 12.01      |
| im1      | im5      | 20071 | 9.80       |
| im1      | im5      | 20071 | 9.80       |
| im2      | im1      | 21344 | 10.42      |
| im2      | im1      | 21344 | 10.42      |
| im2      | im2      | 24307 | 11.86      |
| im2      | im2      | 24307 | 11.86      |
| im2      | im3      | 24364 | 11.89      |
| im2      | im3      | 24364 | 11.89      |
| im2      | im4      | 24729 | 12.07      |
| im2      | im4      | 24729 | 12.07      |
| im2      | im5      | 20198 | 9.86       |
| im2      | im5      | 20198 | 9.86       |
| im3      | im1      | 22288 | 10.88      |
| im3      | im1      | 22288 | 10.88      |
| im3      | im2      | 25251 | 12.32      |
| im3      | im2      | 25251 | 12.32      |
| im3      | im3      | 25308 | 12.35      |
| im3      | im3      | 25308 | 12.35      |
| im3      | im4      | 25673 | 12.53      |
| im3      | im4      | 25673 | 12.53      |
| im3      | im5      | 21142 | 10.32      |
| im3      | im5      | 21142 | 10.32      |
| im4      | im1      | 20650 | 10.08      |
| im4      | im1      | 20650 | 10.08      |
| im4      | im2      | 23613 | 11.52      |
| im4      | im2      | 23613 | 11.52      |
| im4      | im3      | 23670 | 11.55      |
| im4      | im3      | 23670 | 11.55      |
| im4      | im4      | 24035 | 11.73      |
| im4      | im4      | 24035 | 11.73      |
| im4      | im5      | 19504 | 9.52       |
| im4      | im5      | 19504 | 9.52       |
| im5      | im1      | 20650 | 10.08      |
| im5      | im1      | 20650 | 10.08      |
| im5      | im2      | 23613 | 11.52      |
| im5      | im2      | 23613 | 11.52      |
| im5      | im3      | 23670 | 11.55      |
| im5      | im3      | 23670 | 11.55      |
| im5      | im4      | 24035 | 11.73      |
| im5      | im4      | 24035 | 11.73      |
| im5      | im5      | 19504 | 9.52       |
| im5      | im5      | 19504 | 9.52       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im4     | im5     | 19504 | 9.52       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im4     | 25673 | 12.53      |

## 6.12 FFT-4096

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

Функции

- void `nmppsFFT4096Fwd_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.12.1 Подробное описание

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (54258 такта) необходимо положить входной вектор в 5-ый банк, выходной вектор в 5-ый банк

### 6.12.2 Функции

#### 6.12.2.1 nmppsFFT4096Fwd\_32fcr()

```
void nmppsFFT4096Fwd_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im2      | 56336 | 13.75      |
| im1      | im2      | 56336 | 13.75      |
| im1      | im3      | 57946 | 14.14      |
| im1      | im3      | 57946 | 14.14      |
| im1      | im4      | 58258 | 14.22      |
| im1      | im4      | 58258 | 14.22      |
| im1      | im5      | 56336 | 13.75      |
| im1      | im5      | 56336 | 13.75      |
| im3      | im2      | 54258 | 13.24      |
| im3      | im2      | 54258 | 13.24      |
| im3      | im3      | 55868 | 13.63      |
| im3      | im3      | 55868 | 13.63      |
| im3      | im4      | 56180 | 13.71      |
| im3      | im4      | 56180 | 13.71      |
| im3      | im5      | 54258 | 13.24      |
| im3      | im5      | 54258 | 13.24      |
| im4      | im2      | 54258 | 13.24      |
| im4      | im2      | 54258 | 13.24      |
| im4      | im3      | 55868 | 13.63      |
| im4      | im3      | 55868 | 13.63      |
| im4      | im4      | 56180 | 13.71      |
| im4      | im4      | 56180 | 13.71      |
| im4      | im5      | 54258 | 13.24      |
| im4      | im5      | 54258 | 13.24      |
| im5      | im2      | 54258 | 13.24      |
| im5      | im2      | 54258 | 13.24      |
| im5      | im3      | 55868 | 13.63      |
| im5      | im3      | 55868 | 13.63      |
| im5      | im4      | 56180 | 13.71      |
| im5      | im4      | 56180 | 13.71      |
| im5      | im5      | 54258 | 13.24      |
| im5      | im5      | 54258 | 13.24      |

The best configuration:

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im3      | im2      | 54258 | 13.24      |

The worst configuration:

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im4      | 58258 | 14.22      |

## 6.13 IDFT-8

Функция для вычисления обратного ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

Функции

- void nmppsDFT8Inv\_32fcr (const nm32fcr \*pSrcVec, nm32fcr \*pDstVec, NmppsFFTSpec\_32fcr \*spec)

### 6.13.1 Подробное описание

Функция для вычисления обратного ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

## 6.14 IFFT-16

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

Функции

- void nmppsFFT16Inv\_32fcr (const nm32fcr \*pSrcVec, nm32fcr \*pDstVec, NmppsFFTSpec\_32fcr \*spec)

### 6.14.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

## 6.15 IFFT-32

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

Функции

- void `nmppsFFT32Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.15.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

### 6.15.2 Функции

#### 6.15.2.1 nmppsFFT32Inv\_32fcr()

```
void nmppsFFT32Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0



| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 333   | 10.40      |
| im1     | im1     | 333   | 10.40      |
| im1     | im2     | 333   | 10.40      |
| im1     | im2     | 333   | 10.40      |
| im1     | im3     | 333   | 10.40      |
| im1     | im3     | 333   | 10.40      |
| im1     | im4     | 333   | 10.40      |
| im1     | im4     | 333   | 10.40      |
| im1     | im5     | 333   | 10.40      |
| im1     | im5     | 333   | 10.40      |
| im2     | im1     | 333   | 10.40      |
| im2     | im1     | 333   | 10.40      |
| im2     | im2     | 333   | 10.40      |
| im2     | im2     | 333   | 10.40      |
| im2     | im3     | 333   | 10.40      |
| im2     | im3     | 333   | 10.40      |
| im2     | im4     | 333   | 10.40      |
| im2     | im4     | 333   | 10.40      |
| im2     | im5     | 333   | 10.40      |
| im2     | im5     | 333   | 10.40      |
| im3     | im1     | 389   | 12.15      |
| im3     | im1     | 389   | 12.15      |
| im3     | im2     | 389   | 12.15      |
| im3     | im2     | 389   | 12.15      |
| im3     | im3     | 389   | 12.15      |
| im3     | im3     | 389   | 12.15      |
| im3     | im4     | 389   | 12.15      |
| im3     | im4     | 389   | 12.15      |
| im3     | im5     | 389   | 12.15      |
| im3     | im5     | 389   | 12.15      |
| im4     | im1     | 360   | 11.25      |
| im4     | im1     | 360   | 11.25      |
| im4     | im2     | 360   | 11.25      |
| im4     | im2     | 360   | 11.25      |
| im4     | im3     | 360   | 11.25      |
| im4     | im3     | 360   | 11.25      |
| im4     | im4     | 360   | 11.25      |
| im4     | im4     | 360   | 11.25      |
| im4     | im5     | 360   | 11.25      |
| im4     | im5     | 360   | 11.25      |
| im5     | im1     | 333   | 10.40      |
| im5     | im1     | 333   | 10.40      |
| im5     | im2     | 333   | 10.40      |
| im5     | im2     | 333   | 10.40      |
| im5     | im3     | 333   | 10.40      |
| im5     | im3     | 333   | 10.40      |
| im5     | im4     | 333   | 10.40      |
| im5     | im4     | 333   | 10.40      |
| im5     | im5     | 333   | 10.40      |
| im5     | im5     | 333   | 10.40      |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 333   | 10.40      |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im1     | 389   | 12.15      |

## 6.16 IFFT-64

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

Функции

- void `nmppsFFT64Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.16.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

### 6.16.2 Функции

#### 6.16.2.1 nmppsFFT64Inv\_32fcr()

```
void nmppsFFT64Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 520   | 8.12       |
| im1     | im1     | 520   | 8.12       |
| im1     | im2     | 520   | 8.12       |
| im1     | im2     | 520   | 8.12       |
| im1     | im3     | 520   | 8.12       |
| im1     | im3     | 520   | 8.12       |
| im1     | im4     | 520   | 8.12       |
| im1     | im4     | 520   | 8.12       |
| im1     | im5     | 520   | 8.12       |
| im1     | im5     | 520   | 8.12       |
| im2     | im1     | 520   | 8.12       |
| im2     | im1     | 520   | 8.12       |
| im2     | im2     | 520   | 8.12       |
| im2     | im2     | 520   | 8.12       |
| im2     | im3     | 520   | 8.12       |
| im2     | im3     | 520   | 8.12       |
| im2     | im4     | 3105  | 48.51      |
| im2     | im4     | 3105  | 48.51      |
| im2     | im5     | 520   | 8.12       |
| im2     | im5     | 520   | 8.12       |
| im3     | im1     | 630   | 9.84       |
| im3     | im1     | 630   | 9.84       |
| im3     | im2     | 630   | 9.84       |
| im3     | im2     | 630   | 9.84       |
| im3     | im3     | 630   | 9.84       |
| im3     | im3     | 630   | 9.84       |
| im3     | im4     | 630   | 9.84       |
| im3     | im4     | 630   | 9.84       |
| im3     | im5     | 630   | 9.84       |
| im3     | im5     | 630   | 9.84       |
| im4     | im1     | 601   | 9.39       |
| im4     | im1     | 601   | 9.39       |
| im4     | im2     | 601   | 9.39       |
| im4     | im2     | 601   | 9.39       |
| im4     | im3     | 601   | 9.39       |
| im4     | im3     | 601   | 9.39       |
| im4     | im4     | 601   | 9.39       |
| im4     | im4     | 601   | 9.39       |
| im4     | im5     | 601   | 9.39       |
| im4     | im5     | 601   | 9.39       |
| im5     | im1     | 520   | 8.12       |
| im5     | im1     | 520   | 8.12       |
| im5     | im2     | 520   | 8.12       |
| im5     | im2     | 520   | 8.12       |
| im5     | im3     | 520   | 8.12       |
| im5     | im3     | 520   | 8.12       |
| im5     | im4     | 520   | 8.12       |
| im5     | im4     | 520   | 8.12       |
| im5     | im5     | 520   | 8.12       |
| im5     | im5     | 520   | 8.12       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 520   | 8.12       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im4     | 3105  | 48.51      |

## 6.17 IFFT-128

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

Функции

- void `nmppsFFT128Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.17.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

### 6.17.2 Функции

#### 6.17.2.1 nmppsFFT128Inv\_32fcr()

```
void nmppsFFT128Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 1091  | 8.52       |
| im1     | im1     | 1091  | 8.52       |
| im1     | im2     | 1091  | 8.52       |
| im1     | im2     | 1091  | 8.52       |
| im1     | im3     | 1091  | 8.52       |
| im1     | im3     | 1091  | 8.52       |
| im1     | im4     | 1091  | 8.52       |
| im1     | im4     | 1091  | 8.52       |
| im1     | im5     | 1091  | 8.52       |
| im1     | im5     | 1091  | 8.52       |
| im2     | im1     | 1091  | 8.52       |
| im2     | im1     | 1091  | 8.52       |
| im2     | im2     | 1091  | 8.52       |
| im2     | im2     | 1091  | 8.52       |
| im2     | im3     | 1091  | 8.52       |
| im2     | im3     | 1091  | 8.52       |
| im2     | im4     | 1091  | 8.52       |
| im2     | im4     | 1091  | 8.52       |
| im2     | im5     | 1091  | 8.52       |
| im2     | im5     | 1091  | 8.52       |
| im3     | im1     | 1309  | 10.22      |
| im3     | im1     | 1309  | 10.22      |
| im3     | im2     | 1309  | 10.22      |
| im3     | im2     | 1309  | 10.22      |
| im3     | im3     | 1309  | 10.22      |
| im3     | im3     | 1309  | 10.22      |
| im3     | im4     | 1309  | 10.22      |
| im3     | im4     | 1309  | 10.22      |
| im3     | im5     | 1309  | 10.22      |
| im3     | im5     | 1309  | 10.22      |
| im4     | im1     | 1280  | 10.00      |
| im4     | im1     | 1280  | 10.00      |
| im4     | im2     | 1280  | 10.00      |
| im4     | im2     | 1280  | 10.00      |
| im4     | im3     | 1280  | 10.00      |
| im4     | im3     | 1280  | 10.00      |
| im4     | im4     | 1280  | 10.00      |
| im4     | im4     | 1280  | 10.00      |
| im4     | im5     | 1280  | 10.00      |
| im4     | im5     | 1280  | 10.00      |
| im5     | im1     | 1091  | 8.52       |
| im5     | im1     | 1091  | 8.52       |
| im5     | im2     | 1091  | 8.52       |
| im5     | im2     | 1091  | 8.52       |
| im5     | im3     | 1091  | 8.52       |
| im5     | im3     | 1091  | 8.52       |
| im5     | im4     | 1091  | 8.52       |
| im5     | im4     | 1091  | 8.52       |
| im5     | im5     | 1091  | 8.52       |
| im5     | im5     | 1091  | 8.52       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 1091  | 8.52       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im1     | 1309  | 10.22      |



## 6.18 IFFT-256

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

Функции

- void `nmppsFFT256Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.18.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (1763 такта) необходимо положить входной вектор в 1-ый банк, выходной вектор в 5-ый банк

### 6.18.2 Функции

#### 6.18.2.1 nmppsFFT256Inv\_32fcr()

```
void nmppsFFT256Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im1      | 1819  | 7.10       |
| im1      | im1      | 1819  | 7.10       |
| im1      | im2      | 1763  | 6.88       |
| im1      | im2      | 1763  | 6.88       |
| im1      | im3      | 1763  | 6.88       |
| im1      | im3      | 1763  | 6.88       |
| im1      | im4      | 1763  | 6.88       |
| im1      | im4      | 1763  | 6.88       |
| im1      | im5      | 1763  | 6.88       |
| im1      | im5      | 1763  | 6.88       |
| im2      | im1      | 1910  | 7.46       |
| im2      | im1      | 1910  | 7.46       |
| im2      | im2      | 1854  | 7.24       |
| im2      | im2      | 1854  | 7.24       |
| im2      | im3      | 1854  | 7.24       |
| im2      | im3      | 1854  | 7.24       |
| im2      | im4      | 1854  | 7.24       |
| im2      | im4      | 1854  | 7.24       |
| im2      | im5      | 10561 | 41.25      |
| im2      | im5      | 10561 | 41.25      |
| im3      | im1      | 1906  | 7.44       |
| im3      | im1      | 1906  | 7.44       |
| im3      | im2      | 1850  | 7.22       |
| im3      | im2      | 1850  | 7.22       |
| im3      | im3      | 1850  | 7.22       |
| im3      | im3      | 1850  | 7.22       |
| im3      | im4      | 1850  | 7.22       |
| im3      | im4      | 1850  | 7.22       |
| im3      | im5      | 1850  | 7.22       |
| im3      | im5      | 1850  | 7.22       |
| im4      | im1      | 1896  | 7.40       |
| im4      | im1      | 1896  | 7.40       |
| im4      | im2      | 1840  | 7.18       |
| im4      | im2      | 1840  | 7.18       |
| im4      | im3      | 1840  | 7.18       |
| im4      | im3      | 1840  | 7.18       |
| im4      | im4      | 1840  | 7.18       |
| im4      | im4      | 1840  | 7.18       |
| im4      | im5      | 1840  | 7.18       |
| im4      | im5      | 1840  | 7.18       |
| im5      | im1      | 1819  | 7.10       |
| im5      | im1      | 1819  | 7.10       |
| im5      | im2      | 1763  | 6.88       |
| im5      | im2      | 1763  | 6.88       |
| im5      | im3      | 1763  | 6.88       |
| im5      | im3      | 1763  | 6.88       |
| im5      | im4      | 1763  | 6.88       |
| im5      | im4      | 1763  | 6.88       |
| im5      | im5      | 1763  | 6.88       |
| im5      | im5      | 1763  | 6.88       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im2     | 1763  | 6.88       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im5     | 10561 | 41.25      |

## 6.19 IFFT-512

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

Функции

- void `nmppsFFT512Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.19.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (3675 такта) необходимо положить входной вектор в 2-ый банк, выходной вектор в 5-ый банк

### 6.19.2 Функции

#### 6.19.2.1 nmppsFFT512Inv\_32fcr()

```
void nmppsFFT512Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im1     | 3858  | 7.53       |
| im1     | im1     | 3858  | 7.53       |
| im1     | im2     | 3976  | 7.76       |
| im1     | im2     | 3976  | 7.76       |
| im1     | im3     | 3858  | 7.53       |
| im1     | im3     | 3858  | 7.53       |
| im1     | im4     | 3858  | 7.53       |
| im1     | im4     | 3858  | 7.53       |
| im1     | im5     | 3858  | 7.53       |
| im1     | im5     | 3858  | 7.53       |
| im2     | im1     | 3675  | 7.17       |
| im2     | im1     | 3675  | 7.17       |
| im2     | im2     | 3793  | 7.40       |
| im2     | im2     | 3793  | 7.40       |
| im2     | im3     | 3675  | 7.17       |
| im2     | im3     | 3675  | 7.17       |
| im2     | im4     | 3675  | 7.17       |
| im2     | im4     | 3675  | 7.17       |
| im2     | im5     | 3675  | 7.17       |
| im2     | im5     | 3675  | 7.17       |
| im3     | im1     | 3858  | 7.53       |
| im3     | im1     | 3858  | 7.53       |
| im3     | im2     | 3976  | 7.76       |
| im3     | im2     | 3976  | 7.76       |
| im3     | im3     | 3858  | 7.53       |
| im3     | im3     | 3858  | 7.53       |
| im3     | im4     | 3858  | 7.53       |
| im3     | im4     | 3858  | 7.53       |
| im3     | im5     | 3858  | 7.53       |
| im3     | im5     | 3858  | 7.53       |
| im4     | im1     | 3848  | 7.51       |
| im4     | im1     | 3848  | 7.51       |
| im4     | im2     | 3966  | 7.74       |
| im4     | im2     | 3966  | 7.74       |
| im4     | im3     | 3848  | 7.51       |
| im4     | im3     | 3848  | 7.51       |
| im4     | im4     | 3848  | 7.51       |
| im4     | im4     | 3848  | 7.51       |
| im4     | im5     | 3848  | 7.51       |
| im4     | im5     | 3848  | 7.51       |
| im5     | im1     | 3675  | 7.17       |
| im5     | im1     | 3675  | 7.17       |
| im5     | im2     | 3793  | 7.40       |
| im5     | im2     | 3793  | 7.40       |
| im5     | im3     | 3675  | 7.17       |
| im5     | im3     | 3675  | 7.17       |
| im5     | im4     | 3675  | 7.17       |
| im5     | im4     | 3675  | 7.17       |
| im5     | im5     | 3675  | 7.17       |
| im5     | im5     | 3675  | 7.17       |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im1     | 3675  | 7.17       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im1     | im2     | 3976  | 7.76       |

## 6.20 IFFT-1024

### Функции

- void `nmppsFFT1024Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)  
Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 1024 комплексных чисел

#### 6.20.1 Подробное описание

#### 6.20.2 Функции

##### 6.20.2.1 nmppsFFT1024Inv\_32fcr()

```
void nmppsFFT1024Inv_32fcr (  
    const nm32fcr * pSrcVec,  
    nm32fcr * pDstVec,  
    NmppsFFTSpec_32fcr * spec )
```

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 1024 комплексных чисел

#### Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

#### Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

#### Аргументы

|    |      |                                                                                                   |
|----|------|---------------------------------------------------------------------------------------------------|
| in | spec | структра, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|---------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (8655 такта) необходимо положить входной вектор в 6-ый банк, выходной вектор в 5-ый банк

#### Perfomance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im1      | 10749 | 10.49      |
| im1      | im1      | 10749 | 10.49      |
| im1      | im2      | 9687  | 9.45       |
| im1      | im2      | 9687  | 9.45       |
| im1      | im3      | 10800 | 10.54      |
| im1      | im3      | 10800 | 10.54      |
| im1      | im4      | 11110 | 10.84      |
| im1      | im4      | 11110 | 10.84      |
| im1      | im5      | 8988  | 8.77       |
| im1      | im5      | 8988  | 8.77       |
| im2      | im1      | 10687 | 10.43      |
| im2      | im1      | 10687 | 10.43      |
| im2      | im2      | 9625  | 9.39       |
| im2      | im2      | 9625  | 9.39       |
| im2      | im3      | 10738 | 10.48      |
| im2      | im3      | 10738 | 10.48      |
| im2      | im4      | 11048 | 10.78      |
| im2      | im4      | 11048 | 10.78      |
| im2      | im5      | 8926  | 8.71       |
| im2      | im5      | 8926  | 8.71       |
| im3      | im1      | 11214 | 10.95      |
| im3      | im1      | 11214 | 10.95      |
| im3      | im2      | 10152 | 9.91       |
| im3      | im2      | 10152 | 9.91       |
| im3      | im3      | 11265 | 11.00      |
| im3      | im3      | 11265 | 11.00      |
| im3      | im4      | 11575 | 11.30      |
| im3      | im4      | 11575 | 11.30      |
| im3      | im5      | 9453  | 9.23       |
| im3      | im5      | 9453  | 9.23       |
| im4      | im1      | 10408 | 10.16      |
| im4      | im1      | 10408 | 10.16      |
| im4      | im2      | 9346  | 9.12       |
| im4      | im2      | 9346  | 9.12       |
| im4      | im3      | 10459 | 10.21      |
| im4      | im3      | 10459 | 10.21      |
| im4      | im4      | 10769 | 10.51      |
| im4      | im4      | 10769 | 10.51      |
| im4      | im5      | 8647  | 8.44       |
| im4      | im5      | 8647  | 8.44       |
| im5      | im1      | 10408 | 10.16      |
| im5      | im1      | 10408 | 10.16      |
| im5      | im2      | 9346  | 9.12       |
| im5      | im2      | 9346  | 9.12       |
| im5      | im3      | 10459 | 10.21      |
| im5      | im3      | 10459 | 10.21      |
| im5      | im4      | 10769 | 10.51      |
| im5      | im4      | 10769 | 10.51      |
| im5      | im5      | 8647  | 8.44       |
| im5      | im5      | 8647  | 8.44       |

The best configuration:



| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im4     | im5     | 8647  | 8.44       |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im4     | 11575 | 11.30      |

## 6.21 IFFT-2048

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

Функции

- void `nmppsFFT2048Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.21.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (19504 такта) необходимо положить входной вектор в 5-ый банк, выходной вектор в 5-ый банк

### 6.21.2 Функции

#### 6.21.2.1 nmppsFFT2048Inv\_32fcr()

```
void nmppsFFT2048Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im1      | 25301 | 12.35      |
| im1      | im1      | 25301 | 12.35      |
| im1      | im2      | 25369 | 12.38      |
| im1      | im2      | 25369 | 12.38      |
| im1      | im3      | 25723 | 12.56      |
| im1      | im3      | 25723 | 12.56      |
| im1      | im4      | 22349 | 10.91      |
| im1      | im4      | 22349 | 10.91      |
| im1      | im5      | 21203 | 10.35      |
| im1      | im5      | 21203 | 10.35      |
| im2      | im1      | 25356 | 12.38      |
| im2      | im1      | 25356 | 12.38      |
| im2      | im2      | 25424 | 12.41      |
| im2      | im2      | 25424 | 12.41      |
| im2      | im3      | 25778 | 12.58      |
| im2      | im3      | 25778 | 12.58      |
| im2      | im4      | 22404 | 10.93      |
| im2      | im4      | 22404 | 10.93      |
| im2      | im5      | 21258 | 10.37      |
| im2      | im5      | 21258 | 10.37      |
| im3      | im1      | 24789 | 12.10      |
| im3      | im1      | 24789 | 12.10      |
| im3      | im2      | 24857 | 12.13      |
| im3      | im2      | 24857 | 12.13      |
| im3      | im3      | 25211 | 12.31      |
| im3      | im3      | 25211 | 12.31      |
| im3      | im4      | 21837 | 10.66      |
| im3      | im4      | 21837 | 10.66      |
| im3      | im5      | 20691 | 10.10      |
| im3      | im5      | 20691 | 10.10      |
| im4      | im1      | 24789 | 12.10      |
| im4      | im1      | 24789 | 12.10      |
| im4      | im2      | 24857 | 12.13      |
| im4      | im2      | 24857 | 12.13      |
| im4      | im3      | 25211 | 12.31      |
| im4      | im3      | 25211 | 12.31      |
| im4      | im4      | 21837 | 10.66      |
| im4      | im4      | 21837 | 10.66      |
| im4      | im5      | 20691 | 10.10      |
| im4      | im5      | 20691 | 10.10      |
| im5      | im1      | 24789 | 12.10      |
| im5      | im1      | 24789 | 12.10      |
| im5      | im2      | 24857 | 12.13      |
| im5      | im2      | 24857 | 12.13      |
| im5      | im3      | 25211 | 12.31      |
| im5      | im3      | 25211 | 12.31      |
| im5      | im4      | 21837 | 10.66      |
| im5      | im4      | 21837 | 10.66      |
| im5      | im5      | 20691 | 10.10      |
| im5      | im5      | 20691 | 10.10      |

The best configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im3     | im5     | 20691 | 10.10      |

The worst configuration:

| pSrcVec | pDstVec | ticks | ticks/elem |
|---------|---------|-------|------------|
| im2     | im3     | 25778 | 12.58      |

## 6.22 IFFT-4096

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

Функции

- void `nmppsFFT4096Inv_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, `NmppsFFTSpec_32fcr` \*spec)

### 6.22.1 Подробное описание

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                    |
|----|------|----------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера |
|----|------|----------------------------------------------------------------------------------------------------|

Для достижения максимальной производительности (54258 такта) необходимо положить входной вектор в 5-ый банк, выходной вектор в 5-ый банк

### 6.22.2 Функции

#### 6.22.2.1 nmppsFFT4096Inv\_32fcr()

```
void nmppsFFT4096Inv_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    NmppsFFTSpec_32fcr * spec )
```

Performance table 0

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im2      | 61836 | 15.09      |
| im1      | im2      | 61836 | 15.09      |
| im1      | im3      | 63446 | 15.48      |
| im1      | im3      | 63446 | 15.48      |
| im1      | im4      | 63758 | 15.56      |
| im1      | im4      | 63758 | 15.56      |
| im1      | im5      | 61836 | 15.09      |
| im1      | im5      | 61836 | 15.09      |
| im3      | im2      | 59758 | 14.58      |
| im3      | im2      | 59758 | 14.58      |
| im3      | im3      | 61368 | 14.98      |
| im3      | im3      | 61368 | 14.98      |
| im3      | im4      | 61680 | 15.05      |
| im3      | im4      | 61680 | 15.05      |
| im3      | im5      | 59758 | 14.58      |
| im3      | im5      | 59758 | 14.58      |
| im4      | im2      | 59758 | 14.58      |
| im4      | im2      | 59758 | 14.58      |
| im4      | im3      | 61368 | 14.98      |
| im4      | im3      | 61368 | 14.98      |
| im4      | im4      | 61680 | 15.05      |
| im4      | im4      | 61680 | 15.05      |
| im4      | im5      | 59758 | 14.58      |
| im4      | im5      | 59758 | 14.58      |
| im5      | im2      | 59758 | 14.58      |
| im5      | im2      | 59758 | 14.58      |
| im5      | im3      | 61368 | 14.98      |
| im5      | im3      | 61368 | 14.98      |
| im5      | im4      | 61680 | 15.05      |
| im5      | im4      | 61680 | 15.05      |
| im5      | im5      | 59758 | 14.58      |
| im5      | im5      | 59758 | 14.58      |

The best configuration:

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im3      | im2      | 59758 | 14.58      |

The worst configuration:

| pSrc Vec | pDst Vec | ticks | ticks/elem |
|----------|----------|-------|------------|
| im1      | im4      | 63758 | 15.56      |

## 6.23 FFT-Common

### Группы

- [IFFT-Common](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором длины от 8 до 2048.

#### 6.23.1 Подробное описание

## 6.24 IFFT-Common

Функция для вычисления прямого БПФ с плавающей точкой над вектором длины от 8 до 2048.

Функция для вычисления прямого БПФ с плавающей точкой над вектором длины от 8 до 2048.

Аргументы

|    |         |                                                                                     |
|----|---------|-------------------------------------------------------------------------------------|
| in | pSrcVec | входной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|----|---------|-------------------------------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                                                      |
|-------|---------|--------------------------------------------------------------------------------------|
| [out] | pDstVec | выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float) |
|-------|---------|--------------------------------------------------------------------------------------|

Аргументы

|    |      |                                                                                                  |
|----|------|--------------------------------------------------------------------------------------------------|
| in | spec | структура, содержащая необходимые коэффициенты, для вычисления прямого БПФ определенного размера |
|----|------|--------------------------------------------------------------------------------------------------|

Функция инициализации структуры коэффициентов, необходимых для вычисления прямого БПФ с плавающей точкой над вектором длины от 8 до 2048

\param [in] spec двойной указатель на структуру коэффициентов

Аргументы

|    |       |                                                                                                          |
|----|-------|----------------------------------------------------------------------------------------------------------|
| in | order | размерность БПФ, которое нужно вычислить, например, для БПФ256 этот параметр равен 8 (т.к. $2^8 = 256$ ) |
|----|-------|----------------------------------------------------------------------------------------------------------|

Возвращает

Функция возвращают 0 в случае успешной инициализации и отрицательное число (от -1 и меньше) в случае ошибок

Функция для вычисления обратного БПФ с плавающей точкой над вектором длины от 8 до 2048

\param [in] pSrcVec входной вектор комплексных чисел (на мнимая и действительная части имеют тип float)  
 \retval [out] pDstVec выходной вектор комплексных чисел (на мнимая и действительная части имеют тип float)  
 \param [in] spec структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера



## 6.25 nmppsMalloc

Распределение памяти для векторов библиотеки.

### Функции

- `void * nmppsMalloc32 (unsigned sizeInt32)`
- `nm64s * nmppsMalloc_64s (unsigned nSize)`
- `nm1 * nmppsMalloc_1 (unsigned nSize)`
- `nm2s * nmppsMalloc_2s (unsigned nSize)`
- `nm2u * nmppsMalloc_2u (unsigned nSize)`
- `nm4s * nmppsMalloc_4s (unsigned nSize)`
- `nm4u * nmppsMalloc_4u (unsigned nSize)`
- `nm8u * nmppsMalloc_8u (unsigned nSize)`
- `nm8s * nmppsMalloc_8s (unsigned nSize)`
- `nm16u * nmppsMalloc_16u (unsigned nSize)`
- `nm16s * nmppsMalloc_16s (unsigned nSize)`
- `nm32u * nmppsMalloc_32u (unsigned nSize)`
- `nm32s * nmppsMalloc_32s (unsigned nSize)`
- `nm64u * nmppsMalloc_64u (unsigned nSize)`
- `nm32sc * nmppsMalloc_32sc (unsigned sizeCmplxInt32)`
- `nm32fc * nmppsMalloc_32fc (unsigned sizeCmplxFLOAT)`
- `nm32fcr * nmppsMalloc_32fcr (unsigned sizeCmplxFLOAT)`
- `float * nmppsMalloc_32f (unsigned sizeFloat)`
- `double * nmppsMalloc_64f (unsigned sizeDouble)`

### 6.25.1 Подробное описание

Распределение памяти для векторов библиотеки.

#### Аргументы

|       |                                                                     |
|-------|---------------------------------------------------------------------|
| nSize | Число элементов в векторе.                                          |
| hint  | Номер банка памяти. Может принимать значения MEM_LOCAL, MEM_GLOBAL. |

#### Заметки

Память, распределенная с помощью функций nmppsMalloc\_ должна освобождаться с помощью функции nmppsFree().

## 6.26 nmppsFree

Освобождение памяти для векторов.

### Функции

- `void nmppsFree (void *ptr)`

### 6.26.1 Подробное описание

Освобождение памяти для векторов.

### Заметки

Данная функция должна вызываться только для векторов, распределенных с помощью функций `nmppsMalloc_`.

## 6.27 BLASS-LEVEL3

### Файлы

- файл [nmblas\\_sgemv.h](#)  
part of nmblas library, sgemm function implementation

### 6.27.1 Подробное описание

## 6.28 BLASS-LEVEL1

## Группы

- [nmblas\\_dasum](#)  
takes the sum of the absolute values
- [nmblas\\_daxpy](#)  
constant times a vector plus a vector
- [nmblas\\_dcopy](#)  
copies a vector, X, to a vector, Y
- [nmblas\\_ddot](#)  
forms the dot product of two vectors.
- [nmblas\\_dnrm2](#)  
returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^*H*x}$
- [nmblas\\_drot](#)  
applies a plane rotation
- [nmblas\\_drotg](#)  
construct gives plane rotation
- [nmblas\\_drotm](#)  
Apply a Given's rotation constructed by DROTMG.
- [nmblas\\_dscal](#)  
scales a vector by a constant
- [nmblas\\_dsdot](#)  
Compute the inner product of two vectors with double precision accumulation.
- [nmblas\\_dswap](#)  
interchanges two vectors
- [nmblas\\_dznrm2](#)  
returns the euclidean norm of a vector via the function name, so that  $DZNRM2 := \sqrt{x^*H*x}$
- [nmblas\\_idamax](#)  
finds the index of the first element having maximum absolute value.
- [nmblas\\_isamax](#)  
finds the index of the first element having maximum absolute value.
- [nmblas\\_sasum](#)  
takes the sum of the absolute values
- [nmblas\\_saxpy](#)  
constant times a vector plus a vector
- [nmblas\\_scnrm2](#)  
returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^*H*x}$
- [nmblas\\_scopy](#)  
copies a vector, X, to a vector, Y
- [nmblas\\_sdot](#)  
SDOT forms the dot product of two vectors.
- [nmblas\\_sdsdot](#)  
Compute the inner product of two vectors with double precision accumulation.
- [nmblas\\_snrm2](#)  
SNRM2 returns the euclidean norm of a vector via the function name, so that  $SNRM2 := \sqrt{x'^*x}$ .
- [nmblas\\_srot](#)  
applies a plane rotation
- [nmblas\\_srotm](#)  
Apply a Given's rotation constructed by SROTMG.
- [nmblas\\_sscal](#)  
scales a vector by a constant
- [nmblas\\_sswap](#)  
interchanges two vectors

### 6.28.1 Подробное описание

## 6.29 nmblas\_dasum

takes the sum of the absolute values

Функции

- double [nmblas\\_dasum](#) (const int N, const double \*X, const int INCX)

### 6.29.1 Подробное описание

takes the sum of the absolute values

Аргументы

|    |      |                                                |
|----|------|------------------------------------------------|
| in | N    | number of elements in input vector             |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) |
| in | INCX | storage spacing between elements of X          |

Возвращает

the sum of the absolute values.

### 6.29.2 Функции

#### 6.29.2.1 nmblas\_dasum()

```
double nmblas_dasum (
    const int N,
    const double * X,
    const int INCX )
```

Perfomance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 199   | 24.87      |
| 8    | im1 | 1    | 199   | 24.87      |
| 128  | im1 | 1    | 485   | 3.78       |
| 128  | im1 | 1    | 485   | 3.78       |
| 1024 | im1 | 1    | 2417  | 2.36       |
| 1024 | im1 | 1    | 2417  | 2.36       |
| 2048 | im1 | 1    | 4625  | 2.25       |
| 2048 | im1 | 1    | 4625  | 2.25       |

The best configuration:

| N | X   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 199   | 24.87      |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 4625  | 2.25       |

## 6.30 nmblas\_daxpy

constant times a vector plus a vector

### Функции

- void `nmblas_daxpy` (const int N, const double A, const double \*X, const int INCX, double \*Y, const int INCY)

### 6.30.1 Подробное описание

constant times a vector plus a vector

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in     | A    | specifies the scalar alpha                       |
| in     | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |

### 6.30.2 Функции

#### 6.30.2.1 nmblas\_daxpy()

```
void nmblas_daxpy (
    const int N,
    const double A,
    const double * X,
    const int INCX,
    double * Y,
    const int INCY )
```

#### Perfomance table 0

| N | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|---|-----|------|-----|------|-------|------------|
| 8 | 5 | im1 | 1    | im1 | 1    | 126   | 15.75      |
| 8 | 5 | im1 | 1    | im1 | 1    | 126   | 15.75      |
| 8 | 5 | im1 | 1    | im2 | 1    | 126   | 15.75      |
| 8 | 5 | im1 | 1    | im2 | 1    | 126   | 15.75      |
| 8 | 5 | im2 | 1    | im1 | 1    | 126   | 15.75      |



| N    | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|---|-----|------|-----|------|-------|------------|
| 8    | 5 | im2 | 1    | im1 | 1    | 126   | 15.75      |
| 8    | 5 | im2 | 1    | im2 | 1    | 126   | 15.75      |
| 8    | 5 | im2 | 1    | im2 | 1    | 126   | 15.75      |
| 128  | 5 | im1 | 1    | im1 | 1    | 310   | 2.42       |
| 128  | 5 | im1 | 1    | im1 | 1    | 310   | 2.42       |
| 128  | 5 | im1 | 1    | im2 | 1    | 278   | 2.17       |
| 128  | 5 | im1 | 1    | im2 | 1    | 278   | 2.17       |
| 128  | 5 | im2 | 1    | im1 | 1    | 278   | 2.17       |
| 128  | 5 | im2 | 1    | im1 | 1    | 278   | 2.17       |
| 128  | 5 | im2 | 1    | im2 | 1    | 310   | 2.42       |
| 128  | 5 | im2 | 1    | im2 | 1    | 310   | 2.42       |
| 1024 | 5 | im1 | 1    | im1 | 1    | 3054  | 2.98       |
| 1024 | 5 | im1 | 1    | im1 | 1    | 3054  | 2.98       |
| 1024 | 5 | im1 | 1    | im2 | 1    | 2210  | 2.15       |
| 1024 | 5 | im1 | 1    | im2 | 1    | 2210  | 2.15       |
| 1024 | 5 | im2 | 1    | im1 | 1    | 2210  | 2.15       |
| 1024 | 5 | im2 | 1    | im1 | 1    | 2210  | 2.15       |
| 1024 | 5 | im2 | 1    | im2 | 1    | 3054  | 2.98       |
| 1024 | 5 | im2 | 1    | im2 | 1    | 3054  | 2.98       |
| 2048 | 5 | im1 | 1    | im1 | 1    | 6190  | 3.02       |
| 2048 | 5 | im1 | 1    | im1 | 1    | 6190  | 3.02       |
| 2048 | 5 | im1 | 1    | im2 | 1    | 4418  | 2.15       |
| 2048 | 5 | im1 | 1    | im2 | 1    | 4418  | 2.15       |
| 2048 | 5 | im2 | 1    | im1 | 1    | 4418  | 2.15       |
| 2048 | 5 | im2 | 1    | im1 | 1    | 4418  | 2.15       |
| 2048 | 5 | im2 | 1    | im2 | 1    | 6190  | 3.02       |
| 2048 | 5 | im2 | 1    | im2 | 1    | 6190  | 3.02       |

The best configuration:

| N | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|---|-----|------|-----|------|-------|------------|
| 8 | 5 | im1 | 1    | im1 | 1    | 126   | 15.75      |

The worst configuration:

| N    | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|---|-----|------|-----|------|-------|------------|
| 2048 | 5 | im1 | 1    | im1 | 1    | 6190  | 3.02       |

## 6.31 nmblas\_dcopy

copies a vector, X, to a vector, Y

Функции

- void [nmblas\\_dcopy](#) (const int N, const double \*X, const int INCX, double \*Y, const int INCY)

### 6.31.1 Подробное описание

copies a vector, X, to a vector, Y

Аргументы

|     |      |                                                  |
|-----|------|--------------------------------------------------|
| in  | N    | number of elements in input vector               |
| in  | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in  | INCX | storage spacing between elements of X            |
| out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in  | INCY | storage spacing between elements of Y            |

### 6.31.2 Функции

#### 6.31.2.1 nmblas\_dcopy()

```
void nmblas_dcopy (
    const int N,
    const double * X,
    const int INCX,
    double * Y,
    const int INCY )
```

Perfomance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 115   | 14.37      |
| 8 | im1 | 1    | im1 | 1    | 115   | 14.37      |
| 8 | im1 | 1    | im2 | 1    | 115   | 14.37      |
| 8 | im1 | 1    | im2 | 1    | 115   | 14.37      |
| 8 | im2 | 1    | im1 | 1    | 115   | 14.37      |
| 8 | im2 | 1    | im1 | 1    | 115   | 14.37      |
| 8 | im2 | 1    | im2 | 1    | 115   | 14.37      |
| 8 | im2 | 1    | im2 | 1    | 115   | 14.37      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 128  | im1 | 1    | im1 | 1    | 236   | 1.84       |
| 128  | im1 | 1    | im1 | 1    | 236   | 1.84       |
| 128  | im1 | 1    | im2 | 1    | 176   | 1.37       |
| 128  | im1 | 1    | im2 | 1    | 176   | 1.37       |
| 128  | im2 | 1    | im1 | 1    | 176   | 1.37       |
| 128  | im2 | 1    | im1 | 1    | 176   | 1.37       |
| 128  | im2 | 1    | im2 | 1    | 236   | 1.84       |
| 128  | im2 | 1    | im2 | 1    | 236   | 1.84       |
| 1024 | im1 | 1    | im1 | 1    | 2028  | 1.98       |
| 1024 | im1 | 1    | im1 | 1    | 2028  | 1.98       |
| 1024 | im1 | 1    | im2 | 1    | 1072  | 1.04       |
| 1024 | im1 | 1    | im2 | 1    | 1072  | 1.04       |
| 1024 | im2 | 1    | im1 | 1    | 1072  | 1.04       |
| 1024 | im2 | 1    | im1 | 1    | 1072  | 1.04       |
| 1024 | im2 | 1    | im2 | 1    | 2028  | 1.98       |
| 1024 | im2 | 1    | im2 | 1    | 2028  | 1.98       |
| 2048 | im1 | 1    | im1 | 1    | 4076  | 1.99       |
| 2048 | im1 | 1    | im1 | 1    | 4076  | 1.99       |
| 2048 | im1 | 1    | im2 | 1    | 2096  | 1.02       |
| 2048 | im1 | 1    | im2 | 1    | 2096  | 1.02       |
| 2048 | im2 | 1    | im1 | 1    | 2096  | 1.02       |
| 2048 | im2 | 1    | im1 | 1    | 2096  | 1.02       |
| 2048 | im2 | 1    | im2 | 1    | 4076  | 1.99       |
| 2048 | im2 | 1    | im2 | 1    | 4076  | 1.99       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 115   | 14.37      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 4076  | 1.99       |

## 6.32 nmblas\_ddot

forms the dot product of two vectors.

### Функции

- double `nmblas_ddot` (const int N, const double \*X, const int INCX, const double \*Y, const int INCY)

### 6.32.1 Подробное описание

forms the dot product of two vectors.

#### Аргументы

|    |      |                                                  |
|----|------|--------------------------------------------------|
| in | N    | number of elements in input vector               |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in | INCX | storage spacing between elements of X            |
| in | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in | INCY | storage spacing between elements of Y            |

#### Возвращает

Return dot product of two vectors

### 6.32.2 Функции

#### 6.32.2.1 nmblas\_ddot()

```
double nmblas_ddot (
    const int N,
    const double * X,
    const int INCX,
    const double * Y,
    const int INCY )
```

#### Perfomance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 202   | 25.25      |
| 8 | im1 | 1    | im1 | 1    | 202   | 25.25      |
| 8 | im1 | 1    | im2 | 1    | 199   | 24.87      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 8    | im1 | 1    | im2 | 1    | 199   | 24.87      |
| 8    | im2 | 1    | im1 | 1    | 199   | 24.87      |
| 8    | im2 | 1    | im1 | 1    | 199   | 24.87      |
| 8    | im2 | 1    | im2 | 1    | 202   | 25.25      |
| 8    | im2 | 1    | im2 | 1    | 202   | 25.25      |
| 128  | im1 | 1    | im1 | 1    | 470   | 3.67       |
| 128  | im1 | 1    | im1 | 1    | 470   | 3.67       |
| 128  | im1 | 1    | im2 | 1    | 346   | 2.70       |
| 128  | im1 | 1    | im2 | 1    | 346   | 2.70       |
| 128  | im2 | 1    | im1 | 1    | 346   | 2.70       |
| 128  | im2 | 1    | im1 | 1    | 346   | 2.70       |
| 128  | im2 | 1    | im2 | 1    | 470   | 3.67       |
| 128  | im2 | 1    | im2 | 1    | 470   | 3.67       |
| 1024 | im1 | 1    | im1 | 1    | 2262  | 2.20       |
| 1024 | im1 | 1    | im1 | 1    | 2262  | 2.20       |
| 1024 | im1 | 1    | im2 | 1    | 1242  | 1.21       |
| 1024 | im1 | 1    | im2 | 1    | 1242  | 1.21       |
| 1024 | im2 | 1    | im1 | 1    | 1242  | 1.21       |
| 1024 | im2 | 1    | im1 | 1    | 1242  | 1.21       |
| 1024 | im2 | 1    | im2 | 1    | 2262  | 2.20       |
| 1024 | im2 | 1    | im2 | 1    | 2262  | 2.20       |
| 2048 | im1 | 1    | im1 | 1    | 4310  | 2.10       |
| 2048 | im1 | 1    | im1 | 1    | 4310  | 2.10       |
| 2048 | im1 | 1    | im2 | 1    | 2266  | 1.10       |
| 2048 | im1 | 1    | im2 | 1    | 2266  | 1.10       |
| 2048 | im2 | 1    | im1 | 1    | 2266  | 1.10       |
| 2048 | im2 | 1    | im1 | 1    | 2266  | 1.10       |
| 2048 | im2 | 1    | im2 | 1    | 4310  | 2.10       |
| 2048 | im2 | 1    | im2 | 1    | 4310  | 2.10       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im2 | 1    | 199   | 24.87      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 4310  | 2.10       |

## 6.33 nmblas\_dnrm2

returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^T H x}$

### Функции

- double `nmblas_dnrm2` (const int N, const double \*x, const int INCX)

#### 6.33.1 Подробное описание

returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^T H x}$

#### Аргументы

|    |      |                                                     |
|----|------|-----------------------------------------------------|
| in | N    | number of elements in input vector                  |
| in | X    | array, dimension $(1 + (N - 1) * \text{abs}(INCX))$ |
| in | INCX | storage spacing between elements of X               |

#### Возвращает

euclidean norm

#### 6.33.2 Функции

##### 6.33.2.1 nmblas\_dnrm2()

```
double nmblas_dnrm2 (
    const int N,
    const double * x,
    const int INCX )
```

#### Perfomance table 0

| N    | x   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 425   | 53.12      |
| 8    | im1 | 1    | 425   | 53.12      |
| 128  | im1 | 1    | 608   | 4.75       |
| 128  | im1 | 1    | 608   | 4.75       |
| 1024 | im1 | 1    | 1504  | 1.46       |
| 1024 | im1 | 1    | 1504  | 1.46       |
| 2048 | im1 | 1    | 2528  | 1.23       |
| 2048 | im1 | 1    | 2528  | 1.23       |

The best configuration:

| N | x   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 425   | 53.12      |

The worst configuration:

| N    | x   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 2528  | 1.23       |

## 6.34 nmblas\_drot

applies a plane rotation

### Функции

- void `nmblas_drot` (const int N, double \*X, const int INCX, double \*Y, const int INCY, const double C, const double S)

#### 6.34.1 Подробное описание

applies a plane rotation

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |
| in     | C    |                                                  |
| in     | S    |                                                  |

#### 6.34.2 Функции

##### 6.34.2.1 nmblas\_drot()

```
void nmblas_drot (
    const int N,
    double * X,
    const int INCX,
    double * Y,
    const int INCY,
    const double C,
    const double S )
```

Perfomance table 0

| N | X   | INCX | Y   | INCY | C | S | ticks | ticks/elem |
|---|-----|------|-----|------|---|---|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 1 | 1 | 143   | 17.87      |
| 8 | im1 | 1    | im1 | 1    | 1 | 1 | 143   | 17.87      |
| 8 | im1 | 1    | im2 | 1    | 1 | 1 | 143   | 17.87      |



| N    | X   | INCX | Y   | INCY | C | S | ticks | ticks/elem |
|------|-----|------|-----|------|---|---|-------|------------|
| 8    | im1 | 1    | im2 | 1    | 1 | 1 | 143   | 17.87      |
| 8    | im2 | 1    | im1 | 1    | 1 | 1 | 143   | 17.87      |
| 8    | im2 | 1    | im1 | 1    | 1 | 1 | 143   | 17.87      |
| 8    | im2 | 1    | im2 | 1    | 1 | 1 | 143   | 17.87      |
| 8    | im2 | 1    | im2 | 1    | 1 | 1 | 143   | 17.87      |
| 128  | im1 | 1    | im1 | 1    | 1 | 1 | 605   | 4.72       |
| 128  | im1 | 1    | im1 | 1    | 1 | 1 | 605   | 4.72       |
| 128  | im1 | 1    | im2 | 1    | 1 | 1 | 372   | 2.90       |
| 128  | im1 | 1    | im2 | 1    | 1 | 1 | 372   | 2.90       |
| 128  | im2 | 1    | im1 | 1    | 1 | 1 | 372   | 2.90       |
| 128  | im2 | 1    | im1 | 1    | 1 | 1 | 372   | 2.90       |
| 128  | im2 | 1    | im2 | 1    | 1 | 1 | 605   | 4.72       |
| 128  | im2 | 1    | im2 | 1    | 1 | 1 | 605   | 4.72       |
| 1024 | im1 | 1    | im1 | 1    | 1 | 1 | 5085  | 4.96       |
| 1024 | im1 | 1    | im1 | 1    | 1 | 1 | 5085  | 4.96       |
| 1024 | im1 | 1    | im2 | 1    | 1 | 1 | 2458  | 2.40       |
| 1024 | im1 | 1    | im2 | 1    | 1 | 1 | 2458  | 2.40       |
| 1024 | im2 | 1    | im1 | 1    | 1 | 1 | 2458  | 2.40       |
| 1024 | im2 | 1    | im1 | 1    | 1 | 1 | 2458  | 2.40       |
| 1024 | im2 | 1    | im2 | 1    | 1 | 1 | 5085  | 4.96       |
| 1024 | im2 | 1    | im2 | 1    | 1 | 1 | 5085  | 4.96       |
| 2048 | im1 | 1    | im1 | 1    | 1 | 1 | 10205 | 4.98       |
| 2048 | im1 | 1    | im1 | 1    | 1 | 1 | 10205 | 4.98       |
| 2048 | im1 | 1    | im2 | 1    | 1 | 1 | 4842  | 2.36       |
| 2048 | im1 | 1    | im2 | 1    | 1 | 1 | 4842  | 2.36       |
| 2048 | im2 | 1    | im1 | 1    | 1 | 1 | 4842  | 2.36       |
| 2048 | im2 | 1    | im1 | 1    | 1 | 1 | 4842  | 2.36       |
| 2048 | im2 | 1    | im2 | 1    | 1 | 1 | 10205 | 4.98       |
| 2048 | im2 | 1    | im2 | 1    | 1 | 1 | 10205 | 4.98       |

The best configuration:

| N | X   | INCX | Y   | INCY | C | S | ticks | ticks/elem |
|---|-----|------|-----|------|---|---|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 1 | 1 | 143   | 17.87      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | C | S | ticks | ticks/elem |
|------|-----|------|-----|------|---|---|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 1 | 1 | 10205 | 4.98       |

## 6.35 nmblas\_drotg

construct givens plane rotation

### Функции

- void nmblas\_drotg (double \*A, double \*B, double \*C, double \*S)

### 6.35.1 Подробное описание

construct givens plane rotation

#### Аргументы

|     |   |  |
|-----|---|--|
| in  | A |  |
| in  | B |  |
| in  | C |  |
| out | S |  |

## 6.36 nmblas\_drotm

Apply a Given's rotation constructed by DROTMG.

### Функции

- void nmblas\_drotm (const int N, double \*X, const int INCX, double \*Y, const int INCY, double \*param)

### 6.36.1 Подробное описание

Apply a Given's rotation constructed by DROTMG.

#### Аргументы

|        |       |                                                                 |
|--------|-------|-----------------------------------------------------------------|
| in     | N     | number of elements in input vector                              |
| in,out | X     | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) )                |
| in     | INCX  | storage spacing between elements of X                           |
| in,out | Y     | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) )                |
| in     | INCY  | storage spacing between elements of Y                           |
| in     | param | array, dimension (5). The rotation values constructed by SROTMG |

## 6.37 nmblas\_dscal

scales a vector by a constant

Функции

- void [nmblas\\_dscal](#) (const int N, const double ALPHA, double \*X, const int INCX)

### 6.37.1 Подробное описание

scales a vector by a constant

Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in     | A    | specifies the scalar alpha.                      |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |

### 6.37.2 Функции

#### 6.37.2.1 nmblas\_dscal()

```
void nmblas_dscal (
    const int N,
    const double ALPHA,
    double * X,
    const int INCX )
```

Perfomance table 0

| N    | ALPHA | X   | INCX | ticks | ticks/elem |
|------|-------|-----|------|-------|------------|
| 8    | 5     | im1 | 1    | 112   | 14.00      |
| 8    | 5     | im1 | 1    | 112   | 14.00      |
| 128  | 5     | im1 | 1    | 294   | 2.29       |
| 128  | 5     | im1 | 1    | 294   | 2.29       |
| 1024 | 5     | im1 | 1    | 2954  | 2.88       |
| 1024 | 5     | im1 | 1    | 2954  | 2.88       |
| 2048 | 5     | im1 | 1    | 5994  | 2.92       |
| 2048 | 5     | im1 | 1    | 5994  | 2.92       |

The best configuration:

| N | ALPHA | X   | INCX | ticks | ticks/elem |
|---|-------|-----|------|-------|------------|
| 8 | 5     | im1 | 1    | 112   | 14.00      |

The worst configuration:

| N    | ALPHA | X   | INCX | ticks | ticks/elem |
|------|-------|-----|------|-------|------------|
| 2048 | 5     | im1 | 1    | 5994  | 2.92       |

## 6.38 nmblas\_dsdot

Compute the inner product of two vectors with double precision accumulation.

### Функции

- double `nmblas_dsdot` (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)

### 6.38.1 Подробное описание

Compute the inner product of two vectors with double precision accumulation.

#### Аргументы

|    |      |                                                                                          |
|----|------|------------------------------------------------------------------------------------------|
| in | N    | number of elements in input vector                                                       |
| in | B    | single precision scalar to be added to inner product                                     |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) single precision vector with N elements |
| in | INCX | storage spacing between elements of X                                                    |
| in | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) )                                         |
| in | INCY | storage spacing between elements of Y                                                    |

#### Возвращает

Returns D.P. dot product accumulated in D.P., for S.P. SX and SY  $DSDOT = \sum \text{for } I = 0 \text{ to } N-1 \text{ of } SX(LX+I*INCX) * SY(LY+I*INCY)$ , where  $LX = 1$  if  $INCX \geq 0$ , else  $LX = 1+(1-N)*INCX$ , and  $LY$  is defined in a similar way using  $INCY$ .

### 6.38.2 Функции

#### 6.38.2.1 nmblas\_dsdot()

```
double nmblas_dsdot (
    const int N,
    const float * X,
    const int INCX,
    const float * Y,
    const int INCY )
```

#### Performance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 201   | 25.12      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 8    | im1 | 1    | im1 | 1    | 201   | 25.12      |
| 8    | im1 | 1    | im2 | 1    | 201   | 25.12      |
| 8    | im1 | 1    | im2 | 1    | 201   | 25.12      |
| 8    | im2 | 1    | im1 | 1    | 201   | 25.12      |
| 8    | im2 | 1    | im1 | 1    | 201   | 25.12      |
| 8    | im2 | 1    | im2 | 1    | 201   | 25.12      |
| 8    | im2 | 1    | im2 | 1    | 201   | 25.12      |
| 128  | im1 | 1    | im1 | 1    | 523   | 4.08       |
| 128  | im1 | 1    | im1 | 1    | 523   | 4.08       |
| 128  | im1 | 1    | im2 | 1    | 523   | 4.08       |
| 128  | im1 | 1    | im2 | 1    | 523   | 4.08       |
| 128  | im2 | 1    | im1 | 1    | 523   | 4.08       |
| 128  | im2 | 1    | im1 | 1    | 523   | 4.08       |
| 128  | im2 | 1    | im2 | 1    | 523   | 4.08       |
| 128  | im2 | 1    | im2 | 1    | 523   | 4.08       |
| 1024 | im1 | 1    | im1 | 1    | 2357  | 2.30       |
| 1024 | im1 | 1    | im1 | 1    | 2357  | 2.30       |
| 1024 | im1 | 1    | im2 | 1    | 2357  | 2.30       |
| 1024 | im1 | 1    | im2 | 1    | 2357  | 2.30       |
| 1024 | im2 | 1    | im1 | 1    | 2357  | 2.30       |
| 1024 | im2 | 1    | im1 | 1    | 2357  | 2.30       |
| 1024 | im2 | 1    | im2 | 1    | 2357  | 2.30       |
| 1024 | im2 | 1    | im2 | 1    | 2357  | 2.30       |
| 2048 | im1 | 1    | im1 | 1    | 4453  | 2.17       |
| 2048 | im1 | 1    | im1 | 1    | 4453  | 2.17       |
| 2048 | im1 | 1    | im2 | 1    | 4453  | 2.17       |
| 2048 | im1 | 1    | im2 | 1    | 4453  | 2.17       |
| 2048 | im2 | 1    | im1 | 1    | 4453  | 2.17       |
| 2048 | im2 | 1    | im1 | 1    | 4453  | 2.17       |
| 2048 | im2 | 1    | im2 | 1    | 4453  | 2.17       |
| 2048 | im2 | 1    | im2 | 1    | 4453  | 2.17       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 201   | 25.12      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 4453  | 2.17       |

## 6.39 nmblas\_dswap

interchanges two vectors

### Функции

- void nmblas\_dswap (const int N, double \*X, const int INCX, double \*Y, const int INCY)

### 6.39.1 Подробное описание

interchanges two vectors

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |

Возвращает



## 6.40 nmblas\_dznrm2

returns the euclidean norm of a vector via the function name, so that  $DZNRM2 := \sqrt{x^{*H}x}$

### Функции

- `double nmblas_dznrm2 (const int N, const double *X, const int INCX)`

#### 6.40.1 Подробное описание

returns the euclidean norm of a vector via the function name, so that  $DZNRM2 := \sqrt{x^{*H}x}$

#### Аргументы

|    |      |                                                                           |
|----|------|---------------------------------------------------------------------------|
| in | N    | number of elements in input vector                                        |
| in | X    | COMPLEX*16 array of DIMENSION at least $(1 + (m - 1) * \text{abs}(INCX))$ |
| in | INCX | storage spacing between elements of X                                     |

#### Возвращает

euclidean norm

## 6.41 nmblas\_idamax

finds the index of the first element having maximum absolute value.

### Функции

- int `nmblas_idamax` (const int N, const double \*X, const int INCX)

#### 6.41.1 Подробное описание

finds the index of the first element having maximum absolute value.

#### Аргументы

|    |      |                                                  |
|----|------|--------------------------------------------------|
| in | N    | number of elements in input vector               |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in | INCX | storage spacing between elements of X            |

#### Возвращает

index of the first element having maximum absolute value.

#### 6.41.2 Функции

##### 6.41.2.1 nmblas\_idamax()

```
int nmblas_idamax (
    const int N,
    const double * X,
    const int INCX )
```

#### Perfomance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 524   | 65.50      |
| 8    | im1 | 1    | 524   | 65.50      |
| 128  | im1 | 1    | 695   | 5.42       |
| 128  | im1 | 1    | 695   | 5.42       |
| 1024 | im1 | 1    | 3635  | 3.54       |
| 1024 | im1 | 1    | 3635  | 3.54       |
| 2048 | im1 | 1    | 6995  | 3.41       |
| 2048 | im1 | 1    | 6995  | 3.41       |

The best configuration:

| N | X   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 524   | 65.50      |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 6995  | 3.41       |

## 6.42 nmblas\_isamax

finds the index of the first element having maximum absolute value.

### Функции

- int `nmblas_isamax` (const int N, const float \*X, const int INCX)

### 6.42.1 Подробное описание

finds the index of the first element having maximum absolute value.

#### Аргументы

|    |      |                                                |
|----|------|------------------------------------------------|
| in | N    | number of elements in input vector             |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) |
| in | INCX | storage spacing between elements of X          |

#### Возвращает

index of the first element having maximum absolute value.

## 6.42.2 Функции

### 6.42.2.1 nmblas\_isamax()

```
int nmblas_isamax (
    const int N,
    const float * X,
    const int INCX )
```

#### Perfomance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 551   | 68.87      |
| 8    | im1 | 1    | 551   | 68.87      |
| 128  | im1 | 1    | 513   | 4.00       |
| 128  | im1 | 1    | 513   | 4.00       |
| 1024 | im1 | 1    | 1983  | 1.93       |
| 1024 | im1 | 1    | 1983  | 1.93       |
| 2048 | im1 | 1    | 3663  | 1.78       |
| 2048 | im1 | 1    | 3663  | 1.78       |
| 4096 | im1 | 1    | 7023  | 1.71       |
| 4096 | im1 | 1    | 7023  | 1.71       |

The best configuration:

| N   | X   | INCX | ticks | ticks/elem |
|-----|-----|------|-------|------------|
| 128 | im1 | 1    | 513   | 4.00       |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 4096 | im1 | 1    | 7023  | 1.71       |

## 6.43 nmblas\_sasum

takes the sum of the absolute values

Функции

- float `nmblas_sasum` (const int N, const float \*X, const int INCX)

### 6.43.1 Подробное описание

takes the sum of the absolute values

Аргументы

|    |   |                                                |
|----|---|------------------------------------------------|
| in | N | number of elements in input vector             |
| in | A | specifies the scalar alpha                     |
| in | X | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) |

Возвращает

sum of the absolute values

### 6.43.2 Функции

#### 6.43.2.1 nmblas\_sasum()

```
float nmblas_sasum (
    const int N,
    const float * X,
    const int INCX )
```

Perfomance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 192   | 24.00      |
| 8    | im1 | 1    | 192   | 24.00      |
| 128  | im1 | 1    | 363   | 2.83       |
| 128  | im1 | 1    | 363   | 2.83       |
| 1024 | im1 | 1    | 1329  | 1.29       |
| 1024 | im1 | 1    | 1329  | 1.29       |
| 2048 | im1 | 1    | 2433  | 1.18       |
| 2048 | im1 | 1    | 2433  | 1.18       |

The best configuration:

| N | X   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 192   | 24.00      |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 2433  | 1.18       |

## 6.44 nmblas\_saxpy

constant times a vector plus a vector

### Функции

- void `nmblas_saxpy` (int N, const float A, const float \*X, const int INCX, float \*Y, const int INCY)

### 6.44.1 Подробное описание

constant times a vector plus a vector

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in     | A    | specifies the scalar alpha                       |
| in     | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |

### 6.44.2 Функции

#### 6.44.2.1 nmblas\_saxpy()

```
void nmblas_saxpy (
    int N,
    const float A,
    const float * X,
    const int INCX,
    float * Y,
    const int INCY )
```

#### Perfomance table 0

| N | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|---|-----|------|-----|------|-------|------------|
| 8 | 5 | im1 | 1    | im1 | 1    | 152   | 19.00      |
| 8 | 5 | im1 | 1    | im1 | 1    | 152   | 19.00      |
| 8 | 5 | im1 | 1    | im2 | 1    | 152   | 19.00      |
| 8 | 5 | im1 | 1    | im2 | 1    | 152   | 19.00      |
| 8 | 5 | im2 | 1    | im1 | 1    | 152   | 19.00      |
| 8 | 5 | im2 | 1    | im1 | 1    | 152   | 19.00      |



| N    | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|---|-----|------|-----|------|-------|------------|
| 8    | 5 | im2 | 1    | im2 | 1    | 152   | 19.00      |
| 8    | 5 | im2 | 1    | im2 | 1    | 152   | 19.00      |
| 128  | 5 | im1 | 1    | im1 | 1    | 160   | 1.25       |
| 128  | 5 | im1 | 1    | im1 | 1    | 160   | 1.25       |
| 128  | 5 | im1 | 1    | im2 | 1    | 160   | 1.25       |
| 128  | 5 | im1 | 1    | im2 | 1    | 160   | 1.25       |
| 128  | 5 | im2 | 1    | im1 | 1    | 160   | 1.25       |
| 128  | 5 | im2 | 1    | im1 | 1    | 160   | 1.25       |
| 128  | 5 | im2 | 1    | im2 | 1    | 160   | 1.25       |
| 128  | 5 | im2 | 1    | im2 | 1    | 160   | 1.25       |
| 1024 | 5 | im1 | 1    | im1 | 1    | 1500  | 1.46       |
| 1024 | 5 | im1 | 1    | im1 | 1    | 1500  | 1.46       |
| 1024 | 5 | im1 | 1    | im2 | 1    | 1120  | 1.09       |
| 1024 | 5 | im1 | 1    | im2 | 1    | 1120  | 1.09       |
| 1024 | 5 | im2 | 1    | im1 | 1    | 1120  | 1.09       |
| 1024 | 5 | im2 | 1    | im1 | 1    | 1120  | 1.09       |
| 1024 | 5 | im2 | 1    | im2 | 1    | 1500  | 1.46       |
| 1024 | 5 | im2 | 1    | im2 | 1    | 1500  | 1.46       |
| 2048 | 5 | im1 | 1    | im1 | 1    | 3068  | 1.49       |
| 2048 | 5 | im1 | 1    | im1 | 1    | 3068  | 1.49       |
| 2048 | 5 | im1 | 1    | im2 | 1    | 2224  | 1.08       |
| 2048 | 5 | im1 | 1    | im2 | 1    | 2224  | 1.08       |
| 2048 | 5 | im2 | 1    | im1 | 1    | 2224  | 1.08       |
| 2048 | 5 | im2 | 1    | im1 | 1    | 2224  | 1.08       |
| 2048 | 5 | im2 | 1    | im2 | 1    | 3068  | 1.49       |
| 2048 | 5 | im2 | 1    | im2 | 1    | 3068  | 1.49       |

The best configuration:

| N | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|---|-----|------|-----|------|-------|------------|
| 8 | 5 | im1 | 1    | im1 | 1    | 152   | 19.00      |

The worst configuration:

| N    | A | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|---|-----|------|-----|------|-------|------------|
| 2048 | 5 | im1 | 1    | im1 | 1    | 3068  | 1.49       |

## 6.45 nmblas\_scnrm2

returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^*H*x}$

### Функции

- float [nmblas\\_scnrm2](#) (const int N, const float \*X, const int INCX)

#### 6.45.1 Подробное описание

returns the euclidean norm of a vector via the function name, so that  $SCNRM2 := \sqrt{x^*H*x}$

#### Аргументы

|    |      |                                                |
|----|------|------------------------------------------------|
| in | N    | number of elements in input vector             |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) |
| in | INCX | storage spacing between elements of X          |

#### Возвращает

euclidean norm

#### 6.45.2 Функции

##### 6.45.2.1 nmblas\_scnrm2()

```
float nmblas_scnrm2 (
    const int N,
    const float * X,
    const int INCX )
```

#### Perfomance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 456   | 57.00      |
| 8    | im1 | 1    | 456   | 57.00      |
| 128  | im1 | 1    | 610   | 4.76       |
| 128  | im1 | 1    | 610   | 4.76       |
| 1024 | im1 | 1    | 1506  | 1.47       |
| 1024 | im1 | 1    | 1506  | 1.47       |
| 2048 | im1 | 1    | 2530  | 1.23       |
| 2048 | im1 | 1    | 2530  | 1.23       |

The best configuration:

| N | X   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 456   | 57.00      |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 2530  | 1.23       |

## 6.46 nmblas\_scopy

copies a vector, X, to a vector, Y

### Функции

- void [nmblas\\_scopy](#) (const int N, const float \*X, const int INCX, float \*Y, const int INCY)

### 6.46.1 Подробное описание

copies a vector, X, to a vector, Y

#### Аргументы

|     |      |                                                  |
|-----|------|--------------------------------------------------|
| in  | N    | number of elements in input vector               |
| in  | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in  | INCX | storage spacing between elements of X            |
| out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in  | INCY | storage spacing between elements of Y            |

### 6.46.2 Функции

#### 6.46.2.1 nmblas\_scopy()

```
void nmblas_scopy (
    const int N,
    const float * X,
    const int INCX,
    float * Y,
    const int INCY )
```

Perfomance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 127   | 15.87      |
| 8 | im1 | 1    | im1 | 1    | 127   | 15.87      |
| 8 | im1 | 1    | im2 | 1    | 127   | 15.87      |
| 8 | im1 | 1    | im2 | 1    | 127   | 15.87      |
| 8 | im2 | 1    | im1 | 1    | 127   | 15.87      |
| 8 | im2 | 1    | im1 | 1    | 127   | 15.87      |
| 8 | im2 | 1    | im2 | 1    | 127   | 15.87      |
| 8 | im2 | 1    | im2 | 1    | 127   | 15.87      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 128  | im1 | 1    | im1 | 1    | 136   | 1.06       |
| 128  | im1 | 1    | im1 | 1    | 136   | 1.06       |
| 128  | im1 | 1    | im2 | 1    | 136   | 1.06       |
| 128  | im1 | 1    | im2 | 1    | 136   | 1.06       |
| 128  | im2 | 1    | im1 | 1    | 136   | 1.06       |
| 128  | im2 | 1    | im1 | 1    | 136   | 1.06       |
| 128  | im2 | 1    | im2 | 1    | 136   | 1.06       |
| 128  | im2 | 1    | im2 | 1    | 136   | 1.06       |
| 1024 | im1 | 1    | im1 | 1    | 1003  | 0.97       |
| 1024 | im1 | 1    | im1 | 1    | 1003  | 0.97       |
| 1024 | im1 | 1    | im2 | 1    | 559   | 0.54       |
| 1024 | im1 | 1    | im2 | 1    | 559   | 0.54       |
| 1024 | im2 | 1    | im1 | 1    | 559   | 0.54       |
| 1024 | im2 | 1    | im1 | 1    | 559   | 0.54       |
| 1024 | im2 | 1    | im2 | 1    | 1003  | 0.97       |
| 1024 | im2 | 1    | im2 | 1    | 1003  | 0.97       |
| 2048 | im1 | 1    | im1 | 1    | 2027  | 0.98       |
| 2048 | im1 | 1    | im1 | 1    | 2027  | 0.98       |
| 2048 | im1 | 1    | im2 | 1    | 1071  | 0.52       |
| 2048 | im1 | 1    | im2 | 1    | 1071  | 0.52       |
| 2048 | im2 | 1    | im1 | 1    | 1071  | 0.52       |
| 2048 | im2 | 1    | im1 | 1    | 1071  | 0.52       |
| 2048 | im2 | 1    | im2 | 1    | 2027  | 0.98       |
| 2048 | im2 | 1    | im2 | 1    | 2027  | 0.98       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 127   | 15.87      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 2027  | 0.98       |

## 6.47 nmblas\_sdot

SDOT forms the dot product of two vectors.

### Функции

- float `nmblas_sdot` (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)

#### 6.47.1 Подробное описание

SDOT forms the dot product of two vectors.

#### Аргументы

|    |      |                                                  |
|----|------|--------------------------------------------------|
| in | N    | number of elements in input vector               |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in | INCX | storage spacing between elements of X            |
| in | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in | INCY | storage spacing between elements of Y            |

#### Возвращает

Return dot product of two vectors

#### 6.47.2 Функции

##### 6.47.2.1 nmblas\_sdot()

```
float nmblas_sdot (
    const int N,
    const float * X,
    const int INCX,
    const float * Y,
    const int INCY )
```

#### Perfomance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 200   | 25.00      |
| 8 | im1 | 1    | im1 | 1    | 200   | 25.00      |
| 8 | im1 | 1    | im2 | 1    | 197   | 24.62      |
| 8 | im1 | 1    | im2 | 1    | 197   | 24.62      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 8    | im2 | 1    | im1 | 1    | 197   | 24.62      |
| 8    | im2 | 1    | im1 | 1    | 197   | 24.62      |
| 8    | im2 | 1    | im2 | 1    | 200   | 25.00      |
| 8    | im2 | 1    | im2 | 1    | 200   | 25.00      |
| 128  | im1 | 1    | im1 | 1    | 364   | 2.84       |
| 128  | im1 | 1    | im1 | 1    | 364   | 2.84       |
| 128  | im1 | 1    | im2 | 1    | 303   | 2.36       |
| 128  | im1 | 1    | im2 | 1    | 303   | 2.36       |
| 128  | im2 | 1    | im1 | 1    | 303   | 2.36       |
| 128  | im2 | 1    | im1 | 1    | 303   | 2.36       |
| 128  | im2 | 1    | im2 | 1    | 364   | 2.84       |
| 128  | im2 | 1    | im2 | 1    | 364   | 2.84       |
| 1024 | im1 | 1    | im1 | 1    | 1260  | 1.23       |
| 1024 | im1 | 1    | im1 | 1    | 1260  | 1.23       |
| 1024 | im1 | 1    | im2 | 1    | 751   | 0.73       |
| 1024 | im1 | 1    | im2 | 1    | 751   | 0.73       |
| 1024 | im2 | 1    | im1 | 1    | 751   | 0.73       |
| 1024 | im2 | 1    | im1 | 1    | 751   | 0.73       |
| 1024 | im2 | 1    | im2 | 1    | 1260  | 1.23       |
| 1024 | im2 | 1    | im2 | 1    | 1260  | 1.23       |
| 2048 | im1 | 1    | im1 | 1    | 2284  | 1.11       |
| 2048 | im1 | 1    | im1 | 1    | 2284  | 1.11       |
| 2048 | im1 | 1    | im2 | 1    | 1263  | 0.61       |
| 2048 | im1 | 1    | im2 | 1    | 1263  | 0.61       |
| 2048 | im2 | 1    | im1 | 1    | 1263  | 0.61       |
| 2048 | im2 | 1    | im1 | 1    | 1263  | 0.61       |
| 2048 | im2 | 1    | im2 | 1    | 2284  | 1.11       |
| 2048 | im2 | 1    | im2 | 1    | 2284  | 1.11       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im2 | 1    | 197   | 24.62      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 2284  | 1.11       |

## 6.48 nmblas\_sdsdot

Compute the inner product of two vectors with double precision accumulation.

### Функции

- float nmblas\_sdsdot (const int N, const float B, const float \*X, const int INCX, const float \*Y, const int INCY)

### 6.48.1 Подробное описание

Compute the inner product of two vectors with double precision accumulation.

#### Аргументы

|    |      |                                                                                          |
|----|------|------------------------------------------------------------------------------------------|
| in | N    | number of elements in input vector                                                       |
| in | B    | single precision scalar to be added to inner product                                     |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) single precision vector with N elements |
| in | INCX | storage spacing between elements of X                                                    |
| in | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) )                                         |
| in | INCY | storage spacing between elements of Y                                                    |

#### Возвращает

Returns S.P. result with dot product accumulated in D.P.  $SDSDOT = SB + \text{sum for } I = 0 \text{ to } N-1 \text{ of } SX(LX+I*INCX)*SY(LY+I*INCY)$ , where  $LX = 1$  if  $INCX \geq 0$ , else  $LX = 1+(1-N)*INCX$ , and  $LY$  is defined in a similar way using  $INCY$ .



## 6.49 nmblas\_snrm2

SNRM2 returns the euclidean norm of a vector via the function name, so that  $\text{SNRM2} := \sqrt{x'x}$ .

### Функции

- float **nmblas\_snrm2** (const int N, const float \*X, const int INCX)

#### 6.49.1 Подробное описание

SNRM2 returns the euclidean norm of a vector via the function name, so that  $\text{SNRM2} := \sqrt{x'x}$ .

#### Аргументы

|    |      |                                                  |
|----|------|--------------------------------------------------|
| in | N    | number of elements in input vector               |
| in | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in | INCX | storage spacing between elements of X            |

#### Возвращает

euclidean norm

#### 6.49.2 Функции

##### 6.49.2.1 nmblas\_snrm2()

```
float nmblas_snrm2 (
    const int N,
    const float * X,
    const int INCX )
```

#### Performance table 0

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 8    | im1 | 1    | 439   | 54.87      |
| 8    | im1 | 1    | 439   | 54.87      |
| 128  | im1 | 1    | 546   | 4.26       |
| 128  | im1 | 1    | 546   | 4.26       |
| 1024 | im1 | 1    | 994   | 0.97       |
| 1024 | im1 | 1    | 994   | 0.97       |
| 2048 | im1 | 1    | 1506  | 0.73       |
| 2048 | im1 | 1    | 1506  | 0.73       |

The best configuration:

| N | X   | INCX | ticks | ticks/elem |
|---|-----|------|-------|------------|
| 8 | im1 | 1    | 439   | 54.87      |

The worst configuration:

| N    | X   | INCX | ticks | ticks/elem |
|------|-----|------|-------|------------|
| 2048 | im1 | 1    | 1506  | 0.73       |

## 6.50 nmblas\_srot

applies a plane rotation

### Функции

- void nmblas\_srot (const int N, float \*X, const int INCX, float \*Y, const int INCY, const float C, const float S)

### 6.50.1 Подробное описание

applies a plane rotation

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |
| in     | C    |                                                  |
| in     | S    |                                                  |

## 6.51 nmblas\_srotm

Apply a Given's rotation constructed by SROTMG.

### Функции

- void nmblas\_srotm (const int N, float \*X, const int INCX, float \*Y, const int INCY, float \*param)

### 6.51.1 Подробное описание

Apply a Given's rotation constructed by SROTMG.

#### Аргументы

|        |       |                                                                 |
|--------|-------|-----------------------------------------------------------------|
| in     | N     | number of elements in input vector                              |
| in,out | X     | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) )                |
| in     | INCX  | storage spacing between elements of X                           |
| in,out | Y     | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) )                |
| in     | INCY  | storage spacing between elements of Y                           |
| in     | param | array, dimension (5). The rotation values constructed by SROTMG |

## 6.52 nmblas\_sscal

scales a vector by a constant

### Функции

- void [nmblas\\_sscal](#) (const int N, const float A, const float \*X, const int INCX)

#### 6.52.1 Подробное описание

scales a vector by a constant

#### Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in     | A    | specifies the scalar alpha.                      |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |

#### 6.52.2 Функции

##### 6.52.2.1 nmblas\_sscal()

```
void nmblas_sscal (
    const int N,
    const float A,
    const float * X,
    const int INCX )
```

Perfomance table 0

| N    | A | X   | INCX | ticks | ticks/elem |
|------|---|-----|------|-------|------------|
| 8    | 5 | im1 | 1    | 128   | 16.00      |
| 8    | 5 | im1 | 1    | 128   | 16.00      |
| 128  | 5 | im1 | 1    | 143   | 1.11       |
| 128  | 5 | im1 | 1    | 143   | 1.11       |
| 1024 | 5 | im1 | 1    | 1437  | 1.40       |
| 1024 | 5 | im1 | 1    | 1437  | 1.40       |
| 2048 | 5 | im1 | 1    | 2957  | 1.44       |
| 2048 | 5 | im1 | 1    | 2957  | 1.44       |

The best configuration:

| N | A | X   | INCX | ticks | ticks/elem |
|---|---|-----|------|-------|------------|
| 8 | 5 | im1 | 1    | 128   | 16.00      |

The worst configuration:

| N    | A | X   | INCX | ticks | ticks/elem |
|------|---|-----|------|-------|------------|
| 2048 | 5 | im1 | 1    | 2957  | 1.44       |

## 6.53 nmblas\_sswap

interchanges two vectors

Функции

- void [nmblas\\_sswap](#) (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)

### 6.53.1 Подробное описание

interchanges two vectors

Аргументы

|        |      |                                                  |
|--------|------|--------------------------------------------------|
| in     | N    | number of elements in input vector               |
| in,out | X    | array, dimension ( 1 + ( N - 1 ) * abs( INCX ) ) |
| in     | INCX | storage spacing between elements of X            |
| in,out | Y    | array, dimension ( 1 + ( N - 1 ) * abs( INCY ) ) |
| in     | INCY | storage spacing between elements of Y            |

### 6.53.2 Функции

#### 6.53.2.1 nmblas\_sswap()

```
void nmblas_sswap (
    const int N,
    const float * X,
    const int INCX,
    const float * Y,
    const int INCY )
```

Perfomance table 0

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im1 | 1    | 137   | 17.12      |
| 8 | im1 | 1    | im1 | 1    | 137   | 17.12      |
| 8 | im1 | 1    | im2 | 1    | 132   | 16.50      |
| 8 | im1 | 1    | im2 | 1    | 132   | 16.50      |
| 8 | im2 | 1    | im1 | 1    | 132   | 16.50      |
| 8 | im2 | 1    | im1 | 1    | 132   | 16.50      |
| 8 | im2 | 1    | im2 | 1    | 137   | 17.12      |
| 8 | im2 | 1    | im2 | 1    | 137   | 17.12      |

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 128  | im1 | 1    | im1 | 1    | 366   | 2.85       |
| 128  | im1 | 1    | im1 | 1    | 366   | 2.85       |
| 128  | im1 | 1    | im2 | 1    | 269   | 2.10       |
| 128  | im1 | 1    | im2 | 1    | 269   | 2.10       |
| 128  | im2 | 1    | im1 | 1    | 269   | 2.10       |
| 128  | im2 | 1    | im1 | 1    | 269   | 2.10       |
| 128  | im2 | 1    | im2 | 1    | 366   | 2.85       |
| 128  | im2 | 1    | im2 | 1    | 366   | 2.85       |
| 1024 | im1 | 1    | im1 | 1    | 2158  | 2.10       |
| 1024 | im1 | 1    | im1 | 1    | 2158  | 2.10       |
| 1024 | im1 | 1    | im2 | 1    | 1193  | 1.16       |
| 1024 | im1 | 1    | im2 | 1    | 1193  | 1.16       |
| 1024 | im2 | 1    | im1 | 1    | 1193  | 1.16       |
| 1024 | im2 | 1    | im1 | 1    | 1193  | 1.16       |
| 1024 | im2 | 1    | im2 | 1    | 2158  | 2.10       |
| 1024 | im2 | 1    | im2 | 1    | 2158  | 2.10       |
| 2048 | im1 | 1    | im1 | 1    | 4206  | 2.05       |
| 2048 | im1 | 1    | im1 | 1    | 4206  | 2.05       |
| 2048 | im1 | 1    | im2 | 1    | 2249  | 1.09       |
| 2048 | im1 | 1    | im2 | 1    | 2249  | 1.09       |
| 2048 | im2 | 1    | im1 | 1    | 2249  | 1.09       |
| 2048 | im2 | 1    | im1 | 1    | 2249  | 1.09       |
| 2048 | im2 | 1    | im2 | 1    | 4206  | 2.05       |
| 2048 | im2 | 1    | im2 | 1    | 4206  | 2.05       |

The best configuration:

| N | X   | INCX | Y   | INCY | ticks | ticks/elem |
|---|-----|------|-----|------|-------|------------|
| 8 | im1 | 1    | im2 | 1    | 132   | 16.50      |

The worst configuration:

| N    | X   | INCX | Y   | INCY | ticks | ticks/elem |
|------|-----|------|-----|------|-------|------------|
| 2048 | im1 | 1    | im1 | 1    | 4206  | 2.05       |



## 6.54 nmblas\_sgemv

performs one of the matrix-vector operations  $y := \alpha * A * x + \beta * y$ , or  $y := \alpha * A^T * x + \beta * y$ , where  $\alpha$  and  $\beta$  are scalars,  $x$  and  $y$  are vectors and  $A$  is an  $m$  by  $n$  matrix.

### Функции

- void nmblas\_sgemv (const enum nm\_trans TRANS, const int M, const int N, const float ALPHA, const float \*A, const int LDA, const float \*X, const int INCX, const float BETA, float \*Y, const int INCY)

#### 6.54.1 Подробное описание

performs one of the matrix-vector operations  $y := \alpha * A * x + \beta * y$ , or  $y := \alpha * A^T * x + \beta * y$ , where  $\alpha$  and  $\beta$  are scalars,  $x$  and  $y$  are vectors and  $A$  is an  $m$  by  $n$  matrix.

#### Аргументы

|    |       |                                                                                    |
|----|-------|------------------------------------------------------------------------------------|
| in | TRANS | is CHARACTER*1 On entry, TRANS specifies the operation to be performed as follows: |
|----|-------|------------------------------------------------------------------------------------|

TRANS = 'N' or 'n'  $y := \alpha * A * x + \beta * y$ .

TRANS = 'T' or 't'  $y := \alpha * A^T * x + \beta * y$ .

TRANS = 'C' or 'c'  $y := \alpha * A^H * x + \beta * y$ .

#### Аргументы

|        |       |                                                                                                                                                                                                                                                                                          |
|--------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in     | M     | specifies the number of rows of the matrix A. M must be at least zero.                                                                                                                                                                                                                   |
| in     | N     | specifies the number of columns of the matrix A. N must be at least zero.                                                                                                                                                                                                                |
| in     | ALPHA | ALPHA specifies the scalar alpha                                                                                                                                                                                                                                                         |
| in     | A     | array, dimension ( LDA, N ) Before entry, the leading m by n part of the array A must contain the matrix of coefficients                                                                                                                                                                 |
| in     | LDA   | specifies the first dimension of A as declared in the calling (sub) program. LDA must be at least max( 1, m ).                                                                                                                                                                           |
| in     | X     | array, dimension at least ( 1 + ( n - 1 ) * abs( INCX ) ) when TRANS = 'N' or 'n' and at least ( 1 + ( m - 1 ) * abs( INCX ) ) otherwise. Before entry, the incremented array X must contain the vector x.                                                                               |
| in     | INCX  | INCX specifies the increment for the elements of X. INCX must not be zero.                                                                                                                                                                                                               |
| in     | BETA  | specifies the scalar beta. When BETA is supplied as zero then Y need not be set on input.                                                                                                                                                                                                |
| in,out | Y     | is REAL array, dimension at least ( 1 + ( m - 1 ) * abs( INCY ) ) when TRANS = 'N' or 'n' and at least ( 1 + ( n - 1 ) * abs( INCY ) ) otherwise. Before entry with BETA non-zero, the incremented array Y must contain the vector y. On exit, Y is overwritten by the updated vector y. |
| in     | INCY  | specifies the increment for the elements of Y. INCY must not be zero.                                                                                                                                                                                                                    |

Возвращает

## 6.55 nmblas\_dgemv

performs one of the matrix-vector operations  $y := \alpha * A * x + \beta * y$ , or  $y := \alpha * A^T * x + \beta * y$ , where  $\alpha$  and  $\beta$  are scalars,  $x$  and  $y$  are vectors and  $A$  is an  $m$  by  $n$  matrix.

## Функции

- void nmblas\_dgemv (const enum nm\_trans TRANS, const int M, const int N, const double ALPHA, const double \*A, const int LDA, const double \*X, const int INCX, const double BETA, double \*Y, const int INCY)

## 6.55.1 Подробное описание

performs one of the matrix-vector operations  $y := \alpha * A * x + \beta * y$ , or  $y := \alpha * A^T * x + \beta * y$ , where  $\alpha$  and  $\beta$  are scalars,  $x$  and  $y$  are vectors and  $A$  is an  $m$  by  $n$  matrix.

## Аргументы

|    |       |                                                                                    |
|----|-------|------------------------------------------------------------------------------------|
| in | TRANS | is CHARACTER*1 On entry, TRANS specifies the operation to be performed as follows: |
|----|-------|------------------------------------------------------------------------------------|

TRANS = 'N' or 'n'  $y := \alpha * A * x + \beta * y$ .

TRANS = 'T' or 't'  $y := \alpha * A^T * x + \beta * y$ .

TRANS = 'C' or 'c'  $y := \alpha * A^T * x + \beta * y$ .

## Аргументы

|        |       |                                                                                                                                                                                                                                                                                          |
|--------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in     | M     | specifies the number of rows of the matrix A. M must be at least zero.                                                                                                                                                                                                                   |
| in     | N     | specifies the number of columns of the matrix A. N must be at least zero.                                                                                                                                                                                                                |
| in     | ALPHA | LPHA specifies the scalar alpha                                                                                                                                                                                                                                                          |
| in     | A     | array, dimension ( LDA, N ) Before entry, the leading m by n part of the array A must contain the matrix of coefficients                                                                                                                                                                 |
| in     | LDA   | specifies the first dimension of A as declared in the calling (sub) program. LDA must be at least max( 1, m ).                                                                                                                                                                           |
| in     | X     | array, dimension at least ( 1 + ( n - 1 ) * abs( INCX ) ) when TRANS = 'N' or 'n' and at least ( 1 + ( m - 1 ) * abs( INCX ) ) otherwise. Before entry, the incremented array X must contain the vector x.                                                                               |
| in     | INCX  | INCX specifies the increment for the elements of X. INCX must not be zero.                                                                                                                                                                                                               |
| in     | BETA  | specifies the scalar beta. When BETA is supplied as zero then Y need not be set on input.                                                                                                                                                                                                |
| in,out | Y     | is REAL array, dimension at least ( 1 + ( m - 1 ) * abs( INCY ) ) when TRANS = 'N' or 'n' and at least ( 1 + ( n - 1 ) * abs( INCY ) ) otherwise. Before entry with BETA non-zero, the incremented array Y must contain the vector y. On exit, Y is overwritten by the updated vector y. |
| in     | INCY  | specifies the increment for the elements of Y. INCY must not be zero.                                                                                                                                                                                                                    |

Возвращает

## 6.56 nmppcDivC

частное двух комплексных чисел

Функции

- void nmppcDivC (nm64sc \*pnSrcA, nm64s \*pnSrcB, nm64sc \*Dst)

### 6.56.1 Подробное описание

частное двух комплексных чисел

<>

Аргументы

|         |                                                                    |
|---------|--------------------------------------------------------------------|
| *pnSrcA | указатель на делимое. указатель на делитель. указатель на частное. |
|---------|--------------------------------------------------------------------|

## 6.57 nmppcProdC

произведение двух комплексных чисел.

### Функции

- void nmppcProdC (nm64sc \*pnSrcA, nm64sc \*pnSrcB, nm64sc \*Dst)

### 6.57.1 Подробное описание

произведение двух комплексных чисел.

#### Аргументы

|         |                                                                                          |
|---------|------------------------------------------------------------------------------------------|
| *pnSrcA | указатель на первый множитель. указатель на второй множитель. указатель на произведение. |
|---------|------------------------------------------------------------------------------------------|

## 6.58 nmppcFixExp32

Вычисление вычисления экспоненты числа в формате fixed-point (16.16)

Функции

- int [nmppcFixExp32](#) (int nVal)

### 6.58.1 Подробное описание

Вычисление вычисления экспоненты числа в формате fixed-point (16.16)

Аргументы

|      |                                                        |
|------|--------------------------------------------------------|
| nVal | Входное число с фиксированной точкой в формате (16.16) |
|------|--------------------------------------------------------|

Возвращает

Экспонента числа в формате с фиксированной точкой (16.16)

### 6.58.2 Функции

#### 6.58.2.1 nmppcFixExp32()

```
int nmppcFixExp32 (
    int nVal )
```

Perfomance table 0

| nVal    | ticks | ticks/elem |
|---------|-------|------------|
| 0x8000  | 388   | 388.00     |
| 0x8000  | 388   | 388.00     |
| 0x10000 | 420   | 420.00     |
| 0x10000 | 420   | 420.00     |
| 0x4000  | 404   | 404.00     |
| 0x4000  | 404   | 404.00     |

The best configuration:

| nVal   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 388   | 388.00     |

The worst configuration:

| nVal    | ticks | ticks/elem |
|---------|-------|------------|
| 0x10000 | 420   | 420.00     |



## 6.59 nmppcFixSinCos32

Вычисление синуса и косинуса от аргумента в формате fixed-point (16.16)

Функции

- void [nmppcFixSinCos32](#) (int nArg, int \*pnSin, int \*pnCos)

### 6.59.1 Подробное описание

Вычисление синуса и косинуса от аргумента в формате fixed-point (16.16)

Аргументы

|      |                                                                       |
|------|-----------------------------------------------------------------------|
| nArg | Угол в радианах. Угол должен быть в диапазоне от $-\pi/2$ до $+\pi/2$ |
|------|-----------------------------------------------------------------------|

Возвращаемые значения

|       |                      |
|-------|----------------------|
| pnSin | указатель на синус   |
| pnCos | указатель на косинус |

### 6.59.2 Функции

#### 6.59.2.1 nmppcFixSinCos32()

```
void nmppcFixSinCos32 (
    int nArg,
    int * pnSin,
    int * pnCos )
```

Perfomance table 0

| nArg   | pnSin | pnCos | ticks | ticks/elem |
|--------|-------|-------|-------|------------|
| 0x8000 | im0   | im1   | 286   | 286.00     |
| 0x8000 | im0   | im1   | 286   | 286.00     |

The best configuration:

| nArg   | pnSin | pnCos | ticks | ticks/elem |
|--------|-------|-------|-------|------------|
| 0x8000 | im0   | im1   | 286   | 286.00     |

The worst configuration:

| nArg   | pnSin | pnCos | ticks | ticks/elem |
|--------|-------|-------|-------|------------|
| 0x8000 | im0   | im1   | 286   | 286.00     |

## 6.60 nmppcFixArcTan32

Вычисление арктангенса от аргумента в формате fixed-point (16.16)

Функции

- int [nmppcFixArcTan32](#) (int nArg)

### 6.60.1 Подробное описание

Вычисление арктангенса от аргумента в формате fixed-point (16.16)

Аргументы

|      |                 |
|------|-----------------|
| nArg | Угол в радианах |
|------|-----------------|

Возвращает

Арктангенс

### 6.60.2 Функции

#### 6.60.2.1 nmppcFixArcTan32()

```
int nmppcFixArcTan32 (
    int nArg )
```

Perfomance table 0

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 262   | 262.00     |
| 0x8000 | 262   | 262.00     |

The best configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 262   | 262.00     |

The worst configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 262   | 262.00     |

## 6.61 nmppcDoubleToFix32

Функция перевода из Fixed-Point (16.16) в Double.

Функции

- `int nmppcDoubleToFix32 (double arg)`

### 6.61.1 Подробное описание

Функция перевода из Fixed-Point (16.16) в Double.

Аргументы

|                       |                                  |
|-----------------------|----------------------------------|
| <code>arg</code>      | Входное число с плавающей точкой |
| <code>fixpoint</code> | позиция двоичной точки           |

Возвращает

Число с фиксированной точкой

## 6.62 nmppcFix32ToDouble

Преобразование 32р. числа с фиксированной точкой (16.16) в число с плавающей точкой типа double.

### Функции

- double [nmppcFix32ToDouble](#) (int arg)

#### 6.62.1 Подробное описание

Преобразование 32р. числа с фиксированной точкой (16.16) в число с плавающей точкой типа double.

#### Аргументы

|     |                                                             |
|-----|-------------------------------------------------------------|
| arg | Входное 32р. число в формате с фиксированной точкой (16.16) |
|-----|-------------------------------------------------------------|

#### Возвращает

Число с плавающей точкой

#### 6.62.2 Функции

##### 6.62.2.1 nmppcFix32ToDouble()

```
double nmppcFix32ToDouble (
    int arg )
```

#### Perfomance table 0

| arg  | ticks | ticks/elem |
|------|-------|------------|
| 1    | 244   | 244.00     |
| 1    | 244   | 244.00     |
| 10   | 229   | 229.00     |
| 10   | 229   | 229.00     |
| 1000 | 199   | 199.00     |
| 1000 | 199   | 199.00     |

The best configuration:

| arg  | ticks | ticks/elem |
|------|-------|------------|
| 1000 | 199   | 199.00     |

The worst configuration:

| arg | ticks | ticks/elem |
|-----|-------|------------|
| 1   | 244   | 244.00     |

## 6.63 nmppcFixSqrt32

Вычисление квадратного корня числа в формате fixed-point (16.16)

Функции

- unsigned int nmppcFixSqrt32 (unsigned int nVal)

### 6.63.1 Подробное описание

Вычисление квадратного корня числа в формате fixed-point (16.16)

Аргументы

|      |                                                        |
|------|--------------------------------------------------------|
| nVal | Входное число с фиксированной точкой в формате (16.16) |
|------|--------------------------------------------------------|

Возвращает

Квадратный корень в формате с фиксированной точкой (16.16)



## 6.64 nmppcFixMul32

Вычисление произведения двух числе в формате fixed-point (16.16)

Функции

- int [nmppcFixMul32](#) (int nX, int nY)
- int [nmppcFixDiv32](#) (int nX, int nY)

### 6.64.1 Подробное описание

Вычисление произведения двух числе в формате fixed-point (16.16)

Деление двух целых чисел с записью результата в формате fixed-point (16.16)

Аргументы

|    |                                                               |
|----|---------------------------------------------------------------|
| nX | Первое входное число с фиксированной точкой в формате (16.16) |
| nY | Второе входное число с фиксированной точкой в формате (16.16) |

Возвращает

Произведение в формате с фиксированной точкой (16.16)

Аргументы

|    |          |
|----|----------|
| nX | Делимое  |
| nY | Делитель |

Возвращает

Частное от деления в формате с фиксированной точкой (16.16)

### 6.64.2 Функции

#### 6.64.2.1 nmppcFixDiv32()

```
int nmppcFixDiv32 (  
    int nX,  
    int nY )
```

Perfomance table 0

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 226   | 226.00     |
| 1  | 1  | 226   | 226.00     |

The best configuration:

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 226   | 226.00     |

The worst configuration:

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 226   | 226.00     |

Perfomance table 1

| nX       | nY       | ticks | ticks/elem |
|----------|----------|-------|------------|
| 0x102340 | 0x102340 | 226   | 226.00     |
| 0x102340 | 0x102340 | 226   | 226.00     |
| 0x102340 | 0x10234  | 304   | 304.00     |
| 0x102340 | 0x10234  | 304   | 304.00     |
| 0x102340 | 0x0080   | 413   | 413.00     |
| 0x102340 | 0x0080   | 413   | 413.00     |
| 0x10234  | 0x102340 | 226   | 226.00     |
| 0x10234  | 0x102340 | 226   | 226.00     |
| 0x10234  | 0x10234  | 226   | 226.00     |
| 0x10234  | 0x10234  | 226   | 226.00     |
| 0x10234  | 0x0080   | 361   | 361.00     |
| 0x10234  | 0x0080   | 361   | 361.00     |
| 0x0080   | 0x102340 | 226   | 226.00     |
| 0x0080   | 0x102340 | 226   | 226.00     |
| 0x0080   | 0x10234  | 218   | 218.00     |
| 0x0080   | 0x10234  | 218   | 218.00     |
| 0x0080   | 0x0080   | 226   | 226.00     |
| 0x0080   | 0x0080   | 226   | 226.00     |

The best configuration:

| nX     | nY      | ticks | ticks/elem |
|--------|---------|-------|------------|
| 0x0080 | 0x10234 | 218   | 218.00     |

The worst configuration:

| nX       | nY     | ticks | ticks/elem |
|----------|--------|-------|------------|
| 0x102340 | 0x0080 | 413   | 413.00     |

## 6.64.2.2 nmppcFixMul32()

```
int nmppcFixMul32 (
    int nX,
    int nY )
```

Perfomance table 0

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 84    | 84.00      |
| 1  | 1  | 84    | 84.00      |

The best configuration:

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 84    | 84.00      |

The worst configuration:

| nX | nY | ticks | ticks/elem |
|----|----|-------|------------|
| 1  | 1  | 84    | 84.00      |

Perfomance table 1

| nX       | nY       | ticks | ticks/elem |
|----------|----------|-------|------------|
| 0x102340 | 0x102340 | 84    | 84.00      |
| 0x102340 | 0x102340 | 84    | 84.00      |
| 0x102340 | 0x10234  | 84    | 84.00      |
| 0x102340 | 0x10234  | 84    | 84.00      |
| 0x102340 | 0x0080   | 84    | 84.00      |
| 0x102340 | 0x0080   | 84    | 84.00      |
| 0x10234  | 0x102340 | 84    | 84.00      |
| 0x10234  | 0x102340 | 84    | 84.00      |
| 0x10234  | 0x10234  | 84    | 84.00      |
| 0x10234  | 0x10234  | 84    | 84.00      |
| 0x10234  | 0x0080   | 84    | 84.00      |
| 0x10234  | 0x0080   | 84    | 84.00      |
| 0x0080   | 0x102340 | 84    | 84.00      |
| 0x0080   | 0x102340 | 84    | 84.00      |
| 0x0080   | 0x10234  | 84    | 84.00      |
| 0x0080   | 0x10234  | 84    | 84.00      |
| 0x0080   | 0x0080   | 84    | 84.00      |
| 0x0080   | 0x0080   | 84    | 84.00      |

The best configuration:

| nX       | nY       | ticks | ticks/elem |
|----------|----------|-------|------------|
| 0x102340 | 0x102340 | 84    | 84.00      |

The worst configuration:

| nX       | nY       | ticks | ticks/elem |
|----------|----------|-------|------------|
| 0x102340 | 0x102340 | 84    | 84.00      |

## 6.65 nmppcFixInv32

Вычисление обратного значения целого числа с записью результата в формате fixed-point.

Функции

- int [nmppcFixInv32](#) (int nVal, int nFixpoint)

### 6.65.1 Подробное описание

Вычисление обратного значения целого числа с записью результата в формате fixed-point.

$$Res = 2^n Fixpoint / nVal$$

\param nVal

Делитель

Аргументы

|           |                                               |
|-----------|-----------------------------------------------|
| nFixpoint | Позиция бинарной точки в результирующем слове |
|-----------|-----------------------------------------------|

Возвращает

Частное от деления нав формате с фиксированной точкой (16.16)

### 6.65.2 Функции

#### 6.65.2.1 nmppcFixInv32()

```
int nmppcFixInv32 (
    int nVal,
    int nFixpoint )
```

Perfomance table 0

| nVal  | nFixpoint | ticks | ticks/elem |
|-------|-----------|-------|------------|
| 0x102 | 8         | 157   | 157.00     |
| 0x102 | 8         | 157   | 157.00     |
| 0x102 | 16        | 209   | 209.00     |
| 0x102 | 16        | 209   | 209.00     |

| nVal     | nFixpoint | ticks | ticks/elem |
|----------|-----------|-------|------------|
| 0x102    | 24        | 271   | 271.00     |
| 0x102    | 24        | 271   | 271.00     |
| 0x1034   | 8         | 157   | 157.00     |
| 0x1034   | 8         | 157   | 157.00     |
| 0x1034   | 16        | 217   | 217.00     |
| 0x1034   | 16        | 217   | 217.00     |
| 0x1034   | 24        | 269   | 269.00     |
| 0x1034   | 24        | 269   | 269.00     |
| 0x490000 | 8         | 157   | 157.00     |
| 0x490000 | 8         | 157   | 157.00     |
| 0x490000 | 16        | 221   | 221.00     |
| 0x490000 | 16        | 221   | 221.00     |
| 0x490000 | 24        | 285   | 285.00     |
| 0x490000 | 24        | 285   | 285.00     |

The best configuration:

| nVal  | nFixpoint | ticks | ticks/elem |
|-------|-----------|-------|------------|
| 0x102 | 8         | 157   | 157.00     |

The worst configuration:

| nVal     | nFixpoint | ticks | ticks/elem |
|----------|-----------|-------|------------|
| 0x490000 | 24        | 285   | 285.00     |

## 6.66 nmppcTblFixArcSin32

Вычисление функции  $\arcsin$  по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

Функции

- int [nmppcTblFixArcSin32](#) (int nArg)

### 6.66.1 Подробное описание

Вычисление функции  $\arcsin$  по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

Аргументы

| nArg | Входное значение. |
|------|-------------------|
|------|-------------------|

Возвращает

Угол в диапазоне от  $-\pi/2$  до  $+\pi/2$  в формате fixed-point (16.16)

### 6.66.2 Функции

#### 6.66.2.1 nmppcTblFixArcSin32()

```
int nmppcTblFixArcSin32 (
    int nArg )
```

Perfomance table 0

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |
| 0x8000 | 86    | 86.00      |

The best configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |



The worst configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |

## 6.67 nmppcTblFixArcCos32

Вычисление функции `arccos` по таблице. Входные и выходные значения задаются в формате `fixed-point` (16.16)

Функции

- `int nmppcTblFixArcCos32 (int nArg)`

### 6.67.1 Подробное описание

Вычисление функции `arccos` по таблице. Входные и выходные значения задаются в формате `fixed-point` (16.16)

Аргументы

|      |                   |
|------|-------------------|
| nArg | Входное значение. |
|------|-------------------|

Возвращает

Угол в диапазоне от 0 до  $\pi$  в формате `fixed-point` (16.16)

### 6.67.2 Функции

#### 6.67.2.1 nmppcTblFixArcCos32()

```
int nmppcTblFixArcCos32 (
    int nArg )
```

Perfomance table 0

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 87    | 87.00      |
| 0x8000 | 87    | 87.00      |

The best configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 87    | 87.00      |

The worst configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 87    | 87.00      |

## 6.68 nmppcTblFixCos32

Вычисление функции cos по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

### Функции

- int `nmppcTblFixCos32` (int nArg)

#### 6.68.1 Подробное описание

Вычисление функции cos по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

#### Аргументы

|      |                                                           |
|------|-----------------------------------------------------------|
| nArg | Угол в диапазоне от 0 до PI в формате fixed-point (16.16) |
|------|-----------------------------------------------------------|

#### Возвращает

значение cos в формате fixed-point (16.16)

#### 6.68.2 Функции

##### 6.68.2.1 nmppcTblFixCos32()

```
int nmppcTblFixCos32 (
    int nArg )
```

#### Perfomance table 0

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 88    | 88.00      |
| 0x8000 | 88    | 88.00      |

The best configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 88    | 88.00      |

The worst configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 88    | 88.00      |

## 6.69 nmppcTblFixSin32

Вычисление функции  $\sin$  по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

### Функции

- int `nmppcTblFixSin32` (int nArg)

#### 6.69.1 Подробное описание

Вычисление функции  $\sin$  по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)

#### Аргументы

|      |                                                                        |
|------|------------------------------------------------------------------------|
| nArg | Угол в диапазоне от $-\pi/2$ до $+\pi/2$ в формате fixed-point (16.16) |
|------|------------------------------------------------------------------------|

#### Возвращает

значение  $\sin$  в формате fixed-point (16.16)

#### 6.69.2 Функции

##### 6.69.2.1 nmppcTblFixSin32()

```
int nmppcTblFixSin32 (
    int nArg )
```

#### Perfomance table 0

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |
| 0x8000 | 86    | 86.00      |

The best configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |

The worst configuration:

| nArg   | ticks | ticks/elem |
|--------|-------|------------|
| 0x8000 | 86    | 86.00      |

## 6.70 nmppcFixDivMod32

Вычисление частного и остатка при делении чисел с фиксированной запятой в формате fixed-point (16.16)

### Функции

- void [nmppcFixDivMod32](#) (int nDividend, int nDivisor, int \*pnQuotient, int \*pnReminder)
- void nmppcFixDivPosMod32 (unsigned int nDividend, unsigned int nDivisor, int \*pnQuotient, int \*pnReminder)

### 6.70.1 Подробное описание

Вычисление частного и остатка при делении чисел с фиксированной запятой в формате fixed-point (16.16)

#### Аргументы

|           |                                        |
|-----------|----------------------------------------|
| nDividend | Делимое в формате fixed-point (16.16)  |
| nDivisor  | Делитель в формате fixed-point (16.16) |

#### Возвращаемые значения

|            |                                                  |
|------------|--------------------------------------------------|
| pnQuotient | Частное от деления в формате fixed-point (16.16) |
| pnReminder | Остаток от деления в формате fixed-point (16.16) |

### 6.70.2 Функции

#### 6.70.2.1 nmppcFixDivMod32()

```
void nmppcFixDivMod32 (
    int nDividend,
    int nDivisor,
    int * pnQuotient,
    int * pnReminder )
```

#### Perfomance table 0

| nDividend | nDivisor | pnQuotient | pnReminder | ticks | ticks/elem |
|-----------|----------|------------|------------|-------|------------|
| 0x5000    | 0x6000   | im0        | im1        | 118   | 118.00     |
| 0x5000    | 0x6000   | im0        | im1        | 118   | 118.00     |



The best configuration:

| nDividend | nDivisor | pnQuotient | pnReminder | ticks | ticks/elem |
|-----------|----------|------------|------------|-------|------------|
| 0x5000    | 0x6000   | im0        | im1        | 118   | 118.00     |

The worst configuration:

| nDividend | nDivisor | pnQuotient | pnReminder | ticks | ticks/elem |
|-----------|----------|------------|------------|-------|------------|
| 0x5000    | 0x6000   | im0        | im1        | 118   | 118.00     |

## 6.71 nmppcFixSqrt64

Вычисление квадратного корня числа в формате fixed-point (32.32)

Функции

- unsigned long nmppcFixSqrt64 (unsigned long x)

### 6.71.1 Подробное описание

Вычисление квадратного корня числа в формате fixed-point (32.32)

Аргументы

|   |                                                        |
|---|--------------------------------------------------------|
| x | Входное число с фиксированной точкой в формате (32.32) |
|---|--------------------------------------------------------|

Возвращает

Квадратный корень в формате с фиксированной точкой (32.32)

## 6.72 nmppcDoubleToFix64

Функция перевода из Fixed-Point 64 в Double.

Функции

- long nmppcDoubleToFix64 (double arg, int fixpoint)

### 6.72.1 Подробное описание

Функция перевода из Fixed-Point 64 в Double.

Аргументы

|          |                                  |
|----------|----------------------------------|
| arg      | Входное число с плавающей точкой |
| fixpoint | позиция двоичной точки           |

Возвращает

Число с фиксированной точкой

## 6.73 nmppcFix64ToDouble

Преобразование 64р. числа с фиксированной точкой в число с плавающей точкой типа double.

Функции

- double [nmppcFix64ToDouble](#) (long arg, int fixpoint)

### 6.73.1 Подробное описание

Преобразование 64р. числа с фиксированной точкой в число с плавающей точкой типа double.

Аргументы

|          |                                                     |
|----------|-----------------------------------------------------|
| arg      | Входное 64р. число в формате с фиксированной точкой |
| fixpoint | Позиция двоичной точки                              |

Возвращает

Число с плавающей точкой

### 6.73.2 Функции

#### 6.73.2.1 nmppcFix64ToDouble()

```
double nmppcFix64ToDouble (
    long arg,
    int fixpoint )
```

Perfomance table 0

| arg | fixpoint | ticks | ticks/elem |
|-----|----------|-------|------------|
| 1   | 1        | 477   | 477.00     |
| 1   | 1        | 477   | 477.00     |
| 1   | 32       | 477   | 477.00     |
| 1   | 32       | 477   | 477.00     |
| 1   | 63       | 477   | 477.00     |
| 1   | 63       | 477   | 477.00     |
| 10  | 1        | 459   | 459.00     |
| 10  | 1        | 459   | 459.00     |
| 10  | 32       | 459   | 459.00     |
| 10  | 32       | 459   | 459.00     |
| 10  | 63       | 459   | 459.00     |

| arg  | fixpoint | ticks | ticks/elem |
|------|----------|-------|------------|
| 10   | 63       | 459   | 459.00     |
| 1000 | 1        | 423   | 423.00     |
| 1000 | 1        | 423   | 423.00     |
| 1000 | 32       | 423   | 423.00     |
| 1000 | 32       | 423   | 423.00     |
| 1000 | 63       | 423   | 423.00     |
| 1000 | 63       | 423   | 423.00     |

The best configuration:

| arg  | fixpoint | ticks | ticks/elem |
|------|----------|-------|------------|
| 1000 | 1        | 423   | 423.00     |

The worst configuration:

| arg | fixpoint | ticks | ticks/elem |
|-----|----------|-------|------------|
| 1   | 1        | 477   | 477.00     |

## 6.74 nmppcFixDiv64

Деление двух целых чисел с записью результата в формате fixed-point.

Функции

- void [nmppcFixDiv64](#) (long \*nDividend, long \*nDivisor, int nFixpoint, long \*nQuotient)

### 6.74.1 Подробное описание

Деление двух целых чисел с записью результата в формате fixed-point.

Аргументы

|           |                                                              |
|-----------|--------------------------------------------------------------|
| nDividend | Делимое                                                      |
| nDivisor  | Делитель. Делитель должен быть по модулю больше чем делимое. |
| nFixpoint | Позиция двоичной точки                                       |

Возвращаемые значения

|           |                                                |
|-----------|------------------------------------------------|
| nQuotient | Частное в формате числа с фиксированной точкой |
|-----------|------------------------------------------------|

### 6.74.2 Функции

#### 6.74.2.1 nmppcFixDiv64()

```
void nmppcFixDiv64 (
    long * nDividend,
    long * nDivisor,
    int nFixpoint,
    long * nQuotient )
```

Perfomance table 0

| nDividend | nDivisor | nFixpoint | nQuotient | ticks | ticks/elem |
|-----------|----------|-----------|-----------|-------|------------|
| im0       | im0      | 16        | im2       | 614   | 614.00     |
| im0       | im0      | 16        | im2       | 614   | 614.00     |
| im0       | im0      | 32        | im2       | 1062  | 1062.00    |
| im0       | im0      | 32        | im2       | 1062  | 1062.00    |
| im0       | im0      | 48        | im2       | 1510  | 1510.00    |
| im0       | im0      | 48        | im2       | 1510  | 1510.00    |

| nDividend | nDivisor | nFixpoint | nQuotient | ticks | ticks/elem |
|-----------|----------|-----------|-----------|-------|------------|
| im0       | im1      | 16        | im2       | 621   | 621.00     |
| im0       | im1      | 16        | im2       | 621   | 621.00     |
| im0       | im1      | 32        | im2       | 1069  | 1069.00    |
| im0       | im1      | 32        | im2       | 1069  | 1069.00    |
| im0       | im1      | 48        | im2       | 1517  | 1517.00    |
| im0       | im1      | 48        | im2       | 1517  | 1517.00    |
| im1       | im0      | 16        | im2       | 622   | 622.00     |
| im1       | im0      | 16        | im2       | 622   | 622.00     |
| im1       | im0      | 32        | im2       | 1070  | 1070.00    |
| im1       | im0      | 32        | im2       | 1070  | 1070.00    |
| im1       | im0      | 48        | im2       | 1518  | 1518.00    |
| im1       | im0      | 48        | im2       | 1518  | 1518.00    |
| im1       | im1      | 16        | im2       | 615   | 615.00     |
| im1       | im1      | 16        | im2       | 615   | 615.00     |
| im1       | im1      | 32        | im2       | 1063  | 1063.00    |
| im1       | im1      | 32        | im2       | 1063  | 1063.00    |
| im1       | im1      | 48        | im2       | 1511  | 1511.00    |
| im1       | im1      | 48        | im2       | 1511  | 1511.00    |

The best configuration:

| nDividend | nDivisor | nFixpoint | nQuotient | ticks | ticks/elem |
|-----------|----------|-----------|-----------|-------|------------|
| im0       | im0      | 16        | im2       | 614   | 614.00     |

The worst configuration:

| nDividend | nDivisor | nFixpoint | nQuotient | ticks | ticks/elem |
|-----------|----------|-----------|-----------|-------|------------|
| im1       | im0      | 48        | im2       | 1518  | 1518.00    |

## 6.75 nmppcFixSinCos64

Вычисление синуса и косинуса от аргумента в формате fixed-point (32.32)

Функции

- void [nmppcFixSinCos64](#) (long nArg, long \*pnSin, long \*pnCos)

### 6.75.1 Подробное описание

Вычисление синуса и косинуса от аргумента в формате fixed-point (32.32)

Аргументы

|      |                                                                       |
|------|-----------------------------------------------------------------------|
| nArg | Угол в радианах. Угол должен быть в диапазоне от $-\pi/2$ до $+\pi/2$ |
|------|-----------------------------------------------------------------------|

Возвращаемые значения

|       |                      |
|-------|----------------------|
| pnSin | указатель на синус   |
| pnCos | указатель на косинус |

### 6.75.2 Функции

#### 6.75.2.1 nmppcFixSinCos64()

```
void nmppcFixSinCos64 (
    long nArg,
    long * pnSin,
    long * pnCos )
```

Perfomance table 0

| nArg       | pnSin | pnCos | ticks | ticks/elem |
|------------|-------|-------|-------|------------|
| 0x80000000 | im0   | im1   | 1526  | 1526.00    |
| 0x80000000 | im0   | im1   | 1526  | 1526.00    |

The best configuration:

| nArg       | pnSin | pnCos | ticks | ticks/elem |
|------------|-------|-------|-------|------------|
| 0x80000000 | im0   | im1   | 1526  | 1526.00    |



The worst configuration:

| nArg       | pnSin | pnCos | ticks | ticks/elem |
|------------|-------|-------|-------|------------|
| 0x80000000 | im0   | im1   | 1526  | 1526.00    |

## 6.76 nmppcFixArcTan64

Вычисление арктангенса от аргумента в формате fixed-point (32.32)

Функции

- long [nmppcFixArcTan64](#) (long nArg)

### 6.76.1 Подробное описание

Вычисление арктангенса от аргумента в формате fixed-point (32.32)

Аргументы

|      |                 |
|------|-----------------|
| nArg | Угол в радианах |
|------|-----------------|

Арктангенс

### 6.76.2 Функции

#### 6.76.2.1 nmppcFixArcTan64()

```
long nmppcFixArcTan64 (
    long nArg )
```

Perfomance table 0

| nArg       | ticks | ticks/elem |
|------------|-------|------------|
| 0x80000000 | 1507  | 1507.00    |
| 0x80000000 | 1507  | 1507.00    |

The best configuration:

| nArg       | ticks | ticks/elem |
|------------|-------|------------|
| 0x80000000 | 1507  | 1507.00    |

The worst configuration:

| nArg       | ticks | ticks/elem |
|------------|-------|------------|
| 0x80000000 | 1507  | 1507.00    |

## 6.77 nmppcFix64Exp01

Вычисление вычисления экспоненты числа в формате fixed-point (4.60)

Функции

- long [nmppcFix64Exp01](#) (long nArg)

### 6.77.1 Подробное описание

Вычисление вычисления экспоненты числа в формате fixed-point (4.60)

Аргументы

|      |                                                       |
|------|-------------------------------------------------------|
| nVal | Входное число с фиксированной точкой в формате (4.60) |
|------|-------------------------------------------------------|

Возвращает

Экспонента числа в формате с фиксированной точкой (4.60)

### 6.77.2 Функции

#### 6.77.2.1 nmppcFix64Exp01()

```
long nmppcFix64Exp01 (
    long nArg )
```

Perfomance table 0

| nArg                | ticks | ticks/elem |
|---------------------|-------|------------|
| 0x10000000000000000 | 1316  | 1316.00    |
| 0x10000000000000000 | 1316  | 1316.00    |

The best configuration:

| nArg                | ticks | ticks/elem |
|---------------------|-------|------------|
| 0x10000000000000000 | 1316  | 1316.00    |

The worst configuration:

| nArg                | ticks | ticks/elem |
|---------------------|-------|------------|
| 0x10000000000000000 | 1316  | 1316.00    |

## 6.78 nmppsRand

Генерация случайного числа с равномерным распределением.

### Функции

- int `nmppcRandMinMaxDiv` (int nMin, int nMax, int nDivisible)
- int `nmppcRandMinMax` (int nMin, int nMax)
- int `nmppcRand` ()

### 6.78.1 Подробное описание

Генерация случайного числа с равномерным распределением.

#### Аргументы

|            |                                                   |
|------------|---------------------------------------------------|
| nMin       | Минимальное возможное значение случайного числа.  |
| nMax       | Максимальное возможное значение случайного числа. |
| nDivisible | Значение, которому будет кратно случайное число.  |

#### Возвращает

int Случайное число в диапазоне либо [nMin, nMax]. Для функции без параметров данный диапазон  $[-2^{31}; 2^{31}-1]$ .

### 6.78.2 Функции

#### 6.78.2.1 nmppcRandMinMaxDiv()

```
int nmppcRandMinMaxDiv (
    int nMin,
    int nMax,
    int nDivisible )
```

#### Perfomance table 0

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 0    | 10240 | 1          | 1259  | 1259.00    |
| 0    | 10240 | 1          | 1259  | 1259.00    |
| 128  | 10240 | 1          | 1241  | 1241.00    |
| 128  | 10240 | 1          | 1241  | 1241.00    |

The best configuration:

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 10240 | 1          | 1241  | 1241.00    |

The worst configuration:

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 0    | 10240 | 1          | 1259  | 1259.00    |

Perfomance table 1

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 1024  | 1          | 1277  | 1277.00    |
| 128  | 1024  | 1          | 1277  | 1277.00    |
| 128  | 10240 | 1          | 1279  | 1279.00    |
| 128  | 10240 | 1          | 1279  | 1279.00    |

The best configuration:

| nMin | nMax | nDivisible | ticks | ticks/elem |
|------|------|------------|-------|------------|
| 128  | 1024 | 1          | 1277  | 1277.00    |

The worst configuration:

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 10240 | 1          | 1279  | 1279.00    |

Perfomance table 2

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 10240 | 1          | 1282  | 1282.00    |
| 128  | 10240 | 1          | 1282  | 1282.00    |
| 128  | 10240 | 2          | 1238  | 1238.00    |
| 128  | 10240 | 2          | 1238  | 1238.00    |

The best configuration:

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 10240 | 2          | 1238  | 1238.00    |

The worst configuration:

| nMin | nMax  | nDivisible | ticks | ticks/elem |
|------|-------|------------|-------|------------|
| 128  | 10240 | 1          | 1282  | 1282.00    |

## 6.79 nmppcSqrt

Вычисление квадратного корня

Функции

- unsigned int nmppcSqrt\_64u (unsigned long long x)

### 6.79.1 Подробное описание

Вычисление квадратного корня

Аргументы

|   |               |
|---|---------------|
| x | Входное число |
|---|---------------|

Возвращает

Квадратный корень

## 6.80 Инициализация

### Группы

- [nmppsRand](#)

Генерация случайного числа с равномерным распределением.

#### 6.80.1 Подробное описание



## 6.81 Integer operations

### Группы

- [nmppsSqrt](#)

Вычисление квадратного корня

#### 6.81.1 Подробное описание

## 6.82 Fix-point 64

### Группы

- [nmppcFixSqrt64](#)  
Вычисление квадратного корня числа в формате fixed-point (32.32)
- [nmppcDoubleToFix64](#)  
Функция перевода из Fixed-Point 64 в Double.
- [nmppcFix64ToDouble](#)  
Преобразование 64p. числа с фиксированной точкой в число с плавающей точкой типа double.
- [nmppcFixDiv64](#)  
Деление двух целых чисел с записью результата в формате fixed-point.
- [nmppcFixSinCos64](#)  
Вычисление синуса и косинуса от аргумента в формате fixed-point (32.32)
- [nmppcFixArcTan64](#)  
Вычисление арктангенса от аргумента в формате fixed-point (32.32)

#### 6.82.1 Подробное описание

## 6.83 Fix-point 32

### Группы

- [nmppcFixExp32](#)  
Вычисление вычисления экспоненты числа в формате fixed-point (16.16)
- [nmppcFixSinCos32](#)  
Вычисление синуса и косинуса от аргумента в формате fixed-point (16.16)
- [nmppcFixArcTan32](#)  
Вычисление арктангенса от аргумента в формате fixed-point (16.16)
- [nmppcDoubleToFix32](#)  
Функция перевода из Fixed-Point (16.16) в Double.
- [nmppcFix32ToDouble](#)  
Преобразование 32р. числа с фиксированной точкой (16.16) в число с плавающей точкой типа double.
- [nmppcFixSqrt32](#)  
Вычисление квадратного корня числа в формате fixed-point (16.16)
- [nmppcFixMul32](#)  
Вычисление произведения двух числе в формате fixed-point (16.16)
- [nmppcFixInv32](#)  
Вычисление обратного значения целого числа с записью результата в формате fixed-point.
- [nmppcTblFixArcSin32](#)  
Вычисление функции arcsin по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)
- [nmppcTblFixArcCos32](#)  
Вычисление функции arccos по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)
- [nmppcTblFixCos32](#)  
Вычисление функции cos по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)
- [nmppcTblFixSin32](#)  
Вычисление функции sin по таблице. Входные и выходные значения задаются в формате fixed-point (16.16)
- [nmppcFixDivMod32](#)  
Вычисление частного и остатка при делении чисел с фиксированной запятой в формате fixed-point (16.16)
- [nmppcFix64Exp01](#)  
Вычисление вычисления экспоненты числа в формате fixed-point (4.60)

#### 6.83.1 Подробное описание

## 6.84 Арифметические операции

### Группы

- [nmprcDivC](#)  
частное двух комплексных чисел
- [nmprcProdC](#)  
произведение двух комплексных чисел.

#### 6.84.1 Подробное описание

## 6.85 Функции деинтерлейсинга

### Функции

- void `nmppiDeinterlaceSplit` (`nm8u` \*pSrcImg, int nSrcWidth, int nSrcHeight, `nm8u` \*pDstEven, `nm8u` \*pDstOdd)
- void `nmppiDeinterlaceBlend` (`nm8u` \*pSrcEven, `nm8u` \*pSrcOdd, int nSrcWidth, int nSrcHeight, `nm8u` \*pDst)

#### 6.85.1 Подробное описание

#### 6.85.2 Функции

##### 6.85.2.1 `nmppiDeinterlaceBlend()`

```
void nmppiDeinterlaceBlend (  
    nm8u * pSrcEven,  
    nm8u * pSrcOdd,  
    int nSrcWidth,  
    int nSrcHeight,  
    nm8u * pDst )
```

Функция объединяет два полукдра в один полный кадр.

#### Аргументы

|            |                                         |
|------------|-----------------------------------------|
| pSrcEven   | указатель на четный полукдр.            |
| nSrcOdd    | указатель на нечетный полукдр.          |
| nSrcWidth  | ширина исходных полукадров в пикселях.  |
| pSrcHeight | высота исходных полукадров в пикселях.  |
| pDst       | указатель на буфер результирующего кадр |

#### Возвращает

void

##### 6.85.2.2 `nmppiDeinterlaceSplit()`

```
void nmppiDeinterlaceSplit (  
    nm8u * pSrcImg,  
    int nSrcWidth,  
    int nSrcHeight,  
    nm8u * pDstEven,  
    nm8u * pDstOdd )
```

Функция разделяет кадр на два полукадра.

## Аргументы

|            |                                        |
|------------|----------------------------------------|
| pSrcImg    | указатель на исходный кадр.            |
| nSrcWidth  | ширина исходного кадра в пикселях.     |
| nSrcHeight | высота исходного кадра в пикселях.     |
| pDstEven   | указатель на буфер четного полукадра   |
| pDstOdd    | указатель на буфер нечетного полукадра |

## Возвращает

void

## 6.86 КИХ-фильтрация

Двумерная КИХ фильтрация

Двумерная КИХ фильтрация

## 6.87 Floodfill

Исполняет разделение бинарной картинки на односвязные области. Пример вызова: `no=VL_↵ FloodFill32b(pSrcImage, Tetr,Image, pTmpBuff, nSrcWidth, nSrcHeight);`.

### Функции

- `int nmppiFloodFill (unsigned int *pSrcImage, SegmentInfo *pSegmentInfo, unsigned int *p↵ SegmentImage, int nWidth, int nHeight, unsigned int *pTmpBuff)`
- `int FloodFill8 (void *src, void *dst, int nWidth, int nHeight, spot\_struct *spot, int lenSpot, unsigned *pixels, int mSpot, int dtFull, int dtSpot, int lDiag, int lDropSpot, ds\_struct *drop↵ Spot, int nPxlMin, int nPxlMax, int dXYmin, int dXYmax)`

Функция `FloodFill8` выполняет поиск пятен (сегментов, односвязных областей) во входной 8-битной матрице (изображения в градациях серого от 0 до 255), и строит такие же пятна в выходной матрице, заполняя их одним и тем же значением (цветом, соответствующим номеру пятна).

### 6.87.1 Подробное описание

Исполняет разделение бинарной картинки на односвязные области. Пример вызова: `no=VL_↵ FloodFill32b(pSrcImage, Tetr,Image, pTmpBuff, nSrcWidth, nSrcHeight);`.

#### Аргументы

|                           |                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <code>pSrcImage</code>    | Входное изображение                                                                                                          |
| <code>pSegmentInfo</code> | массив структур, где содержатся минимальные и максимальные координаты прямоугольника, описывающего сегментированную область. |
| <code>nWidth</code>       | ширина изображения                                                                                                           |
| <code>nHeight</code>      | высота изображения                                                                                                           |
| <code>pTmpBuff</code>     | Временный массив. Его размер должен быть $2*nWidth*nHeight$                                                                  |

#### Возвращаемые значения

|                            |                                                                                               |
|----------------------------|-----------------------------------------------------------------------------------------------|
| <code>pSegmentImage</code> | Результирующее "изображение", где все точки одного сегмента имеют одинаковое значение (1,...) |
|----------------------------|-----------------------------------------------------------------------------------------------|

#### Возвращает

Число сегментов на изображении

#### Заметки

Массив `pSegmentImage` должен быть обнулен. Функция изменяет массив `pSrcImage`.

### 6.87.2 Функции



## 6.87.2.1 FloodFill8()

```
int FloodFill8 (
    void * src,
    void * dst,
    int nWidth,
    int nHeight,
    spot_struct * spot,
    int lenSpot,
    unsigned * pixels,
    int mSpot,
    int dtFull,
    int dtSpot,
    int lDiag,
    int lDropSpot,
    ds_struct * dropSpot,
    int nPxlMin,
    int nPxlMax,
    int dXYmin,
    int dXYmax )
```

Функция FloodFill8 выполняет поиск пятен (сегментов, односвязных областей) во входной 8-битной матрице (изображения в градациях серого от 0 до 255), и строит такие же пятна в выходной матрице, заполняя их одним и тем же значением (цветом, соответствующим номеру пятна).

## Аргументы

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| src     | входная 8-битная матрица размером nHeight x nWidth.<br>Внимание. Входная матрица src модифицируется: <ul style="list-style-type: none"> <li>• в целях оптимизации программы обнуляются верхняя и нижняя строки, и крайний левый и крайний правый столбцы (границы матрицы),</li> <li>• в процессе обработки найденные пятна обнуляются.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| dst     | результатирующая 8-битная матрица того же размера nHeight x nWidth.<br>Внимание. Перед обращением к функции выходная матрица dst должна быть обнулена.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| nWidth  | ширина входной (и выходной) матрицы в 8-битных элементах.<br>Ограничения. Ширина матрицы должна быть кратна 32-битным словам: nWidth - положительное, (nWidth & 3) = 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| nHeight | высота входной (и выходной) матрицы.<br>Ограничения. Высота матрицы не должна превышать 1080 строк: nHeight - положительное, (nHeight <= 1080). Это ограничение можно обойти задав в начале файла FloodFill8.asm вместо 1080 нужное значение константы: const NHEIGHT = 1080;                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| spot    | массив, содержащий обобщенную информацию о найденных пятнах – минимально 6 параметров формата int для каждого пятна: Замечание1. Чтобы гарантированно избежать переполнения массива spot в общем случае (когда характер матрицы, количество, размер пятен неизвестны, и не задан параметр mSpot>0), размер массива spot в 32-битных словах должен быть: ( nSpotMax + 1 + [(nSpotMax-1)/255] ) * lenSpot, nSpotMax = [(nHeight-2)*(nWidth-2)/2 + 0.5], при lDiag=0, = [(nHeight-2)/2+0.5] * [(nWidth-2)/2+0.5], при lDiag=1, квадратные скобки [...] обозначают целую часть числа, напомним: 2 строки и 2 столбца входной матрицы обнулены. Замечание2. Если задан параметр mSpot>0, то массив spot достаточно определить для mSpot пятен (mSpot задается с учетом служебного и фиктивных пятен). |

## Аргументы

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| lenSpot | размер пятна в 32-битных словах, позволяет задать размер пятна более 6 слов, резервируя место для дополнительных параметров пятна довычисляемых другими функциями (например, угол наклона пятна к горизонтальной оси... ).<br>Замечание. Если параметр lenSpot задан некорректно lenSpot<6, то ему присваивается минимальное возможное значение lenSpot=6.                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| pixels  | массив координат пикселей, принадлежащих найденным пятнам, каждый элемент массива состоит из одного 32 битного слова, включающего в себя координаты пикселя в матрице src (или dst): unsigned int ij; биты [31..16] - i - номер строки, биты [15..0] - j - номер столбца. (Отсюда ограничение: nHeight, nWidth не могут быть больше $2^{16}-1 = 65535$ ).<br>Замечание. В общем случае (когда заранее неизвестно о количестве и размерах пятен в матрице), чтобы гарантированно избежать переполнения массива pixels, его размер должен быть (nHeight-2)*(nWidth-2) 32-битных слов (что почти в 4 раза больше входной 8-битной матрицы, напомним: 2 строки и 2 столбца входной матрицы обнулены).                                                                                                                               |
| mSpot   | число пятен, ограничивающее поиск. Если задано mSpot>0, то после обработки очередного пятна проверяется: достигло ли количество найденных пятен nSpot заданного mSpot (mSpot<=nSpot)? Если «Да», то происходит (принудительный) выход из программы с записью в spot[0].Ymin=2. Если задано mSpot=0, то проверок на количество найденных пятен не производится.<br>Замечание1. Если задать mSpot<=255, то пятна будут однозначно соответствовать своим номерам, то есть не будет пятен с повторяющимися номерами.<br>Замечание2. mSpot следует задавать с учетом служебного пятна (с номером 0) и фиктивных пятен (с номерами 256,512,768...), то есть, если поиск надо ограничить нахождением первых mSpotReal>0 пятен, то $mSpot = 1 + mSpotReal + [(mSpotReal-1)/255]$ квадратные скобки [...] обозначают целую часть числа   |
| dtFull  | время, выделенное на работу программы (в тактах процессора). Если задано dtFull>0, то после обработки очередного пятна (и в конце каждой строки) проверяется: осталось ли время на обработку следующего пятна dtRest<dtSpot? (dtRest=dtFull-(T-T0), T – текущее время, T0 – начало работы программы). Если «Нет», то происходит выход из программы с записью в spot[0].Ymin=1. Если задано dtFull=0, то проверок на достаточность времени на обработку следующего пятна не производится.<br>Замечание1. Задание dtFull>0 маленьким может значительно сократить время работы программы, однако, не гарантирует корректного завершения программы (например, когда обрабатываемое пятно занимает всю оставшуюся часть матрицы).<br>Замечание2. При выборе значения dtFull следует учитывать время обработки типового пятна dtSpot. |
| dtSpot  | время обработки типового пятна (в тактах процессора). Используется совместно с параметром dtFull при оценке достаточного времени на обработку следующего пятна.<br>Ограничение. Если задано dtFull>0, то dtSpot должно быть меньше него: dtSpot - неотрицательное, (dtFull>0)&&(dtSpot<dtFull).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## Аргументы

|           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| lDiag     | <p>параметр, определяющий связность пятна по диагоналям:</p> <ul style="list-style-type: none"> <li>• lDiag=0 - элементы являются соседними, если они граничат или по вертикали, или по горизонтали (по ходам ладьи, но не по ходам слона),</li> <li>• lDiag=1 - элементы являются соседними, если они граничат или по вертикали, или по горизонтали, или по диагоналям (по ходам ферзя).</li> </ul> <p>Замечание. Как показывает практика, результат обработки видеоизображений существенно не зависит от параметра lDiag, но при lDiag=1 значительно увеличивает время работы функции. Если нет особой необходимости, рекомендуется задавать lDiag=0.</p>                                                                                                                                                                                                                                           |
| lDropSpot | <p>параметр, задающий необходимость отбраковки найденных пятен по условиям НЕвхождения числа пикселей или размеров пятна в заданные диапазоны (см. ниже параметры nPxlmin, nPxlmax, dXYmin, dXYmax), статистика по отбракованным пятнам ведется в массиве dropSpot.</p> <p>lDropSpot=0 – найденные пятна НЕ отбраковываются, то есть все найденные пятна учитываются в массивах spot, pixels и отмечаются в выходной матрице dst, lDropSpot=1 – пятна с числом пикселей или размеров пятна не попадающими в заданные диапазоны отбраковываются, то есть не учитываются в массивах spot, pixels и не отмечаются в выходной матрице dst, статистика по отбракованным пятнам ведется в массиве dropSpot.</p> <p>Замечание. Задание параметра lDropSpot=1 приводит к увеличению работы функции и требует выделения памяти для массива dropSpot. Если нет особой необходимости, задавайте lDropSpot=0.</p> |
| dropSpot  | <p>– массив, в который опционально собирается статистика по отбракованным пятнам: на каждый из 4 признаков отбраковки: nPxlMin, nPxlMax, dXYmin, dXYmax, - отведено по 3 параметра формата int: dropSpot[4*3] +0 +1 +2</p> <ul style="list-style-type: none"> <li>• 0 – dropSpot(nPxlMin): [ nnSpot, nnPxl, dttSpot]</li> <li>• 3 - dropSpot(nPxlMax): [ nnSpot, nnPxl, dttSpot]</li> <li>• 6 - dropSpot(dXYmin) : [ nnSpot, nnPxl, dttSpot]</li> <li>• 9 - dropSpot(dXYmax) : [ nnSpot, nnPxl, dttSpot]</li> </ul> <p>Перед обращением к функции массив dropSpot (размером 12 int) должен быть обнулен.</p> <p>При ldropSpot=0 статистика не ведется и массив dropSpot не требуется, то есть вместо него можно задать фиктивный параметр, например, 0.</p>                                                                                                                                           |
| nPxlMin   | <p>минимальное число пикселей в пятне,<br/>при lDropSpot=1: пятно отбраковывается, если число его пикселей меньше nPxlMin,<br/>при lDropSpot=0 проверок не производится, параметр можно задать 0.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| nPxlMax   | <p>максимальное число пикселей в пятне,</p> <ul style="list-style-type: none"> <li>• при lDropSpot=1: пятно отбраковывается, если число его пикселей больше nPxlMax,</li> <li>• при lDropSpot=0 проверок не производится, параметр можно задать 0.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Аргументы

|        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dXYmin | <p>минимальная протяженность пятна по X или по Y в пикселах,</p> <ul style="list-style-type: none"> <li>при IDropSpot=1: пятно отбраковывается, если его протяженность меньше dXYMax,</li> <li>при IDropSpot=0 проверок не производится, но параметр используется для определения шага поиска по строкам входной матрицы src (и начальной строки поиска): <math>dI = \max(1, dXYmin)</math></li> </ul> <p>Замечание. Задание dXYmin&gt;1 приводит к просмотру входной матрицы src через каждые dXYmin строк, что может значительно сократить время работы функции, так как пятна меньшего размера, попавшие между строк поиска "обходятся" (однако, не гарантирует этого, например, когда пятно занимает всю матрицу). Поэтому, независимо от параметра IDropSpot надо задавать dXYmin, по возможности, большим.</p> |
| dXYmax | <p>максимальная протяженность пятна по X или по Y в пикселах,</p> <ul style="list-style-type: none"> <li>при IDropSpot=1: пятно отбраковывается, если его протяженность больше dXYMax,</li> <li>при IDropSpot=0 проверок не производится, параметр можно задать 0.</li> </ul> <p>Ограничения: nPxMin, nPxMax, dXYmin, dXYmax - неотрицательные, (nPxMin &lt;= nPxMax), (dXYmin &lt;= dXYmax).</p>                                                                                                                                                                                                                                                                                                                                                                                                                    |

## Возвращает

Функция возвращает целое число:

- либо положительное целое, тогда это - nSpot - число найденных пятен,
- либо отрицательное целое, тогда это - nError - код ошибки.

Число найденных пятен nSpot включает в себя также служебное и фиктивные пятна (с номерами 0,256,512,768...). Таким образом, число реальных пятен равно  $nSpotReal = nSpot - 1 - [nSpot/256]$

Если не найдено ни одного реального пятна, то nSpot=1 – одно служебное пятно.

Код ошибки возвращается, если в начале работы функции при проверке входных данных обнаружится недопустимая комбинация параметров: тогда происходит (принудительный) выход из программы с кодом ошибки:

```

dI = max(1,dXYmin);
if ( (nHeight<=0) || (nHeight < dI +2) || (1080<nHeight) ) return -1;
if ( (nWidth <=0) || (nWidth < dI +2) || ((nWidth & 3)!=0)) return -2;
if ( (dtSpot <0) || ((dtFull != 0) && (dtFull < dtSpot)) ) return -3;
if ( (nPxMin <0) || (nPxMax < nPxMin) ) return -4;
if ( (dXYmin <0) || (dXYmax < dXYmin) ) return -5;
if ( mSpot <0) return -6;

```

При нормальном выходе из функции (без ошибок) в параметре Ymin служебного пятна сообщается дополнительный признак завершения программы nExitCode:

- spot[0].Ymin=0 – вся матрица src просмотрена, найдены все пятна, естественный выход из программы.
- spot[0].Ymin=1 – принудительный выход из программы по исчерпанию заданного времени работы программы dtFull>0.
- spot[0].Ymin=2 – принудительный выход из программы по нахождению заданного количества пятен mSpot>0.

## Заметки

## Предельные случаи

Следует выделить предельные случаи:

- одно сплошное пятно на всю матрицу - максимальное время работы функции и максимальный размер массива `pixels` (но минимальный размер массива `spot`),
- «шахматная доска» - однопиксельные пятна, расположенные по диагоналям - максимальный размер массива `spot`.

## Алгоритм

В процессе поиска координаты всех пикселей каждого пятна сохраняются в массиве `pixels`:

- пиксели каждого пятна занимают непрерывную область в массиве `pixels`,
- параметр `spot[i].noPx1` *i*-го пятна в массиве `spot` указывает на начальный пиксел (*i*+1)-го пятна. В начале работы программы устанавливается указатель на начальный пиксел следующего пятна `noPx1 = 0`.

1. Цикл поиск пятна (осуществляется построчным попиксельным проходом матрицы `src`), если очередной пиксел нулевой, то переход к следующему пикселу иначе (встретился ненулевой пиксел)

2. начало обработки (очередного) пятна:

- засечь время начала обработки пятна `tSpot0=clock()`;
- устанавливаются указатели начала и конца очереди пикселей пятна в массиве `pixels`: `noPx10 = noPx11 = noPx1`.
- пиксел в матрице `src` обнуляется,
- координаты пиксела заносятся в массив `pixels`, указатель `noPx11` увеличивается на 1

3. while-цикл обработки пятна:

- выбирается пиксел `noPx10` из массива `pixels`, указатель `noPx10` увеличивается на 1
  - просматриваются все его соседние элементы (с учетом параметра `lDiag`) координаты ненулевых пикселей записываются в массив `pixels`
  - пиксели в матрице `src` обнуляются
  - если указатели `noPx10`, `noPx11` не равны (не все пиксели пятна и их соседи просмотрены), то переход на начало while-цикла
- // пятно найдено! `noPx10=noPx11` - указывают на начальный пиксел  
// следующего пятна в массиве `pixels`.

Вычисление минимального прямоугольника (`Xmin`, `Ymin`, `Xmax`, `Ymax`), объемлющего пятно

Если задан параметр `lDropSpot=1`, то проверка условий отбраковки пятна по признакам `nPx1Min`, `nPx1Max`, `dXYmin`, `dXYmax` если пятно отбраковано, то

формирование записи массива `dropSpot` по соответствующему признаку, переход на поиск следующего пятна (`noPx1` не изменяется).

Меняется указатель `noPx1=noPx11`.

Формирование параметров данного пятна в массиве `spot[i] =`

(`Xmin`, `Ymin`, `Xmax`, `Ymax`, `noPx1`, 0, 0, 0)

Если задан параметр вычисления угла наклона пятна (`lAngleSpot=1`), то

вычисление `codeAngle`, `lBadSpot` и запись в `spot[i] =`

(., ., ., ., ., `codeAngle`, `lBadSpot`, .)

Запись `dtSpot=clock()-tSpot0` в `spot[i] =` (., ., ., ., ., ., `dtSpot`).

Выбор следующего пиксела, переход на начало цикла поиска очередного пятна (1).

4. Конец программы.

## 6.88 nmppiConvert

Преобразование типов для элементов изображения.

Функция конвертирует элементы RGB между 8-м разрядным представлением и 10-ти разрядным.

### Функции

- void nmppiConvertRGB32\_8u10u (RGB32\_nm8u \*pSrcImg, RGB32\_nm10u \*pDstImg, int nSize)
- void nmppiConvertRGB32\_10u8u (RGB32\_nm10u \*pSrcImg, RGB32\_nm8u \*pDstImg, int nSize)

### 6.88.1 Подробное описание

Преобразование типов для элементов изображения.

Функция конвертирует элементы RGB между 8-м разрядным представлением и 10-ти разрядным.

#### Аргументы

|         |                                                          |
|---------|----------------------------------------------------------|
| pSrcImg | Указатель на первый элемент исходного изображения.       |
| pDstImg | Указатель на первый элемент результирующего изображения. |
| nSize   | Количество элементов в изображении.                      |

#### Возвращает

void

#### Restrictions:

- Указатели на изображения должны быть выровнены в памяти на границу 64-х разрядного слова;

## 6.89 nmppiRGB32ToGray

Преобразование пикселей из RGB в яркость

### Функции

- void nmppiRGB32ToGray\_8u32u (RGB32\_nm8u \*pRGB, nm32u \*pDstGray, int nSize)
- void nmppiRGB32ToGray\_8u32s (RGB32\_nm8u \*pRGB, nm32s \*pDstGray, int nSize)
- void nmppiRGB32ToGray\_8u8s (RGB32\_nm8u \*pRGB, nm8s \*pDstGray, int nSize, void \*p←TmpBuf)
- void nmppiRGB32ToGray\_10s32s (RGB32\_nm10s \*pRGB, nm32s \*pDstGray, int nSize)
- void nmppiRGB32ToGray\_10s32u (RGB32\_nm10s \*pRGB, nm32u \*pDstGray, int nSize)
- void nmppiRGB32ToGray\_10s8s (RGB32\_nm10s \*pRGB, nm8s \*pDstGray, int nSize, void \*p←TmpBuf)

### 6.89.1 Подробное описание

Преобразование пикселей из RGB в яркость

#### Аргументы

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pRGB | Вход, по 4 байта на пиксель. Порядок байтов B, G, R, 0. Результат в виде 32 битных целых чисел, в которых полезные данные занимают младшие 24 бита. Для получения восьмибитовых пикселей необходимо вырезать биты 16..23, например, с помощью nmppsClipRShiftConvert_Add_32s(nm32s* pSrcVec, int nClipFactor,int nShift, nm64u* nAddValue,nm8s* pDstVec, int nSize); с параметрами nClipFactor=24, nShift=16. Количество пикселей на входе и выходе. nSize=[64,128,192,...] Временный массив размером nm32s[nSize] . |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## 6.90 Переупорядочивание изображений

### Группы

- [Блочное переупорядочивание](#)

Блочное перупорядочивание изображений.

#### 6.90.1 Подробное описание



## 6.91 Блочное переупорядочивание

Блочное переупорядочивание изображений.

### Группы

- [nmppiSplitIntoBlocks](#)  
Преобразует изображение в последовательность квадратных блоков.
- [nmppiMergeFromBlocks](#)  
Объединяет последовательность квадратных блоков в изображение.

#### 6.91.1 Подробное описание

Блочное переупорядочивание изображений.

## 6.92 nmppiSplitIntoBlocks

Преобразует изображение в последовательность квадратных блоков.

Функции

- void [nmppiSplitIntoBlocks8x8](#) ([nm8s](#) \*pSrcImg, [nm8s](#) \*pDstBlockSeq, int nWidth, int nHeight)

### 6.92.1 Подробное описание

Преобразует изображение в последовательность квадратных блоков.

Аргументы

|         |                                                                                                                                                                                                                     |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pSrcImg | Исходное изображение Выходная последовательность блоков Ширина исходного изображения в пикселах. nWidth =[8,16,24...] Высота исходного изображения в пикселах. nHeight =[8,16,24...] Исходное изображение имеет вид |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

```
[A00|A01|A02|A03|A04|A05|A06|A07|B00|B01|...|B07|C00|...H07|
A10|A11|A12|A13|A14|A15|A16|A17|B10|B11|...|B17|C10|...H17|
A20|A21|A22|A23|A24|A25|A26|A27|B20|B21|...|B27|C20|...H27|
A30|A31|A32|A33|A34|A35|A36|A37|B30|B31|...|B37|C30|...H37|
.....
A70|A71|A72|A73|A74|A75|A76|A77|B70|B71|...|B77|C70|...H77|
.....
I00|I01|I02|I03|I04|I05|I06|I07|J00|J01|.....]
I10|I11|I12|I13|I14|I15|I16|I17|J00|I09|.....
.....Z77]
```

Выходная последовательность блоков имеет вид

```
[A00|A01|A02|...|A06|A07|A20|A21|...|A77|B00|B01|B02|...|B07|B10|...|H77|
I00|I01|I02|...|I06|I07|I10|I11|...|I77|J00|J01|J02|.....|Z77|
```

### 6.92.2 Функции

#### 6.92.2.1 nmppiSplitIntoBlocks8x8()

```
void nmppiSplitIntoBlocks8x8 (
    nm8s * pSrcImg,
    nm8s * pDstBlockSeq,
    int nWidth,
    int nHeight )
```

Perfomance table 0

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im0     | im0          | 128     | 128    | 4340  | 0.26       |
| im0     | im0          | 128     | 128    | 4340  | 0.26       |
| im0     | im1          | 128     | 128    | 3207  | 0.19       |
| im0     | im1          | 128     | 128    | 3207  | 0.19       |
| im1     | im0          | 128     | 128    | 3205  | 0.19       |
| im1     | im0          | 128     | 128    | 3205  | 0.19       |
| im1     | im1          | 128     | 128    | 5428  | 0.33       |
| im1     | im1          | 128     | 128    | 5428  | 0.33       |

The best configuration:

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im1     | im0          | 128     | 128    | 3205  | 0.19       |

The worst configuration:

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im1     | im1          | 128     | 128    | 5428  | 0.33       |

Perfomance table 1

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im0     | im1          | 128     | 8      | 404   | 0.39       |
| im0     | im1          | 128     | 8      | 404   | 0.39       |
| im0     | im1          | 128     | 32     | 903   | 0.22       |
| im0     | im1          | 128     | 32     | 903   | 0.22       |
| im0     | im1          | 128     | 128    | 3207  | 0.19       |
| im0     | im1          | 128     | 128    | 3207  | 0.19       |

The best configuration:

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im0     | im1          | 128     | 8      | 404   | 0.39       |

The worst configuration:

| pSrcImg | pDstBlockSeq | nHeight | nWidth | ticks | ticks/elem |
|---------|--------------|---------|--------|-------|------------|
| im0     | im1          | 128     | 128    | 3207  | 0.19       |

## 6.93 nmppiMergeFromBlocks

Объединяет последовательность квадратных блоков в изображение.

Функции

- void [nmppiMergeFromBlocks8x8](#) ([nm8s](#) \*pSrcBlockSeq, [nm8s](#) \*pDstImg, int nWidth, int nHeight)

### 6.93.1 Подробное описание

Объединяет последовательность квадратных блоков в изображение.

Аргументы

|              |                                                                                                                                                                                                                                       |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pSrcBlockSeq | Входная последовательность блоков . Результирующее изображение Ширина исходного изображения в пикселах. nWidth=[8,16,24...] Высота исходного изображения в пикселах. nHeight=[8,16,24...] Входная последовательность блоков имеет вид |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

```
[A00|A01|A02|...|A06|A07|A20|A21|...|A77|B00|B01|B02|...|B07|B10|...|H77|
[I00|I01|I02|...|I06|I07|I10|I11|...|I77|J00|J01|J02|.....|Z77]
```

Результирующее изображение имеет вид

```
[A00|A01|A02|A03|A04|A05|A06|A07|B00|B01|..|B07|C00|...H07]
[A10|A11|A12|A13|A14|A15|A16|A17|B10|B11|..|B17|C10|...H17]
[A20|A21|A22|A23|A24|A25|A26|A27|B20|B21|..|B27|C20|...H27]
[A30|A31|A32|A33|A34|A35|A36|A37|B30|B31|..|B37|C30|...H37]
.....
[A70|A71|A72|A73|A74|A75|A76|A77|B70|B71|..|B77|C70|...H77]
.....
[I00|I01|I02|I03|I04|I05|I06|I07|J00|J01|.....]
[I10|I11|I12|I13|I14|I15|I16|I17|J00|J09|.....]
.....Z77]
```

### 6.93.2 Функции

#### 6.93.2.1 nmppiMergeFromBlocks8x8()

```
void nmppiMergeFromBlocks8x8 (
    nm8s * pSrcBlockSeq,
    nm8s * pDstImg,
    int nWidth,
    int nHeight )
```

Perfomance table 0

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im0          | im0     | 128     | 128    | 4354  | 0.26       |
| im0          | im0     | 128     | 128    | 4354  | 0.26       |
| im0          | im1     | 128     | 128    | 3033  | 0.18       |
| im0          | im1     | 128     | 128    | 3033  | 0.18       |
| im1          | im0     | 128     | 128    | 3059  | 0.18       |
| im1          | im0     | 128     | 128    | 3059  | 0.18       |
| im1          | im1     | 128     | 128    | 5362  | 0.32       |
| im1          | im1     | 128     | 128    | 5362  | 0.32       |

The best configuration:

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im0          | im1     | 128     | 128    | 3033  | 0.18       |

The worst configuration:

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im1          | im1     | 128     | 128    | 5362  | 0.32       |

Perfomance table 1

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im0          | im1     | 128     | 8      | 389   | 0.37       |
| im0          | im1     | 128     | 8      | 389   | 0.37       |
| im0          | im1     | 128     | 32     | 897   | 0.21       |
| im0          | im1     | 128     | 32     | 897   | 0.21       |
| im0          | im1     | 128     | 128    | 3033  | 0.18       |
| im0          | im1     | 128     | 128    | 3033  | 0.18       |

The best configuration:

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im0          | im1     | 128     | 8      | 389   | 0.37       |

The worst configuration:

| pSrcBlockSeq | pDstImg | nHeight | nWidth | ticks | ticks/elem |
|--------------|---------|---------|--------|-------|------------|
| im0          | im1     | 128     | 128    | 3033  | 0.18       |

## 6.94 nmpriFree

Освобождение памяти для изображений.

### Функции

- `void nmpriFree (void *ptr)`

### 6.94.1 Подробное описание

Освобождение памяти для изображений.

### Заметки

Данная функция должна вызываться только для векторов, распределенных с помощью функций `nmpriMalloc`.

## 6.95 nmppiRelease

Освобождение блоков памяти, выделенных функциями nmppiCreate\*\*\*.

Функции

- `__INLINE__ void nmppiReleaseObject (nm64s *pKernel)`

### 6.95.1 Подробное описание

Освобождение блоков памяти, выделенных функциями nmppiCreate\*\*\*.

## 6.96 Арифметические действия



## 6.97 Масочная фильтрация

### Группы

- [КИХ-фильтрация](#)

Двумерная КИХ фильтрация

#### 6.97.1 Подробное описание

## 6.98 Инициализация и копирование

### Группы

- [nmpriConvert](#)

Преобразование типов для элементов изображения.

Функция конвертирует элементы RGB между 8-м разрядным представлением и 10-ти разрядным.

- [nmpriRGB32ToGray](#)

Преобразование пикселей из RGB в яркость

### 6.98.1 Подробное описание

## 6.99 Функции поддержки

### Группы

- [nmpriFree](#)  
Освобождение памяти для изображений.
- [nmpriRelease](#)  
Освобождение блоков памяти, выделенных функциями `nmpriCreate***`.

### 6.99.1 Подробное описание

## 6.100 Функции графического вывода текста

- `int nmppiPrint8x15 (char *str, void *img, int imgWidth, int x, int y, int FGcolor, int BGcolor)`  
Prints text over 8-bit grayscale image buffer with 8x15 font size.
- `char * hex2ascii (int value, char *str)`  
Converts integer to 8-byte string in hexadecimal base.

### 6.100.1 Подробное описание

### 6.100.2 Функции

#### 6.100.2.1 `hex2ascii()`

```
char* hex2ascii (  
    int value,  
    char * str )
```

Converts integer to 8-byte string in hexadecimal base.

Аргументы

|    |       |                                                                     |
|----|-------|---------------------------------------------------------------------|
| in | value | Value to be converted to a string                                   |
| in | str   | Array in memory where to store the resulting null-terminated string |

Возвращает

A pointer to the resulting null-terminated string, same as parameter str.

Details

#### 6.100.2.2 `nmppiPrint8x15()`

```
int nmppiPrint8x15 (  
    char * str,  
    void * img,  
    int imgWidth,  
    int x,  
    int y,  
    int FGcolor,  
    int BGcolor )
```

Prints text over 8-bit grayscale image buffer with 8x15 font size.

## Аргументы

|    |          |                                                                 |
|----|----------|-----------------------------------------------------------------|
| in | str      | String to be printed                                            |
| in | img      | Pointer to 8-bit grayscale image , where text should be printed |
| in | imgWidth | Image width                                                     |
| in | x        | Horizontal positon in image where text should be printed        |
| in | y        | Vertical positon in image where text should be printed          |
| in | FGcolor  | Foreground text color                                           |
| in | BGcolor  | Background text color                                           |

## Возвращает

x-coordinate behind printed text

## Заметки

Russian text is supported in win-1251 codepage

## 6.101 Инициализация и копирование

### Группы

- [nmprpmCopyua](#)

Копирование подматрицы с невыровненной по границам 64- разрядного слова позиции в выровненную.

- [MTR\\_Copyau](#)

Копирование подматрицы с выровненной по границе 64-х разрядных слов позиции в невыровненную позицию.

- [MTR\\_Copy](#)

Функции копирования прямоугольных областей памяти между двумерными массивами.

### 6.101.1 Подробное описание

## 6.102 nmppmCopyua

Копирование подматрицы с невыровненной по границам 64- разрядного слова позиции в выровненную.

### Функции

- void `nmppmCopyua_8s` (`nm8s` \*pSrcMtr, int nSrcStride, int nSrcOffset, `nm8s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopyua_16s` (`nm16s` \*pSrcMtr, int nSrcStride, int nSrcOffset, `nm16s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopyua_32s` (`nm32s` \*pSrcMtr, int nSrcStride, int nSrcOffset, `nm32s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)

### 6.102.1 Подробное описание

Копирование подматрицы с невыровненной по границам 64- разрядного слова позиции в выровненную.

#### Аргументы

|            |                                                   |
|------------|---------------------------------------------------|
| pSrcMtr    | Исходная матрица.                                 |
| nSrcStride | Ширина исходной матрицы в элементах.              |
| nSrcOffset | Смещение в элементах от начала матрицы-источника. |
| nDstStride | Ширина результирующей матрицы в элементах.        |
| nHeight    | Число строк подматрицы.                           |
| nWidth     | Число столбцов подматрицы.                        |

#### Возвращаемые значения

|         |                         |
|---------|-------------------------|
| pDstMtr | Результирующая матрица. |
|---------|-------------------------|

#### Возвращает

void

#### Restrictions:

- Входная и выходная матрицы не должны перекрываться в памяти
- Число столбцов копируемой подматрицы nWidth должна быть кратно числу элементов в 64-х разрядном слове.

### 6.102.2 Функции

## 6.102.2.1 nmppmCopyua\_16s()

```
void nmppmCopyua_16s (
    nm16s * pSrcMtr,
    int nSrcStride,
    int nSrcOffset,
    nm16s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 906   | 1.76       |
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 906   | 1.76       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 1042  | 2.03       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 1042  | 2.03       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1170  | 2.28       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1170  | 2.28       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 906   | 1.76       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1170  | 2.28       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 946   | 3.69       |



| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 946   | 3.69       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 946   | 3.69       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 304   | 4.75       |
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 946   | 3.69       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1138  | 2.22       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1138  | 2.22       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 304   | 4.75       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1138  | 2.22       |

#### 6.102.2.2 nmppmCopyua\_32s()

```
void nmppmCopyua_32s (
    nm32s * pSrcMtr,
    int nSrcStride,
    int nSrcOffset,
    nm32s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 1162  | 2.26       |
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 1162  | 2.26       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 1418  | 2.76       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 1418  | 2.76       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1674  | 3.26       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1674  | 3.26       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 1162  | 2.26       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 1674  | 3.26       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 460   | 7.18       |

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 460   | 7.18       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 1530  | 2.98       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 1530  | 2.98       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 460   | 7.18       |
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 1138  | 4.44       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 1530  | 2.98       |

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1530  | 2.98       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 1530  | 2.98       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 460   | 7.18       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 1530  | 2.98       |

### 6.102.2.3 nmppmCopyua\_8s()

```
void nmppmCopyua_8s (
    nm8s * pSrcMtr,
    int nSrcStride,
    int nSrcOffset,
    nm8s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 842   | 1.64       |
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 842   | 1.64       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 914   | 1.78       |
| im1     | im0     | 128             | 8               | 8       | 64     | 256             | 914   | 1.78       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 978   | 1.91       |
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 978   | 1.91       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 128             | 8               | 8       | 64     | 256             | 842   | 1.64       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im1     | im1     | 128             | 8               | 8       | 64     | 256             | 978   | 1.91       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 32     | 32              | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 16              | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 16              | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 32     | 32              | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 64              | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 8      | 32              | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 8      | 128             | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 8      | 256             | 226   | 3.53       |
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 460   | 1.79       |

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 8               | 8       | 32     | 32              | 460   | 1.79       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 128             | 8               | 8       | 32     | 128             | 460   | 1.79       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 128             | 8               | 8       | 32     | 256             | 460   | 1.79       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 32              | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 128             | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 946   | 1.84       |
| im0     | im1     | 128             | 8               | 8       | 64     | 256             | 946   | 1.84       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 8      | 32              | 226   | 3.53       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nSrc↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 16              | 8               | 8       | 64     | 32              | 946   | 1.84       |

## 6.103 MTR\_Соруаи

Копирование подматрицы с выровненной по границе 64-х разрядных слов позиции в невыровненную позицию.

### Функции

- void `nmppmCopyau_8s` (`nm8s` \*pSrcMtr, int nSrcStride, `nm8s` \*pDstMtr, int nDstStride, int nDstOffset, int nHeight, int nWidth)
- void `nmppmCopyau_16s` (`nm16s` \*pSrcMtr, int nSrcStride, `nm16s` \*pDstMtr, int nDstStride, int nDstOffset, int nHeight, int nWidth)
- void `nmppmCopyau_32s` (`nm32s` \*pSrcMtr, int nSrcStride, `nm32s` \*pDstMtr, int nDstStride, int nDstOffset, int nHeight, int nWidth)

### 6.103.1 Подробное описание

Копирование подматрицы с выровненной по границе 64-х разрядных слов позиции в невыровненную позицию.

#### Аргументы

|            |                                      |
|------------|--------------------------------------|
| pSrcMtr    | Исходная матрица.                    |
| nSrcStride | Ширина исходной матрицы в элементах. |

#### Возвращаемые значения

|         |                         |
|---------|-------------------------|
| pDstMtr | Результирующая матрица. |
|---------|-------------------------|

#### Аргументы

|            |                                                         |
|------------|---------------------------------------------------------|
| nDstStride | Ширина результирующей матрицы в элементах.              |
| nDstOffset | Индекс столбца (в элементах) куда производится вставка. |
| nHeight    | Число строк подматрицы.                                 |
| nWidth     | Число столбцов подматрицы.                              |

#### Возвращает

void

#### Restrictions:

- Входная и выходная матрицы не должны перекрываться в памяти
- Число столбцов копируемой подматрицы nWidth должна быть кратной 64-битам.

## 6.103.2 Функции

## 6.103.2.1 nmppmCopyau\_8s()

```
void nmppmCopyau_8s (
    nm8s * pSrcMtr,
    int nSrcStride,
    nm8s * pDstMtr,
    int nDstStride,
    int nDstOffset,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 32              | 32              | 32      | 32     | 32              | 765   | 0.74       |
| im0     | im0     | 32              | 32              | 32      | 32     | 32              | 765   | 0.74       |
| im0     | im1     | 32              | 32              | 32      | 32     | 32              | 1029  | 1.00       |
| im0     | im1     | 32              | 32              | 32      | 32     | 32              | 1029  | 1.00       |
| im1     | im0     | 32              | 32              | 32      | 32     | 32              | 869   | 0.84       |
| im1     | im0     | 32              | 32              | 32      | 32     | 32              | 869   | 0.84       |
| im1     | im1     | 32              | 32              | 32      | 32     | 32              | 1085  | 1.05       |
| im1     | im1     | 32              | 32              | 32      | 32     | 32              | 1085  | 1.05       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im0     | 32              | 32              | 32      | 32     | 32              | 765   | 0.74       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im1     | im1     | 32              | 32              | 32      | 32     | 32              | 1085  | 1.05       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 32              | 8       | 16     | 64              | 456   | 3.56       |
| im0     | im1     | 128             | 32              | 8       | 16     | 64              | 456   | 3.56       |
| im0     | im1     | 128             | 32              | 8       | 16     | 128             | 456   | 3.56       |
| im0     | im1     | 128             | 32              | 8       | 16     | 128             | 456   | 3.56       |
| im0     | im1     | 128             | 32              | 8       | 32     | 64              | 612   | 2.39       |
| im0     | im1     | 128             | 32              | 8       | 32     | 64              | 612   | 2.39       |



| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 32              | 8       | 32     | 128             | 612   | 2.39       |
| im0     | im1     | 128             | 32              | 8       | 32     | 128             | 612   | 2.39       |
| im0     | im1     | 128             | 64              | 8       | 16     | 64              | 456   | 3.56       |
| im0     | im1     | 128             | 64              | 8       | 16     | 64              | 456   | 3.56       |
| im0     | im1     | 128             | 64              | 8       | 16     | 128             | 456   | 3.56       |
| im0     | im1     | 128             | 64              | 8       | 16     | 128             | 456   | 3.56       |
| im0     | im1     | 128             | 64              | 8       | 32     | 64              | 612   | 2.39       |
| im0     | im1     | 128             | 64              | 8       | 32     | 64              | 612   | 2.39       |
| im0     | im1     | 128             | 64              | 8       | 32     | 128             | 612   | 2.39       |
| im0     | im1     | 128             | 64              | 8       | 32     | 128             | 612   | 2.39       |

The best configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 32              | 8       | 16     | 64              | 456   | 3.56       |

The worst configuration:

| pSrcMtr | pDstMtr | nDst↔<br>Stride | nDst↔<br>Offset | nHeight | nWidth | nSrc↔<br>Stride | ticks | ticks/elem |
|---------|---------|-----------------|-----------------|---------|--------|-----------------|-------|------------|
| im0     | im1     | 128             | 32              | 8       | 32     | 64              | 612   | 2.39       |

## 6.104 MTR\_Copy

Функции копирования прямоугольных областей памяти между двумерными массивами.

### Функции

- void `nmppmCopy_1` (`nm1` \*pSrcMtr, int nSrcStride, `nm1` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_2s` (`nm2s` \*pSrcMtr, int nSrcStride, `nm2s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_4s` (`nm4s` \*pSrcMtr, int nSrcStride, `nm4s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_8s` (`nm8s` \*pSrcMtr, int nSrcStride, `nm8s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_16s` (`nm16s` \*pSrcMtr, int nSrcStride, `nm16s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_32s` (`nm32s` \*pSrcMtr, int nSrcStride, `nm32s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)
- void `nmppmCopy_64s` (`nm64s` \*pSrcMtr, int nSrcStride, `nm64s` \*pDstMtr, int nDstStride, int nHeight, int nWidth)

### 6.104.1 Подробное описание

Функции копирования прямоугольных областей памяти между двумерными массивами.

#### Аргументы

|            |                                            |
|------------|--------------------------------------------|
| pSrcMtr    | Исходная матрица.                          |
| nSrcStride | Ширина исходной матрицы в элементах.       |
| nDstStride | Ширина результирующей матрицы в элементах. |
| nHeight    | Число строк подматрицы.                    |
| nWidth     | Число столбцов подматрицы.                 |

#### Возвращаемые значения

|         |                         |
|---------|-------------------------|
| pDstMtr | Результирующая матрица. |
|---------|-------------------------|

#### Возвращает

void

#### Restrictions:

Выходная матрица не может быть перезаписана (т.е. помещена непосредственно на место входной)

## 6.104.2 Функции

## 6.104.2.1 nmppmCopy\_1()

```
void nmppmCopy_1 (
    nm1 * pSrcMtr,
    int nSrcStride,
    nm1 * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 1394  | 0.02       |
| im0     | im1     | 128        | 256     | 256    | 128        | 1394  | 0.02       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 1394  | 0.02       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 1394  | 0.02       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 478   | 0.02       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 478   | 0.02       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 834   | 0.02       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 834   | 0.02       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 484   | 0.02       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 484   | 0.02       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 846   | 0.02       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 846   | 0.02       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 484   | 0.02       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 484   | 0.02       |

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 256        | 128     | 256    | nDstStride | 846   | 0.02       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 846   | 0.02       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 128     | 256    | nDstStride | 846   | 0.02       |

#### 6.104.2.2 nmppmCopy\_16s()

```
void nmppmCopy_16s (
    nm16s * pSrcMtr,
    int nSrcStride,
    nm16s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 18106 | 0.27       |
| im0     | im1     | 128        | 256     | 256    | 128        | 18106 | 0.27       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 18106 | 0.27       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 18106 | 0.27       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 484   | 0.47       |

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 484   | 0.47       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 5914  | 0.36       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 5914  | 0.36       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 11706 | 0.35       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 11706 | 0.35       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 484   | 0.47       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 484   | 0.47       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 5914  | 0.36       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 5914  | 0.36       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 11706 | 0.35       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 11706 | 0.35       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 420   | 0.41       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 420   | 0.41       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 4890  | 0.29       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 4890  | 0.29       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 9658  | 0.29       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 9658  | 0.29       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 256        | 128     | 8      | nDstStride | 420   | 0.41       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 256    | nDstStride | 11706 | 0.35       |

### 6.104.2.3 nmppmCopy\_2s()

```
void nmppmCopy_2s (
    nm2s * pSrcMtr,
    int nSrcStride,
    nm2s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 2666  | 0.04       |
| im0     | im1     | 128        | 256     | 256    | 128        | 2666  | 0.04       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 2666  | 0.04       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 2666  | 0.04       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 834   | 0.05       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 834   | 0.05       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 1546  | 0.04       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 1546  | 0.04       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 846   | 0.05       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 846   | 0.05       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 1570  | 0.04       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 1570  | 0.04       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 846   | 0.05       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 846   | 0.05       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 1570  | 0.04       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 1570  | 0.04       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 300   | 0.29       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 128     | 256    | nDstStride | 1570  | 0.04       |

#### 6.104.2.4 nmppmCopy\_32s()

```
void nmppmCopy_32s (
    nm32s * pSrcMtr,
```

```

    int nSrcStride,
    nm32s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )

```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 38138 | 0.58       |
| im0     | im1     | 128        | 256     | 256    | 128        | 38138 | 0.58       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 38138 | 0.58       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 38138 | 0.58       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 846   | 0.82       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 846   | 0.82       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 11706 | 0.71       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 11706 | 0.71       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 23287 | 0.71       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 23287 | 0.71       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 718   | 0.70       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 718   | 0.70       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 9658  | 0.58       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 9658  | 0.58       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 21357 | 0.65       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 21357 | 0.65       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 798   | 0.77       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 798   | 0.77       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 10938 | 0.66       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 10938 | 0.66       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 20493 | 0.62       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 20493 | 0.62       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 128     | 8      | nDstStride | 718   | 0.70       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 256    | nDstStride | 23287 | 0.71       |

#### 6.104.2.5 nmppmCopy\_4s()

```
void nmppmCopy_4s (
    nm4s * pSrcMtr,
    int nSrcStride,
    nm4s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 5210  | 0.07       |
| im0     | im1     | 128        | 256     | 256    | 128        | 5210  | 0.07       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 5210  | 0.07       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 5210  | 0.07       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 302   | 0.29       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 302   | 0.29       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 1562  | 0.09       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 1562  | 0.09       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 3002  | 0.09       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 3002  | 0.09       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 1570  | 0.09       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 1570  | 0.09       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 3018  | 0.09       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 3018  | 0.09       |



| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 1570  | 0.09       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 1570  | 0.09       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 3018  | 0.09       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 3018  | 0.09       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 302   | 0.29       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 128     | 256    | nDstStride | 3018  | 0.09       |

#### 6.104.2.6 nmppmCopy\_64s()

```
void nmppmCopy_64s (
    nm64s * pSrcMtr,
    int nSrcStride,
    nm64s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 76824 | 1.17       |
| im0     | im1     | 128        | 256     | 256    | 128        | 76824 | 1.17       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 76824 | 1.17       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 76824 | 1.17       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 1570  | 1.53       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 1570  | 1.53       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 23287 | 1.42       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 23287 | 1.42       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 40823 | 1.24       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 40823 | 1.24       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 1474  | 1.43       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 1474  | 1.43       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 20493 | 1.25       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 20493 | 1.25       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 37352 | 1.13       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 37352 | 1.13       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 1426  | 1.39       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 1426  | 1.39       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 21423 | 1.30       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 21423 | 1.30       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 38488 | 1.17       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 38488 | 1.17       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 256        | 128     | 8      | nDstStride | 1426  | 1.39       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 256    | nDstStride | 40823 | 1.24       |

#### 6.104.2.7 nmppmCopy\_8s()

```
void nmppmCopy_8s (
    nm8s * pSrcMtr,
    int nSrcStride,
    nm8s * pDstMtr,
    int nDstStride,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 10298 | 0.15       |
| im0     | im1     | 128        | 256     | 256    | 128        | 10298 | 0.15       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 10298 | 0.15       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 128        | 256     | 256    | 128        | 10298 | 0.15       |

Perfomance table 1

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 8          | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 8          | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 5914  | 0.18       |
| im0     | im1     | 8          | 128     | 256    | nDstStride | 5914  | 0.18       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 128        | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 5914  | 0.18       |
| im0     | im1     | 128        | 128     | 256    | nDstStride | 5914  | 0.18       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 8      | nDstStride | 303   | 0.29       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 256        | 128     | 128    | nDstStride | 3018  | 0.18       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 5914  | 0.18       |
| im0     | im1     | 256        | 128     | 256    | nDstStride | 5914  | 0.18       |

The best configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 8      | nDstStride | 303   | 0.29       |

The worst configuration:

| pSrcMtr | pDstMtr | nDstStride | nHeight | nWidth | nSrcStride | ticks | ticks/elem |
|---------|---------|------------|---------|--------|------------|-------|------------|
| im0     | im1     | 8          | 128     | 256    | nDstStride | 5914  | 0.18       |

## 6.105 MTR\_fpResolve\_Gauss

Решение системы линейных уравнений методом Гаусса.

Функции

- void MTR\_fpResolve\_Gauss (double \*pSrcMtrA, double \*pSrcVecB, double \*pDstVecX, int nSize)

### 6.105.1 Подробное описание

Решение системы линейных уравнений методом Гаусса.

$$A \cdot x = b$$

Аргументы

|          |                                  |
|----------|----------------------------------|
| pSrcMtrA | Квадтратная матрица уравнения A. |
| pSrcVecB | Свободный вектор.                |
| nSize    | Размер матрицы.                  |

Возвращаемые значения

|          |                   |
|----------|-------------------|
| pDstVecX | Вектор решения x. |
|----------|-------------------|

Возвращает

void

Restrictions:

Выбор ведущего элемента не производится - возможно деление на нуль.

Заметки

Функция в данный момент не оптимизирована.

## 6.106 MTR\_fpResolve\_PivotGauss

Решение системы линейных уравнений методом Гаусса с выбором ведущего элемента.

Функции

- void MTR\_fpResolve\_PivotGauss (double \*pSrcMtrAB, double \*pDstVecX, int nSize)

### 6.106.1 Подробное описание

Решение системы линейных уравнений методом Гаусса с выбором ведущего элемента.

$$A \cdot X = B$$

Аргументы

|           |                                                     |
|-----------|-----------------------------------------------------|
| pSrcMtrAB | Матрица уравнения A, дополненная справа вектором B. |
| nSize     | Высота матрицы.                                     |

Возвращаемые значения

|          |                   |
|----------|-------------------|
| pDstVecX | Вектор решения X. |
|----------|-------------------|

Возвращает

void

## 6.107 nmppmLUDecomp

LU-разложение матрицы

Функции

- void nmppmLUDecomp\_64f (const double \*A, double \*L, double \*U, int N)

### 6.107.1 Подробное описание

LU-разложение матрицы

Аргументы

|     |   |                                |
|-----|---|--------------------------------|
| in  | A | Матрица A                      |
| out | L | Левая диагональная матрица L   |
| out | U | верхняя диагональная матрица U |
| in  | N | Размерность матрицы            |

## 6.108 nmppmLUResolve

Решение системы линейных уравнений из LU-разложения  $L*y = b$ ;  $U*x = y$ .

### Функции

- `void nmppmLUResolve_64f (const double *L, const double *U, const double *b, double *x, double *y, int N)`

### 6.108.1 Подробное описание

Решение системы линейных уравнений из LU-разложения  $L*y = b$ ;  $U*x = y$ .

#### Аргументы

|     |   |                                |
|-----|---|--------------------------------|
| in  | L | Левая диагональная матрица L   |
| in  | U | верхняя диагональная матрица U |
| in  | b | вектор b                       |
| out | x | вектор x                       |
| out | y | вектор y                       |
| in  | N | размер                         |

More details

## 6.109 nmppmMul\_mm

Умножение матрицы на матрицу.

## Функции

- void nmppmMul\_mm\_2s64s (nm2s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_4s64s (nm4s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_8s8s (nm8s \*pSrcMtr1, int nHeight1, int nWidth1, nm8s \*pSrcMtr2, nm8s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_8s16s (nm8s \*pSrcMtr1, int nHeight1, int nWidth1, nm16s \*pSrcMtr2, nm16s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_8s32s (nm8s \*pSrcMtr1, int nHeight1, int nWidth1, nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_8s64s (nm8s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_16s16s (nm16s \*pSrcMtr1, int nHeight1, int nWidth1, nm16s \*pSrcMtr2, nm16s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_16s32s (nm16s \*pSrcMtr1, int nHeight1, int nWidth1, nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_16s64s (nm16s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_32s32s (nm32s \*pSrcMtr1, int nHeight1, int nWidth1, nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_32s64s (nm32s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_64s64s (nm64s \*pSrcMtr1, int nHeight1, int nWidth1, nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_8s8s (const nm8s \*pSrcMtr1, int nHeight1, int nWidth1, const nm8s \*pSrcMtr2, nm8s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_8s16s (const nm8s \*pSrcMtr1, int nHeight1, int nWidth1, const nm16s \*pSrcMtr2, nm16s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_8s32s (const nm8s \*pSrcMtr1, int nHeight1, int nWidth1, const nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_8s64s (const nm8s \*pSrcMtr1, int nHeight1, int nWidth1, const nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_16s16s (const nm16s \*pSrcMtr1, int nHeight1, int nWidth1, const nm16s \*pSrcMtr2, nm16s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_16s32s (const nm16s \*pSrcMtr1, int nHeight1, int nWidth1, const nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_16s64s (const nm16s \*pSrcMtr1, int nHeight1, int nWidth1, const nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_32s32s (const nm32s \*pSrcMtr1, int nHeight1, int nWidth1, const nm32s \*pSrcMtr2, nm32s \*pDstMtr, int nWidth2)
- void nmppmMul\_mm\_colmajor\_32s64s (const nm32s \*pSrcMtr1, int nHeight1, int nWidth1, const nm64s \*pSrcMtr2, nm64s \*pDstMtr, int nWidth2)

## 6.109.1 Подробное описание

Умножение матрицы на матрицу.



## Аргументы

|          |                                   |
|----------|-----------------------------------|
| pSrcMtr1 | Исходная матрица.                 |
| pSrcMtr2 | Матрица-множитель.                |
| nHeight1 | Число строк исходной матрицы.     |
| nWidth1  | Число столбцов исходной матрицы.  |
| nWidth2  | Число столбцов матрицы множителя. |

## Возвращаемые значения

|         |                         |
|---------|-------------------------|
| pDstMtr | Результирующая матрица. |
|---------|-------------------------|

## Возвращает

void

## 6.109.2 Функции

## 6.109.2.1 nmppmMul\_mm\_16s32s()

```
void nmppmMul_mm_16s32s (
    nm16s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm32s * pSrcMtr2,
    nm32s * pDstMtr,
    int nWidth2 )
```

## Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 86246 | 0.16       |
| im1      | im1      | im1     | 64       | 64      | 128     | 86246 | 0.16       |
| im1      | im1      | im2     | 64       | 64      | 128     | 84230 | 0.16       |
| im1      | im1      | im2     | 64       | 64      | 128     | 84230 | 0.16       |
| im1      | im1      | im3     | 64       | 64      | 128     | 84230 | 0.16       |
| im1      | im1      | im3     | 64       | 64      | 128     | 84230 | 0.16       |
| im1      | im2      | im1     | 64       | 64      | 128     | 77766 | 0.14       |
| im1      | im2      | im1     | 64       | 64      | 128     | 77766 | 0.14       |
| im1      | im2      | im2     | 64       | 64      | 128     | 75814 | 0.14       |
| im1      | im2      | im2     | 64       | 64      | 128     | 75814 | 0.14       |
| im1      | im2      | im3     | 64       | 64      | 128     | 75878 | 0.14       |
| im1      | im2      | im3     | 64       | 64      | 128     | 75878 | 0.14       |
| im1      | im3      | im1     | 64       | 64      | 128     | 77766 | 0.14       |
| im1      | im3      | im1     | 64       | 64      | 128     | 77766 | 0.14       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im3      | im2     | 64       | 64      | 128     | 75878 | 0.14       |
| im1      | im3      | im2     | 64       | 64      | 128     | 75878 | 0.14       |
| im1      | im3      | im3     | 64       | 64      | 128     | 75814 | 0.14       |
| im1      | im3      | im3     | 64       | 64      | 128     | 75814 | 0.14       |
| im2      | im1      | im1     | 64       | 64      | 128     | 68774 | 0.13       |
| im2      | im1      | im1     | 64       | 64      | 128     | 68774 | 0.13       |
| im2      | im1      | im2     | 64       | 64      | 128     | 67750 | 0.12       |
| im2      | im1      | im2     | 64       | 64      | 128     | 67750 | 0.12       |
| im2      | im1      | im3     | 64       | 64      | 128     | 66790 | 0.12       |
| im2      | im1      | im3     | 64       | 64      | 128     | 66790 | 0.12       |
| im2      | im2      | im1     | 64       | 64      | 128     | 76742 | 0.14       |
| im2      | im2      | im1     | 64       | 64      | 128     | 76742 | 0.14       |
| im2      | im2      | im2     | 64       | 64      | 128     | 76376 | 0.14       |
| im2      | im2      | im2     | 64       | 64      | 128     | 76376 | 0.14       |
| im2      | im2      | im3     | 64       | 64      | 128     | 74790 | 0.14       |
| im2      | im2      | im3     | 64       | 64      | 128     | 74790 | 0.14       |
| im2      | im3      | im1     | 64       | 64      | 128     | 68678 | 0.13       |
| im2      | im3      | im1     | 64       | 64      | 128     | 68678 | 0.13       |
| im2      | im3      | im2     | 64       | 64      | 128     | 67686 | 0.12       |
| im2      | im3      | im2     | 64       | 64      | 128     | 67686 | 0.12       |
| im2      | im3      | im3     | 64       | 64      | 128     | 66946 | 0.12       |
| im2      | im3      | im3     | 64       | 64      | 128     | 66946 | 0.12       |
| im3      | im1      | im1     | 64       | 64      | 128     | 68774 | 0.13       |
| im3      | im1      | im1     | 64       | 64      | 128     | 68774 | 0.13       |
| im3      | im1      | im2     | 64       | 64      | 128     | 66790 | 0.12       |
| im3      | im1      | im2     | 64       | 64      | 128     | 66790 | 0.12       |
| im3      | im1      | im3     | 64       | 64      | 128     | 67750 | 0.12       |
| im3      | im1      | im3     | 64       | 64      | 128     | 67750 | 0.12       |
| im3      | im2      | im1     | 64       | 64      | 128     | 68678 | 0.13       |
| im3      | im2      | im1     | 64       | 64      | 128     | 68678 | 0.13       |
| im3      | im2      | im2     | 64       | 64      | 128     | 66946 | 0.12       |
| im3      | im2      | im2     | 64       | 64      | 128     | 66946 | 0.12       |
| im3      | im2      | im3     | 64       | 64      | 128     | 67686 | 0.12       |
| im3      | im2      | im3     | 64       | 64      | 128     | 67686 | 0.12       |
| im3      | im3      | im1     | 64       | 64      | 128     | 76742 | 0.14       |
| im3      | im3      | im1     | 64       | 64      | 128     | 76742 | 0.14       |
| im3      | im3      | im2     | 64       | 64      | 128     | 74790 | 0.14       |
| im3      | im3      | im2     | 64       | 64      | 128     | 74790 | 0.14       |
| im3      | im3      | im3     | 64       | 64      | 128     | 76376 | 0.14       |
| im3      | im3      | im3     | 64       | 64      | 128     | 76376 | 0.14       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im2      | im1      | im3     | 64       | 64      | 128     | 66790 | 0.12       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 86246 | 0.16       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 9481  | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 2       | 9481  | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 4       | 18860 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 4       | 18860 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 8       | 37618 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 8       | 37618 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 2       | 18713 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 2       | 18713 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 4       | 37324 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 4       | 37324 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 8       | 74546 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 8       | 74546 | 0.14       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 9481  | 0.14       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 74546 | 0.14       |

### 6.109.2.2 nmppmMul\_mm\_16s64s()

```
void nmppmMul_mm_16s64s (
    nm16s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm64s * pSrcMtr2,
    nm64s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 164805 | 0.31       |
| im1      | im1      | im1     | 64       | 64      | 128     | 164805 | 0.31       |
| im1      | im1      | im2     | 64       | 64      | 128     | 160773 | 0.30       |
| im1      | im1      | im2     | 64       | 64      | 128     | 160773 | 0.30       |
| im1      | im1      | im3     | 64       | 64      | 128     | 160773 | 0.30       |
| im1      | im1      | im3     | 64       | 64      | 128     | 160773 | 0.30       |
| im1      | im2      | im1     | 64       | 64      | 128     | 155815 | 0.29       |
| im1      | im2      | im1     | 64       | 64      | 128     | 155815 | 0.29       |
| im1      | im2      | im2     | 64       | 64      | 128     | 151525 | 0.28       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im2      | im2     | 64       | 64      | 128     | 151525 | 0.28       |
| im1      | im2      | im3     | 64       | 64      | 128     | 151653 | 0.28       |
| im1      | im2      | im3     | 64       | 64      | 128     | 151653 | 0.28       |
| im1      | im3      | im1     | 64       | 64      | 128     | 155815 | 0.29       |
| im1      | im3      | im1     | 64       | 64      | 128     | 155815 | 0.29       |
| im1      | im3      | im2     | 64       | 64      | 128     | 151653 | 0.28       |
| im1      | im3      | im2     | 64       | 64      | 128     | 151653 | 0.28       |
| im1      | im3      | im3     | 64       | 64      | 128     | 151525 | 0.28       |
| im1      | im3      | im3     | 64       | 64      | 128     | 151525 | 0.28       |
| im2      | im1      | im1     | 64       | 64      | 128     | 145029 | 0.27       |
| im2      | im1      | im1     | 64       | 64      | 128     | 145029 | 0.27       |
| im2      | im1      | im2     | 64       | 64      | 128     | 143168 | 0.27       |
| im2      | im1      | im2     | 64       | 64      | 128     | 143168 | 0.27       |
| im2      | im1      | im3     | 64       | 64      | 128     | 140933 | 0.26       |
| im2      | im1      | im3     | 64       | 64      | 128     | 140933 | 0.26       |
| im2      | im2      | im1     | 64       | 64      | 128     | 153381 | 0.29       |
| im2      | im2      | im1     | 64       | 64      | 128     | 153381 | 0.29       |
| im2      | im2      | im2     | 64       | 64      | 128     | 152663 | 0.29       |
| im2      | im2      | im2     | 64       | 64      | 128     | 152663 | 0.29       |
| im2      | im2      | im3     | 64       | 64      | 128     | 149477 | 0.28       |
| im2      | im2      | im3     | 64       | 64      | 128     | 149477 | 0.28       |
| im2      | im3      | im1     | 64       | 64      | 128     | 137253 | 0.26       |
| im2      | im3      | im1     | 64       | 64      | 128     | 137253 | 0.26       |
| im2      | im3      | im2     | 64       | 64      | 128     | 135584 | 0.25       |
| im2      | im3      | im2     | 64       | 64      | 128     | 135584 | 0.25       |
| im2      | im3      | im3     | 64       | 64      | 128     | 133793 | 0.25       |
| im2      | im3      | im3     | 64       | 64      | 128     | 133793 | 0.25       |
| im3      | im1      | im1     | 64       | 64      | 128     | 137445 | 0.26       |
| im3      | im1      | im1     | 64       | 64      | 128     | 137445 | 0.26       |
| im3      | im1      | im2     | 64       | 64      | 128     | 133477 | 0.25       |
| im3      | im1      | im2     | 64       | 64      | 128     | 133477 | 0.25       |
| im3      | im1      | im3     | 64       | 64      | 128     | 135710 | 0.25       |
| im3      | im1      | im3     | 64       | 64      | 128     | 135710 | 0.25       |
| im3      | im2      | im1     | 64       | 64      | 128     | 137253 | 0.26       |
| im3      | im2      | im1     | 64       | 64      | 128     | 137253 | 0.26       |
| im3      | im2      | im2     | 64       | 64      | 128     | 133793 | 0.25       |
| im3      | im2      | im2     | 64       | 64      | 128     | 133793 | 0.25       |
| im3      | im2      | im3     | 64       | 64      | 128     | 135584 | 0.25       |
| im3      | im2      | im3     | 64       | 64      | 128     | 135584 | 0.25       |
| im3      | im3      | im1     | 64       | 64      | 128     | 153381 | 0.29       |
| im3      | im3      | im1     | 64       | 64      | 128     | 153381 | 0.29       |
| im3      | im3      | im2     | 64       | 64      | 128     | 149477 | 0.28       |
| im3      | im3      | im2     | 64       | 64      | 128     | 149477 | 0.28       |
| im3      | im3      | im3     | 64       | 64      | 128     | 152663 | 0.29       |
| im3      | im3      | im3     | 64       | 64      | 128     | 152663 | 0.29       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im3      | im1      | im2     | 64       | 64      | 128     | 133477 | 0.25       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 164805 | 0.31       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 18859  | 0.28       |
| im1      | im2      | im3     | 256     | 128      | 2       | 18859  | 0.28       |
| im1      | im2      | im3     | 256     | 128      | 4       | 37617  | 0.28       |
| im1      | im2      | im3     | 256     | 128      | 4       | 37617  | 0.28       |
| im1      | im2      | im3     | 256     | 128      | 8       | 75133  | 0.28       |
| im1      | im2      | im3     | 256     | 128      | 8       | 75133  | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 2       | 37323  | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 2       | 37323  | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 4       | 74545  | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 4       | 74545  | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 8       | 148989 | 0.28       |
| im1      | im2      | im3     | 256     | 256      | 8       | 148989 | 0.28       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 18859 | 0.28       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 148989 | 0.28       |

### 6.109.2.3 nmppmMul\_mm\_2s64s()

```
void nmppmMul_mm_2s64s (
    nm2s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm64s * pSrcMtr2,
    nm64s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 37956 | 0.07       |
| im1      | im1      | im1     | 64       | 64      | 128     | 37956 | 0.07       |
| im1      | im1      | im2     | 64       | 64      | 128     | 33797 | 0.06       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im2     | 64       | 64      | 128     | 33797 | 0.06       |
| im1      | im1      | im3     | 64       | 64      | 128     | 33797 | 0.06       |
| im1      | im1      | im3     | 64       | 64      | 128     | 33797 | 0.06       |
| im1      | im2      | im1     | 64       | 64      | 128     | 27492 | 0.05       |
| im1      | im2      | im1     | 64       | 64      | 128     | 27492 | 0.05       |
| im1      | im2      | im2     | 64       | 64      | 128     | 26963 | 0.05       |
| im1      | im2      | im2     | 64       | 64      | 128     | 26963 | 0.05       |
| im1      | im2      | im3     | 64       | 64      | 128     | 24484 | 0.04       |
| im1      | im2      | im3     | 64       | 64      | 128     | 24484 | 0.04       |
| im1      | im3      | im1     | 64       | 64      | 128     | 27492 | 0.05       |
| im1      | im3      | im1     | 64       | 64      | 128     | 27492 | 0.05       |
| im1      | im3      | im2     | 64       | 64      | 128     | 24484 | 0.04       |
| im1      | im3      | im2     | 64       | 64      | 128     | 24484 | 0.04       |
| im1      | im3      | im3     | 64       | 64      | 128     | 26963 | 0.05       |
| im1      | im3      | im3     | 64       | 64      | 128     | 26963 | 0.05       |
| im2      | im1      | im1     | 64       | 64      | 128     | 39524 | 0.07       |
| im2      | im1      | im1     | 64       | 64      | 128     | 39524 | 0.07       |
| im2      | im1      | im2     | 64       | 64      | 128     | 36532 | 0.06       |
| im2      | im1      | im2     | 64       | 64      | 128     | 36532 | 0.06       |
| im2      | im1      | im3     | 64       | 64      | 128     | 35556 | 0.06       |
| im2      | im1      | im3     | 64       | 64      | 128     | 35556 | 0.06       |
| im2      | im2      | im1     | 64       | 64      | 128     | 41828 | 0.07       |
| im2      | im2      | im1     | 64       | 64      | 128     | 41828 | 0.07       |
| im2      | im2      | im2     | 64       | 64      | 128     | 43311 | 0.08       |
| im2      | im2      | im2     | 64       | 64      | 128     | 43311 | 0.08       |
| im2      | im2      | im3     | 64       | 64      | 128     | 38995 | 0.07       |
| im2      | im2      | im3     | 64       | 64      | 128     | 38995 | 0.07       |
| im2      | im3      | im1     | 64       | 64      | 128     | 25700 | 0.04       |
| im2      | im3      | im1     | 64       | 64      | 128     | 25700 | 0.04       |
| im2      | im3      | im2     | 64       | 64      | 128     | 23588 | 0.04       |
| im2      | im3      | im2     | 64       | 64      | 128     | 23588 | 0.04       |
| im2      | im3      | im3     | 64       | 64      | 128     | 26959 | 0.05       |
| im2      | im3      | im3     | 64       | 64      | 128     | 26959 | 0.05       |
| im3      | im1      | im1     | 64       | 64      | 128     | 31396 | 0.05       |
| im3      | im1      | im1     | 64       | 64      | 128     | 31396 | 0.05       |
| im3      | im1      | im2     | 64       | 64      | 128     | 27300 | 0.05       |
| im3      | im1      | im2     | 64       | 64      | 128     | 27300 | 0.05       |
| im3      | im1      | im3     | 64       | 64      | 128     | 27300 | 0.05       |
| im3      | im1      | im3     | 64       | 64      | 128     | 27300 | 0.05       |
| im3      | im2      | im1     | 64       | 64      | 128     | 25700 | 0.04       |
| im3      | im2      | im1     | 64       | 64      | 128     | 25700 | 0.04       |
| im3      | im2      | im2     | 64       | 64      | 128     | 26959 | 0.05       |
| im3      | im2      | im2     | 64       | 64      | 128     | 26959 | 0.05       |
| im3      | im2      | im3     | 64       | 64      | 128     | 23588 | 0.04       |
| im3      | im2      | im3     | 64       | 64      | 128     | 23588 | 0.04       |
| im3      | im3      | im1     | 64       | 64      | 128     | 41828 | 0.07       |
| im3      | im3      | im1     | 64       | 64      | 128     | 41828 | 0.07       |
| im3      | im3      | im2     | 64       | 64      | 128     | 38995 | 0.07       |
| im3      | im3      | im2     | 64       | 64      | 128     | 38995 | 0.07       |
| im3      | im3      | im3     | 64       | 64      | 128     | 43311 | 0.08       |
| im3      | im3      | im3     | 64       | 64      | 128     | 43311 | 0.08       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im2      | im3      | im2     | 64       | 64      | 128     | 23588 | 0.04       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im2      | im2      | im2     | 64       | 64      | 128     | 43311 | 0.08       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 2708  | 0.04       |
| im1      | im2      | im3     | 256     | 128      | 2       | 2708  | 0.04       |
| im1      | im2      | im3     | 256     | 128      | 4       | 5316  | 0.04       |
| im1      | im2      | im3     | 256     | 128      | 4       | 5316  | 0.04       |
| im1      | im2      | im3     | 256     | 128      | 8       | 10532 | 0.04       |
| im1      | im2      | im3     | 256     | 128      | 8       | 10532 | 0.04       |
| im1      | im2      | im3     | 256     | 256      | 2       | 5228  | 0.03       |
| im1      | im2      | im3     | 256     | 256      | 2       | 5228  | 0.03       |
| im1      | im2      | im3     | 256     | 256      | 4       | 10356 | 0.03       |
| im1      | im2      | im3     | 256     | 256      | 4       | 10356 | 0.03       |
| im1      | im2      | im3     | 256     | 256      | 8       | 20612 | 0.03       |
| im1      | im2      | im3     | 256     | 256      | 8       | 20612 | 0.03       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 2708  | 0.04       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 20612 | 0.03       |

#### 6.109.2.4 nmppmMul\_mm\_32s32s()

```
void nmppmMul_mm_32s32s (
    nm32s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm32s * pSrcMtr2,
    nm32s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 159973 | 0.30       |
| im1      | im1      | im1     | 64       | 64      | 128     | 159973 | 0.30       |
| im1      | im1      | im2     | 64       | 64      | 128     | 157701 | 0.30       |
| im1      | im1      | im2     | 64       | 64      | 128     | 157701 | 0.30       |
| im1      | im1      | im3     | 64       | 64      | 128     | 157701 | 0.30       |
| im1      | im1      | im3     | 64       | 64      | 128     | 157701 | 0.30       |
| im1      | im2      | im1     | 64       | 64      | 128     | 151493 | 0.28       |
| im1      | im2      | im1     | 64       | 64      | 128     | 151493 | 0.28       |
| im1      | im2      | im2     | 64       | 64      | 128     | 149541 | 0.28       |
| im1      | im2      | im2     | 64       | 64      | 128     | 149541 | 0.28       |
| im1      | im2      | im3     | 64       | 64      | 128     | 149605 | 0.28       |
| im1      | im2      | im3     | 64       | 64      | 128     | 149605 | 0.28       |
| im1      | im3      | im1     | 64       | 64      | 128     | 151493 | 0.28       |
| im1      | im3      | im1     | 64       | 64      | 128     | 151493 | 0.28       |
| im1      | im3      | im2     | 64       | 64      | 128     | 149605 | 0.28       |
| im1      | im3      | im2     | 64       | 64      | 128     | 149605 | 0.28       |
| im1      | im3      | im3     | 64       | 64      | 128     | 149541 | 0.28       |
| im1      | im3      | im3     | 64       | 64      | 128     | 149541 | 0.28       |
| im2      | im1      | im1     | 64       | 64      | 128     | 134181 | 0.25       |
| im2      | im1      | im1     | 64       | 64      | 128     | 134181 | 0.25       |
| im2      | im1      | im2     | 64       | 64      | 128     | 133376 | 0.25       |
| im2      | im1      | im2     | 64       | 64      | 128     | 133376 | 0.25       |
| im2      | im1      | im3     | 64       | 64      | 128     | 132261 | 0.25       |
| im2      | im1      | im3     | 64       | 64      | 128     | 132261 | 0.25       |
| im2      | im2      | im1     | 64       | 64      | 128     | 142277 | 0.27       |
| im2      | im2      | im1     | 64       | 64      | 128     | 142277 | 0.27       |
| im2      | im2      | im2     | 64       | 64      | 128     | 141849 | 0.27       |
| im2      | im2      | im2     | 64       | 64      | 128     | 141849 | 0.27       |
| im2      | im2      | im3     | 64       | 64      | 128     | 140325 | 0.26       |
| im2      | im2      | im3     | 64       | 64      | 128     | 140325 | 0.26       |
| im2      | im3      | im1     | 64       | 64      | 128     | 134213 | 0.25       |
| im2      | im3      | im1     | 64       | 64      | 128     | 134213 | 0.25       |
| im2      | im3      | im2     | 64       | 64      | 128     | 133376 | 0.25       |
| im2      | im3      | im2     | 64       | 64      | 128     | 133376 | 0.25       |
| im2      | im3      | im3     | 64       | 64      | 128     | 132355 | 0.25       |
| im2      | im3      | im3     | 64       | 64      | 128     | 132355 | 0.25       |
| im3      | im1      | im1     | 64       | 64      | 128     | 134181 | 0.25       |
| im3      | im1      | im1     | 64       | 64      | 128     | 134181 | 0.25       |
| im3      | im1      | im2     | 64       | 64      | 128     | 132261 | 0.25       |
| im3      | im1      | im2     | 64       | 64      | 128     | 132261 | 0.25       |
| im3      | im1      | im3     | 64       | 64      | 128     | 133376 | 0.25       |
| im3      | im1      | im3     | 64       | 64      | 128     | 133376 | 0.25       |
| im3      | im2      | im1     | 64       | 64      | 128     | 134213 | 0.25       |
| im3      | im2      | im1     | 64       | 64      | 128     | 134213 | 0.25       |
| im3      | im2      | im2     | 64       | 64      | 128     | 132355 | 0.25       |
| im3      | im2      | im2     | 64       | 64      | 128     | 132355 | 0.25       |
| im3      | im2      | im3     | 64       | 64      | 128     | 133376 | 0.25       |
| im3      | im2      | im3     | 64       | 64      | 128     | 133376 | 0.25       |
| im3      | im3      | im1     | 64       | 64      | 128     | 142277 | 0.27       |
| im3      | im3      | im1     | 64       | 64      | 128     | 142277 | 0.27       |
| im3      | im3      | im2     | 64       | 64      | 128     | 140325 | 0.26       |
| im3      | im3      | im2     | 64       | 64      | 128     | 140325 | 0.26       |



| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im3      | im3      | im3     | 64       | 64      | 128     | 141849 | 0.27       |
| im3      | im3      | im3     | 64       | 64      | 128     | 141849 | 0.27       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im2      | im1      | im3     | 64       | 64      | 128     | 132261 | 0.25       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 159973 | 0.30       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 18296  | 0.27       |
| im1      | im2      | im3     | 256     | 128      | 2       | 18296  | 0.27       |
| im1      | im2      | im3     | 256     | 128      | 4       | 36491  | 0.27       |
| im1      | im2      | im3     | 256     | 128      | 4       | 36491  | 0.27       |
| im1      | im2      | im3     | 256     | 128      | 8       | 72881  | 0.27       |
| im1      | im2      | im3     | 256     | 128      | 8       | 72881  | 0.27       |
| im1      | im2      | im3     | 256     | 256      | 2       | 35030  | 0.26       |
| im1      | im2      | im3     | 256     | 256      | 2       | 35030  | 0.26       |
| im1      | im2      | im3     | 256     | 256      | 4       | 69959  | 0.26       |
| im1      | im2      | im3     | 256     | 256      | 4       | 69959  | 0.26       |
| im1      | im2      | im3     | 256     | 256      | 8       | 139841 | 0.26       |
| im1      | im2      | im3     | 256     | 256      | 8       | 139841 | 0.26       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 18296 | 0.27       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 139841 | 0.26       |

### 6.109.2.5 nmppmMul\_mm\_32s64s()

```
void nmppmMul_mm_32s64s (
    nm32s * pSrcMtr1,
```

```

int nHeight1,
int nWidth1,
nm64s * pSrcMtr2,
nm64s * pDstMtr,
int nWidth2 )

```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 312517 | 0.59       |
| im1      | im1      | im1     | 64       | 64      | 128     | 312517 | 0.59       |
| im1      | im1      | im2     | 64       | 64      | 128     | 307973 | 0.58       |
| im1      | im1      | im2     | 64       | 64      | 128     | 307973 | 0.58       |
| im1      | im1      | im3     | 64       | 64      | 128     | 307973 | 0.58       |
| im1      | im1      | im3     | 64       | 64      | 128     | 307973 | 0.58       |
| im1      | im2      | im1     | 64       | 64      | 128     | 302885 | 0.57       |
| im1      | im2      | im1     | 64       | 64      | 128     | 302885 | 0.57       |
| im1      | im2      | im2     | 64       | 64      | 128     | 298981 | 0.57       |
| im1      | im2      | im2     | 64       | 64      | 128     | 298981 | 0.57       |
| im1      | im2      | im3     | 64       | 64      | 128     | 299109 | 0.57       |
| im1      | im2      | im3     | 64       | 64      | 128     | 299109 | 0.57       |
| im1      | im3      | im1     | 64       | 64      | 128     | 302885 | 0.57       |
| im1      | im3      | im1     | 64       | 64      | 128     | 302885 | 0.57       |
| im1      | im3      | im2     | 64       | 64      | 128     | 299109 | 0.57       |
| im1      | im3      | im2     | 64       | 64      | 128     | 299109 | 0.57       |
| im1      | im3      | im3     | 64       | 64      | 128     | 298981 | 0.57       |
| im1      | im3      | im3     | 64       | 64      | 128     | 298981 | 0.57       |
| im2      | im1      | im1     | 64       | 64      | 128     | 275589 | 0.52       |
| im2      | im1      | im1     | 64       | 64      | 128     | 275589 | 0.52       |
| im2      | im1      | im2     | 64       | 64      | 128     | 273667 | 0.52       |
| im2      | im1      | im2     | 64       | 64      | 128     | 273667 | 0.52       |
| im2      | im1      | im3     | 64       | 64      | 128     | 271621 | 0.51       |
| im2      | im1      | im3     | 64       | 64      | 128     | 271621 | 0.51       |
| im2      | im2      | im1     | 64       | 64      | 128     | 284453 | 0.54       |
| im2      | im2      | im1     | 64       | 64      | 128     | 284453 | 0.54       |
| im2      | im2      | im2     | 64       | 64      | 128     | 283357 | 0.54       |
| im2      | im2      | im2     | 64       | 64      | 128     | 283357 | 0.54       |
| im2      | im2      | im3     | 64       | 64      | 128     | 280549 | 0.53       |
| im2      | im2      | im3     | 64       | 64      | 128     | 280549 | 0.53       |
| im2      | im3      | im1     | 64       | 64      | 128     | 268325 | 0.51       |
| im2      | im3      | im1     | 64       | 64      | 128     | 268325 | 0.51       |
| im2      | im3      | im2     | 64       | 64      | 128     | 266341 | 0.50       |
| im2      | im3      | im2     | 64       | 64      | 128     | 266341 | 0.50       |
| im2      | im3      | im3     | 64       | 64      | 128     | 264611 | 0.50       |
| im2      | im3      | im3     | 64       | 64      | 128     | 264611 | 0.50       |
| im3      | im1      | im1     | 64       | 64      | 128     | 268261 | 0.51       |
| im3      | im1      | im1     | 64       | 64      | 128     | 268261 | 0.51       |
| im3      | im1      | im2     | 64       | 64      | 128     | 264421 | 0.50       |
| im3      | im1      | im2     | 64       | 64      | 128     | 264421 | 0.50       |
| im3      | im1      | im3     | 64       | 64      | 128     | 266341 | 0.50       |
| im3      | im1      | im3     | 64       | 64      | 128     | 266341 | 0.50       |
| im3      | im2      | im1     | 64       | 64      | 128     | 268325 | 0.51       |
| im3      | im2      | im1     | 64       | 64      | 128     | 268325 | 0.51       |
| im3      | im2      | im2     | 64       | 64      | 128     | 264611 | 0.50       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im3      | im2      | im2     | 64       | 64      | 128     | 264611 | 0.50       |
| im3      | im2      | im3     | 64       | 64      | 128     | 266341 | 0.50       |
| im3      | im2      | im3     | 64       | 64      | 128     | 266341 | 0.50       |
| im3      | im3      | im1     | 64       | 64      | 128     | 284453 | 0.54       |
| im3      | im3      | im1     | 64       | 64      | 128     | 284453 | 0.54       |
| im3      | im3      | im2     | 64       | 64      | 128     | 280549 | 0.53       |
| im3      | im3      | im2     | 64       | 64      | 128     | 280549 | 0.53       |
| im3      | im3      | im3     | 64       | 64      | 128     | 283357 | 0.54       |
| im3      | im3      | im3     | 64       | 64      | 128     | 283357 | 0.54       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im3      | im1      | im2     | 64       | 64      | 128     | 264421 | 0.50       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 312517 | 0.59       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 36491  | 0.55       |
| im1      | im2      | im3     | 256     | 128      | 2       | 36491  | 0.55       |
| im1      | im2      | im3     | 256     | 128      | 4       | 72881  | 0.55       |
| im1      | im2      | im3     | 256     | 128      | 4       | 72881  | 0.55       |
| im1      | im2      | im3     | 256     | 128      | 8       | 145661 | 0.55       |
| im1      | im2      | im3     | 256     | 128      | 8       | 145661 | 0.55       |
| im1      | im2      | im3     | 256     | 256      | 2       | 69959  | 0.53       |
| im1      | im2      | im3     | 256     | 256      | 2       | 69959  | 0.53       |
| im1      | im2      | im3     | 256     | 256      | 4       | 139841 | 0.53       |
| im1      | im2      | im3     | 256     | 256      | 4       | 139841 | 0.53       |
| im1      | im2      | im3     | 256     | 256      | 8       | 279581 | 0.53       |
| im1      | im2      | im3     | 256     | 256      | 8       | 279581 | 0.53       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 36491 | 0.55       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 279581 | 0.53       |

## 6.109.2.6 nmppmMul\_mm\_4s64s()

```
void nmppmMul_mm_4s64s (
    nm4s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm64s * pSrcMtr2,
    nm64s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 55364 | 0.10       |
| im1      | im1      | im1     | 64       | 64      | 128     | 55364 | 0.10       |
| im1      | im1      | im2     | 64       | 64      | 128     | 51333 | 0.09       |
| im1      | im1      | im2     | 64       | 64      | 128     | 51333 | 0.09       |
| im1      | im1      | im3     | 64       | 64      | 128     | 51333 | 0.09       |
| im1      | im1      | im3     | 64       | 64      | 128     | 51333 | 0.09       |
| im1      | im2      | im1     | 64       | 64      | 128     | 45090 | 0.08       |
| im1      | im2      | im1     | 64       | 64      | 128     | 45090 | 0.08       |
| im1      | im2      | im2     | 64       | 64      | 128     | 42082 | 0.08       |
| im1      | im2      | im2     | 64       | 64      | 128     | 42082 | 0.08       |
| im1      | im2      | im3     | 64       | 64      | 128     | 41186 | 0.07       |
| im1      | im2      | im3     | 64       | 64      | 128     | 41186 | 0.07       |
| im1      | im3      | im1     | 64       | 64      | 128     | 45090 | 0.08       |
| im1      | im3      | im1     | 64       | 64      | 128     | 45090 | 0.08       |
| im1      | im3      | im2     | 64       | 64      | 128     | 41186 | 0.07       |
| im1      | im3      | im2     | 64       | 64      | 128     | 41186 | 0.07       |
| im1      | im3      | im3     | 64       | 64      | 128     | 42082 | 0.08       |
| im1      | im3      | im3     | 64       | 64      | 128     | 42082 | 0.08       |
| im2      | im1      | im1     | 64       | 64      | 128     | 50436 | 0.09       |
| im2      | im1      | im1     | 64       | 64      | 128     | 50436 | 0.09       |
| im2      | im1      | im2     | 64       | 64      | 128     | 47684 | 0.09       |
| im2      | im1      | im2     | 64       | 64      | 128     | 47684 | 0.09       |
| im2      | im1      | im3     | 64       | 64      | 128     | 45764 | 0.08       |
| im2      | im1      | im3     | 64       | 64      | 128     | 45764 | 0.08       |
| im2      | im2      | im1     | 64       | 64      | 128     | 56164 | 0.10       |
| im2      | im2      | im1     | 64       | 64      | 128     | 56164 | 0.10       |
| im2      | im2      | im2     | 64       | 64      | 128     | 56643 | 0.10       |
| im2      | im2      | im2     | 64       | 64      | 128     | 56643 | 0.10       |
| im2      | im2      | im3     | 64       | 64      | 128     | 52323 | 0.09       |
| im2      | im2      | im3     | 64       | 64      | 128     | 52323 | 0.09       |
| im2      | im3      | im1     | 64       | 64      | 128     | 38948 | 0.07       |
| im2      | im3      | im1     | 64       | 64      | 128     | 38948 | 0.07       |
| im2      | im3      | im2     | 64       | 64      | 128     | 37092 | 0.07       |
| im2      | im3      | im2     | 64       | 64      | 128     | 37092 | 0.07       |
| im2      | im3      | im3     | 64       | 64      | 128     | 37398 | 0.07       |
| im2      | im3      | im3     | 64       | 64      | 128     | 37398 | 0.07       |
| im3      | im1      | im1     | 64       | 64      | 128     | 41316 | 0.07       |
| im3      | im1      | im1     | 64       | 64      | 128     | 41316 | 0.07       |
| im3      | im1      | im2     | 64       | 64      | 128     | 36772 | 0.07       |
| im3      | im1      | im2     | 64       | 64      | 128     | 36772 | 0.07       |
| im3      | im1      | im3     | 64       | 64      | 128     | 38692 | 0.07       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im3      | im1      | im3     | 64       | 64      | 128     | 38692 | 0.07       |
| im3      | im2      | im1     | 64       | 64      | 128     | 38948 | 0.07       |
| im3      | im2      | im1     | 64       | 64      | 128     | 38948 | 0.07       |
| im3      | im2      | im2     | 64       | 64      | 128     | 37398 | 0.07       |
| im3      | im2      | im2     | 64       | 64      | 128     | 37398 | 0.07       |
| im3      | im2      | im3     | 64       | 64      | 128     | 37092 | 0.07       |
| im3      | im2      | im3     | 64       | 64      | 128     | 37092 | 0.07       |
| im3      | im3      | im1     | 64       | 64      | 128     | 56164 | 0.10       |
| im3      | im3      | im1     | 64       | 64      | 128     | 56164 | 0.10       |
| im3      | im3      | im2     | 64       | 64      | 128     | 52323 | 0.09       |
| im3      | im3      | im2     | 64       | 64      | 128     | 52323 | 0.09       |
| im3      | im3      | im3     | 64       | 64      | 128     | 56643 | 0.10       |
| im3      | im3      | im3     | 64       | 64      | 128     | 56643 | 0.10       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im3      | im1      | im2     | 64       | 64      | 128     | 36772 | 0.07       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im2      | im2      | im2     | 64       | 64      | 128     | 56643 | 0.10       |

Performance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 4854  | 0.07       |
| im1      | im2      | im3     | 256     | 128      | 2       | 4854  | 0.07       |
| im1      | im2      | im3     | 256     | 128      | 4       | 9610  | 0.07       |
| im1      | im2      | im3     | 256     | 128      | 4       | 9610  | 0.07       |
| im1      | im2      | im3     | 256     | 128      | 8       | 19122 | 0.07       |
| im1      | im2      | im3     | 256     | 128      | 8       | 19122 | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 2       | 9494  | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 2       | 9494  | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 4       | 18890 | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 4       | 18890 | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 8       | 37682 | 0.07       |
| im1      | im2      | im3     | 256     | 256      | 8       | 37682 | 0.07       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 4854  | 0.07       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 37682 | 0.07       |

## 6.109.2.7 nmppmMul\_mm\_64s64s()

```
void nmppmMul_mm_64s64s (
    nm64s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm64s * pSrcMtr2,
    nm64s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 590837 | 1.12       |
| im1      | im1      | im1     | 64       | 64      | 128     | 590837 | 1.12       |
| im1      | im1      | im2     | 64       | 64      | 128     | 586484 | 1.11       |
| im1      | im1      | im2     | 64       | 64      | 128     | 586484 | 1.11       |
| im1      | im1      | im3     | 64       | 64      | 128     | 584885 | 1.11       |
| im1      | im1      | im3     | 64       | 64      | 128     | 584885 | 1.11       |
| im1      | im2      | im1     | 64       | 64      | 128     | 613349 | 1.16       |
| im1      | im2      | im1     | 64       | 64      | 128     | 613349 | 1.16       |
| im1      | im2      | im2     | 64       | 64      | 128     | 609666 | 1.16       |
| im1      | im2      | im2     | 64       | 64      | 128     | 609666 | 1.16       |
| im1      | im2      | im3     | 64       | 64      | 128     | 607397 | 1.15       |
| im1      | im2      | im3     | 64       | 64      | 128     | 607397 | 1.15       |
| im1      | im3      | im1     | 64       | 64      | 128     | 565797 | 1.07       |
| im1      | im3      | im1     | 64       | 64      | 128     | 565797 | 1.07       |
| im1      | im3      | im2     | 64       | 64      | 128     | 561508 | 1.07       |
| im1      | im3      | im2     | 64       | 64      | 128     | 561508 | 1.07       |
| im1      | im3      | im3     | 64       | 64      | 128     | 560867 | 1.06       |
| im1      | im3      | im3     | 64       | 64      | 128     | 560867 | 1.06       |
| im2      | im1      | im1     | 64       | 64      | 128     | 537541 | 1.02       |
| im2      | im1      | im1     | 64       | 64      | 128     | 537541 | 1.02       |
| im2      | im1      | im2     | 64       | 64      | 128     | 535999 | 1.02       |
| im2      | im1      | im2     | 64       | 64      | 128     | 535999 | 1.02       |
| im2      | im1      | im3     | 64       | 64      | 128     | 533701 | 1.01       |
| im2      | im1      | im3     | 64       | 64      | 128     | 533701 | 1.01       |
| im2      | im2      | im1     | 64       | 64      | 128     | 546597 | 1.04       |
| im2      | im2      | im1     | 64       | 64      | 128     | 546597 | 1.04       |
| im2      | im2      | im2     | 64       | 64      | 128     | 545565 | 1.04       |
| im2      | im2      | im2     | 64       | 64      | 128     | 545565 | 1.04       |
| im2      | im2      | im3     | 64       | 64      | 128     | 542757 | 1.03       |
| im2      | im2      | im3     | 64       | 64      | 128     | 542757 | 1.03       |
| im2      | im3      | im1     | 64       | 64      | 128     | 530469 | 1.01       |
| im2      | im3      | im1     | 64       | 64      | 128     | 530469 | 1.01       |
| im2      | im3      | im2     | 64       | 64      | 128     | 528800 | 1.00       |
| im2      | im3      | im2     | 64       | 64      | 128     | 528800 | 1.00       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im2      | im3      | im3     | 64       | 64      | 128     | 526628 | 1.00       |
| im2      | im3      | im3     | 64       | 64      | 128     | 526628 | 1.00       |
| im3      | im1      | im1     | 64       | 64      | 128     | 530341 | 1.01       |
| im3      | im1      | im1     | 64       | 64      | 128     | 530341 | 1.01       |
| im3      | im1      | im2     | 64       | 64      | 128     | 526565 | 1.00       |
| im3      | im1      | im2     | 64       | 64      | 128     | 526565 | 1.00       |
| im3      | im1      | im3     | 64       | 64      | 128     | 528800 | 1.00       |
| im3      | im1      | im3     | 64       | 64      | 128     | 528800 | 1.00       |
| im3      | im2      | im1     | 64       | 64      | 128     | 530469 | 1.01       |
| im3      | im2      | im1     | 64       | 64      | 128     | 530469 | 1.01       |
| im3      | im2      | im2     | 64       | 64      | 128     | 526628 | 1.00       |
| im3      | im2      | im2     | 64       | 64      | 128     | 526628 | 1.00       |
| im3      | im2      | im3     | 64       | 64      | 128     | 528800 | 1.00       |
| im3      | im2      | im3     | 64       | 64      | 128     | 528800 | 1.00       |
| im3      | im3      | im1     | 64       | 64      | 128     | 546597 | 1.04       |
| im3      | im3      | im1     | 64       | 64      | 128     | 546597 | 1.04       |
| im3      | im3      | im2     | 64       | 64      | 128     | 542757 | 1.03       |
| im3      | im3      | im2     | 64       | 64      | 128     | 542757 | 1.03       |
| im3      | im3      | im3     | 64       | 64      | 128     | 545565 | 1.04       |
| im3      | im3      | im3     | 64       | 64      | 128     | 545565 | 1.04       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im3      | im1      | im2     | 64       | 64      | 128     | 526565 | 1.00       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks  | ticks/elem |
|----------|----------|---------|----------|---------|---------|--------|------------|
| im1      | im2      | im1     | 64       | 64      | 128     | 613349 | 1.16       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 69355  | 1.05       |
| im1      | im2      | im3     | 256     | 128      | 2       | 69355  | 1.05       |
| im1      | im2      | im3     | 256     | 128      | 4       | 138609 | 1.05       |
| im1      | im2      | im3     | 256     | 128      | 4       | 138609 | 1.05       |
| im1      | im2      | im3     | 256     | 128      | 8       | 277118 | 1.05       |
| im1      | im2      | im3     | 256     | 128      | 8       | 277118 | 1.05       |
| im1      | im2      | im3     | 256     | 256      | 2       | 221602 | 1.69       |
| im1      | im2      | im3     | 256     | 256      | 2       | 221602 | 1.69       |
| im1      | im2      | im3     | 256     | 256      | 4       | 443117 | 1.69       |
| im1      | im2      | im3     | 256     | 256      | 4       | 443117 | 1.69       |
| im1      | im2      | im3     | 256     | 256      | 8       | 886215 | 1.69       |
| im1      | im2      | im3     | 256     | 256      | 8       | 886215 | 1.69       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 69355 | 1.05       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks  | ticks/elem |
|----------|----------|---------|---------|----------|---------|--------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 886215 | 1.69       |

#### 6.109.2.8 nmppmMul\_mm\_8s64s()

```
void nmppmMul_mm_8s64s (
    nm8s * pSrcMtr1,
    int nHeight1,
    int nWidth1,
    nm64s * pSrcMtr2,
    nm64s * pDstMtr,
    int nWidth2 )
```

Perfomance table 0

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 91521 | 0.17       |
| im1      | im1      | im1     | 64       | 64      | 128     | 91521 | 0.17       |
| im1      | im1      | im2     | 64       | 64      | 128     | 87490 | 0.16       |
| im1      | im1      | im2     | 64       | 64      | 128     | 87490 | 0.16       |
| im1      | im1      | im3     | 64       | 64      | 128     | 87490 | 0.16       |
| im1      | im1      | im3     | 64       | 64      | 128     | 87490 | 0.16       |
| im1      | im2      | im1     | 64       | 64      | 128     | 81701 | 0.15       |
| im1      | im2      | im1     | 64       | 64      | 128     | 81701 | 0.15       |
| im1      | im2      | im2     | 64       | 64      | 128     | 78117 | 0.14       |
| im1      | im2      | im2     | 64       | 64      | 128     | 78117 | 0.14       |
| im1      | im2      | im3     | 64       | 64      | 128     | 77925 | 0.14       |
| im1      | im2      | im3     | 64       | 64      | 128     | 77925 | 0.14       |
| im1      | im3      | im1     | 64       | 64      | 128     | 81701 | 0.15       |
| im1      | im3      | im1     | 64       | 64      | 128     | 81701 | 0.15       |
| im1      | im3      | im2     | 64       | 64      | 128     | 77925 | 0.14       |
| im1      | im3      | im2     | 64       | 64      | 128     | 77925 | 0.14       |
| im1      | im3      | im3     | 64       | 64      | 128     | 78117 | 0.14       |
| im1      | im3      | im3     | 64       | 64      | 128     | 78117 | 0.14       |
| im2      | im1      | im1     | 64       | 64      | 128     | 80515 | 0.15       |
| im2      | im1      | im1     | 64       | 64      | 128     | 80515 | 0.15       |
| im2      | im1      | im2     | 64       | 64      | 128     | 78339 | 0.14       |
| im2      | im1      | im2     | 64       | 64      | 128     | 78339 | 0.14       |
| im2      | im1      | im3     | 64       | 64      | 128     | 76419 | 0.14       |
| im2      | im1      | im3     | 64       | 64      | 128     | 76419 | 0.14       |
| im2      | im2      | im1     | 64       | 64      | 128     | 88037 | 0.16       |
| im2      | im2      | im1     | 64       | 64      | 128     | 88037 | 0.16       |
| im2      | im2      | im2     | 64       | 64      | 128     | 87886 | 0.16       |
| im2      | im2      | im2     | 64       | 64      | 128     | 87886 | 0.16       |



| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im2      | im2      | im3     | 64       | 64      | 128     | 84133 | 0.16       |
| im2      | im2      | im3     | 64       | 64      | 128     | 84133 | 0.16       |
| im2      | im3      | im1     | 64       | 64      | 128     | 71589 | 0.13       |
| im2      | im3      | im1     | 64       | 64      | 128     | 71589 | 0.13       |
| im2      | im3      | im2     | 64       | 64      | 128     | 69733 | 0.13       |
| im2      | im3      | im2     | 64       | 64      | 128     | 69733 | 0.13       |
| im2      | im3      | im3     | 64       | 64      | 128     | 68765 | 0.13       |
| im2      | im3      | im3     | 64       | 64      | 128     | 68765 | 0.13       |
| im3      | im1      | im1     | 64       | 64      | 128     | 72419 | 0.13       |
| im3      | im1      | im1     | 64       | 64      | 128     | 72419 | 0.13       |
| im3      | im1      | im2     | 64       | 64      | 128     | 68451 | 0.13       |
| im3      | im1      | im2     | 64       | 64      | 128     | 68451 | 0.13       |
| im3      | im1      | im3     | 64       | 64      | 128     | 70371 | 0.13       |
| im3      | im1      | im3     | 64       | 64      | 128     | 70371 | 0.13       |
| im3      | im2      | im1     | 64       | 64      | 128     | 71589 | 0.13       |
| im3      | im2      | im1     | 64       | 64      | 128     | 71589 | 0.13       |
| im3      | im2      | im2     | 64       | 64      | 128     | 68765 | 0.13       |
| im3      | im2      | im2     | 64       | 64      | 128     | 68765 | 0.13       |
| im3      | im2      | im3     | 64       | 64      | 128     | 69733 | 0.13       |
| im3      | im2      | im3     | 64       | 64      | 128     | 69733 | 0.13       |
| im3      | im3      | im1     | 64       | 64      | 128     | 88037 | 0.16       |
| im3      | im3      | im1     | 64       | 64      | 128     | 88037 | 0.16       |
| im3      | im3      | im2     | 64       | 64      | 128     | 84133 | 0.16       |
| im3      | im3      | im2     | 64       | 64      | 128     | 84133 | 0.16       |
| im3      | im3      | im3     | 64       | 64      | 128     | 87886 | 0.16       |
| im3      | im3      | im3     | 64       | 64      | 128     | 87886 | 0.16       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im3      | im1      | im2     | 64       | 64      | 128     | 68451 | 0.13       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nHeight1 | nWidth2 | nWidth1 | ticks | ticks/elem |
|----------|----------|---------|----------|---------|---------|-------|------------|
| im1      | im1      | im1     | 64       | 64      | 128     | 91521 | 0.17       |

Perfomance table 1

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 9473  | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 2       | 9473  | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 4       | 18845 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 4       | 18845 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 8       | 37589 | 0.14       |
| im1      | im2      | im3     | 256     | 128      | 8       | 37589 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 2       | 19567 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 2       | 19567 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 4       | 39033 | 0.14       |

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 256      | 4       | 39033 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 8       | 77965 | 0.14       |
| im1      | im2      | im3     | 256     | 256      | 8       | 77965 | 0.14       |

The best configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 128      | 2       | 9473  | 0.14       |

The worst configuration:

| pSrcMtr1 | pSrcMtr2 | pDstMtr | nWidth1 | nHeight1 | nWidth2 | ticks | ticks/elem |
|----------|----------|---------|---------|----------|---------|-------|------------|
| im1      | im2      | im3     | 256     | 256      | 8       | 77965 | 0.14       |

## 6.110 nmppmMul\_mv\_

Умножение матрицы на вектор.

## Функции

- void nmppmMul\_mv\_8s64s (nm8s \*pSrcMtr, nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)
- void nmppmMul\_mv\_16s64s (nm16s \*pSrcMtr, nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)
- void nmppmMul\_mv\_32s64s (nm32s \*pSrcMtr, nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)
- void nmppmMul\_mv\_colmajor\_8s64s (const nm8s \*pSrcMtr, const nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)
- void nmppmMul\_mv\_colmajor\_16s64s (const nm16s \*pSrcMtr, const nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)
- void nmppmMul\_mv\_colmajor\_32s64s (const nm32s \*pSrcMtr, const nm64s \*pSrcVec, nm64s \*pDstVec, int nHeight, int nWidth)

## 6.110.1 Подробное описание

Умножение матрицы на вектор.

## Аргументы

|          |                                  |
|----------|----------------------------------|
| pSrcMtr  | Исходная матрица.                |
| pSrcVec  | Вектор-множитель.                |
| pSrcVec8 | Вектор-множитель размерности 8.  |
| nHeight  | Число строк исходной матрицы.    |
| nWidth   | Число столбцов исходной матрицы. |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.110.2 Функции

## 6.110.2.1 nmppmMul\_mv\_16s64s()

```
void nmppmMul_mv_16s64s (
    nm16s * pSrcMtr,
    nm64s * pSrcVec,
    nm64s * pDstVec,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im0     | im2     | 128     | 256    | 9330  | 0.28       |
| im0     | im0     | im2     | 128     | 256    | 9330  | 0.28       |
| im0     | im0     | im3     | 128     | 256    | 9330  | 0.28       |
| im0     | im0     | im3     | 128     | 256    | 9330  | 0.28       |
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |
| im0     | im1     | im3     | 128     | 256    | 8308  | 0.25       |
| im0     | im1     | im3     | 128     | 256    | 8308  | 0.25       |
| im1     | im0     | im2     | 128     | 256    | 8874  | 0.27       |
| im1     | im0     | im2     | 128     | 256    | 8874  | 0.27       |
| im1     | im0     | im3     | 128     | 256    | 8839  | 0.26       |
| im1     | im0     | im3     | 128     | 256    | 8839  | 0.26       |
| im1     | im1     | im2     | 128     | 256    | 9380  | 0.28       |
| im1     | im1     | im2     | 128     | 256    | 9380  | 0.28       |
| im1     | im1     | im3     | 128     | 256    | 9347  | 0.28       |
| im1     | im1     | im3     | 128     | 256    | 9347  | 0.28       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im1     | im1     | im2     | 128     | 256    | 9380  | 0.28       |

Perfomance table 1

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 628   | 0.30       |
| im0     | im1     | im2     | 128     | 16     | 628   | 0.30       |
| im0     | im1     | im2     | 128     | 128    | 4212  | 0.25       |
| im0     | im1     | im2     | 128     | 128    | 4212  | 0.25       |
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 628   | 0.30       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 8308  | 0.25       |

### 6.110.2.2 nmppmMul\_mv\_32s64s()

```
void nmppmMul_mv_32s64s (
    nm32s * pSrcMtr,
    nm64s * pSrcVec,
    nm64s * pDstVec,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im0     | im2     | 128     | 256    | 17546 | 0.53       |
| im0     | im0     | im2     | 128     | 256    | 17546 | 0.53       |
| im0     | im0     | im3     | 128     | 256    | 17546 | 0.53       |
| im0     | im0     | im3     | 128     | 256    | 17546 | 0.53       |
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |
| im0     | im1     | im3     | 128     | 256    | 16531 | 0.50       |
| im0     | im1     | im3     | 128     | 256    | 16531 | 0.50       |
| im1     | im0     | im2     | 128     | 256    | 17101 | 0.52       |
| im1     | im0     | im2     | 128     | 256    | 17101 | 0.52       |
| im1     | im0     | im3     | 128     | 256    | 17021 | 0.51       |
| im1     | im0     | im3     | 128     | 256    | 17021 | 0.51       |
| im1     | im1     | im2     | 128     | 256    | 17351 | 0.52       |
| im1     | im1     | im2     | 128     | 256    | 17351 | 0.52       |
| im1     | im1     | im3     | 128     | 256    | 17278 | 0.52       |
| im1     | im1     | im3     | 128     | 256    | 17278 | 0.52       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im0     | im2     | 128     | 256    | 17546 | 0.53       |

Perfomance table 1

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 1141  | 0.55       |
| im0     | im1     | im2     | 128     | 16     | 1141  | 0.55       |
| im0     | im1     | im2     | 128     | 128    | 8309  | 0.50       |
| im0     | im1     | im2     | 128     | 128    | 8309  | 0.50       |
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 1141  | 0.55       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 16531 | 0.50       |

### 6.110.2.3 nmppmMul\_mv\_8s64s()

```
void nmppmMul_mv_8s64s (
    nm8s * pSrcMtr,
    nm64s * pSrcVec,
    nm64s * pDstVec,
    int nHeight,
    int nWidth )
```

Perfomance table 0

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im0     | im2     | 128     | 256    | 5245  | 0.16       |
| im0     | im0     | im2     | 128     | 256    | 5245  | 0.16       |
| im0     | im0     | im3     | 128     | 256    | 5245  | 0.16       |
| im0     | im0     | im3     | 128     | 256    | 5245  | 0.16       |
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |
| im0     | im1     | im3     | 128     | 256    | 4222  | 0.12       |
| im0     | im1     | im3     | 128     | 256    | 4222  | 0.12       |
| im1     | im0     | im2     | 128     | 256    | 4762  | 0.14       |
| im1     | im0     | im2     | 128     | 256    | 4762  | 0.14       |
| im1     | im0     | im3     | 128     | 256    | 4762  | 0.14       |
| im1     | im0     | im3     | 128     | 256    | 4762  | 0.14       |
| im1     | im1     | im2     | 128     | 256    | 5814  | 0.17       |
| im1     | im1     | im2     | 128     | 256    | 5814  | 0.17       |
| im1     | im1     | im3     | 128     | 256    | 5814  | 0.17       |
| im1     | im1     | im3     | 128     | 256    | 5814  | 0.17       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im1     | im1     | im2     | 128     | 256    | 5814  | 0.17       |

Perfomance table 1

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 386   | 0.18       |
| im0     | im1     | im2     | 128     | 16     | 386   | 0.18       |
| im0     | im1     | im2     | 128     | 128    | 2174  | 0.13       |
| im0     | im1     | im2     | 128     | 128    | 2174  | 0.13       |
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |

The best configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 16     | 386   | 0.18       |

The worst configuration:

| pSrcMtr | pSrcVec | pDstVec | nHeight | nWidth | ticks | ticks/elem |
|---------|---------|---------|---------|--------|-------|------------|
| im0     | im1     | im2     | 128     | 256    | 4222  | 0.12       |

## 6.111 nmppmMul\_mv\_8xH

Умножение матрицы на вектор.

### Функции

- void nmppmMul\_mv\_8s16s\_8xH (v8nm8s \*pSrcMtr, v8nm16s \*pSrcVec, nm16s \*pDstVec, int nHeight)
- void nmppmMul\_mv\_16s16s\_8xH (v8nm16s \*pSrcMtr, v8nm16s \*pSrcVec, nm16s \*pDstVec, int nHeight)

### 6.111.1 Подробное описание

Умножение матрицы на вектор.

#### Аргументы

|          |                                 |
|----------|---------------------------------|
| pSrcMtr  | Исходная матрица.               |
| pSrcVec  | Вектор-множитель.               |
| pSrcVec8 | Вектор-множитель размерности 8. |
| nHeight  | Число строк исходной матрицы.   |

#### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

#### Возвращает

void



## 6.112 nmppmMul\_mv\_\_AddC

Умножение матрицы на вектор с добавлением константы.

Функции

- void nmppmMul\_mv\_\_AddC (v2nm32s \*pSrcMtr, v2nm32s \*pnSrcVec, int nAddVal, nm32s \*pDstVec, int nHeight)

### 6.112.1 Подробное описание

Умножение матрицы на вектор с добавлением константы.

Аргументы

|         |                                                         |
|---------|---------------------------------------------------------|
| pSrcMtr | Исходная матрица.                                       |
| pSrcVec | Вектор-множитель.                                       |
| pSrcVec | Вектор-множитель размерности 2.                         |
| nAddVal | Константа.                                              |
| nHeight | Число строк исходной матрицы. nHeight = [0, 2, 4, ...]. |
| nWidth  | Число столбцов исходной матрицы.                        |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.113 MTR\_ProdUnitV

Умножение матрицы на единичный вектор.

Функции

- void MTR\_ProdUnitV\_16s\_4xH (v4nm16s \*pSrcMtr, nm16s \*pDstVec, int nHeight)
- void MTR\_ProdUnitV\_16s\_16xH (v16nm8s \*pSrcMtr, nm16s \*pDstVec, int nHeight)

### 6.113.1 Подробное описание

Умножение матрицы на единичный вектор.

$$pDstVec(i) = \sum_{j=0}^{w-1} pSrcMtr(i, j)$$

Данная функция эквивалентна суммированию столбцов матрицы. Ширины матрицы, для которых имеется реализация данной функции указываются в ее названии.

Аргументы

|         |                                            |
|---------|--------------------------------------------|
| pSrcMtr | Матрица.                                   |
| nHeight | Число строк матрицы. nHeight=[128,256,...] |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.114 MTR\_Malloc

Распределение памяти для матриц библиотеки.

Начало и конец распределяемой памяти выравнивается на начало 64-х разрядного слова.

Распределение памяти для матриц библиотеки.

Начало и конец распределяемой памяти выравнивается на начало 64-х разрядного слова.

Аргументы

|         |                                                                     |
|---------|---------------------------------------------------------------------|
| nHeight | Число строк в матрице.                                              |
| nWidth  | Число столбцов в матрице.                                           |
| hint    | Номер банка памяти. Может принимать значения MEM_LOCAL, MEM_GLOBAL. |

Заметки

Память, распределенная с помощью функций MTR\_Malloc должна освобождаться с помощью функции MTR\_Free.

## 6.115 MTR\_Free

Освобождение памяти для матриц.

### Функции

- `__INLINE__ void MTR_Free (void *ptr)`

### 6.115.1 Подробное описание

Освобождение памяти для матриц.

### Заметки

Данная функция должна вызываться только для векторов, распределенных с помощью функций MTR\_Malloc.

## 6.116 MTR\_Addr

Возвращает адрес ячейки памяти, содержащей указанный элемент.

Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.

### Функции

- `__INLINE__ nm1 * MTR_Addr_1 (nm1 *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm2s * MTR_Addr_2s (nm2s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm4s * MTR_Addr_4s (nm4s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm8s * MTR_Addr_8s (nm8s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm16s * MTR_Addr_16s (nm16s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm32s * MTR_Addr_32s (nm32s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm64s * MTR_Addr_64s (nm64s *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm2u * MTR_Addr_2u (nm2u *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm4u * MTR_Addr_4u (nm4u *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm8u * MTR_Addr_8u (nm8u *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm16u * MTR_Addr_16u (nm16u *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm32u * MTR_Addr_32u (nm32u *pMTR, int nWidth, int nY, int nX)`
- `__INLINE__ nm64u * MTR_Addr_64u (nm64u *pMTR, int nWidth, int nY, int nX)`

### 6.116.1 Подробное описание

Возвращает адрес ячейки памяти, содержащей указанный элемент.

Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.

#### Аргументы

|        |                             |
|--------|-----------------------------|
| pMtr   | Входная матрица.            |
| nWidth | Ширина таблицы в элементах. |
| nY     | Номер строки.               |
| nX     | Номер колонки.              |

Возвращает

Адрес ячейки памяти.

#### Заметки

Для ускорения работы на PC возможно использование макроса ADDR(ptr, index), который раскрывается на PC как (ptr+index), а на NM как вызов функции MTR\_Addr.

### 6.116.2 Функции

## 6.116.2.1 MTR\_Addr\_1()

```
__INLINE__ nm1 * MTR_Addr_1 (
    nm1 * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 180

## 6.116.2.2 MTR\_Addr\_16s()

```
__INLINE__ nm16s * MTR_Addr_16s (
    nm16s * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 184

### 6.116.2.3 MTR\_Addr\_16u()

```
__INLINE__ nm16u * MTR_Addr_16u (
    nm16u * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 191

### 6.116.2.4 MTR\_Addr\_2s()

```
__INLINE__ nm2s * MTR_Addr_2s (
    nm2s * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 181

#### 6.116.2.5 MTR\_Addr\_2u()

```
__INLINE__ nm2u * MTR_Addr_2u (
    nm2u * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 188

#### 6.116.2.6 MTR\_Addr\_32s()

```
__INLINE__ nm32s * MTR_Addr_32s (
    nm32s * pMTR,
```



```

    int nWidth,
    int nY,
    int nX )

```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |
| im0  | 4      | 2  | 2  | 141   | 141.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |

См. определение в файле mSupport.h строка 185

#### 6.116.2.7 MTR\_Addr\_32u()

```

__INLINE__ nm32u * MTR_Addr_32u (
    nm32u * pMTR,
    int nWidth,
    int nY,
    int nX )

```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |
| im0  | 4      | 2  | 2  | 141   | 141.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 141   | 141.00     |

См. определение в файле mSupport.h строка 192

#### 6.116.2.8 MTR\_Addr\_4s()

```
__INLINE__ nm4s * MTR_Addr_4s (
    nm4s * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 182

#### 6.116.2.9 MTR\_Addr\_4u()

```
__INLINE__ nm4u * MTR_Addr_4u (
    nm4u * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 189

#### 6.116.2.10 MTR\_Addr\_64s()

```
__INLINE__ nm64s * MTR_Addr_64s (
    nm64s * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 186

#### 6.116.2.11 MTR\_Addr\_64u()

```
__INLINE__ nm64u * MTR_Addr_64u (
    nm64u * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 193

#### 6.116.2.12 MTR\_Addr\_8s()

```
__INLINE__ nm8s * MTR_Addr_8s (
    nm8s * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 183

## 6.116.2.13 MTR\_Addr\_8u()

```
__INLINE__ nm8u * MTR_Addr_8u (
    nm8u * pMTR,
    int nWidth,
    int nY,
    int nX )
```

Perfomance table 0

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The best configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

The worst configuration:

| pMTR | nWidth | nY | nX | ticks | ticks/elem |
|------|--------|----|----|-------|------------|
| im0  | 4      | 2  | 2  | 186   | 186.00     |

См. определение в файле mSupport.h строка 190

## 6.117 MTR\_SetVal

Записывают число в элемент матрицы.

### Функции

- `__INLINE__ void MTR_SetVal_1 (nm1 *pMtr, int nWidth, int nY, int nX, int1b nVal)`
- `__INLINE__ void MTR_SetVal_2s (nm2s *pMtr, int nWidth, int nY, int nX, int2b nVal)`
- `__INLINE__ void MTR_SetVal_4s (nm4s *pMtr, int nWidth, int nY, int nX, int4b nVal)`
- `__INLINE__ void MTR_SetVal_8s (nm8s *pMtr, int nWidth, int nY, int nX, int8b nVal)`
- `__INLINE__ void MTR_SetVal_16s (nm16s *pMtr, int nWidth, int nY, int nX, int16b nVal)`
- `__INLINE__ void MTR_SetVal_32s (nm32s *pMtr, int nWidth, int nY, int nX, int32b nVal)`
- `__INLINE__ void MTR_SetVal_64s (nm64s *pMtr, int nWidth, int nY, int nX, int64b nVal)`
- `__INLINE__ void MTR_SetVal_2u (nm2u *pMtr, int nWidth, int nY, int nX, uint2b nVal)`
- `__INLINE__ void MTR_SetVal_4u (nm4u *pMtr, int nWidth, int nY, int nX, uint4b nVal)`
- `__INLINE__ void MTR_SetVal_8u (nm8u *pMtr, int nWidth, int nY, int nX, uint8b nVal)`
- `__INLINE__ void MTR_SetVal_16u (nm16u *pMtr, int nWidth, int nY, int nX, uint16b nVal)`
- `__INLINE__ void MTR_SetVal_32u (nm32u *pMtr, int nWidth, int nY, int nX, uint32b nVal)`
- `__INLINE__ void MTR_SetVal_64u (nm64u *pMtr, int nWidth, int nY, int nX, uint64b nVal)`

### 6.117.1 Подробное описание

Записывают число в элемент матрицы.

$$pMtr[nY][nX] = nVal$$

### Аргументы

|        |                            |
|--------|----------------------------|
| pMtr   | Матрица.                   |
| nWidth | Ширина матрицы в элементах |
| nY     | Номер строки               |
| nX     | Номер столбца              |
| nVal   | Значение элемента          |

Возвращает

void

### 6.117.2 Функции

## 6.117.2.1 MTR\_SetVal\_1()

```
__INLINE__ void MTR_SetVal_1 (
    nm1 * pMtr,
    int nWidth,
    int nY,
    int nX,
    int1b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

См. определение в файле mSupport.h строка 245

## 6.117.2.2 MTR\_SetVal\_16s()

```
__INLINE__ void MTR_SetVal_16s (
    nm16s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int16b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

См. определение в файле mSupport.h строка 249

### 6.117.2.3 MTR\_SetVal\_16u()

```
__INLINE__ void MTR_SetVal_16u (
    nm16u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint16b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

См. определение в файле mSupport.h строка 256

### 6.117.2.4 MTR\_SetVal\_2s()

```
__INLINE__ void MTR_SetVal_2s (
    nm2s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int2b nVal )
```

Perfomance table 0



| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

См. определение в файле mSupport.h строка 246

#### 6.117.2.5 MTR\_SetVal\_2u()

```
__INLINE__ void MTR_SetVal_2u (
    nm2u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint2b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

См. определение в файле mSupport.h строка 253

## 6.117.2.6 MTR\_SetVal\_32s()

```
__INLINE__ void MTR_SetVal_32s (
    nm32s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int32b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

См. определение в файле mSupport.h строка 250

## 6.117.2.7 MTR\_SetVal\_32u()

```
__INLINE__ void MTR_SetVal_32u (
    nm32u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint32b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 195   | 195.00     |

См. определение в файле mSupport.h строка 257

#### 6.117.2.8 MTR\_SetVal\_4s()

```
__INLINE__ void MTR_SetVal_4s (
    nm4s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int4b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

См. определение в файле mSupport.h строка 247

#### 6.117.2.9 MTR\_SetVal\_4u()

```
__INLINE__ void MTR_SetVal_4u (
    nm4u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint4b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

См. определение в файле mSupport.h строка 254

#### 6.117.2.10 MTR\_SetVal\_64s()

```
__INLINE__ void MTR_SetVal_64s (
    nm64s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int64b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

См. определение в файле mSupport.h строка 251

## 6.117.2.11 MTR\_SetVal\_64u()

```
__INLINE__ void MTR_SetVal_64u (
    nm64u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint64b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 259   | 259.00     |

См. определение в файле mSupport.h строка 258

## 6.117.2.12 MTR\_SetVal\_8s()

```
__INLINE__ void MTR_SetVal_8s (
    nm8s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int8b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 194   | 194.00     |

См. определение в файле mSupport.h строка 248

#### 6.117.2.13 MTR\_SetVal\_8u()

```
__INLINE__ void MTR_SetVal_8u (
    nm8u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint8b nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 3    | 247   | 247.00     |

См. определение в файле mSupport.h строка 255

## 6.118 MTR\_GetVal

Считывает значение элемента матрицы.

### Функции

- `__INLINE__ void MTR_GetVal_1 (nm1 *pMtr, int nWidth, int nY, int nX, int1b *nVal)`
- `__INLINE__ void MTR_GetVal_2s (nm2s *pMtr, int nWidth, int nY, int nX, int2b *nVal)`
- `__INLINE__ void MTR_GetVal_4s (nm4s *pMtr, int nWidth, int nY, int nX, int4b *nVal)`
- `__INLINE__ void MTR_GetVal_8s (nm8s *pMtr, int nWidth, int nY, int nX, int8b *nVal)`
- `__INLINE__ void MTR_GetVal_16s (nm16s *pMtr, int nWidth, int nY, int nX, int16b *nVal)`
- `__INLINE__ void MTR_GetVal_32s (nm32s *pMtr, int nWidth, int nY, int nX, int32b *nVal)`
- `__INLINE__ void MTR_GetVal_64s (nm64s *pMtr, int nWidth, int nY, int nX, int64b *nVal)`
- `__INLINE__ void MTR_GetVal_2u (nm2u *pMtr, int nWidth, int nY, int nX, uint2b *nVal)`
- `__INLINE__ void MTR_GetVal_4u (nm4u *pMtr, int nWidth, int nY, int nX, uint4b *nVal)`
- `__INLINE__ void MTR_GetVal_8u (nm8u *pMtr, int nWidth, int nY, int nX, uint8b *nVal)`
- `__INLINE__ void MTR_GetVal_16u (nm16u *pMtr, int nWidth, int nY, int nX, uint16b *nVal)`
- `__INLINE__ void MTR_GetVal_32u (nm32u *pMtr, int nWidth, int nY, int nX, uint32b *nVal)`
- `__INLINE__ void MTR_GetVal_64u (nm64u *pMtr, int nWidth, int nY, int nX, uint64b *nVal)`

### 6.118.1 Подробное описание

Считывает значение элемента матрицы.

$$nVal = pMtr[nY][nX]$$

### Аргументы

|        |                            |
|--------|----------------------------|
| pMtr   | Матрица.                   |
| nWidth | Ширина матрицы в элементах |
| nY     | Номер строки               |
| nX     | Номер столбца              |

### Возвращаемые значения

|      |                   |
|------|-------------------|
| nVal | Значение элемента |
|------|-------------------|

### Возвращает

void

### 6.118.2 Функции

## 6.118.2.1 MTR\_GetVal\_1()

```
__INLINE__ void MTR_GetVal_1 (
    nm1 * pMtr,
    int nWidth,
    int nY,
    int nX,
    int1b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 310

## 6.118.2.2 MTR\_GetVal\_16s()

```
__INLINE__ void MTR_GetVal_16s (
    nm16s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int16b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |



The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 314

### 6.118.2.3 MTR\_GetVal\_16u()

```
__INLINE__ void MTR_GetVal_16u (
    nm16u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint16b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 321

### 6.118.2.4 MTR\_GetVal\_2s()

```
__INLINE__ void MTR_GetVal_2s (
    nm2s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int2b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 311

#### 6.118.2.5 MTR\_GetVal\_2u()

```
__INLINE__ void MTR_GetVal_2u (
    nm2u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint2b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 318

## 6.118.2.6 MTR\_GetVal\_32s()

```
__INLINE__ void MTR_GetVal_32s (
    nm32s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int32b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

См. определение в файле mSupport.h строка 315

## 6.118.2.7 MTR\_GetVal\_32u()

```
__INLINE__ void MTR_GetVal_32u (
    nm32u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint32b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 200   | 200.00     |

См. определение в файле mSupport.h строка 322

#### 6.118.2.8 MTR\_GetVal\_4s()

```
__INLINE__ void MTR_GetVal_4s (
    nm4s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int4b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 312

#### 6.118.2.9 MTR\_GetVal\_4u()

```
__INLINE__ void MTR_GetVal_4u (
    nm4u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint4b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 319

#### 6.118.2.10 MTR\_GetVal\_64s()

```
__INLINE__ void MTR_GetVal_64s (
    nm64s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int64b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

См. определение в файле mSupport.h строка 316

## 6.118.2.11 MTR\_GetVal\_64u()

```
__INLINE__ void MTR_GetVal_64u (
    nm64u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint64b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 262   | 262.00     |

См. определение в файле mSupport.h строка 323

## 6.118.2.12 MTR\_GetVal\_8s()

```
__INLINE__ void MTR_GetVal_8s (
    nm8s * pMtr,
    int nWidth,
    int nY,
    int nX,
    int8b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 313

#### 6.118.2.13 MTR\_GetVal\_8u()

```
__INLINE__ void MTR_GetVal_8u (
    nm8u * pMtr,
    int nWidth,
    int nY,
    int nX,
    uint8b * nVal )
```

Perfomance table 0

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The best configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

The worst configuration:

| pMtr | nWidth | nY | nX | nVal | ticks | ticks/elem |
|------|--------|----|----|------|-------|------------|
| im0  | 18     | 2  | 2  | 5    | 191   | 191.00     |

См. определение в файле mSupport.h строка 320

## 6.119 Функции поддержки

### Группы

- [MTR\\_Malloc](#)

Распределение памяти для матриц библиотеки.

Начало и конец распределяемой памяти выравнивается на начало 64-х разрядного слова.

- [MTR\\_Free](#)

Освобождение памяти для матриц.

- [MTR\\_Addr](#)

Возвращает адрес ячейки памяти, содержащей указанный элемент.

Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.

- [MTR\\_SetVal](#)

Записывает число в элемент матрицы.

- [MTR\\_GetVal](#)

Считывает значение элемента матрицы.

### 6.119.1 Подробное описание



## 6.120 Векторно-матричные операции

### Группы

- [nmppmMul\\_mm](#)  
Умножение матрицы на матрицу.
- [nmppmMul\\_mv\\_](#)  
Умножение матрицы на вектор.
- [nmppmMul\\_mv\\_8xH](#)  
Умножение матрицы на вектор.
- [nmppmMul\\_mv\\_\\_AddC](#)  
Умножение матрицы на вектор с добавлением константы.
- [MTR\\_ProdUnitV](#)  
Умножение матрицы на единичный вектор.

### 6.120.1 Подробное описание

## 6.121 Обращение матрицы

### Группы

- [MTR\\_fpResolve\\_Gauss](#)

Решение системы линейных уравнений методом Гаусса.

- [MTR\\_fpResolve\\_PivotGauss](#)

Решение системы линейных уравнений методом Гаусса с выбором ведущего элемента.

- [nmppmLUDcomp](#)

LU-разложение матрицы

- [nmppmLUResolve](#)

Решение системы линейных уравнений из LU-разложения  $L*y = b$ ;  $U*x = y$ .

### 6.121.1 Подробное описание

## 6.122 FFT-256

### Функции

- void [FFT\\_Fwd256Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Fwd256Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Fwd256](#) (nm32sc \*GSrcBuffer, nm32sc \*LDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR=-1)  
Прямое быстрое преобразование Фурье-256.

### 6.122.1 Подробное описание

### 6.122.2 Функции

#### 6.122.2.1 FFT\_Fwd256()

```
void FFT_Fwd256 (
    nm32sc * GSrcBuffer,
    nm32sc * LDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR = -1 )
```

Прямое быстрое преобразование Фурье-256.

Функция выполняет дискретное комплексное 256-точечное преобразование Фурье на базе алгоритма БПФ по основанию 16-16

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 256 64-р. слов                                                                                                                                                                                                                                                                                                                                                                                       |
| out | LDstBuffer | Результирующий массив размером 256*3 64-р. слов                                                                                                                                                                                                                                                                                                                                                                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 256*3 64-р. слов                                                                                                                                                                                                                                                                                                                                                     |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 256*2 64-р. слов                                                                                                                                                                                                                                                                                                                                                   |
| in  | ShiftR     | Коэффициент нормализации, выполняет арифметический сдвиг результирующего массива на ShiftR бит вправо для получения нормализованного массива LDstBuffer. При передаче значения по умолчанию (-1) ShiftR автоматически принимается равным 14 если ранее установлена точность 7-бит функцией <a href="#">FFT_Fwd256Set7bit</a> () и 12 - если ранее установлена точность 6-бит функцией <a href="#">FFT_Fwd256Set6bit</a> () . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | LDstBuffer | LBuffer | GBuffer | Shift R | clocks |
|------------|------------|---------|---------|---------|--------|
| L          | L          | L       | L       | -1      | 21.54  |
| L          | L          | L       | G       | -1      | 16.60  |
| L          | L          | G       | L       | -1      | 22.17  |
| L          | L          | G       | G       | -1      | 20.98  |
| L          | G          | L       | L       | -1      | 20.53  |
| L          | G          | L       | G       | -1      | 17.56  |
| L          | G          | G       | L       | -1      | 21.17  |
| L          | G          | G       | G       | -1      | 21.94  |
| G          | L          | L       | L       | -1      | 20.57  |
| G          | L          | L       | G       | -1      | 15.64  |
| G          | L          | G       | L       | -1      | 21.21  |
| G          | L          | G       | G       | -1      | 20.02  |
| G          | G          | L       | L       | -1      | 19.57  |
| G          | G          | L       | G       | -1      | 16.59  |
| G          | G          | G       | L       | -1      | 20.20  |
| G          | G          | G       | G       | -1      | 20.97  |
| L          | L          | L       | L       | 0       | 21.51  |
| L          | L          | L       | G       | 0       | 16.58  |
| L          | L          | G       | L       | 0       | 22.15  |
| L          | L          | G       | G       | 0       | 20.96  |
| L          | G          | L       | L       | 0       | 20.51  |
| L          | G          | L       | G       | 0       | 17.53  |
| L          | G          | G       | L       | 0       | 21.14  |
| L          | G          | G       | G       | 0       | 21.91  |
| G          | L          | L       | L       | 0       | 20.55  |
| G          | L          | L       | G       | 0       | 15.62  |
| G          | L          | G       | L       | 0       | 21.19  |
| G          | L          | G       | G       | 0       | 20.00  |
| G          | G          | L       | L       | 0       | 19.54  |
| G          | G          | L       | G       | 0       | 16.57  |
| G          | G          | G       | L       | 0       | 20.18  |
| G          | G          | G       | G       | 0       | 20.95  |

## 6.123 IFFT-256

### Функции

- void [FFT\\_Inv256Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Inv256Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Inv256](#) (nm32sc \*GSrcBuffer, nm32sc \*GDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR1=8, int ShiftR2=-1)  
Обратное быстрое преобразование Фурье. ОБПФ-256.

### 6.123.1 Подробное описание

### 6.123.2 Функции

#### 6.123.2.1 FFT\_Inv256()

```
void FFT_Inv256 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR1 = 8,
    int ShiftR2 = -1 )
```

Обратное быстрое преобразование Фурье. ОБПФ-256.

Функция выполняет обратное дискретное комплексное 256-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 16-16.

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                   |
|-----|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 256 64-р. слов                                                                                                                                                                                                                                                                                                            |
| out | GDstBuffer | Результирующий массив размером 256 64-р. слов                                                                                                                                                                                                                                                                                                     |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 256*3 64-р. слов                                                                                                                                                                                                                                                                          |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 256*2 64-р. слов                                                                                                                                                                                                                                                                        |
| in  | ShiftR1    | Промежуточный сдвиг результатов на ShiftR1 бит вправо (первая нормализация). Необходимо для предотвращения переполнения. По умолчанию равен 8                                                                                                                                                                                                     |
| in  | ShiftR2    | Заключительный сдвиг результатов на ShiftR2 бит вправо (вторая нормализация) в конце вычисления обратного БПФ. По умолчанию ShiftR2 принимается равным 14 при установленной точности 7-бит с помощью функции <a href="#">FFT_Inv256Set7bit()</a> и 12 - при точности 6-бит, установленной с помощью функции <a href="#">FFT_Inv256Set6bit()</a> . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | GDstBuffer | LBuffer | GBuffer | ShiftR1 | ShiftR2 | clocks |
|------------|------------|---------|---------|---------|---------|--------|
|------------|------------|---------|---------|---------|---------|--------|

|   |   |   |   |   |    |       |
|---|---|---|---|---|----|-------|
| L | L | L | L | 8 | -1 | 26.76 |
| L | L | L | L | 0 | -1 | 26.76 |
| L | L | L | G | 8 | -1 | 19.07 |
| L | L | L | G | 0 | -1 | 19.07 |
| L | L | G | L | 8 | -1 | 22.87 |
| L | L | G | L | 0 | -1 | 22.87 |
| L | L | G | G | 8 | -1 | 20.77 |
| L | L | G | G | 0 | -1 | 20.77 |
| L | G | L | L | 8 | -1 | 25.75 |
| L | G | L | L | 0 | -1 | 25.75 |
| L | G | L | G | 8 | -1 | 18.06 |
| L | G | L | G | 0 | -1 | 18.06 |
| L | G | G | L | 8 | -1 | 23.82 |
| L | G | G | L | 0 | -1 | 23.82 |
| L | G | G | G | 8 | -1 | 21.72 |
| L | G | G | G | 0 | -1 | 21.72 |
| G | L | L | L | 8 | -1 | 25.80 |
| G | L | L | L | 0 | -1 | 25.80 |
| G | L | L | G | 8 | -1 | 18.10 |
| G | L | L | G | 0 | -1 | 18.10 |
| G | L | G | L | 8 | -1 | 21.91 |
| G | L | G | L | 0 | -1 | 21.91 |
| G | L | G | G | 8 | -1 | 19.80 |
| G | L | G | G | 0 | -1 | 19.80 |
| G | G | L | L | 8 | -1 | 24.79 |
| G | G | L | L | 0 | -1 | 24.79 |
| G | G | L | G | 8 | -1 | 17.09 |
| G | G | L | G | 0 | -1 | 17.09 |
| G | G | G | L | 8 | -1 | 22.86 |
| G | G | G | L | 0 | -1 | 22.86 |
| G | G | G | G | 8 | -1 | 20.76 |
| G | G | G | G | 0 | -1 | 20.76 |
| L | L | L | L | 8 | 0  | 26.74 |
| L | L | L | L | 0 | 0  | 26.74 |
| L | L | L | G | 8 | 0  | 19.04 |
| L | L | L | G | 0 | 0  | 19.04 |
| L | L | G | L | 8 | 0  | 22.85 |
| L | L | G | L | 0 | 0  | 22.85 |
| L | L | G | G | 8 | 0  | 20.75 |
| L | L | G | G | 0 | 0  | 20.75 |
| L | G | L | L | 8 | 0  | 25.73 |
| L | G | L | L | 0 | 0  | 25.73 |
| L | G | L | G | 8 | 0  | 18.03 |
| L | G | L | G | 0 | 0  | 18.03 |
| L | G | G | L | 8 | 0  | 23.80 |
| L | G | G | L | 0 | 0  | 23.80 |
| L | G | G | G | 8 | 0  | 21.70 |
| L | G | G | G | 0 | 0  | 21.70 |
| G | L | L | L | 8 | 0  | 25.77 |
| G | L | L | L | 0 | 0  | 25.77 |
| G | L | L | G | 8 | 0  | 18.08 |
| G | L | L | G | 0 | 0  | 18.08 |
| G | L | G | L | 8 | 0  | 21.88 |
| G | L | G | L | 0 | 0  | 21.88 |
| G | L | G | G | 8 | 0  | 19.78 |
| G | L | G | G | 0 | 0  | 19.78 |
| G | G | L | L | 8 | 0  | 24.76 |
| G | G | L | L | 0 | 0  | 24.76 |
| G | G | L | G | 8 | 0  | 17.07 |
| G | G | L | G | 0 | 0  | 17.07 |
| G | G | G | L | 8 | 0  | 22.83 |

|   |  |   |  |   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|---|--|---|--|-------|
| G |  | G |  | G |  | L |  | 0 |  | 0 |  | 22.83 |
| G |  | G |  | G |  | G |  | 8 |  | 0 |  | 20.73 |
| G |  | G |  | G |  | G |  | 0 |  | 0 |  | 20.73 |

## 6.124 FFT-512

### Функции

- void [FFT\\_Fwd512Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Fwd512Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Fwd512](#) (nm32sc \*GSrcBuffer, nm32sc \*GDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR=-1)  
Прямое быстрое преобразование Фурье-512.

### 6.124.1 Подробное описание

### 6.124.2 Функции

#### 6.124.2.1 FFT\_Fwd512()

```
void FFT_Fwd512 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR = -1 )
```

Прямое быстрое преобразование Фурье-512.

Функция выполняет дискретное комплексное 512-точечное преобразование Фурье на базе алгоритма БПФ по основанию 2-16-16

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 512 64-р. слов                                                                                                                                                                                                                                                                                                                                                                                     |
| out | GDstBuffer | Результирующий массив размером 512 64-р. слов                                                                                                                                                                                                                                                                                                                                                                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 512*2 64-р. слов                                                                                                                                                                                                                                                                                                                                                   |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 512*3 64-р. слов                                                                                                                                                                                                                                                                                                                                                 |
| in  | ShiftR     | Коэффициент нормализации, выполняет арифметический сдвиг результирующего массива на ShiftR бит вправо для получения нормализованного массива LDstBuffer. При передаче значения по умолчанию (-1) ShiftR автоматически принимается равным 14 если ранее установлена точность 7-бит функцией <a href="#">FFT_Fwd512Set7bit()</a> и 12 - если ранее установлена точность 6-бит функцией <a href="#">FFT_Fwd512Set6bit()</a> . |



Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | GDstBuffer | LBuffer | GBuffer | ShiftR | clocks |
|------------|------------|---------|---------|--------|--------|
| L          | L          | L       | L       | -1     | 24.12  |
| L          | L          | L       | G       | -1     | 19.29  |
| L          | L          | G       | L       | -1     | 22.81  |
| L          | L          | G       | G       | -1     | 21.62  |
| L          | G          | L       | L       | -1     | 23.10  |
| L          | G          | L       | G       | -1     | 18.27  |
| L          | G          | G       | L       | -1     | 23.77  |
| L          | G          | G       | G       | -1     | 22.58  |
| G          | L          | L       | L       | -1     | 23.06  |
| G          | L          | L       | G       | -1     | 18.23  |
| G          | L          | G       | L       | -1     | 23.79  |
| G          | L          | G       | G       | -1     | 22.60  |
| G          | G          | L       | L       | -1     | 22.04  |
| G          | G          | L       | G       | -1     | 17.21  |
| G          | G          | G       | L       | -1     | 24.75  |
| G          | G          | G       | G       | -1     | 23.56  |
| L          | L          | L       | L       | 0      | 24.11  |
| L          | L          | L       | G       | 0      | 19.28  |
| L          | L          | G       | L       | 0      | 22.80  |
| L          | L          | G       | G       | 0      | 21.61  |
| L          | G          | L       | L       | 0      | 23.09  |
| L          | G          | L       | G       | 0      | 18.26  |
| L          | G          | G       | L       | 0      | 23.76  |
| L          | G          | G       | G       | 0      | 22.57  |
| G          | L          | L       | L       | 0      | 23.05  |
| G          | L          | L       | G       | 0      | 18.22  |
| G          | L          | G       | L       | 0      | 23.78  |
| G          | L          | G       | G       | 0      | 22.59  |
| G          | G          | L       | L       | 0      | 22.03  |
| G          | G          | L       | G       | 0      | 17.20  |
| G          | G          | G       | L       | 0      | 24.74  |
| G          | G          | G       | G       | 0      | 23.55  |

## 6.125 IFFT-512

### Функции

- void [FFT\\_Inv512Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Inv512Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Inv512](#) (nm32sc \*GSrcBuffer, nm32sc \*LDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR1=9, int ShiftR2=-1)  
Обратное быстрое преобразование Фурье. ОБПФ-512.

### 6.125.1 Подробное описание

### 6.125.2 Функции

#### 6.125.2.1 FFT\_Inv512()

```
void FFT_Inv512 (
    nm32sc * GSrcBuffer,
    nm32sc * LDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR1 = 9,
    int ShiftR2 = -1 )
```

Обратное быстрое преобразование Фурье. ОБПФ-512.

Функция выполняет обратное дискретное комплексное 512-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 2-16-16.

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                     |
|-----|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 512 64-р. слов                                                                                                                                                                                                                                                                                                              |
| out | LDstBuffer | Результирующий массив размером 512 64-р. слов                                                                                                                                                                                                                                                                                                       |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 512*3 64-р. слов                                                                                                                                                                                                                                                                            |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 512*3 64-р. слов                                                                                                                                                                                                                                                                          |
| in  | ShiftR1    | Промежуточный сдвиг результатов на ShiftR1 бит вправо (первая нормализация). Необходимо для предотвращения переполнения. По умолчанию равен 9                                                                                                                                                                                                       |
| in  | ShiftR2    | Заключительный сдвиг результатов на ShiftR2 бит вправо (вторая нормализация) в конце вычисления обратного БПФ. По умолчанию ShiftR2 принимается равным 14 при установленной точности 7-бит с помощью функции <a href="#">FFT_Inv512Set7bit</a> () и 12 - при точности 6-бит, установленной с помощью функции <a href="#">FFT_Inv512Set6bit</a> () . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

-----  
 GSrcBuffer| LDstBuffer| LBuffer | GBuffer | ShiftR1 | ShiftR2 | clocks

|   |   |   |   |   |    |       |
|---|---|---|---|---|----|-------|
| L | L | L | L | 9 | -1 | 23.41 |
| L | L | L | L | 0 | -1 | 23.40 |
| L | L | L | G | 9 | -1 | 19.39 |
| L | L | L | G | 0 | -1 | 19.38 |
| L | L | G | L | 9 | -1 | 24.86 |
| L | L | G | L | 0 | -1 | 24.86 |
| L | L | G | G | 9 | -1 | 26.47 |
| L | L | G | G | 0 | -1 | 26.46 |
| L | G | L | L | 9 | -1 | 22.39 |
| L | G | L | L | 0 | -1 | 22.38 |
| L | G | L | G | 9 | -1 | 20.35 |
| L | G | L | G | 0 | -1 | 20.34 |
| L | G | G | L | 9 | -1 | 23.84 |
| L | G | G | L | 0 | -1 | 23.84 |
| L | G | G | G | 9 | -1 | 27.43 |
| L | G | G | G | 0 | -1 | 27.42 |
| G | L | L | L | 9 | -1 | 22.35 |
| G | L | L | L | 0 | -1 | 22.34 |
| G | L | L | G | 9 | -1 | 18.34 |
| G | L | L | G | 0 | -1 | 18.33 |
| G | L | G | L | 9 | -1 | 25.84 |
| G | L | G | L | 0 | -1 | 25.84 |
| G | L | G | G | 9 | -1 | 27.45 |
| G | L | G | G | 0 | -1 | 27.44 |
| G | G | L | L | 9 | -1 | 21.33 |
| G | G | L | L | 0 | -1 | 21.33 |
| G | G | L | G | 9 | -1 | 19.30 |
| G | G | L | G | 0 | -1 | 19.29 |
| G | G | G | L | 9 | -1 | 24.83 |
| G | G | G | L | 0 | -1 | 24.82 |
| G | G | G | G | 9 | -1 | 28.41 |
| G | G | G | G | 0 | -1 | 28.41 |
| L | L | L | L | 9 | 0  | 23.40 |
| L | L | L | L | 0 | 0  | 23.39 |
| L | L | L | G | 9 | 0  | 19.38 |
| L | L | L | G | 0 | 0  | 19.37 |
| L | L | G | L | 9 | 0  | 24.85 |
| L | L | G | L | 0 | 0  | 24.85 |
| L | L | G | G | 9 | 0  | 26.45 |
| L | L | G | G | 0 | 0  | 26.45 |
| L | G | L | L | 9 | 0  | 22.38 |
| L | G | L | L | 0 | 0  | 22.37 |
| L | G | L | G | 9 | 0  | 20.34 |
| L | G | L | G | 0 | 0  | 20.33 |
| L | G | G | L | 9 | 0  | 23.83 |
| L | G | G | L | 0 | 0  | 23.83 |
| L | G | G | G | 9 | 0  | 27.41 |
| L | G | G | G | 0 | 0  | 27.41 |
| G | L | L | L | 9 | 0  | 22.34 |
| G | L | L | L | 0 | 0  | 22.33 |
| G | L | L | G | 9 | 0  | 18.33 |
| G | L | L | G | 0 | 0  | 18.32 |
| G | L | G | L | 9 | 0  | 25.83 |
| G | L | G | L | 0 | 0  | 25.83 |
| G | L | G | G | 9 | 0  | 27.43 |
| G | L | G | G | 0 | 0  | 27.43 |
| G | G | L | L | 9 | 0  | 21.32 |
| G | G | L | L | 0 | 0  | 21.31 |
| G | G | L | G | 9 | 0  | 19.29 |
| G | G | L | G | 0 | 0  | 19.28 |
| G | G | G | L | 9 | 0  | 24.81 |

|   |  |   |  |   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|---|--|---|--|-------|
| G |  | G |  | G |  | L |  | 0 |  | 0 |  | 24.81 |
| G |  | G |  | G |  | G |  | 9 |  | 0 |  | 28.40 |
| G |  | G |  | G |  | G |  | 0 |  | 0 |  | 28.39 |

## 6.126 FFT-1024

### Функции

- void [FFT\\_Fwd1024Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Fwd1024Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Fwd1024](#) (nm32sc \*GSrcBuffer, nm32sc \*LDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR=-1)  
Прямое быстрое преобразование Фурье-1024.

### 6.126.1 Подробное описание

### 6.126.2 Функции

#### 6.126.2.1 FFT\_Fwd1024()

```
void FFT_Fwd1024 (
    nm32sc * GSrcBuffer,
    nm32sc * LDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR = -1 )
```

Прямое быстрое преобразование Фурье-1024.

Функция выполняет дискретное комплексное 1024-точечное преобразование Фурье на базе алгоритма БПФ по основанию 2-32-16

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 1024 64-р. слов                                                                                                                                                                                                                                                                                                                                                                                       |
| out | LDstBuffer | Результирующий массив размером 1024 64-р. слов                                                                                                                                                                                                                                                                                                                                                                                |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 1024*3 64-р. слов                                                                                                                                                                                                                                                                                                                                                     |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 1024*3 64-р. слов                                                                                                                                                                                                                                                                                                                                                   |
| in  | ShiftR     | Коэффициент нормализации, выполняет арифметический сдвиг результирующего массива на ShiftR бит вправо для получения нормализованного массива LDstBuffer. При передаче значения по умолчанию (-1) ShiftR автоматически принимается равным 14 если ранее установленна точность 7-бит функцией <a href="#">FFT_Fwd1024Set7bit()</a> и 12 - если ранее установлена точность 6-бит функцией <a href="#">FFT_Fwd1024Set6bit()</a> . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | LDstBuffer | LBuffer | GBuffer | ShiftR | clocks |
|------------|------------|---------|---------|--------|--------|
| L          | L          | L       | L       | -1     | 22.55  |
| L          | L          | L       | G       | -1     | 20.52  |
| L          | L          | G       | L       | -1     | 26.08  |
| L          | L          | G       | G       | -1     | 25.93  |
| L          | G          | L       | L       | -1     | 21.53  |
| L          | G          | L       | G       | -1     | 21.48  |
| L          | G          | G       | L       | -1     | 25.06  |
| L          | G          | G       | G       | -1     | 26.89  |
| G          | L          | L       | L       | -1     | 21.55  |
| G          | L          | L       | G       | -1     | 19.51  |
| G          | L          | G       | L       | -1     | 27.03  |
| G          | L          | G       | G       | -1     | 26.87  |
| G          | G          | L       | L       | -1     | 20.52  |
| G          | G          | L       | G       | -1     | 20.48  |
| G          | G          | G       | L       | -1     | 26.00  |
| G          | G          | G       | G       | -1     | 27.84  |
| L          | L          | L       | L       | 0      | 22.55  |
| L          | L          | L       | G       | 0      | 20.51  |
| L          | L          | G       | L       | 0      | 26.08  |
| L          | L          | G       | G       | 0      | 25.92  |
| L          | G          | L       | L       | 0      | 21.52  |
| L          | G          | L       | G       | 0      | 21.48  |
| L          | G          | G       | L       | 0      | 25.05  |
| L          | G          | G       | G       | 0      | 26.89  |
| G          | L          | L       | L       | 0      | 21.54  |
| G          | L          | L       | G       | 0      | 19.51  |
| G          | L          | G       | L       | 0      | 27.02  |
| G          | L          | G       | G       | 0      | 26.87  |
| G          | G          | L       | L       | 0      | 20.52  |
| G          | G          | L       | G       | 0      | 20.47  |
| G          | G          | G       | L       | 0      | 26.00  |
| G          | G          | G       | G       | 0      | 27.83  |

## 6.127 IFFT-1024

### Функции

- void [FFT\\_Inv1024Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Inv1024Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Inv1024](#) ([nm32sc](#) \*GSrcBuffer, [nm32sc](#) \*GDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR1=10, int ShiftR2=-1)  
Обратное быстрое преобразование Фурье. ОБПФ-1024.

### 6.127.1 Подробное описание

### 6.127.2 Функции

#### 6.127.2.1 FFT\_Inv1024()

```
void FFT_Inv1024 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR1 = 10,
    int ShiftR2 = -1 )
```

Обратное быстрое преобразование Фурье. ОБПФ-1024.

Функция выполняет обратное дискретное комплексное 1024-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 2-16-16.

### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                     |
|-----|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 1024 64-р. слов                                                                                                                                                                                                                                                                                                             |
| out | GDstBuffer | Результирующий массив размером 1024 64-р. слов                                                                                                                                                                                                                                                                                                      |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 1024*3 64-р. слов                                                                                                                                                                                                                                                                           |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 1024*3 64-р. слов                                                                                                                                                                                                                                                                         |
| in  | ShiftR1    | Промежуточный сдвиг результатов на ShiftR1 бит вправо (первая нормализация). Необходимо для предотвращения переполнения. По умолчанию равен 10                                                                                                                                                                                                      |
| in  | ShiftR2    | Заключительный сдвиг результатов на ShiftR2 бит вправо (вторая нормализация) в конце вычисления обратного БПФ. По умолчанию ShiftR2 принимается равным 14 при установленной точности 7-бит с помощью функции <a href="#">FFT_Inv1024Set7bit()</a> и 12 - при точности 6-бит, установленной с помощью функции <a href="#">FFT_Inv1024Set6bit()</a> . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | GDstBuffer | LBuffer | GBuffer | ShiftR1 | ShiftR2 | clocks |
|------------|------------|---------|---------|---------|---------|--------|
|------------|------------|---------|---------|---------|---------|--------|

|   |   |   |   |    |    |       |
|---|---|---|---|----|----|-------|
| L | L | L | L | 10 | -1 | 28.82 |
| L | L | L | L | 0  | -1 | 28.82 |
| L | L | L | G | 10 | -1 | 23.90 |
| L | L | L | G | 0  | -1 | 23.90 |
| L | L | G | L | 10 | -1 | 26.60 |
| L | L | G | L | 0  | -1 | 26.60 |
| L | L | G | G | 10 | -1 | 25.55 |
| L | L | G | G | 0  | -1 | 25.55 |
| L | G | L | L | 10 | -1 | 27.80 |
| L | G | L | L | 0  | -1 | 27.80 |
| L | G | L | G | 10 | -1 | 22.88 |
| L | G | L | G | 0  | -1 | 22.88 |
| L | G | G | L | 10 | -1 | 27.57 |
| L | G | G | L | 0  | -1 | 27.57 |
| L | G | G | G | 10 | -1 | 26.52 |
| L | G | G | G | 0  | -1 | 26.52 |
| G | L | L | L | 10 | -1 | 27.81 |
| G | L | L | L | 0  | -1 | 27.81 |
| G | L | L | G | 10 | -1 | 22.89 |
| G | L | L | G | 0  | -1 | 22.89 |
| G | L | G | L | 10 | -1 | 27.55 |
| G | L | G | L | 0  | -1 | 27.55 |
| G | L | G | G | 10 | -1 | 26.50 |
| G | L | G | G | 0  | -1 | 26.50 |
| G | G | L | L | 10 | -1 | 26.79 |
| G | G | L | L | 0  | -1 | 26.79 |
| G | G | L | G | 10 | -1 | 21.87 |
| G | G | L | G | 0  | -1 | 21.87 |
| G | G | G | L | 10 | -1 | 28.51 |
| G | G | G | L | 0  | -1 | 28.51 |
| G | G | G | G | 10 | -1 | 27.46 |
| G | G | G | G | 0  | -1 | 27.46 |
| L | L | L | L | 10 | 0  | 28.82 |
| L | L | L | L | 0  | 0  | 28.82 |
| L | L | L | G | 10 | 0  | 23.90 |
| L | L | L | G | 0  | 0  | 23.90 |
| L | L | G | L | 10 | 0  | 26.60 |
| L | L | G | L | 0  | 0  | 26.60 |
| L | L | G | G | 10 | 0  | 25.54 |
| L | L | G | G | 0  | 0  | 25.54 |
| L | G | L | L | 10 | 0  | 27.79 |
| L | G | L | L | 0  | 0  | 27.79 |
| L | G | L | G | 10 | 0  | 22.87 |
| L | G | L | G | 0  | 0  | 22.87 |
| L | G | G | L | 10 | 0  | 27.56 |
| L | G | G | L | 0  | 0  | 27.56 |
| L | G | G | G | 10 | 0  | 26.51 |
| L | G | G | G | 0  | 0  | 26.51 |
| G | L | L | L | 10 | 0  | 27.81 |
| G | L | L | L | 0  | 0  | 27.81 |
| G | L | L | G | 10 | 0  | 22.89 |
| G | L | L | G | 0  | 0  | 22.89 |
| G | L | G | L | 10 | 0  | 27.54 |
| G | L | G | L | 0  | 0  | 27.54 |
| G | L | G | G | 10 | 0  | 26.49 |
| G | L | G | G | 0  | 0  | 26.49 |
| G | G | L | L | 10 | 0  | 26.78 |
| G | G | L | L | 0  | 0  | 26.78 |
| G | G | L | G | 10 | 0  | 21.86 |
| G | G | L | G | 0  | 0  | 21.86 |
| G | G | G | L | 10 | 0  | 28.51 |



|   |  |   |  |   |  |   |  |    |  |   |  |       |
|---|--|---|--|---|--|---|--|----|--|---|--|-------|
| G |  | G |  | G |  | L |  | 0  |  | 0 |  | 28.51 |
| G |  | G |  | G |  | G |  | 10 |  | 0 |  | 27.46 |
| G |  | G |  | G |  | G |  | 0  |  | 0 |  | 27.46 |

## 6.128 FFT-2048

### Функции

- void [FFT\\_Fwd2048Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Fwd2048Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Fwd2048](#) (nm32sc \*GSrcBuffer, nm32sc \*GDstBuffer, void \*LBuffer, int ShiftR=-1)  
Прямое быстрое преобразование Фурье-2048.

### 6.128.1 Подробное описание

### 6.128.2 Функции

#### 6.128.2.1 FFT\_Fwd2048()

```
void FFT_Fwd2048 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    int ShiftR = -1 )
```

Прямое быстрое преобразование Фурье-2048.

Функция выполняет дискретное комплексное 2048-точечное преобразование Фурье на базе алгоритма БПФ по основанию 2-32-32

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 2048 64-р. слов                                                                                                                                                                                                                                                                                                                                                                                      |
| out | GDstBuffer | Результирующий массив размером 2048 64-р. слов                                                                                                                                                                                                                                                                                                                                                                               |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 2048*4 64-р. слов                                                                                                                                                                                                                                                                                                                                                    |
| in  | ShiftR     | Коэффициент нормализации, выполняет арифметический сдвиг результирующего массива на ShiftR бит вправо для получения нормализованного массива LDstBuffer. При передаче значения по умолчанию (-1) ShiftR автоматически принимается равным 14 если ранее установлена точность 7-бит функцией <a href="#">FFT_Fwd2048Set7bit()</a> и 12 - если ранее установлена точность 6-бит функцией <a href="#">FFT_Fwd2048Set6bit()</a> . |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | GDstBuffer | LBuffer | Shift R | clocks |
|------------|------------|---------|---------|--------|
| L          | L          | L       | -1      | 26.41  |
| L          | L          | G       | -1      | 28.97  |
| L          | G          | L       | -1      | 25.38  |
| L          | G          | G       | -1      | 29.93  |
| G          | L          | L       | -1      | 25.38  |
| G          | L          | G       | -1      | 29.93  |
| G          | G          | L       | -1      | 24.36  |
| G          | G          | G       | -1      | 30.89  |
| L          | L          | L       | 0       | 26.40  |
| L          | L          | G       | 0       | 28.96  |
| L          | G          | L       | 0       | 25.38  |
| L          | G          | G       | 0       | 29.93  |
| G          | L          | L       | 0       | 25.38  |
| G          | L          | G       | 0       | 29.93  |
| G          | G          | L       | 0       | 24.35  |
| G          | G          | G       | 0       | 30.89  |

## 6.129 IFFT-2048

### Функции

- void [FFT\\_Inv2048Set6bit](#) ()  
Устанавливает 6-битную точность вычислений
- void [FFT\\_Inv2048Set7bit](#) ()  
Устанавливает 7-битную точность вычислений
- void [FFT\\_Inv2048](#) ([nm32sc](#) \*GSrcBuffer, [nm32sc](#) \*LDstBuffer, void \*LBuffer, void \*GBuffer, int ShiftR1=11, int ShiftR2=-1)  
Обратное быстрое преобразование Фурье. ОБПФ-2048.

### 6.129.1 Подробное описание

### 6.129.2 Функции

#### 6.129.2.1 FFT\_Inv2048()

```
void FFT_Inv2048 (
    nm32sc * GSrcBuffer,
    nm32sc * LDstBuffer,
    void * LBuffer,
    void * GBuffer,
    int ShiftR1 = 11,
    int ShiftR2 = -1 )
```

Обратное быстрое преобразование Фурье. ОБПФ-2048.

Функция выполняет обратное дискретное комплексное 2048-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 2-32-32.

#### Аргументы

|     |            |                                                                                                                                                                                                                                                                                                                                                       |
|-----|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 2048 64-р. слов                                                                                                                                                                                                                                                                                                               |
| out | LDstBuffer | Результирующий массив размером 2048 64-р. слов                                                                                                                                                                                                                                                                                                        |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 2048*4 64-р. слов                                                                                                                                                                                                                                                                             |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 2048*4 64-р. слов                                                                                                                                                                                                                                                                           |
| in  | ShiftR1    | Промежуточный сдвиг результатов на ShiftR1 бит вправо (первая нормализация). Необходимо для предотвращения переполнения. По умолчанию равен 11                                                                                                                                                                                                        |
| in  | ShiftR2    | Заключительный сдвиг результатов на ShiftR2 бит вправо (вторая нормализация) в конце вычисления обратного БПФ. По умолчанию ShiftR2 принимается равным 14 при установленной точности 7-бит с помощью функции <a href="#">FFT_Inv2048Set7bit</a> () и 12 - при точности 6-бит, установленной с помощью функции <a href="#">FFT_Inv2048Set6bit</a> ()). |

Возвращает

void

Заметки

Использование inplace параметров не допускается ( все указатели должны быть разными)

| GSrcBuffer | LDstBuffer | LBuffer | GBuffer | ShiftR1 | ShiftR2 | clocks |
|------------|------------|---------|---------|---------|---------|--------|
| L          | L          | L       | L       | 11      | -1      | 30.58  |
| L          | L          | L       | L       | 0       | -1      | 30.58  |
| L          | L          | L       | G       | 11      | -1      | 26.52  |
| L          | L          | L       | G       | 0       | -1      | 26.52  |
| L          | L          | G       | L       | 11      | -1      | 31.33  |
| L          | L          | G       | L       | 0       | -1      | 31.33  |
| L          | L          | G       | G       | 11      | -1      | 29.26  |
| L          | L          | G       | G       | 0       | -1      | 29.26  |
| L          | G          | L       | L       | 11      | -1      | 29.56  |
| L          | G          | L       | L       | 0       | -1      | 29.55  |
| L          | G          | L       | G       | 11      | -1      | 27.48  |
| L          | G          | L       | G       | 0       | -1      | 27.48  |
| L          | G          | G       | L       | 11      | -1      | 30.30  |
| L          | G          | G       | L       | 0       | -1      | 30.30  |
| L          | G          | G       | G       | 11      | -1      | 30.22  |
| L          | G          | G       | G       | 0       | -1      | 30.22  |
| G          | L          | L       | L       | 11      | -1      | 29.56  |
| G          | L          | L       | L       | 0       | -1      | 29.56  |
| G          | L          | L       | G       | 11      | -1      | 25.49  |
| G          | L          | L       | G       | 0       | -1      | 25.49  |
| G          | L          | G       | L       | 11      | -1      | 32.29  |
| G          | L          | G       | L       | 0       | -1      | 32.29  |
| G          | L          | G       | G       | 11      | -1      | 30.22  |
| G          | L          | G       | G       | 0       | -1      | 30.22  |
| G          | G          | L       | L       | 11      | -1      | 28.53  |
| G          | G          | L       | L       | 0       | -1      | 28.53  |
| G          | G          | L       | G       | 11      | -1      | 26.46  |
| G          | G          | L       | G       | 0       | -1      | 26.46  |
| G          | G          | G       | L       | 11      | -1      | 31.26  |
| G          | G          | G       | L       | 0       | -1      | 31.26  |
| G          | G          | G       | G       | 11      | -1      | 31.19  |
| G          | G          | G       | G       | 0       | -1      | 31.18  |
| L          | L          | L       | L       | 11      | 0       | 30.58  |
| L          | L          | L       | L       | 0       | 0       | 30.58  |
| L          | L          | L       | G       | 11      | 0       | 26.51  |
| L          | L          | L       | G       | 0       | 0       | 26.51  |
| L          | L          | G       | L       | 11      | 0       | 31.33  |
| L          | L          | G       | L       | 0       | 0       | 31.32  |
| L          | L          | G       | G       | 11      | 0       | 29.25  |
| L          | L          | G       | G       | 0       | 0       | 29.25  |
| L          | G          | L       | L       | 11      | 0       | 29.55  |
| L          | G          | L       | L       | 0       | 0       | 29.55  |
| L          | G          | L       | G       | 11      | 0       | 27.48  |
| L          | G          | L       | G       | 0       | 0       | 27.48  |
| L          | G          | G       | L       | 11      | 0       | 30.30  |
| L          | G          | G       | L       | 0       | 0       | 30.30  |
| L          | G          | G       | G       | 11      | 0       | 30.22  |
| L          | G          | G       | G       | 0       | 0       | 30.22  |
| G          | L          | L       | L       | 11      | 0       | 29.56  |
| G          | L          | L       | L       | 0       | 0       | 29.56  |
| G          | L          | L       | G       | 11      | 0       | 25.49  |
| G          | L          | L       | G       | 0       | 0       | 25.49  |
| G          | L          | G       | L       | 11      | 0       | 32.29  |
| G          | L          | G       | L       | 0       | 0       | 32.29  |
| G          | L          | G       | G       | 11      | 0       | 30.22  |
| G          | L          | G       | G       | 0       | 0       | 30.22  |
| G          | G          | L       | L       | 11      | 0       | 28.53  |
| G          | G          | L       | L       | 0       | 0       | 28.53  |
| G          | G          | L       | G       | 11      | 0       | 26.46  |
| G          | G          | L       | G       | 0       | 0       | 26.46  |
| G          | G          | G       | L       | 11      | 0       | 31.26  |

|   |  |   |  |   |  |   |  |    |  |   |  |       |
|---|--|---|--|---|--|---|--|----|--|---|--|-------|
| G |  | G |  | G |  | L |  | 0  |  | 0 |  | 31.26 |
| G |  | G |  | G |  | G |  | 11 |  | 0 |  | 31.18 |
| G |  | G |  | G |  | G |  | 0  |  | 0 |  | 31.18 |

## 6.130 FFT-4096

### Функции

- void `FFT_Fwd4096` (`nm32sc` \*GSrcBuffer, `nm32sc` \*GDstBuffer, void \*LBuffer, void \*GBuffer)

Прямое быстрое преобразование Фурье-4096.

#### 6.130.1 Подробное описание

#### 6.130.2 Функции

##### 6.130.2.1 FFT\_Fwd4096()

```
void FFT_Fwd4096 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer )
```

Прямое быстрое преобразование Фурье-4096.

Функция выполняет дискретное комплексное 4096-точечное преобразование Фурье на базе алгоритма БПФ по основанию 16-16-16

#### Аргументы

|     |            |                                                                             |
|-----|------------|-----------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 4096 64-р. слов                                     |
| out | GDstBuffer | Результирующий массив размером 4096 64-р. слов                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 4096*2 64-р. слов   |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 4096*3 64-р. слов |

#### Возвращает

void

#### Заметки

Использование inplace параметров не допускается (все указатели должны быть разными) Диапазон входных данных: -4096..4096

\perf

GSrcBuffer | GDstBuffer | LBuffer | GBuffer | Cycles

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | L |  | L |  | L |  | 38.25 |
| L |  | L |  | L |  | G |  | 26.82 |
| L |  | L |  | G |  | L |  | 32.21 |
| L |  | L |  | G |  | G |  | 30.74 |

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | G |  | L |  | L |  | 37.22 |
| L |  | G |  | L |  | G |  | 25.79 |
| L |  | G |  | G |  | L |  | 33.17 |
| L |  | G |  | G |  | G |  | 31.71 |
| G |  | L |  | L |  | L |  | 37.26 |
| G |  | L |  | L |  | G |  | 25.83 |
| G |  | L |  | G |  | L |  | 31.21 |
| G |  | L |  | G |  | G |  | 29.75 |
| G |  | G |  | L |  | L |  | 36.23 |
| G |  | G |  | L |  | G |  | 24.80 |
| G |  | G |  | G |  | L |  | 32.18 |
| G |  | G |  | G |  | G |  | 30.71 |



## 6.131 IFFT-4096

### Функции

- void `FFT_Inv4096` (`nm32sc *GSrcBuffer`, `nm32sc *GDstBuffer`, `void *LBuffer`, `void *GBuffer`)

Обратное быстрое преобразование Фурье. ОБПФ-4096.

#### 6.131.1 Подробное описание

#### 6.131.2 Функции

##### 6.131.2.1 `FFT_Inv4096()`

```
void FFT_Inv4096 (
    nm32sc * GSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer )
```

Обратное быстрое преобразование Фурье. ОБПФ-4096.

Функция выполняет обратное дискретное комплексное 4096-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 16-16-16.

#### Аргументы

|     |            |                                                                             |
|-----|------------|-----------------------------------------------------------------------------|
| in  | GSrcBuffer | Входной массив размером 4096 64-р. слов                                     |
| out | GDstBuffer | Результирующий массив размером 4096 64-р. слов                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 4096*2 64-р. слов   |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 4096*3 64-р. слов |

#### Возвращает

void

#### Заметки

Использование inplace параметров не допускается (все указатели должны быть разными) Диапазон входных данных: -4096..4096

\perf

GSrcBuffer | GDstBuffer | LBuffer | GBuffer | CLOCKS

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | L |  | L |  | L |  | 38.25 |
| L |  | L |  | L |  | G |  | 26.82 |
| L |  | L |  | G |  | L |  | 32.21 |
| L |  | L |  | G |  | G |  | 30.74 |

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | G |  | L |  | L |  | 37.22 |
| L |  | G |  | L |  | G |  | 25.79 |
| L |  | G |  | G |  | L |  | 33.17 |
| L |  | G |  | G |  | G |  | 31.71 |
| G |  | L |  | L |  | L |  | 37.26 |
| G |  | L |  | L |  | G |  | 25.83 |
| G |  | L |  | G |  | L |  | 31.21 |
| G |  | L |  | G |  | G |  | 29.75 |
| G |  | G |  | L |  | L |  | 36.23 |
| G |  | G |  | L |  | G |  | 24.80 |
| G |  | G |  | G |  | L |  | 32.18 |
| G |  | G |  | G |  | G |  | 30.71 |

## 6.132 FFT-8192

### Функции

- void `FFT_Fwd8192` (`nm32sc` \*LSrcBuffer, `nm32sc` \*GDstBuffer, void \*LBuffer, void \*GBuffer)

Прямое быстрое преобразование Фурье-8192.

#### 6.132.1 Подробное описание

#### 6.132.2 Функции

##### 6.132.2.1 FFT\_Fwd8192()

```
void FFT_Fwd8192 (
    nm32sc * LSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer )
```

Прямое быстрое преобразование Фурье-8192.

Функция выполняет дискретное комплексное 8192-точечное преобразование Фурье на базе алгоритма БПФ по основанию 2-16-16-16

#### Аргументы

|     |            |                                                                             |
|-----|------------|-----------------------------------------------------------------------------|
| in  | LSrcBuffer | Входной массив размером 8192 64-р. слов                                     |
| out | GDstBuffer | Результирующий массив размером 8192 64-р. слов                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 8192 64-р. слов     |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 8192*3 64-р. слов |

#### Возвращает

void

#### Заметки

Использование inplace параметров не допускается (все указатели должны быть разными) Диапазон входных данных: -2048..2048

\perf

LSrcBuffer | GDstBuffer | LBuffer | GBuffer | Clocks

|   |   |   |   |       |
|---|---|---|---|-------|
| L | L | L | L | 40.70 |
| L | L | L | G | 28.89 |
| L | L | G | L | 35.55 |
| L | L | G | G | 31.88 |
| L | G | L | L | 39.67 |

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | G |  | L |  | G |  | 27.86 |
| L |  | G |  | G |  | L |  | 36.52 |
| L |  | G |  | G |  | G |  | 32.85 |
| G |  | L |  | L |  | L |  | 40.17 |
| G |  | L |  | L |  | G |  | 29.40 |
| G |  | L |  | G |  | L |  | 35.02 |
| G |  | L |  | G |  | G |  | 32.39 |
| G |  | G |  | L |  | L |  | 39.14 |
| G |  | G |  | L |  | G |  | 28.37 |
| G |  | G |  | G |  | L |  | 35.99 |
| G |  | G |  | G |  | G |  | 33.36 |

## 6.133 IFFT-8192

### Функции

- void `FFT_Inv8192` (`nm32sc` \*LSrcBuffer, `nm32sc` \*GDstBuffer, void \*LBuffer, void \*GBuffer)

Обратное быстрое преобразование Фурье. ОБПФ-8192.

#### 6.133.1 Подробное описание

#### 6.133.2 Функции

##### 6.133.2.1 `FFT_Inv8192()`

```
void FFT_Inv8192 (
    nm32sc * LSrcBuffer,
    nm32sc * GDstBuffer,
    void * LBuffer,
    void * GBuffer )
```

Обратное быстрое преобразование Фурье. ОБПФ-8192.

Функция выполняет обратное дискретное комплексное 8192-точечное быстрое преобразование Фурье на базе алгоритма ОБПФ по онованию 2-16-16-16.

#### Аргументы

|     |            |                                                                             |
|-----|------------|-----------------------------------------------------------------------------|
| in  | LSrcBuffer | Входной массив размером 8192 64-р. слов                                     |
| out | GDstBuffer | Результирующий массив размером 8192 64-р. слов                              |
| in  | LBuffer    | Временный массив на локальной шине (Local Bus) размером 8192 64-р. слов     |
| in  | GBuffer    | Временный массив на глобальной шине (Global Bus) размером 8192*3 64-р. слов |

#### Возвращает

void

#### Заметки

Использование inplace параметров не допускается (все указатели должны быть разными) Диапазон входных данных: -2048..2048

\perf

GSrcBuffer | GDstBuffer | LBuffer | GBuffer | Clocks

|   |  |   |  |   |  |   |  |       |
|---|--|---|--|---|--|---|--|-------|
| L |  | L |  | L |  | L |  | 40.70 |
| L |  | L |  | L |  | G |  | 28.89 |

|   |   |   |   |       |
|---|---|---|---|-------|
| L | L | G | L | 35.55 |
| L | L | G | G | 31.88 |
| L | G | L | L | 39.67 |
| L | G | L | G | 27.86 |
| L | G | G | L | 36.52 |
| L | G | G | G | 32.85 |
| G | L | L | L | 40.17 |
| G | L | L | G | 29.40 |
| G | L | G | L | 35.02 |
| G | L | G | G | 32.39 |
| G | G | L | L | 39.14 |
| G | G | L | G | 28.37 |
| G | G | G | L | 35.99 |
| G | G | G | G | 33.36 |

## 6.134 Свертка

### Группы

- [SIG\\_XCorr](#)

Свертка двух векторов.

### 6.134.1 Подробное описание

## 6.135 Масочная фильтрация

### Группы

- [SIG\\_Median3](#)  
Вычисление медианы трех чисел
- [КИХ-фильтрация](#)  
Одномерная КИХ-фильтрация.

### 6.135.1 Подробное описание



## 6.136 Изменение размеров

### Группы

- [SIG\\_ResampleDown2](#)  
Уменьшение числа отсчетов в двое.
- [SIG\\_ResampleUp3Down2](#)  
Передискретизации сигнала в  $3/2$  раза  
Передискретизации сигнала осуществляется методом Polyphase:
- [SIG\\_CreateResample](#)  
Создание ядра для функции передискретизации SIG\_Resample().  
Функции выделяют память и инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.
- [SIG\\_SetResample](#)  
Создание ядра для функции передискретизации SIG\_Resample().  
Функции инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.
- [SIG\\_Resample\\_perf](#)  
Функции для оценки производительности функций фильтрации SIG\_Resample()  
Функция эквивалентно следующим псевдоинструкциям:

### 6.136.1 Подробное описание

## 6.137 Быстрое преобразование Фурье

### Группы

- [FFT-256](#)
- [IFFT-256](#)
- [FFT-512](#)
- [IFFT-512](#)
- [FFT-1024](#)
- [IFFT-1024](#)
- [FFT-2048](#)
- [IFFT-2048](#)
- [FFT-4096](#)
- [IFFT-4096](#)
- [FFT-8192](#)
- [IFFT-8192](#)

### 6.137.1 Подробное описание

## 6.138 Быстрое преобразование Фурье

### Группы

- [FFTFwdInitAlloc](#)

Функции инициализации структур коэффициентов, необходимых для вычисления прямого БПФ

- [FFTInvInitAlloc](#)

Функции инициализации структур коэффициентов, необходимых для вычисления обратного БПФ

- [DFT-8](#)

Функция для вычисления прямого ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

- [FFT-16](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

- [FFT-32](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

- [FFT-64](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

- [FFT-128](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

- [FFT-256](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

- [FFT-512](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

- [FFT-1024](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 1024 комплексных чисел

- [FFT-2048](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

- [FFT-4096](#)

Функция для вычисления прямого БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

- [IDFT-8](#)

Функция для вычисления обратного ДПФ с плавающей точкой над вектором, состоящим из 8 комплексных чисел

- [IFFT-16](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 16 комплексных чисел

- [IFFT-32](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 32 комплексных чисел

- [IFFT-64](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 64 комплексных чисел

- [IFFT-128](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 128 комплексных чисел

- [IFFT-256](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 256 комплексных чисел

- [IFFT-512](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 512 комплексных чисел

- [IFFT-1024](#)

- [IFFT-2048](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 2048 комплексных чисел

- [IFFT-4096](#)

Функция для вычисления обратного БПФ с плавающей точкой над вектором, состоящим из 4096 комплексных чисел

- [FFT-Common](#)

## Функции

- `int nmppsFFTFree_32fcr (NmppsFFTSpec_32fcr *spec)`

Функция инициализации структуры коэффициентов, необходимых для вычисления обратного БПФ с плавающей точкой над вектором длины от 8 до 2048.

### 6.138.1 Подробное описание

### 6.138.2 Функции

#### 6.138.2.1 nmppsFFTFree\_32fcr()

```
int nmppsFFTFree_32fcr (
    NmppsFFTSpec_32fcr * spec )
```

Функция инициализации структуры коэффициентов, необходимых для вычисления обратного БПФ с плавающей точкой над вектором длины от 8 до 2048.

#### Аргументы

|    |       |                                                                                                          |
|----|-------|----------------------------------------------------------------------------------------------------------|
| in | spec  | двойной указатель на структуру коэффициентов                                                             |
| in | order | размерность БПФ, которое нужно вычислить, например, для БПФ256 этот параметр равен 8 (т.к. $2^8 = 256$ ) |

#### Возвращает

Функция возвращают 0 в случае успешной инициализации и отрицательное число (от -1 и меньше) в случае ошибок

Функция освобождает память, выделенную под коэффициенты, необходимые для вычисления БПФ определенного размера

\param [in] spec структура, содержащая необходимые коэффициенты, для вычисления обратного БПФ определенного размера

Возвращает

Функция возвращают 0 в случае успешной инициализации и отрицательное число (от -1 и меньше) в случае ошибок

## 6.139 SIG\_XCorr

Свертка двух векторов.

## Функции

- void SIG\_XCorr\_32s (nm32s \*pSrcVec, int nSrcVecSize, nm32s \*pKernel, int nKernelSize, nm32s \*pDstVec, void \*pTmpBuf)
- void SIG\_XCorr\_16s32s (nm16s \*pSrcVec, int nSrcVecSize, nm32s \*pKernel, int nKernelSize, nm32s \*pDstVec, void \*pTmpBuf)
- void SIG\_XCorr\_8s32s (nm8s \*pSrcVec, int nSrcVecSize, nm32s \*pKernel, int nKernelSize, nm32s \*pDstVec, void \*pTmpBuf)

## 6.139.1 Подробное описание

Свертка двух векторов.

$$DstVec_i = \sum_{j=0}^{nKernelSize-1} pSrcVec[i+j] \cdot pKernel[j]$$

$$i = \overline{0 \dots nSrcVecSize - nKernelSize + 1}$$

## Аргументы

|         |                                                                          |
|---------|--------------------------------------------------------------------------|
| pSrcVec | Входной вектор.                                                          |
| pKernel | Вектор коэффициентов окна свертки.                                       |
| pTmpBuf | Указатель на временный буффер размера 2*nKernelSize + 32 32-битных слов; |

2\*nKernelSize + 32 32-bit words;

## Аргументы

|             |                                                                                    |
|-------------|------------------------------------------------------------------------------------|
| nKernelSize | Размер окна свертки [1,2,3,4...nSrcVecSize-1].                                     |
| nSrcVecSize | Размер входного вектора в элементах .Размер кратен 8,4 или 2 согласно типу данных. |

## Возвращаемые значения

|         |                                                                                                                             |
|---------|-----------------------------------------------------------------------------------------------------------------------------|
| pDstVec | Результирующий вектор, размером nSrcVecSize-nKernelSize+1. после которого могут записываться еще до 7 незначачих 32p слова. |
|---------|-----------------------------------------------------------------------------------------------------------------------------|

## Заметки

По сути функции осуществляют фильтрацию данных окном свертки.

## 6.140 SIG\_Median3

Вычисление медианы трех чисел

Функции

- `int SIG_Median3 (int a, int b, int c)`
- `uint32b SIG_Median3 (uint32b a, uint32b b, uint32b c)`

### 6.140.1 Подробное описание

Вычисление медианы трех чисел

Аргументы

|   |              |
|---|--------------|
| a | Первое число |
| b | Второе число |
| c | Третье число |

Возвращает

Медианное значение



## 6.141 КИХ-фильтрация

Одномерная КИХ-фильтрация.

### Группы

- [nmppsFIR\\_Xs](#)  
Одномерная КИХ-фильтрация
- [nmppsFIRInit\\_Xs](#)  
Инициализация функции одномерной фильтрации
- [nmppsFIRInitAlloc\\_Xs](#)  
Выделение и инициализация служебной структуры для функции одномерной фильтрации
- [nmppsFIRGetStateSize\\_Xs](#)  
Возвращает размер памяти (в 32р.-словах) необходимый для хранения служебной структуры
- [nmppsFIRFree](#)  
освобождает структуру pState в куче

### 6.141.1 Подробное описание

Одномерная КИХ-фильтрация.

## 6.142 nmppsFIR\_Xs

Одномерная КИХ-фильтрация

Функции

- void nmppsFIR\_8s (nm8s \*pSrc, nm8s \*pDst, int srcSize, NmppsFIRState \*pState)
- void nmppsFIR\_8s16s (nm8s \*pSrc, nm16s \*pDst, int srcSize, NmppsFIRState \*pState)
- void nmppsFIR\_8s32s (nm8s \*pSrc, nm32s \*pDst, int srcSize, NmppsFIRState \*pState)
- void nmppsFIR\_16s (nm16s \*pSrc, nm16s \*pDst, int srcSize, NmppsFIRState \*pState)
- void nmppsFIR\_16s32s (nm16s \*pSrc, nm32s \*pDst, int srcSize, NmppsFIRState \*pState)
- void nmppsFIR\_32s (nm32s \*pSrc, nm32s \*pDst, int srcSize, NmppsFIRState \*pState)

### 6.142.1 Подробное описание

Одномерная КИХ-фильтрация

Аргументы

|     |               |                                                                                                            |
|-----|---------------|------------------------------------------------------------------------------------------------------------|
| in  | pSrc          | Входной вектор                                                                                             |
| in  | srcSize       | Размер входного вектора в элементах. Размер вектора должен быть кратен количеству элементов в 64-р. слове. |
| out | pDst          | Результирующий вектор                                                                                      |
| in  | NmppsFIRState | Служебная структура, содержащая весовые коэффициенты фильтра во внутреннем формате.                        |

Заметки

Инициализация служебной структуры производится соответствующей функцией nmppsFIR←Init\_Xs() или nmppsFIRInitAlloc\_Xs(). Максимальная производительность достигается при размещении pSrc, pDst и pPstate в разных банках памяти .

## 6.143 nmppsFIRInit\_Xs

Инициализация функции одномерной фильтрации

Функции

- int nmppsFIRInit\_8s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)
- int nmppsFIRInit\_8s16s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)
- int nmppsFIRInit\_8s32s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)
- int nmppsFIRInit\_16s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)
- int nmppsFIRInit\_16s32s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)
- int nmppsFIRInit\_32s (NmppsFIRState \*pState, int \*pTaps, int tapsLen)

### 6.143.1 Подробное описание

Инициализация функции одномерной фильтрации

Функция преобразует таблицу весовых коэффициентов окна фильтра во внутренний формат

Аргументы

|     |         |                                                                                                                                                                                                                                 |
|-----|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| in  | pTaps   | Указатель на коэффициенты фильтра                                                                                                                                                                                               |
| in  | tapsLen | Размер окна фильтра. tapsLen=[3,5,7,9....]                                                                                                                                                                                      |
| out | pState  | Указатель на служебную структуру, содержащую весовые коэффициенты фильтра во внутреннем формате. Размер памяти (в 32р.-словах) необходимый для хранения служебной структуры можно получить с помощью функции nmppsFIRGetSize_Xs |

Возвращает

Размер проинициализированной структуры pState в 32р. словах

## 6.144 nmppsFIRInitAlloc\_Xs

Выделение и инициализация служебной структуры для функции одномерной фильтрации

### Функции

- int nmppsFIRInitAlloc\_8s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)
- int nmppsFIRInitAlloc\_8s16s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)
- int nmppsFIRInitAlloc\_8s32s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)
- int nmppsFIRInitAlloc\_16s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)
- int nmppsFIRInitAlloc\_16s32s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)
- int nmppsFIRInitAlloc\_32s (NmppsFIRState \*\*ppState, int \*pTaps, int tapsLen)

### 6.144.1 Подробное описание

Выделение и инициализация служебной структуры для функции одномерной фильтрации

Функция выделяет структуру в куче и преобразует таблицу весовых коэффициентов окна фильтра во внутренний формат

#### Аргументы

|    |         |                                             |
|----|---------|---------------------------------------------|
| in | pTaps   | Указатель на коэффициенты фильтра           |
| in | tapsLen | Размер окна фильтра. nWeights=[3,5,7,9....] |

#### Возвращаемые значения

|       |                                                                                                                    |
|-------|--------------------------------------------------------------------------------------------------------------------|
| [out] | ppState Возвращает указатель на служебную структуру, содержащую весовые коэффициенты фильтра во внутреннем формате |
|-------|--------------------------------------------------------------------------------------------------------------------|

#### Возвращает

Размер проинициализированной структуры pState в 32р. словах

## 6.145 nmppsFIRGetStateSize\_Xs

Возвращает размер памяти (в 32р.-словах) необходимый для хранения служебной структуры

### Функции

- int nmppsFIRGetStateSize\_8s (int tapsLen)
- int nmppsFIRGetStateSize\_8s16s (int tapsLen)
- int nmppsFIRGetStateSize\_8s32s (int tapsLen)
- int nmppsFIRGetStateSize\_16s (int tapsLen)
- int nmppsFIRGetStateSize\_16s32s (int tapsLen)
- int nmppsFIRGetStateSize\_32s (int tapsLen)

### 6.145.1 Подробное описание

Возвращает размер памяти (в 32р.-словах) необходимый для хранения служебной структуры

#### Аргументы

|    |         |                                            |
|----|---------|--------------------------------------------|
| in | tapsLen | Размер окна фильтра. tapsLen=[3,5,7,9....] |
|----|---------|--------------------------------------------|

#### Возвращает

Возвращает размер памяти (в 32р.-словах), необходимый для хранения служебной структуры  
NmppsFIRState

## 6.146 nmppsFIRFree

освобождает структуру pState в куче

### Функции

- void nmppsFIRFree (NmppsFIRState \*pState)

### 6.146.1 Подробное описание

освобождает структуру pState в куче

### Аргументы

|    |        |                                                |
|----|--------|------------------------------------------------|
| in | pState | указатель на служебную структуру NmppsFIRState |
|----|--------|------------------------------------------------|

## 6.147 SIG\_ResampleDown2

Уменьшение числа отсчетов в двое.

### Функции

- void SIG\_ResampleDown2\_8u (nm8u7b \*pSrcVec, nm8u7b \*pDstVec, int nSrcVecSize, nm64s \*pKernel)
- void SIG\_ResampleDown2\_16u (nm16u15b \*pSrcVec, nm16u15b \*pDstVec, int nSrcVecSize, nm64s \*pKernel)

### 6.147.1 Подробное описание

Уменьшение числа отсчетов в двое.

$$pDstVec = \frac{1}{2} (pSrcVec(2 * x) + pSrcVec(2 * x + 1))$$

### Аргументы

|         |                                |
|---------|--------------------------------|
| pSrcVec | Входной сигнал.                |
| nSize   | Размер массива входных данных. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий сигнал. |
|---------|------------------------|

### Заметки

Для того чтобы избежать переполнения при усреднении, динамический диапазон исходного изображения должен принадлежать диапазону, определенному типом.

## 6.148 SIG\_ResampleUp3Down2

Передискретизации сигнала в 3/2 раза

Передискретизации сигнала осуществляется методом Polyphase:

.

### Функции

- void SIG\_ResampleUp3Down2 (nm8s \*pSrcVec, nm16s \*pDstVec, int nSrcVecSize, nm64s \*pKernel)

### 6.148.1 Подробное описание

Передискретизации сигнала в 3/2 раза

Передискретизации сигнала осуществляется методом Polyphase:

.

- Между отсчетами входного сигнала вставляется по 2 нуля
- Полученный сигнал пропускается через фильтр ФНЧ. Длина фильтра 17
- Из полученного сигнала выбирается каждый 2 отсчет

### Аргументы

|             |                                                           |
|-------------|-----------------------------------------------------------|
| pSrcVec     | Входной вектор. Элементы вектора - целые числа со знаком. |
| nSrcVecSize | Размер входного вектора.                                  |

### Возвращаемые значения

|         |                                                                                                                         |
|---------|-------------------------------------------------------------------------------------------------------------------------|
| pDstVec | Результирующий вектор. Элементы вектора возвращаются в формате fixed-point: [12.4] (целая часть-12 бит, дробная -4бита) |
|---------|-------------------------------------------------------------------------------------------------------------------------|

### Возвращает

void



## 6.149 SIG\_CreateResample

Создание ядра для функции передискретизации SIG\_Resample().

Функции выделяют память и инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.

### Функции

- void SIG\_CreateResampleUp3Down2\_8s16s (nm64s \*\*pKernel, int nHint)
- void SIG\_CreateResampleDown2\_8u8u (nm64s \*\*pKernel, int nHint)
- void SIG\_CreateResampleDown2\_16u16u (nm64s \*\*pKernel, int nHint)

### 6.149.1 Подробное описание

Создание ядра для функции передискретизации SIG\_Resample().

Функции выделяют память и инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.

### Аргументы

|       |                                                                                                                |
|-------|----------------------------------------------------------------------------------------------------------------|
| nHint | Определяет память(Local или Global) в которой создается служебная структура.<br>nHint=[MEM_LOCAL, MEM_GLOBAL]. |
|-------|----------------------------------------------------------------------------------------------------------------|

### Возвращаемые значения

|         |                                                                                                  |
|---------|--------------------------------------------------------------------------------------------------|
| pKernel | Указатель на служебную структуру, содержащую весовые коэффициенты фильтра во внутреннем формате. |
|---------|--------------------------------------------------------------------------------------------------|

### Заметки

Используется перед вызовом функции SIG\_Filter.

## 6.150 SIG\_SetResample

Создание ядра для функции передискретизации SIG\_Resample().

Функции инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.

### Функции

- int SIG\_SetResampleUp3Down2\_8s16s (nm64s \*pKernel)
- int SIG\_SetResampleDown2\_8u8u (nm64s \*pKernel)
- int SIG\_SetResampleDown2\_16u16u (nm64s \*pKernel)

### 6.150.1 Подробное описание

Создание ядра для функции передискретизации SIG\_Resample().

Функции инициализируют таблицы весовых коэффициентов для использования в функциях передискретизации.

### Аргументы

|         |                                                                                                  |
|---------|--------------------------------------------------------------------------------------------------|
| pKernel | Указатель на служебную структуру, содержащую весовые коэффициенты фильтра во внутреннем формате. |
|---------|--------------------------------------------------------------------------------------------------|

### Возвращает

Размер проинициализированной структуры pKernel в 32р. словах

### Заметки

Используется перед вызовом функции SIG\_Filter.

Используется перед вызовом функции SIG\_Resample().

## 6.151 SIG\_Resample\_perf

Функции для оценки производительности функций фильтрации SIG\_Resample()

Функция эквивалентно следующим псевдоинструкциям:

Функции

- void SIG\_ResampleUp3Down2\_perf (nm8s \*pSrcVec, nm16s \*pDstVec, int nSrcVecSize, nm64s \*pKernel)
- void SIG\_ResampleDown2\_perf\_8u (nm8u7b \*pSrcVec, nm8u7b \*pDstVec, int nSrcVecSize, nm64s \*pKernel)
- void SIG\_ResampleDown2\_perf\_16u (nm16u15b \*pSrcVec, nm16u15b \*pDstVec, int nSrcVecSize, nm64s \*pKernel)

### 6.151.1 Подробное описание

Функции для оценки производительности функций фильтрации SIG\_Resample()

Функция эквивалентно следующим псевдоинструкциям:

```
void SIG_Resample***_perf(nm16u15b* pSrcVec, nm16u15b* pDstVec, int nSrcVecSize,
nm64s* pKernel);
{
    SIG_SetResample***_(pKernel);
    clock_t t0,t1;
    t0=clock();
    SIG_Resample***(pSrcVec, pDstVec, nSize,pKernel);
    t1=clock();
    exit(t1-t0);
}
```

\retval pKernel

Указатель на служебную структуру, содержащую весовые коэффициенты фильтра во внутреннем формате.

Аргументы

|             |                                      |
|-------------|--------------------------------------|
| pSrcVec     | Входной вектор.                      |
| nSrcVecSize | Размер входного вектора в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Заметки

Инициализация служебной структуры производится соответствующей функцией SIG\_Set↵Filter() и SIG\_CreateFilter().

## 6.152 Типы векторных данных

### Классы

- struct [v16nm4s](#)
- struct [v4nm8s](#)
- struct [s\\_v8nm8s](#)
- struct [s\\_v16nm8s](#)
- struct [s\\_v4nm16s](#)
- struct [s\\_v8nm16s](#)
- struct [s\\_v16nm16s](#)
- struct [s\\_v2nm32s](#)
- struct [s\\_v4nm32s](#)
- struct [s\\_v8nm32s](#)
- struct [s\\_v16nm32s](#)
- struct [s\\_v16nm4u](#)
- struct [s\\_v4nm8u](#)
- struct [s\\_v8nm8u](#)
- struct [s\\_v16nm8u](#)
- struct [s\\_v4nm16u](#)
- struct [s\\_v8nm16u](#)
- struct [s\\_v16nm16u](#)
- struct [s\\_v2nm32u](#)
- struct [s\\_v4nm32u](#)
- struct [s\\_v8nm32u](#)
- struct [s\\_v16nm32u](#)

### Определения типов

- typedef void [nm1](#)
- typedef void [nm2s](#)
- typedef void [nm4s](#)
- typedef void [nm8s](#)
- typedef signed char [nm8s7b](#)
- typedef void [nm16s](#)
- typedef [nm16s](#) [nm16s15b](#)
- typedef int [nm32s](#)
- typedef int [nm32s31b](#)
- typedef int [nm32s30b](#)
- typedef long long [nm64s](#)
- typedef [nm64s](#) [nm64s63b](#)
- typedef void [nm2u](#)
- typedef void [nm4u](#)
- typedef [nm4u](#) [nm4u3b](#)
- typedef void [nm8u](#)
- typedef unsigned char [nm8u7b](#)
- typedef void [nm16u](#)
- typedef [nm16u](#) [nm16u15b](#)
- typedef unsigned int [nm32u](#)
- typedef unsigned int [nm32u31b](#)
- typedef unsigned long long [nm64u](#)
- typedef struct [s\\_v8nm8s](#) [v8nm8s](#)
- typedef struct [s\\_v16nm8s](#) [v16nm8s](#)

- typedef struct [s\\_v4nm16s](#) v4nm16s
- typedef struct [s\\_v8nm16s](#) v8nm16s
- typedef struct [s\\_v16nm16s](#) v16nm16s
- typedef struct [s\\_v2nm32s](#) v2nm32s
- typedef struct [s\\_v4nm32s](#) v4nm32s
- typedef struct [s\\_v8nm32s](#) v8nm32s
- typedef struct [s\\_v16nm32s](#) v16nm32s
- typedef [v16nm8s](#) v16nm8s7b
- typedef struct [s\\_v16nm4u](#) v16nm4u
- typedef struct [s\\_v4nm8u](#) v4nm8u
- typedef struct [s\\_v8nm8u](#) v8nm8u
- typedef struct [s\\_v16nm8u](#) v16nm8u
- typedef struct [s\\_v4nm16u](#) v4nm16u
- typedef struct [s\\_v8nm16u](#) v8nm16u
- typedef struct [s\\_v16nm16u](#) v16nm16u
- typedef struct [s\\_v2nm32u](#) v2nm32u
- typedef struct [s\\_v4nm32u](#) v4nm32u
- typedef struct [s\\_v8nm32u](#) v8nm32u
- typedef struct [s\\_v16nm32u](#) v16nm32u
- typedef [v16nm4u](#) v16nm4b3u

### 6.152.1 Подробное описание

В данном разделе описываются типы векторных данных с которыми могут работать функции библиотеки, задействующие векторный узел. Также рассматриваются соглашения о передаче параметров.

Поскольку векторный узел работает с данными произвольной разрядности, упакованными в 64-разрядные слова, то это накладывает следующие ограничения на работу с массивами данных и их типами:

1. Указатель на векторные данные всегда является четным адресом. Т.е. выровнен в памяти по границе 64р. слов.
2. Размер массива, передаваемый на вход функций, как правило, исчисляется в отдельных элементах, составляющих этот массив.  
Кратность этого размера по умолчанию определяется кол-вом чисел, упакованных в 64р. слово.  
Например:  
для nm8s кратность-8  
для nm16s кратность-4  
для nm32s кратность-2  
для nm64s кратность-1  
Если в описании указаны другие условия кратности, как например [32,64,96,128...], то это означает, что допустимые размеры могут только из этого диапазона с кратностью 32.
3. Типы nm8s , nm16s, nm32s... хоть и созданы для обозначения разрядности упакованных данных, но с точки зрения C++ таковыми не являются , так как определяются через typedef как производные от стандартных типов char, short и int, которые все три в свою очередь являются 32-разрядными типами в архитектуре NeuroMatrix. Поэтому эти векторные типы можно использовать только с оператором \* (nm8s\*,nm16s\*,... ). Операции же sizeof() к массивам этих типов будут выполняться некорректно.

Расшифровка мнемоники в названии типа:

1. Префикс `nm` - означает что данные являются векторными ,упакованными в 64р слова (`nm8s,nm8u,nm16s....` ).
2. Разрядность данных указывается после префикса `nm` (`nm8s,nm8u` - байтовые массивы, `nm16s,nm16u` - 16р. массивы ).
3. суффикс `s` или `u` оначает знаковый или беззнаковый тип данных.
4. Для работы некоторых функций во избежании переполнения требуется суженный диапазон возможных значений, чем позволяет разрядность. Такие данные имеют суффикс в виде кол-ва значащих бит в слове и символом `b`. (`nm8s7b`)

## 6.152.2 Типы

### 6.152.2.1 `nm1`

```
typedef void nm1
```

Большинство функций библиотеки получают и возвращают массивы упакованных чисел. Обращение к элементам данных массивов должно производиться с помощью функций доступа к элементам `Getval()` and `Setval()`.

Тип характеризует векторные данные как массив одноразрядных чисел. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 64.

Диапазон значений:

$[-1, 0]$

См. определение в файле `nmtype.h` строка 96

### 6.152.2.2 `nm16s`

```
typedef void nm16s
```

Тип характеризует векторные данные как массив 16-ти разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 4.

Диапазон значений:

$[-2^{15}, \dots, +2^{15} - 1]$ .

См. определение в файле `nmtype.h` строка 210

## 6.152.2.3 nm16s15b

```
typedef nm16s nm16s15b
```

Тип характеризует векторные данные как массив 16-ти разрядных чисел со знаком с ограниченным диапазоном принимаемых значений.

Начальный адрес массива должен быть выровнен по границе 64р слова.

Предполагается , что размер массива данного типа как минимум кратен 4.

Диапазон значений:

$$[-2^{14}, \dots, +2^{14} - 1]$$

См. определение в файле nmtype.h строка 232

## 6.152.2.4 nm16u

```
typedef void nm16u
```

Тип характеризует векторные данные как массив 16-ти разрядных чисел без знака.

Начальный адрес массива должен быть выровнен по границе 64р слова.

Предполагается , что размер массива данного типа как минимум кратен 4.

Диапазон значений:

$$[0, \dots, 2^{16} - 1].$$

См. определение в файле nmtype.h строка 445

## 6.152.2.5 nm16u15b

```
typedef nm16u nm16u15b
```

Тип характеризует векторные данные как массив 16-ти разрядных чисел без знака.

Начальный адрес массива должен быть выровнен по границе 64р слова.

Предполагается , что размер массива данного типа как минимум кратен 4.

Диапазон значений:

$$[-2^{14}, \dots, +2^{14} - 1].$$

См. определение в файле nmtype.h строка 461

## 6.152.2.6 nm2s

```
typedef void nm2s
```

Тип характеризует векторные данные как массив 2-х разрядных чисел со знаком.  
Начальный адрес массива должен быть выровнен по границе 64-бита слова.  
Предполагается, что размер массива данного типа как минимум кратен 32.

Диапазон значений:

$$[-2^1, \dots, +2^1 - 1] = [-2, \dots, +1]$$

См. определение в файле nmtypes.h строка 114

## 6.152.2.7 nm2u

```
typedef void nm2u
```

Тип характеризует векторные данные как массив 2-х разрядных чисел без знака.  
Начальный адрес массива должен быть выровнен по границе 64-бита слова.  
Предполагается, что размер массива данного типа как минимум кратен 32.

Диапазон значений:

$$[0, \dots, +2^2 - 1] = [0, \dots, 3]$$

См. определение в файле nmtypes.h строка 370

## 6.152.2.8 nm32s

```
typedef int nm32s
```

Тип характеризует векторные данные как массив 32-х разрядных чисел со знаком.  
Начальный адрес массива должен быть выровнен по границе 64-бита слова.  
Предполагается, что размер массива данного типа как минимум кратен 2.

Диапазон значений:

$$[-2^{31}, \dots, +2^{31} - 1].$$

См. определение в файле nmtypes.h строка 258



## 6.152.2.9 nm32s30b

```
typedef int nm32s30b
```

Тип характеризует векторные данные как массив 32-х разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 2.

Диапазон значений:

$$[-2^{29}, \dots, 2^{29} - 1].$$

См. определение в файле nmtype.h строка 320

## 6.152.2.10 nm32s31b

```
typedef int nm32s31b
```

Тип характеризует векторные данные как массив 32-х разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 2.

Диапазон значений:

$$[-2^{30}, \dots, 2^{30} - 1].$$

См. определение в файле nmtype.h строка 300

## 6.152.2.11 nm32u

```
typedef unsigned int nm32u
```

Тип характеризует векторные данные как массив 32-х разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 2.

Диапазон значений

$$[0, \dots, 2^{32} - 1].$$

См. определение в файле nmtype.h строка 475

## 6.152.2.12 nm32u31b

```
typedef unsigned int nm32u31b
```

Тип характеризует векторные данные как массив 32-х разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64-р слова. Предполагается, что размер массива данного типа как минимум кратен 2.

Диапазон значений

$$[0, \dots, 2^{31} - 1].$$

См. определение в файле `nmtype.h` строка 489

## 6.152.2.13 nm4s

```
typedef void nm4s
```

Тип характеризует векторные данные как массив 4-х разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64-р слова. Предполагается, что размер массива данного типа как минимум кратен 16.

Диапазон значений:

$$[-2^3, \dots, +2^3 - 1] = [-8, \dots, +7]$$

См. определение в файле `nmtype.h` строка 127

## 6.152.2.14 nm4u

```
typedef void nm4u
```

Тип характеризует векторные данные как массив 4-х разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64-р слова. Предполагается, что размер массива данного типа как минимум кратен 16.

Диапазон значений:

$$[0, \dots, +2^4 - 1] = [0, \dots, 15]$$

См. определение в файле `nmtype.h` строка 383

## 6.152.2.15 nm4u3b

```
typedef nm4u nm4u3b
```

Тип характеризует векторные данные как массив 4-х разрядных чисел без знака.  
Начальный адрес массива должен быть выровнен по границе 64р слова.  
Предполагается , что размер массива данного типа как минимум кратен 16.

Диапазон значений:

$$[0, \dots, +2^3 - 1] = [0, \dots, 7]$$

См. определение в файле nmtype.h строка 396

## 6.152.2.16 nm64s

```
typedef long long nm64s
```

Тип характеризует векторные данные как массив 64-х разрядных чисел со знаком.  
Начальный адрес массива должен быть выровнен по границе 64р слова.  
По умолчанию размер массива произвольный .

Диапазон значений:

$$[-2^{63}, \dots, +2^{63} - 1]$$

См. определение в файле nmtype.h строка 341

## 6.152.2.17 nm64s63b

```
typedef nm64s nm64s63b
```

Тип характеризует векторные данные как массив 64-х разрядных чисел со знаком.  
Начальный адрес массива должен быть выровнен по границе 64р слова.  
По умолчанию размер массива произвольный .

Диапазон значений:

$$[-2^{62}, \dots, +2^{62} - 1]$$

См. определение в файле nmtype.h строка 356

## 6.152.2.18 nm64u

```
typedef unsigned long long nm64u
```

Тип характеризует векторные данные как массив 64-х разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64-х слова. По умолчанию размер массива произвольный.

Диапазон значений

$$[0, \dots, 2^{64} - 1].$$

См. определение в файле nmtype.h строка 502

## 6.152.2.19 nm8s

```
typedef void nm8s
```

Тип характеризует векторные данные как массив 8-ми разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64-х слова. Предполагается, что размер массива данного типа как минимум кратен 8.

Диапазон значений:

$$[-2^7, \dots, +2^7 - 1] = [-128, \dots, +127]$$

См. определение в файле nmtype.h строка 150

## 6.152.2.20 nm8s7b

```
typedef signed char nm8s7b
```

Тип характеризует векторные данные как массив 8-ми разрядных чисел со знаком. Начальный адрес массива должен быть выровнен по границе 64-х слова. Предполагается, что размер массива данного типа как минимум кратен 8.

Диапазон значений:

$$[-2^6, \dots, +2^6 - 1] = [-64, \dots, +63]$$

См. определение в файле nmtype.h строка 179

## 6.152.2.21 nm8u

```
typedef void nm8u
```

Тип характеризует векторные данные как массив 8-ми разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 8.

Диапазон значений:

$$[0, \dots, +2^8 - 1] = [0, \dots, 255]$$

См. определение в файле nmtype.h строка 412

## 6.152.2.22 nm8u7b

```
typedef unsigned char nm8u7b
```

Тип характеризует векторные данные как массив 8-ми разрядных чисел без знака. Начальный адрес массива должен быть выровнен по границе 64р слова. Предполагается , что размер массива данного типа как минимум кратен 8.

Диапазон значений:

$$[0, \dots, +2^7 - 1] = [0, \dots, 127]$$

См. определение в файле nmtype.h строка 429

## 6.152.2.23 v16nm16s

```
typedef struct s_v16nm16s v16nm16s
```

Тип векторной структуры, состоящей из 16-ти 16р. чисел со знаком.

## 6.152.2.24 v16nm16u

```
typedef struct s_v16nm16u v16nm16u
```

Тип векторной структуры, состоящей из 16-ти 16р. чисел без знака.

## 6.152.2.25 v16nm32s

```
typedef struct s_v16nm32s v16nm32s
```

Тип векторной структуры, состоящей из 16-ти 32р. чисел со знаком.

6.152.2.26 `v16nm32u`

```
typedef struct s\_v16nm32u v16nm32u
```

Тип векторной структуры, состоящей из 16-ти 32р. чисел без знака.

6.152.2.27 `v16nm4b3u`

```
typedef v16nm4u v16nm4b3u
```

Тип векторной структуры, состоящей из 16-ти 32р. чисел со знаком.

Диапазон значений элементов структуры:

$[0, \dots, 7]$

См. определение в файле `nmtype.h` строка 1222

6.152.2.28 `v16nm4u`

```
typedef struct s\_v16nm4u v16nm4u
```

Тип векторной структуры, состоящей из 16-ти 4-р. чисел без знака.

6.152.2.29 `v16nm8s`

```
typedef struct s\_v16nm8s v16nm8s
```

Тип векторной структуры, состоящей из 16-ти 8р. чисел со знаком.

6.152.2.30 `v16nm8s7b`

```
typedef v16nm8s v16nm8s7b
```

Тип векторной структуры, состоящей из 16-ти 32р. чисел со знаком.

Диапазон значений элементов структуры:

$[-64, \dots, +63]$

См. определение в файле `nmtype.h` строка 1084

6.152.2.31 `v16nm8u`

```
typedef struct s\_v16nm8u v16nm8u
```

Тип векторной структуры, состоящей из 16-ти 8р. чисел без знака.

6.152.2.32 `v2nm32s`

```
typedef struct s\_v2nm32s v2nm32s
```

Тип векторной структуры, состоящей из 2-х 32р. чисел со знаком.

6.152.2.33 `v2nm32u`

```
typedef struct s\_v2nm32u v2nm32u
```

Тип векторной структуры, состоящей из 2-х 32р. чисел без знака.

6.152.2.34 `v4nm16s`

```
typedef struct s\_v4nm16s v4nm16s
```

Тип векторной структуры, состоящей из 4-х 16р. чисел со знаком.

6.152.2.35 `v4nm16u`

```
typedef struct s\_v4nm16u v4nm16u
```

Тип векторной структуры, состоящей из 4-х 16р. чисел без знака.

6.152.2.36 `v4nm32s`

```
typedef struct s\_v4nm32s v4nm32s
```

Тип векторной структуры, состоящей из 4-х 32р. чисел со знаком.

6.152.2.37 `v4nm32u`

```
typedef struct s\_v4nm32u v4nm32u
```

Тип векторной структуры, состоящей из 4-х 32р. чисел без знака.

6.152.2.38 `v4nm8u`

```
typedef struct s\_v4nm8u v4nm8u
```

Тип векторной структуры, состоящей из 4-х 8р. чисел без знака.

6.152.2.39 `v8nm16s`

```
typedef struct s\_v8nm16s v8nm16s
```

Тип векторной структуры, состоящей из 8-ми 16р. чисел со знаком.

6.152.2.40 `v8nm16u`

```
typedef struct s\_v8nm16u v8nm16u
```

Тип векторной структуры, состоящей из 8-ми 16р. чисел без знака.

6.152.2.41 `v8nm32s`

```
typedef struct s\_v8nm32s v8nm32s
```

Тип векторной структуры, состоящей из 8-ми 32р. чисел со знаком.

6.152.2.42 `v8nm32u`

```
typedef struct s\_v8nm32u v8nm32u
```

Тип векторной структуры, состоящей из 8-ми 32р. чисел без знака.

6.152.2.43 `v8nm8s`

```
typedef struct s\_v8nm8s v8nm8s
```

Тип векторной структуры, состоящей из 8-ми 8р. чисел со знаком.

6.152.2.44 `v8nm8u`

```
typedef struct s\_v8nm8u v8nm8u
```

Тип векторной структуры, состоящей из 8-ми 8р. чисел без знака.



## 6.153 Типы скалярных данных

### Определения типов

- typedef int [int1b](#)
- typedef int [int2b](#)
- typedef int [int3b](#)
- typedef int [int4b](#)
- typedef int [int7b](#)
- typedef int [int8b](#)
- typedef int [int15b](#)
- typedef int [int16b](#)
- typedef int [int30b](#)
- typedef int [int31b](#)
- typedef int [int32b](#)
- typedef INT64 [int63b](#)
- typedef INT64 [int64b](#)
- typedef unsigned int [uint1b](#)
- typedef unsigned int [uint2b](#)
- typedef unsigned int [uint3b](#)
- typedef unsigned int [uint4b](#)
- typedef unsigned int [uint7b](#)
- typedef unsigned int [uint8b](#)
- typedef unsigned int [uint15b](#)
- typedef unsigned int [uint16b](#)
- typedef unsigned int [uint31b](#)
- typedef unsigned int [uint32b](#)
- typedef UINT64 [uint63b](#)
- typedef [nm64u](#) [uint64b](#)

### 6.153.1 Подробное описание

Назначением данной библиотеки является предоставление базовых операций по обработке одномерных массивов (векторов) для процессоров семейства NeroMatrix.

В состав библиотеки входят логические и арифметические функции, операции сравнения, инициализации, копирования, преобразования разрядностей и т.п. Библиотека предназначена для быстрой разработки эффективных пользовательских программ как на языке высокого уровня(C++), так и на языке ассемблера с помощью прилагаемой библиотеки ядра низкоуровневых функций. Функции библиотеки имеют C++ интерфейс.

Большинство функций библиотеки реализованы на языке ассемблера с использованием векторных инструкций и оптимизированы под архитектуру процессоров семейства NeuroMatrix. Для удобства разработки прикладных программ библиотека содержит аналогичные реализации функций для процессоров серии x86, выполненных на языке C++. Данные реализации позволяют выполнять написанные с использованием данной библиотеки прикладные программы на персональном компьютере.

### 6.153.2 Типы

## 6.153.2.1 int15b

```
typedef int int15b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{14}, \dots, +2^{14} - 1]$$

См. определение в файле nmtypes.h строка 597

## 6.153.2.2 int16b

```
typedef int int16b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{15}, \dots, +2^{15} - 1]$$

См. определение в файле nmtypes.h строка 610

## 6.153.2.3 int1b

```
typedef int int1b
```

Тип для 32-разрядных скалярных переменных с ограниченным диапазоном значений.

Диапазон значений:

$$[-1, 0]$$

См. определение в файле nmtypes.h строка 519

## 6.153.2.4 int2b

```
typedef int int2b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^1, \dots, +2^1 - 1] = [-2, \dots, +1]$$

См. определение в файле nmtypes.h строка 532

## 6.153.2.5 int30b

```
typedef int int30b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{29}, \dots, +2^{29} - 1]$$

См. определение в файле nmtypes.h строка 623

## 6.153.2.6 int31b

```
typedef int int31b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{30}, \dots, +2^{30} - 1]$$

См. определение в файле nmtypes.h строка 636

## 6.153.2.7 int32b

```
typedef int int32b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{31}, \dots, +2^{31} - 1]$$

См. определение в файле nmtypes.h строка 649

## 6.153.2.8 int3b

```
typedef int int3b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^2, \dots, +2^2 - 1] = [-4, \dots, +3]$$

См. определение в файле nmtypes.h строка 545

## 6.153.2.9 int4b

```
typedef int int4b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^3, \dots, +2^3 - 1] = [-8, \dots, +7]$$

См. определение в файле nmtypes.h строка 558

## 6.153.2.10 int63b

```
typedef INT64 int63b
```

Тип для 64-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{62}, \dots, +2^{62} - 1]$$

См. определение в файле nmtypes.h строка 662

## 6.153.2.11 int64b

```
typedef INT64 int64b
```

Тип для 64-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^{63}, \dots, +2^{63} - 1]$$

См. определение в файле nmtypes.h строка 675

## 6.153.2.12 int7b

```
typedef int int7b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[-2^6, \dots, +2^6 - 1] = [-64, \dots, +63]$$

См. определение в файле nmtypes.h строка 571

## 6.153.2.13 int8b

```
typedef int int8b
```

Тип для 32-разрядных скалярных переменных с ограниченным диапазоном значений.

Диапазон значений:

$$[-2^7, \dots, +2^7 - 1] = [-128, \dots, +127]$$

См. определение в файле nmtypes.h строка 584

## 6.153.2.14 uint15b

```
typedef unsigned int uint15b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{15} - 1]$$

См. определение в файле nmtypes.h строка 766

## 6.153.2.15 uint16b

```
typedef unsigned int uint16b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{16} - 1]$$

См. определение в файле nmtypes.h строка 779

## 6.153.2.16 uint1b

```
typedef unsigned int uint1b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, 1] = [0, 1]$$

См. определение в файле nmtypes.h строка 688

## 6.153.2.17 uint2b

```
typedef unsigned int uint2b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^2 - 1] = [0, \dots, 3]$$

См. определение в файле nmtypes.h строка 701

## 6.153.2.18 uint31b

```
typedef unsigned int uint31b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{31} - 1]$$

См. определение в файле nmtypes.h строка 792

## 6.153.2.19 uint32b

```
typedef unsigned int uint32b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{32} - 1]$$

См. определение в файле nmtypes.h строка 805

## 6.153.2.20 uint3b

```
typedef unsigned int uint3b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^3 - 1] = [0, \dots, 7]$$

См. определение в файле nmtypes.h строка 714

## 6.153.2.21 uint4b

```
typedef unsigned int uint4b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^4 - 1] = [0, \dots, 15]$$

См. определение в файле `nmtype.h` строка 727

## 6.153.2.22 uint63b

```
typedef UINT64 uint63b
```

Тип для 64-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{63} - 1]$$

См. определение в файле `nmtype.h` строка 818

## 6.153.2.23 uint64b

```
typedef nm64u uint64b
```

Тип для 64-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^{64} - 1]$$

См. определение в файле `nmtype.h` строка 831

## 6.153.2.24 uint7b

```
typedef unsigned int uint7b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^7 - 1] = [0, \dots, 127]$$

См. определение в файле `nmtype.h` строка 740

## 6.153.2.25 uint8b

```
typedef unsigned int uint8b
```

Тип для 32-разрядных скалярных переменных с ограниченным допустимым диапазоном значений.

Диапазон значений:

$$[0, \dots, 2^8 - 1] = [0, \dots, 255]$$

См. определение в файле `nmtype.h` строка 753

## 6.154 Функции поддержки

### Группы

- [nmppsMalloc](#)  
Распределение памяти для векторов библиотеки.
- [nmppsFree](#)  
Освобождение памяти для векторов.
- [nmppsAddr\\_](#)  
Возвращает адрес ячейки памяти, содержащей указанный элемент.  
Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.
- [nmppsSetVal\\_](#)  
Модификация элемента вектора.
- [nmppsGetVal\\_](#)  
Извлекает значение элемента вектора.
- [nmppsGetVal\\_\(return\)](#)  
Извлекает значение элемента вектора.

### 6.154.1 Подробное описание



## 6.155 Инициализация и копирование

### Группы

- [nmppsSet-инициализация](#)

Функция инициализации элементов массива постоянным значением.

- [nmppsRandUniform](#)

Инициализация массива случайными числами.

- [nmppsRandUniform\\_](#)

Генерация случайного числа с равномерным распределением.

- [nmppsRamp\\_](#)

Инициализация массива элементами арифметической прогрессии.

- [nmppsConvert](#)

Изменение разрядности элементов вектора.

Преобразование знаковых данных к меньшей разрядности осуществляется отбрасыванием старших битов. Преобразование знаковых данных к большей разрядности осуществляется с распространением влево старшего (знакового) бита. Преобразование беззнаковых данных к большей разрядности осуществляется добавлением слева старших нулевых битов.

- [nmppsCopy\\_](#)

Копирование вектора.

- [nmppsCopyua\\_](#)

Копирование вектора с невыровненной байтовой позиции в выровненную.

- [nmppsSwap\\_](#)

Перестановка двух векторов.

- [Vec\\_ClipRShiftConvert\\_AddC](#)

Изменение разрядности элементов вектора с клиппированием

Преобразование 32p знаковых данных к восьмиразрядным осуществляется вырезанием битов 16..31., затем клиппированием и вырезанием восьми младших бит результата. При клиппировании используется глобальная константа типа pm64s с именем `_F1CR_16_to_x`. По умолчанию, там лежит `ff00_ff00_ff00_ff00` соответствующая диапазону результата `[-255..256]`. К каждому полученному 64p слову результата прибавляется значение глобальной переменной `_VR_16_to_x`, которая по умолчанию равна нулю.

### 6.155.1 Подробное описание

## 6.156 Инициализация и копирование

### Группы

- [nmppsRandUniform](#)  
Инициализация массива случайными числами.
- [nmppsConvert](#)  
Изменение разрядности элементов вектора. Входной вектор.
- [nmppsConvert\\_rounding](#)  
Функции конвертации вектора чисел с плавающей точкой одинарной точности в вектор целых 32-битных чисел с разной степенью округления
- [nmppsMerge](#)  
Функция соединяет 2 массива 32-х чисел с плавающей точкой (float) в один. Результирующий массив { pSrcVec1[0], pSrcVec2[0], pSrcVec1[1], pSrcVec2[1], pSrcVec1[2], pSrcVec2[2] и т.д. }.
- [nmppsConvertRisc](#)  
Функции изменения разрядности чисел входного вектора

### 6.156.1 Подробное описание

## 6.157 Арифметические операции

### Группы

- [nmppsAbs](#)  
Вычисление абсолютных значений для элементов вектора.
- [nmppsAbs1](#)  
Функция логического вычисления модулей элементов вектора.
- [nmppsNeg](#)  
Изменение знака элементов вектора на противоположный.
- [nmppsAddC](#)  
Добавление к вектору константы.
- [nmppsAddC\\_p](#)  
Добавление к вектору константы.
- [nmppsAdd](#)  
Сложение двух векторов.
- [nmppsAdd\\_AddC](#)  
Сложение двух векторов с прибавлением константы.
- [nmppsSubC](#)  
Вычитание константы из вектора.
- [nmppsSubCRev](#)  
Вычитание константы из вектора с переменной знака элементов вектора.
- [nmppsSub](#)  
Вычитание двух векторов.
- [nmppsAbsDiff](#)  
Вычисление вектора модулей разности элементов двух векторов.
- [nmppsAbsDiff1](#)  
Функция логического вычисления модулей разностей элементов двух векторов.
- [nmppsMulC](#)  
Умножение вектора на константу.
- [nmppsMul\\_AddC](#)  
Поэлементное умножение векторов с прибавлением константы.
- [nmppsMulC\\_AddC](#)  
Умножение вектора на константу с прибавлением константы.
- [nmppsRShiftC\\_MulC\\_AddC](#)
- [nmppsMulC\\_AddV\\_AddC](#)  
Умножение вектора на константу с прибавлением вектора и константы.
- [nmppsSumN](#)  
Сложение нескольких векторов.
- [nmppsDivC](#)  
Деление вектора на константу.
- [nmppsSum](#)  
Возвращает сумму всех элементов вектора.
- [nmppsSum\\_1](#)  
Возвращает сумму всех элементов вектора.
- [nmppsDotProd\\_sm](#)  
Скалярное умножение двух векторов.
- [nmppsDotProd](#)  
Скалярное умножение двух векторов.
- [nmppsWeightedSum](#)  
Поэлементное взвешенное суммирование элементов двух векторов

### 6.157.1 Подробное описание

## 6.158 Арифметические операции

### Группы

- [nmppsAddC](#)  
Добавление к вектору константы.
- [nmppsAdd](#)  
Сложение двух векторов.
- [nmppsSub](#)  
Вычитание двух векторов.
- [nmppsMulC](#)  
Умножение вектора на константу.
- [nmppsMul\\_Mul\\_Add](#)  
Умножение двух пар векторов с последующим суммированием результатов ( $pDstVec = pSrcVec1 * pSrcVec2 + pSrcVec3 * pSrcVec4$ )
- [nmppsMul\\_Add](#)  
Умножение с накоплением
- [nmppsMul\\_ConjMul\\_Add](#)  
Умножение пары векторов (вектор  $pSrcVec4$  комплексно-сопряженный) с последующим суммированием результата ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] + pSrcVec3[i] * Conj(pSrcVec4[i])$ )
- [nmppsMulC\\_AddC](#)  
Умножает вектор на константу с прибавлением другой константы ( $pDstVec[i] = nMulC * pSrcVec[i] + nAddC$ )
- [nmppsMul](#)  
Умножение векторов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i]$ )
- [nmppsConjMul](#)  
Умножение вектора на комплексно-сопряженный вектор ( $pDstVec[i] = pSrcVec1[i] * Conj(pSrcVec2[i])$ )
- [nmppsMul\\_Mul\\_Sub](#)  
Умножение пары векторов с последующим вычитанием результатов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] - pSrcVec3[i] * pSrcVec4[i]$ )
- [nmppsMulC\\_Add](#)  
Умножение вектора на константу с накоплением ( $pDstVec[i] = pSrcVec1[i] * C + pSrcVec2[i]$ )
- [nmppsMul\\_AddC](#)  
Поэлементное умножение векторов с прибавлением константы.

### 6.158.1 Подробное описание

## 6.159 Логические и бинарные операции

### Группы

- [nmppsNot\\_](#)  
Функция логического "НЕ".
- [nmppsAndC](#)  
Функция логического "И" между вектором и константой.
- [nmppsAnd](#)  
Функция логического "И" между двумя векторами.
- [nmppsAnd4V\\_](#)  
Функция логического "И" между четырьмя векторами.
- [nmppsAndNotV\\_](#)  
Функция логического "И-НЕ" между двумя векторами.
- [nmppsOrC](#)  
Функция логического "ИЛИ" между вектором и константой.
- [nmppsOr](#)  
Функция логического "ИЛИ" между двумя векторами.
- [nmppsOr3V\\_](#)  
Функция логического "ИЛИ" между четырьмя векторами.
- [nmppsOr4V\\_](#)  
Функция логического "ИЛИ" между четырьмя векторами.
- [nmppsXorC](#)  
Функция логического "Исключающего ИЛИ" между вектором и константой.
- [nmppsXor](#)  
Функция логического "Исключающего ИЛИ" между двумя векторами.
- [nmppsMaskV\\_](#)  
Функция логического ИЛИ с предварительным маскированием двух векторов.
- [nmppsRShiftC](#)  
Операция арифметического сдвига вправо.
- [nmppsRShiftC\\_](#)  
Операция логического сдвига.
- [nmppsRShiftC\\_AddC\\_](#)  
Операция логического сдвига.
- [nmppsDisplaceBits](#)  
Непрерывное смещение битов внутри бинарного массива в сторону конца массива  
Функция смещает биты внутри бинарного массива на несколько позиций (nBits) в сторону конца массива.  
Внутри 64р. слова младшие биты сдвигаются на старшие позиции того же слова, а старшие биты перемещаются в младшие позиции следующего 64р. слова. Освободившееся место в первом 64р. слове заполняется старшими битами 64р. слова с адреса pnBits. Сдвинутые биты сохраняются в массиве pDst.  
Пример сдвига на 8 бит :

### 6.159.1 Подробное описание

## 6.160 Операции сравнения

## Группы

- [nmppsMax\\_](#)  
Поиск значения максимального элемента вектора.
- [nmppsMin](#)  
Поиск значения минимального элемента вектора.
- [nmppsMaxIndx\\_](#)  
Поиск значения максимального элемента вектора и его положения (положений) в векторе.
- [nmppsMinIndx\\_](#)  
Поиск значения минимального элемента вектора и его положения (положений) в векторе.
- [nmppsMinIndxVN\\_](#)  
Поиск значения минимального элемента вектора длины N и его положения в векторе.
- [nmppsFirstZeroIndx](#)  
Поиск позиции первого нулевого элемента в векторе .
- [nmppsFirstNonZeroIndx](#)  
Поиск позиции первого ненулевого элемента в векторе .
- [nmppsLastZeroIndx](#)  
Поиск позиции последнего нулевого элемента в векторе .
- [nmppsLastNonZeroIndx](#)  
Поиск позиции последнего ненулевого элемента в векторе .
- [nmppsMinEvery\\_](#)  
Поэлементный минимум из двух векторов.
- [nmppsMaxEvery\\_](#)  
Поэлементный максимум из двух векторов.
- [nmppsMinCmpLtV\\_](#)  
Поэлементный минимум из двух векторов.
- [nmppsCmpLt0](#)  
Сравнивает элементы массива на меньше нуля.
- [nmppsCmpEq0](#)  
Сравнивает элементы массива на признак равенства нулю.
- [nmppsCmpMinMaxV\\_](#)  
Поэлементное сравнение двух векторов.
- [nmppsClipPowC\\_](#)  
Функция насыщения.
- [nmppsClipCC\\_](#)  
Функция насыщения с произвольными порогами.
- [nmppsClipRShiftConvert\\_AddC\\_](#)  
Сокращение разрядности данных с предварительной их обработкой.
- [nmppsClipConvert\\_AddC\\_](#)  
Сокращение разрядности данных с предварительной их обработкой.
- [nmppsCmpEqC](#)  
Сравнивает элементы массива на признак равенства константе.
- [nmppsCmpNe0](#)  
Сравнивает элементы массива на признак неравенства нулю.
- [nmppsCmpNeC](#)  
Сравнивает элементы массива на признак неравенства константе.
- [nmppsCmpNeC\\_flag](#)

Сравнивает элементы массива на признак неравенства константе.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) \neq nCmpVal \\ 0, & \text{if } pSrcVec(i) = nCmpVal \end{cases}$$

- [nmppsCmpLtC](#)

Функция сравнения элементов массива с константой (меньше)

- [nmppsCmpGtC](#)

Функция сравнения элементов массива с константой (больше)

- [nmppsCmpEqV\\_](#)

Поэлементное сравнение элементов двух векторов на признак равенства.

- [nmppsCmpNe](#)

Сравнивает элементы массива на признак неравенства константе.

- [nmppsCmpLt](#)

Функция сравнения элементов массива с константой (меньше)

- [nmppsCmpNeV\\_](#)

Поэлементное сравнение элементов двух векторов на признак неравенства.

### 6.160.1 Подробное описание

## 6.161 Операции сравнения

### Группы

- [nmppsCmpLteC](#)  
Функция сравнения элементов массива с константой (меньше или равно)
- [nmppsCmpLtC](#)  
Функция сравнения элементов массива с константой (меньше или равно)
- [nmppsCmpGteC](#)  
Функция сравнения элементов массива с константой (больше или равно)
- [nmppsCmpGtC](#)  
Функция сравнения элементов массива с константой (больше)

### 6.161.1 Подробное описание



## 6.162 Переупорядочивание и сортировка

### Группы

- [VEC\\_QSort](#)  
Сортировка массива по убыванию.
- [nmppsRemap\\_](#)  
Переупорядочивание элементов вектора по таблице.
- [nmppsSplitTmp](#)  
Расщепляет массив на два, группируя по четным и нечетным элементам
- [nmppsSplit](#)  
Расщепляет массив на два массива, группируя по четным и нечетным элементам
- [nmppsMerge](#)  
Собирает массив из двух, чередуя элементы из каждого. Функция обратная nmppsSplit.
- [nmppsSplit\\_32fcr](#)  
Расщепляет массив на два, группируя по четным и нечетным элементам
- [nmppsDecimate](#)  
Делает выборку элементов из массива с некоторым шагом

### 6.162.1 Подробное описание

## 6.163 Элементарные математические функции

### Группы

- [Тригонометрические функции](#)

Функция, вычисляющая синус/косинус над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

- [Функция деления векторов](#)

Функция, вычисляющая  $x/y$  (`pSrcVec1 / pSrcVec2`), где  $x$  и  $y$  (делимые числа и делители) это вектора чисел с плавающей точкой одинарной

- [Экспонента](#)

Функция, вычисляющая экспоненту над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

- [Натуральный логарифм](#)

Функция, вычисляющая натуральный логарифм над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

- [Возведение в степень](#)

Функция возводит в степень (`Deg`) каждый элемент вектора чисел с плавающей точкой двойной точности (64f)

### 6.163.1 Подробное описание

## 6.164 Тригонометрические функции

Функция, вычисляющая синус/косинус над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

### Функции

- void nmppsSin\_32f (const nm32f \*pSrcVec, nm32f \*pDstVec, int nSize)
- void nmppsCos\_32f (const nm32f \*pSrcVec, nm32f \*pDstVec, int nSize)

### 6.164.1 Подробное описание

Функция, вычисляющая синус/косинус над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

#### Аргументы

|    |         |                                         |
|----|---------|-----------------------------------------|
| in | pSrcVec | входной вектор чисел с плавающей точкой |
|----|---------|-----------------------------------------|

#### Возвращаемые значения

|       |         |                                          |
|-------|---------|------------------------------------------|
| [out] | pDstVec | выходной вектор чисел с плавающей точкой |
|-------|---------|------------------------------------------|

#### Аргументы

|    |       |                           |
|----|-------|---------------------------|
| in | nSize | число элементов в векторе |
|----|-------|---------------------------|

Функции совместимы ТОЛЬКО с NMC-GCC-SDK и входит в состав библиотеки libnmpp-nmc4f.a

## 6.165 Функция деления векторов

Функция, вычисляющая  $x/y$  (`pSrcVec1 / pSrcVec2`), где  $x$  и  $y$  (делимые числа и делители) это вектора чисел с плавающей точкой одинарной

Функции

- `void nmppsDiv_32f (const nm32f *pSrcVec1, const nm32f *pSrcVec2, nm32f *pDstVec, int nSize)`

### 6.165.1 Подробное описание

Функция, вычисляющая  $x/y$  (`pSrcVec1 / pSrcVec2`), где  $x$  и  $y$  (делимые числа и делители) это вектора чисел с плавающей точкой одинарной

Аргументы

|    |          |                                                                    |
|----|----------|--------------------------------------------------------------------|
| in | pSrcVec1 | входной вектор делимых чисел с плавающей точкой одинарной точности |
| in | pSrcVec2 | входной вектор делителей с плавающей точкой одинарной точности     |

Возвращаемые значения

|       |                                                                       |
|-------|-----------------------------------------------------------------------|
| [out] | pDstVec выходной вектор частных с плавающей точкой одинарной точности |
|-------|-----------------------------------------------------------------------|

Аргументы

|    |       |                           |
|----|-------|---------------------------|
| in | nSize | число элементов в векторе |
|----|-------|---------------------------|

Функция совместима ТОЛЬКО с NMC-GCC-SDK и входит в состав библиотеки `libnmpp-nmc4f.a`

## 6.166 Экспонента

Функция, вычисляющая экспоненту над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

### Функции

- void nmppsExp\_32f (const nm32f \*pSrcVec, nm32f \*pDstVec, int nSize)
- void nmppsExp\_64f (const nm64f \*pSrcVec, nm64f \*pDstVec, int nSize)

### 6.166.1 Подробное описание

Функция, вычисляющая экспоненту над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

#### Аргументы

|    |         |                                                            |
|----|---------|------------------------------------------------------------|
| in | pSrcVec | входной вектор чисел с плавающей точкой одинарной точности |
|----|---------|------------------------------------------------------------|

#### Возвращаемые значения

|       |         |                                                             |
|-------|---------|-------------------------------------------------------------|
| [out] | pDstVec | выходной вектор чисел с плавающей точкой одинарной точности |
|-------|---------|-------------------------------------------------------------|

#### Аргументы

|    |       |                           |
|----|-------|---------------------------|
| in | nSize | число элементов в векторе |
|----|-------|---------------------------|

Функция совместима ТОЛЬКО с NMC-GCC-SDK и входит в состав библиотеки libnmpp-nmc4f.a

## 6.167    Натуральный логарифм

Функция, вычисляющая натуральный логарифм над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

Функции

- void nmppsLn\_32f (const nm32f \*pSrcVec, nm32f \*pDstVec, int nSize)
- void nmppsLn\_64f (const nm64f \*pSrcVec, nm64f \*pDstVec, int nSize)

### 6.167.1    Подробное описание

Функция, вычисляющая натуральный логарифм над каждым элементом вектора чисел с плавающей точкой одинарной точности (32f)

Аргументы

|    |         |                                                            |
|----|---------|------------------------------------------------------------|
| in | pSrcVec | входной вектор чисел с плавающей точкой одинарной точности |
|----|---------|------------------------------------------------------------|

Возвращаемые значения

|       |         |                                                             |
|-------|---------|-------------------------------------------------------------|
| [out] | pDstVec | выходной вектор чисел с плавающей точкой одинарной точности |
|-------|---------|-------------------------------------------------------------|

Аргументы

|    |       |                           |
|----|-------|---------------------------|
| in | nSize | число элементов в векторе |
|----|-------|---------------------------|

Функция совместима ТОЛЬКО с NMC-GCC-SDK и входит в состав библиотеки libnmpp-nmc4f.a

## 6.168 Возведение в степень

Функция возводит в степень (Deg) каждый элемент вектора чисел с плавающей точкой двойной точности (64f)

Функции

- void nmppsPowx\_64f (const nm64f \*pSrcVec, nm64f \*pDstVec, nm32u Deg, int nSize)

### 6.168.1 Подробное описание

Функция возводит в степень (Deg) каждый элемент вектора чисел с плавающей точкой двойной точности (64f)

Аргументы

|    |         |                                                          |
|----|---------|----------------------------------------------------------|
| in | pSrcVec | входной вектор чисел с плавающей точкой двойной точности |
|----|---------|----------------------------------------------------------|

Возвращаемые значения

|       |         |                                                           |
|-------|---------|-----------------------------------------------------------|
| [out] | pDstVec | выходной вектор чисел с плавающей точкой двойной точности |
|-------|---------|-----------------------------------------------------------|

Аргументы

|    |       |                                                               |
|----|-------|---------------------------------------------------------------|
| in | Deg   | степень, в которую возводится каждый элемент входного вектора |
| in | nSize | число элементов в векторе                                     |

Функция совместима ТОЛЬКО с NMC-GCC-SDK и входит в состав библиотеки libnmpp-nmc4f.a

## 6.169 nmppsAbs

Вычисление абсолютных значений для элементов вектора.

Функции

- void `nmppsAbs_4s` (const `nm4s` \*pSrcVec, `nm4s` \*pDstVec, int nSize)
- void `nmppsAbs_8s` (const `nm8s` \*pSrcVec, `nm8s` \*pDstVec, int nSize)
- void `nmppsAbs_16s` (const `nm16s` \*pSrcVec, `nm16s` \*pDstVec, int nSize)
- void `nmppsAbs_32s` (const `nm32s` \*pSrcVec, `nm32s` \*pDstVec, int nSize)
- void `nmppsAbs_64s` (const `nm64s` \*pSrcVec, `nm64s` \*pDstVec, int nSize)

### 6.169.1 Подробное описание

Вычисление абсолютных значений для элементов вектора.

$$pDstVec[i] = abs\{pSrcVec[i]\},$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

Restrictions:

Значения элементов вектора не должны быть равны минимальному значению для соответствующего типа (т.е. -128, -2<sup>15</sup> и т.д). В противном случае, абсолютное значение для таких элементов вычисляется неверно, давая на выходе то же самое число.

### 6.169.2 Функции



## 6.169.2.1 nmppsAbs\_16s()

```
void nmppsAbs_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1535  | 0.74       |
| im0     | im0     | 2048  | 1535  | 0.74       |
| im0     | im1     | 2048  | 1178  | 0.57       |
| im0     | im1     | 2048  | 1178  | 0.57       |
| im1     | im0     | 2048  | 1163  | 0.56       |
| im1     | im0     | 2048  | 1163  | 0.56       |
| im1     | im1     | 2048  | 1676  | 0.81       |
| im1     | im1     | 2048  | 1676  | 0.81       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1163  | 0.56       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1676  | 0.81       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 116   | 14.50      |
| im0     | im1     | 8     | 116   | 14.50      |
| im0     | im1     | 128   | 205   | 1.60       |
| im0     | im1     | 128   | 205   | 1.60       |
| im0     | im1     | 1024  | 658   | 0.64       |
| im0     | im1     | 1024  | 658   | 0.64       |
| im0     | im1     | 2048  | 1178  | 0.57       |
| im0     | im1     | 2048  | 1178  | 0.57       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 116   | 14.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1178  | 0.57       |

### 6.169.2.2 nmppsAbs\_32s()

```
void nmppsAbs_32s (
    const nm32s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 3052  | 1.49       |
| im0     | im0     | 2048  | 3052  | 1.49       |
| im0     | im1     | 2048  | 2215  | 1.08       |
| im0     | im1     | 2048  | 2215  | 1.08       |
| im1     | im0     | 2048  | 2248  | 1.09       |
| im1     | im0     | 2048  | 2248  | 1.09       |
| im1     | im1     | 2048  | 3241  | 1.58       |
| im1     | im1     | 2048  | 3241  | 1.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2215  | 1.08       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3241  | 1.58       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 128   | 265   | 2.07       |
| im0     | im1     | 128   | 265   | 2.07       |
| im0     | im1     | 1024  | 1175  | 1.14       |
| im0     | im1     | 1024  | 1175  | 1.14       |
| im0     | im1     | 2048  | 2215  | 1.08       |
| im0     | im1     | 2048  | 2215  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2215  | 1.08       |

### 6.169.2.3 nmppsAbs\_4s()

```
void nmppsAbs_4s (
    const nm4s * pSrcVec,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 408   | 0.19       |
| im0     | im0     | 2048  | 408   | 0.19       |
| im0     | im1     | 2048  | 411   | 0.20       |
| im0     | im1     | 2048  | 411   | 0.20       |
| im1     | im0     | 2048  | 360   | 0.17       |
| im1     | im0     | 2048  | 360   | 0.17       |
| im1     | im1     | 2048  | 513   | 0.25       |
| im1     | im1     | 2048  | 513   | 0.25       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 360   | 0.17       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 513   | 0.25       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 129   | 16.12      |
| im0     | im1     | 8     | 129   | 16.12      |
| im0     | im1     | 128   | 148   | 1.15       |
| im0     | im1     | 128   | 148   | 1.15       |
| im0     | im1     | 1024  | 281   | 0.27       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 1024  | 281   | 0.27       |
| im0     | im1     | 2048  | 411   | 0.20       |
| im0     | im1     | 2048  | 411   | 0.20       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 129   | 16.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 411   | 0.20       |

#### 6.169.2.4 nmppsAbs\_64s()

```
void nmppsAbs_64s (
    const nm64s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 6096  | 2.97       |
| im0     | im0     | 2048  | 6096  | 2.97       |
| im0     | im1     | 2048  | 4299  | 2.09       |
| im0     | im1     | 2048  | 4299  | 2.09       |
| im1     | im0     | 2048  | 4428  | 2.16       |
| im1     | im0     | 2048  | 4428  | 2.16       |
| im1     | im1     | 2048  | 6381  | 3.11       |
| im1     | im1     | 2048  | 6381  | 3.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 4299  | 2.09       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 6381  | 3.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 136   | 17.00      |
| im0     | im1     | 8     | 136   | 17.00      |
| im0     | im1     | 128   | 399   | 3.11       |
| im0     | im1     | 128   | 399   | 3.11       |
| im0     | im1     | 1024  | 2219  | 2.16       |
| im0     | im1     | 1024  | 2219  | 2.16       |
| im0     | im1     | 2048  | 4299  | 2.09       |
| im0     | im1     | 2048  | 4299  | 2.09       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 136   | 17.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 4299  | 2.09       |

### 6.169.2.5 nmppsAbs\_8s()

```
void nmppsAbs_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 780   | 0.38       |
| im0     | im0     | 2048  | 780   | 0.38       |
| im0     | im1     | 2048  | 663   | 0.32       |
| im0     | im1     | 2048  | 663   | 0.32       |
| im1     | im0     | 2048  | 624   | 0.30       |
| im1     | im0     | 2048  | 624   | 0.30       |
| im1     | im1     | 2048  | 897   | 0.43       |
| im1     | im1     | 2048  | 897   | 0.43       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 624   | 0.30       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 897   | 0.43       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 121   | 15.12      |
| im0     | im1     | 8     | 121   | 15.12      |
| im0     | im1     | 128   | 164   | 1.28       |
| im0     | im1     | 128   | 164   | 1.28       |
| im0     | im1     | 1024  | 403   | 0.39       |
| im0     | im1     | 1024  | 403   | 0.39       |
| im0     | im1     | 2048  | 663   | 0.32       |
| im0     | im1     | 2048  | 663   | 0.32       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 121   | 15.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 663   | 0.32       |

## 6.170 nmppsAbs1

Функция логического вычисления модулей элементов вектора.

Функции

- void nmppsAbs1\_4s (const nm4s \*pSrcVec, nm4s \*pDstVec, int nSize)
- void nmppsAbs1\_8s (const nm8s \*pSrcVec, nm8s \*pDstVec, int nSize)
- void nmppsAbs1\_16s (const nm16s \*pSrcVec, nm16s \*pDstVec, int nSize)
- void nmppsAbs1\_32s (const nm32s \*pSrcVec, nm32s \*pDstVec, int nSize)
- void nmppsAbs1\_64s (const nm64s \*pSrcVec, nm64s \*pDstVec, int nSize)

### 6.170.1 Подробное описание

Функция логического вычисления модулей элементов вектора.

$$pDstVec[i] = \begin{cases} pSrcVec[i], & \text{if } pSrcVec[i] \geq 0 \\ -pSrcVec[i] - 1, & \text{if } pSrcVec[i] < 0 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.170.2 Функции

## 6.170.2.1 nmppsAbs1\_16s()

```
void nmppsAbs1_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1064  | 0.51       |
| im0     | im0     | 2048  | 1064  | 0.51       |
| im0     | im0     | 2048  | 1064  | 0.51       |
| im0     | im0     | 2048  | 1064  | 0.51       |
| im1     | im0     | 2048  | 651   | 0.31       |
| im1     | im0     | 2048  | 651   | 0.31       |
| im1     | im0     | 2048  | 651   | 0.31       |
| im1     | im0     | 2048  | 651   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 651   | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1064  | 0.51       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 128   | 144   | 1.12       |
| im0     | im1     | 128   | 144   | 1.12       |
| im0     | im1     | 1024  | 383   | 0.37       |
| im0     | im1     | 1024  | 383   | 0.37       |
| im0     | im1     | 2048  | 653   | 0.31       |
| im0     | im1     | 2048  | 653   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |

The worst configuration:



| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 653   | 0.31       |

## 6.170.2.2 nmppsAbs1\_32s()

```
void nmppsAbs1_32s (
    const nm32s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2088  | 1.01       |
| im0     | im0     | 2048  | 2088  | 1.01       |
| im0     | im0     | 2048  | 2088  | 1.01       |
| im0     | im0     | 2048  | 2088  | 1.01       |
| im1     | im0     | 2048  | 1195  | 0.58       |
| im1     | im0     | 2048  | 1195  | 0.58       |
| im1     | im0     | 2048  | 1195  | 0.58       |
| im1     | im0     | 2048  | 1195  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1195  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2088  | 1.01       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 111   | 13.87      |
| im0     | im1     | 8     | 111   | 13.87      |
| im0     | im1     | 128   | 179   | 1.39       |
| im0     | im1     | 128   | 179   | 1.39       |
| im0     | im1     | 1024  | 653   | 0.63       |
| im0     | im1     | 1024  | 653   | 0.63       |
| im0     | im1     | 2048  | 1197  | 0.58       |
| im0     | im1     | 2048  | 1197  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 111   | 13.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1197  | 0.58       |

### 6.170.2.3 nmppsAbs1\_64s()

```
void nmppsAbs1_64s (
    const nm64s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4139  | 2.02       |
| im0     | im0     | 2048  | 4139  | 2.02       |
| im0     | im0     | 2048  | 4139  | 2.02       |
| im0     | im0     | 2048  | 4139  | 2.02       |
| im1     | im0     | 2048  | 2286  | 1.11       |
| im1     | im0     | 2048  | 2286  | 1.11       |
| im1     | im0     | 2048  | 2286  | 1.11       |
| im1     | im0     | 2048  | 2286  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2286  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4139  | 2.02       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 121   | 15.12      |
| im0     | im1     | 8     | 121   | 15.12      |
| im0     | im1     | 128   | 249   | 1.94       |
| im0     | im1     | 128   | 249   | 1.94       |
| im0     | im1     | 1024  | 1200  | 1.17       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 1024  | 1200  | 1.17       |
| im0     | im1     | 2048  | 2288  | 1.11       |
| im0     | im1     | 2048  | 2288  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 121   | 15.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2288  | 1.11       |

#### 6.170.2.4 nmppsAbs1\_8s()

```
void nmppsAbs1_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 552   | 0.26       |
| im0     | im0     | 2048  | 552   | 0.26       |
| im0     | im0     | 2048  | 552   | 0.26       |
| im0     | im0     | 2048  | 552   | 0.26       |
| im1     | im0     | 2048  | 379   | 0.18       |
| im1     | im0     | 2048  | 379   | 0.18       |
| im1     | im0     | 2048  | 379   | 0.18       |
| im1     | im0     | 2048  | 379   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 379   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 552   | 0.26       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 128   | 126   | 0.98       |
| im0     | im1     | 128   | 126   | 0.98       |
| im0     | im1     | 1024  | 246   | 0.24       |
| im0     | im1     | 1024  | 246   | 0.24       |
| im0     | im1     | 2048  | 383   | 0.18       |
| im0     | im1     | 2048  | 383   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 383   | 0.18       |

## 6.171 nmppsNeg

Изменение знака элементов вектора на противоположный.

### Функции

- void `nmppsNeg_8s` (const `nm8s` \*pSrcVec, `nm8s` \*pDstVec, int nSize)
- void `nmppsNeg_16s` (const `nm16s` \*pSrcVec, `nm16s` \*pDstVec, int nSize)
- void `nmppsNeg_32s` (const `nm32s` \*pSrcVec, `nm32s` \*pDstVec, int nSize)
- void `nmppsNeg_64s` (const `nm64s` \*pSrcVec, `nm64s` \*pDstVec, int nSize)

### 6.171.1 Подробное описание

Изменение знака элементов вектора на противоположный.

$$pDstVec[i] = -pSrcVec[i]$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.171.2 Функции

#### 6.171.2.1 nmppsNeg\_16s()

```
void nmppsNeg_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Performance table 0

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 1060  | 0.51       |
| im0      | im0      | 2048  | 1060  | 0.51       |
| im0      | im1      | 2048  | 649   | 0.31       |
| im0      | im1      | 2048  | 649   | 0.31       |
| im1      | im0      | 2048  | 647   | 0.31       |
| im1      | im0      | 2048  | 647   | 0.31       |
| im1      | im1      | 2048  | 1159  | 0.56       |
| im1      | im1      | 2048  | 1159  | 0.56       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 647   | 0.31       |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 1159  | 0.56       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 104   | 13.00      |
| im0      | im1      | 8     | 104   | 13.00      |
| im0      | im1      | 128   | 140   | 1.09       |
| im0      | im1      | 128   | 140   | 1.09       |
| im0      | im1      | 1024  | 379   | 0.37       |
| im0      | im1      | 1024  | 379   | 0.37       |
| im0      | im1      | 2048  | 649   | 0.31       |
| im0      | im1      | 2048  | 649   | 0.31       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 104   | 13.00      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 649   | 0.31       |

6.171.2.2 nmppsNeg\_32s()

void nmppsNeg\_32s (

```

const nm32s * pSrcVec,
nm32s * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2084  | 1.01       |
| im0     | im0     | 2048  | 2084  | 1.01       |
| im0     | im1     | 2048  | 1193  | 0.58       |
| im0     | im1     | 2048  | 1193  | 0.58       |
| im1     | im0     | 2048  | 1191  | 0.58       |
| im1     | im0     | 2048  | 1191  | 0.58       |
| im1     | im1     | 2048  | 2215  | 1.08       |
| im1     | im1     | 2048  | 2215  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1191  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2215  | 1.08       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 107   | 13.37      |
| im0     | im1     | 8     | 107   | 13.37      |
| im0     | im1     | 128   | 175   | 1.36       |
| im0     | im1     | 128   | 175   | 1.36       |
| im0     | im1     | 1024  | 649   | 0.63       |
| im0     | im1     | 1024  | 649   | 0.63       |
| im0     | im1     | 2048  | 1193  | 0.58       |
| im0     | im1     | 2048  | 1193  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 107   | 13.37      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1193  | 0.58       |

## 6.171.2.3 nmppsNeg\_64s()

```
void nmppsNeg_64s (
    const nm64s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4132  | 2.01       |
| im0     | im0     | 2048  | 4132  | 2.01       |
| im0     | im1     | 2048  | 2281  | 1.11       |
| im0     | im1     | 2048  | 2281  | 1.11       |
| im1     | im0     | 2048  | 2279  | 1.11       |
| im1     | im0     | 2048  | 2279  | 1.11       |
| im1     | im1     | 2048  | 4327  | 2.11       |
| im1     | im1     | 2048  | 4327  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2279  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4327  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 114   | 14.25      |
| im0     | im1     | 8     | 114   | 14.25      |
| im0     | im1     | 128   | 242   | 1.89       |
| im0     | im1     | 128   | 242   | 1.89       |
| im0     | im1     | 1024  | 1193  | 1.16       |
| im0     | im1     | 1024  | 1193  | 1.16       |
| im0     | im1     | 2048  | 2281  | 1.11       |
| im0     | im1     | 2048  | 2281  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 114   | 14.25      |



The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2281  | 1.11       |

#### 6.171.2.4 nmppsNeg\_8s()

```
void nmppsNeg_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 548   | 0.26       |
| im0     | im0     | 2048  | 548   | 0.26       |
| im0     | im1     | 2048  | 379   | 0.18       |
| im0     | im1     | 2048  | 379   | 0.18       |
| im1     | im0     | 2048  | 375   | 0.18       |
| im1     | im0     | 2048  | 375   | 0.18       |
| im1     | im1     | 2048  | 631   | 0.30       |
| im1     | im1     | 2048  | 631   | 0.30       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 375   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 631   | 0.30       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 104   | 13.00      |
| im0     | im1     | 8     | 104   | 13.00      |
| im0     | im1     | 128   | 122   | 0.95       |
| im0     | im1     | 128   | 122   | 0.95       |
| im0     | im1     | 1024  | 242   | 0.23       |
| im0     | im1     | 1024  | 242   | 0.23       |
| im0     | im1     | 2048  | 379   | 0.18       |
| im0     | im1     | 2048  | 379   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 104   | 13.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 379   | 0.18       |

## 6.172 nmppsAddC

Добавление к вектору константы.

Функции

- void `nmppsAddC_8s` (const `nm8s` \*pSrcVec, `int8b` nVal, `nm8s` \*pDstVec, int nSize)
- void `nmppsAddC_16s` (const `nm16s` \*pSrcVec, `int16b` nVal, `nm16s` \*pDstVec, int nSize)
- void `nmppsAddC_32s` (const `nm32s` \*pSrcVec, `int32b` nVal, `nm32s` \*pDstVec, int nSize)
- void `nmppsAddC_64s` (const `nm64s` \*pSrcVec, `int64b` nVal, `nm64s` \*pDstVec, int nSize)

### 6.172.1 Подробное описание

Добавление к вектору константы.

$$pDstVec[i] = pSrcVec[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Добавляемая константа.       |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.172.2 Функции

## 6.172.2.1 nmppsAddC\_16s()

```
void nmppsAddC_16s (
    const nm16s * pSrcVec,
    int16b nVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 1088  | 0.53       |
| im0     | 3    | im0     | 2048  | 1088  | 0.53       |
| im0     | 3    | im1     | 2048  | 698   | 0.34       |
| im0     | 3    | im1     | 2048  | 698   | 0.34       |
| im1     | 3    | im0     | 2048  | 701   | 0.34       |
| im1     | 3    | im0     | 2048  | 701   | 0.34       |
| im1     | 3    | im1     | 2048  | 1212  | 0.59       |
| im1     | 3    | im1     | 2048  | 1212  | 0.59       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 698   | 0.34       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 1212  | 0.59       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 131   | 16.37      |
| im0     | 3    | im1     | 8     | 131   | 16.37      |
| im0     | 3    | im1     | 128   | 188   | 1.46       |
| im0     | 3    | im1     | 128   | 188   | 1.46       |
| im0     | 3    | im1     | 1024  | 426   | 0.41       |
| im0     | 3    | im1     | 1024  | 426   | 0.41       |
| im0     | 3    | im1     | 2048  | 698   | 0.34       |
| im0     | 3    | im1     | 2048  | 698   | 0.34       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 131   | 16.37      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 698   | 0.34       |

## 6.172.2.2 nmppsAddC\_32s()

```
void nmppsAddC_32s (
    const nm32s * pSrcVec,
    int32b nVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 2106  | 1.02       |
| im0     | 3    | im0     | 2048  | 2106  | 1.02       |
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |
| im1     | 3    | im0     | 2048  | 1239  | 0.60       |
| im1     | 3    | im0     | 2048  | 1239  | 0.60       |
| im1     | 3    | im1     | 2048  | 2262  | 1.10       |
| im1     | 3    | im1     | 2048  | 2262  | 1.10       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 2262  | 1.10       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 129   | 16.12      |
| im0     | 3    | im1     | 8     | 129   | 16.12      |
| im0     | 3    | im1     | 128   | 219   | 1.71       |
| im0     | 3    | im1     | 128   | 219   | 1.71       |
| im0     | 3    | im1     | 1024  | 692   | 0.67       |
| im0     | 3    | im1     | 1024  | 692   | 0.67       |
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 129   | 16.12      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1236  | 0.60       |

### 6.172.2.3 nmppsAddC\_64s()

```
void nmppsAddC_64s (
    const nm64s * pSrcVec,
    int64b nVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 4154  | 2.02       |
| im0     | 3    | im0     | 2048  | 4154  | 2.02       |
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |
| im1     | 3    | im0     | 2048  | 2327  | 1.13       |
| im1     | 3    | im0     | 2048  | 2327  | 1.13       |
| im1     | 3    | im1     | 2048  | 4374  | 2.13       |
| im1     | 3    | im1     | 2048  | 4374  | 2.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 4374  | 2.13       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 139   | 17.37      |
| im0     | 3    | im1     | 8     | 139   | 17.37      |
| im0     | 3    | im1     | 128   | 284   | 2.21       |
| im0     | 3    | im1     | 128   | 284   | 2.21       |

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 1024  | 1236  | 1.20       |
| im0     | 3    | im1     | 1024  | 1236  | 1.20       |
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 139   | 17.37      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2324  | 1.13       |

#### 6.172.2.4 nmppsAddC\_8s()

```
void nmppsAddC_8s (
    const nm8s * pSrcVec,
    int8b nVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 579   | 0.28       |
| im0     | 3    | im0     | 2048  | 579   | 0.28       |
| im0     | 3    | im1     | 2048  | 429   | 0.20       |
| im0     | 3    | im1     | 2048  | 429   | 0.20       |
| im1     | 3    | im0     | 2048  | 432   | 0.21       |
| im1     | 3    | im0     | 2048  | 432   | 0.21       |
| im1     | 3    | im1     | 2048  | 687   | 0.33       |
| im1     | 3    | im1     | 2048  | 687   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 429   | 0.20       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 687   | 0.33       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 133   | 16.62      |
| im0     | 3    | im1     | 8     | 133   | 16.62      |
| im0     | 3    | im1     | 128   | 163   | 1.27       |
| im0     | 3    | im1     | 128   | 163   | 1.27       |
| im0     | 3    | im1     | 1024  | 293   | 0.28       |
| im0     | 3    | im1     | 1024  | 293   | 0.28       |
| im0     | 3    | im1     | 2048  | 429   | 0.20       |
| im0     | 3    | im1     | 2048  | 429   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 133   | 16.62      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 429   | 0.20       |



## 6.173 nmppsAddC\_p

Добавление к вектору константы.

Функции

- void nmppsAddC\_p64s (const nm64s \*pSrcVec, int64b \*pnVal, nm64s \*pDstVec, int nSize)

## 6.173.1 Подробное описание

Добавление к вектору константы.

$$pDstVec[i] = pSrcVec[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                                     |
|---------|-------------------------------------|
| pSrcVec | Входной вектор.                     |
| pnVal   | Указатель на добавляемую константу. |
| nSize   | Размер векторов в элементах.        |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.173.2 Функции

## 6.173.2.1 nmppsAddC\_p64s()

```
void nmppsAddC_p64s (
    const nm64s * pSrcVec,
    int64b * pnVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im0     | im2   | im1     | 2048  | 2294  | 1.12       |
| im0     | im2   | im1     | 2048  | 2294  | 1.12       |
| im0     | im2   | im0     | 2048  | 4146  | 2.02       |
| im0     | im2   | im0     | 2048  | 4146  | 2.02       |
| im1     | im2   | im1     | 2048  | 4341  | 2.11       |
| im1     | im2   | im1     | 2048  | 4341  | 2.11       |
| im1     | im2   | im0     | 2048  | 2293  | 1.11       |
| im1     | im2   | im0     | 2048  | 2293  | 1.11       |

The best configuration:

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im1     | im2   | im0     | 2048  | 2293  | 1.11       |

The worst configuration:

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im1     | im2   | im1     | 2048  | 4341  | 2.11       |

Perfomance table 1

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im0     | im2   | im1     | 8     | 131   | 16.37      |
| im0     | im2   | im1     | 8     | 131   | 16.37      |
| im0     | im2   | im1     | 128   | 254   | 1.98       |
| im0     | im2   | im1     | 128   | 254   | 1.98       |
| im0     | im2   | im1     | 1024  | 1206  | 1.17       |
| im0     | im2   | im1     | 1024  | 1206  | 1.17       |
| im0     | im2   | im1     | 2048  | 2294  | 1.12       |
| im0     | im2   | im1     | 2048  | 2294  | 1.12       |

The best configuration:

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im0     | im2   | im1     | 8     | 131   | 16.37      |

The worst configuration:

| pSrcVec | pnVal | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|---------|-------|-------|------------|
| im0     | im2   | im1     | 2048  | 2294  | 1.12       |

## 6.174 nmppsAddC

Добавление к вектору константы.

Функции

- void `nmppsAddC_32fcr` (const `nm32fcr` \*pSrcVec, `nm32fcr` \*pDstVec, float nVal, int nSize)

### 6.174.1 Подробное описание

Добавление к вектору константы.

$$pDstVec[i] = pSrcVec[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Добавляемая константа.       |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.174.2 Функции

#### 6.174.2.1 nmppsAddC\_32fcr()

```
void nmppsAddC_32fcr (
    const nm32fcr * pSrcVec,
    nm32fcr * pDstVec,
    float nVal,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im3     | 2048  | 2077  | 1.01       |
| im1     | 3    | im3     | 2048  | 2077  | 1.01       |
| im1     | 3    | im4     | 2048  | 2077  | 1.01       |
| im1     | 3    | im4     | 2048  | 2077  | 1.01       |
| im2     | 3    | im3     | 2048  | 2077  | 1.01       |
| im2     | 3    | im3     | 2048  | 2077  | 1.01       |
| im2     | 3    | im4     | 2048  | 2077  | 1.01       |
| im2     | 3    | im4     | 2048  | 2077  | 1.01       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im3     | 2048  | 2077  | 1.01       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im3     | 2048  | 2077  | 1.01       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im2     | 8     | 90    | 11.24      |
| im1     | 3    | im2     | 8     | 90    | 11.24      |
| im1     | 3    | im2     | 128   | 157   | 1.22       |
| im1     | 3    | im2     | 128   | 157   | 1.22       |
| im1     | 3    | im2     | 1024  | 1053  | 1.02       |
| im1     | 3    | im2     | 1024  | 1053  | 1.02       |
| im1     | 3    | im2     | 2048  | 2077  | 1.01       |
| im1     | 3    | im2     | 2048  | 2077  | 1.01       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im2     | 8     | 90    | 11.24      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im2     | 2048  | 2077  | 1.01       |

## 6.175 nmppsAdd

Сложение двух векторов.

## Функции

- void `nmppsAdd_4s` (const `nm4s` \*pSrcVec1, const `nm4s` \*pSrcVec2, `nm4s` \*pDstVec, int nSize)
- void `nmppsAdd_8s` (const `nm8s` \*pSrcVec1, const `nm8s` \*pSrcVec2, `nm8s` \*pDstVec, int nSize)
- void `nmppsAdd_16s` (const `nm16s` \*pSrcVec1, const `nm16s` \*pSrcVec2, `nm16s` \*pDstVec, int nSize)
- void `nmppsAdd_32s` (const `nm32s` \*pSrcVec1, const `nm32s` \*pSrcVec2, `nm32s` \*pDstVec, int nSize)
- void `nmppsAdd_64s` (const `nm64s` \*pSrcVec1, const `nm64s` \*pSrcVec2, `nm64s` \*pDstVec, int nSize)

## 6.175.1 Подробное описание

Сложение двух векторов.

$$pDstVec[i] = pSrcVec1[i] + pSrcVec2[i],$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Первый входной вектор.      |
| pSrcVec2 | Второй входной вектор.      |
| nSize    | Размер вектора в элементах. |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.175.2 Функции

## 6.175.2.1 nmppsAdd\_16s()

```
void nmppsAdd_16s (
    const nm16s * pSrcVec1,
    const nm16s * pSrcVec2,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im1     | 2048  | 1142  | 0.55       |
| im0      | im0      | im1     | 2048  | 1142  | 0.55       |
| im0      | im0      | im2     | 2048  | 1080  | 0.52       |
| im0      | im0      | im2     | 2048  | 1080  | 0.52       |
| im0      | im1      | im1     | 2048  | 1199  | 0.58       |
| im0      | im1      | im1     | 2048  | 1199  | 0.58       |
| im0      | im1      | im2     | 2048  | 675   | 0.32       |
| im0      | im1      | im2     | 2048  | 675   | 0.32       |
| im1      | im0      | im1     | 2048  | 1185  | 0.57       |
| im1      | im0      | im1     | 2048  | 1185  | 0.57       |
| im1      | im0      | im2     | 2048  | 674   | 0.32       |
| im1      | im0      | im2     | 2048  | 674   | 0.32       |
| im1      | im1      | im1     | 2048  | 1698  | 0.82       |
| im1      | im1      | im1     | 2048  | 1698  | 0.82       |
| im1      | im1      | im2     | 2048  | 1187  | 0.57       |
| im1      | im1      | im2     | 2048  | 1187  | 0.57       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 674   | 0.32       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 1698  | 0.82       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 128   | 149   | 1.16       |
| im0      | im1      | im2     | 128   | 149   | 1.16       |
| im0      | im1      | im2     | 1024  | 395   | 0.38       |
| im0      | im1      | im2     | 1024  | 395   | 0.38       |
| im0      | im1      | im2     | 2048  | 675   | 0.32       |
| im0      | im1      | im2     | 2048  | 675   | 0.32       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 675   | 0.32       |

### 6.175.2.2 nmppsAdd\_32s()

```
void nmppsAdd_32s (
    const nm32s * pSrcVec1,
    const nm32s * pSrcVec2,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im1     | 2048  | 2166  | 1.05       |
| im0      | im0      | im1     | 2048  | 2166  | 1.05       |
| im0      | im0      | im2     | 2048  | 2104  | 1.02       |
| im0      | im0      | im2     | 2048  | 2104  | 1.02       |
| im0      | im1      | im1     | 2048  | 2287  | 1.11       |
| im0      | im1      | im1     | 2048  | 2287  | 1.11       |
| im0      | im1      | im2     | 2048  | 1235  | 0.60       |
| im0      | im1      | im2     | 2048  | 1235  | 0.60       |
| im1      | im0      | im1     | 2048  | 2257  | 1.10       |
| im1      | im0      | im1     | 2048  | 2257  | 1.10       |
| im1      | im0      | im2     | 2048  | 1234  | 0.60       |
| im1      | im0      | im2     | 2048  | 1234  | 0.60       |
| im1      | im1      | im1     | 2048  | 3282  | 1.60       |
| im1      | im1      | im1     | 2048  | 3282  | 1.60       |
| im1      | im1      | im2     | 2048  | 2259  | 1.10       |
| im1      | im1      | im2     | 2048  | 2259  | 1.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 1234  | 0.60       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 3282  | 1.60       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 122   | 15.25      |
| im0      | im1      | im2     | 8     | 122   | 15.25      |
| im0      | im1      | im2     | 128   | 184   | 1.43       |
| im0      | im1      | im2     | 128   | 184   | 1.43       |
| im0      | im1      | im2     | 1024  | 675   | 0.65       |
| im0      | im1      | im2     | 1024  | 675   | 0.65       |
| im0      | im1      | im2     | 2048  | 1235  | 0.60       |
| im0      | im1      | im2     | 2048  | 1235  | 0.60       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 122   | 15.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1235  | 0.60       |

### 6.175.2.3 nmppsAdd\_4s()

```
void nmppsAdd_4s (
    const nm4s * pSrcVec1,
    const nm4s * pSrcVec2,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im1     | 2048  | 374   | 0.18       |
| im0      | im0      | im1     | 2048  | 374   | 0.18       |
| im0      | im0      | im2     | 2048  | 312   | 0.15       |
| im0      | im0      | im2     | 2048  | 312   | 0.15       |
| im0      | im1      | im1     | 2048  | 383   | 0.18       |
| im0      | im1      | im1     | 2048  | 383   | 0.18       |
| im0      | im1      | im2     | 2048  | 255   | 0.12       |
| im0      | im1      | im2     | 2048  | 255   | 0.12       |
| im1      | im0      | im1     | 2048  | 381   | 0.18       |
| im1      | im0      | im1     | 2048  | 381   | 0.18       |
| im1      | im0      | im2     | 2048  | 254   | 0.12       |
| im1      | im0      | im2     | 2048  | 254   | 0.12       |
| im1      | im1      | im1     | 2048  | 510   | 0.24       |
| im1      | im1      | im1     | 2048  | 510   | 0.24       |
| im1      | im1      | im2     | 2048  | 383   | 0.18       |
| im1      | im1      | im2     | 2048  | 383   | 0.18       |



The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 254   | 0.12       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 510   | 0.24       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 128   | 128   | 1.00       |
| im0      | im1      | im2     | 128   | 128   | 1.00       |
| im0      | im1      | im2     | 1024  | 184   | 0.17       |
| im0      | im1      | im2     | 1024  | 184   | 0.17       |
| im0      | im1      | im2     | 2048  | 255   | 0.12       |
| im0      | im1      | im2     | 2048  | 255   | 0.12       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 255   | 0.12       |

#### 6.175.2.4 nmppsAdd\_64s()

```
void nmppsAdd_64s (
    const nm64s * pSrcVec1,
    const nm64s * pSrcVec2,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im1     | 2048  | 4214  | 2.05       |
| im0      | im0      | im1     | 2048  | 4214  | 2.05       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4152  | 2.02       |
| im0      | im0      | im2     | 2048  | 4152  | 2.02       |
| im0      | im1      | im1     | 2048  | 4463  | 2.17       |
| im0      | im1      | im1     | 2048  | 4463  | 2.17       |
| im0      | im1      | im2     | 2048  | 2355  | 1.14       |
| im0      | im1      | im2     | 2048  | 2355  | 1.14       |
| im1      | im0      | im1     | 2048  | 4401  | 2.14       |
| im1      | im0      | im1     | 2048  | 4401  | 2.14       |
| im1      | im0      | im2     | 2048  | 2354  | 1.14       |
| im1      | im0      | im2     | 2048  | 2354  | 1.14       |
| im1      | im1      | im1     | 2048  | 6450  | 3.14       |
| im1      | im1      | im1     | 2048  | 6450  | 3.14       |
| im1      | im1      | im2     | 2048  | 4403  | 2.14       |
| im1      | im1      | im2     | 2048  | 4403  | 2.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 2354  | 1.14       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 6450  | 3.14       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 128   | 16.00      |
| im0      | im1      | im2     | 8     | 128   | 16.00      |
| im0      | im1      | im2     | 128   | 255   | 1.99       |
| im0      | im1      | im2     | 128   | 255   | 1.99       |
| im0      | im1      | im2     | 1024  | 1235  | 1.20       |
| im0      | im1      | im2     | 1024  | 1235  | 1.20       |
| im0      | im1      | im2     | 2048  | 2355  | 1.14       |
| im0      | im1      | im2     | 2048  | 2355  | 1.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 128   | 16.00      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2355  | 1.14       |

## 6.175.2.5 nmppsAdd\_8s()

```
void nmppsAdd_8s (
    const nm8s * pSrcVec1,
    const nm8s * pSrcVec2,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im1     | 2048  | 630   | 0.30       |
| im0      | im0      | im1     | 2048  | 630   | 0.30       |
| im0      | im0      | im2     | 2048  | 568   | 0.27       |
| im0      | im0      | im2     | 2048  | 568   | 0.27       |
| im0      | im1      | im1     | 2048  | 655   | 0.31       |
| im0      | im1      | im1     | 2048  | 655   | 0.31       |
| im0      | im1      | im2     | 2048  | 395   | 0.19       |
| im0      | im1      | im2     | 2048  | 395   | 0.19       |
| im1      | im0      | im1     | 2048  | 649   | 0.31       |
| im1      | im0      | im1     | 2048  | 649   | 0.31       |
| im1      | im0      | im2     | 2048  | 394   | 0.19       |
| im1      | im0      | im2     | 2048  | 394   | 0.19       |
| im1      | im1      | im1     | 2048  | 906   | 0.44       |
| im1      | im1      | im1     | 2048  | 906   | 0.44       |
| im1      | im1      | im2     | 2048  | 651   | 0.31       |
| im1      | im1      | im2     | 2048  | 651   | 0.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 394   | 0.19       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 906   | 0.44       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 128   | 136   | 1.06       |
| im0      | im1      | im2     | 128   | 136   | 1.06       |
| im0      | im1      | im2     | 1024  | 255   | 0.24       |
| im0      | im1      | im2     | 1024  | 255   | 0.24       |
| im0      | im1      | im2     | 2048  | 395   | 0.19       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 395   | 0.19       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 395   | 0.19       |

## 6.176 nmppsAdd

Сложение двух векторов.

Функции

- void [nmppsAdd\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, nm32f \*pDstVec, int nSize)

### 6.176.1 Подробное описание

Сложение двух векторов.

$$pDstVec[i] = pSrcVec1[i] + pSrcVec2[i],$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Первый входной вектор.      |
| pSrcVec2 | Второй входной вектор.      |
| nSize    | Размер вектора в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.176.2 Функции

#### 6.176.2.1 nmppsAdd\_32f()

```
void nmppsAdd_32f (
    const nm32f * pSrcVec1,
    const nm32f * pSrcVec2,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |
| im1      | im1      | im3     | 2048  | 2015  | 0.98       |
| im1      | im1      | im3     | 2048  | 2015  | 0.98       |
| im1      | im2      | im1     | 2048  | 1998  | 0.97       |
| im1      | im2      | im1     | 2048  | 1998  | 0.97       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im2      | im1      | im1     | 2048  | 1999  | 0.97       |
| im2      | im1      | im1     | 2048  | 1999  | 0.97       |
| im2      | im1      | im3     | 2048  | 1056  | 0.51       |
| im2      | im1      | im3     | 2048  | 1056  | 0.51       |
| im2      | im2      | im1     | 2048  | 2015  | 0.98       |
| im2      | im2      | im1     | 2048  | 2015  | 0.98       |
| im2      | im2      | im3     | 2048  | 2015  | 0.98       |
| im2      | im2      | im3     | 2048  | 2015  | 0.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 8     | 94    | 11.74      |
| im1      | im2      | im3     | 8     | 94    | 11.74      |
| im1      | im2      | im3     | 128   | 109   | 0.85       |
| im1      | im2      | im3     | 128   | 109   | 0.85       |
| im1      | im2      | im3     | 1024  | 544   | 0.53       |
| im1      | im2      | im3     | 1024  | 544   | 0.53       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 8     | 94    | 11.74      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |

## 6.177 nmppsAdd\_AddC

Сложение двух векторов с прибавлением константы.

Функции

- void `nmppsAdd_AddC_32s` (`nm32s` \*pSrcVec1, `nm32s` \*pSrcVec2, int nVal, `nm32s` \*pDstVec, int nSize)

### 6.177.1 Подробное описание

Сложение двух векторов с прибавлением константы.

$$pDstVec[i] = pSrcVec1[i] + pSrcVec2[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| nVal     | Добавляемая константа.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.177.2 Функции

#### 6.177.2.1 nmppsAdd\_AddC\_32s()

```
void nmppsAdd_AddC_32s (
    nm32s * pSrcVec1,
```



```

nm32s * pSrcVec2,
int nVal,
nm32s * pDstVec,
int nSize )

```

Performance table 0

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 3    | im0      | im2     | 2048  | 2171  | 1.06       |
| im0      | 3    | im0      | im2     | 2048  | 2171  | 1.06       |
| im0      | 3    | im0      | im3     | 2048  | 2171  | 1.06       |
| im0      | 3    | im0      | im3     | 2048  | 2171  | 1.06       |
| im0      | 3    | im1      | im2     | 2048  | 2353  | 1.14       |
| im0      | 3    | im1      | im2     | 2048  | 2353  | 1.14       |
| im0      | 3    | im1      | im3     | 2048  | 2353  | 1.14       |
| im0      | 3    | im1      | im3     | 2048  | 2353  | 1.14       |
| im1      | 3    | im0      | im2     | 2048  | 2179  | 1.06       |
| im1      | 3    | im0      | im2     | 2048  | 2179  | 1.06       |
| im1      | 3    | im0      | im3     | 2048  | 2179  | 1.06       |
| im1      | 3    | im0      | im3     | 2048  | 2179  | 1.06       |
| im1      | 3    | im1      | im2     | 2048  | 2275  | 1.11       |
| im1      | 3    | im1      | im2     | 2048  | 2275  | 1.11       |
| im1      | 3    | im1      | im3     | 2048  | 2275  | 1.11       |
| im1      | 3    | im1      | im3     | 2048  | 2275  | 1.11       |

The best configuration:

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 3    | im0      | im2     | 2048  | 2171  | 1.06       |

The worst configuration:

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 3    | im1      | im2     | 2048  | 2353  | 1.14       |

Performance table 1

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 5    | im1      | im2     | 8     | 139   | 17.37      |
| im0      | 5    | im1      | im2     | 8     | 139   | 17.37      |
| im0      | 5    | im1      | im2     | 128   | 257   | 2.00       |
| im0      | 5    | im1      | im2     | 128   | 257   | 2.00       |
| im0      | 5    | im1      | im2     | 1024  | 1233  | 1.20       |
| im0      | 5    | im1      | im2     | 1024  | 1233  | 1.20       |
| im0      | 5    | im1      | im2     | 2048  | 2353  | 1.14       |
| im0      | 5    | im1      | im2     | 2048  | 2353  | 1.14       |

The best configuration:

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 5    | im1      | im2     | 8     | 139   | 17.37      |

The worst configuration:

| pSrcVec1 | nVal | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|------|----------|---------|-------|-------|------------|
| im0      | 5    | im1      | im2     | 2048  | 2353  | 1.14       |

## 6.178 nmppsSubC

Вычитание константы из вектора.

### Функции

- void `nmppsSubC_4s` (const `nm4s` \*pSrcVec, `int4b` nVal, `nm4s` \*pDstVec, int nSize)
- void `nmppsSubC_8s` (const `nm8s` \*pSrcVec, `int8b` nVal, `nm8s` \*pDstVec, int nSize)
- void `nmppsSubC_16s` (const `nm16s` \*pSrcVec, `int16b` nVal, `nm16s` \*pDstVec, int nSize)
- void `nmppsSubC_32s` (const `nm32s` \*pSrcVec, `int32b` nVal, `nm32s` \*pDstVec, int nSize)
- void `nmppsSubC_64s` (const `nm64s` \*pSrcVec, `int64b` nVal, `nm64s` \*pDstVec, int nSize)

### 6.178.1 Подробное описание

Вычитание константы из вектора.

$$pDstVec[i] = pSrcVec[i] - nVal$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Вычитаемая константа.        |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.178.2 Функции

## 6.178.2.1 nmppsSubC\_16s()

```
void nmppsSubC_16s (
    const nm16s * pSrcVec,
    int16b nVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 1085  | 0.52       |
| im0     | 3    | im0     | 2048  | 1085  | 0.52       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |
| im1     | 3    | im0     | 2048  | 699   | 0.34       |
| im1     | 3    | im0     | 2048  | 699   | 0.34       |
| im1     | 3    | im1     | 2048  | 1209  | 0.59       |
| im1     | 3    | im1     | 2048  | 1209  | 0.59       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 695   | 0.33       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 1209  | 0.59       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 128   | 185   | 1.44       |
| im0     | 3    | im1     | 128   | 185   | 1.44       |
| im0     | 3    | im1     | 1024  | 423   | 0.41       |
| im0     | 3    | im1     | 1024  | 423   | 0.41       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 695   | 0.33       |

## 6.178.2.2 nmppsSubC\_32s()

```
void nmppsSubC_32s (
    const nm32s * pSrcVec,
    int32b nVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 2109  | 1.02       |
| im0     | 3    | im0     | 2048  | 2109  | 1.02       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |
| im1     | 3    | im0     | 2048  | 1243  | 0.60       |
| im1     | 3    | im0     | 2048  | 1243  | 0.60       |
| im1     | 3    | im1     | 2048  | 2265  | 1.10       |
| im1     | 3    | im1     | 2048  | 2265  | 1.10       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 2265  | 1.10       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 130   | 16.25      |
| im0     | 3    | im1     | 8     | 130   | 16.25      |
| im0     | 3    | im1     | 128   | 221   | 1.72       |
| im0     | 3    | im1     | 128   | 221   | 1.72       |
| im0     | 3    | im1     | 1024  | 695   | 0.67       |
| im0     | 3    | im1     | 1024  | 695   | 0.67       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 130   | 16.25      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |

### 6.178.2.3 nmppsSubC\_4s()

```
void nmppsSubC_4s (
    const nm4s * pSrcVec,
    int4b nVal,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 320   | 0.15       |
| im0     | 3    | im0     | 2048  | 320   | 0.15       |
| im0     | 3    | im1     | 2048  | 288   | 0.14       |
| im0     | 3    | im1     | 2048  | 288   | 0.14       |
| im1     | 3    | im0     | 2048  | 292   | 0.14       |
| im1     | 3    | im0     | 2048  | 292   | 0.14       |
| im1     | 3    | im1     | 2048  | 418   | 0.20       |
| im1     | 3    | im1     | 2048  | 418   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 288   | 0.14       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 418   | 0.20       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 128   | 143   | 1.11       |
| im0     | 3    | im1     | 128   | 143   | 1.11       |

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 1024  | 224   | 0.21       |
| im0     | 3    | im1     | 1024  | 224   | 0.21       |
| im0     | 3    | im1     | 2048  | 288   | 0.14       |
| im0     | 3    | im1     | 2048  | 288   | 0.14       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 288   | 0.14       |

#### 6.178.2.4 nmppsSubC\_64s()

```
void nmppsSubC_64s (
    const nm64s * pSrcVec,
    int64b nVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 4151  | 2.02       |
| im0     | 3    | im0     | 2048  | 4151  | 2.02       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |
| im1     | 3    | im0     | 2048  | 2325  | 1.13       |
| im1     | 3    | im0     | 2048  | 2325  | 1.13       |
| im1     | 3    | im1     | 2048  | 4371  | 2.13       |
| im1     | 3    | im1     | 2048  | 4371  | 2.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 4371  | 2.13       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 138   | 17.25      |
| im0     | 3    | im1     | 8     | 138   | 17.25      |
| im0     | 3    | im1     | 128   | 281   | 2.19       |
| im0     | 3    | im1     | 128   | 281   | 2.19       |
| im0     | 3    | im1     | 1024  | 1233  | 1.20       |
| im0     | 3    | im1     | 1024  | 1233  | 1.20       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 138   | 17.25      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |

#### 6.178.2.5 nmppsSubC\_8s()

```
void nmppsSubC_8s (
    const nm8s * pSrcVec,
    int8b nVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 574   | 0.28       |
| im0     | 3    | im0     | 2048  | 574   | 0.28       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |
| im1     | 3    | im0     | 2048  | 428   | 0.20       |
| im1     | 3    | im0     | 2048  | 428   | 0.20       |
| im1     | 3    | im1     | 2048  | 682   | 0.33       |
| im1     | 3    | im1     | 2048  | 682   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 424   | 0.20       |



The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 682   | 0.33       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 128   | 156   | 1.21       |
| im0     | 3    | im1     | 128   | 156   | 1.21       |
| im0     | 3    | im1     | 1024  | 286   | 0.27       |
| im0     | 3    | im1     | 1024  | 286   | 0.27       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 424   | 0.20       |

## 6.179 nmppsSubCRev

Вычитание константы из вектора с переменной знака элементов вектора.

### Функции

- void `nmppsSubCRev_8s` (const `nm8s` \*pSrcVec, `int8b` nVal, `nm8s` \*pDstVec, int nSize)
- void `nmppsSubCRev_16s` (const `nm16s` \*pSrcVec, `int16b` nVal, `nm16s` \*pDstVec, int nSize)
- void `nmppsSubCRev_32s` (const `nm32s` \*pSrcVec, `int32b` nVal, `nm32s` \*pDstVec, int nSize)
- void `nmppsSubCRev_64s` (const `nm64s` \*pSrcVec, `int64b` nVal, `nm64s` \*pDstVec, int nSize)

### 6.179.1 Подробное описание

Вычитание константы из вектора с переменной знака элементов вектора.

$$pDstVec[i] = nVal - pSrcVec[i],$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа.                   |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.179.2 Функции

## 6.179.2.1 nmppsSubCRev\_16s()

```
void nmppsSubCRev_16s (
    const nm16s * pSrcVec,
    int16b nVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 1085  | 0.52       |
| im0     | 5    | im0     | 2048  | 1085  | 0.52       |
| im0     | 5    | im1     | 2048  | 695   | 0.33       |
| im0     | 5    | im1     | 2048  | 695   | 0.33       |
| im1     | 5    | im0     | 2048  | 699   | 0.34       |
| im1     | 5    | im0     | 2048  | 699   | 0.34       |
| im1     | 5    | im1     | 2048  | 1209  | 0.59       |
| im1     | 5    | im1     | 2048  | 1209  | 0.59       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 695   | 0.33       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 1209  | 0.59       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 128   | 185   | 1.44       |
| im0     | 3    | im1     | 128   | 185   | 1.44       |
| im0     | 3    | im1     | 1024  | 423   | 0.41       |
| im0     | 3    | im1     | 1024  | 423   | 0.41       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |
| im0     | 3    | im1     | 2048  | 695   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 695   | 0.33       |

### 6.179.2.2 nmppsSubCRev\_32s()

```
void nmppsSubCRev_32s (
    const nm32s * pSrcVec,
    int32b nVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 2109  | 1.02       |
| im0     | 5    | im0     | 2048  | 2109  | 1.02       |
| im0     | 5    | im1     | 2048  | 1239  | 0.60       |
| im0     | 5    | im1     | 2048  | 1239  | 0.60       |
| im1     | 5    | im0     | 2048  | 1243  | 0.60       |
| im1     | 5    | im0     | 2048  | 1243  | 0.60       |
| im1     | 5    | im1     | 2048  | 2265  | 1.10       |
| im1     | 5    | im1     | 2048  | 2265  | 1.10       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 1239  | 0.60       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 2265  | 1.10       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 130   | 16.25      |
| im0     | 3    | im1     | 8     | 130   | 16.25      |
| im0     | 3    | im1     | 128   | 221   | 1.72       |
| im0     | 3    | im1     | 128   | 221   | 1.72       |
| im0     | 3    | im1     | 1024  | 695   | 0.67       |
| im0     | 3    | im1     | 1024  | 695   | 0.67       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 130   | 16.25      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1239  | 0.60       |

### 6.179.2.3 nmppsSubCRev\_64s()

```
void nmppsSubCRev_64s (
    const nm64s * pSrcVec,
    int64b nVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 4151  | 2.02       |
| im0     | 5    | im0     | 2048  | 4151  | 2.02       |
| im0     | 5    | im1     | 2048  | 2321  | 1.13       |
| im0     | 5    | im1     | 2048  | 2321  | 1.13       |
| im1     | 5    | im0     | 2048  | 2325  | 1.13       |
| im1     | 5    | im0     | 2048  | 2325  | 1.13       |
| im1     | 5    | im1     | 2048  | 4371  | 2.13       |
| im1     | 5    | im1     | 2048  | 4371  | 2.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 2321  | 1.13       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 4371  | 2.13       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 138   | 17.25      |
| im0     | 3    | im1     | 8     | 138   | 17.25      |
| im0     | 3    | im1     | 128   | 281   | 2.19       |
| im0     | 3    | im1     | 128   | 281   | 2.19       |

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 1024  | 1233  | 1.20       |
| im0     | 3    | im1     | 1024  | 1233  | 1.20       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 138   | 17.25      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2321  | 1.13       |

#### 6.179.2.4 nmppsSubCRev\_8s()

```
void nmppsSubCRev_8s (
    const nm8s * pSrcVec,
    int8b nVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 574   | 0.28       |
| im0     | 5    | im0     | 2048  | 574   | 0.28       |
| im0     | 5    | im1     | 2048  | 424   | 0.20       |
| im0     | 5    | im1     | 2048  | 424   | 0.20       |
| im1     | 5    | im0     | 2048  | 428   | 0.20       |
| im1     | 5    | im0     | 2048  | 428   | 0.20       |
| im1     | 5    | im1     | 2048  | 682   | 0.33       |
| im1     | 5    | im1     | 2048  | 682   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 424   | 0.20       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 682   | 0.33       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 8     | 126   | 15.75      |
| im0     | 3    | im1     | 128   | 156   | 1.21       |
| im0     | 3    | im1     | 128   | 156   | 1.21       |
| im0     | 3    | im1     | 1024  | 286   | 0.27       |
| im0     | 3    | im1     | 1024  | 286   | 0.27       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |
| im0     | 3    | im1     | 2048  | 424   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 424   | 0.20       |

## 6.180 nmppsSub

Вычитание двух вектров.

Функции

- void nmppsSub\_4s (const nm4s \*pSrcVec1, nm4s \*pSrcVec2, nm4s \*pDstVec, int nSize)
- void nmppsSub\_8s (const nm8s \*pSrcVec1, nm8s \*pSrcVec2, nm8s \*pDstVec, int nSize)
- void nmppsSub\_16s (const nm16s \*pSrcVec1, nm16s \*pSrcVec2, nm16s \*pDstVec, int nSize)
- void nmppsSub\_32s (const nm32s \*pSrcVec1, nm32s \*pSrcVec2, nm32s \*pDstVec, int nSize)
- void nmppsSub\_64s (const nm64s \*pSrcVec1, nm64s \*pSrcVec2, nm64s \*pDstVec, int nSize)

### 6.180.1 Подробное описание

Вычитание двух вектров.

$$pDstVec[i] = pSrcVec1[i] - pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Уменьшаемый вектор.          |
| pSrcVec2 | Вычитаемый вектор.           |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.180.2 Функции



## 6.180.2.1 nmppsSub\_16s()

```
void nmppsSub_16s (
    const nm16s * pSrcVec1,
    nm16s * pSrcVec2,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1084  | 0.52       |
| im0      | im0      | im2     | 2048  | 1084  | 0.52       |
| im0      | im0      | im3     | 2048  | 1084  | 0.52       |
| im0      | im0      | im3     | 2048  | 1084  | 0.52       |
| im0      | im1      | im2     | 2048  | 665   | 0.32       |
| im0      | im1      | im2     | 2048  | 665   | 0.32       |
| im0      | im1      | im3     | 2048  | 665   | 0.32       |
| im0      | im1      | im3     | 2048  | 665   | 0.32       |
| im1      | im0      | im2     | 2048  | 660   | 0.32       |
| im1      | im0      | im2     | 2048  | 660   | 0.32       |
| im1      | im0      | im3     | 2048  | 660   | 0.32       |
| im1      | im0      | im3     | 2048  | 660   | 0.32       |
| im1      | im1      | im2     | 2048  | 1173  | 0.57       |
| im1      | im1      | im2     | 2048  | 1173  | 0.57       |
| im1      | im1      | im3     | 2048  | 1173  | 0.57       |
| im1      | im1      | im3     | 2048  | 1173  | 0.57       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 660   | 0.32       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 1173  | 0.57       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 128   | 152   | 1.18       |
| im0      | im1      | im2     | 128   | 152   | 1.18       |
| im0      | im1      | im2     | 1024  | 393   | 0.38       |
| im0      | im1      | im2     | 1024  | 393   | 0.38       |
| im0      | im1      | im2     | 2048  | 665   | 0.32       |
| im0      | im1      | im2     | 2048  | 665   | 0.32       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 665   | 0.32       |

#### 6.180.2.2 nmppsSub\_32s()

```
void nmppsSub_32s (
    const nm32s * pSrcVec1,
    nm32s * pSrcVec2,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2108  | 1.02       |
| im0      | im0      | im2     | 2048  | 2108  | 1.02       |
| im0      | im0      | im3     | 2048  | 2108  | 1.02       |
| im0      | im0      | im3     | 2048  | 2108  | 1.02       |
| im0      | im1      | im2     | 2048  | 1209  | 0.59       |
| im0      | im1      | im2     | 2048  | 1209  | 0.59       |
| im0      | im1      | im3     | 2048  | 1209  | 0.59       |
| im0      | im1      | im3     | 2048  | 1209  | 0.59       |
| im1      | im0      | im2     | 2048  | 1204  | 0.58       |
| im1      | im0      | im2     | 2048  | 1204  | 0.58       |
| im1      | im0      | im3     | 2048  | 1204  | 0.58       |
| im1      | im0      | im3     | 2048  | 1204  | 0.58       |
| im1      | im1      | im2     | 2048  | 2229  | 1.08       |
| im1      | im1      | im2     | 2048  | 2229  | 1.08       |
| im1      | im1      | im3     | 2048  | 2229  | 1.08       |
| im1      | im1      | im3     | 2048  | 2229  | 1.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 1204  | 0.58       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 2229  | 1.08       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 122   | 15.25      |
| im0      | im1      | im2     | 8     | 122   | 15.25      |
| im0      | im1      | im2     | 128   | 189   | 1.47       |
| im0      | im1      | im2     | 128   | 189   | 1.47       |
| im0      | im1      | im2     | 1024  | 665   | 0.64       |
| im0      | im1      | im2     | 1024  | 665   | 0.64       |
| im0      | im1      | im2     | 2048  | 1209  | 0.59       |
| im0      | im1      | im2     | 2048  | 1209  | 0.59       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 122   | 15.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1209  | 0.59       |

### 6.180.2.3 nmppsSub\_64s()

```
void nmppsSub_64s (
    const nm64s * pSrcVec1,
    nm64s * pSrcVec2,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4154  | 2.02       |
| im0      | im0      | im2     | 2048  | 4154  | 2.02       |
| im0      | im0      | im3     | 2048  | 4154  | 2.02       |
| im0      | im0      | im3     | 2048  | 4154  | 2.02       |
| im0      | im1      | im2     | 2048  | 2295  | 1.12       |
| im0      | im1      | im2     | 2048  | 2295  | 1.12       |
| im0      | im1      | im3     | 2048  | 2295  | 1.12       |
| im0      | im1      | im3     | 2048  | 2295  | 1.12       |
| im1      | im0      | im2     | 2048  | 2290  | 1.11       |
| im1      | im0      | im2     | 2048  | 2290  | 1.11       |
| im1      | im0      | im3     | 2048  | 2290  | 1.11       |
| im1      | im0      | im3     | 2048  | 2290  | 1.11       |
| im1      | im1      | im2     | 2048  | 4339  | 2.11       |
| im1      | im1      | im2     | 2048  | 4339  | 2.11       |
| im1      | im1      | im3     | 2048  | 4339  | 2.11       |
| im1      | im1      | im3     | 2048  | 4339  | 2.11       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 2290  | 1.11       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 4339  | 2.11       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 126   | 15.75      |
| im0      | im1      | im2     | 8     | 126   | 15.75      |
| im0      | im1      | im2     | 128   | 255   | 1.99       |
| im0      | im1      | im2     | 128   | 255   | 1.99       |
| im0      | im1      | im2     | 1024  | 1207  | 1.17       |
| im0      | im1      | im2     | 1024  | 1207  | 1.17       |
| im0      | im1      | im2     | 2048  | 2295  | 1.12       |
| im0      | im1      | im2     | 2048  | 2295  | 1.12       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2295  | 1.12       |

#### 6.180.2.4 nmppsSub\_8s()

```
void nmppsSub_8s (
    const nm8s * pSrcVec1,
    nm8s * pSrcVec2,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 572   | 0.27       |
| im0      | im0      | im2     | 2048  | 572   | 0.27       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im3     | 2048  | 572   | 0.27       |
| im0      | im0      | im3     | 2048  | 572   | 0.27       |
| im0      | im1      | im2     | 2048  | 393   | 0.19       |
| im0      | im1      | im2     | 2048  | 393   | 0.19       |
| im0      | im1      | im3     | 2048  | 393   | 0.19       |
| im0      | im1      | im3     | 2048  | 393   | 0.19       |
| im1      | im0      | im2     | 2048  | 388   | 0.18       |
| im1      | im0      | im2     | 2048  | 388   | 0.18       |
| im1      | im0      | im3     | 2048  | 388   | 0.18       |
| im1      | im0      | im3     | 2048  | 388   | 0.18       |
| im1      | im1      | im2     | 2048  | 645   | 0.31       |
| im1      | im1      | im2     | 2048  | 645   | 0.31       |
| im1      | im1      | im3     | 2048  | 645   | 0.31       |
| im1      | im1      | im3     | 2048  | 645   | 0.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 388   | 0.18       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 645   | 0.31       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 128   | 136   | 1.06       |
| im0      | im1      | im2     | 128   | 136   | 1.06       |
| im0      | im1      | im2     | 1024  | 257   | 0.25       |
| im0      | im1      | im2     | 1024  | 257   | 0.25       |
| im0      | im1      | im2     | 2048  | 393   | 0.19       |
| im0      | im1      | im2     | 2048  | 393   | 0.19       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 393   | 0.19       |

## 6.181 nmppsSub

Вычитание двух вектров.

Функции

- void [nmppsSub\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, nm32f \*pDstVec, int nSize)

### 6.181.1 Подробное описание

Вычитание двух вектров.

$$pDstVec[i] = pSrcVec1[i] - pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Уменьшаемый вектор.          |
| pSrcVec2 | Вычитаемый вектор.           |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.181.2 Функции

#### 6.181.2.1 nmppsSub\_32f()

```
void nmppsSub_32f (
    const nm32f * pSrcVec1,
    const nm32f * pSrcVec2,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |
| im1      | im1      | im3     | 2048  | 2015  | 0.98       |
| im1      | im1      | im3     | 2048  | 2015  | 0.98       |
| im1      | im2      | im1     | 2048  | 1998  | 0.97       |
| im1      | im2      | im1     | 2048  | 1998  | 0.97       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im2      | im1      | im1     | 2048  | 1999  | 0.97       |
| im2      | im1      | im1     | 2048  | 1999  | 0.97       |
| im2      | im1      | im3     | 2048  | 1056  | 0.51       |
| im2      | im1      | im3     | 2048  | 1056  | 0.51       |
| im2      | im2      | im1     | 2048  | 2015  | 0.98       |
| im2      | im2      | im1     | 2048  | 2015  | 0.98       |
| im2      | im2      | im3     | 2048  | 2015  | 0.98       |
| im2      | im2      | im3     | 2048  | 2015  | 0.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im1     | 2048  | 2958  | 1.44       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 8     | 94    | 11.75      |
| im1      | im2      | im3     | 8     | 94    | 11.75      |
| im1      | im2      | im3     | 128   | 109   | 0.85       |
| im1      | im2      | im3     | 128   | 109   | 0.85       |
| im1      | im2      | im3     | 1024  | 544   | 0.53       |
| im1      | im2      | im3     | 1024  | 544   | 0.53       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 8     | 94    | 11.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 1056  | 0.51       |



## 6.182 nmppsAbsDiff

Вычисление вектора модулей разности элементов двух векторов.

Функции

- void `nmppsAbsDiff_8s` (const `nm8s` \*pSrcVec1, `nm8s` \*pSrcVec2, `nm8s` \*pDstVec, int nSize)
- void `nmppsAbsDiff_16s` (const `nm16s` \*pSrcVec1, `nm16s` \*pSrcVec2, `nm16s` \*pDstVec, int nSize)
- void `nmppsAbsDiff_32s` (const `nm32s` \*pSrcVec1, `nm32s` \*pSrcVec2, `nm32s` \*pDstVec, int nSize)
- void `nmppsAbsDiff_64s` (const `nm64s` \*pSrcVec1, `nm64s` \*pSrcVec2, `nm64s` \*pDstVec, int nSize)

### 6.182.1 Подробное описание

Вычисление вектора модулей разности элементов двух векторов.

$$pDstVec[i] = abs\{pSrcVec1[i] - pSrcVec2[i]\},$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Входной вектор.              |
| pSrcVec2 | Вычитаемый вектор.           |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

Restrictions:

Разность элементов векторов не должна быть равна минимальному значению для соответствующего типа (т.е. -128, -2<sup>15</sup> и т.д). В противном случае, абсолютное значение для таких элементов вычисляется не верно, давая на выходе то же самое число.

### 6.182.2 Функции

## 6.182.2.1 nmppsAbsDiff\_16s()

```
void nmppsAbsDiff_16s (
    const nm16s * pSrcVec1,
    nm16s * pSrcVec2,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2158  | 1.05       |
| im0      | im0      | im2     | 2048  | 2158  | 1.05       |
| im0      | im0      | im3     | 2048  | 2158  | 1.05       |
| im0      | im0      | im3     | 2048  | 2158  | 1.05       |
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |
| im0      | im1      | im3     | 2048  | 2202  | 1.07       |
| im0      | im1      | im3     | 2048  | 2202  | 1.07       |
| im1      | im0      | im2     | 2048  | 2161  | 1.05       |
| im1      | im0      | im2     | 2048  | 2161  | 1.05       |
| im1      | im0      | im3     | 2048  | 2161  | 1.05       |
| im1      | im0      | im3     | 2048  | 2161  | 1.05       |
| im1      | im1      | im2     | 2048  | 2176  | 1.06       |
| im1      | im1      | im2     | 2048  | 2176  | 1.06       |
| im1      | im1      | im3     | 2048  | 2176  | 1.06       |
| im1      | im1      | im3     | 2048  | 2176  | 1.06       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2158  | 1.05       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 140   | 17.50      |
| im0      | im1      | im2     | 8     | 140   | 17.50      |
| im0      | im1      | im2     | 128   | 223   | 1.74       |
| im0      | im1      | im2     | 128   | 223   | 1.74       |
| im0      | im1      | im2     | 1024  | 1146  | 1.11       |
| im0      | im1      | im2     | 1024  | 1146  | 1.11       |
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 140   | 17.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2202  | 1.07       |

### 6.182.2.2 nmppsAbsDiff\_32s()

```
void nmppsAbsDiff_32s (
    const nm32s * pSrcVec1,
    nm32s * pSrcVec2,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4219  | 2.06       |
| im0      | im0      | im2     | 2048  | 4219  | 2.06       |
| im0      | im0      | im3     | 2048  | 4219  | 2.06       |
| im0      | im0      | im3     | 2048  | 4219  | 2.06       |
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |
| im0      | im1      | im3     | 2048  | 4311  | 2.10       |
| im0      | im1      | im3     | 2048  | 4311  | 2.10       |
| im1      | im0      | im2     | 2048  | 4222  | 2.06       |
| im1      | im0      | im2     | 2048  | 4222  | 2.06       |
| im1      | im0      | im3     | 2048  | 4222  | 2.06       |
| im1      | im0      | im3     | 2048  | 4222  | 2.06       |
| im1      | im1      | im2     | 2048  | 4253  | 2.07       |
| im1      | im1      | im2     | 2048  | 4253  | 2.07       |
| im1      | im1      | im3     | 2048  | 4253  | 2.07       |
| im1      | im1      | im3     | 2048  | 4253  | 2.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4219  | 2.06       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 147   | 18.37      |
| im0      | im1      | im2     | 8     | 147   | 18.37      |
| im0      | im1      | im2     | 128   | 351   | 2.74       |
| im0      | im1      | im2     | 128   | 351   | 2.74       |
| im0      | im1      | im2     | 1024  | 2199  | 2.14       |
| im0      | im1      | im2     | 1024  | 2199  | 2.14       |
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 147   | 18.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4311  | 2.10       |

### 6.182.2.3 nmppsAbsDiff\_64s()

```
void nmppsAbsDiff_64s (
    const nm64s * pSrcVec1,
    nm64s * pSrcVec2,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 8349  | 4.07       |
| im0      | im0      | im2     | 2048  | 8349  | 4.07       |
| im0      | im0      | im3     | 2048  | 8349  | 4.07       |
| im0      | im0      | im3     | 2048  | 8349  | 4.07       |
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |
| im0      | im1      | im3     | 2048  | 8537  | 4.16       |
| im0      | im1      | im3     | 2048  | 8537  | 4.16       |
| im1      | im0      | im2     | 2048  | 8352  | 4.07       |
| im1      | im0      | im2     | 2048  | 8352  | 4.07       |
| im1      | im0      | im3     | 2048  | 8352  | 4.07       |
| im1      | im0      | im3     | 2048  | 8352  | 4.07       |
| im1      | im1      | im2     | 2048  | 8415  | 4.10       |
| im1      | im1      | im2     | 2048  | 8415  | 4.10       |
| im1      | im1      | im3     | 2048  | 8415  | 4.10       |
| im1      | im1      | im3     | 2048  | 8415  | 4.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 8349  | 4.07       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 157   | 19.62      |
| im0      | im1      | im2     | 8     | 157   | 19.62      |
| im0      | im1      | im2     | 128   | 617   | 4.82       |
| im0      | im1      | im2     | 128   | 617   | 4.82       |
| im0      | im1      | im2     | 1024  | 4313  | 4.21       |
| im0      | im1      | im2     | 1024  | 4313  | 4.21       |
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 157   | 19.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 8537  | 4.16       |

#### 6.182.2.4 nmppsAbsDiff\_8s()

```
void nmppsAbsDiff_8s (
    const nm8s * pSrcVec1,
    nm8s * pSrcVec2,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1130  | 0.55       |
| im0      | im0      | im2     | 2048  | 1130  | 0.55       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im3     | 2048  | 1130  | 0.55       |
| im0      | im0      | im3     | 2048  | 1130  | 0.55       |
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |
| im0      | im1      | im3     | 2048  | 1150  | 0.56       |
| im0      | im1      | im3     | 2048  | 1150  | 0.56       |
| im1      | im0      | im2     | 2048  | 1133  | 0.55       |
| im1      | im0      | im2     | 2048  | 1133  | 0.55       |
| im1      | im0      | im3     | 2048  | 1133  | 0.55       |
| im1      | im0      | im3     | 2048  | 1133  | 0.55       |
| im1      | im1      | im2     | 2048  | 1140  | 0.55       |
| im1      | im1      | im2     | 2048  | 1140  | 0.55       |
| im1      | im1      | im3     | 2048  | 1140  | 0.55       |
| im1      | im1      | im3     | 2048  | 1140  | 0.55       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1130  | 0.55       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 141   | 17.62      |
| im0      | im1      | im2     | 8     | 141   | 17.62      |
| im0      | im1      | im2     | 128   | 186   | 1.45       |
| im0      | im1      | im2     | 128   | 186   | 1.45       |
| im0      | im1      | im2     | 1024  | 622   | 0.60       |
| im0      | im1      | im2     | 1024  | 622   | 0.60       |
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 141   | 17.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1150  | 0.56       |

## 6.183 nmppsAbsDiff1

Функция логического вычисления модулей разностей элементов двух векторов.

Функции

- void `nmppsAbsDiff1_8s` (`nm8s` \*pSrcVec1, `nm8s` \*pSrcVec2, `nm8s` \*pDstVec, int nSize)

### 6.183.1 Подробное описание

Функция логического вычисления модулей разностей элементов двух векторов.

$$pDstVec[i] = \begin{cases} pSrcVec1[i] - pSrcVec2[i], & \text{if } pSrcVec1[i] - pSrcVec2[i] \geq 0 \\ pSrcVec1[i] - pSrcVec2[i] - 1, & \text{if } pSrcVec1[i] - pSrcVec2[i] < 0 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Входной вектор.              |
| pSrcVec2 | Вычитаемый вектор.           |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

Restrictions:

Разность элементов векторов не должна быть равна минимальному значению для соответствующего типа (т.е. -128, -2<sup>15</sup> и т.д.). В противном случае, абсолютное значение для таких элементов вычисляется не верно, давая на выходе то же самое число.

### 6.183.2 Функции

## 6.183.2.1 nmppsAbsDiff1\_8s()

```
void nmppsAbsDiff1_8s (
    nm8s * pSrcVec1,
    nm8s * pSrcVec2,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 844   | 0.41       |
| im0      | im0      | im2     | 2048  | 844   | 0.41       |
| im0      | im0      | im3     | 2048  | 844   | 0.41       |
| im0      | im0      | im3     | 2048  | 844   | 0.41       |
| im0      | im1      | im2     | 2048  | 903   | 0.44       |
| im0      | im1      | im2     | 2048  | 903   | 0.44       |
| im0      | im1      | im3     | 2048  | 903   | 0.44       |
| im0      | im1      | im3     | 2048  | 903   | 0.44       |
| im1      | im0      | im2     | 2048  | 850   | 0.41       |
| im1      | im0      | im2     | 2048  | 850   | 0.41       |
| im1      | im0      | im3     | 2048  | 850   | 0.41       |
| im1      | im0      | im3     | 2048  | 850   | 0.41       |
| im1      | im1      | im2     | 2048  | 891   | 0.43       |
| im1      | im1      | im2     | 2048  | 891   | 0.43       |
| im1      | im1      | im3     | 2048  | 891   | 0.43       |
| im1      | im1      | im3     | 2048  | 891   | 0.43       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 844   | 0.41       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 903   | 0.44       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 141   | 17.62      |
| im0      | im1      | im2     | 8     | 141   | 17.62      |
| im0      | im1      | im2     | 128   | 140   | 1.09       |
| im0      | im1      | im2     | 128   | 140   | 1.09       |
| im0      | im1      | im2     | 1024  | 401   | 0.39       |
| im0      | im1      | im2     | 1024  | 401   | 0.39       |
| im0      | im1      | im2     | 2048  | 903   | 0.44       |
| im0      | im1      | im2     | 2048  | 903   | 0.44       |



The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 128   | 140   | 1.09       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 903   | 0.44       |

## 6.184 nmppsMulC

Умножение вектора на константу.

### Функции

- void nmppsMulC\_8s (const nm8s \*pSrcVec, int8b nVal, nm8s \*pDstVec, int nSize)
- void nmppsMulC\_8s16s (const nm8s \*pSrcVec, int16b nVal, nm16s \*pDstVec, int nSize)
- void nmppsMulC\_16s (const nm16s \*pSrcVec, int16b nVal, nm16s \*pDstVec, int nSize)
- void nmppsMulC\_16s32s (const nm16s \*pSrcVec, int32b nVal, nm32s \*pDstVec, int nSize)
- void nmppsMulC\_32s (const nm32s \*pSrcVec, int32b nVal, nm32s \*pDstVec, int nSize)
- void nmppsMulC\_32s64s (const nm32s \*pSrcVec, int64b nVal, nm64s \*pDstVec, int nSize)
- void nmppsMulC\_64s (const nm64s \*pSrcVec, int64b nVal, nm64s \*pDstVec, int nSize)
- void nmppsMulC\_2s16s (const nm2s \*pSrcVec, int16b nVal, nm16s \*pDstVec, int nSize)

### 6.184.1 Подробное описание

Умножение вектора на константу.

$$pDstVec[i] = nVal \cdot pSrcVec[i],$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа-множитель.         |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.184.2 Функции

## 6.184.2.1 nmppsMulC\_16s()

```
void nmppsMulC_16s (
    const nm16s * pSrcVec,
    int16b nVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 1096  | 0.53       |
| im0     | 3    | im0     | 2048  | 1096  | 0.53       |
| im0     | 3    | im1     | 2048  | 685   | 0.33       |
| im0     | 3    | im1     | 2048  | 685   | 0.33       |
| im1     | 3    | im0     | 2048  | 683   | 0.33       |
| im1     | 3    | im0     | 2048  | 683   | 0.33       |
| im1     | 3    | im1     | 2048  | 1195  | 0.58       |
| im1     | 3    | im1     | 2048  | 1195  | 0.58       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 683   | 0.33       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 1195  | 0.58       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 141   | 17.62      |
| im0     | 3    | im1     | 8     | 141   | 17.62      |
| im0     | 3    | im1     | 128   | 176   | 1.37       |
| im0     | 3    | im1     | 128   | 176   | 1.37       |
| im0     | 3    | im1     | 1024  | 415   | 0.40       |
| im0     | 3    | im1     | 1024  | 415   | 0.40       |
| im0     | 3    | im1     | 2048  | 685   | 0.33       |
| im0     | 3    | im1     | 2048  | 685   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 141   | 17.62      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 685   | 0.33       |

#### 6.184.2.2 nmppsMulC\_16s32s()

```
void nmppsMulC_16s32s (
    const nm16s * pSrcVec,
    int32b nVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 2131  | 1.04       |
| im0     | 3    | im0     | 2048  | 2131  | 1.04       |
| im0     | 3    | im1     | 2048  | 1270  | 0.62       |
| im0     | 3    | im1     | 2048  | 1270  | 0.62       |
| im1     | 3    | im0     | 2048  | 1266  | 0.61       |
| im1     | 3    | im0     | 2048  | 1266  | 0.61       |
| im1     | 3    | im1     | 2048  | 2304  | 1.12       |
| im1     | 3    | im1     | 2048  | 2304  | 1.12       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 1266  | 0.61       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 2304  | 1.12       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 182   | 22.75      |
| im0     | 3    | im1     | 8     | 182   | 22.75      |
| im0     | 3    | im1     | 128   | 252   | 1.96       |
| im0     | 3    | im1     | 128   | 252   | 1.96       |
| im0     | 3    | im1     | 1024  | 730   | 0.71       |
| im0     | 3    | im1     | 1024  | 730   | 0.71       |
| im0     | 3    | im1     | 2048  | 1270  | 0.62       |
| im0     | 3    | im1     | 2048  | 1270  | 0.62       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 182   | 22.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1270  | 0.62       |

### 6.184.2.3 nmppsMulC\_32s()

```
void nmppsMulC_32s (
    const nm32s * pSrcVec,
    int32b nVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 2114  | 1.03       |
| im0     | 3    | im0     | 2048  | 2114  | 1.03       |
| im0     | 3    | im1     | 2048  | 1223  | 0.59       |
| im0     | 3    | im1     | 2048  | 1223  | 0.59       |
| im1     | 3    | im0     | 2048  | 1221  | 0.59       |
| im1     | 3    | im0     | 2048  | 1221  | 0.59       |
| im1     | 3    | im1     | 2048  | 2245  | 1.09       |
| im1     | 3    | im1     | 2048  | 2245  | 1.09       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 1221  | 0.59       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 2245  | 1.09       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 137   | 17.12      |
| im0     | 3    | im1     | 8     | 137   | 17.12      |
| im0     | 3    | im1     | 128   | 205   | 1.60       |
| im0     | 3    | im1     | 128   | 205   | 1.60       |

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 1024  | 679   | 0.66       |
| im0     | 3    | im1     | 1024  | 679   | 0.66       |
| im0     | 3    | im1     | 2048  | 1223  | 0.59       |
| im0     | 3    | im1     | 2048  | 1223  | 0.59       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 137   | 17.12      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 1223  | 0.59       |

#### 6.184.2.4 nmppsMulC\_32s64s()

```
void nmppsMulC_32s64s (
    const nm32s * pSrcVec,
    int64b nVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 4174  | 2.03       |
| im0     | 3    | im0     | 2048  | 4174  | 2.03       |
| im0     | 3    | im1     | 2048  | 2353  | 1.14       |
| im0     | 3    | im1     | 2048  | 2353  | 1.14       |
| im1     | 3    | im0     | 2048  | 2349  | 1.14       |
| im1     | 3    | im0     | 2048  | 2349  | 1.14       |
| im1     | 3    | im1     | 2048  | 4411  | 2.15       |
| im1     | 3    | im1     | 2048  | 4411  | 2.15       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 2349  | 1.14       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 4411  | 2.15       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 182   | 22.75      |
| im0     | 3    | im1     | 8     | 182   | 22.75      |
| im0     | 3    | im1     | 128   | 317   | 2.47       |
| im0     | 3    | im1     | 128   | 317   | 2.47       |
| im0     | 3    | im1     | 1024  | 1265  | 1.23       |
| im0     | 3    | im1     | 1024  | 1265  | 1.23       |
| im0     | 3    | im1     | 2048  | 2353  | 1.14       |
| im0     | 3    | im1     | 2048  | 2353  | 1.14       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 182   | 22.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2353  | 1.14       |

#### 6.184.2.5 nmppsMulC\_64s()

```
void nmppsMulC_64s (
    const nm64s * pSrcVec,
    int64b nVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 4164  | 2.03       |
| im0     | 3    | im0     | 2048  | 4164  | 2.03       |
| im0     | 3    | im1     | 2048  | 2313  | 1.12       |
| im0     | 3    | im1     | 2048  | 2313  | 1.12       |
| im1     | 3    | im0     | 2048  | 2311  | 1.12       |
| im1     | 3    | im0     | 2048  | 2311  | 1.12       |
| im1     | 3    | im1     | 2048  | 4359  | 2.12       |
| im1     | 3    | im1     | 2048  | 4359  | 2.12       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 2311  | 1.12       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 4359  | 2.12       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 146   | 18.25      |
| im0     | 3    | im1     | 8     | 146   | 18.25      |
| im0     | 3    | im1     | 128   | 274   | 2.14       |
| im0     | 3    | im1     | 128   | 274   | 2.14       |
| im0     | 3    | im1     | 1024  | 1225  | 1.19       |
| im0     | 3    | im1     | 1024  | 1225  | 1.19       |
| im0     | 3    | im1     | 2048  | 2313  | 1.12       |
| im0     | 3    | im1     | 2048  | 2313  | 1.12       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 146   | 18.25      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 2313  | 1.12       |

#### 6.184.2.6 nmppsMulC\_8s()

```
void nmppsMulC_8s (
    const nm8s * pSrcVec,
    int8b nVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 597   | 0.29       |
| im0     | 3    | im0     | 2048  | 597   | 0.29       |
| im0     | 3    | im1     | 2048  | 428   | 0.20       |
| im0     | 3    | im1     | 2048  | 428   | 0.20       |
| im1     | 3    | im0     | 2048  | 424   | 0.20       |
| im1     | 3    | im0     | 2048  | 424   | 0.20       |
| im1     | 3    | im1     | 2048  | 680   | 0.33       |
| im1     | 3    | im1     | 2048  | 680   | 0.33       |



The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 424   | 0.20       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 680   | 0.33       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 153   | 19.12      |
| im0     | 3    | im1     | 8     | 153   | 19.12      |
| im0     | 3    | im1     | 128   | 171   | 1.33       |
| im0     | 3    | im1     | 128   | 171   | 1.33       |
| im0     | 3    | im1     | 1024  | 291   | 0.28       |
| im0     | 3    | im1     | 1024  | 291   | 0.28       |
| im0     | 3    | im1     | 2048  | 428   | 0.20       |
| im0     | 3    | im1     | 2048  | 428   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 153   | 19.12      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 428   | 0.20       |

#### 6.184.2.7 nmppsMulC\_8s16s()

```
void nmppsMulC_8s16s (
    const nm8s * pSrcVec,
    int16b nVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im0     | 2048  | 1119  | 0.54       |
| im0     | 3    | im0     | 2048  | 1119  | 0.54       |

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 743   | 0.36       |
| im0     | 3    | im1     | 2048  | 743   | 0.36       |
| im1     | 3    | im0     | 2048  | 735   | 0.35       |
| im1     | 3    | im0     | 2048  | 735   | 0.35       |
| im1     | 3    | im1     | 2048  | 1261  | 0.61       |
| im1     | 3    | im1     | 2048  | 1261  | 0.61       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im0     | 2048  | 735   | 0.35       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 3    | im1     | 2048  | 1261  | 0.61       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 193   | 24.12      |
| im0     | 3    | im1     | 8     | 193   | 24.12      |
| im0     | 3    | im1     | 128   | 228   | 1.78       |
| im0     | 3    | im1     | 128   | 228   | 1.78       |
| im0     | 3    | im1     | 1024  | 469   | 0.45       |
| im0     | 3    | im1     | 1024  | 469   | 0.45       |
| im0     | 3    | im1     | 2048  | 743   | 0.36       |
| im0     | 3    | im1     | 2048  | 743   | 0.36       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 8     | 193   | 24.12      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 3    | im1     | 2048  | 743   | 0.36       |

# 6.185 nmppsMulC

Умножение вектора на константу.

## Функции

- void nmppsMulC\_32f (const nm32f \*pSrcVec, nm32f \*pDstVec, float C, int nSize)

### 6.185.1 Подробное описание

Умножение вектора на константу.

#### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа-множитель.         |
| nSize   | Размер векторов в элементах. |

#### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

#### Возвращает

void

### 6.185.2 Функции

#### 6.185.2.1 nmppsMulC\_32f()

```
void nmppsMulC_32f (  
    const nm32f * pSrcVec,  
    nm32f * pDstVec,  
    float C,  
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im1     | 555 | 2048  | 2004  | 0.97       |
| im1     | im1     | 555 | 2048  | 2004  | 0.97       |
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |
| im2     | im1     | 555 | 2048  | 1055  | 0.51       |
| im2     | im1     | 555 | 2048  | 1055  | 0.51       |
| im2     | im2     | 555 | 2048  | 2004  | 0.97       |
| im2     | im2     | 555 | 2048  | 2004  | 0.97       |

The best configuration:

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |

The worst configuration:

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im1     | 555 | 2048  | 2004  | 0.97       |

Perfomance table 1

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im2     | 555 | 8     | 106   | 13.25      |
| im1     | im2     | 555 | 8     | 106   | 13.25      |
| im1     | im2     | 555 | 128   | 108   | 0.84       |
| im1     | im2     | 555 | 128   | 108   | 0.84       |
| im1     | im2     | 555 | 1024  | 543   | 0.53       |
| im1     | im2     | 555 | 1024  | 543   | 0.53       |
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |

The best configuration:

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im2     | 555 | 8     | 106   | 13.25      |

The worst configuration:

| pSrcVec | pDstVec | C   | nSize | ticks | ticks/elem |
|---------|---------|-----|-------|-------|------------|
| im1     | im2     | 555 | 2048  | 1055  | 0.51       |

## 6.186 nmppsMul\_Mul\_Add

Умножение двух пар векторов с последующим суммированием результатов ( $pDstVec = pSrcVec1 * pSrcVec2 + pSrcVec3 * pSrcVec4$ )

### Функции

- void [nmppsMul\\_Mul\\_Add\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, const nm32f \*pSrcVec3, const nm32f \*pSrcVec4, nm32f \*pDstVec, int nSize)
- void [nmppsMul\\_Mul\\_Add\\_32fcr](#) (const [nm32fcr](#) \*pSrcVec1, const [nm32fcr](#) \*pSrcVec2, const [nm32fcr](#) \*pSrcVec3, const [nm32fcr](#) \*pSrcVec4, [nm32fcr](#) \*pDstVec, int nSize)
- void [nmppsMul\\_Mul\\_Add\\_64f](#) (const nm64f \*pSrcVec1, const nm64f \*pSrcVec2, const nm64f \*pSrcVec3, const nm64f \*pSrcVec4, nm64f \*pDstVec, int nSize)

### 6.186.1 Подробное описание

Умножение двух пар векторов с последующим суммированием результатов ( $pDstVec = pSrcVec1 * pSrcVec2 + pSrcVec3 * pSrcVec4$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| pSrcVec3 | Входной вектор3             |
| pSrcVec4 | Входной вектор4             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.186.2 Функции

#### 6.186.2.1 nmppsMul\_Mul\_Add\_32f()

```
void nmppsMul_Mul_Add_32f (
    const nm32f * pSrcVec1,
```

```

const nm32f * pSrcVec2,
const nm32f * pSrcVec3,
const nm32f * pSrcVec4,
nm32f * pDstVec,
int nSize )

```

Performance table 0

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

Performance table 1

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |
| im1      | im2      | im3      | im4      | im5     | 128   | 148   | 1.15       |
| im1      | im2      | im3      | im4      | im5     | 128   | 148   | 1.15       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 596   | 0.58       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 596   | 0.58       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

## 6.186.2.2 nmppsMul\_Mul\_Add\_32fcr()

```
void nmppsMul_Mul_Add_32fcr (
    const nm32fcr * pSrcVec1,
    const nm32fcr * pSrcVec2,
    const nm32fcr * pSrcVec3,
    const nm32fcr * pSrcVec4,
    nm32fcr * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |
| im1      | im2      | im3      | im4      | im5     | 128   | 211   | 1.64       |
| im1      | im2      | im3      | im4      | im5     | 128   | 211   | 1.64       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 1107  | 1.08       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 1107  | 1.08       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

## 6.186.2.3 nmppsMul\_Mul\_Add\_64f()

```
void nmppsMul_Mul_Add_64f (
    const nm64f * pSrcVec1,
    const nm64f * pSrcVec2,
    const nm64f * pSrcVec3,
    const nm64f * pSrcVec4,
    nm64f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |
| im1      | im2      | im3      | im4      | im5     | 128   | 211   | 1.64       |
| im1      | im2      | im3      | im4      | im5     | 128   | 211   | 1.64       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 1107  | 1.08       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 1107  | 1.08       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 105   | 13.12      |

The worst configuration:



| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 2131  | 1.04       |

## 6.187 nmppsMul\_Add

Умножение с накоплением

Функции

- void [nmppsMul\\_Add\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, const nm32f \*pSrcVecAdd, nm32f \*pDstVec, int nSize)
- void [nmppsMul\\_Add\\_64f](#) (const nm64f \*pSrcVec1, const nm64f \*pSrcVec2, const nm64f \*pSrcVecAdd, nm64f \*pDstVec, int nSize)
- void [nmppsMul\\_Add\\_32fcr](#) (const [nm32fcr](#) \*pSrcVec1, const [nm32fcr](#) \*pSrcVec2, const [nm32fcr](#) \*pSrcVecAdd, [nm32fcr](#) \*pDstVec, int nSize)

### 6.187.1 Подробное описание

Умножение с накоплением

Аргументы

|            |                             |
|------------|-----------------------------|
| pSrcVec1   | Входной вектор1             |
| pSrcVec2   | Входной вектор2             |
| pSrcVecAdd | Входной вектор              |
| nSize      | Размер векторов в элементах |

Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

Возвращает

void

### 6.187.2 Функции

#### 6.187.2.1 nmppsMul\_Add\_32f()

```
void nmppsMul_Add_32f (
    const nm32f * pSrcVec1,
    const nm32f * pSrcVec2,
    const nm32f * pSrcVecAdd,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 2087  | 1.01       |
| im1      | im1      | im3        | im4     | 2048  | 2087  | 1.01       |
| im1      | im2      | im3        | im4     | 2048  | 1579  | 0.77       |
| im1      | im2      | im3        | im4     | 2048  | 1579  | 0.77       |
| im2      | im1      | im3        | im4     | 2048  | 1579  | 0.77       |
| im2      | im1      | im3        | im4     | 2048  | 1579  | 0.77       |
| im2      | im2      | im3        | im4     | 2048  | 2087  | 1.01       |
| im2      | im2      | im3        | im4     | 2048  | 2087  | 1.01       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 1579  | 0.77       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 2087  | 1.01       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 103   | 12.87      |
| im1      | im2      | im3        | im4     | 8     | 103   | 12.87      |
| im1      | im2      | im3        | im4     | 128   | 136   | 1.06       |
| im1      | im2      | im3        | im4     | 128   | 136   | 1.06       |
| im1      | im2      | im3        | im4     | 1024  | 810   | 0.79       |
| im1      | im2      | im3        | im4     | 1024  | 810   | 0.79       |
| im1      | im2      | im3        | im4     | 2048  | 1578  | 0.77       |
| im1      | im2      | im3        | im4     | 2048  | 1578  | 0.77       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 103   | 12.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 1578  | 0.77       |

### 6.187.2.2 nmppsMul\_Add\_32fcr()

```
void nmppsMul_Add_32fcr (
```

```

const nm32fcr * pSrcVec1,
const nm32fcr * pSrcVec2,
const nm32fcr * pSrcVecAdd,
nm32fcr * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im1      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im1      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im2      | im3        | im4     | 2048  | 4112  | 2.00       |
| im2      | im2      | im3        | im4     | 2048  | 4112  | 2.00       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |
| im1      | im2      | im3        | im4     | 128   | 202   | 1.57       |
| im1      | im2      | im3        | im4     | 128   | 202   | 1.57       |
| im1      | im2      | im3        | im4     | 1024  | 1098  | 1.07       |
| im1      | im2      | im3        | im4     | 1024  | 1098  | 1.07       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

## 6.187.2.3 nmppsMul\_Add\_64f()

```
void nmppsMul_Add_64f (
    const nm64f * pSrcVec1,
    const nm64f * pSrcVec2,
    const nm64f * pSrcVecAdd,
    nm64f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im1      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im1      | im3        | im4     | 2048  | 2122  | 1.03       |
| im2      | im2      | im3        | im4     | 2048  | 4112  | 2.00       |
| im2      | im2      | im3        | im4     | 2048  | 4112  | 2.00       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | im3        | im4     | 2048  | 4112  | 2.00       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |
| im1      | im2      | im3        | im4     | 128   | 202   | 1.57       |
| im1      | im2      | im3        | im4     | 128   | 202   | 1.57       |
| im1      | im2      | im3        | im4     | 1024  | 1098  | 1.07       |
| im1      | im2      | im3        | im4     | 1024  | 1098  | 1.07       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVecAdd | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | im3        | im4     | 2048  | 2122  | 1.03       |

## 6.188 nmppsMul\_ConjMul\_Add

Умножение пары векторов (вектор pSrcVec4 комплексно-сопряженный) с последующим суммированием результата ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] + pSrcVec3[i] * Conj(pSrcVec4[i])$ )

### Функции

- void nmppsMul\_ConjMul\_Add\_32fcr (const nm32fcr \*pSrcVec1, const nm32fcr \*pSrcVec2, const nm32fcr \*pSrcVec3, const nm32fcr \*pSrcVec4, nm32fcr \*pDstVec, int nSize)

### 6.188.1 Подробное описание

Умножение пары векторов (вектор pSrcVec4 комплексно-сопряженный) с последующим суммированием результата ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] + pSrcVec3[i] * Conj(pSrcVec4[i])$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| pSrcVec3 | Входной вектор3             |
| pSrcVec4 | Входной вектор4             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.188.2 Функции

#### 6.188.2.1 nmppsMul\_ConjMul\_Add\_32fcr()

```
void nmppsMul_ConjMul_Add_32fcr (
    const nm32fcr * pSrcVec1,
    const nm32fcr * pSrcVec2,
    const nm32fcr * pSrcVec3,
    const nm32fcr * pSrcVec4,
    nm32fcr * pDstVec,
    int nSize )
```

#### Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im1      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im1      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im4      | im3      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |
| im2      | im2      | im4      | im4      | im5     | 2048  | 4459  | 2.17       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3      | im3      | im5     | 2048  | 4459  | 2.17       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 108   | 13.50      |
| im1      | im2      | im3      | im4      | im5     | 8     | 108   | 13.50      |
| im1      | im2      | im3      | im4      | im5     | 128   | 319   | 2.49       |



| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 128   | 319   | 2.49       |
| im1      | im2      | im3      | im4      | im5     | 256   | 595   | 2.32       |
| im1      | im2      | im3      | im4      | im5     | 256   | 595   | 2.32       |
| im1      | im2      | im3      | im4      | im5     | 512   | 1147  | 2.24       |
| im1      | im2      | im3      | im4      | im5     | 512   | 1147  | 2.24       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 2251  | 2.19       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 2251  | 2.19       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 108   | 13.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 4459  | 2.17       |

## 6.189 nmppsMulC\_AddC

Умножает вектор на константу с прибавлением другой константы ( $pDstVec[i] = nMulC * pSrcVec[i] + nAddC$ )

### Функции

- void [nmppsMulC\\_AddC\\_32f](#) (const nm32f \*pSrcVec, float nMulC, float nAddC, nm32f \*pDstVec, int nSize)

### 6.189.1 Подробное описание

Умножает вектор на константу с прибавлением другой константы ( $pDstVec[i] = nMulC * pSrcVec[i] + nAddC$ )

#### Аргументы

|         |                             |
|---------|-----------------------------|
| pSrcVec | Входной вектор              |
| nMulC   | Константа                   |
| nAddC   | Константа                   |
| nSize   | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.189.2 Функции

#### 6.189.2.1 nmppsMulC\_AddC\_32f()

```
void nmppsMulC_AddC_32f (
    const nm32f * pSrcVec,
    float nMulC,
    float nAddC,
    nm32f * pDstVec,
    int nSize )
```

#### Perfomance table 0

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im1     | 5     | 5     | im2     | 2048  | 1085  | 0.52       |
| im1     | 5     | 5     | im2     | 2048  | 1085  | 0.52       |
| im1     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |
| im1     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |
| im2     | 5     | 5     | im2     | 2048  | 2032  | 0.99       |
| im2     | 5     | 5     | im2     | 2048  | 2032  | 0.99       |
| im2     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |
| im2     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |

The best configuration:

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im1     | 5     | 5     | im2     | 2048  | 1085  | 0.52       |

The worst configuration:

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im2     | 5     | 5     | im2     | 2048  | 2032  | 0.99       |

Perfomance table 1

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im1     | 5     | 5     | im3     | 8     | 153   | 19.12      |
| im1     | 5     | 5     | im3     | 8     | 153   | 19.12      |
| im1     | 5     | 5     | im3     | 128   | 145   | 1.13       |
| im1     | 5     | 5     | im3     | 128   | 145   | 1.13       |
| im1     | 5     | 5     | im3     | 256   | 189   | 0.73       |
| im1     | 5     | 5     | im3     | 256   | 189   | 0.73       |
| im1     | 5     | 5     | im3     | 512   | 317   | 0.61       |
| im1     | 5     | 5     | im3     | 512   | 317   | 0.61       |
| im1     | 5     | 5     | im3     | 1024  | 573   | 0.55       |
| im1     | 5     | 5     | im3     | 1024  | 573   | 0.55       |
| im1     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |
| im1     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |

The best configuration:

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im1     | 5     | 5     | im3     | 128   | 145   | 1.13       |

The worst configuration:

| pSrcVec | nMulC | nAddC | pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|---------|-------|-------|------------|
| im1     | 5     | 5     | im3     | 2048  | 1085  | 0.52       |

## 6.190 nmppsMul

Умножение векторов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec1[i]$ )

### Функции

- void `nmppsMul_32fcr` (const `nm32fcr` \*pSrcVec1, const `nm32fcr` \*pSrcVec2, `nm32fcr` \*pDstVec, int nSize)

### 6.190.1 Подробное описание

Умножение векторов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec1[i]$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.190.2 Функции

#### 6.190.2.1 nmppsMul\_32fcr()

```
void nmppsMul_32fcr (
    const nm32fcr * pSrcVec1,
    const nm32fcr * pSrcVec2,
    nm32fcr * pDstVec,
    int nSize )
```

#### Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 2081  | 1.01       |
| im1      | im2      | im3     | 2048  | 2081  | 1.01       |
| im2      | im1      | im3     | 2048  | 2081  | 1.01       |
| im2      | im1      | im3     | 2048  | 2081  | 1.01       |
| im2      | im2      | im3     | 2048  | 4064  | 1.98       |
| im2      | im2      | im3     | 2048  | 4064  | 1.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 2081  | 1.01       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 91    | 11.37      |
| im1      | im1      | im3     | 8     | 91    | 11.37      |
| im1      | im1      | im3     | 128   | 224   | 1.75       |
| im1      | im1      | im3     | 128   | 224   | 1.75       |
| im1      | im1      | im3     | 256   | 480   | 1.87       |
| im1      | im1      | im3     | 256   | 480   | 1.87       |
| im1      | im1      | im3     | 512   | 992   | 1.93       |
| im1      | im1      | im3     | 512   | 992   | 1.93       |
| im1      | im1      | im3     | 1024  | 2016  | 1.96       |
| im1      | im1      | im3     | 1024  | 2016  | 1.96       |
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |
| im1      | im2      | im3     | 8     | 91    | 11.37      |
| im1      | im2      | im3     | 8     | 91    | 11.37      |
| im1      | im2      | im3     | 128   | 161   | 1.25       |
| im1      | im2      | im3     | 128   | 161   | 1.25       |
| im1      | im2      | im3     | 256   | 289   | 1.12       |
| im1      | im2      | im3     | 256   | 289   | 1.12       |
| im1      | im2      | im3     | 512   | 545   | 1.06       |
| im1      | im2      | im3     | 512   | 545   | 1.06       |
| im1      | im2      | im3     | 1024  | 1057  | 1.03       |
| im1      | im2      | im3     | 1024  | 1057  | 1.03       |
| im1      | im2      | im3     | 2048  | 2081  | 1.01       |
| im1      | im2      | im3     | 2048  | 2081  | 1.01       |
| im2      | im1      | im3     | 8     | 91    | 11.37      |
| im2      | im1      | im3     | 8     | 91    | 11.37      |
| im2      | im1      | im3     | 128   | 161   | 1.25       |
| im2      | im1      | im3     | 128   | 161   | 1.25       |
| im2      | im1      | im3     | 256   | 289   | 1.12       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im2      | im1      | im3     | 256   | 289   | 1.12       |
| im2      | im1      | im3     | 512   | 545   | 1.06       |
| im2      | im1      | im3     | 512   | 545   | 1.06       |
| im2      | im1      | im3     | 1024  | 1057  | 1.03       |
| im2      | im1      | im3     | 1024  | 1057  | 1.03       |
| im2      | im1      | im3     | 2048  | 2081  | 1.01       |
| im2      | im1      | im3     | 2048  | 2081  | 1.01       |
| im2      | im2      | im3     | 8     | 91    | 11.37      |
| im2      | im2      | im3     | 8     | 91    | 11.37      |
| im2      | im2      | im3     | 128   | 224   | 1.75       |
| im2      | im2      | im3     | 128   | 224   | 1.75       |
| im2      | im2      | im3     | 256   | 480   | 1.87       |
| im2      | im2      | im3     | 256   | 480   | 1.87       |
| im2      | im2      | im3     | 512   | 992   | 1.93       |
| im2      | im2      | im3     | 512   | 992   | 1.93       |
| im2      | im2      | im3     | 1024  | 2016  | 1.96       |
| im2      | im2      | im3     | 1024  | 2016  | 1.96       |
| im2      | im2      | im3     | 2048  | 4064  | 1.98       |
| im2      | im2      | im3     | 2048  | 4064  | 1.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 91    | 11.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 4064  | 1.98       |

## 6.191 nmppsConjMul

Умножение вектора на комплексно-сопряженный вектор( $pDstVec[i] = pSrcVec1[i] * Conj(pSrcVec1[i])$ )

### Функции

- void `nmppsConjMul_32fcr` (const `nm32fcr` \*pSrcVec1, const `nm32fcr` \*pSrcVec2, `nm32fcr` \*pDstVec, int nSize)

### 6.191.1 Подробное описание

Умножение вектора на комплексно-сопряженный вектор( $pDstVec[i] = pSrcVec1[i] * Conj(pSrcVec1[i])$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.191.2 Функции

#### 6.191.2.1 nmppsConjMul\_32fcr()

```
void nmppsConjMul_32fcr (
    const nm32fcr * pSrcVec1,
    const nm32fcr * pSrcVec2,
    nm32fcr * pDstVec,
    int nSize )
```

#### Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 8641  | 4.21       |
| im1      | im2      | im3     | 2048  | 8641  | 4.21       |
| im2      | im1      | im3     | 2048  | 8641  | 4.21       |
| im2      | im1      | im3     | 2048  | 8641  | 4.21       |
| im2      | im2      | im3     | 2048  | 8796  | 4.29       |
| im2      | im2      | im3     | 2048  | 8796  | 4.29       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 8641  | 4.21       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 95    | 11.87      |
| im1      | im1      | im3     | 8     | 95    | 11.87      |
| im1      | im1      | im3     | 128   | 516   | 4.03       |
| im1      | im1      | im3     | 128   | 516   | 4.03       |
| im1      | im1      | im3     | 256   | 1068  | 4.17       |
| im1      | im1      | im3     | 256   | 1068  | 4.17       |
| im1      | im1      | im3     | 512   | 2172  | 4.24       |
| im1      | im1      | im3     | 512   | 2172  | 4.24       |
| im1      | im1      | im3     | 1024  | 4380  | 4.27       |
| im1      | im1      | im3     | 1024  | 4380  | 4.27       |
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |
| im1      | im2      | im3     | 8     | 95    | 11.87      |
| im1      | im2      | im3     | 8     | 95    | 11.87      |
| im1      | im2      | im3     | 128   | 481   | 3.75       |
| im1      | im2      | im3     | 128   | 481   | 3.75       |
| im1      | im2      | im3     | 256   | 1025  | 4.00       |
| im1      | im2      | im3     | 256   | 1025  | 4.00       |
| im1      | im2      | im3     | 512   | 2113  | 4.12       |
| im1      | im2      | im3     | 512   | 2113  | 4.12       |
| im1      | im2      | im3     | 1024  | 4289  | 4.18       |
| im1      | im2      | im3     | 1024  | 4289  | 4.18       |
| im1      | im2      | im3     | 2048  | 8641  | 4.21       |
| im1      | im2      | im3     | 2048  | 8641  | 4.21       |
| im2      | im1      | im3     | 8     | 95    | 11.87      |
| im2      | im1      | im3     | 8     | 95    | 11.87      |
| im2      | im1      | im3     | 128   | 481   | 3.75       |
| im2      | im1      | im3     | 128   | 481   | 3.75       |
| im2      | im1      | im3     | 256   | 1025  | 4.00       |



| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im2      | im1      | im3     | 256   | 1025  | 4.00       |
| im2      | im1      | im3     | 512   | 2113  | 4.12       |
| im2      | im1      | im3     | 512   | 2113  | 4.12       |
| im2      | im1      | im3     | 1024  | 4289  | 4.18       |
| im2      | im1      | im3     | 1024  | 4289  | 4.18       |
| im2      | im1      | im3     | 2048  | 8641  | 4.21       |
| im2      | im1      | im3     | 2048  | 8641  | 4.21       |
| im2      | im2      | im3     | 8     | 95    | 11.87      |
| im2      | im2      | im3     | 8     | 95    | 11.87      |
| im2      | im2      | im3     | 128   | 516   | 4.03       |
| im2      | im2      | im3     | 128   | 516   | 4.03       |
| im2      | im2      | im3     | 256   | 1068  | 4.17       |
| im2      | im2      | im3     | 256   | 1068  | 4.17       |
| im2      | im2      | im3     | 512   | 2172  | 4.24       |
| im2      | im2      | im3     | 512   | 2172  | 4.24       |
| im2      | im2      | im3     | 1024  | 4380  | 4.27       |
| im2      | im2      | im3     | 1024  | 4380  | 4.27       |
| im2      | im2      | im3     | 2048  | 8796  | 4.29       |
| im2      | im2      | im3     | 2048  | 8796  | 4.29       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 95    | 11.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 8796  | 4.29       |

## 6.192 nmppsMul\_Mul\_Sub

Умножение пары векторов с последующим вычитанием результатов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] - pSrcVec3[i] * pSrcVec4[i]$ )

### Функции

- void [nmppsMul\\_Mul\\_Sub\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, const nm32f \*pSrcVec3, const nm32f \*pSrcVec4, nm32f \*pDstVec, int nSize)

### 6.192.1 Подробное описание

Умножение пары векторов с последующим вычитанием результатов ( $pDstVec[i] = pSrcVec1[i] * pSrcVec2[i] - pSrcVec3[i] * pSrcVec4[i]$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| pSrcVec3 | Входной вектор1             |
| pSrcVec4 | Входной вектор4             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.192.2 Функции

#### 6.192.2.1 nmppsMul\_Mul\_Sub\_32f()

```
void nmppsMul_Mul_Sub_32f (
    const nm32f * pSrcVec1,
    const nm32f * pSrcVec2,
    const nm32f * pSrcVec3,
    const nm32f * pSrcVec4,
    nm32f * pDstVec,
    int nSize )
```

#### Perfomance table 0

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im2      | im2      | im2     | 2048  | 4019  | 1.96       |
| im1      | im2      | im2      | im2      | im2     | 2048  | 4019  | 1.96       |
| im1      | im2      | im2      | im2      | im5     | 2048  | 3059  | 1.49       |
| im1      | im2      | im2      | im2      | im5     | 2048  | 3059  | 1.49       |
| im1      | im2      | im2      | im4      | im2     | 2048  | 3034  | 1.48       |
| im1      | im2      | im2      | im4      | im2     | 2048  | 3034  | 1.48       |
| im1      | im2      | im2      | im4      | im5     | 2048  | 2074  | 1.01       |
| im1      | im2      | im2      | im4      | im5     | 2048  | 2074  | 1.01       |
| im1      | im2      | im3      | im2      | im2     | 2048  | 3034  | 1.48       |
| im1      | im2      | im3      | im2      | im2     | 2048  | 3034  | 1.48       |
| im1      | im2      | im3      | im2      | im5     | 2048  | 2074  | 1.01       |
| im1      | im2      | im3      | im2      | im5     | 2048  | 2074  | 1.01       |
| im1      | im2      | im3      | im4      | im2     | 2048  | 2064  | 1.00       |
| im1      | im2      | im3      | im4      | im2     | 2048  | 2064  | 1.00       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |
| im2      | im2      | im2      | im2      | im2     | 2048  | 5039  | 2.46       |
| im2      | im2      | im2      | im2      | im2     | 2048  | 5039  | 2.46       |
| im2      | im2      | im2      | im2      | im5     | 2048  | 4079  | 1.99       |
| im2      | im2      | im2      | im2      | im5     | 2048  | 4079  | 1.99       |
| im2      | im2      | im2      | im4      | im2     | 2048  | 4047  | 1.97       |
| im2      | im2      | im2      | im4      | im2     | 2048  | 4047  | 1.97       |
| im2      | im2      | im2      | im4      | im5     | 2048  | 3087  | 1.50       |
| im2      | im2      | im2      | im4      | im5     | 2048  | 3087  | 1.50       |
| im2      | im2      | im3      | im2      | im2     | 2048  | 4047  | 1.97       |
| im2      | im2      | im3      | im2      | im2     | 2048  | 4047  | 1.97       |
| im2      | im2      | im3      | im2      | im5     | 2048  | 3087  | 1.50       |
| im2      | im2      | im3      | im2      | im5     | 2048  | 3087  | 1.50       |
| im2      | im2      | im3      | im4      | im2     | 2048  | 3055  | 1.49       |
| im2      | im2      | im3      | im4      | im2     | 2048  | 3055  | 1.49       |
| im2      | im2      | im3      | im4      | im5     | 2048  | 2095  | 1.02       |
| im2      | im2      | im3      | im4      | im5     | 2048  | 2095  | 1.02       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im2      | im2      | im2      | im2      | im2     | 2048  | 5039  | 2.46       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |
| im1      | im2      | im3      | im4      | im5     | 128   | 148   | 1.15       |

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 128   | 148   | 1.15       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 596   | 0.58       |
| im1      | im2      | im3      | im4      | im5     | 1024  | 596   | 0.58       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 8     | 110   | 13.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pSrcVec3 | pSrcVec4 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3      | im4      | im5     | 2048  | 1108  | 0.54       |

## 6.193 nmppsMulC\_Add

Умножение вектора на константу с накоплением ( $pDstVec[i] = pSrcVec1[i] * C + pSrcVec2[i]$ )

### Функции

- void **nmppsMulC\_Add\_32fcr** (const **nm32fcr** \*pSrcVec1, const **nm32fcr** \*pSrcVec2, **nm32fcr** \*pDstVec, float C, int nSize)

### 6.193.1 Подробное описание

Умножение вектора на константу с накоплением ( $pDstVec[i] = pSrcVec1[i] * C + pSrcVec2[i]$ )

#### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| nSize    | Размер векторов в элементах |

#### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

#### Возвращает

void

### 6.193.2 Функции

#### 6.193.2.1 nmppsMulC\_Add\_32fcr()

```
void nmppsMulC_Add_32fcr (  
    const nm32fcr * pSrcVec1,  
    const nm32fcr * pSrcVec2,  
    nm32fcr * pDstVec,  
    float C,  
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im1      | im2      | im2     | 5 | 2048  | 4060  | 1.98       |
| im1      | im2      | im2     | 5 | 2048  | 4060  | 1.98       |
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |
| im2      | im2      | im2     | 5 | 2048  | 6043  | 2.95       |
| im2      | im2      | im2     | 5 | 2048  | 6043  | 2.95       |
| im2      | im2      | im3     | 5 | 2048  | 4074  | 1.98       |
| im2      | im2      | im3     | 5 | 2048  | 4074  | 1.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im2      | im2      | im2     | 5 | 2048  | 6043  | 2.95       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im1      | im2      | im3     | 5 | 8     | 104   | 13.00      |
| im1      | im2      | im3     | 5 | 8     | 104   | 13.00      |
| im1      | im2      | im3     | 5 | 128   | 171   | 1.33       |
| im1      | im2      | im3     | 5 | 128   | 171   | 1.33       |
| im1      | im2      | im3     | 5 | 1024  | 1067  | 1.04       |
| im1      | im2      | im3     | 5 | 1024  | 1067  | 1.04       |
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im1      | im2      | im3     | 5 | 8     | 104   | 13.00      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | C | nSize | ticks | ticks/elem |
|----------|----------|---------|---|-------|-------|------------|
| im1      | im2      | im3     | 5 | 2048  | 2091  | 1.02       |

## 6.194 nmppsMul\_AddC

Поэлементное умножение векторов с прибавлением константы.

### Функции

- void nmppsMul\_AddC\_64s (const nm64s \*pSrcVec1, const nm64s \*pSrcVec2, const nm64s \*pnVal, nm64s \*pDstVec, int nSize)

#### 6.194.1 Подробное описание

Поэлементное умножение векторов с прибавлением константы.

$$pDstVec[i] = pSrcVec1[i] \cdot pSrcVec2[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|          |                                  |
|----------|----------------------------------|
| pSrcVec1 | Входной вектор.                  |
| pSrcVec2 | Входной вектор.                  |
| pnVal    | указаель на константу-инкремент. |
| nSize    | Размер векторов в элементах.     |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

## 6.195 nmppsMul\_AddC

Поэлементное умножение векторов с прибавлением константы.

Функции

- void [nmppsMul\\_AddC\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, float nValueAddC, nm32f \*pDstVec, int nSize)

### 6.195.1 Подробное описание

Поэлементное умножение векторов с прибавлением константы.

$$pDstVec[i] = pSrcVec1[i] \cdot pSrcVec2[i] + nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|            |                                  |
|------------|----------------------------------|
| pSrcVec1   | Входной вектор.                  |
| pSrcVec2   | Входной вектор.                  |
| nValueAddC | указаель на константу-инкремент. |
| nSize      | Размер векторов в элементах.     |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.195.2 Функции

#### 6.195.2.1 nmppsMul\_AddC\_32f()

```
void nmppsMul_AddC_32f (
    const nm32f * pSrcVec1,
```



```

const nm32f * pSrcVec2,
float nValueAddC,
nm32f * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | 5          | im2     | 2048  | 2024  | 0.98       |
| im1      | im1      | 5          | im2     | 2048  | 2024  | 0.98       |
| im1      | im1      | 5          | im3     | 2048  | 2024  | 0.98       |
| im1      | im1      | 5          | im3     | 2048  | 2024  | 0.98       |
| im1      | im2      | 5          | im2     | 2048  | 2037  | 0.99       |
| im1      | im2      | 5          | im2     | 2048  | 2037  | 0.99       |
| im1      | im2      | 5          | im3     | 2048  | 1092  | 0.53       |
| im1      | im2      | 5          | im3     | 2048  | 1092  | 0.53       |
| im2      | im1      | 5          | im2     | 2048  | 2015  | 0.98       |
| im2      | im1      | 5          | im2     | 2048  | 2015  | 0.98       |
| im2      | im1      | 5          | im3     | 2048  | 1092  | 0.53       |
| im2      | im1      | 5          | im3     | 2048  | 1092  | 0.53       |
| im2      | im2      | 5          | im2     | 2048  | 2969  | 1.44       |
| im2      | im2      | 5          | im2     | 2048  | 2969  | 1.44       |
| im2      | im2      | 5          | im3     | 2048  | 2024  | 0.98       |
| im2      | im2      | 5          | im3     | 2048  | 2024  | 0.98       |

The best configuration:

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | 5          | im3     | 2048  | 1092  | 0.53       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im2      | im2      | 5          | im2     | 2048  | 2969  | 1.44       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | 5          | im3     | 8     | 121   | 15.12      |
| im1      | im1      | 5          | im3     | 8     | 121   | 15.12      |
| im1      | im1      | 5          | im3     | 128   | 132   | 1.03       |
| im1      | im1      | 5          | im3     | 128   | 132   | 1.03       |
| im1      | im1      | 5          | im3     | 1024  | 1000  | 0.97       |
| im1      | im1      | 5          | im3     | 1024  | 1000  | 0.97       |
| im1      | im1      | 5          | im3     | 2048  | 2024  | 0.98       |
| im1      | im1      | 5          | im3     | 2048  | 2024  | 0.98       |
| im1      | im2      | 5          | im3     | 8     | 121   | 15.12      |
| im1      | im2      | 5          | im3     | 8     | 121   | 15.12      |
| im1      | im2      | 5          | im3     | 128   | 132   | 1.03       |
| im1      | im2      | 5          | im3     | 128   | 132   | 1.03       |
| im1      | im2      | 5          | im3     | 1024  | 580   | 0.56       |

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im2      | 5          | im3     | 1024  | 580   | 0.56       |
| im1      | im2      | 5          | im3     | 2048  | 1092  | 0.53       |
| im1      | im2      | 5          | im3     | 2048  | 1092  | 0.53       |

The best configuration:

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | 5          | im3     | 8     | 121   | 15.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | nValueAddC | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|------------|---------|-------|-------|------------|
| im1      | im1      | 5          | im3     | 2048  | 2024  | 0.98       |

## 6.196 nmppsMulC\_AddC

Умножение вектора на константу с прибавлением константы.

## Функции

- void nmppsMulC\_AddC\_32s (const nm32s \*pSrcVec, int nMulVal, int nAddVal, nm32s \*pDstVec, int nSize)
- void nmppsMulC\_AddC\_2x32s (int32x2 \*dataSparseSrc, int32x2 \*mulArg, int32x2 \*addArg, int32x2 \*dataSparseDst, int size, int stepSparseSrc, int stepSparseDst)

Sparse vector by constant multiplication with addition of constant.

$$dataSparseDst[i \cdot stepSparseDst][k] = dataSparseSrc[i \cdot stepSparseSrc][k] \cdot mulArg[k] + addArg[k],$$

$$i = \overline{0 \dots size - 1}; k = \overline{0 \dots K - 1},$$

where K is value of intWxK type.

## 6.196.1 Подробное описание

Умножение вектора на константу с прибавлением константы.

$$pDstVec[i] = nMulVal \cdot pSrcVec[i] + nAddVal,$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nMulVal | Константа-множитель.         |
| nAddVal | Добавляемая константа.       |
| nSize   | Размер векторов в элементах. |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.196.2 Функции

## 6.196.2.1 nmppsMulC\_AddC\_2x32s()

```
void nmppsMulC_AddC_2x32s (
    int32x2 * dataSparseSrc,
    int32x2 * mulArg,
    int32x2 * addArg,
    int32x2 * dataSparseDst,
    int size,
    int stepSparseSrc,
    int stepSparseDst )
```

Sparse vector by constant multiplication with addition of constant.

$$dataSparseDst[i \cdot stepSparseDst][k] = dataSparseSrc[i \cdot stepSparseSrc][k] \cdot mulArg[k] + addArg[k],$$

$$i = \overline{0 \dots size - 1}; k = \overline{0 \dots K - 1},$$

where K is value of int WxK type.

## Аргументы

|    |               |                                                                                                        |
|----|---------------|--------------------------------------------------------------------------------------------------------|
| in | dataSparseSrc | Input sparse vector of 64-bit packed words                                                             |
| in | mulArg        | Packed 64-bit word with values to multiply                                                             |
| in | addArg        | Packed 64-bit word with values to add                                                                  |
| in | dataSparseDst | Ouput sparse vector of 64-bit packed words                                                             |
| in | size          | actual amount of 64-bit packed words in sparse vector to be processed                                  |
| in | stepSparseSrc | 64-bit step between input packed words in memory . By default=1 means that input vector is continuous  |
| in | stepSparseDst | 64-bit step between output packed words in memory. By default=1 means that output vector is continuous |

## Возвращает

void

## 6.197 nmppsRShiftC\_MulC\_AddC

## Функции

- void nmppsRShiftC\_MulC\_AddC\_2x32s (int32x2 \*dataSparseSrc, int32x2 \*preshiftArg, int32x2 \*mulArg, int32x2 \*addArg, int32x2 \*dataSparseDst, int size, int stepSparseSrc, int stepSparseDst)

## 6.197.1 Подробное описание

Sparse vector by constant multiplication with addition of constant.

$$dataSparseDst[i \cdot stepSparseDst][k] = (dataSparseSrc[i \cdot stepSparseSrc][k] \gg preshiftArg[k]) \cdot mulArg[k] + addArg[k],$$

$$i = \overline{0 \dots size - 1}; k = \overline{0 \dots K - 1},$$

where K is value of int WxK type

## Аргументы

|    |               |                                                                                                        |
|----|---------------|--------------------------------------------------------------------------------------------------------|
| in | dataSparseSrc | Input sparse vector of 64-bit packed words                                                             |
| in | preshiftArg   | Packed 64-bit word of values for preshifting of input data = [2,4,6,8...28,30]                         |
| in | mulArg        | Packed 64-bit word with values to multiply                                                             |
| in | addArg        | Packed 64-bit word with values to add                                                                  |
| in | dataSparseDst | Ouput sparse vector of 64-bit packed words                                                             |
| in | size          | actual amount of 64-bit packed words in sparse vector to be processed                                  |
| in | stepSparseSrc | 64-bit step between input packed words in memory . By default=1 means that input vector is continuous  |
| in | stepSparseDst | 64-bit step between output packed words in memory. By default=1 means that output vector is continuous |

## Возвращает

void

## 6.198 nmppsMulC\_AddV\_AddC

Умножение вектора на константу с прибавлением вектора и константы.

Функции

- void `nmppsMulC_AddV_AddC_32s` (`nm32s` \*pSrcVec1, int nMulVal, `nm32s` \*pSrcVec2, int nAddVal, `nm32s` \*pDstVec, int nSize)

### 6.198.1 Подробное описание

Умножение вектора на константу с прибавлением вектора и константы.

$$pDstVec[i] = nMulVal \cdot pSrcVec1[i] + pSrcVec2[i] + nAddVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| nMulVal  | Константа-множитель.         |
| pSrcVec2 | Второй входной вектор.       |
| nAddVal  | Добавляемая константа.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.198.2 Функции

## 6.198.2.1 nmppsMulC\_AddV\_AddC\_32s()

```
void nmppsMulC_AddV_AddC_32s (
    nm32s * pSrcVec1,
    int nMulVal,
    nm32s * pSrcVec2,
    int nAddVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im0      | 5       | im0     | 2048  | 3153  | 1.53       |
| im0      | 5       | im0      | 5       | im0     | 2048  | 3153  | 1.53       |
| im0      | 5       | im0      | 5       | im2     | 2048  | 2192  | 1.07       |
| im0      | 5       | im0      | 5       | im2     | 2048  | 2192  | 1.07       |
| im0      | 5       | im1      | 5       | im0     | 2048  | 3158  | 1.54       |
| im0      | 5       | im1      | 5       | im0     | 2048  | 3158  | 1.54       |
| im0      | 5       | im1      | 5       | im2     | 2048  | 2374  | 1.15       |
| im0      | 5       | im1      | 5       | im2     | 2048  | 2374  | 1.15       |
| im1      | 5       | im0      | 5       | im0     | 2048  | 2200  | 1.07       |
| im1      | 5       | im0      | 5       | im0     | 2048  | 2200  | 1.07       |
| im1      | 5       | im0      | 5       | im2     | 2048  | 2200  | 1.07       |
| im1      | 5       | im0      | 5       | im2     | 2048  | 2200  | 1.07       |
| im1      | 5       | im1      | 5       | im0     | 2048  | 2295  | 1.12       |
| im1      | 5       | im1      | 5       | im0     | 2048  | 2295  | 1.12       |
| im1      | 5       | im1      | 5       | im2     | 2048  | 2295  | 1.12       |
| im1      | 5       | im1      | 5       | im2     | 2048  | 2295  | 1.12       |

The best configuration:

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im0      | 5       | im2     | 2048  | 2192  | 1.07       |

The worst configuration:

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im1      | 5       | im0     | 2048  | 3158  | 1.54       |

Perfomance table 1

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im1      | 5       | im2     | 8     | 160   | 20.00      |
| im0      | 5       | im1      | 5       | im2     | 8     | 160   | 20.00      |
| im0      | 5       | im1      | 5       | im2     | 128   | 278   | 2.17       |
| im0      | 5       | im1      | 5       | im2     | 128   | 278   | 2.17       |
| im0      | 5       | im1      | 5       | im2     | 1024  | 1254  | 1.22       |
| im0      | 5       | im1      | 5       | im2     | 1024  | 1254  | 1.22       |
| im0      | 5       | im1      | 5       | im2     | 2048  | 2374  | 1.15       |
| im0      | 5       | im1      | 5       | im2     | 2048  | 2374  | 1.15       |

The best configuration:

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im1      | 5       | im2     | 8     | 160   | 20.00      |

The worst configuration:

| pSrcVec1 | nMulVal | pSrcVec2 | nAddVal | pDstVec | nSize | ticks | ticks/elem |
|----------|---------|----------|---------|---------|-------|-------|------------|
| im0      | 5       | im1      | 5       | im2     | 2048  | 2374  | 1.15       |



## 6.199 nmppsSumN

Сложение нескольких векторов.

### Функции

- void nmppsSumN\_8s16s (nm8s \*\*ppSrcVec, nm16s \*pDstVec, int nSize, int nNumberOfVectors)
- void nmppsSumN\_16s (nm16s \*\*ppSrcVec, nm16s \*pDstVec, int nSize, int nNumberOfVectors)

### 6.199.1 Подробное описание

Сложение нескольких векторов.

$$pDstVec[i] = \sum_{j=0}^{(nNumberOfVectors-1)} ppSrcVec(j)(i)$$

### Аргументы

|                  |                                           |
|------------------|-------------------------------------------|
| ppSrcVec         | Массив указателей на суммируемые вектора. |
| nNumberOfVectors | Число суммируемых векторов.               |
| nSize            | Размер векторов в элементах = [32*PACK]   |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

## 6.200 nmppsDivC

Деление вектора на константу.

Функции

- void `nmppsDivC_32s` (`nm32s` \*pSrcVec, int nDivisor, `nm32s` \*pDstVec, int nSize, void \*pTmpBuf1, void \*pTmpBuf2)

### 6.200.1 Подробное описание

Деление вектора на константу.

$$pDstVec[i] = \frac{pSrcVec[i]}{Divisor},$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                                                      |
|----------|------------------------------------------------------|
| pSrcVec  | Входной вектор.                                      |
| nDivisor | Константа-делитель.                                  |
| nSize    | Размер входного вектора в элементах.                 |
| pTmpBuf1 | Временный массив размером nSize 64-х разрядных слов. |
| pTmpBuf2 | Временный массив размером nSize 64-х разрядных слов. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

Restrictions:

- Допустимые значения для элементов входного вектора лежат в диапазоне [-4095,...,4095];
- Допустимые значения для делителя лежат в диапазоне [1,2,..145].

Заметки

Для корректного использования in-place параметров следует учитывать порядок получения промежуточных результатов:

the order of obtaining intermediate results:

pSrcVec => pTmpBuf1 (1cpl:L<=>G) - 1/x Multiplying (in-place is supported)

pTmpBuf1 => pTmpBuf2 (1cpl:G<=>L) - Scaling down (in-place is supported)

pTmpBuf2 => pDstVec (2cpl:L<=>G) - Result correction (in-place is supported)

Примеры использования in-place параметров:

```
nmppsDiv_(L0,G0,10240,3,G0,L0);
```

```
nmppsDiv_(L0,L0,10240,3,L0,L0);
```

## 6.200.2 Функции

### 6.200.2.1 nmppsDivC\_32s()

```
void nmppsDivC_32s (
    nm32s * pSrcVec,
    int nDivisor,
    nm32s * pDstVec,
    int nSize,
    void * pTmpBuf1,
    void * pTmpBuf2 )
```

Perfomance table 0

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im0     | 5        | im0      | im0      | 4096  | 14329 | 3.49       |
| im0     | im0     | 5        | im0      | im0      | 4096  | 14329 | 3.49       |
| im0     | im0     | 5        | im0      | im3      | 4096  | 10449 | 2.55       |
| im0     | im0     | 5        | im0      | im3      | 4096  | 10449 | 2.55       |
| im0     | im0     | 5        | im2      | im0      | 4096  | 10301 | 2.51       |
| im0     | im0     | 5        | im2      | im0      | 4096  | 10301 | 2.51       |
| im0     | im0     | 5        | im2      | im3      | 4096  | 8441  | 2.06       |
| im0     | im0     | 5        | im2      | im3      | 4096  | 8441  | 2.06       |
| im0     | im1     | 5        | im0      | im0      | 4096  | 12536 | 3.06       |
| im0     | im1     | 5        | im0      | im0      | 4096  | 12536 | 3.06       |
| im0     | im1     | 5        | im0      | im3      | 4096  | 10512 | 2.56       |
| im0     | im1     | 5        | im0      | im3      | 4096  | 10512 | 2.56       |
| im0     | im1     | 5        | im2      | im0      | 4096  | 8504  | 2.07       |
| im0     | im1     | 5        | im2      | im0      | 4096  | 8504  | 2.07       |
| im0     | im1     | 5        | im2      | im3      | 4096  | 8504  | 2.07       |
| im0     | im1     | 5        | im2      | im3      | 4096  | 8504  | 2.07       |
| im1     | im0     | 5        | im0      | im0      | 4096  | 12476 | 3.04       |
| im1     | im0     | 5        | im0      | im0      | 4096  | 12476 | 3.04       |
| im1     | im0     | 5        | im0      | im3      | 4096  | 8596  | 2.09       |
| im1     | im0     | 5        | im0      | im3      | 4096  | 8596  | 2.09       |
| im1     | im0     | 5        | im2      | im0      | 4096  | 10456 | 2.55       |
| im1     | im0     | 5        | im2      | im0      | 4096  | 10456 | 2.55       |
| im1     | im0     | 5        | im2      | im3      | 4096  | 8596  | 2.09       |
| im1     | im0     | 5        | im2      | im3      | 4096  | 8596  | 2.09       |
| im1     | im1     | 5        | im0      | im0      | 4096  | 10683 | 2.60       |

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im1     | im1     | 5        | im0      | im0      | 4096  | 10683 | 2.60       |
| im1     | im1     | 5        | im0      | im3      | 4096  | 8659  | 2.11       |
| im1     | im1     | 5        | im0      | im3      | 4096  | 8659  | 2.11       |
| im1     | im1     | 5        | im2      | im0      | 4096  | 8659  | 2.11       |
| im1     | im1     | 5        | im2      | im0      | 4096  | 8659  | 2.11       |
| im1     | im1     | 5        | im2      | im3      | 4096  | 8659  | 2.11       |
| im1     | im1     | 5        | im2      | im3      | 4096  | 8659  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im0     | 5        | im2      | im3      | 4096  | 8441  | 2.06       |

The worst configuration:

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im0     | 5        | im0      | im0      | 4096  | 14329 | 3.49       |

Performance table 1

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im1     | 5        | im2      | im3      | 8     | 288   | 36.00      |
| im0     | im1     | 5        | im2      | im3      | 8     | 288   | 36.00      |
| im0     | im1     | 5        | im2      | im3      | 128   | 510   | 3.98       |
| im0     | im1     | 5        | im2      | im3      | 128   | 510   | 3.98       |
| im0     | im1     | 5        | im2      | im3      | 1024  | 2312  | 2.25       |
| im0     | im1     | 5        | im2      | im3      | 1024  | 2312  | 2.25       |
| im0     | im1     | 5        | im2      | im3      | 2048  | 4376  | 2.13       |
| im0     | im1     | 5        | im2      | im3      | 2048  | 4376  | 2.13       |
| im0     | im1     | 5        | im2      | im3      | 4096  | 8504  | 2.07       |
| im0     | im1     | 5        | im2      | im3      | 4096  | 8504  | 2.07       |

The best configuration:

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im1     | 5        | im2      | im3      | 8     | 288   | 36.00      |

The worst configuration:

| pSrcVec | pDstVec | nDivisor | pTmpBuf1 | pTmpBuf2 | nSize | ticks | ticks/elem |
|---------|---------|----------|----------|----------|-------|-------|------------|
| im0     | im1     | 5        | im2      | im3      | 4096  | 8504  | 2.07       |

## 6.201 nmppsSum

Возвращает сумму всех элементов вектора.

### Функции

- void `nmppsSum_8s` (const `nm8s` \*pSrcVec, int nSize, `int32b` \*pnRes)
- void `nmppsSum_16s` (const `nm16s` \*pSrcVec, int nSize, `int64b` \*pnRes)
- void `nmppsSum_32s` (const `nm32s` \*pSrcVec, int nSize, `int64b` \*pnRes)
- void `nmppsSum_64s` (const `nm64s` \*pSrcVec, int nSize, `int64b` \*pnRes)

### 6.201.1 Подробное описание

Возвращает сумму всех элементов вектора.

$$return = \sum_{i=0}^{(nSize-1)} pSrcVec[i]$$

### Аргументы

|         |                                                     |
|---------|-----------------------------------------------------|
| pSrcVec | Входной вектор.                                     |
| pTmpBuf | Временный массив размера nSize 64-х разрядных слов. |
| nSize   | Размер векторов в элементах.                        |

Возвращает

Сумма элементов вектора.

### 6.201.2 Функции

#### 6.201.2.1 nmppsSum\_16s()

```
void nmppsSum_16s (
    const nm16s * pSrcVec,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 901   | 0.43       |
| im0     | im2   | 2048  | 901   | 0.43       |
| im1     | im2   | 2048  | 941   | 0.45       |
| im1     | im2   | 2048  | 941   | 0.45       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 901   | 0.43       |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 941   | 0.45       |

Perfomance table 1

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 181   | 22.62      |
| im1     | im2   | 8     | 181   | 22.62      |
| im1     | im2   | 128   | 433   | 3.38       |
| im1     | im2   | 128   | 433   | 3.38       |
| im1     | im2   | 1024  | 669   | 0.65       |
| im1     | im2   | 1024  | 669   | 0.65       |
| im1     | im2   | 2048  | 941   | 0.45       |
| im1     | im2   | 2048  | 941   | 0.45       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 181   | 22.62      |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 941   | 0.45       |

6.201.2.2 nmppsSum\_32s()

```
void nmppsSum_32s (
    const nm32s * pSrcVec,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 1411  | 0.68       |
| im0     | im2   | 2048  | 1411  | 0.68       |
| im1     | im2   | 2048  | 1483  | 0.72       |
| im1     | im2   | 2048  | 1483  | 0.72       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 1411  | 0.68       |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 1483  | 0.72       |

Perfomance table 1

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 194   | 24.25      |
| im1     | im2   | 8     | 194   | 24.25      |
| im1     | im2   | 128   | 462   | 3.60       |
| im1     | im2   | 128   | 462   | 3.60       |
| im1     | im2   | 1024  | 939   | 0.91       |
| im1     | im2   | 1024  | 939   | 0.91       |
| im1     | im2   | 2048  | 1483  | 0.72       |
| im1     | im2   | 2048  | 1483  | 0.72       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 194   | 24.25      |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 1483  | 0.72       |

### 6.201.2.3 nmppsSum\_64s()

```
void nmppsSum_64s (
    const nm64s * pSrcVec,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 2330  | 1.13       |
| im0     | im2   | 2048  | 2330  | 1.13       |
| im1     | im2   | 2048  | 2461  | 1.20       |
| im1     | im2   | 2048  | 2461  | 1.20       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 2330  | 1.13       |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 2461  | 1.20       |

Perfomance table 1

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 161   | 20.12      |
| im1     | im2   | 8     | 161   | 20.12      |
| im1     | im2   | 128   | 421   | 3.28       |
| im1     | im2   | 128   | 421   | 3.28       |
| im1     | im2   | 1024  | 1373  | 1.34       |
| im1     | im2   | 1024  | 1373  | 1.34       |
| im1     | im2   | 2048  | 2461  | 1.20       |
| im1     | im2   | 2048  | 2461  | 1.20       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 161   | 20.12      |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 2461  | 1.20       |

#### 6.201.2.4 nmppsSum\_8s()

```
void nmppsSum_8s (
    const nm8s * pSrcVec,
    int nSize,
    int32b * pnRes )
```

Perfomance table 0



| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 654   | 0.31       |
| im0     | im2   | 2048  | 654   | 0.31       |
| im1     | im2   | 2048  | 678   | 0.33       |
| im1     | im2   | 2048  | 678   | 0.33       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im0     | im2   | 2048  | 654   | 0.31       |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 678   | 0.33       |

Perfomance table 1

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 183   | 22.87      |
| im1     | im2   | 8     | 183   | 22.87      |
| im1     | im2   | 128   | 311   | 2.42       |
| im1     | im2   | 128   | 311   | 2.42       |
| im1     | im2   | 1024  | 542   | 0.52       |
| im1     | im2   | 1024  | 542   | 0.52       |
| im1     | im2   | 2048  | 678   | 0.33       |
| im1     | im2   | 2048  | 678   | 0.33       |

The best configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 8     | 183   | 22.87      |

The worst configuration:

| pSrcVec | pnRes | nSize | ticks | ticks/elem |
|---------|-------|-------|-------|------------|
| im1     | im2   | 2048  | 678   | 0.33       |

## 6.202 nmppsSum\_1

Возвращает сумму всех элементов вектора.

### Функции

- void nmppsSum\_1s (const nm1 \*pSrcVec, int nSize, int32b \*pnRes, void \*pTmpBuf)

### 6.202.1 Подробное описание

Возвращает сумму всех элементов вектора.

$$return = \sum_{i=0}^{(nSize-1)} pSrcVec[i]$$

### Аргументы

|         |                                                     |
|---------|-----------------------------------------------------|
| pSrcVec | Входной вектор.                                     |
| pTmpBuf | Временный массив размера nSize 64-х разрядных слов. |
| nSize   | Размер векторов в элементах.                        |

Возвращает

Сумма элементов вектора.

## 6.203 nmppsDotProd\_sm

Скалярное умножение двух векторов.

### Функции

- `int nmppsDotProd_8s8sm` (`const nm8s *pSrcVec1`, `const nm8s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)
- `int nmppsDotProd_8s16sm` (`const nm8s *pSrcVec1`, `const nm16s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)
- `int nmppsDotProd_8s32sm` (`const nm8s *pSrcVec1`, `const nm32s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)
- `int nmppsDotProd_16s16sm` (`const nm16s *pSrcVec1`, `const nm16s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)
- `int nmppsDotProd_16s32sm` (`const nm16s *pSrcVec1`, `const nm32s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)
- `int nmppsDotProd_32s32sm` (`const nm32s *pSrcVec1`, `const nm32s *pSrcVec2`, `int nSize`, `int64b *pnRes`, `nm64s *tmp`)

### 6.203.1 Подробное описание

Скалярное умножение двух векторов.

$$nRes = \sum_{i=0}^{nSize-1} pSrcVec1[i] \cdot pSrcVec2[i]$$

### Аргументы

|                       |                              |
|-----------------------|------------------------------|
| <code>pSrcVec1</code> | Первый вектор.               |
| <code>pSrcVec2</code> | Второй вектор.               |
| <code>nSize</code>    | Размер векторов в элементах. |

### Возвращаемые значения

|                    |                                       |
|--------------------|---------------------------------------|
| <code>pnRes</code> | Указатель на результирующее значение. |
|--------------------|---------------------------------------|

### Возвращает

`pTmpBuff` Временный массив из `nSize` элементов.  
`void`

### 6.203.2 Функции

## 6.203.2.1 nmppsDotProd\_16s16sm()

```
int nmppsDotProd_16s16sm (
    const nm16s * pSrcVec1,
    const nm16s * pSrcVec2,
    int nSize,
    int64b * pnRes,
    nm64s * tmp )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 15568 | 3.80       |
| im0      | im0      | im0 | im3   | 4096  | 15568 | 3.80       |
| im0      | im0      | im2 | im3   | 4096  | 11545 | 2.81       |
| im0      | im0      | im2 | im3   | 4096  | 11545 | 2.81       |
| im0      | im1      | im0 | im3   | 4096  | 11909 | 2.90       |
| im0      | im1      | im0 | im3   | 4096  | 11909 | 2.90       |
| im0      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |
| im0      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |
| im1      | im0      | im0 | im3   | 4096  | 15568 | 3.80       |
| im1      | im0      | im0 | im3   | 4096  | 15568 | 3.80       |
| im1      | im0      | im2 | im3   | 4096  | 11545 | 2.81       |
| im1      | im0      | im2 | im3   | 4096  | 11545 | 2.81       |
| im1      | im1      | im0 | im3   | 4096  | 11909 | 2.90       |
| im1      | im1      | im0 | im3   | 4096  | 11909 | 2.90       |
| im1      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |
| im1      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 11545 | 2.81       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 15568 | 3.80       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 413   | 51.62      |
| im0      | im1      | im2 | im3   | 8     | 413   | 51.62      |
| im0      | im1      | im2 | im3   | 128   | 745   | 5.82       |
| im0      | im1      | im2 | im3   | 128   | 745   | 5.82       |
| im0      | im1      | im2 | im3   | 1024  | 3269  | 3.19       |
| im0      | im1      | im2 | im3   | 1024  | 3269  | 3.19       |
| im0      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |
| im0      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 413   | 51.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 11909 | 2.90       |

### 6.203.2.2 nmppsDotProd\_16s32sm()

```
int nmppsDotProd_16s32sm (
    const nm16s * pSrcVec1,
    const nm32s * pSrcVec2,
    int nSize,
    int64b * pnRes,
    nm64s * tmp )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 15565 | 3.80       |
| im0      | im0      | im0 | im3   | 4096  | 15565 | 3.80       |
| im0      | im0      | im2 | im3   | 4096  | 11512 | 2.81       |
| im0      | im0      | im2 | im3   | 4096  | 11512 | 2.81       |
| im0      | im1      | im0 | im3   | 4096  | 11822 | 2.88       |
| im0      | im1      | im0 | im3   | 4096  | 11822 | 2.88       |
| im0      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |
| im0      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |
| im1      | im0      | im0 | im3   | 4096  | 15565 | 3.80       |
| im1      | im0      | im0 | im3   | 4096  | 15565 | 3.80       |
| im1      | im0      | im2 | im3   | 4096  | 11512 | 2.81       |
| im1      | im0      | im2 | im3   | 4096  | 11512 | 2.81       |
| im1      | im1      | im0 | im3   | 4096  | 11822 | 2.88       |
| im1      | im1      | im0 | im3   | 4096  | 11822 | 2.88       |
| im1      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |
| im1      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 11512 | 2.81       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 15565 | 3.80       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 326   | 40.75      |
| im0      | im1      | im2 | im3   | 8     | 326   | 40.75      |
| im0      | im1      | im2 | im3   | 128   | 662   | 5.17       |
| im0      | im1      | im2 | im3   | 128   | 662   | 5.17       |
| im0      | im1      | im2 | im3   | 1024  | 3182  | 3.10       |
| im0      | im1      | im2 | im3   | 1024  | 3182  | 3.10       |
| im0      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |
| im0      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 326   | 40.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 11822 | 2.88       |

### 6.203.2.3 nmppsDotProd\_32s32sm()

```
int nmppsDotProd_32s32sm (
    const nm32s * pSrcVec1,
    const nm32s * pSrcVec2,
    int nSize,
    int64b * pnRes,
    nm64s * tmp )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 22734 | 5.55       |
| im0      | im0      | im0 | im3   | 4096  | 22734 | 5.55       |
| im0      | im0      | im2 | im3   | 4096  | 18681 | 4.56       |
| im0      | im0      | im2 | im3   | 4096  | 18681 | 4.56       |
| im0      | im1      | im0 | im3   | 4096  | 18991 | 4.63       |
| im0      | im1      | im0 | im3   | 4096  | 18991 | 4.63       |
| im0      | im1      | im2 | im3   | 4096  | 18991 | 4.63       |
| im0      | im1      | im2 | im3   | 4096  | 18991 | 4.63       |
| im1      | im0      | im0 | im3   | 4096  | 22735 | 5.55       |

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im1      | im0      | im0 | im3   | 4096  | 22735 | 5.55       |
| im1      | im0      | im2 | im3   | 4096  | 18682 | 4.56       |
| im1      | im0      | im2 | im3   | 4096  | 18682 | 4.56       |
| im1      | im1      | im0 | im3   | 4096  | 18992 | 4.63       |
| im1      | im1      | im0 | im3   | 4096  | 18992 | 4.63       |
| im1      | im1      | im2 | im3   | 4096  | 18992 | 4.63       |
| im1      | im1      | im2 | im3   | 4096  | 18992 | 4.63       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 18681 | 4.56       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im1      | im0      | im0 | im3   | 4096  | 22735 | 5.55       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 338   | 42.25      |
| im0      | im1      | im2 | im3   | 8     | 338   | 42.25      |
| im0      | im1      | im2 | im3   | 128   | 887   | 6.92       |
| im0      | im1      | im2 | im3   | 128   | 887   | 6.92       |
| im0      | im1      | im2 | im3   | 1024  | 4975  | 4.85       |
| im0      | im1      | im2 | im3   | 1024  | 4975  | 4.85       |
| im0      | im1      | im2 | im3   | 4096  | 18991 | 4.63       |
| im0      | im1      | im2 | im3   | 4096  | 18991 | 4.63       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 338   | 42.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 18991 | 4.63       |

#### 6.203.2.4 nmppsDotProd\_8s16sm()

```
int nmppsDotProd_8s16sm (
    const nm8s * pSrcVec1,
```

```

const nm16s * pSrcVec2,
int nSize,
int64b * pnRes,
nm64s * tmp )

```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13017 | 3.17       |
| im0      | im0      | im0 | im3   | 4096  | 13017 | 3.17       |
| im0      | im0      | im2 | im3   | 4096  | 8479  | 2.07       |
| im0      | im0      | im2 | im3   | 4096  | 8479  | 2.07       |
| im0      | im1      | im0 | im3   | 4096  | 9353  | 2.28       |
| im0      | im1      | im0 | im3   | 4096  | 9353  | 2.28       |
| im0      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |
| im0      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |
| im1      | im0      | im0 | im3   | 4096  | 12506 | 3.05       |
| im1      | im0      | im0 | im3   | 4096  | 12506 | 3.05       |
| im1      | im0      | im2 | im3   | 4096  | 8479  | 2.07       |
| im1      | im0      | im2 | im3   | 4096  | 8479  | 2.07       |
| im1      | im1      | im0 | im3   | 4096  | 8843  | 2.15       |
| im1      | im1      | im0 | im3   | 4096  | 8843  | 2.15       |
| im1      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |
| im1      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 8479  | 2.07       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13017 | 3.17       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 410   | 51.25      |
| im0      | im1      | im2 | im3   | 8     | 410   | 51.25      |
| im0      | im1      | im2 | im3   | 128   | 655   | 5.11       |
| im0      | im1      | im2 | im3   | 128   | 655   | 5.11       |
| im0      | im1      | im2 | im3   | 1024  | 2507  | 2.44       |
| im0      | im1      | im2 | im3   | 1024  | 2507  | 2.44       |
| im0      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |
| im0      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |

The best configuration:



| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 410   | 51.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 8843  | 2.15       |

### 6.203.2.5 nmppsDotProd\_8s32sm()

```
int nmppsDotProd_8s32sm (
    const nm8s * pSrcVec1,
    const nm32s * pSrcVec2,
    int nSize,
    int64b * pnRes,
    nm64s * tmp )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13014 | 3.17       |
| im0      | im0      | im0 | im3   | 4096  | 13014 | 3.17       |
| im0      | im0      | im2 | im3   | 4096  | 8446  | 2.06       |
| im0      | im0      | im2 | im3   | 4096  | 8446  | 2.06       |
| im0      | im1      | im0 | im3   | 4096  | 9266  | 2.26       |
| im0      | im1      | im0 | im3   | 4096  | 9266  | 2.26       |
| im0      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |
| im0      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |
| im1      | im0      | im0 | im3   | 4096  | 12503 | 3.05       |
| im1      | im0      | im0 | im3   | 4096  | 12503 | 3.05       |
| im1      | im0      | im2 | im3   | 4096  | 8446  | 2.06       |
| im1      | im0      | im2 | im3   | 4096  | 8446  | 2.06       |
| im1      | im1      | im0 | im3   | 4096  | 8756  | 2.13       |
| im1      | im1      | im0 | im3   | 4096  | 8756  | 2.13       |
| im1      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |
| im1      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 8446  | 2.06       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13014 | 3.17       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 323   | 40.37      |
| im0      | im1      | im2 | im3   | 8     | 323   | 40.37      |
| im0      | im1      | im2 | im3   | 128   | 572   | 4.46       |
| im0      | im1      | im2 | im3   | 128   | 572   | 4.46       |
| im0      | im1      | im2 | im3   | 1024  | 2420  | 2.36       |
| im0      | im1      | im2 | im3   | 1024  | 2420  | 2.36       |
| im0      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |
| im0      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 323   | 40.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 8756  | 2.13       |

#### 6.203.2.6 nmppsDotProd\_8s8sm()

```
int nmppsDotProd_8s8sm (
    const nm8s * pSrcVec1,
    const nm8s * pSrcVec2,
    int nSize,
    int64b * pnRes,
    nm64s * tmp )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13036 | 3.18       |
| im0      | im0      | im0 | im3   | 4096  | 13036 | 3.18       |
| im0      | im0      | im2 | im3   | 4096  | 8571  | 2.09       |
| im0      | im0      | im2 | im3   | 4096  | 8571  | 2.09       |
| im0      | im1      | im0 | im3   | 4096  | 9553  | 2.33       |
| im0      | im1      | im0 | im3   | 4096  | 9553  | 2.33       |
| im0      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |
| im0      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |
| im1      | im0      | im0 | im3   | 4096  | 12525 | 3.05       |
| im1      | im0      | im0 | im3   | 4096  | 12525 | 3.05       |
| im1      | im0      | im2 | im3   | 4096  | 8571  | 2.09       |
| im1      | im0      | im2 | im3   | 4096  | 8571  | 2.09       |
| im1      | im1      | im0 | im3   | 4096  | 9043  | 2.20       |
| im1      | im1      | im0 | im3   | 4096  | 9043  | 2.20       |
| im1      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |
| im1      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im2 | im3   | 4096  | 8571  | 2.09       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im0      | im0 | im3   | 4096  | 13036 | 3.18       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 613   | 76.62      |
| im0      | im1      | im2 | im3   | 8     | 613   | 76.62      |
| im0      | im1      | im2 | im3   | 128   | 826   | 6.45       |
| im0      | im1      | im2 | im3   | 128   | 826   | 6.45       |
| im0      | im1      | im2 | im3   | 1024  | 2707  | 2.64       |
| im0      | im1      | im2 | im3   | 1024  | 2707  | 2.64       |
| im0      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |
| im0      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |

The best configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 8     | 613   | 76.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | tmp | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-----|-------|-------|-------|------------|
| im0      | im1      | im2 | im3   | 4096  | 9043  | 2.20       |

## 6.204 nmppsDotProd

Скалярное умножение двух векторов.

### Функции

- void `nmppsDotProd_8s64s` (const `nm8s` \*pSrcVec1, const `nm64s` \*pSrcVec2, int nSize, `int64b` \*pnRes)
- void `nmppsDotProd_16s64s` (const `nm16s` \*pSrcVec1, const `nm64s` \*pSrcVec2, int nSize, `int64b` \*pnRes)
- void `nmppsDotProd_32s64s` (const `nm32s` \*pSrcVec1, const `nm64s` \*pSrcVec2, int nSize, `int64b` \*pnRes)
- void `nmppsDotProd_64s64s` (const `nm64s` \*pSrcVec1, const `nm64s` \*pSrcVec2, int nSize, `int64b` \*pnRes)

### 6.204.1 Подробное описание

Скалярное умножение двух векторов.

$$nRes = \sum_{i=0}^{nSize-1} pSrcVec1[i] \cdot pSrcVec2[i]$$

### Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый вектор.               |
| pSrcVec2 | Второй вектор.               |
| nSize    | Размер векторов в элементах. |

### Возвращаемые значения

|       |                                       |
|-------|---------------------------------------|
| pnRes | Указатель на результирующее значение. |
|-------|---------------------------------------|

### Возвращает

pTmpBuff Временный массив из nSize элементов.  
void

### 6.204.2 Функции

## 6.204.2.1 nmppsDotProd\_16s64s()

```
void nmppsDotProd_16s64s (
    const nm16s * pSrcVec1,
    const nm64s * pSrcVec2,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 3675  | 1.79       |
| im0      | im0      | im2   | 2048  | 3675  | 1.79       |
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |
| im1      | im0      | im2   | 2048  | 3675  | 1.79       |
| im1      | im0      | im2   | 2048  | 3675  | 1.79       |
| im1      | im1      | im2   | 2048  | 3692  | 1.80       |
| im1      | im1      | im2   | 2048  | 3692  | 1.80       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 3675  | 1.79       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 122   | 15.25      |
| im0      | im1      | im2   | 8     | 122   | 15.25      |
| im0      | im1      | im2   | 128   | 323   | 2.52       |
| im0      | im1      | im2   | 128   | 323   | 2.52       |
| im0      | im1      | im2   | 1024  | 1904  | 1.85       |
| im0      | im1      | im2   | 1024  | 1904  | 1.85       |
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 122   | 15.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 2048  | 3707  | 1.81       |

#### 6.204.2.2 nmppsDotProd\_32s64s()

```
void nmppsDotProd_32s64s (
    const nm32s * pSrcVec1,
    const nm64s * pSrcVec2,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 7260  | 3.54       |
| im0      | im0      | im2   | 2048  | 7260  | 3.54       |
| im0      | im1      | im2   | 2048  | 7260  | 3.54       |
| im0      | im1      | im2   | 2048  | 7260  | 3.54       |
| im1      | im0      | im2   | 2048  | 7261  | 3.54       |
| im1      | im0      | im2   | 2048  | 7261  | 3.54       |
| im1      | im1      | im2   | 2048  | 7261  | 3.54       |
| im1      | im1      | im2   | 2048  | 7261  | 3.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 7260  | 3.54       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im1      | im0      | im2   | 2048  | 7261  | 3.54       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 123   | 15.37      |
| im0      | im1      | im2   | 8     | 123   | 15.37      |
| im0      | im1      | im2   | 128   | 540   | 4.21       |
| im0      | im1      | im2   | 128   | 540   | 4.21       |
| im0      | im1      | im2   | 1024  | 3676  | 3.58       |
| im0      | im1      | im2   | 1024  | 3676  | 3.58       |
| im0      | im1      | im2   | 2048  | 7260  | 3.54       |
| im0      | im1      | im2   | 2048  | 7260  | 3.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 123   | 15.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 2048  | 7260  | 3.54       |

### 6.204.2.3 nmppsDotProd\_64s64s()

```
void nmppsDotProd_64s64s (
    const nm64s * pSrcVec1,
    const nm64s * pSrcVec2,
    int nSize,
    int64b * pnRes )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 14427 | 7.04       |
| im0      | im0      | im2   | 2048  | 14427 | 7.04       |
| im0      | im1      | im2   | 2048  | 14427 | 7.04       |
| im0      | im1      | im2   | 2048  | 14427 | 7.04       |
| im1      | im0      | im2   | 2048  | 14427 | 7.04       |
| im1      | im0      | im2   | 2048  | 14427 | 7.04       |
| im1      | im1      | im2   | 2048  | 14427 | 7.04       |
| im1      | im1      | im2   | 2048  | 14427 | 7.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 14427 | 7.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 14427 | 7.04       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 147   | 18.37      |
| im0      | im1      | im2   | 8     | 147   | 18.37      |
| im0      | im1      | im2   | 128   | 987   | 7.71       |
| im0      | im1      | im2   | 128   | 987   | 7.71       |

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 1024  | 7259  | 7.08       |
| im0      | im1      | im2   | 1024  | 7259  | 7.08       |
| im0      | im1      | im2   | 2048  | 14427 | 7.04       |
| im0      | im1      | im2   | 2048  | 14427 | 7.04       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 147   | 18.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 2048  | 14427 | 7.04       |

#### 6.204.2.4 nmppsDotProd\_8s64s()

```
void nmppsDotProd_8s64s (
    const nm8s * pSrcVec1,
    const nm64s * pSrcVec2,
    int nSize,
    int64b * pnRes )
```

Performance table 0

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im0      | im2   | 2048  | 2399  | 1.17       |
| im0      | im0      | im2   | 2048  | 2399  | 1.17       |
| im0      | im1      | im2   | 2048  | 2671  | 1.30       |
| im0      | im1      | im2   | 2048  | 2671  | 1.30       |
| im1      | im0      | im2   | 2048  | 2145  | 1.04       |
| im1      | im0      | im2   | 2048  | 2145  | 1.04       |
| im1      | im1      | im2   | 2048  | 2928  | 1.42       |
| im1      | im1      | im2   | 2048  | 2928  | 1.42       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im1      | im0      | im2   | 2048  | 2145  | 1.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im1      | im1      | im2   | 2048  | 2928  | 1.42       |



Perfomance table 1

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 121   | 15.12      |
| im0      | im1      | im2   | 8     | 121   | 15.12      |
| im0      | im1      | im2   | 128   | 271   | 2.11       |
| im0      | im1      | im2   | 128   | 271   | 2.11       |
| im0      | im1      | im2   | 1024  | 1391  | 1.35       |
| im0      | im1      | im2   | 1024  | 1391  | 1.35       |
| im0      | im1      | im2   | 2048  | 2671  | 1.30       |
| im0      | im1      | im2   | 2048  | 2671  | 1.30       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 8     | 121   | 15.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pnRes | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|-------|------------|
| im0      | im1      | im2   | 2048  | 2671  | 1.30       |

## 6.205 nmppsWeightedSum

Поэлементное взвешенное суммирование элементов двух векторов

Функции

- void `nmppsWeightedSum_8s16s` (`nm8s` \*pSrcVec1, int nW1, `nm8s` \*pSrcVec2, int nW2, `nm16s` \*pDstVec, int nSize)
- void `nmppsWeightedSum_16s32s` (`nm16s` \*pSrcVec1, int nW1, `nm16s` \*pSrcVec2, int nW2, `nm32s` \*pDstVec, int nSize)
- void `nmppsWeightedSum_32s64s` (`nm32s` \*pSrcVec1, `nm64s` nW1, `nm32s` \*pSrcVec2, `nm64s` nW2, `nm64s` \*pDstVec, int nSize)

### 6.205.1 Подробное описание

Поэлементное взвешенное суммирование элементов двух векторов

$$pDstVec[i] = nW1 \cdot pSrcVec1[i] + nW2 \cdot pSrcVec2[i],$$

$$= \overline{0 \dots nSize - 1}$$

\param pSrcVec1

1-ый входной вектор.

\param nW1

1-ый весовой коэффициент

\param pSrcVec2

2-ой входной вектор.

\param nW2

2-ой весовой коэффициент

\param nSize

Размер векторов в элементах.

\retval pDstVec

Результирующий вектор.

Возвращает

void

## 6.205.2 Функции

## 6.205.2.1 nmppsWeightedSum\_16s32s()

```
void nmppsWeightedSum_16s32s (
    nm16s * pSrcVec1,
    int nW1,
    nm16s * pSrcVec2,
    int nW2,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im1     | 0   | 0   | 8     | 317   | 39.62      |
| im0      | im0      | im1     | 0   | 0   | 8     | 317   | 39.62      |
| im0      | im0      | im1     | 0   | 0   | 128   | 487   | 3.80       |
| im0      | im0      | im1     | 0   | 0   | 128   | 487   | 3.80       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 1425  | 1.39       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 1425  | 1.39       |
| im0      | im0      | im1     | 0   | 0   | 2048  | 2497  | 1.21       |
| im0      | im0      | im1     | 0   | 0   | 2048  | 2497  | 1.21       |
| im0      | im0      | im2     | 0   | 0   | 8     | 313   | 39.12      |
| im0      | im0      | im2     | 0   | 0   | 8     | 313   | 39.12      |
| im0      | im0      | im2     | 0   | 0   | 128   | 400   | 3.12       |
| im0      | im0      | im2     | 0   | 0   | 128   | 400   | 3.12       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 1338  | 1.30       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 1338  | 1.30       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 2410  | 1.17       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 2410  | 1.17       |
| im0      | im1      | im1     | 0   | 0   | 8     | 322   | 40.25      |
| im0      | im1      | im1     | 0   | 0   | 8     | 322   | 40.25      |
| im0      | im1      | im1     | 0   | 0   | 128   | 489   | 3.82       |
| im0      | im1      | im1     | 0   | 0   | 128   | 489   | 3.82       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 1565  | 1.52       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 1565  | 1.52       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 2781  | 1.35       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 2781  | 1.35       |
| im0      | im1      | im2     | 0   | 0   | 8     | 318   | 39.75      |
| im0      | im1      | im2     | 0   | 0   | 8     | 318   | 39.75      |
| im0      | im1      | im2     | 0   | 0   | 128   | 427   | 3.33       |
| im0      | im1      | im2     | 0   | 0   | 128   | 427   | 3.33       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 1413  | 1.37       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 1413  | 1.37       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 2549  | 1.24       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 2549  | 1.24       |
| im1      | im0      | im1     | 0   | 0   | 8     | 322   | 40.25      |
| im1      | im0      | im1     | 0   | 0   | 8     | 322   | 40.25      |
| im1      | im0      | im1     | 0   | 0   | 128   | 495   | 3.86       |
| im1      | im0      | im1     | 0   | 0   | 128   | 495   | 3.86       |

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im0      | im1     | 0   | 0   | 1024  | 1873  | 1.82       |
| im1      | im0      | im1     | 0   | 0   | 1024  | 1873  | 1.82       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 3425  | 1.67       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 3425  | 1.67       |
| im1      | im0      | im2     | 0   | 0   | 8     | 318   | 39.75      |
| im1      | im0      | im2     | 0   | 0   | 8     | 318   | 39.75      |
| im1      | im0      | im2     | 0   | 0   | 128   | 408   | 3.18       |
| im1      | im0      | im2     | 0   | 0   | 128   | 408   | 3.18       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 1354  | 1.32       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 1354  | 1.32       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 2426  | 1.18       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 2426  | 1.18       |
| im1      | im1      | im1     | 0   | 0   | 8     | 324   | 40.50      |
| im1      | im1      | im1     | 0   | 0   | 8     | 324   | 40.50      |
| im1      | im1      | im1     | 0   | 0   | 128   | 509   | 3.97       |
| im1      | im1      | im1     | 0   | 0   | 128   | 509   | 3.97       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 2037  | 1.98       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 2037  | 1.98       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 3765  | 1.83       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 3765  | 1.83       |
| im1      | im1      | im2     | 0   | 0   | 8     | 320   | 40.00      |
| im1      | im1      | im2     | 0   | 0   | 8     | 320   | 40.00      |
| im1      | im1      | im2     | 0   | 0   | 128   | 449   | 3.50       |
| im1      | im1      | im2     | 0   | 0   | 128   | 449   | 3.50       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 1537  | 1.50       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 1537  | 1.50       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 2785  | 1.35       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 2785  | 1.35       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im2     | 0   | 0   | 8     | 313   | 39.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im1      | im1     | 0   | 0   | 2048  | 3765  | 1.83       |

#### 6.205.2.2 nmppsWeightedSum\_32s64s()

```
void nmppsWeightedSum_32s64s (
    nm32s * pSrcVec1,
    nm64s nW1,
    nm32s * pSrcVec2,
    nm64s nW2,
```

```
nm64s * pDstVec,
int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im1     | 0   | 0   | 8     | 342   | 42.75      |
| im0      | im0      | im1     | 0   | 0   | 8     | 342   | 42.75      |
| im0      | im0      | im1     | 0   | 0   | 128   | 650   | 5.07       |
| im0      | im0      | im1     | 0   | 0   | 128   | 650   | 5.07       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 2526  | 2.46       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 2526  | 2.46       |
| im0      | im0      | im1     | 0   | 0   | 2048  | 4670  | 2.28       |
| im0      | im0      | im1     | 0   | 0   | 2048  | 4670  | 2.28       |
| im0      | im0      | im2     | 0   | 0   | 8     | 332   | 41.50      |
| im0      | im0      | im2     | 0   | 0   | 8     | 332   | 41.50      |
| im0      | im0      | im2     | 0   | 0   | 128   | 538   | 4.20       |
| im0      | im0      | im2     | 0   | 0   | 128   | 538   | 4.20       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 2414  | 2.35       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 2414  | 2.35       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 4558  | 2.22       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 4558  | 2.22       |
| im0      | im1      | im1     | 0   | 0   | 8     | 352   | 44.00      |
| im0      | im1      | im1     | 0   | 0   | 8     | 352   | 44.00      |
| im0      | im1      | im1     | 0   | 0   | 128   | 666   | 5.20       |
| im0      | im1      | im1     | 0   | 0   | 128   | 666   | 5.20       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 2678  | 2.61       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 2678  | 2.61       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 4982  | 2.43       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 4982  | 2.43       |
| im0      | im1      | im2     | 0   | 0   | 8     | 345   | 43.12      |
| im0      | im1      | im2     | 0   | 0   | 8     | 345   | 43.12      |
| im0      | im1      | im2     | 0   | 0   | 128   | 588   | 4.59       |
| im0      | im1      | im2     | 0   | 0   | 128   | 588   | 4.59       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 2542  | 2.48       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 2542  | 2.48       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 4782  | 2.33       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 4782  | 2.33       |
| im1      | im0      | im1     | 0   | 0   | 8     | 349   | 43.62      |
| im1      | im0      | im1     | 0   | 0   | 8     | 349   | 43.62      |
| im1      | im0      | im1     | 0   | 0   | 128   | 728   | 5.68       |
| im1      | im0      | im1     | 0   | 0   | 128   | 728   | 5.68       |
| im1      | im0      | im1     | 0   | 0   | 1024  | 3446  | 3.36       |
| im1      | im0      | im1     | 0   | 0   | 1024  | 3446  | 3.36       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 6550  | 3.19       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 6550  | 3.19       |
| im1      | im0      | im2     | 0   | 0   | 8     | 337   | 42.12      |
| im1      | im0      | im2     | 0   | 0   | 8     | 337   | 42.12      |
| im1      | im0      | im2     | 0   | 0   | 128   | 546   | 4.26       |
| im1      | im0      | im2     | 0   | 0   | 128   | 546   | 4.26       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 2422  | 2.36       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 2422  | 2.36       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 4566  | 2.22       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 4566  | 2.22       |

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im1      | im1     | 0   | 0   | 8     | 352   | 44.00      |
| im1      | im1      | im1     | 0   | 0   | 8     | 352   | 44.00      |
| im1      | im1      | im1     | 0   | 0   | 128   | 748   | 5.84       |
| im1      | im1      | im1     | 0   | 0   | 128   | 748   | 5.84       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 3634  | 3.54       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 3634  | 3.54       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 6930  | 3.38       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 6930  | 3.38       |
| im1      | im1      | im2     | 0   | 0   | 8     | 343   | 42.87      |
| im1      | im1      | im2     | 0   | 0   | 8     | 343   | 42.87      |
| im1      | im1      | im2     | 0   | 0   | 128   | 614   | 4.79       |
| im1      | im1      | im2     | 0   | 0   | 128   | 614   | 4.79       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 2682  | 2.61       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 2682  | 2.61       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 5050  | 2.46       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 5050  | 2.46       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im2     | 0   | 0   | 8     | 332   | 41.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im1      | im1     | 0   | 0   | 2048  | 6930  | 3.38       |

#### 6.205.2.3 nmppsWeightedSum\_8s16s()

```
void nmppsWeightedSum_8s16s (
    nm8s * pSrcVec1,
    int nW1,
    nm8s * pSrcVec2,
    int nW2,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im1     | 0   | 0   | 8     | 372   | 46.50      |
| im0      | im0      | im1     | 0   | 0   | 8     | 372   | 46.50      |
| im0      | im0      | im1     | 0   | 0   | 128   | 453   | 3.53       |
| im0      | im0      | im1     | 0   | 0   | 128   | 453   | 3.53       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 942   | 0.91       |
| im0      | im0      | im1     | 0   | 0   | 1024  | 942   | 0.91       |

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im1     | 0   | 0   | 2048  | 1462  | 0.71       |
| im0      | im0      | im1     | 0   | 0   | 2048  | 1462  | 0.71       |
| im0      | im0      | im2     | 0   | 0   | 8     | 368   | 46.00      |
| im0      | im0      | im2     | 0   | 0   | 8     | 368   | 46.00      |
| im0      | im0      | im2     | 0   | 0   | 128   | 400   | 3.12       |
| im0      | im0      | im2     | 0   | 0   | 128   | 400   | 3.12       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 850   | 0.83       |
| im0      | im0      | im2     | 0   | 0   | 1024  | 850   | 0.83       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 1370  | 0.66       |
| im0      | im0      | im2     | 0   | 0   | 2048  | 1370  | 0.66       |
| im0      | im1      | im1     | 0   | 0   | 8     | 373   | 46.62      |
| im0      | im1      | im1     | 0   | 0   | 8     | 373   | 46.62      |
| im0      | im1      | im1     | 0   | 0   | 128   | 456   | 3.56       |
| im0      | im1      | im1     | 0   | 0   | 128   | 456   | 3.56       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 1094  | 1.06       |
| im0      | im1      | im1     | 0   | 0   | 1024  | 1094  | 1.06       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 1760  | 0.85       |
| im0      | im1      | im1     | 0   | 0   | 2048  | 1760  | 0.85       |
| im0      | im1      | im2     | 0   | 0   | 8     | 370   | 46.25      |
| im0      | im1      | im2     | 0   | 0   | 8     | 370   | 46.25      |
| im0      | im1      | im2     | 0   | 0   | 128   | 423   | 3.30       |
| im0      | im1      | im2     | 0   | 0   | 128   | 423   | 3.30       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 966   | 0.94       |
| im0      | im1      | im2     | 0   | 0   | 1024  | 966   | 0.94       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 1566  | 0.76       |
| im0      | im1      | im2     | 0   | 0   | 2048  | 1566  | 0.76       |
| im1      | im0      | im1     | 0   | 0   | 8     | 373   | 46.62      |
| im1      | im0      | im1     | 0   | 0   | 8     | 373   | 46.62      |
| im1      | im0      | im1     | 0   | 0   | 128   | 461   | 3.60       |
| im1      | im0      | im1     | 0   | 0   | 128   | 461   | 3.60       |
| im1      | im0      | im1     | 0   | 0   | 1024  | 1142  | 1.11       |
| im1      | im0      | im1     | 0   | 0   | 1024  | 1142  | 1.11       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 1902  | 0.92       |
| im1      | im0      | im1     | 0   | 0   | 2048  | 1902  | 0.92       |
| im1      | im0      | im2     | 0   | 0   | 8     | 370   | 46.25      |
| im1      | im0      | im2     | 0   | 0   | 8     | 370   | 46.25      |
| im1      | im0      | im2     | 0   | 0   | 128   | 413   | 3.22       |
| im1      | im0      | im2     | 0   | 0   | 128   | 413   | 3.22       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 858   | 0.83       |
| im1      | im0      | im2     | 0   | 0   | 1024  | 858   | 0.83       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 1378  | 0.67       |
| im1      | im0      | im2     | 0   | 0   | 2048  | 1378  | 0.67       |
| im1      | im1      | im1     | 0   | 0   | 8     | 376   | 47.00      |
| im1      | im1      | im1     | 0   | 0   | 8     | 376   | 47.00      |
| im1      | im1      | im1     | 0   | 0   | 128   | 468   | 3.65       |
| im1      | im1      | im1     | 0   | 0   | 128   | 468   | 3.65       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 1296  | 1.26       |
| im1      | im1      | im1     | 0   | 0   | 1024  | 1296  | 1.26       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 2216  | 1.08       |
| im1      | im1      | im1     | 0   | 0   | 2048  | 2216  | 1.08       |
| im1      | im1      | im2     | 0   | 0   | 8     | 371   | 46.37      |
| im1      | im1      | im2     | 0   | 0   | 8     | 371   | 46.37      |

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im1      | im2     | 0   | 0   | 128   | 435   | 3.39       |
| im1      | im1      | im2     | 0   | 0   | 128   | 435   | 3.39       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 1026  | 1.00       |
| im1      | im1      | im2     | 0   | 0   | 1024  | 1026  | 1.00       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 1690  | 0.82       |
| im1      | im1      | im2     | 0   | 0   | 2048  | 1690  | 0.82       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im0      | im0      | im2     | 0   | 0   | 8     | 368   | 46.00      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nW1 | nW2 | nSize | ticks | ticks/elem |
|----------|----------|---------|-----|-----|-------|-------|------------|
| im1      | im1      | im1     | 0   | 0   | 2048  | 2216  | 1.08       |



## 6.206 nmppsNot\_

Функция логического "НЕ".

Функции

- void nmppsNot\_2u (const nm2u \*pSrcVec, nm2u \*pDstVec, int nSize)
- void nmppsNot\_4u (const nm4u \*pSrcVec, nm4u \*pDstVec, int nSize)
- void nmppsNot\_8u (const nm8u \*pSrcVec, nm8u \*pDstVec, int nSize)
- void nmppsNot\_16u (const nm16u \*pSrcVec, nm16u \*pDstVec, int nSize)
- void nmppsNot\_32u (const nm32u \*pSrcVec, nm32u \*pDstVec, int nSize)
- void nmppsNot\_64u (const nm64u \*pSrcVec, nm64u \*pDstVec, int nSize)

## 6.206.1 Подробное описание

Функция логического "НЕ".

$$pDstVec[i] = \overline{pSrcVec[i]},$$

$$i = \overline{0 \dots nSize - 1}$$

Функция изменяет значения всех битов входного вектора на противоположные.

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.206.2 Функции

## 6.206.2.1 nmppsNot\_16u()

```
void nmppsNot_16u (
    const nm16u * pSrcVec,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1063  | 0.51       |
| im0     | im0     | 2048  | 1063  | 0.51       |
| im0     | im1     | 2048  | 652   | 0.31       |
| im0     | im1     | 2048  | 652   | 0.31       |
| im1     | im0     | 2048  | 650   | 0.31       |
| im1     | im0     | 2048  | 650   | 0.31       |
| im1     | im1     | 2048  | 1162  | 0.56       |
| im1     | im1     | 2048  | 1162  | 0.56       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 650   | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1162  | 0.56       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 128   | 143   | 1.11       |
| im0     | im1     | 128   | 143   | 1.11       |
| im0     | im1     | 1024  | 382   | 0.37       |
| im0     | im1     | 1024  | 382   | 0.37       |
| im0     | im1     | 2048  | 652   | 0.31       |
| im0     | im1     | 2048  | 652   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 652   | 0.31       |

## 6.206.2.2 nmppsNot\_2u()

```
void nmppsNot_2u (
    const nm2u * pSrcVec,
    nm2u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 169   | 0.08       |
| im0     | im0     | 2048  | 169   | 0.08       |
| im0     | im1     | 2048  | 178   | 0.08       |
| im0     | im1     | 2048  | 178   | 0.08       |
| im1     | im0     | 2048  | 174   | 0.08       |
| im1     | im0     | 2048  | 174   | 0.08       |
| im1     | im1     | 2048  | 238   | 0.11       |
| im1     | im1     | 2048  | 238   | 0.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 169   | 0.08       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 238   | 0.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 107   | 13.37      |
| im0     | im1     | 8     | 107   | 13.37      |
| im0     | im1     | 128   | 110   | 0.85       |
| im0     | im1     | 128   | 110   | 0.85       |
| im0     | im1     | 1024  | 143   | 0.13       |
| im0     | im1     | 1024  | 143   | 0.13       |
| im0     | im1     | 2048  | 178   | 0.08       |
| im0     | im1     | 2048  | 178   | 0.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 107   | 13.37      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 178   | 0.08       |

### 6.206.2.3 nmppsNot\_32u()

```
void nmppsNot_32u (
    const nm32u * pSrcVec,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2087  | 1.01       |
| im0     | im0     | 2048  | 2087  | 1.01       |
| im0     | im1     | 2048  | 1196  | 0.58       |
| im0     | im1     | 2048  | 1196  | 0.58       |
| im1     | im0     | 2048  | 1194  | 0.58       |
| im1     | im0     | 2048  | 1194  | 0.58       |
| im1     | im1     | 2048  | 2218  | 1.08       |
| im1     | im1     | 2048  | 2218  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1194  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2218  | 1.08       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 110   | 13.75      |
| im0     | im1     | 8     | 110   | 13.75      |
| im0     | im1     | 128   | 178   | 1.39       |
| im0     | im1     | 128   | 178   | 1.39       |
| im0     | im1     | 1024  | 652   | 0.63       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 1024  | 652   | 0.63       |
| im0     | im1     | 2048  | 1196  | 0.58       |
| im0     | im1     | 2048  | 1196  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 110   | 13.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1196  | 0.58       |

#### 6.206.2.4 nmppsNot\_4u()

```
void nmppsNot_4u (
    const nm4u * pSrcVec,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4135  | 2.01       |
| im0     | im0     | 2048  | 4135  | 2.01       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im1     | im0     | 2048  | 2282  | 1.11       |
| im1     | im0     | 2048  | 2282  | 1.11       |
| im1     | im1     | 2048  | 4330  | 2.11       |
| im1     | im1     | 2048  | 4330  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2282  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4330  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 128   | 245   | 1.91       |
| im0     | im1     | 128   | 245   | 1.91       |
| im0     | im1     | 1024  | 1196  | 1.16       |
| im0     | im1     | 1024  | 1196  | 1.16       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im0     | im1     | 2048  | 2284  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2284  | 1.11       |

#### 6.206.2.5 nmppsNot\_64u()

```
void nmppsNot_64u (
    const nm64u * pSrcVec,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4131  | 2.01       |
| im0     | im0     | 2048  | 4131  | 2.01       |
| im0     | im1     | 2048  | 2280  | 1.11       |
| im0     | im1     | 2048  | 2280  | 1.11       |
| im1     | im0     | 2048  | 2278  | 1.11       |
| im1     | im0     | 2048  | 2278  | 1.11       |
| im1     | im1     | 2048  | 4326  | 2.11       |
| im1     | im1     | 2048  | 4326  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2278  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4326  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 115   | 14.37      |
| im0     | im1     | 8     | 115   | 14.37      |
| im0     | im1     | 128   | 241   | 1.88       |
| im0     | im1     | 128   | 241   | 1.88       |
| im0     | im1     | 1024  | 1192  | 1.16       |
| im0     | im1     | 1024  | 1192  | 1.16       |
| im0     | im1     | 2048  | 2280  | 1.11       |
| im0     | im1     | 2048  | 2280  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 115   | 14.37      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2280  | 1.11       |

#### 6.206.2.6 nmppsNot\_8u()

```
void nmppsNot_8u (
    const nm8u * pSrcVec,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4135  | 2.01       |
| im0     | im0     | 2048  | 4135  | 2.01       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im1     | im0     | 2048  | 2282  | 1.11       |
| im1     | im0     | 2048  | 2282  | 1.11       |
| im1     | im1     | 2048  | 4330  | 2.11       |
| im1     | im1     | 2048  | 4330  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2282  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4330  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 8     | 117   | 14.62      |
| im0     | im1     | 128   | 245   | 1.91       |
| im0     | im1     | 128   | 245   | 1.91       |
| im0     | im1     | 1024  | 1196  | 1.16       |
| im0     | im1     | 1024  | 1196  | 1.16       |
| im0     | im1     | 2048  | 2284  | 1.11       |
| im0     | im1     | 2048  | 2284  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2284  | 1.11       |



## 6.207 nmppsAndC

Функция логического "И" между вектором и константой.

Функции

- void nmppsAndC\_4u (const nm4u \*pSrcVec, uint4b nVal, nm4u \*pDstVec, int nSize)
- void nmppsAndC\_8u (const nm8u \*pSrcVec, uint8b nVal, nm8u \*pDstVec, int nSize)
- void nmppsAndC\_16u (const nm16u \*pSrcVec, uint16b nVal, nm16u \*pDstVec, int nSize)
- void nmppsAndC\_32u (const nm32u \*pSrcVec, uint32b nVal, nm32u \*pDstVec, int nSize)
- void nmppsAndC\_64u (const nm64u \*pSrcVec, uint64b nVal, nm64u \*pDstVec, int nSize)

### 6.207.1 Подробное описание

Функция логического "И" между вектором и константой.

$$pDstVec[i] = pSrcVec[i] \wedge nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа.                   |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.208 nmppsAnd

Функция логического "И" между двумя векторами.

Функции

- void `nmppsAnd_1` (const `nm1` \*pSrcVec1, const `nm1` \*pSrcVec2, `nm1` \*pDstVec, int nSize)
- void `nmppsAnd_2u` (const `nm2u` \*pSrcVec1, const `nm2u` \*pSrcVec2, `nm2u` \*pDstVec, int nSize)
- void `nmppsAnd_4u` (const `nm4u` \*pSrcVec1, const `nm4u` \*pSrcVec2, `nm4u` \*pDstVec, int nSize)
- void `nmppsAnd_8u` (const `nm8u` \*pSrcVec1, const `nm8u` \*pSrcVec2, `nm8u` \*pDstVec, int nSize)
- void `nmppsAnd_16u` (const `nm16u` \*pSrcVec1, const `nm16u` \*pSrcVec2, `nm16u` \*pDstVec, int nSize)
- void `nmppsAnd_32u` (const `nm32u` \*pSrcVec1, const `nm32u` \*pSrcVec2, `nm32u` \*pDstVec, int nSize)
- void `nmppsAnd_64u` (const `nm64u` \*pSrcVec1, const `nm64u` \*pSrcVec2, `nm64u` \*pDstVec, int nSize)

### 6.208.1 Подробное описание

Функция логического "И" между двумя векторами.

$$pDstVec[i] = pSrcVec1[i] \wedge pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.208.2 Функции

## 6.208.2.1 nmppsAnd\_1()

```
void nmppsAnd_1 (
    const nm1 * pSrcVec1,
    const nm1 * pSrcVec2,
    nm1 * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 115   | 0.05       |
| im0      | im0      | im2     | 2048  | 115   | 0.05       |
| im0      | im0      | im3     | 2048  | 115   | 0.05       |
| im0      | im0      | im3     | 2048  | 115   | 0.05       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im0      | im1      | im3     | 2048  | 145   | 0.07       |
| im0      | im1      | im3     | 2048  | 145   | 0.07       |
| im1      | im0      | im2     | 2048  | 142   | 0.06       |
| im1      | im0      | im2     | 2048  | 142   | 0.06       |
| im1      | im0      | im3     | 2048  | 142   | 0.06       |
| im1      | im0      | im3     | 2048  | 142   | 0.06       |
| im1      | im1      | im2     | 2048  | 174   | 0.08       |
| im1      | im1      | im2     | 2048  | 174   | 0.08       |
| im1      | im1      | im3     | 2048  | 174   | 0.08       |
| im1      | im1      | im3     | 2048  | 174   | 0.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 115   | 0.05       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 174   | 0.08       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 116   | 0.90       |
| im0      | im1      | im2     | 128   | 116   | 0.90       |
| im0      | im1      | im2     | 1024  | 133   | 0.12       |
| im0      | im1      | im2     | 1024  | 133   | 0.12       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 145   | 0.07       |

#### 6.208.2.2 nmppsAnd\_16u()

```
void nmppsAnd_16u (
    const nm16u * pSrcVec1,
    const nm16u * pSrcVec2,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1075  | 0.52       |
| im0      | im0      | im2     | 2048  | 1075  | 0.52       |
| im0      | im0      | im3     | 2048  | 1075  | 0.52       |
| im0      | im0      | im3     | 2048  | 1075  | 0.52       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im0      | im1      | im3     | 2048  | 671   | 0.32       |
| im0      | im1      | im3     | 2048  | 671   | 0.32       |
| im1      | im0      | im2     | 2048  | 670   | 0.32       |
| im1      | im0      | im2     | 2048  | 670   | 0.32       |
| im1      | im0      | im3     | 2048  | 670   | 0.32       |
| im1      | im0      | im3     | 2048  | 670   | 0.32       |
| im1      | im1      | im2     | 2048  | 1183  | 0.57       |
| im1      | im1      | im2     | 2048  | 1183  | 0.57       |
| im1      | im1      | im3     | 2048  | 1183  | 0.57       |
| im1      | im1      | im3     | 2048  | 1183  | 0.57       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 670   | 0.32       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 1183  | 0.57       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 116   | 14.50      |
| im0      | im1      | im2     | 8     | 116   | 14.50      |
| im0      | im1      | im2     | 128   | 145   | 1.13       |
| im0      | im1      | im2     | 128   | 145   | 1.13       |
| im0      | im1      | im2     | 1024  | 391   | 0.38       |
| im0      | im1      | im2     | 1024  | 391   | 0.38       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 116   | 14.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 671   | 0.32       |

### 6.208.2.3 nmppsAnd\_2u()

```
void nmppsAnd_2u (
    const nm2u * pSrcVec1,
    const nm2u * pSrcVec2,
    nm2u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 179   | 0.08       |
| im0      | im0      | im2     | 2048  | 179   | 0.08       |
| im0      | im0      | im3     | 2048  | 179   | 0.08       |
| im0      | im0      | im3     | 2048  | 179   | 0.08       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im0      | im1      | im3     | 2048  | 180   | 0.08       |
| im0      | im1      | im3     | 2048  | 180   | 0.08       |
| im1      | im0      | im2     | 2048  | 179   | 0.08       |
| im1      | im0      | im2     | 2048  | 179   | 0.08       |
| im1      | im0      | im3     | 2048  | 179   | 0.08       |
| im1      | im0      | im3     | 2048  | 179   | 0.08       |
| im1      | im1      | im2     | 2048  | 244   | 0.11       |
| im1      | im1      | im2     | 2048  | 244   | 0.11       |
| im1      | im1      | im3     | 2048  | 244   | 0.11       |
| im1      | im1      | im3     | 2048  | 244   | 0.11       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 179   | 0.08       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 244   | 0.11       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 119   | 0.92       |
| im0      | im1      | im2     | 128   | 119   | 0.92       |
| im0      | im1      | im2     | 1024  | 145   | 0.14       |
| im0      | im1      | im2     | 1024  | 145   | 0.14       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 180   | 0.08       |

#### 6.208.2.4 nmppsAnd\_32u()

```
void nmppsAnd_32u (
    const nm32u * pSrcVec1,
    const nm32u * pSrcVec2,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2099  | 1.02       |
| im0      | im0      | im2     | 2048  | 2099  | 1.02       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im3     | 2048  | 2099  | 1.02       |
| im0      | im0      | im3     | 2048  | 2099  | 1.02       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im0      | im1      | im3     | 2048  | 1231  | 0.60       |
| im0      | im1      | im3     | 2048  | 1231  | 0.60       |
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |
| im1      | im0      | im3     | 2048  | 1230  | 0.60       |
| im1      | im0      | im3     | 2048  | 1230  | 0.60       |
| im1      | im1      | im2     | 2048  | 2255  | 1.10       |
| im1      | im1      | im2     | 2048  | 2255  | 1.10       |
| im1      | im1      | im3     | 2048  | 2255  | 1.10       |
| im1      | im1      | im3     | 2048  | 2255  | 1.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 2255  | 1.10       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 128   | 180   | 1.40       |
| im0      | im1      | im2     | 128   | 180   | 1.40       |
| im0      | im1      | im2     | 1024  | 671   | 0.65       |
| im0      | im1      | im2     | 1024  | 671   | 0.65       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |

## 6.208.2.5 nmppsAnd\_4u()

```
void nmppsAnd_4u (
    const nm4u * pSrcVec1,
    const nm4u * pSrcVec2,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 307   | 0.14       |
| im0      | im0      | im2     | 2048  | 307   | 0.14       |
| im0      | im0      | im3     | 2048  | 307   | 0.14       |
| im0      | im0      | im3     | 2048  | 307   | 0.14       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |
| im0      | im1      | im3     | 2048  | 251   | 0.12       |
| im0      | im1      | im3     | 2048  | 251   | 0.12       |
| im1      | im0      | im2     | 2048  | 250   | 0.12       |
| im1      | im0      | im2     | 2048  | 250   | 0.12       |
| im1      | im0      | im3     | 2048  | 250   | 0.12       |
| im1      | im0      | im3     | 2048  | 250   | 0.12       |
| im1      | im1      | im2     | 2048  | 379   | 0.18       |
| im1      | im1      | im2     | 2048  | 379   | 0.18       |
| im1      | im1      | im3     | 2048  | 379   | 0.18       |
| im1      | im1      | im3     | 2048  | 379   | 0.18       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 250   | 0.12       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 379   | 0.18       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 125   | 0.97       |
| im0      | im1      | im2     | 128   | 125   | 0.97       |
| im0      | im1      | im2     | 1024  | 180   | 0.17       |
| im0      | im1      | im2     | 1024  | 180   | 0.17       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |



| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 251   | 0.12       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 251   | 0.12       |

### 6.208.2.6 nmppsAnd\_64u()

```
void nmppsAnd_64u (
    const nm64u * pSrcVec1,
    const nm64u * pSrcVec2,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4146  | 2.02       |
| im0      | im0      | im2     | 2048  | 4146  | 2.02       |
| im0      | im0      | im3     | 2048  | 4146  | 2.02       |
| im0      | im0      | im3     | 2048  | 4146  | 2.02       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im0      | im1      | im3     | 2048  | 2350  | 1.14       |
| im0      | im1      | im3     | 2048  | 2350  | 1.14       |
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |
| im1      | im0      | im3     | 2048  | 2349  | 1.14       |
| im1      | im0      | im3     | 2048  | 2349  | 1.14       |
| im1      | im1      | im2     | 2048  | 4398  | 2.14       |
| im1      | im1      | im2     | 2048  | 4398  | 2.14       |
| im1      | im1      | im3     | 2048  | 4398  | 2.14       |
| im1      | im1      | im3     | 2048  | 4398  | 2.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 4398  | 2.14       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 124   | 15.50      |
| im0      | im1      | im2     | 8     | 124   | 15.50      |
| im0      | im1      | im2     | 128   | 250   | 1.95       |
| im0      | im1      | im2     | 128   | 250   | 1.95       |
| im0      | im1      | im2     | 1024  | 1230  | 1.20       |
| im0      | im1      | im2     | 1024  | 1230  | 1.20       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 124   | 15.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |

#### 6.208.2.7 nmppsAnd\_8u()

```
void nmppsAnd_8u (
    const nm8u * pSrcVec1,
    const nm8u * pSrcVec2,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 563   | 0.27       |
| im0      | im0      | im2     | 2048  | 563   | 0.27       |
| im0      | im0      | im3     | 2048  | 563   | 0.27       |
| im0      | im0      | im3     | 2048  | 563   | 0.27       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |
| im0      | im1      | im3     | 2048  | 391   | 0.19       |
| im0      | im1      | im3     | 2048  | 391   | 0.19       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 390   | 0.19       |
| im1      | im0      | im2     | 2048  | 390   | 0.19       |
| im1      | im0      | im3     | 2048  | 390   | 0.19       |
| im1      | im0      | im3     | 2048  | 390   | 0.19       |
| im1      | im1      | im2     | 2048  | 647   | 0.31       |
| im1      | im1      | im2     | 2048  | 647   | 0.31       |
| im1      | im1      | im3     | 2048  | 647   | 0.31       |
| im1      | im1      | im3     | 2048  | 647   | 0.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 390   | 0.19       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im2     | 2048  | 647   | 0.31       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 133   | 1.03       |
| im0      | im1      | im2     | 128   | 133   | 1.03       |
| im0      | im1      | im2     | 1024  | 251   | 0.24       |
| im0      | im1      | im2     | 1024  | 251   | 0.24       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 391   | 0.19       |

## 6.209 nmppsAnd4V\_

Функция логического "И" между четырьмя векторами.

Функции

- void nmppsAnd4V\_64u (nm64u \*pSrcVec1, nm64u \*pSrcVec2, nm64u \*pSrcVec3, nm64u \*pSrcVec4, nm64u \*pDstVec, int nSize)

### 6.209.1 Подробное описание

Функция логического "И" между четырьмя векторами.

$$pDstVec[i] = pSrcVec1[i] \wedge pSrcVec2[i] \wedge pSrcVec3[i] \wedge pSrcVec4[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| pSrcVec3 | Третий входной вектор.       |
| pSrcVec4 | Четвертый входной вектор.    |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.210 nmppsAndNotV\_

Функция логического "И-НЕ" между двумя векторами.

Функции

- void `nmppsAndNotV_64u` (`nm64u` \*pSrcVec1, `nm64u` \*pSrcVec2, `nm64u` \*pDstVec, int nSize)

### 6.210.1 Подробное описание

Функция логического "И-НЕ" между двумя векторами.

$$pDstVec[i] = pSrcVec1[i] \wedge not pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.210.2 Функции

#### 6.210.2.1 nmppsAndNotV\_64u()

```
void nmppsAndNotV_64u (
    nm64u * pSrcVec1,
    nm64u * pSrcVec2,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |
| im0      | im1      | im0     | 2048  | 6159  | 3.00       |
| im0      | im1      | im0     | 2048  | 6159  | 3.00       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im1      | im0      | im0     | 2048  | 4240  | 2.07       |
| im1      | im0      | im0     | 2048  | 4240  | 2.07       |
| im1      | im0      | im2     | 2048  | 4240  | 2.07       |
| im1      | im0      | im2     | 2048  | 4240  | 2.07       |
| im1      | im1      | im0     | 2048  | 4401  | 2.14       |
| im1      | im1      | im0     | 2048  | 4401  | 2.14       |
| im1      | im1      | im2     | 2048  | 4401  | 2.14       |
| im1      | im1      | im2     | 2048  | 4401  | 2.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 129   | 16.12      |
| im0      | im1      | im2     | 8     | 129   | 16.12      |
| im0      | im1      | im2     | 128   | 375   | 2.92       |
| im0      | im1      | im2     | 128   | 375   | 2.92       |
| im0      | im1      | im2     | 1024  | 2335  | 2.28       |
| im0      | im1      | im2     | 1024  | 2335  | 2.28       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 129   | 16.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |

## 6.211 nmppsOrC

Функция логического "ИЛИ" между вектором и константой.

Функции

- void nmppsOrC\_8u (const nm8u \*pSrcVec, uint8b nVal, nm8u \*pDstVec, int nSize)
- void nmppsOrC\_16u (const nm16u \*pSrcVec, uint16b nVal, nm16u \*pDstVec, int nSize)
- void nmppsOrC\_32u (const nm32u \*pSrcVec, uint32b nVal, nm32u \*pDstVec, int nSize)
- void nmppsOrC\_64u (const nm64u \*pSrcVec, uint64b nVal, nm64u \*pDstVec, int nSize)

### 6.211.1 Подробное описание

Функция логического "ИЛИ" между вектором и константой.

$$pDstVec[i] = pSrcVec[i] \vee nVal$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа.                   |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void



## 6.212 nmppsOr

Функция логического "ИЛИ" между двумя векторами.

Функции

- void nmppsOr\_1 (const nm1 \*pSrcVec1, const nm1 \*pSrcVec2, nm1 \*pDstVec, int nSize)
- void nmppsOr\_2u (const nm2u \*pSrcVec1, const nm2u \*pSrcVec2, nm2u \*pDstVec, int nSize)
- void nmppsOr\_4u (const nm4u \*pSrcVec1, const nm4u \*pSrcVec2, nm4u \*pDstVec, int nSize)
- void nmppsOr\_8u (const nm8u \*pSrcVec1, const nm8u \*pSrcVec2, nm8u \*pDstVec, int nSize)
- void nmppsOr\_16u (const nm16u \*pSrcVec1, const nm16u \*pSrcVec2, nm16u \*pDstVec, int nSize)
- void nmppsOr\_32u (const nm32u \*pSrcVec1, const nm32u \*pSrcVec2, nm32u \*pDstVec, int nSize)
- void nmppsOr\_64u (const nm64u \*pSrcVec1, const nm64u \*pSrcVec2, nm64u \*pDstVec, int nSize)

### 6.212.1 Подробное описание

Функция логического "ИЛИ" между двумя векторами.

$$pDstVec[i] = pSrcVec1[i] \vee pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.212.2 Функции

## 6.212.2.1 nmppsOr\_1()

```
void nmppsOr_1 (
    const nm1 * pSrcVec1,
    const nm1 * pSrcVec2,
    nm1 * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 115   | 0.05       |
| im0      | im0      | im0     | 2048  | 115   | 0.05       |
| im0      | im0      | im2     | 2048  | 115   | 0.05       |
| im0      | im0      | im2     | 2048  | 115   | 0.05       |
| im0      | im1      | im0     | 2048  | 145   | 0.07       |
| im0      | im1      | im0     | 2048  | 145   | 0.07       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im1      | im0      | im0     | 2048  | 142   | 0.06       |
| im1      | im0      | im0     | 2048  | 142   | 0.06       |
| im1      | im0      | im2     | 2048  | 142   | 0.06       |
| im1      | im0      | im2     | 2048  | 142   | 0.06       |
| im1      | im1      | im0     | 2048  | 174   | 0.08       |
| im1      | im1      | im0     | 2048  | 174   | 0.08       |
| im1      | im1      | im2     | 2048  | 174   | 0.08       |
| im1      | im1      | im2     | 2048  | 174   | 0.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 115   | 0.05       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im0     | 2048  | 174   | 0.08       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 116   | 0.90       |
| im0      | im1      | im2     | 128   | 116   | 0.90       |
| im0      | im1      | im2     | 1024  | 133   | 0.12       |
| im0      | im1      | im2     | 1024  | 133   | 0.12       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |
| im0      | im1      | im2     | 2048  | 145   | 0.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 145   | 0.07       |

### 6.212.2.2 nmppsOr\_16u()

```
void nmppsOr_16u (
    const nm16u * pSrcVec1,
    const nm16u * pSrcVec2,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 1547  | 0.75       |
| im0      | im0      | im0     | 2048  | 1547  | 0.75       |
| im0      | im0      | im2     | 2048  | 1075  | 0.52       |
| im0      | im0      | im2     | 2048  | 1075  | 0.52       |
| im0      | im1      | im0     | 2048  | 1101  | 0.53       |
| im0      | im1      | im0     | 2048  | 1101  | 0.53       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im1      | im0      | im0     | 2048  | 1075  | 0.52       |
| im1      | im0      | im0     | 2048  | 1075  | 0.52       |
| im1      | im0      | im2     | 2048  | 670   | 0.32       |
| im1      | im0      | im2     | 2048  | 670   | 0.32       |
| im1      | im1      | im0     | 2048  | 1183  | 0.57       |
| im1      | im1      | im0     | 2048  | 1183  | 0.57       |
| im1      | im1      | im2     | 2048  | 1183  | 0.57       |
| im1      | im1      | im2     | 2048  | 1183  | 0.57       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 670   | 0.32       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 1547  | 0.75       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 116   | 14.50      |
| im0      | im1      | im2     | 8     | 116   | 14.50      |
| im0      | im1      | im2     | 128   | 145   | 1.13       |
| im0      | im1      | im2     | 128   | 145   | 1.13       |
| im0      | im1      | im2     | 1024  | 391   | 0.38       |
| im0      | im1      | im2     | 1024  | 391   | 0.38       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |
| im0      | im1      | im2     | 2048  | 671   | 0.32       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 116   | 14.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 671   | 0.32       |

### 6.212.2.3 nmppsOr\_2u()

```
void nmppsOr_2u (
    const nm2u * pSrcVec1,
    const nm2u * pSrcVec2,
    nm2u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 203   | 0.09       |
| im0      | im0      | im0     | 2048  | 203   | 0.09       |
| im0      | im0      | im2     | 2048  | 179   | 0.08       |
| im0      | im0      | im2     | 2048  | 179   | 0.08       |
| im0      | im1      | im0     | 2048  | 206   | 0.10       |
| im0      | im1      | im0     | 2048  | 206   | 0.10       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im1      | im0      | im0     | 2048  | 180   | 0.08       |
| im1      | im0      | im0     | 2048  | 180   | 0.08       |
| im1      | im0      | im2     | 2048  | 179   | 0.08       |
| im1      | im0      | im2     | 2048  | 179   | 0.08       |
| im1      | im1      | im0     | 2048  | 244   | 0.11       |
| im1      | im1      | im0     | 2048  | 244   | 0.11       |
| im1      | im1      | im2     | 2048  | 244   | 0.11       |
| im1      | im1      | im2     | 2048  | 244   | 0.11       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 179   | 0.08       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im0     | 2048  | 244   | 0.11       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 119   | 0.92       |
| im0      | im1      | im2     | 128   | 119   | 0.92       |
| im0      | im1      | im2     | 1024  | 145   | 0.14       |
| im0      | im1      | im2     | 1024  | 145   | 0.14       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |
| im0      | im1      | im2     | 2048  | 180   | 0.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 180   | 0.08       |

#### 6.212.2.4 nmppsOr\_32u()

```
void nmppsOr_32u (
    const nm32u * pSrcVec1,
    const nm32u * pSrcVec2,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 3083  | 1.50       |
| im0      | im0      | im0     | 2048  | 3083  | 1.50       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2099  | 1.02       |
| im0      | im0      | im2     | 2048  | 2099  | 1.02       |
| im0      | im1      | im0     | 2048  | 2125  | 1.03       |
| im0      | im1      | im0     | 2048  | 2125  | 1.03       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im1      | im0      | im0     | 2048  | 2099  | 1.02       |
| im1      | im0      | im0     | 2048  | 2099  | 1.02       |
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |
| im1      | im1      | im0     | 2048  | 2255  | 1.10       |
| im1      | im1      | im0     | 2048  | 2255  | 1.10       |
| im1      | im1      | im2     | 2048  | 2255  | 1.10       |
| im1      | im1      | im2     | 2048  | 2255  | 1.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 1230  | 0.60       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 3083  | 1.50       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 8     | 119   | 14.87      |
| im0      | im1      | im2     | 128   | 180   | 1.40       |
| im0      | im1      | im2     | 128   | 180   | 1.40       |
| im0      | im1      | im2     | 1024  | 671   | 0.65       |
| im0      | im1      | im2     | 1024  | 671   | 0.65       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 119   | 14.87      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1231  | 0.60       |

## 6.212.2.5 nmppsOr\_4u()

```
void nmppsOr_4u (
    const nm4u * pSrcVec1,
    const nm4u * pSrcVec2,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 395   | 0.19       |
| im0      | im0      | im0     | 2048  | 395   | 0.19       |
| im0      | im0      | im2     | 2048  | 307   | 0.14       |
| im0      | im0      | im2     | 2048  | 307   | 0.14       |
| im0      | im1      | im0     | 2048  | 333   | 0.16       |
| im0      | im1      | im0     | 2048  | 333   | 0.16       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |
| im1      | im0      | im0     | 2048  | 307   | 0.14       |
| im1      | im0      | im0     | 2048  | 307   | 0.14       |
| im1      | im0      | im2     | 2048  | 250   | 0.12       |
| im1      | im0      | im2     | 2048  | 250   | 0.12       |
| im1      | im1      | im0     | 2048  | 379   | 0.18       |
| im1      | im1      | im0     | 2048  | 379   | 0.18       |
| im1      | im1      | im2     | 2048  | 379   | 0.18       |
| im1      | im1      | im2     | 2048  | 379   | 0.18       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 250   | 0.12       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 395   | 0.19       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 125   | 0.97       |
| im0      | im1      | im2     | 128   | 125   | 0.97       |
| im0      | im1      | im2     | 1024  | 180   | 0.17       |
| im0      | im1      | im2     | 1024  | 180   | 0.17       |
| im0      | im1      | im2     | 2048  | 251   | 0.12       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 251   | 0.12       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 251   | 0.12       |

#### 6.212.2.6 nmppsOr\_64u()

```
void nmppsOr_64u (
    const nm64u * pSrcVec1,
    const nm64u * pSrcVec2,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6154  | 3.00       |
| im0      | im0      | im0     | 2048  | 6154  | 3.00       |
| im0      | im0      | im2     | 2048  | 4146  | 2.02       |
| im0      | im0      | im2     | 2048  | 4146  | 2.02       |
| im0      | im1      | im0     | 2048  | 4172  | 2.03       |
| im0      | im1      | im0     | 2048  | 4172  | 2.03       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im1      | im0      | im0     | 2048  | 4146  | 2.02       |
| im1      | im0      | im0     | 2048  | 4146  | 2.02       |
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |
| im1      | im1      | im0     | 2048  | 4398  | 2.14       |
| im1      | im1      | im0     | 2048  | 4398  | 2.14       |
| im1      | im1      | im2     | 2048  | 4398  | 2.14       |
| im1      | im1      | im2     | 2048  | 4398  | 2.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 2349  | 1.14       |



The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6154  | 3.00       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 124   | 15.50      |
| im0      | im1      | im2     | 8     | 124   | 15.50      |
| im0      | im1      | im2     | 128   | 250   | 1.95       |
| im0      | im1      | im2     | 128   | 250   | 1.95       |
| im0      | im1      | im2     | 1024  | 1230  | 1.20       |
| im0      | im1      | im2     | 1024  | 1230  | 1.20       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 124   | 15.50      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2350  | 1.14       |

#### 6.212.2.7 nmppsOr\_8u()

```
void nmppsOr_8u (
    const nm8u * pSrcVec1,
    const nm8u * pSrcVec2,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 779   | 0.38       |
| im0      | im0      | im0     | 2048  | 779   | 0.38       |
| im0      | im0      | im2     | 2048  | 563   | 0.27       |
| im0      | im0      | im2     | 2048  | 563   | 0.27       |
| im0      | im1      | im0     | 2048  | 589   | 0.28       |
| im0      | im1      | im0     | 2048  | 589   | 0.28       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im0     | 2048  | 563   | 0.27       |
| im1      | im0      | im0     | 2048  | 563   | 0.27       |
| im1      | im0      | im2     | 2048  | 390   | 0.19       |
| im1      | im0      | im2     | 2048  | 390   | 0.19       |
| im1      | im1      | im0     | 2048  | 647   | 0.31       |
| im1      | im1      | im0     | 2048  | 647   | 0.31       |
| im1      | im1      | im2     | 2048  | 647   | 0.31       |
| im1      | im1      | im2     | 2048  | 647   | 0.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im2     | 2048  | 390   | 0.19       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 779   | 0.38       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 133   | 1.03       |
| im0      | im1      | im2     | 128   | 133   | 1.03       |
| im0      | im1      | im2     | 1024  | 251   | 0.24       |
| im0      | im1      | im2     | 1024  | 251   | 0.24       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |
| im0      | im1      | im2     | 2048  | 391   | 0.19       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 391   | 0.19       |

## 6.213 nmppsOr3V\_

Функция логического "ИЛИ" между четырьмя векторами.

Функции

- void nmppsOr3V\_64u (nm64u \*pSrcVec1, nm64u \*pSrcVec2, nm64u \*pSrcVec3, nm64u \*pDstVec, int nSize)

### 6.213.1 Подробное описание

Функция логического "ИЛИ" между четырьмя векторами.

$$pDstVec[i] = pSrcVec1[i] \vee pSrcVec2[i] \vee pSrcVec3[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| pSrcVec3 | Третий входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.214 nmppsOr4V\_

Функция логического "ИЛИ" между четырьмя векторами.

Функции

- void nmppsOr4V\_64u (nm64u \*pSrcVec1, nm64u \*pSrcVec2, nm64u \*pSrcVec3, nm64u \*pSrcVec4, nm64u \*pDstVec, int nSize)

## 6.214.1 Подробное описание

Функция логического "ИЛИ" между четырьмя векторами.

$$pDstVec[i] = pSrcVec1[i] \vee pSrcVec2[i] \vee pSrcVec3[i] \vee pSrcVec4[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| pSrcVec3 | Третий входной вектор.       |
| pSrcVec4 | Четвертый входной вектор.    |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.215 nmppsXorC

Функция логического "Исключающего ИЛИ" между вектором и константой.

Функции

- void `nmppsXorC_8u` (const `nm8u` \*pSrcVec, `uint8b` nVal, `nm8u` \*pDstVec, int nSize)
- void `nmppsXorC_16u` (const `nm16u` \*pSrcVec, `uint16b` nVal, `nm16u` \*pDstVec, int nSize)
- void `nmppsXorC_32u` (const `nm32u` \*pSrcVec, `uint32b` nVal, `nm32u` \*pDstVec, int nSize)
- void `nmppsXorC_64u` (const `nm64u` \*pSrcVec, `uint64b` nVal, `nm64u` \*pDstVec, int nSize)

### 6.215.1 Подробное описание

Функция логического "Исключающего ИЛИ" между вектором и константой.

$$pDstVec[i] = pSrcVec[i] \vee nVal$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nVal    | Константа.                   |
| pnVal   | Указатель на константу.      |
| nSize   | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.215.2 Функции

## 6.215.2.1 nmppsXorC\_16u()

```
void nmppsXorC_16u (
    const nm16u * pSrcVec,
    uint16b nVal,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 1082  | 0.52       |
| im0     | 5    | im0     | 2048  | 1082  | 0.52       |
| im0     | 5    | im1     | 2048  | 692   | 0.33       |
| im0     | 5    | im1     | 2048  | 692   | 0.33       |
| im1     | 5    | im0     | 2048  | 694   | 0.33       |
| im1     | 5    | im0     | 2048  | 694   | 0.33       |
| im1     | 5    | im1     | 2048  | 1206  | 0.58       |
| im1     | 5    | im1     | 2048  | 1206  | 0.58       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 692   | 0.33       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 1206  | 0.58       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 126   | 15.75      |
| im0     | 5    | im1     | 8     | 126   | 15.75      |
| im0     | 5    | im1     | 128   | 182   | 1.42       |
| im0     | 5    | im1     | 128   | 182   | 1.42       |
| im0     | 5    | im1     | 1024  | 420   | 0.41       |
| im0     | 5    | im1     | 1024  | 420   | 0.41       |
| im0     | 5    | im1     | 2048  | 692   | 0.33       |
| im0     | 5    | im1     | 2048  | 692   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 692   | 0.33       |

## 6.215.2.2 nmppsXorC\_32u()

```
void nmppsXorC_32u (
    const nm32u * pSrcVec,
    uint32b nVal,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 2103  | 1.02       |
| im0     | 5    | im0     | 2048  | 2103  | 1.02       |
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |
| im1     | 5    | im0     | 2048  | 1235  | 0.60       |
| im1     | 5    | im0     | 2048  | 1235  | 0.60       |
| im1     | 5    | im1     | 2048  | 2259  | 1.10       |
| im1     | 5    | im1     | 2048  | 2259  | 1.10       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 2259  | 1.10       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 127   | 15.87      |
| im0     | 5    | im1     | 8     | 127   | 15.87      |
| im0     | 5    | im1     | 128   | 216   | 1.68       |
| im0     | 5    | im1     | 128   | 216   | 1.68       |
| im0     | 5    | im1     | 1024  | 689   | 0.67       |
| im0     | 5    | im1     | 1024  | 689   | 0.67       |
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 127   | 15.87      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 1233  | 0.60       |

### 6.215.2.3 nmppsXorC\_64u()

```
void nmppsXorC_64u (
    const nm64u * pSrcVec,
    uint64b nVal,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 4147  | 2.02       |
| im0     | 5    | im0     | 2048  | 4147  | 2.02       |
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |
| im1     | 5    | im0     | 2048  | 2322  | 1.13       |
| im1     | 5    | im0     | 2048  | 2322  | 1.13       |
| im1     | 5    | im1     | 2048  | 4367  | 2.13       |
| im1     | 5    | im1     | 2048  | 4367  | 2.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 4367  | 2.13       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 136   | 17.00      |
| im0     | 5    | im1     | 8     | 136   | 17.00      |
| im0     | 5    | im1     | 128   | 277   | 2.16       |
| im0     | 5    | im1     | 128   | 277   | 2.16       |



| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 1024  | 1229  | 1.20       |
| im0     | 5    | im1     | 1024  | 1229  | 1.20       |
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 136   | 17.00      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 2317  | 1.13       |

#### 6.215.2.4 nmppsXorC\_8u()

```
void nmppsXorC_8u (
    const nm8u * pSrcVec,
    uint8b nVal,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im0     | 2048  | 572   | 0.27       |
| im0     | 5    | im0     | 2048  | 572   | 0.27       |
| im0     | 5    | im1     | 2048  | 422   | 0.20       |
| im0     | 5    | im1     | 2048  | 422   | 0.20       |
| im1     | 5    | im0     | 2048  | 424   | 0.20       |
| im1     | 5    | im0     | 2048  | 424   | 0.20       |
| im1     | 5    | im1     | 2048  | 680   | 0.33       |
| im1     | 5    | im1     | 2048  | 680   | 0.33       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 422   | 0.20       |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im1     | 5    | im1     | 2048  | 680   | 0.33       |

Perfomance table 1

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 126   | 15.75      |
| im0     | 5    | im1     | 8     | 126   | 15.75      |
| im0     | 5    | im1     | 128   | 157   | 1.22       |
| im0     | 5    | im1     | 128   | 157   | 1.22       |
| im0     | 5    | im1     | 1024  | 286   | 0.27       |
| im0     | 5    | im1     | 1024  | 286   | 0.27       |
| im0     | 5    | im1     | 2048  | 422   | 0.20       |
| im0     | 5    | im1     | 2048  | 422   | 0.20       |

The best configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | nVal | pDstVec | nSize | ticks | ticks/elem |
|---------|------|---------|-------|-------|------------|
| im0     | 5    | im1     | 2048  | 422   | 0.20       |

## 6.216 nmppsXor

Функция логического "Исключающего ИЛИ" между двумя векторами.

Функции

- void `nmppsXor_4u` (const `nm4u` \*pSrcVec1, const `nm4u` \*pSrcVec2, `nm4u` \*pDstVec, int nSize)
- void `nmppsXor_8u` (const `nm8u` \*pSrcVec1, const `nm8u` \*pSrcVec2, `nm8u` \*pDstVec, int nSize)
- void `nmppsXor_16u` (const `nm16u` \*pSrcVec1, const `nm16u` \*pSrcVec2, `nm16u` \*pDstVec, int nSize)
- void `nmppsXor_32u` (const `nm32u` \*pSrcVec1, const `nm32u` \*pSrcVec2, `nm32u` \*pDstVec, int nSize)
- void `nmppsXor_64u` (const `nm64u` \*pSrcVec1, const `nm64u` \*pSrcVec2, `nm64u` \*pDstVec, int nSize)

### 6.216.1 Подробное описание

Функция логического "Исключающего ИЛИ" между двумя векторами.

$$pDstVec[i] = pSrcVec1[i] \vee pSrcVec2[i]$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.216.2 Функции

## 6.216.2.1 nmppsXor\_16u()

```
void nmppsXor_16u (
    const nm16u * pSrcVec1,
    const nm16u * pSrcVec2,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1113  | 0.54       |
| im0      | im0      | im2     | 2048  | 1113  | 0.54       |
| im0      | im0      | im0     | 2048  | 1578  | 0.77       |
| im0      | im0      | im0     | 2048  | 1578  | 0.77       |
| im0      | im1      | im2     | 2048  | 1216  | 0.59       |
| im0      | im1      | im2     | 2048  | 1216  | 0.59       |
| im0      | im1      | im0     | 2048  | 1600  | 0.78       |
| im0      | im1      | im0     | 2048  | 1600  | 0.78       |
| im1      | im0      | im2     | 2048  | 1121  | 0.54       |
| im1      | im0      | im2     | 2048  | 1121  | 0.54       |
| im1      | im0      | im0     | 2048  | 1121  | 0.54       |
| im1      | im0      | im0     | 2048  | 1121  | 0.54       |
| im1      | im1      | im2     | 2048  | 1186  | 0.57       |
| im1      | im1      | im2     | 2048  | 1186  | 0.57       |
| im1      | im1      | im0     | 2048  | 1186  | 0.57       |
| im1      | im1      | im0     | 2048  | 1186  | 0.57       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 1113  | 0.54       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im0     | 2048  | 1600  | 0.78       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 8     | 117   | 14.62      |
| im0      | im1      | im2     | 128   | 171   | 1.33       |
| im0      | im1      | im2     | 128   | 171   | 1.33       |
| im0      | im1      | im2     | 1024  | 656   | 0.64       |
| im0      | im1      | im2     | 1024  | 656   | 0.64       |
| im0      | im1      | im2     | 2048  | 1216  | 0.59       |
| im0      | im1      | im2     | 2048  | 1216  | 0.59       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1216  | 0.59       |

### 6.216.2.2 nmppsXor\_32u()

```
void nmppsXor_32u (
    const nm32u * pSrcVec1,
    const nm32u * pSrcVec2,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2153  | 1.05       |
| im0      | im0      | im2     | 2048  | 2153  | 1.05       |
| im0      | im0      | im0     | 2048  | 3114  | 1.52       |
| im0      | im0      | im0     | 2048  | 3114  | 1.52       |
| im0      | im1      | im2     | 2048  | 2336  | 1.14       |
| im0      | im1      | im2     | 2048  | 2336  | 1.14       |
| im0      | im1      | im0     | 2048  | 3120  | 1.52       |
| im0      | im1      | im0     | 2048  | 3120  | 1.52       |
| im1      | im0      | im2     | 2048  | 2161  | 1.05       |
| im1      | im0      | im2     | 2048  | 2161  | 1.05       |
| im1      | im0      | im0     | 2048  | 2161  | 1.05       |
| im1      | im0      | im0     | 2048  | 2161  | 1.05       |
| im1      | im1      | im2     | 2048  | 2258  | 1.10       |
| im1      | im1      | im2     | 2048  | 2258  | 1.10       |
| im1      | im1      | im0     | 2048  | 2258  | 1.10       |
| im1      | im1      | im0     | 2048  | 2258  | 1.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 2153  | 1.05       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im0     | 2048  | 3120  | 1.52       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 123   | 15.37      |
| im0      | im1      | im2     | 8     | 123   | 15.37      |
| im0      | im1      | im2     | 128   | 240   | 1.87       |
| im0      | im1      | im2     | 128   | 240   | 1.87       |
| im0      | im1      | im2     | 1024  | 1216  | 1.18       |
| im0      | im1      | im2     | 1024  | 1216  | 1.18       |
| im0      | im1      | im2     | 2048  | 2336  | 1.14       |
| im0      | im1      | im2     | 2048  | 2336  | 1.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 123   | 15.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2336  | 1.14       |

### 6.216.2.3 nmppsXor\_4u()

```
void nmppsXor_4u (
    const nm4u * pSrcVec1,
    const nm4u * pSrcVec2,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 333   | 0.16       |
| im0      | im0      | im2     | 2048  | 333   | 0.16       |
| im0      | im0      | im0     | 2048  | 426   | 0.20       |
| im0      | im0      | im0     | 2048  | 426   | 0.20       |
| im0      | im1      | im2     | 2048  | 376   | 0.18       |
| im0      | im1      | im2     | 2048  | 376   | 0.18       |
| im0      | im1      | im0     | 2048  | 460   | 0.22       |
| im0      | im1      | im0     | 2048  | 460   | 0.22       |
| im1      | im0      | im2     | 2048  | 341   | 0.16       |
| im1      | im0      | im2     | 2048  | 341   | 0.16       |
| im1      | im0      | im0     | 2048  | 341   | 0.16       |
| im1      | im0      | im0     | 2048  | 341   | 0.16       |
| im1      | im1      | im2     | 2048  | 382   | 0.18       |
| im1      | im1      | im2     | 2048  | 382   | 0.18       |
| im1      | im1      | im0     | 2048  | 382   | 0.18       |
| im1      | im1      | im0     | 2048  | 382   | 0.18       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 333   | 0.16       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im0     | 2048  | 460   | 0.22       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 130   | 1.01       |
| im0      | im1      | im2     | 128   | 130   | 1.01       |
| im0      | im1      | im2     | 1024  | 240   | 0.23       |
| im0      | im1      | im2     | 1024  | 240   | 0.23       |
| im0      | im1      | im2     | 2048  | 376   | 0.18       |
| im0      | im1      | im2     | 2048  | 376   | 0.18       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 376   | 0.18       |

#### 6.216.2.4 nmppsXor\_64u()

```
void nmppsXor_64u (
    const nm64u * pSrcVec1,
    const nm64u * pSrcVec2,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im0      | im1      | im0     | 2048  | 6159  | 3.00       |
| im0      | im1      | im0     | 2048  | 6159  | 3.00       |
| im1      | im0      | im2     | 2048  | 4240  | 2.07       |
| im1      | im0      | im2     | 2048  | 4240  | 2.07       |
| im1      | im0      | im0     | 2048  | 4240  | 2.07       |
| im1      | im0      | im0     | 2048  | 4240  | 2.07       |
| im1      | im1      | im2     | 2048  | 4401  | 2.14       |
| im1      | im1      | im2     | 2048  | 4401  | 2.14       |
| im1      | im1      | im0     | 2048  | 4401  | 2.14       |
| im1      | im1      | im0     | 2048  | 4401  | 2.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 4232  | 2.06       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0     | 2048  | 6185  | 3.02       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 129   | 16.12      |
| im0      | im1      | im2     | 8     | 129   | 16.12      |
| im0      | im1      | im2     | 128   | 375   | 2.92       |
| im0      | im1      | im2     | 128   | 375   | 2.92       |
| im0      | im1      | im2     | 1024  | 2335  | 2.28       |
| im0      | im1      | im2     | 1024  | 2335  | 2.28       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 129   | 16.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4575  | 2.23       |



## 6.216.2.5 nmppsXor\_8u()

```
void nmppsXor_8u (
    const nm8u * pSrcVec1,
    const nm8u * pSrcVec2,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 593   | 0.28       |
| im0      | im0      | im2     | 2048  | 593   | 0.28       |
| im0      | im0      | im0     | 2048  | 810   | 0.39       |
| im0      | im0      | im0     | 2048  | 810   | 0.39       |
| im0      | im1      | im2     | 2048  | 656   | 0.32       |
| im0      | im1      | im2     | 2048  | 656   | 0.32       |
| im0      | im1      | im0     | 2048  | 840   | 0.41       |
| im0      | im1      | im0     | 2048  | 840   | 0.41       |
| im1      | im0      | im2     | 2048  | 601   | 0.29       |
| im1      | im0      | im2     | 2048  | 601   | 0.29       |
| im1      | im0      | im0     | 2048  | 601   | 0.29       |
| im1      | im0      | im0     | 2048  | 601   | 0.29       |
| im1      | im1      | im2     | 2048  | 650   | 0.31       |
| im1      | im1      | im2     | 2048  | 650   | 0.31       |
| im1      | im1      | im0     | 2048  | 650   | 0.31       |
| im1      | im1      | im0     | 2048  | 650   | 0.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im2     | 2048  | 593   | 0.28       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im0     | 2048  | 840   | 0.41       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 8     | 114   | 14.25      |
| im0      | im1      | im2     | 128   | 144   | 1.12       |
| im0      | im1      | im2     | 128   | 144   | 1.12       |
| im0      | im1      | im2     | 1024  | 376   | 0.36       |
| im0      | im1      | im2     | 1024  | 376   | 0.36       |
| im0      | im1      | im2     | 2048  | 656   | 0.32       |

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 656   | 0.32       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 8     | 114   | 14.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 656   | 0.32       |

## 6.217 nmppsMaskV\_

Функция логического ИЛИ с предварительным маскированием двух векторов.

Функции

- void `nmppsMaskV_64u` (`nm64u` \*pSrcVec1, `nm64u` \*pSrcVec2, `nm64u` \*pMaskVec, `nm64u` \*pDstVec, int nSize)

### 6.217.1 Подробное описание

Функция логического ИЛИ с предварительным маскированием двух векторов.

$$pDstVec[i] = (pSrcVec1[i] \quad and \quad pMaskVec[i]) \quad or \quad (pSrcVec2[i] \quad and \quad \overline{pMaskVec[i]})$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| pMaskVec | Вектор маски.                |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.217.2 Функции

#### 6.217.2.1 nmppsMaskV\_64u()

```
void nmppsMaskV_64u (
    nm64u * pSrcVec1,
```

```

nm64u * pSrcVec2,
nm64u * pMaskVec,
nm64u * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im0      | im0     | 2048  | 6194  | 3.02       |
| im1      | im0      | im0      | im0     | 2048  | 6194  | 3.02       |
| im1      | im0      | im0      | im3     | 2048  | 4308  | 2.10       |
| im1      | im0      | im0      | im3     | 2048  | 4308  | 2.10       |
| im1      | im0      | im2      | im0     | 2048  | 6131  | 2.99       |
| im1      | im0      | im2      | im0     | 2048  | 6131  | 2.99       |
| im1      | im0      | im2      | im3     | 2048  | 4308  | 2.10       |
| im1      | im0      | im2      | im3     | 2048  | 4308  | 2.10       |
| im1      | im1      | im0      | im0     | 2048  | 6420  | 3.13       |
| im1      | im1      | im0      | im0     | 2048  | 6420  | 3.13       |
| im1      | im1      | im0      | im3     | 2048  | 6420  | 3.13       |
| im1      | im1      | im0      | im3     | 2048  | 6420  | 3.13       |
| im1      | im1      | im2      | im0     | 2048  | 6420  | 3.13       |
| im1      | im1      | im2      | im0     | 2048  | 6420  | 3.13       |
| im1      | im1      | im2      | im3     | 2048  | 6420  | 3.13       |
| im1      | im1      | im2      | im3     | 2048  | 6420  | 3.13       |
| im0      | im0      | im0      | im0     | 2048  | 8241  | 4.02       |
| im0      | im0      | im0      | im0     | 2048  | 8241  | 4.02       |
| im0      | im0      | im0      | im3     | 2048  | 6288  | 3.07       |
| im0      | im0      | im0      | im3     | 2048  | 6288  | 3.07       |
| im0      | im0      | im2      | im0     | 2048  | 8178  | 3.99       |
| im0      | im0      | im2      | im0     | 2048  | 8178  | 3.99       |
| im0      | im0      | im2      | im3     | 2048  | 6288  | 3.07       |
| im0      | im0      | im2      | im3     | 2048  | 6288  | 3.07       |
| im0      | im1      | im0      | im0     | 2048  | 6172  | 3.01       |
| im0      | im1      | im0      | im0     | 2048  | 6172  | 3.01       |
| im0      | im1      | im0      | im3     | 2048  | 4312  | 2.10       |
| im0      | im1      | im0      | im3     | 2048  | 4312  | 2.10       |
| im0      | im1      | im2      | im0     | 2048  | 4312  | 2.10       |
| im0      | im1      | im2      | im0     | 2048  | 4312  | 2.10       |
| im0      | im1      | im2      | im3     | 2048  | 4312  | 2.10       |
| im0      | im1      | im2      | im3     | 2048  | 4312  | 2.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im1      | im0      | im0      | im3     | 2048  | 4308  | 2.10       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im0      | im0      | im0      | im0     | 2048  | 8241  | 4.02       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2      | im3     | 8     | 131   | 16.37      |
| im0      | im1      | im2      | im3     | 8     | 131   | 16.37      |
| im0      | im1      | im2      | im3     | 128   | 352   | 2.75       |
| im0      | im1      | im2      | im3     | 128   | 352   | 2.75       |
| im0      | im1      | im2      | im3     | 1024  | 2200  | 2.14       |
| im0      | im1      | im2      | im3     | 1024  | 2200  | 2.14       |
| im0      | im1      | im2      | im3     | 2048  | 4312  | 2.10       |
| im0      | im1      | im2      | im3     | 2048  | 4312  | 2.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2      | im3     | 8     | 131   | 16.37      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pMaskVec | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2      | im3     | 2048  | 4312  | 2.10       |

## 6.218 nmppsRShiftC

Операция арифметического сдвига вправо.

### Функции

- void nmppsRShiftC\_8s (const nm8s \*pSrcVec, int nShift, nm8s \*pDstVec, int nSize)
- void nmppsRShiftC\_16s (const nm16s \*pSrcVec, int nShift, nm16s \*pDstVec, int nSize)
- void nmppsRShiftC\_32s (const nm32s \*pSrcVec, int nShift, nm32s \*pDstVec, int nSize)
- void nmppsRShiftC\_64s (const nm64s \*pSrcVec, int nShift, nm64s \*pDstVec, int nSize)

### 6.218.1 Подробное описание

Операция арифметического сдвига вправо.

$$pDstVec[i] = pSrcVec[i] >> nShift,$$

$$i = \overline{0 \dots nSize - 1}$$

Функции реализуют операции арифметического сдвига вправо битов элементов вектора. Освободившиеся биты заполняются знаковым битом - старшим битом.

### Аргументы

|         |                                                                               |
|---------|-------------------------------------------------------------------------------|
| pSrcVec | Входной вектор.                                                               |
| nSize   | Размер вектора в элементах.                                                   |
| nShift  | Параметр определяет на сколько позиций нужно сдвинуть биты элементов вектора. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.218.2 Функции

## 6.218.2.1 nmppsRShiftC\_16s()

```
void nmppsRShiftC_16s (
    const nm16s * pSrcVec,
    int nShift,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 1105  | 0.53       |
| im0     | im0     | 1      | 2048  | 1105  | 0.53       |
| im0     | im2     | 1      | 2048  | 633   | 0.30       |
| im0     | im2     | 1      | 2048  | 633   | 0.30       |
| im1     | im0     | 1      | 2048  | 692   | 0.33       |
| im1     | im0     | 1      | 2048  | 692   | 0.33       |
| im1     | im2     | 1      | 2048  | 692   | 0.33       |
| im1     | im2     | 1      | 2048  | 692   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im2     | 1      | 2048  | 633   | 0.30       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 1105  | 0.53       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 694   | 0.33       |
| im0     | im1     | 1      | 2048  | 694   | 0.33       |
| im0     | im1     | 2      | 2048  | 691   | 0.33       |
| im0     | im1     | 2      | 2048  | 691   | 0.33       |
| im0     | im1     | 3      | 2048  | 694   | 0.33       |
| im0     | im1     | 3      | 2048  | 694   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 2      | 2048  | 691   | 0.33       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 694   | 0.33       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 149   | 18.62      |
| im0     | im1     | 1      | 8     | 149   | 18.62      |
| im0     | im1     | 1      | 128   | 185   | 1.44       |
| im0     | im1     | 1      | 128   | 185   | 1.44       |
| im0     | im1     | 1      | 1024  | 424   | 0.41       |
| im0     | im1     | 1      | 1024  | 424   | 0.41       |
| im0     | im1     | 1      | 2048  | 694   | 0.33       |
| im0     | im1     | 1      | 2048  | 694   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 149   | 18.62      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 694   | 0.33       |

#### 6.218.2.2 nmppsRShiftC\_32s()

```
void nmppsRShiftC_32s (
    const nm32s * pSrcVec,
    int nShift,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 2129  | 1.03       |
| im0     | im0     | 1      | 2048  | 2129  | 1.03       |
| im0     | im2     | 1      | 2048  | 1145  | 0.55       |
| im0     | im2     | 1      | 2048  | 1145  | 0.55       |
| im1     | im0     | 1      | 2048  | 1236  | 0.60       |
| im1     | im0     | 1      | 2048  | 1236  | 0.60       |
| im1     | im2     | 1      | 2048  | 1236  | 0.60       |
| im1     | im2     | 1      | 2048  | 1236  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im2     | 1      | 2048  | 1145  | 0.55       |



The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 2129  | 1.03       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |
| im0     | im1     | 2      | 2048  | 1231  | 0.60       |
| im0     | im1     | 2      | 2048  | 1231  | 0.60       |
| im0     | im1     | 3      | 2048  | 1238  | 0.60       |
| im0     | im1     | 3      | 2048  | 1238  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 2      | 2048  | 1231  | 0.60       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 150   | 18.75      |
| im0     | im1     | 1      | 8     | 150   | 18.75      |
| im0     | im1     | 1      | 128   | 220   | 1.71       |
| im0     | im1     | 1      | 128   | 220   | 1.71       |
| im0     | im1     | 1      | 1024  | 694   | 0.67       |
| im0     | im1     | 1      | 1024  | 694   | 0.67       |
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 150   | 18.75      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1238  | 0.60       |

## 6.218.2.3 nmppsRShiftC\_64s()

```
void nmppsRShiftC_64s (
    const nm64s * pSrcVec,
    int nShift,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 4168  | 2.03       |
| im0     | im0     | 1      | 2048  | 4168  | 2.03       |
| im0     | im2     | 1      | 2048  | 2160  | 1.05       |
| im0     | im2     | 1      | 2048  | 2160  | 1.05       |
| im1     | im0     | 1      | 2048  | 2315  | 1.13       |
| im1     | im0     | 1      | 2048  | 2315  | 1.13       |
| im1     | im2     | 1      | 2048  | 2315  | 1.13       |
| im1     | im2     | 1      | 2048  | 2315  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im2     | 1      | 2048  | 2160  | 1.05       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 4168  | 2.03       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 2      | 2048  | 2314  | 1.12       |
| im0     | im1     | 2      | 2048  | 2314  | 1.12       |
| im0     | im1     | 3      | 2048  | 2317  | 1.13       |
| im0     | im1     | 3      | 2048  | 2317  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 2      | 2048  | 2314  | 1.12       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 146   | 18.25      |
| im0     | im1     | 1      | 8     | 146   | 18.25      |
| im0     | im1     | 1      | 128   | 278   | 2.17       |
| im0     | im1     | 1      | 128   | 278   | 2.17       |
| im0     | im1     | 1      | 1024  | 1229  | 1.20       |
| im0     | im1     | 1      | 1024  | 1229  | 1.20       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 146   | 18.25      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

#### 6.218.2.4 nmppsRShiftC\_8s()

```
void nmppsRShiftC_8s (
    const nm8s * pSrcVec,
    int nShift,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 605   | 0.29       |
| im0     | im0     | 1      | 2048  | 605   | 0.29       |
| im0     | im2     | 1      | 2048  | 389   | 0.18       |
| im0     | im2     | 1      | 2048  | 389   | 0.18       |
| im1     | im0     | 1      | 2048  | 432   | 0.21       |
| im1     | im0     | 1      | 2048  | 432   | 0.21       |
| im1     | im2     | 1      | 2048  | 432   | 0.21       |
| im1     | im2     | 1      | 2048  | 432   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im2     | 1      | 2048  | 389   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 605   | 0.29       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 436   | 0.21       |
| im0     | im1     | 1      | 2048  | 436   | 0.21       |
| im0     | im1     | 2      | 2048  | 433   | 0.21       |
| im0     | im1     | 2      | 2048  | 433   | 0.21       |
| im0     | im1     | 3      | 2048  | 436   | 0.21       |
| im0     | im1     | 3      | 2048  | 436   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 2      | 2048  | 433   | 0.21       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 436   | 0.21       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 161   | 20.12      |
| im0     | im1     | 1      | 8     | 161   | 20.12      |
| im0     | im1     | 1      | 128   | 175   | 1.36       |
| im0     | im1     | 1      | 128   | 175   | 1.36       |
| im0     | im1     | 1      | 1024  | 299   | 0.29       |
| im0     | im1     | 1      | 1024  | 299   | 0.29       |
| im0     | im1     | 1      | 2048  | 436   | 0.21       |
| im0     | im1     | 1      | 2048  | 436   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 161   | 20.12      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 436   | 0.21       |

## 6.219 nmppsRShiftC\_

Операция логического сдвига.

Функции

- void `nmppsRShiftC_4u` (const `nm4u` \*pSrcVec, int nShift, `nm4u` \*pDstVec, int nSize)
- void `nmppsRShiftC_8u` (const `nm8u` \*pSrcVec, int nShift, `nm8u` \*pDstVec, int nSize)
- void `nmppsRShiftC_16u` (const `nm16u` \*pSrcVec, int nShift, `nm16u` \*pDstVec, int nSize)
- void `nmppsRShiftC_32u` (const `nm32u` \*pSrcVec, int nShift, `nm32u` \*pDstVec, int nSize)
- void `nmppsRShiftC_64u` (const `nm64u` \*pSrcVec, int nShift, `nm64u` \*pDstVec, int nSize)

### 6.219.1 Подробное описание

Операция логического сдвига.

$$pDstVec[i] = pSrcVec[i] >> nShift,$$

$$i = \overline{0 \dots nSize - 1}$$

Функции реализуют операции логического сдвига вправо битов элементов вектора. Сдвиг осуществляется на число бит, указанных в соответствующем операнде. Освободившиеся биты заполняются нулями.

Аргументы

|         |                                                                     |
|---------|---------------------------------------------------------------------|
| pSrcVec | Входной вектор.                                                     |
| nSize   | Размер векторов в элементах.                                        |
| nShift  | Определяет на сколько позиций нужно сдвинуть биты элемента вектора. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

### 6.219.2 Функции

## 6.219.2.1 nmppsRShiftC\_16u()

```
void nmppsRShiftC_16u (
    const nm16u * pSrcVec,
    int nShift,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 1108  | 0.54       |
| im0     | im0     | 1      | 2048  | 1108  | 0.54       |
| im0     | im1     | 1      | 2048  | 697   | 0.34       |
| im0     | im1     | 1      | 2048  | 697   | 0.34       |
| im1     | im0     | 1      | 2048  | 695   | 0.33       |
| im1     | im0     | 1      | 2048  | 695   | 0.33       |
| im1     | im1     | 1      | 2048  | 1207  | 0.58       |
| im1     | im1     | 1      | 2048  | 1207  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im0     | 1      | 2048  | 695   | 0.33       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im1     | 1      | 2048  | 1207  | 0.58       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 697   | 0.34       |
| im0     | im1     | 1      | 2048  | 697   | 0.34       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 697   | 0.34       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 697   | 0.34       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 152   | 19.00      |
| im0     | im1     | 1      | 8     | 152   | 19.00      |
| im0     | im1     | 1      | 128   | 188   | 1.46       |
| im0     | im1     | 1      | 128   | 188   | 1.46       |
| im0     | im1     | 1      | 1024  | 427   | 0.41       |
| im0     | im1     | 1      | 1024  | 427   | 0.41       |
| im0     | im1     | 1      | 2048  | 697   | 0.34       |
| im0     | im1     | 1      | 2048  | 697   | 0.34       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 152   | 19.00      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 697   | 0.34       |

#### 6.219.2.2 nmppsRShiftC\_32u()

```
void nmppsRShiftC_32u (
    const nm32u * pSrcVec,
    int nShift,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 2128  | 1.03       |
| im0     | im0     | 1      | 2048  | 2128  | 1.03       |
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |
| im1     | im0     | 1      | 2048  | 1235  | 0.60       |
| im1     | im0     | 1      | 2048  | 1235  | 0.60       |
| im1     | im1     | 1      | 2048  | 2259  | 1.10       |
| im1     | im1     | 1      | 2048  | 2259  | 1.10       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im0     | 1      | 2048  | 1235  | 0.60       |

The worst configuration:



| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im1     | 1      | 2048  | 2259  | 1.10       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 148   | 18.50      |
| im0     | im1     | 1      | 8     | 148   | 18.50      |
| im0     | im1     | 1      | 128   | 219   | 1.71       |
| im0     | im1     | 1      | 128   | 219   | 1.71       |
| im0     | im1     | 1      | 1024  | 693   | 0.67       |
| im0     | im1     | 1      | 1024  | 693   | 0.67       |
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 148   | 18.50      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 1237  | 0.60       |

### 6.219.2.3 nmppsRShiftC\_4u()

```
void nmppsRShiftC_4u (
```

```

const nm4u * pSrcVec,
int nShift,
nm4u * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 364   | 0.17       |
| im0     | im0     | 1      | 2048  | 364   | 0.17       |
| im0     | im1     | 1      | 2048  | 314   | 0.15       |
| im0     | im1     | 1      | 2048  | 314   | 0.15       |
| im1     | im0     | 1      | 2048  | 311   | 0.15       |
| im1     | im0     | 1      | 2048  | 311   | 0.15       |
| im1     | im1     | 1      | 2048  | 439   | 0.21       |
| im1     | im1     | 1      | 2048  | 439   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im0     | 1      | 2048  | 311   | 0.15       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im1     | 1      | 2048  | 439   | 0.21       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 314   | 0.15       |
| im0     | im1     | 1      | 2048  | 314   | 0.15       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 314   | 0.15       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 314   | 0.15       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 176   | 22.00      |
| im0     | im1     | 1      | 8     | 176   | 22.00      |

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 128   | 182   | 1.42       |
| im0     | im1     | 1      | 128   | 182   | 1.42       |
| im0     | im1     | 1      | 1024  | 247   | 0.24       |
| im0     | im1     | 1      | 1024  | 247   | 0.24       |
| im0     | im1     | 1      | 2048  | 314   | 0.15       |
| im0     | im1     | 1      | 2048  | 314   | 0.15       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 176   | 22.00      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 314   | 0.15       |

#### 6.219.2.4 nmppsRShiftC\_64u()

```
void nmppsRShiftC_64u (
    const nm64u * pSrcVec,
    int nShift,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 4168  | 2.03       |
| im0     | im0     | 1      | 2048  | 4168  | 2.03       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im1     | im0     | 1      | 2048  | 2315  | 1.13       |
| im1     | im0     | 1      | 2048  | 2315  | 1.13       |
| im1     | im1     | 1      | 2048  | 4363  | 2.13       |
| im1     | im1     | 1      | 2048  | 4363  | 2.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im0     | 1      | 2048  | 2315  | 1.13       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im1     | 1      | 2048  | 4363  | 2.13       |

Performance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

Performance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 146   | 18.25      |
| im0     | im1     | 1      | 8     | 146   | 18.25      |
| im0     | im1     | 1      | 128   | 278   | 2.17       |
| im0     | im1     | 1      | 128   | 278   | 2.17       |
| im0     | im1     | 1      | 1024  | 1229  | 1.20       |
| im0     | im1     | 1      | 1024  | 1229  | 1.20       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 146   | 18.25      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 2317  | 1.13       |

#### 6.219.2.5 nmppsRShiftC\_8u()

```
void nmppsRShiftC_8u (
```

```

const nm8u * pSrcVec,
int nShift,
nm8u * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im0     | 1      | 2048  | 604   | 0.29       |
| im0     | im0     | 1      | 2048  | 604   | 0.29       |
| im0     | im1     | 1      | 2048  | 435   | 0.21       |
| im0     | im1     | 1      | 2048  | 435   | 0.21       |
| im1     | im0     | 1      | 2048  | 431   | 0.21       |
| im1     | im0     | 1      | 2048  | 431   | 0.21       |
| im1     | im1     | 1      | 2048  | 687   | 0.33       |
| im1     | im1     | 1      | 2048  | 687   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im0     | 1      | 2048  | 431   | 0.21       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im1     | im1     | 1      | 2048  | 687   | 0.33       |

Perfomance table 1

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 435   | 0.21       |
| im0     | im1     | 1      | 2048  | 435   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 435   | 0.21       |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 435   | 0.21       |

Perfomance table 2

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 160   | 20.00      |
| im0     | im1     | 1      | 8     | 160   | 20.00      |

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 128   | 174   | 1.35       |
| im0     | im1     | 1      | 128   | 174   | 1.35       |
| im0     | im1     | 1      | 1024  | 298   | 0.29       |
| im0     | im1     | 1      | 1024  | 298   | 0.29       |
| im0     | im1     | 1      | 2048  | 435   | 0.21       |
| im0     | im1     | 1      | 2048  | 435   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 8     | 160   | 20.00      |

The worst configuration:

| pSrcVec | pDstVec | nShift | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|------------|
| im0     | im1     | 1      | 2048  | 435   | 0.21       |

## 6.220 nmppsRShiftC\_AddC\_

Операция логического сдвига.

Операция логического сдвига.

$$pDstVec[i] = (pSrcVec[i] >> nShift) + nAddVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Функции реализуют операции логического сдвига вправо битов элементов вектора с прибавлением константы. Сдвиг осуществляется на число бит, указанных в соответствующем операнде. Освободившиеся биты заполняются нулями.

Аргументы

|         |                                                                     |
|---------|---------------------------------------------------------------------|
| pSrcVec | Входной вектор.                                                     |
| nAddVal | Константа для суммирования.                                         |
| nSize   | Размер векторов в элементах.                                        |
| nShift  | Определяет на сколько позиций нужно сдвинуть биты элемента вектора. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.221 nmppsDisplaceBits

Непрерывное смещение битов внутри бинарного массива в сторону конца массива

Функция смещает биты внутри бинарного массива на несколько позиций (nBits) в сторону конца массива. Внутри 64р. слова младшие биты сдвигаются на старшие позиции того же слова, а старшие биты перемещаются в младшие позиции следующего 64р. слова. Освободившееся место в первом 64р. слове заполняется старшими битами 64р. слова с адреса pnBits. Сдвинутые биты сохраняются в массиве pDst. Пример сдвига на 8 бит :

### Функции

- void **nmppsFwdShiftBitstream** (const **nm64u** \*pSrcVec, **nm64u** \*pDstVec, **nm64u** \*pnBits, int nBits, int nSize)

### 6.221.1 Подробное описание

Непрерывное смещение битов внутри бинарного массива в сторону конца массива

Функция смещает биты внутри бинарного массива на несколько позиций (nBits) в сторону конца массива. Внутри 64р. слова младшие биты сдвигаются на старшие позиции того же слова, а старшие биты перемещаются в младшие позиции следующего 64р. слова. Освободившееся место в первом 64р. слове заполняется старшими битами 64р. слова с адреса pnBits. Сдвинутые биты сохраняются в массиве pDst. Пример сдвига на 8 бит :

```
pnBits=[AB00000000000000]
pSrcVec=[0807060504030201][FF0F0E0D0C0B0A09]
pDstVec=[07060504030201AB][0F0E0D0C0B0A0908]
```

Последние 8 бит массива pDstVec будут потеряны. Если же указатель pBits установить на последнее 64р. слово в результате получится циклическое перемещение бит.

### Аргументы

|         |                                                                                                                           |
|---------|---------------------------------------------------------------------------------------------------------------------------|
| pSrcVec | Входной вектор.                                                                                                           |
| nSize   | Размер векторов в 64р. элементах.                                                                                         |
| pnBits  | Указатель на 64р-слово, старшие биты которого записываются на освобождающуюся при сдвиге младшую часть первого 64р. слова |
| nBits   | Кол-во позиций на которое происходит смещение бит :nBits=[2,4,6....62].                                                   |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|



Возвращает

void

## 6.221.2 Функции

### 6.221.2.1 nmppsFwdShiftBitstream()

```
void nmppsFwdShiftBitstream (
    const nm64u * pSrcVec,
    nm64u * pDstVec,
    nm64u * pnBits,
    int nBits,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | pnBits | nBits | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|-------|------------|
| im0     | im0     | im2    | 2     | 8     | 440   | 55.00      |
| im0     | im0     | im2    | 2     | 8     | 440   | 55.00      |
| im0     | im0     | im2    | 2     | 128   | 684   | 5.34       |
| im0     | im0     | im2    | 2     | 128   | 684   | 5.34       |
| im0     | im0     | im2    | 2     | 1024  | 3512  | 3.42       |
| im0     | im0     | im2    | 2     | 1024  | 3512  | 3.42       |
| im0     | im0     | im2    | 2     | 2048  | 6744  | 3.29       |
| im0     | im0     | im2    | 2     | 2048  | 6744  | 3.29       |
| im0     | im0     | im2    | 8     | 8     | 442   | 55.25      |
| im0     | im0     | im2    | 8     | 8     | 442   | 55.25      |
| im0     | im0     | im2    | 8     | 128   | 686   | 5.35       |
| im0     | im0     | im2    | 8     | 128   | 686   | 5.35       |
| im0     | im0     | im2    | 8     | 1024  | 3514  | 3.43       |
| im0     | im0     | im2    | 8     | 1024  | 3514  | 3.43       |
| im0     | im0     | im2    | 8     | 2048  | 6746  | 3.29       |
| im0     | im0     | im2    | 8     | 2048  | 6746  | 3.29       |
| im0     | im0     | im2    | 18    | 8     | 442   | 55.25      |
| im0     | im0     | im2    | 18    | 8     | 442   | 55.25      |
| im0     | im0     | im2    | 18    | 128   | 686   | 5.35       |
| im0     | im0     | im2    | 18    | 128   | 686   | 5.35       |
| im0     | im0     | im2    | 18    | 1024  | 3514  | 3.43       |
| im0     | im0     | im2    | 18    | 1024  | 3514  | 3.43       |
| im0     | im0     | im2    | 18    | 2048  | 6746  | 3.29       |
| im0     | im0     | im2    | 18    | 2048  | 6746  | 3.29       |
| im0     | im1     | im2    | 2     | 8     | 449   | 56.12      |
| im0     | im1     | im2    | 2     | 8     | 449   | 56.12      |
| im0     | im1     | im2    | 2     | 128   | 696   | 5.43       |
| im0     | im1     | im2    | 2     | 128   | 696   | 5.43       |
| im0     | im1     | im2    | 2     | 1024  | 3608  | 3.52       |
| im0     | im1     | im2    | 2     | 1024  | 3608  | 3.52       |

| pSrcVec | pDstVec | pnBits | nBits | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|-------|------------|
| im0     | im1     | im2    | 2     | 2048  | 6936  | 3.38       |
| im0     | im1     | im2    | 2     | 2048  | 6936  | 3.38       |
| im0     | im1     | im2    | 8     | 8     | 451   | 56.37      |
| im0     | im1     | im2    | 8     | 8     | 451   | 56.37      |
| im0     | im1     | im2    | 8     | 128   | 698   | 5.45       |
| im0     | im1     | im2    | 8     | 128   | 698   | 5.45       |
| im0     | im1     | im2    | 8     | 1024  | 3610  | 3.52       |
| im0     | im1     | im2    | 8     | 1024  | 3610  | 3.52       |
| im0     | im1     | im2    | 8     | 2048  | 6938  | 3.38       |
| im0     | im1     | im2    | 8     | 2048  | 6938  | 3.38       |
| im0     | im1     | im2    | 18    | 8     | 451   | 56.37      |
| im0     | im1     | im2    | 18    | 8     | 451   | 56.37      |
| im0     | im1     | im2    | 18    | 128   | 698   | 5.45       |
| im0     | im1     | im2    | 18    | 128   | 698   | 5.45       |
| im0     | im1     | im2    | 18    | 1024  | 3610  | 3.52       |
| im0     | im1     | im2    | 18    | 1024  | 3610  | 3.52       |
| im0     | im1     | im2    | 18    | 2048  | 6938  | 3.38       |
| im0     | im1     | im2    | 18    | 2048  | 6938  | 3.38       |
| im1     | im0     | im2    | 2     | 8     | 447   | 55.87      |
| im1     | im0     | im2    | 2     | 8     | 447   | 55.87      |
| im1     | im0     | im2    | 2     | 128   | 629   | 4.91       |
| im1     | im0     | im2    | 2     | 128   | 629   | 4.91       |
| im1     | im0     | im2    | 2     | 1024  | 2813  | 2.74       |
| im1     | im0     | im2    | 2     | 1024  | 2813  | 2.74       |
| im1     | im0     | im2    | 2     | 2048  | 5309  | 2.59       |
| im1     | im0     | im2    | 2     | 2048  | 5309  | 2.59       |
| im1     | im0     | im2    | 8     | 8     | 449   | 56.12      |
| im1     | im0     | im2    | 8     | 8     | 449   | 56.12      |
| im1     | im0     | im2    | 8     | 128   | 631   | 4.92       |
| im1     | im0     | im2    | 8     | 128   | 631   | 4.92       |
| im1     | im0     | im2    | 8     | 1024  | 2815  | 2.74       |
| im1     | im0     | im2    | 8     | 1024  | 2815  | 2.74       |
| im1     | im0     | im2    | 8     | 2048  | 5311  | 2.59       |
| im1     | im0     | im2    | 8     | 2048  | 5311  | 2.59       |
| im1     | im0     | im2    | 18    | 8     | 449   | 56.12      |
| im1     | im0     | im2    | 18    | 8     | 449   | 56.12      |
| im1     | im0     | im2    | 18    | 128   | 631   | 4.92       |
| im1     | im0     | im2    | 18    | 128   | 631   | 4.92       |
| im1     | im0     | im2    | 18    | 1024  | 2815  | 2.74       |
| im1     | im0     | im2    | 18    | 1024  | 2815  | 2.74       |
| im1     | im0     | im2    | 18    | 2048  | 5311  | 2.59       |
| im1     | im0     | im2    | 18    | 2048  | 5311  | 2.59       |
| im1     | im1     | im2    | 2     | 8     | 463   | 57.87      |
| im1     | im1     | im2    | 2     | 8     | 463   | 57.87      |
| im1     | im1     | im2    | 2     | 128   | 752   | 5.87       |
| im1     | im1     | im2    | 2     | 128   | 752   | 5.87       |
| im1     | im1     | im2    | 2     | 1024  | 4000  | 3.90       |
| im1     | im1     | im2    | 2     | 1024  | 4000  | 3.90       |
| im1     | im1     | im2    | 2     | 2048  | 7712  | 3.76       |
| im1     | im1     | im2    | 2     | 2048  | 7712  | 3.76       |
| im1     | im1     | im2    | 8     | 8     | 465   | 58.12      |
| im1     | im1     | im2    | 8     | 8     | 465   | 58.12      |

| pSrcVec | pDstVec | pnBits | nBits | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|-------|------------|
| im1     | im1     | im2    | 8     | 128   | 754   | 5.89       |
| im1     | im1     | im2    | 8     | 128   | 754   | 5.89       |
| im1     | im1     | im2    | 8     | 1024  | 4002  | 3.90       |
| im1     | im1     | im2    | 8     | 1024  | 4002  | 3.90       |
| im1     | im1     | im2    | 8     | 2048  | 7714  | 3.76       |
| im1     | im1     | im2    | 8     | 2048  | 7714  | 3.76       |
| im1     | im1     | im2    | 18    | 8     | 465   | 58.12      |
| im1     | im1     | im2    | 18    | 8     | 465   | 58.12      |
| im1     | im1     | im2    | 18    | 128   | 754   | 5.89       |
| im1     | im1     | im2    | 18    | 128   | 754   | 5.89       |
| im1     | im1     | im2    | 18    | 1024  | 4002  | 3.90       |
| im1     | im1     | im2    | 18    | 1024  | 4002  | 3.90       |
| im1     | im1     | im2    | 18    | 2048  | 7714  | 3.76       |
| im1     | im1     | im2    | 18    | 2048  | 7714  | 3.76       |

The best configuration:

| pSrcVec | pDstVec | pnBits | nBits | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|-------|------------|
| im0     | im0     | im2    | 2     | 8     | 440   | 55.00      |

The worst configuration:

| pSrcVec | pDstVec | pnBits | nBits | nSize | ticks | ticks/elem |
|---------|---------|--------|-------|-------|-------|------------|
| im1     | im1     | im2    | 8     | 2048  | 7714  | 3.76       |

## 6.222 nmppsSet-инициализация

Функция инициализации элементов массива постоянным значением.

Функции

- void `nmppsSet_8s` (`nm8s` \*pVec, `int8b` nVal, int nSize)
- void `nmppsSet_16s` (`nm16s` \*pVec, `int16b` nVal, int nSize)
- void `nmppsSet_32s` (`nm32s` \*pVec, `int32b` nVal, int nSize)
- void `nmppsSet_64sp` (`nm64s` \*pVec, `int64b` \*nVal, int nSize)
- `__INLINE__` void `nmppsSet_64s` (`nm64s` \*pVec, `int64b` nVal, int nSize)
- `__INLINE__` void `nmppsSet_8u` (`nm8u` \*pVec, `uint8b` nVal, int nSize)
- `__INLINE__` void `nmppsSet_16u` (`nm16u` \*pVec, `uint16b` nVal, int nSize)
- `__INLINE__` void `nmppsSet_32u` (`nm32u` \*pVec, `uint32b` nVal, int nSize)
- `__INLINE__` void `nmppsSet_64u` (`nm64u` \*pVec, `uint64b` nVal, int nSize)
- `__INLINE__` void `nmppsSet_64up` (`nm64u` \*pVec, `uint64b` \*nVal, int nSize)

### 6.222.1 Подробное описание

Функция инициализации элементов массива постоянным значением.

$$pVec[i] = nVal,$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|       |                                                                                        |
|-------|----------------------------------------------------------------------------------------|
| nSize | Размер вектора в элементах.                                                            |
| nVal  | Константа. Диапазон значений nVal должен соответствовать типу результирующего вектора. |

Возвращаемые значения

|      |                        |
|------|------------------------|
| pVec | Результирующий вектор. |
|------|------------------------|

Возвращает

void

### 6.222.2 Функции

## 6.222.2.1 nmppsSet\_16s()

```
void nmppsSet_16s (
    nm16s * pVec,
    int16b nVal,
    int nSize )
```

Performance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 609   | 0.29       |
| im0  | 555  | 2048  | 609   | 0.29       |
| im1  | 555  | 2048  | 689   | 0.33       |
| im1  | 555  | 2048  | 689   | 0.33       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 609   | 0.29       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 689   | 0.33       |

Performance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 116   | 14.50      |
| im0  | 555  | 8     | 116   | 14.50      |
| im0  | 555  | 128   | 147   | 1.14       |
| im0  | 555  | 128   | 147   | 1.14       |
| im0  | 555  | 1024  | 353   | 0.34       |
| im0  | 555  | 1024  | 353   | 0.34       |
| im0  | 555  | 2048  | 609   | 0.29       |
| im0  | 555  | 2048  | 609   | 0.29       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 116   | 14.50      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 609   | 0.29       |

## 6.222.2.2 nmppsSet\_16u()

```
__INLINE__ void nmppsSet_16u (
    nm16u * pVec,
    uint16b nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 662   | 0.32       |
| im0  | 555  | 2048  | 662   | 0.32       |
| im1  | 555  | 2048  | 742   | 0.36       |
| im1  | 555  | 2048  | 742   | 0.36       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 662   | 0.32       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 742   | 0.36       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 169   | 21.12      |
| im0  | 555  | 8     | 169   | 21.12      |
| im0  | 555  | 128   | 200   | 1.56       |
| im0  | 555  | 128   | 200   | 1.56       |
| im0  | 555  | 1024  | 406   | 0.39       |
| im0  | 555  | 1024  | 406   | 0.39       |
| im0  | 555  | 2048  | 662   | 0.32       |
| im0  | 555  | 2048  | 662   | 0.32       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 169   | 21.12      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 662   | 0.32       |

См. определение в файле vInit.h строка 86

## 6.222.2.3 nmppsSet\_32s()

```
void nmppsSet_32s (
    nm32s * pVec,
    int32b nVal,
    int nSize )
```

Performace table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1119  | 0.54       |
| im0  | 555  | 2048  | 1119  | 0.54       |
| im1  | 555  | 2048  | 1231  | 0.60       |
| im1  | 555  | 2048  | 1231  | 0.60       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1119  | 0.54       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 1231  | 0.60       |

Performace table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 116   | 14.50      |
| im0  | 555  | 8     | 116   | 14.50      |
| im0  | 555  | 128   | 156   | 1.21       |
| im0  | 555  | 128   | 156   | 1.21       |
| im0  | 555  | 1024  | 607   | 0.59       |
| im0  | 555  | 1024  | 607   | 0.59       |
| im0  | 555  | 2048  | 1119  | 0.54       |
| im0  | 555  | 2048  | 1119  | 0.54       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 116   | 14.50      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1119  | 0.54       |

## 6.222.2.4 nmppsSet\_32u()

```
__INLINE__ void nmppsSet_32u (
    nm32u * pVec,
    uint32b nVal,
    int nSize )
```

Performace table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1172  | 0.57       |
| im0  | 555  | 2048  | 1172  | 0.57       |
| im1  | 555  | 2048  | 1284  | 0.62       |
| im1  | 555  | 2048  | 1284  | 0.62       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1172  | 0.57       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 1284  | 0.62       |

Performace table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 169   | 21.12      |
| im0  | 555  | 8     | 169   | 21.12      |
| im0  | 555  | 128   | 209   | 1.63       |
| im0  | 555  | 128   | 209   | 1.63       |
| im0  | 555  | 1024  | 660   | 0.64       |
| im0  | 555  | 1024  | 660   | 0.64       |
| im0  | 555  | 2048  | 1172  | 0.57       |
| im0  | 555  | 2048  | 1172  | 0.57       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 169   | 21.12      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 1172  | 0.57       |



См. определение в файле vInit.h строка 87

### 6.222.2.5 nmppsSet\_64s()

```
__INLINE__ void nmppsSet_64s (
    nm64s * pVec,
    int64b nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2189  | 1.06       |
| im0  | 555  | 2048  | 2189  | 1.06       |
| im1  | 555  | 2048  | 2365  | 1.15       |
| im1  | 555  | 2048  | 2365  | 1.15       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2189  | 1.06       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 2365  | 1.15       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 172   | 21.50      |
| im0  | 555  | 8     | 172   | 21.50      |
| im0  | 555  | 128   | 271   | 2.11       |
| im0  | 555  | 128   | 271   | 2.11       |
| im0  | 555  | 1024  | 1165  | 1.13       |
| im0  | 555  | 1024  | 1165  | 1.13       |
| im0  | 555  | 2048  | 2189  | 1.06       |
| im0  | 555  | 2048  | 2189  | 1.06       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 172   | 21.50      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2189  | 1.06       |

См. определение в файле vInit.h строка 83

#### 6.222.2.6 nmppsSet\_64sp()

```
void nmppsSet_64sp (
    nm64s * pVec,
    int64b * nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2143  | 1.04       |
| im0  | 555  | 2048  | 2143  | 1.04       |
| im1  | 555  | 2048  | 2276  | 1.11       |
| im1  | 555  | 2048  | 2276  | 1.11       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2143  | 1.04       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 2276  | 1.11       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 105   | 13.12      |
| im0  | 555  | 8     | 105   | 13.12      |
| im0  | 555  | 128   | 223   | 1.74       |
| im0  | 555  | 128   | 223   | 1.74       |
| im0  | 555  | 1024  | 1119  | 1.09       |
| im0  | 555  | 1024  | 1119  | 1.09       |
| im0  | 555  | 2048  | 2143  | 1.04       |
| im0  | 555  | 2048  | 2143  | 1.04       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 105   | 13.12      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2143  | 1.04       |

#### 6.222.2.7 nmppsSet\_64u()

```
__INLINE__ void nmppsSet_64u (
    nm64u * pVec,
    uint64b nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2249  | 1.09       |
| im0  | 555  | 2048  | 2249  | 1.09       |
| im1  | 555  | 2048  | 2425  | 1.18       |
| im1  | 555  | 2048  | 2425  | 1.18       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2249  | 1.09       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 2425  | 1.18       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 232   | 29.00      |
| im0  | 555  | 8     | 232   | 29.00      |
| im0  | 555  | 128   | 331   | 2.58       |
| im0  | 555  | 128   | 331   | 2.58       |
| im0  | 555  | 1024  | 1225  | 1.19       |
| im0  | 555  | 1024  | 1225  | 1.19       |
| im0  | 555  | 2048  | 2249  | 1.09       |
| im0  | 555  | 2048  | 2249  | 1.09       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 232   | 29.00      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2249  | 1.09       |

См. определение в файле vInit.h строка 88

#### 6.222.2.8 nmppsSet\_64up()

```
__INLINE__ void nmppsSet_64up (
    nm64u * pVec,
    uint64b * nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2192  | 1.07       |
| im0  | 555  | 2048  | 2192  | 1.07       |
| im1  | 555  | 2048  | 2325  | 1.13       |
| im1  | 555  | 2048  | 2325  | 1.13       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2192  | 1.07       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 2325  | 1.13       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 154   | 19.25      |
| im0  | 555  | 8     | 154   | 19.25      |
| im0  | 555  | 128   | 272   | 2.12       |
| im0  | 555  | 128   | 272   | 2.12       |
| im0  | 555  | 1024  | 1168  | 1.14       |
| im0  | 555  | 1024  | 1168  | 1.14       |
| im0  | 555  | 2048  | 2192  | 1.07       |
| im0  | 555  | 2048  | 2192  | 1.07       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 154   | 19.25      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 2192  | 1.07       |

См. определение в файле vInit.h строка 89

#### 6.222.2.9 nmppsSet\_8s()

```
void nmppsSet_8s (
    nm8s * pVec,
    int8b nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 355   | 0.17       |
| im0  | 555  | 2048  | 355   | 0.17       |
| im1  | 555  | 2048  | 419   | 0.20       |
| im1  | 555  | 2048  | 419   | 0.20       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 355   | 0.17       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 419   | 0.20       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 117   | 14.62      |
| im0  | 555  | 8     | 117   | 14.62      |
| im0  | 555  | 128   | 132   | 1.03       |
| im0  | 555  | 128   | 132   | 1.03       |
| im0  | 555  | 1024  | 228   | 0.22       |
| im0  | 555  | 1024  | 228   | 0.22       |
| im0  | 555  | 2048  | 355   | 0.17       |
| im0  | 555  | 2048  | 355   | 0.17       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 117   | 14.62      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 355   | 0.17       |

#### 6.222.2.10 nmppsSet\_8u()

```
__INLINE__ void nmppsSet_8u (
    nm8u * pVec,
    uint8b nVal,
    int nSize )
```

Perfomance table 0

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 408   | 0.19       |
| im0  | 555  | 2048  | 408   | 0.19       |
| im1  | 555  | 2048  | 472   | 0.23       |
| im1  | 555  | 2048  | 472   | 0.23       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 408   | 0.19       |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im1  | 555  | 2048  | 472   | 0.23       |

Perfomance table 1

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 170   | 21.25      |
| im0  | 555  | 8     | 170   | 21.25      |
| im0  | 555  | 128   | 185   | 1.44       |
| im0  | 555  | 128   | 185   | 1.44       |
| im0  | 555  | 1024  | 281   | 0.27       |
| im0  | 555  | 1024  | 281   | 0.27       |
| im0  | 555  | 2048  | 408   | 0.19       |
| im0  | 555  | 2048  | 408   | 0.19       |

The best configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 8     | 170   | 21.25      |

The worst configuration:

| pVec | nVal | nSize | ticks | ticks/elem |
|------|------|-------|-------|------------|
| im0  | 555  | 2048  | 408   | 0.19       |

См. определение в файле vInit.h строка 85

## 6.223 nmppsRandUniform

Инициализация массива случайными числами.

### Функции

- void [nmppsRandUniform\\_64s](#) ([nm64s](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_8s](#) ([nm8s](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_16s](#) ([nm16s](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_32s](#) ([nm32s](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_8u](#) ([nm8u](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_16u](#) ([nm16u](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_32u](#) ([nm32u](#) \*pDstVec, int nSize)
- `__INLINE__` void [nmppsRandUniform\\_64u](#) ([nm64u](#) \*pDstVec, int nSize)

### 6.223.1 Подробное описание

Инициализация массива случайными числами.

#### Аргументы

|       |                 |
|-------|-----------------|
| nSize | Размер вектора. |
|-------|-----------------|

#### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

#### Возвращает

void

### 6.223.2 Функции

#### 6.223.2.1 nmppsRandUniform\_16s()

```
__INLINE__ void nmppsRandUniform_16s (
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0



| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |
| im1     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |
| im2     | 8     | 169   | 21.12      |
| im2     | 128   | 375   | 2.92       |
| im2     | 128   | 375   | 2.92       |
| im2     | 1024  | 1943  | 1.89       |
| im2     | 1024  | 1943  | 1.89       |
| im2     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 3735  | 1.82       |

См. определение в файле vInit.h строка 135

#### 6.223.2.2 nmppsRandUniform\_16u()

```
__INLINE__ void nmppsRandUniform_16u (
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |
| im1     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 3735  | 1.82       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |
| im2     | 8     | 169   | 21.12      |
| im2     | 128   | 375   | 2.92       |
| im2     | 128   | 375   | 2.92       |
| im2     | 1024  | 1943  | 1.89       |
| im2     | 1024  | 1943  | 1.89       |
| im2     | 2048  | 3735  | 1.82       |
| im2     | 2048  | 3735  | 1.82       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 3735  | 1.82       |

См. определение в файле vInit.h строка 138

### 6.223.2.3 nmppsRandUniform\_32s()

```
__INLINE__ void nmppsRandUniform_32s (
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |
| im1     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 179   | 22.37      |
| im2     | 8     | 179   | 22.37      |
| im2     | 128   | 599   | 4.67       |
| im2     | 128   | 599   | 4.67       |
| im2     | 1024  | 3735  | 3.64       |
| im2     | 1024  | 3735  | 3.64       |
| im2     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 179   | 22.37      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 7319  | 3.57       |

См. определение в файле vInit.h строка 136

#### 6.223.2.4 nmppsRandUniform\_32u()

```
__INLINE__ void nmppsRandUniform_32u (
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |
| im1     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 7319  | 3.57       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 179   | 22.37      |
| im2     | 8     | 179   | 22.37      |
| im2     | 128   | 599   | 4.67       |
| im2     | 128   | 599   | 4.67       |
| im2     | 1024  | 3735  | 3.64       |
| im2     | 1024  | 3735  | 3.64       |
| im2     | 2048  | 7319  | 3.57       |
| im2     | 2048  | 7319  | 3.57       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 179   | 22.37      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 7319  | 3.57       |

См. определение в файле vInit.h строка 139

#### 6.223.2.5 nmppsRandUniform\_64s()

```
void nmppsRandUniform_64s (
    nm64s * pDstVec,
    int nSize )
```

/ru Инициализация массива 32-разрядными случайными числами. /en Random initialization of 32-bit buffer /~

Performance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14409 | 7.03       |
| im1     | 2048  | 14409 | 7.03       |
| im2     | 2048  | 14409 | 7.03       |
| im2     | 2048  | 14409 | 7.03       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14409 | 7.03       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14409 | 7.03       |

Performance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 129   | 16.12      |
| im2     | 8     | 129   | 16.12      |
| im2     | 128   | 969   | 7.57       |
| im2     | 128   | 969   | 7.57       |
| im2     | 1024  | 7241  | 7.07       |
| im2     | 1024  | 7241  | 7.07       |
| im2     | 2048  | 14409 | 7.03       |
| im2     | 2048  | 14409 | 7.03       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 129   | 16.12      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 14409 | 7.03       |

#### 6.223.2.6 nmppsRandUniform\_64u()

```
__INLINE__ void nmppsRandUniform_64u (
```

```
nm64u * pDstVec,
int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14449 | 7.05       |
| im1     | 2048  | 14449 | 7.05       |
| im2     | 2048  | 14449 | 7.05       |
| im2     | 2048  | 14449 | 7.05       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14449 | 7.05       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 14449 | 7.05       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |
| im2     | 8     | 169   | 21.12      |
| im2     | 128   | 1009  | 7.88       |
| im2     | 128   | 1009  | 7.88       |
| im2     | 1024  | 7281  | 7.11       |
| im2     | 1024  | 7281  | 7.11       |
| im2     | 2048  | 14449 | 7.05       |
| im2     | 2048  | 14449 | 7.05       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 169   | 21.12      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 14449 | 7.05       |

См. определение в файле vInit.h строка 140

## 6.223.2.7 nmppsRandUniform\_8s()

```
__INLINE__ void nmppsRandUniform_8s (
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |
| im1     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 167   | 20.87      |
| im2     | 8     | 167   | 20.87      |
| im2     | 128   | 263   | 2.05       |
| im2     | 128   | 263   | 2.05       |
| im2     | 1024  | 1047  | 1.02       |
| im2     | 1024  | 1047  | 1.02       |
| im2     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 167   | 20.87      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 1943  | 0.94       |

См. определение в файле vInit.h строка 134

## 6.223.2.8 nmppsRandUniform\_8u()

```
__INLINE__ void nmppsRandUniform_8u (
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |
| im1     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 2048  | 1943  | 0.94       |

Perfomance table 1

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 167   | 20.87      |
| im2     | 8     | 167   | 20.87      |
| im2     | 128   | 263   | 2.05       |
| im2     | 128   | 263   | 2.05       |
| im2     | 1024  | 1047  | 1.02       |
| im2     | 1024  | 1047  | 1.02       |
| im2     | 2048  | 1943  | 0.94       |
| im2     | 2048  | 1943  | 0.94       |

The best configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 8     | 167   | 20.87      |

The worst configuration:

| pDstVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im2     | 2048  | 1943  | 0.94       |



См. определение в файле vInit.h строка 137

## 6.224 nmppsRandUniform

Инициализация массива случайными числами.

### Функции

- void [nmppsRandUniform\\_64f](#) (nm64f \*pDstVec, int nSize, double low, double hi)
- void [nmppsRand\\_32f](#) (nm32f \*pDstVec, int nSize, float low, float hi)
- void [nmppsRandUniform\\_32f\\_integer](#) (nm32f \*pDstVec, int nSize, int hi, int low)

### 6.224.1 Подробное описание

Инициализация массива случайными числами.

/ru Инициализация массива 64-разрядными случайными числами. /en Random initialization of 64-bit buffer /~

#### Аргументы

|       |                 |
|-------|-----------------|
| nSize | Размер вектора. |
|-------|-----------------|

#### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

#### Возвращает

void

### 6.224.2 Функции

#### 6.224.2.1 nmppsRand\_32f()

```
void nmppsRand_32f (
    nm32f * pDstVec,
    int nSize,
    float low,
    float hi )
```

Perfomance table 0

| pDstVec | nSize | low  | hi   | ticks   | ticks/elem |
|---------|-------|------|------|---------|------------|
| im1     | 2048  | 2048 | 2048 | 1086219 | 530.38     |
| im1     | 2048  | 2048 | 2048 | 1086219 | 530.38     |
| im2     | 2048  | 2048 | 2048 | 1096796 | 535.54     |
| im2     | 2048  | 2048 | 2048 | 1096796 | 535.54     |

The best configuration:

pDstVec | nSize | low | hi | ticks | ticks/elem —|---|---|---|---|— 0.54

The worst configuration:

| pDstVec | nSize | low  | hi   | ticks   | ticks/elem |
|---------|-------|------|------|---------|------------|
| im2     | 2048  | 2048 | 2048 | 1096796 | 535.54     |

Perfomance table 1

| pDstVec | nSize | low  | hi   | ticks   | ticks/elem |
|---------|-------|------|------|---------|------------|
| im2     | 8     | 2048 | 2048 | 4691    | 2.29       |
| im2     | 8     | 2048 | 2048 | 4691    | 2.29       |
| im2     | 128   | 2048 | 2048 | 67929   | 33.16      |
| im2     | 128   | 2048 | 2048 | 67929   | 33.16      |
| im2     | 1024  | 2048 | 2048 | 542887  | 265.08     |
| im2     | 1024  | 2048 | 2048 | 542887  | 265.08     |
| im2     | 2048  | 2048 | 2048 | 1085492 | 530.02     |
| im2     | 2048  | 2048 | 2048 | 1085492 | 530.02     |

The best configuration:

| pDstVec | nSize | low  | hi   | ticks | ticks/elem |
|---------|-------|------|------|-------|------------|
| im2     | 8     | 2048 | 2048 | 4691  | 2.29       |

The worst configuration:

| pDstVec | nSize | low  | hi   | ticks   | ticks/elem |
|---------|-------|------|------|---------|------------|
| im2     | 2048  | 2048 | 2048 | 1085492 | 530.02     |

### 6.224.2.2 nmppsRandUniform\_32f\_integer()

```
void nmppsRandUniform_32f_integer (
    nm32f * pDstVec,
    int nSize,
    int hi,
    int low )
```

Perfomance table 0

| pDst Vec | nSize | low  | hi   | ticks   | ticks/elem |
|----------|-------|------|------|---------|------------|
| im1      | 2048  | 2048 | 2048 | 1085727 | 530.14     |
| im1      | 2048  | 2048 | 2048 | 1085727 | 530.14     |
| im2      | 2048  | 2048 | 2048 | 1085727 | 530.14     |
| im2      | 2048  | 2048 | 2048 | 1085727 | 530.14     |

The best configuration:

pDstVec | nSize | low | hi | ticks | ticks/elem —|---|---|---|---|— imim1 imim1 im1 | im2 | im1 | im2  
| im3 | im4 | im5 | 2048 | 1108 | 0.54

The worst configuration:

| pDst Vec | nSize | low  | hi   | ticks   | ticks/elem |
|----------|-------|------|------|---------|------------|
| im1      | 2048  | 2048 | 2048 | 1085727 | 530.14     |

Perfomance table 1

| pDst Vec | nSize | low  | hi   | ticks   | ticks/elem |
|----------|-------|------|------|---------|------------|
| im2      | 8     | 2048 | 2048 | 4527    | 2.21       |
| im2      | 8     | 2048 | 2048 | 4527    | 2.21       |
| im2      | 128   | 2048 | 2048 | 79426   | 38.78      |
| im2      | 128   | 2048 | 2048 | 79426   | 38.78      |
| im2      | 1024  | 2048 | 2048 | 543007  | 265.14     |
| im2      | 1024  | 2048 | 2048 | 543007  | 265.14     |
| im2      | 2048  | 2048 | 2048 | 1085727 | 530.14     |
| im2      | 2048  | 2048 | 2048 | 1085727 | 530.14     |

The best configuration:

| pDst Vec | nSize | low  | hi   | ticks | ticks/elem |
|----------|-------|------|------|-------|------------|
| im2      | 8     | 2048 | 2048 | 4527  | 2.21       |

The worst configuration:

| pDst Vec | nSize | low  | hi   | ticks   | ticks/elem |
|----------|-------|------|------|---------|------------|
| im2      | 2048  | 2048 | 2048 | 1085727 | 530.14     |

### 6.224.2.3 nmppsRandUniform\_64f()

```
void nmppsRandUniform_64f (
    nm64f * pDstVec,
    int nSize,
    double low,
    double hi )
```

Perfomance table 0

| pDstVec | nSize | low  | hi   | ticks  | ticks/elem |
|---------|-------|------|------|--------|------------|
| im1     | 2048  | 2048 | 2048 | 813454 | 397.19     |
| im1     | 2048  | 2048 | 2048 | 813454 | 397.19     |
| im2     | 2048  | 2048 | 2048 | 813454 | 397.19     |
| im2     | 2048  | 2048 | 2048 | 813454 | 397.19     |

The best configuration:

| pDstVec | nSize | low  | hi   | ticks  | ticks/elem |
|---------|-------|------|------|--------|------------|
| im1     | 2048  | 2048 | 2048 | 813454 | 397.19     |

The worst configuration:

| pDstVec | nSize | low  | hi   | ticks  | ticks/elem |
|---------|-------|------|------|--------|------------|
| im1     | 2048  | 2048 | 2048 | 813454 | 397.19     |

Perfomance table 1

| pDstVec | nSize | low  | hi   | ticks  | ticks/elem |
|---------|-------|------|------|--------|------------|
| im2     | 8     | 2048 | 2048 | 3574   | 1.74       |
| im2     | 8     | 2048 | 2048 | 3574   | 1.74       |
| im2     | 128   | 2048 | 2048 | 51214  | 25.00      |
| im2     | 128   | 2048 | 2048 | 51214  | 25.00      |
| im2     | 1024  | 2048 | 2048 | 406926 | 198.69     |
| im2     | 1024  | 2048 | 2048 | 406926 | 198.69     |
| im2     | 2048  | 2048 | 2048 | 813454 | 397.19     |
| im2     | 2048  | 2048 | 2048 | 813454 | 397.19     |

The best configuration:

| pDstVec | nSize | low  | hi   | ticks | ticks/elem |
|---------|-------|------|------|-------|------------|
| im2     | 8     | 2048 | 2048 | 3574  | 1.74       |

The worst configuration:

| pDstVec | nSize | low  | hi   | ticks  | ticks/elem |
|---------|-------|------|------|--------|------------|
| im2     | 2048  | 2048 | 2048 | 813454 | 397.19     |

## 6.225 nmppsRandUniform\_\_

Генерация случайного числа с равномерным распределением.

### Функции

- `int nmppsRandUniform2_32s (int nMin, int nMax, int nDivisible)`
- `int nmppsRandUniform3_32s (int nMin, int nMax)`
- `int nmppsRandUniform ()`

### 6.225.1 Подробное описание

Генерация случайного числа с равномерным распределением.

#### Аргументы

|            |                                                   |
|------------|---------------------------------------------------|
| nMin       | Минимальное возможное значение случайного числа.  |
| nMax       | Максимальное возможное значение случайного числа. |
| nDivisible | Значение, которому будет кратно случайное число.  |

#### Возвращает

`int` Случайное число в диапазоне либо `[nMin, nMax]`. Для функции без параметров данный диапазон `[-231; 231-1]`.

## 6.226 nmppsRamp\_

Инициализация массива элементами арифметической прогрессии.

Функции

- void `nmppsRamp_8s` (`nm8s` \*pVec, `int8b` nOffset, `int8b` nSlope, int nSize)
- void `nmppsRamp_16s` (`nm16s` \*pVec, `int16b` nOffset, `int16b` nSlope, int nSize)
- void `nmppsRamp_32s` (`nm32s` \*pVec, `int32b` nOffset, `int32b` nSlope, int nSize)
- void `nmppsRamp_64s` (`nm64s` \*pVec, `int64b` nOffset, `int64b` nSlope, int nSize)

### 6.226.1 Подробное описание

Инициализация массива элементами арифметической прогрессии.

$$pVec[i] = nOffset + nSlope \cdot i$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                                        |
|---------|----------------------------------------|
| nOffset | Первый член арифметической прогрессии. |
| nSlope  | Разность арифметической прогрессии.    |
| nSize   | Размер вектора.                        |

Возвращаемые значения

|      |                        |
|------|------------------------|
| pVec | Результирующий массив. |
|------|------------------------|

Возвращает

void

### 6.226.2 Функции

## 6.226.2.1 nmppsRamp\_16s()

```
void nmppsRamp_16s (
    nm16s * pVec,
    int16b nOffset,
    int16b nSlope,
    int nSize )
```

Performance table 0

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im1  | 10      | 2      | 2048  | 147562 | 72.05      |
| im1  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

Performance table 1

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |

Performance table 2

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |



| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |

Perfomance table 3

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 8     | 682    | 85.25      |
| im0  | 10      | 2      | 8     | 682    | 85.25      |
| im0  | 10      | 2      | 128   | 9322   | 72.82      |
| im0  | 10      | 2      | 128   | 9322   | 72.82      |
| im0  | 10      | 2      | 1024  | 73834  | 72.10      |
| im0  | 10      | 2      | 1024  | 73834  | 72.10      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 682   | 85.25      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

### 6.226.2.2 nmppsRamp\_32s()

```
void nmppsRamp_32s (
    nm32s * pVec,
    int32b nOffset,
    int32b nSlope,
    int nSize )
```

Perfomance table 0

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |
| im1  | 10      | 2      | 2048  | 36963 | 18.04      |
| im1  | 10      | 2      | 2048  | 36963 | 18.04      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

Perfomance table 1

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 36963 | 18.04      |
| im0  | -10     | 2      | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 36963 | 18.04      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 36963 | 18.04      |

Perfomance table 2

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 36963 | 18.04      |
| im0  | 10      | -2     | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 36963 | 18.04      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 36963 | 18.04      |

Perfomance table 3

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 243   | 30.37      |
| im0  | 10      | 2      | 8     | 243   | 30.37      |
| im0  | 10      | 2      | 128   | 2403  | 18.77      |
| im0  | 10      | 2      | 128   | 2403  | 18.77      |
| im0  | 10      | 2      | 1024  | 18531 | 18.09      |
| im0  | 10      | 2      | 1024  | 18531 | 18.09      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 243   | 30.37      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 36963 | 18.04      |

### 6.226.2.3 nmppsRamp\_64s()

```
void nmppsRamp_64s (
    nm64s * pVec,
    int64b nOffset,
    int64b nSlope,
    int nSize )
```

Perfomance table 0

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |
| im1  | 10      | 2      | 2048  | 43124 | 21.05      |
| im1  | 10      | 2      | 2048  | 43124 | 21.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im1  | 10      | 2      | 2048  | 43124 | 21.05      |

Perfomance table 1

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 41076 | 20.05      |
| im0  | -10     | 2      | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 41076 | 20.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | -10     | 2      | 2048  | 41076 | 20.05      |

Perfomance table 2

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 41076 | 20.05      |
| im0  | 10      | -2     | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 41076 | 20.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | -2     | 2048  | 41076 | 20.05      |

Perfomance table 3

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 276   | 34.50      |
| im0  | 10      | 2      | 8     | 276   | 34.50      |
| im0  | 10      | 2      | 128   | 2676  | 20.90      |
| im0  | 10      | 2      | 128   | 2676  | 20.90      |
| im0  | 10      | 2      | 1024  | 20596 | 20.11      |
| im0  | 10      | 2      | 1024  | 20596 | 20.11      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 276   | 34.50      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 2048  | 41076 | 20.05      |

#### 6.226.2.4 nmppsRamp\_8s()

```
void nmppsRamp_8s (
    nm8s * pVec,
    int8b nOffset,
    int8b nSlope,
    int nSize )
```

Perfomance table 0

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im1  | 10      | 2      | 2048  | 147562 | 72.05      |
| im1  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

Performance table 1

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | -10     | 2      | 2048  | 147562 | 72.05      |

Performance table 2

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | -2     | 2048  | 147562 | 72.05      |

Performance table 3

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 8     | 682    | 85.25      |
| im0  | 10      | 2      | 8     | 682    | 85.25      |
| im0  | 10      | 2      | 128   | 9322   | 72.82      |
| im0  | 10      | 2      | 128   | 9322   | 72.82      |
| im0  | 10      | 2      | 1024  | 73834  | 72.10      |
| im0  | 10      | 2      | 1024  | 73834  | 72.10      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

The best configuration:

| pVec | nOffset | nSlope | nSize | ticks | ticks/elem |
|------|---------|--------|-------|-------|------------|
| im0  | 10      | 2      | 8     | 682   | 85.25      |

The worst configuration:

| pVec | nOffset | nSlope | nSize | ticks  | ticks/elem |
|------|---------|--------|-------|--------|------------|
| im0  | 10      | 2      | 2048  | 147562 | 72.05      |

## 6.227 nmppsConvert

Изменение разрядности элементов вектора.

Преобразование знаковых данных к меньшей разрядности осуществляется отбрасыванием старших битов. Преобразование знаковых данных к большей разрядности осуществляется с распространением влево старшего (знакового) бита. Преобразование беззнаковых данных к большей разрядности осуществляется добавлением слева старших нулевых битов.

### Функции

- void nmppsConvert\_1s2s (const nm1 \*pSrcVec, nm2s \*pDstVec, int nSize)
- void nmppsConvert\_1u2u (const nm1 \*pSrcVec, nm2u \*pDstVec, int nSize)
- void nmppsConvert\_1u4u (const nm1 \*pSrcVec, nm4u \*pDstVec, int nSize)
- void nmppsConvert\_2s1s (const nm2s \*pSrcVec, nm1 \*pDstVec, int nSize)
- void nmppsConvert\_2s4s (const nm2s \*pSrcVec, nm4s \*pDstVec, int nSize)
- void nmppsConvert\_2u4u (const nm2u \*pSrcVec, nm4u \*pDstVec, int nSize)
- void nmppsConvert\_4s1s (const nm4s \*pSrcVec, nm1 \*pDstVec, int nSize)
- void nmppsConvert\_4s2s (const nm4s \*pSrcVec, nm2s \*pDstVec, int nSize)
- void nmppsConvert\_4s8s (const nm4s \*pSrcVec, nm8s \*pDstVec, int nSize)
- void nmppsConvert\_4u8u (const nm4u \*pSrcVec, nm8u \*pDstVec, int nSize)
- void nmppsConvert\_8s4s (const nm8s \*pSrcVec, nm4s \*pDstVec, int nSize)
- void nmppsConvert\_8s16s (const nm8s \*pSrcVec, nm16s \*pDstVec, int nSize)
- void nmppsConvert\_8s32s (const nm8s \*pSrcVec, nm32s \*pDstVec, int nSize)
- void nmppsConvert\_8s64s (const nm8s \*pSrcVec, nm64s \*pDstVec, int nSize)
- void nmppsConvert\_8u16u (const nm8u \*pSrcVec, nm16u \*pDstVec, int nSize)
- void nmppsConvert\_8u32u (const nm8u \*pSrcVec, nm32u \*pDstVec, int nSize)
- void nmppsConvert\_8u64u (const nm8u \*pSrcVec, nm64u \*pDstVec, int nSize)
- void nmppsConvert\_16s4s (const nm16s \*pSrcVec, nm4s \*pDstVec, int nSize)
- void nmppsConvert\_16s8s (const nm16s \*pSrcVec, nm8s \*pDstVec, int nSize)
- void nmppsConvert\_16s32s (const nm16s \*pSrcVec, nm32s \*pDstVec, int nSize)
- void nmppsConvert\_16s64s (const nm16s \*pSrcVec, nm64s \*pDstVec, int nSize)
- void nmppsConvert\_16u32u (const nm16u \*pSrcVec, nm32u \*pDstVec, int nSize)
- void nmppsConvert\_16u64u (const nm16u \*pSrcVec, nm64u \*pDstVec, int nSize)
- void nmppsConvert\_32s8s (const nm32s \*pSrcVec, nm8s \*pDstVec, int nSize)
- void nmppsConvert\_32s16s (const nm32s \*pSrcVec, nm16s \*pDstVec, int nSize)
- void nmppsConvert\_32s64s (const nm32s \*pSrcVec, nm64s \*pDstVec, int nSize)
- void nmppsConvert\_32u64u (const nm32u \*pSrcVec, nm64u \*pDstVec, int nSize)
- void nmppsConvert\_64s32s (const nm64s \*pSrcVec, nm32s \*pDstVec, int nSize)
- void nmppsConvert\_64s16s (const nm64s \*pSrcVec, nm16s \*pDstVec, int nSize)

### 6.227.1 Подробное описание

Изменение разрядности элементов вектора.

Преобразование знаковых данных к меньшей разрядности осуществляется отбрасыванием старших битов. Преобразование знаковых данных к большей разрядности осуществляется с распространением влево старшего (знакового) бита. Преобразование беззнаковых данных к большей разрядности осуществляется добавлением слева старших нулевых битов.



## Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.227.2 Функции

## 6.227.2.1 nmppsConvert\_16s32s()

```
void nmppsConvert_16s32s (
    const nm16s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

## Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2118  | 1.03       |
| im0     | im0     | 2048  | 2118  | 1.03       |
| im0     | im1     | 2048  | 1256  | 0.61       |
| im0     | im1     | 2048  | 1256  | 0.61       |
| im1     | im0     | 2048  | 1252  | 0.61       |
| im1     | im0     | 2048  | 1252  | 0.61       |
| im1     | im1     | 2048  | 2290  | 1.11       |
| im1     | im1     | 2048  | 2290  | 1.11       |

## The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1252  | 0.61       |

## The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2290  | 1.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 167   | 20.87      |
| im0     | im1     | 8     | 167   | 20.87      |
| im0     | im1     | 128   | 238   | 1.85       |
| im0     | im1     | 128   | 238   | 1.85       |
| im0     | im1     | 1024  | 716   | 0.69       |
| im0     | im1     | 1024  | 716   | 0.69       |
| im0     | im1     | 2048  | 1256  | 0.61       |
| im0     | im1     | 2048  | 1256  | 0.61       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 167   | 20.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1256  | 0.61       |

#### 6.227.2.2 nmppsConvert\_16s4s()

```
void nmppsConvert_16s4s (
    const nm16s * pSrcVec,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 762   | 0.37       |
| im0     | im0     | 2048  | 762   | 0.37       |
| im0     | im1     | 2048  | 697   | 0.34       |
| im0     | im1     | 2048  | 697   | 0.34       |
| im1     | im0     | 2048  | 749   | 0.36       |
| im1     | im0     | 2048  | 749   | 0.36       |
| im1     | im1     | 2048  | 874   | 0.42       |
| im1     | im1     | 2048  | 874   | 0.42       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 697   | 0.34       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 874   | 0.42       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 158   | 19.75      |
| im0     | im1     | 8     | 158   | 19.75      |
| im0     | im1     | 128   | 184   | 1.43       |
| im0     | im1     | 128   | 184   | 1.43       |
| im0     | im1     | 1024  | 435   | 0.42       |
| im0     | im1     | 1024  | 435   | 0.42       |
| im0     | im1     | 2048  | 697   | 0.34       |
| im0     | im1     | 2048  | 697   | 0.34       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 158   | 19.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 697   | 0.34       |

### 6.227.2.3 nmppsConvert\_16s64s()

```
void nmppsConvert_16s64s (
    const nm16s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4165  | 2.03       |
| im0     | im0     | 2048  | 4165  | 2.03       |
| im0     | im1     | 2048  | 2434  | 1.18       |
| im0     | im1     | 2048  | 2434  | 1.18       |
| im1     | im0     | 2048  | 2426  | 1.18       |
| im1     | im0     | 2048  | 2426  | 1.18       |
| im1     | im1     | 2048  | 4474  | 2.18       |
| im1     | im1     | 2048  | 4474  | 2.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2426  | 1.18       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4474  | 2.18       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 255   | 31.87      |
| im0     | im1     | 8     | 255   | 31.87      |
| im0     | im1     | 128   | 398   | 3.10       |
| im0     | im1     | 128   | 398   | 3.10       |
| im0     | im1     | 1024  | 1354  | 1.32       |
| im0     | im1     | 1024  | 1354  | 1.32       |
| im0     | im1     | 2048  | 2434  | 1.18       |
| im0     | im1     | 2048  | 2434  | 1.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 255   | 31.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2434  | 1.18       |

#### 6.227.2.4 nmppsConvert\_16s8s()

```
void nmppsConvert_16s8s (
    const nm16s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 869   | 0.42       |
| im0     | im0     | 2048  | 869   | 0.42       |
| im0     | im1     | 2048  | 691   | 0.33       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 691   | 0.33       |
| im1     | im0     | 2048  | 747   | 0.36       |
| im1     | im0     | 2048  | 747   | 0.36       |
| im1     | im1     | 2048  | 990   | 0.48       |
| im1     | im1     | 2048  | 990   | 0.48       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 691   | 0.33       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 990   | 0.48       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 137   | 17.12      |
| im0     | im1     | 8     | 137   | 17.12      |
| im0     | im1     | 128   | 180   | 1.40       |
| im0     | im1     | 128   | 180   | 1.40       |
| im0     | im1     | 1024  | 423   | 0.41       |
| im0     | im1     | 1024  | 423   | 0.41       |
| im0     | im1     | 2048  | 691   | 0.33       |
| im0     | im1     | 2048  | 691   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 137   | 17.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 691   | 0.33       |

#### 6.227.2.5 nmppsConvert\_16u32u()

```
void nmppsConvert_16u32u (
    const nm16u * pSrcVec,
```

```
nm32u * pDstVec,  
int nSize )
```

Perfomance table 0

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 2114  | 1.03       |
| im0      | im0      | 2048  | 2114  | 1.03       |
| im0      | im1      | 2048  | 1252  | 0.61       |
| im0      | im1      | 2048  | 1252  | 0.61       |
| im1      | im0      | 2048  | 1248  | 0.60       |
| im1      | im0      | 2048  | 1248  | 0.60       |
| im1      | im1      | 2048  | 2286  | 1.11       |
| im1      | im1      | 2048  | 2286  | 1.11       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 1248  | 0.60       |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 2286  | 1.11       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 163   | 20.37      |
| im0      | im1      | 8     | 163   | 20.37      |
| im0      | im1      | 128   | 234   | 1.82       |
| im0      | im1      | 128   | 234   | 1.82       |
| im0      | im1      | 1024  | 712   | 0.69       |
| im0      | im1      | 1024  | 712   | 0.69       |
| im0      | im1      | 2048  | 1252  | 0.61       |
| im0      | im1      | 2048  | 1252  | 0.61       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 163   | 20.37      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 1252  | 0.61       |

#### 6.227.2.6 nmppsConvert\_1s2s()

void nmppsConvert\_1s2s (

```
const nm1 * pSrcVec,
nm2s * pDstVec,
int nSize )
```

Restrictions: nSize =[32\*64,32\*64\*2,32\*64\*3,...]

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 303   | 0.14       |
| im0     | im0     | 2048  | 303   | 0.14       |
| im0     | im1     | 2048  | 366   | 0.17       |
| im0     | im1     | 2048  | 366   | 0.17       |
| im1     | im0     | 2048  | 335   | 0.16       |
| im1     | im0     | 2048  | 335   | 0.16       |
| im1     | im1     | 2048  | 398   | 0.19       |
| im1     | im1     | 2048  | 398   | 0.19       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 303   | 0.14       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 398   | 0.19       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 169   | 21.12      |
| im0     | im1     | 8     | 169   | 21.12      |
| im0     | im1     | 128   | 242   | 1.89       |
| im0     | im1     | 128   | 242   | 1.89       |
| im0     | im1     | 1024  | 283   | 0.27       |
| im0     | im1     | 1024  | 283   | 0.27       |
| im0     | im1     | 2048  | 366   | 0.17       |
| im0     | im1     | 2048  | 366   | 0.17       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 169   | 21.12      |

The worst configuration:



| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 366   | 0.17       |

## 6.227.2.7 nmppsConvert\_1u2u()

```
void nmppsConvert_1u2u (
    const nm1 * pSrcVec,
    nm2u * pDstVec,
    int nSize )
```

Restrictions: nSize =[32\*64,32\*64\*2,32\*64\*3,...]

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 299   | 0.14       |
| im0     | im0     | 2048  | 299   | 0.14       |
| im0     | im1     | 2048  | 363   | 0.17       |
| im0     | im1     | 2048  | 363   | 0.17       |
| im1     | im0     | 2048  | 331   | 0.16       |
| im1     | im0     | 2048  | 331   | 0.16       |
| im1     | im1     | 2048  | 395   | 0.19       |
| im1     | im1     | 2048  | 395   | 0.19       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 299   | 0.14       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 395   | 0.19       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 168   | 21.00      |
| im0     | im1     | 8     | 168   | 21.00      |
| im0     | im1     | 128   | 238   | 1.85       |
| im0     | im1     | 128   | 238   | 1.85       |
| im0     | im1     | 1024  | 280   | 0.27       |
| im0     | im1     | 1024  | 280   | 0.27       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 363   | 0.17       |
| im0     | im1     | 2048  | 363   | 0.17       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 168   | 21.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 363   | 0.17       |

#### 6.227.2.8 nmppsConvert\_1u4u()

```
void nmppsConvert_1u4u (
    const nm1 * pSrcVec,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 565   | 0.27       |
| im0     | im0     | 2048  | 565   | 0.27       |
| im0     | im1     | 2048  | 693   | 0.33       |
| im0     | im1     | 2048  | 693   | 0.33       |
| im1     | im0     | 2048  | 693   | 0.33       |
| im1     | im0     | 2048  | 693   | 0.33       |
| im1     | im1     | 2048  | 821   | 0.40       |
| im1     | im1     | 2048  | 821   | 0.40       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 565   | 0.27       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 821   | 0.40       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 150   | 18.75      |
| im0     | im1     | 8     | 150   | 18.75      |
| im0     | im1     | 128   | 434   | 3.39       |
| im0     | im1     | 128   | 434   | 3.39       |
| im0     | im1     | 1024  | 490   | 0.47       |
| im0     | im1     | 1024  | 490   | 0.47       |
| im0     | im1     | 2048  | 693   | 0.33       |
| im0     | im1     | 2048  | 693   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 150   | 18.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 693   | 0.33       |

#### 6.227.2.9 nmppsConvert\_2s1s()

```
void nmppsConvert_2s1s (
    const nm2s * pSrcVec,
    nm1 * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 215   | 0.10       |
| im0     | im0     | 2048  | 215   | 0.10       |
| im0     | im1     | 2048  | 247   | 0.12       |
| im0     | im1     | 2048  | 247   | 0.12       |
| im1     | im0     | 2048  | 279   | 0.13       |
| im1     | im0     | 2048  | 279   | 0.13       |
| im1     | im1     | 2048  | 311   | 0.15       |
| im1     | im1     | 2048  | 311   | 0.15       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 215   | 0.10       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 311   | 0.15       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 149   | 18.62      |
| im0     | im1     | 8     | 149   | 18.62      |
| im0     | im1     | 128   | 185   | 1.44       |
| im0     | im1     | 128   | 185   | 1.44       |
| im0     | im1     | 1024  | 212   | 0.20       |
| im0     | im1     | 1024  | 212   | 0.20       |
| im0     | im1     | 2048  | 247   | 0.12       |
| im0     | im1     | 2048  | 247   | 0.12       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 149   | 18.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 247   | 0.12       |

#### 6.227.2.10 nmppsConvert\_2s4s()

```
void nmppsConvert_2s4s (
    const nm2s * pSrcVec,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 356   | 0.17       |
| im0     | im0     | 2048  | 356   | 0.17       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im1     | im0     | 2048  | 363   | 0.17       |
| im1     | im0     | 2048  | 363   | 0.17       |
| im1     | im1     | 2048  | 499   | 0.24       |
| im1     | im1     | 2048  | 499   | 0.24       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 356   | 0.17       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 499   | 0.24       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 229   | 28.62      |
| im0     | im1     | 8     | 229   | 28.62      |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 1024  | 301   | 0.29       |
| im0     | im1     | 1024  | 301   | 0.29       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im0     | im1     | 2048  | 371   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 229   | 28.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 371   | 0.18       |

#### 6.227.2.11 nmppsConvert\_2u4u()

```
void nmppsConvert_2u4u (
    const nm2u * pSrcVec,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 356   | 0.17       |
| im0     | im0     | 2048  | 356   | 0.17       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im1     | im0     | 2048  | 363   | 0.17       |
| im1     | im0     | 2048  | 363   | 0.17       |
| im1     | im1     | 2048  | 499   | 0.24       |
| im1     | im1     | 2048  | 499   | 0.24       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 356   | 0.17       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 499   | 0.24       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 229   | 28.62      |
| im0     | im1     | 8     | 229   | 28.62      |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 1024  | 301   | 0.29       |
| im0     | im1     | 1024  | 301   | 0.29       |
| im0     | im1     | 2048  | 371   | 0.18       |
| im0     | im1     | 2048  | 371   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 229   | 28.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 371   | 0.18       |

6.227.2.12 nmppsConvert\_32s16s()

```
void nmppsConvert_32s16s (
    const nm32s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1656  | 0.80       |
| im0     | im0     | 2048  | 1656  | 0.80       |
| im0     | im1     | 2048  | 1242  | 0.60       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1242  | 0.60       |
| im1     | im0     | 2048  | 1320  | 0.64       |
| im1     | im0     | 2048  | 1320  | 0.64       |
| im1     | im1     | 2048  | 1788  | 0.87       |
| im1     | im1     | 2048  | 1788  | 0.87       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1242  | 0.60       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1788  | 0.87       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 128   | 237   | 1.85       |
| im0     | im1     | 1024  | 706   | 0.68       |
| im0     | im1     | 1024  | 706   | 0.68       |
| im0     | im1     | 2048  | 1242  | 0.60       |
| im0     | im1     | 2048  | 1242  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 144   | 18.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1242  | 0.60       |

#### 6.227.2.13 nmppsConvert\_32s64s()

```
void nmppsConvert_32s64s (
    const nm32s * pSrcVec,
```

```
nm64s * pDstVec,  
int nSize )
```

Perfomance table 0



| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 4162  | 2.03       |
| im0      | im0      | 2048  | 4162  | 2.03       |
| im0      | im1      | 2048  | 2343  | 1.14       |
| im0      | im1      | 2048  | 2343  | 1.14       |
| im1      | im0      | 2048  | 2339  | 1.14       |
| im1      | im0      | 2048  | 2339  | 1.14       |
| im1      | im1      | 2048  | 4401  | 2.14       |
| im1      | im1      | 2048  | 4401  | 2.14       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 2339  | 1.14       |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 4401  | 2.14       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 169   | 21.12      |
| im0      | im1      | 8     | 169   | 21.12      |
| im0      | im1      | 128   | 307   | 2.39       |
| im0      | im1      | 128   | 307   | 2.39       |
| im0      | im1      | 1024  | 1255  | 1.22       |
| im0      | im1      | 1024  | 1255  | 1.22       |
| im0      | im1      | 2048  | 2343  | 1.14       |
| im0      | im1      | 2048  | 2343  | 1.14       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 169   | 21.12      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 2343  | 1.14       |

6.227.2.14 nmppsConvert\_32s8s()

void nmppsConvert\_32s8s (

```
const nm32s * pSrcVec,
nm8s * pDstVec,
int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1404  | 0.68       |
| im0     | im0     | 2048  | 1404  | 0.68       |
| im0     | im1     | 2048  | 1214  | 0.59       |
| im0     | im1     | 2048  | 1214  | 0.59       |
| im1     | im0     | 2048  | 1272  | 0.62       |
| im1     | im0     | 2048  | 1272  | 0.62       |
| im1     | im1     | 2048  | 1524  | 0.74       |
| im1     | im1     | 2048  | 1524  | 0.74       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1214  | 0.59       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1524  | 0.74       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 154   | 19.25      |
| im0     | im1     | 8     | 154   | 19.25      |
| im0     | im1     | 128   | 220   | 1.71       |
| im0     | im1     | 128   | 220   | 1.71       |
| im0     | im1     | 1024  | 690   | 0.67       |
| im0     | im1     | 1024  | 690   | 0.67       |
| im0     | im1     | 2048  | 1214  | 0.59       |
| im0     | im1     | 2048  | 1214  | 0.59       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 154   | 19.25      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1214  | 0.59       |

## 6.227.2.15 nmppsConvert\_32u64u()

```
void nmppsConvert_32u64u (
    const nm32u * pSrcVec,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4158  | 2.03       |
| im0     | im0     | 2048  | 4158  | 2.03       |
| im0     | im1     | 2048  | 2339  | 1.14       |
| im0     | im1     | 2048  | 2339  | 1.14       |
| im1     | im0     | 2048  | 2335  | 1.14       |
| im1     | im0     | 2048  | 2335  | 1.14       |
| im1     | im1     | 2048  | 4397  | 2.14       |
| im1     | im1     | 2048  | 4397  | 2.14       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2335  | 1.14       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4397  | 2.14       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 165   | 20.62      |
| im0     | im1     | 8     | 165   | 20.62      |
| im0     | im1     | 128   | 303   | 2.36       |
| im0     | im1     | 128   | 303   | 2.36       |
| im0     | im1     | 1024  | 1251  | 1.22       |
| im0     | im1     | 1024  | 1251  | 1.22       |
| im0     | im1     | 2048  | 2339  | 1.14       |
| im0     | im1     | 2048  | 2339  | 1.14       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 165   | 20.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2339  | 1.14       |

#### 6.227.2.16 nmppsConvert\_4s1s()

```
void nmppsConvert_4s1s (
    const nm4s * pSrcVec,
    nm1 * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 237   | 0.11       |
| im0     | im0     | 2048  | 237   | 0.11       |
| im0     | im1     | 2048  | 279   | 0.13       |
| im0     | im1     | 2048  | 279   | 0.13       |
| im1     | im0     | 2048  | 342   | 0.16       |
| im1     | im0     | 2048  | 342   | 0.16       |
| im1     | im1     | 2048  | 367   | 0.17       |
| im1     | im1     | 2048  | 367   | 0.17       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 237   | 0.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 367   | 0.17       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 128   | 178   | 1.39       |
| im0     | im1     | 128   | 178   | 1.39       |
| im0     | im1     | 1024  | 202   | 0.19       |
| im0     | im1     | 1024  | 202   | 0.19       |
| im0     | im1     | 2048  | 279   | 0.13       |
| im0     | im1     | 2048  | 279   | 0.13       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 144   | 18.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 279   | 0.13       |

#### 6.227.2.17 nmppsConvert\_4s2s()

```
void nmppsConvert_4s2s (
    const nm4s * pSrcVec,
    nm2s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 273   | 0.13       |
| im0     | im0     | 2048  | 273   | 0.13       |
| im0     | im1     | 2048  | 287   | 0.14       |
| im0     | im1     | 2048  | 287   | 0.14       |
| im1     | im0     | 2048  | 346   | 0.16       |
| im1     | im0     | 2048  | 346   | 0.16       |
| im1     | im1     | 2048  | 405   | 0.19       |
| im1     | im1     | 2048  | 405   | 0.19       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 273   | 0.13       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 405   | 0.19       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 145   | 18.12      |
| im0     | im1     | 8     | 145   | 18.12      |
| im0     | im1     | 128   | 146   | 1.14       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 128   | 146   | 1.14       |
| im0     | im1     | 1024  | 216   | 0.21       |
| im0     | im1     | 1024  | 216   | 0.21       |
| im0     | im1     | 2048  | 287   | 0.14       |
| im0     | im1     | 2048  | 287   | 0.14       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 145   | 18.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 287   | 0.14       |

#### 6.227.2.18 nmppsConvert\_4s8s()

```
void nmppsConvert_4s8s (
    const nm4s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 612   | 0.29       |
| im0     | im0     | 2048  | 612   | 0.29       |
| im0     | im1     | 2048  | 474   | 0.23       |
| im0     | im1     | 2048  | 474   | 0.23       |
| im1     | im0     | 2048  | 468   | 0.22       |
| im1     | im0     | 2048  | 468   | 0.22       |
| im1     | im1     | 2048  | 738   | 0.36       |
| im1     | im1     | 2048  | 738   | 0.36       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 468   | 0.22       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 738   | 0.36       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 198   | 24.75      |
| im0     | im1     | 8     | 198   | 24.75      |
| im0     | im1     | 128   | 219   | 1.71       |
| im0     | im1     | 128   | 219   | 1.71       |
| im0     | im1     | 1024  | 340   | 0.33       |
| im0     | im1     | 1024  | 340   | 0.33       |
| im0     | im1     | 2048  | 474   | 0.23       |
| im0     | im1     | 2048  | 474   | 0.23       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 198   | 24.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 474   | 0.23       |

#### 6.227.2.19 nmppsConvert\_4u8u()

```
void nmppsConvert_4u8u (
    const nm4u * pSrcVec,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 613   | 0.29       |
| im0     | im0     | 2048  | 613   | 0.29       |
| im0     | im1     | 2048  | 475   | 0.23       |
| im0     | im1     | 2048  | 475   | 0.23       |
| im1     | im0     | 2048  | 469   | 0.22       |
| im1     | im0     | 2048  | 469   | 0.22       |
| im1     | im1     | 2048  | 739   | 0.36       |
| im1     | im1     | 2048  | 739   | 0.36       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 469   | 0.22       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 739   | 0.36       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 199   | 24.87      |
| im0     | im1     | 8     | 199   | 24.87      |
| im0     | im1     | 128   | 220   | 1.71       |
| im0     | im1     | 128   | 220   | 1.71       |
| im0     | im1     | 1024  | 341   | 0.33       |
| im0     | im1     | 1024  | 341   | 0.33       |
| im0     | im1     | 2048  | 475   | 0.23       |
| im0     | im1     | 2048  | 475   | 0.23       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 199   | 24.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 475   | 0.23       |

6.227.2.20 nmppsConvert\_64s16s()

```
void nmppsConvert_64s16s (
    const nm64s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2704  | 1.32       |
| im0     | im0     | 2048  | 2704  | 1.32       |
| im0     | im1     | 2048  | 2267  | 1.10       |
| im0     | im1     | 2048  | 2267  | 1.10       |
| im1     | im0     | 2048  | 2323  | 1.13       |
| im1     | im0     | 2048  | 2323  | 1.13       |
| im1     | im1     | 2048  | 2832  | 1.38       |
| im1     | im1     | 2048  | 2832  | 1.38       |



The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2267  | 1.10       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2832  | 1.38       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 153   | 19.12      |
| im0     | im1     | 8     | 153   | 19.12      |
| im0     | im1     | 128   | 302   | 2.35       |
| im0     | im1     | 128   | 302   | 2.35       |
| im0     | im1     | 1024  | 1219  | 1.19       |
| im0     | im1     | 1024  | 1219  | 1.19       |
| im0     | im1     | 2048  | 2267  | 1.10       |
| im0     | im1     | 2048  | 2267  | 1.10       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 153   | 19.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2267  | 1.10       |

#### 6.227.2.21 nmppsConvert\_64s32s()

```
void nmppsConvert_64s32s (
    const nm64s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 3221  | 1.57       |
| im0     | im0     | 2048  | 3221  | 1.57       |
| im0     | im1     | 2048  | 2311  | 1.12       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2311  | 1.12       |
| im1     | im0     | 2048  | 2437  | 1.18       |
| im1     | im0     | 2048  | 2437  | 1.18       |
| im1     | im1     | 2048  | 3369  | 1.64       |
| im1     | im1     | 2048  | 3369  | 1.64       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2311  | 1.12       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3369  | 1.64       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 147   | 18.37      |
| im0     | im1     | 8     | 147   | 18.37      |
| im0     | im1     | 128   | 301   | 2.35       |
| im0     | im1     | 128   | 301   | 2.35       |
| im0     | im1     | 1024  | 1239  | 1.20       |
| im0     | im1     | 1024  | 1239  | 1.20       |
| im0     | im1     | 2048  | 2311  | 1.12       |
| im0     | im1     | 2048  | 2311  | 1.12       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 147   | 18.37      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2311  | 1.12       |

#### 6.227.2.22 nmppsConvert\_8s16s()

```
void nmppsConvert_8s16s (
    const nm8s * pSrcVec,
```

```
nm16s * pDstVec,  
int nSize )
```

Perfomance table 0

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 1101  | 0.53       |
| im0      | im0      | 2048  | 1101  | 0.53       |
| im0      | im1      | 2048  | 723   | 0.35       |
| im0      | im1      | 2048  | 723   | 0.35       |
| im1      | im0      | 2048  | 715   | 0.34       |
| im1      | im0      | 2048  | 715   | 0.34       |
| im1      | im1      | 2048  | 1241  | 0.60       |
| im1      | im1      | 2048  | 1241  | 0.60       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 715   | 0.34       |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 1241  | 0.60       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 173   | 21.62      |
| im0      | im1      | 8     | 173   | 21.62      |
| im0      | im1      | 128   | 210   | 1.64       |
| im0      | im1      | 128   | 210   | 1.64       |
| im0      | im1      | 1024  | 449   | 0.43       |
| im0      | im1      | 1024  | 449   | 0.43       |
| im0      | im1      | 2048  | 723   | 0.35       |
| im0      | im1      | 2048  | 723   | 0.35       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 173   | 21.62      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 723   | 0.35       |

6.227.2.23 nmppsConvert\_8s32s()

void nmppsConvert\_8s32s (

```

const nm8s * pSrcVec,
nm32s * pDstVec,
int nSize )

```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2133  | 1.04       |
| im0     | im0     | 2048  | 2133  | 1.04       |
| im0     | im1     | 2048  | 1370  | 0.66       |
| im0     | im1     | 2048  | 1370  | 0.66       |
| im1     | im0     | 2048  | 1354  | 0.66       |
| im1     | im0     | 2048  | 1354  | 0.66       |
| im1     | im1     | 2048  | 2378  | 1.16       |
| im1     | im1     | 2048  | 2378  | 1.16       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1354  | 0.66       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2378  | 1.16       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 270   | 33.75      |
| im0     | im1     | 8     | 270   | 33.75      |
| im0     | im1     | 128   | 338   | 2.64       |
| im0     | im1     | 128   | 338   | 2.64       |
| im0     | im1     | 1024  | 822   | 0.80       |
| im0     | im1     | 1024  | 822   | 0.80       |
| im0     | im1     | 2048  | 1370  | 0.66       |
| im0     | im1     | 2048  | 1370  | 0.66       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 270   | 33.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1370  | 0.66       |

## 6.227.2.24 nmppsConvert\_8s4s()

```
void nmppsConvert_8s4s (
    const nm8s * pSrcVec,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 487   | 0.23       |
| im0     | im0     | 2048  | 487   | 0.23       |
| im0     | im1     | 2048  | 440   | 0.21       |
| im0     | im1     | 2048  | 440   | 0.21       |
| im1     | im0     | 2048  | 482   | 0.23       |
| im1     | im0     | 2048  | 482   | 0.23       |
| im1     | im1     | 2048  | 610   | 0.29       |
| im1     | im1     | 2048  | 610   | 0.29       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 440   | 0.21       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 610   | 0.29       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 153   | 19.12      |
| im0     | im1     | 8     | 153   | 19.12      |
| im0     | im1     | 128   | 173   | 1.35       |
| im0     | im1     | 128   | 173   | 1.35       |
| im0     | im1     | 1024  | 310   | 0.30       |
| im0     | im1     | 1024  | 310   | 0.30       |
| im0     | im1     | 2048  | 440   | 0.21       |
| im0     | im1     | 2048  | 440   | 0.21       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 153   | 19.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 440   | 0.21       |

#### 6.227.2.25 nmppsConvert\_8s64s()

```
void nmppsConvert_8s64s (
    const nm8s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4184  | 2.04       |
| im0     | im0     | 2048  | 4184  | 2.04       |
| im0     | im1     | 2048  | 2658  | 1.29       |
| im0     | im1     | 2048  | 2658  | 1.29       |
| im1     | im0     | 2048  | 2626  | 1.28       |
| im1     | im0     | 2048  | 2626  | 1.28       |
| im1     | im1     | 2048  | 4674  | 2.28       |
| im1     | im1     | 2048  | 4674  | 2.28       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2626  | 1.28       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4674  | 2.28       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 458   | 57.25      |
| im0     | im1     | 8     | 458   | 57.25      |
| im0     | im1     | 128   | 596   | 4.65       |
| im0     | im1     | 128   | 596   | 4.65       |
| im0     | im1     | 1024  | 1562  | 1.52       |
| im0     | im1     | 1024  | 1562  | 1.52       |
| im0     | im1     | 2048  | 2658  | 1.29       |
| im0     | im1     | 2048  | 2658  | 1.29       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 458   | 57.25      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2658  | 1.29       |

#### 6.227.2.26 nmppsConvert\_8u16u()

```
void nmppsConvert_8u16u (
    const nm8u * pSrcVec,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1096  | 0.53       |
| im0     | im0     | 2048  | 1096  | 0.53       |
| im0     | im1     | 2048  | 718   | 0.35       |
| im0     | im1     | 2048  | 718   | 0.35       |
| im1     | im0     | 2048  | 710   | 0.34       |
| im1     | im0     | 2048  | 710   | 0.34       |
| im1     | im1     | 2048  | 1236  | 0.60       |
| im1     | im1     | 2048  | 1236  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 710   | 0.34       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1236  | 0.60       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 168   | 21.00      |
| im0     | im1     | 8     | 168   | 21.00      |
| im0     | im1     | 128   | 205   | 1.60       |



| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 128   | 205   | 1.60       |
| im0     | im1     | 1024  | 444   | 0.43       |
| im0     | im1     | 1024  | 444   | 0.43       |
| im0     | im1     | 2048  | 718   | 0.35       |
| im0     | im1     | 2048  | 718   | 0.35       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 168   | 21.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 718   | 0.35       |

#### 6.227.2.27 nmppsConvert\_8u32u()

```
void nmppsConvert_8u32u (
    const nm8u * pSrcVec,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2128  | 1.03       |
| im0     | im0     | 2048  | 2128  | 1.03       |
| im0     | im1     | 2048  | 1365  | 0.66       |
| im0     | im1     | 2048  | 1365  | 0.66       |
| im1     | im0     | 2048  | 1349  | 0.65       |
| im1     | im0     | 2048  | 1349  | 0.65       |
| im1     | im1     | 2048  | 2373  | 1.15       |
| im1     | im1     | 2048  | 2373  | 1.15       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1349  | 0.65       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2373  | 1.15       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 265   | 33.12      |
| im0      | im1      | 8     | 265   | 33.12      |
| im0      | im1      | 128   | 333   | 2.60       |
| im0      | im1      | 128   | 333   | 2.60       |
| im0      | im1      | 1024  | 817   | 0.79       |
| im0      | im1      | 1024  | 817   | 0.79       |
| im0      | im1      | 2048  | 1365  | 0.66       |
| im0      | im1      | 2048  | 1365  | 0.66       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 265   | 33.12      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 1365  | 0.66       |

#### 6.227.2.28 nmppsConvert\_8u64u()

```
void nmppsConvert_8u64u (
    const nm8u * pSrcVec,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 4180  | 2.04       |
| im0      | im0      | 2048  | 4180  | 2.04       |
| im0      | im1      | 2048  | 2654  | 1.29       |
| im0      | im1      | 2048  | 2654  | 1.29       |
| im1      | im0      | 2048  | 2622  | 1.28       |
| im1      | im0      | 2048  | 2622  | 1.28       |
| im1      | im1      | 2048  | 4670  | 2.28       |
| im1      | im1      | 2048  | 4670  | 2.28       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 2622  | 1.28       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4670  | 2.28       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 454   | 56.75      |
| im0     | im1     | 8     | 454   | 56.75      |
| im0     | im1     | 128   | 592   | 4.62       |
| im0     | im1     | 128   | 592   | 4.62       |
| im0     | im1     | 1024  | 1558  | 1.52       |
| im0     | im1     | 1024  | 1558  | 1.52       |
| im0     | im1     | 2048  | 2654  | 1.29       |
| im0     | im1     | 2048  | 2654  | 1.29       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 454   | 56.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2654  | 1.29       |

## 6.228 nmppsConvert

Изменение разрядности элементов вектора. Входной вектор.

### Функции

- void `nmppsConvert_32u32fcr` (const `nm32u` \*pSrcVec, `nm32fcr` \*pDstVec, int nSize)
- void `nmppsConvert_32s32fcr` (const `nm32s` \*pSrcVec, `nm32fcr` \*pDstVec, int nSize)
- void `nmppsConvert_32f32fcr` (const `nm32f` \*pSrcVec, `nm32fcr` \*pDstVec, int nSize)
- void `nmppsConvert_32sc32fcr` (const `nm32sc` \*pSrcVec, `nm32fcr` \*pDstVec, int nSize)
- void `nmppsConvert_32s32f` (const `nm32s` \*pSrcVec, `nm32f` \*pDstVec, int nSize)
- void `nmppsConvert_32f64f` (const `nm32f` \*pSrcVec, `nm64f` \*pDstVec, int nSize)
- void `nmppsConvert_64f32f` (const `nm64f` \*pSrcVec, `nm32f` \*pDstVec, int nSize)

### 6.228.1 Подробное описание

Изменение разрядности элементов вектора. Входной вектор.

#### Аргументы

|       |                              |
|-------|------------------------------|
| nSize | Размер векторов в элементах. |
|-------|------------------------------|

#### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

#### Возвращает

void

Функции выполняются на сопроцессоре (NM6407) с плавающей точкой с использованием переупаковщика данных

## 6.228.2 Функции

### 6.228.2.1 nmppsConvert\_32f32fcr()

```
void nmppsConvert_32f32fcr (
    const nm32f * pSrcVec,
    nm32fcr * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im2     | 2048  | 3067  | 1.49       |
| im2     | im2     | 2048  | 3067  | 1.49       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 2090  | 1.02       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 108   | 13.49      |
| im3     | im4     | 8     | 108   | 13.49      |
| im3     | im4     | 128   | 170   | 1.32       |
| im3     | im4     | 128   | 170   | 1.32       |
| im3     | im4     | 1024  | 1066  | 1.04       |
| im3     | im4     | 1024  | 1066  | 1.04       |
| im3     | im4     | 2048  | 2090  | 1.02       |
| im3     | im4     | 2048  | 2090  | 1.02       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 108   | 13.49      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 2090  | 1.02       |

### 6.228.2.2 nmppsConvert\_32f64f()

void nmppsConvert\_32f64f (

```
const nm32f * pSrcVec,
nm64f * pDstVec,
int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3059  | 1.49       |
| im1     | im1     | 2048  | 3059  | 1.49       |
| im1     | im2     | 2048  | 2068  | 1.00       |
| im1     | im2     | 2048  | 2068  | 1.00       |
| im2     | im1     | 2048  | 2068  | 1.00       |
| im2     | im1     | 2048  | 2068  | 1.00       |
| im2     | im2     | 2048  | 3059  | 1.49       |
| im2     | im2     | 2048  | 3059  | 1.49       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 2068  | 1.00       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3059  | 1.49       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 89    | 11.12      |
| im3     | im4     | 8     | 89    | 11.12      |
| im3     | im4     | 128   | 148   | 1.15       |
| im3     | im4     | 128   | 148   | 1.15       |
| im3     | im4     | 1024  | 1044  | 1.01       |
| im3     | im4     | 1024  | 1044  | 1.01       |
| im3     | im4     | 2048  | 2068  | 1.00       |
| im3     | im4     | 2048  | 2068  | 1.00       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 89    | 11.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 2068  | 1.00       |

## 6.228.2.3 nmppsConvert\_32s32f()

```
void nmppsConvert_32s32f (
    const nm32s * pSrcVec,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2005  | 0.97       |
| im1     | im1     | 2048  | 2005  | 0.97       |
| im1     | im2     | 2048  | 2002  | 0.97       |
| im1     | im2     | 2048  | 2002  | 0.97       |
| im2     | im1     | 2048  | 2002  | 0.97       |
| im2     | im1     | 2048  | 2002  | 0.97       |
| im2     | im2     | 2048  | 2005  | 0.97       |
| im2     | im2     | 2048  | 2005  | 0.97       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 2002  | 0.97       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2005  | 0.97       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 84    | 10.50      |
| im3     | im4     | 8     | 84    | 10.50      |
| im3     | im4     | 128   | 100   | 0.78       |
| im3     | im4     | 128   | 100   | 0.78       |
| im3     | im4     | 1024  | 978   | 0.95       |
| im3     | im4     | 1024  | 978   | 0.95       |
| im3     | im4     | 2048  | 2002  | 0.97       |
| im3     | im4     | 2048  | 2002  | 0.97       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 84    | 10.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 2002  | 0.97       |

#### 6.228.2.4 nmppsConvert\_32s32fcr()

```
void nmppsConvert_32s32fcr (
    const nm32s * pSrcVec,
    nm32fcr * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im2     | 2048  | 3067  | 1.49       |
| im2     | im2     | 2048  | 3067  | 1.49       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 2090  | 1.02       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 108   | 13.50      |
| im3     | im4     | 8     | 108   | 13.50      |
| im3     | im4     | 128   | 170   | 1.32       |
| im3     | im4     | 128   | 170   | 1.32       |
| im3     | im4     | 1024  | 1066  | 1.04       |
| im3     | im4     | 1024  | 1066  | 1.04       |
| im3     | im4     | 2048  | 2090  | 1.02       |
| im3     | im4     | 2048  | 2090  | 1.02       |



The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 108   | 13.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 2090  | 1.02       |

#### 6.228.2.5 nmppsConvert\_32sc32fcr()

```
void nmppsConvert_32sc32fcr (
    const nm32sc * pSrcVec,
    nm32fcr * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4588  | 2.24       |
| im1     | im1     | 2048  | 4588  | 2.24       |
| im1     | im2     | 2048  | 4065  | 1.98       |
| im1     | im2     | 2048  | 4065  | 1.98       |
| im2     | im1     | 2048  | 4065  | 1.98       |
| im2     | im1     | 2048  | 4065  | 1.98       |
| im2     | im2     | 2048  | 4588  | 2.24       |
| im2     | im2     | 2048  | 4588  | 2.24       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 4065  | 1.98       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4588  | 2.24       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 87    | 10.87      |
| im3     | im4     | 8     | 87    | 10.87      |
| im3     | im4     | 128   | 225   | 1.75       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 128   | 225   | 1.75       |
| im3     | im4     | 1024  | 2017  | 1.96       |
| im3     | im4     | 1024  | 2017  | 1.96       |
| im3     | im4     | 2048  | 4065  | 1.98       |
| im3     | im4     | 2048  | 4065  | 1.98       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 87    | 10.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 4065  | 1.98       |

#### 6.228.2.6 nmppsConvert\_32u32fcr()

```
void nmppsConvert_32u32fcr (
    const nm32u * pSrcVec,
    nm32fcr * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im1     | 2048  | 3067  | 1.49       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im1     | im2     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im1     | 2048  | 2090  | 1.02       |
| im2     | im2     | 2048  | 3067  | 1.49       |
| im2     | im2     | 2048  | 3067  | 1.49       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im2     | 2048  | 2090  | 1.02       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3067  | 1.49       |

Perfomance table 1

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im3      | im4      | 8     | 108   | 13.50      |
| im3      | im4      | 8     | 108   | 13.50      |
| im3      | im4      | 128   | 170   | 1.32       |
| im3      | im4      | 128   | 170   | 1.32       |
| im3      | im4      | 1024  | 1066  | 1.04       |
| im3      | im4      | 1024  | 1066  | 1.04       |
| im3      | im4      | 2048  | 2090  | 1.02       |
| im3      | im4      | 2048  | 2090  | 1.02       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im3      | im4      | 8     | 108   | 13.50      |

The worst configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im3      | im4      | 2048  | 2090  | 1.02       |

### 6.228.2.7 nmppsConvert\_64f32f()

```
void nmppsConvert_64f32f (
    const nm64f * pSrcVec,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 3055  | 1.49       |
| im1      | im1      | 2048  | 3055  | 1.49       |
| im1      | im2      | 2048  | 2079  | 1.01       |
| im1      | im2      | 2048  | 2079  | 1.01       |
| im2      | im1      | 2048  | 2079  | 1.01       |
| im2      | im1      | 2048  | 2079  | 1.01       |
| im2      | im2      | 2048  | 3055  | 1.49       |
| im2      | im2      | 2048  | 3055  | 1.49       |

The best configuration:

| pSrc Vec | pDst Vec | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im2      | 2048  | 2079  | 1.01       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3055  | 1.49       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 89    | 11.12      |
| im3     | im4     | 8     | 89    | 11.12      |
| im3     | im4     | 128   | 159   | 1.24       |
| im3     | im4     | 128   | 159   | 1.24       |
| im3     | im4     | 1024  | 1055  | 1.03       |
| im3     | im4     | 1024  | 1055  | 1.03       |
| im3     | im4     | 2048  | 2079  | 1.01       |
| im3     | im4     | 2048  | 2079  | 1.01       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 8     | 89    | 11.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im3     | im4     | 2048  | 2079  | 1.01       |

## 6.229 nmppsConvert\_rounding

Функции конвертации вектора чисел с плавающей точкой одинарной точности в вектор целых 32-битных чисел с разной степенью округления

### Функции

- void nmppsConvert\_32f32s\_rounding (const nm32f \*pSrcVec, nm32s \*pDstVec, int scale, int nSize)
- void nmppsConvert\_32f32s\_ceiling (const nm32f \*pSrcVec, nm32s \*pDstVec, int scale, int nSize)
- void nmppsConvert\_32f32s\_floor (const nm32f \*pSrcVec, nm32s \*pDstVec, int scale, int nSize)
- void nmppsConvert\_32f32s\_truncate (const nm32f \*pSrcVec, nm32s \*pDstVec, int scale, int nSize)

### 6.229.1 Подробное описание

Функции конвертации вектора чисел с плавающей точкой одинарной точности в вектор целых 32-битных чисел с разной степенью округления

#### Аргументы

|         |                                                                                                                |
|---------|----------------------------------------------------------------------------------------------------------------|
| pSrcVec | указатель на входной вектор                                                                                    |
| pDstVec | указатель на выходной вектор                                                                                   |
| scale   | степень двойки (умножение каждого элемента выходного вектора на $2^{\text{scale}}$ ), может быть отрицательной |
| nSize   | число элементов во входном векторе (может быть только четным)                                                  |

Функция выполняется на сопроцессоре (процессор 1879ВМ6Я) с плавающей точкой с использованием переупаковщика данных

nmppsConvert\_32f32s\_rounding округляет все дробные числа из pSrcVec до ближайших целых (например, 1.5 будет округлено до 2, 1.7 до 2, а 1.4 до 1)

nmppsConvert\_32f32s\_ceiling округляет все дробные числа из pSrcVec к меньшему по модулю (например, 1.5 будет округлено до 1, 1.7 до 1, а -1.1 до -1)

### 6.229.2 Функции

#### 6.229.2.1 nmppsConvert\_32f32s\_ceiling()

```
void nmppsConvert_32f32s_ceiling (
    const nm32f * pSrcVec,
    nm32s * pDstVec,
    int scale,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im2     | im1     | 1     | 2048  | 2233  | 1.09       |
| im2     | im1     | 1     | 2048  | 2233  | 1.09       |
| im2     | im2     | 1     | 2048  | 3857  | 1.88       |
| im2     | im2     | 1     | 2048  | 3857  | 1.88       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |

Perfomance table 1

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 117   | 14.62      |
| im1     | im2     | 1     | 8     | 117   | 14.62      |
| im1     | im2     | 1     | 128   | 133   | 1.03       |
| im1     | im2     | 1     | 128   | 133   | 1.03       |
| im1     | im2     | 1     | 1024  | 1129  | 1.10       |
| im1     | im2     | 1     | 1024  | 1129  | 1.10       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

#### 6.229.2.2 nmppsConvert\_32f32s\_floor()

```
void nmppsConvert_32f32s_floor (
```

```

const nm32f * pSrcVec,
nm32s * pDstVec,
int scale,
int nSize )

```

Perfomance table 0

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im2     | im1     | 1     | 2048  | 2233  | 1.09       |
| im2     | im1     | 1     | 2048  | 2233  | 1.09       |
| im2     | im2     | 1     | 2048  | 3857  | 1.88       |
| im2     | im2     | 1     | 2048  | 3857  | 1.88       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 3857  | 1.88       |

Perfomance table 1

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 117   | 14.62      |
| im1     | im2     | 1     | 8     | 117   | 14.62      |
| im1     | im2     | 1     | 128   | 133   | 1.03       |
| im1     | im2     | 1     | 128   | 133   | 1.03       |
| im1     | im2     | 1     | 1024  | 1129  | 1.10       |
| im1     | im2     | 1     | 1024  | 1129  | 1.10       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2233  | 1.09       |

## 6.229.2.3 nmppsConvert\_32f32s\_rounding()

```
void nmppsConvert_32f32s_rounding (
    const nm32f * pSrcVec,
    nm32s * pDstVec,
    int scale,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 2326  | 1.13       |
| im1     | im1     | 1     | 2048  | 2326  | 1.13       |
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |
| im2     | im1     | 1     | 2048  | 2034  | 0.99       |
| im2     | im1     | 1     | 2048  | 2034  | 0.99       |
| im2     | im2     | 1     | 2048  | 2326  | 1.13       |
| im2     | im2     | 1     | 2048  | 2326  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 2326  | 1.13       |

Perfomance table 1

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 112   | 14.00      |
| im1     | im2     | 1     | 8     | 112   | 14.00      |
| im1     | im2     | 1     | 128   | 130   | 1.01       |
| im1     | im2     | 1     | 128   | 130   | 1.01       |
| im1     | im2     | 1     | 1024  | 1010  | 0.98       |
| im1     | im2     | 1     | 1024  | 1010  | 0.98       |
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 112   | 14.00      |



The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 2034  | 0.99       |

#### 6.229.2.4 nmppsConvert\_32f32s\_truncate()

```
void nmppsConvert_32f32s_truncate (
    const nm32f * pSrcVec,
    nm32s * pDstVec,
    int scale,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 5086  | 2.48       |
| im1     | im1     | 1     | 2048  | 5086  | 2.48       |
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |
| im2     | im1     | 1     | 2048  | 4156  | 2.02       |
| im2     | im1     | 1     | 2048  | 4156  | 2.02       |
| im2     | im2     | 1     | 2048  | 5086  | 2.48       |
| im2     | im2     | 1     | 2048  | 5086  | 2.48       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im1     | 1     | 2048  | 5086  | 2.48       |

Perfomance table 1

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 150   | 18.75      |
| im1     | im2     | 1     | 8     | 150   | 18.75      |
| im1     | im2     | 1     | 128   | 286   | 2.23       |
| im1     | im2     | 1     | 128   | 286   | 2.23       |
| im1     | im2     | 1     | 1024  | 2092  | 2.04       |
| im1     | im2     | 1     | 1024  | 2092  | 2.04       |
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |

The best configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 8     | 150   | 18.75      |

The worst configuration:

| pSrcVec | pDstVec | scale | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|-------|------------|
| im1     | im2     | 1     | 2048  | 4156  | 2.02       |

## 6.230 nmppsMerge

Функция соединяет 2 массива 32-х чисел с плавающей точкой (float) в один. Результирующий массив { pSrcVec1[0], pSrcVec2[0], pSrcVec1[1], pSrcVec2[1], pSrcVec1[2], pSrcVec2[2] и т.д. }.

### Функции

- void [nmppsMerge\\_32f](#) (const nm32f \*pSrcVec1, const nm32f \*pSrcVec2, nm32f \*pDstVec, int nSize)

### 6.230.1 Подробное описание

Функция соединяет 2 массива 32-х чисел с плавающей точкой (float) в один. Результирующий массив { pSrcVec1[0], pSrcVec2[0], pSrcVec1[1], pSrcVec2[1], pSrcVec1[2], pSrcVec2[2] и т.д. }.

### Аргументы

|          |                                                                                            |
|----------|--------------------------------------------------------------------------------------------|
| pSrcVec1 | указатель на входной массив чисел float                                                    |
| pSrcVec2 | указатель на входной массив чисел float                                                    |
| pDstVec  | указатель на выходной массив чисел float                                                   |
| nSize    | количество элементов в массиве pSrcVec (в массиве pDstVec элементов будет в 2 раза больше) |

### 6.230.2 Функции

#### 6.230.2.1 nmppsMerge\_32f()

```
void nmppsMerge_32f (
    const nm32f * pSrcVec1,
    const nm32f * pSrcVec2,
    nm32f * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |
| im1      | im2      | im3     | 2048  | 2144  | 1.04       |
| im1      | im2      | im3     | 2048  | 2144  | 1.04       |
| im2      | im1      | im3     | 2048  | 2144  | 1.04       |
| im2      | im1      | im3     | 2048  | 2144  | 1.04       |
| im2      | im2      | im3     | 2048  | 2535  | 1.23       |
| im2      | im2      | im3     | 2048  | 2535  | 1.23       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im2      | im3     | 2048  | 2144  | 1.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 169   | 21.12      |
| im1      | im1      | im3     | 8     | 169   | 21.12      |
| im1      | im1      | im3     | 128   | 255   | 1.99       |
| im1      | im1      | im3     | 128   | 255   | 1.99       |
| im1      | im1      | im3     | 1024  | 1319  | 1.28       |
| im1      | im1      | im3     | 1024  | 1319  | 1.28       |
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |
| im1      | im2      | im3     | 8     | 169   | 21.12      |
| im1      | im2      | im3     | 8     | 169   | 21.12      |
| im1      | im2      | im3     | 128   | 224   | 1.75       |
| im1      | im2      | im3     | 128   | 224   | 1.75       |
| im1      | im2      | im3     | 1024  | 1120  | 1.09       |
| im1      | im2      | im3     | 1024  | 1120  | 1.09       |
| im1      | im2      | im3     | 2048  | 2144  | 1.04       |
| im1      | im2      | im3     | 2048  | 2144  | 1.04       |
| im2      | im1      | im3     | 8     | 169   | 21.12      |
| im2      | im1      | im3     | 8     | 169   | 21.12      |
| im2      | im1      | im3     | 128   | 224   | 1.75       |
| im2      | im1      | im3     | 128   | 224   | 1.75       |
| im2      | im1      | im3     | 1024  | 1120  | 1.09       |
| im2      | im1      | im3     | 1024  | 1120  | 1.09       |
| im2      | im1      | im3     | 2048  | 2144  | 1.04       |
| im2      | im1      | im3     | 2048  | 2144  | 1.04       |
| im2      | im2      | im3     | 8     | 169   | 21.12      |
| im2      | im2      | im3     | 8     | 169   | 21.12      |
| im2      | im2      | im3     | 128   | 255   | 1.99       |
| im2      | im2      | im3     | 128   | 255   | 1.99       |
| im2      | im2      | im3     | 1024  | 1319  | 1.28       |
| im2      | im2      | im3     | 1024  | 1319  | 1.28       |
| im2      | im2      | im3     | 2048  | 2535  | 1.23       |
| im2      | im2      | im3     | 2048  | 2535  | 1.23       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 8     | 169   | 21.12      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im1      | im1      | im3     | 2048  | 2535  | 1.23       |

## 6.231 nmppsConvertRisc

Функции изменения разрядности чисел входного вектора

Функции

- void `nmppsConvertRisc_8u32u` (const `nm8u` \*pSrcVec, `nm32u` \*pDstVec, int nSize)
- void `nmppsConvertRisc_32u8u` (const `nm32u` \*pSrcVec, `nm8u` \*pDstVec, int nSize)

### 6.231.1 Подробное описание

Функции изменения разрядности чисел входного вектора

Аргументы

|         |                                                                                    |
|---------|------------------------------------------------------------------------------------|
| pSrcVec | указатель на входной вектор                                                        |
| pDstVec | указатель на выходной вектор                                                       |
| nSize   | число элементов во входном векторе (должно быть кратно 8 и не может быть меньше 8) |

Функция выполняется на RISC-процессоре

### 6.231.2 Функции

#### 6.231.2.1 nmppsConvertRisc\_32u8u()

```
void nmppsConvertRisc_32u8u (
    const nm32u * pSrcVec,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 10324 | 5.04       |
| im0     | im2     | 2048  | 10324 | 5.04       |
| im1     | im2     | 2048  | 10324 | 5.04       |
| im1     | im2     | 2048  | 10324 | 5.04       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 10324 | 5.04       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 10324 | 5.04       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 125   | 15.62      |
| im0     | im1     | 8     | 125   | 15.62      |
| im0     | im1     | 128   | 725   | 5.66       |
| im0     | im1     | 128   | 725   | 5.66       |
| im0     | im1     | 1024  | 5205  | 5.08       |
| im0     | im1     | 1024  | 5205  | 5.08       |
| im0     | im1     | 2048  | 10325 | 5.04       |
| im0     | im1     | 2048  | 10325 | 5.04       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 125   | 15.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 10325 | 5.04       |

### 6.231.2.2 nmppsConvertRisc\_8u32u()

```
void nmppsConvertRisc_8u32u (
    const nm8u * pSrcVec,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 7257  | 3.54       |
| im0     | im2     | 2048  | 7257  | 3.54       |
| im1     | im2     | 2048  | 7257  | 3.54       |
| im1     | im2     | 2048  | 7257  | 3.54       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 7257  | 3.54       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im2     | 2048  | 7257  | 3.54       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 128   | 538   | 4.20       |
| im0     | im1     | 128   | 538   | 4.20       |
| im0     | im1     | 1024  | 3674  | 3.58       |
| im0     | im1     | 1024  | 3674  | 3.58       |
| im0     | im1     | 2048  | 7258  | 3.54       |
| im0     | im1     | 2048  | 7258  | 3.54       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 118   | 14.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 7258  | 3.54       |



## 6.232 nmppsCopy\_

Копирование вектора.

## Функции

- void nmppsCopy\_2s (const nm2s \*pSrcVec, nm2s \*pDstVec, int nSize)
- void nmppsCopy\_4s (const nm4s \*pSrcVec, nm4s \*pDstVec, int nSize)
- void nmppsCopy\_8s (const nm8s \*pSrcVec, nm8s \*pDstVec, int nSize)
- void nmppsCopy\_16s (const nm16s \*pSrcVec, nm16s \*pDstVec, int nSize)
- void nmppsCopy\_32s (const nm32s \*pSrcVec, nm32s \*pDstVec, int nSize)
- void nmppsCopy\_64s (const nm64s \*pSrcVec, nm64s \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_4u (const nm4u \*pSrcVec, nm4u \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_8u (const nm8u \*pSrcVec, nm8u \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_16u (const nm16u \*pSrcVec, nm16u \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_32u (const nm32u \*pSrcVec, nm32u \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_64u (const nm64u \*pSrcVec, nm64u \*pDstVec, int nSize)
- \_\_INLINE\_\_ void nmppsCopy\_64sc (const nm64sc \*pSrcVec, nm64sc \*pDstVec, int nSize)

## 6.232.1 Подробное описание

Копирование вектора.

$$pDstVec[i] = pSrcVec[i],$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.232.2 Функции

## 6.232.2.1 nmppsCopy\_16s()

```
void nmppsCopy_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 647   | 0.31       |
| im0     | im1     | 2048  | 647   | 0.31       |
| im0     | im1     | 2048  | 647   | 0.31       |
| im0     | im1     | 2048  | 647   | 0.31       |
| im1     | im1     | 2048  | 1157  | 0.56       |
| im1     | im1     | 2048  | 1157  | 0.56       |
| im1     | im1     | 2048  | 1157  | 0.56       |
| im1     | im1     | 2048  | 1157  | 0.56       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 647   | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1157  | 0.56       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 128   | 138   | 1.07       |
| im0     | im1     | 128   | 138   | 1.07       |
| im0     | im1     | 1024  | 377   | 0.36       |
| im0     | im1     | 1024  | 377   | 0.36       |
| im0     | im1     | 2048  | 647   | 0.31       |
| im0     | im1     | 2048  | 647   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 647   | 0.31       |

## 6.232.2.2 nmppsCopy\_16u()

```
__INLINE__ void nmppsCopy_16u (
    const nm16u * pSrcVec,
    nm16u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 696   | 0.33       |
| im0     | im1     | 2048  | 696   | 0.33       |
| im0     | im1     | 2048  | 696   | 0.33       |
| im0     | im1     | 2048  | 696   | 0.33       |
| im1     | im1     | 2048  | 1206  | 0.58       |
| im1     | im1     | 2048  | 1206  | 0.58       |
| im1     | im1     | 2048  | 1206  | 0.58       |
| im1     | im1     | 2048  | 1206  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 696   | 0.33       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1206  | 0.58       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 128   | 187   | 1.46       |
| im0     | im1     | 128   | 187   | 1.46       |
| im0     | im1     | 1024  | 426   | 0.41       |
| im0     | im1     | 1024  | 426   | 0.41       |
| im0     | im1     | 2048  | 696   | 0.33       |
| im0     | im1     | 2048  | 696   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 696   | 0.33       |

См. определение в файле vInit.h строка 627

### 6.232.2.3 nmppsCopy\_2s()

```
void nmppsCopy_2s (
    const nm2s * pSrcVec,
    nm2s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 173   | 0.08       |
| im0     | im1     | 2048  | 173   | 0.08       |
| im0     | im1     | 2048  | 173   | 0.08       |
| im0     | im1     | 2048  | 173   | 0.08       |
| im1     | im1     | 2048  | 233   | 0.11       |
| im1     | im1     | 2048  | 233   | 0.11       |
| im1     | im1     | 2048  | 233   | 0.11       |
| im1     | im1     | 2048  | 233   | 0.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 173   | 0.08       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 233   | 0.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 128   | 105   | 0.82       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 128   | 105   | 0.82       |
| im0     | im1     | 1024  | 138   | 0.13       |
| im0     | im1     | 1024  | 138   | 0.13       |
| im0     | im1     | 2048  | 173   | 0.08       |
| im0     | im1     | 2048  | 173   | 0.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 173   | 0.08       |

#### 6.232.2.4 nmppsCopy\_32s()

```
void nmppsCopy_32s (
    const nm32s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1191  | 0.58       |
| im0     | im1     | 2048  | 1191  | 0.58       |
| im0     | im1     | 2048  | 1191  | 0.58       |
| im0     | im1     | 2048  | 1191  | 0.58       |
| im1     | im1     | 2048  | 2213  | 1.08       |
| im1     | im1     | 2048  | 2213  | 1.08       |
| im1     | im1     | 2048  | 2213  | 1.08       |
| im1     | im1     | 2048  | 2213  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1191  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2213  | 1.08       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 105   | 13.12      |
| im0     | im1     | 8     | 105   | 13.12      |
| im0     | im1     | 128   | 173   | 1.35       |
| im0     | im1     | 128   | 173   | 1.35       |
| im0     | im1     | 1024  | 647   | 0.63       |
| im0     | im1     | 1024  | 647   | 0.63       |
| im0     | im1     | 2048  | 1191  | 0.58       |
| im0     | im1     | 2048  | 1191  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 105   | 13.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1191  | 0.58       |

#### 6.232.2.5 nmppsCopy\_32u()

```
__INLINE__ void nmppsCopy_32u (
    const nm32u * pSrcVec,
    nm32u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1240  | 0.60       |
| im0     | im1     | 2048  | 1240  | 0.60       |
| im0     | im1     | 2048  | 1240  | 0.60       |
| im0     | im1     | 2048  | 1240  | 0.60       |
| im1     | im1     | 2048  | 2262  | 1.10       |
| im1     | im1     | 2048  | 2262  | 1.10       |
| im1     | im1     | 2048  | 2262  | 1.10       |
| im1     | im1     | 2048  | 2262  | 1.10       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1240  | 0.60       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2262  | 1.10       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 154   | 19.25      |
| im0     | im1     | 8     | 154   | 19.25      |
| im0     | im1     | 128   | 222   | 1.73       |
| im0     | im1     | 128   | 222   | 1.73       |
| im0     | im1     | 1024  | 696   | 0.67       |
| im0     | im1     | 1024  | 696   | 0.67       |
| im0     | im1     | 2048  | 1240  | 0.60       |
| im0     | im1     | 2048  | 1240  | 0.60       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 154   | 19.25      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1240  | 0.60       |

См. определение в файле vInit.h строка 628

#### 6.232.2.6 nmppsCopy\_4s()

```
void nmppsCopy_4s (
    const nm4s * pSrcVec,
    nm4s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 240   | 0.11       |
| im0     | im1     | 2048  | 240   | 0.11       |
| im0     | im1     | 2048  | 240   | 0.11       |
| im0     | im1     | 2048  | 240   | 0.11       |
| im1     | im1     | 2048  | 365   | 0.17       |
| im1     | im1     | 2048  | 365   | 0.17       |
| im1     | im1     | 2048  | 365   | 0.17       |
| im1     | im1     | 2048  | 365   | 0.17       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 240   | 0.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 365   | 0.17       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 128   | 112   | 0.87       |
| im0     | im1     | 128   | 112   | 0.87       |
| im0     | im1     | 1024  | 173   | 0.16       |
| im0     | im1     | 1024  | 173   | 0.16       |
| im0     | im1     | 2048  | 240   | 0.11       |
| im0     | im1     | 2048  | 240   | 0.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 240   | 0.11       |

#### 6.232.2.7 nmppsCopy\_4u()

```
__INLINE__ void nmppsCopy_4u (
    const nm4u * pSrcVec,
    nm4u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 289   | 0.14       |
| im0     | im1     | 2048  | 289   | 0.14       |
| im0     | im1     | 2048  | 289   | 0.14       |



| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 289   | 0.14       |
| im1     | im1     | 2048  | 414   | 0.20       |
| im1     | im1     | 2048  | 414   | 0.20       |
| im1     | im1     | 2048  | 414   | 0.20       |
| im1     | im1     | 2048  | 414   | 0.20       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 289   | 0.14       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 414   | 0.20       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 128   | 161   | 1.25       |
| im0     | im1     | 128   | 161   | 1.25       |
| im0     | im1     | 1024  | 222   | 0.21       |
| im0     | im1     | 1024  | 222   | 0.21       |
| im0     | im1     | 2048  | 289   | 0.14       |
| im0     | im1     | 2048  | 289   | 0.14       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 289   | 0.14       |

См. определение в файле vInit.h строка 625

#### 6.232.2.8 nmppsCopy\_64s()

```
void nmppsCopy_64s (
    const nm64s * pSrcVec,
```

```
nm64s * pDstVec,
int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2279  | 1.11       |
| im0     | im1     | 2048  | 2279  | 1.11       |
| im0     | im1     | 2048  | 2279  | 1.11       |
| im0     | im1     | 2048  | 2279  | 1.11       |
| im1     | im1     | 2048  | 4325  | 2.11       |
| im1     | im1     | 2048  | 4325  | 2.11       |
| im1     | im1     | 2048  | 4325  | 2.11       |
| im1     | im1     | 2048  | 4325  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2279  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4325  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 112   | 14.00      |
| im0     | im1     | 8     | 112   | 14.00      |
| im0     | im1     | 128   | 240   | 1.87       |
| im0     | im1     | 128   | 240   | 1.87       |
| im0     | im1     | 1024  | 1191  | 1.16       |
| im0     | im1     | 1024  | 1191  | 1.16       |
| im0     | im1     | 2048  | 2279  | 1.11       |
| im0     | im1     | 2048  | 2279  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 112   | 14.00      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2279  | 1.11       |

## 6.232.2.9 nmppsCopy\_64sc()

```
__INLINE__ void nmppsCopy_64sc (
    const nm64sc * pSrcVec,
    nm64sc * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 4553  | 2.22       |
| im0     | im1     | 2048  | 4553  | 2.22       |
| im0     | im1     | 2048  | 4553  | 2.22       |
| im0     | im1     | 2048  | 4553  | 2.22       |
| im1     | im1     | 2048  | 8647  | 4.22       |
| im1     | im1     | 2048  | 8647  | 4.22       |
| im1     | im1     | 2048  | 8647  | 4.22       |
| im1     | im1     | 2048  | 8647  | 4.22       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 4553  | 2.22       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 8647  | 4.22       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 218   | 27.25      |
| im0     | im1     | 8     | 218   | 27.25      |
| im0     | im1     | 128   | 475   | 3.71       |
| im0     | im1     | 128   | 475   | 3.71       |
| im0     | im1     | 1024  | 2377  | 2.32       |
| im0     | im1     | 1024  | 2377  | 2.32       |
| im0     | im1     | 2048  | 4553  | 2.22       |
| im0     | im1     | 2048  | 4553  | 2.22       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 218   | 27.25      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 4553  | 2.22       |

См. определение в файле vInit.h строка 630

#### 6.232.2.10 nmppsCopy\_64u()

```
__INLINE__ void nmppsCopy_64u (
    const nm64u * pSrcVec,
    nm64u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2328  | 1.13       |
| im0     | im1     | 2048  | 2328  | 1.13       |
| im0     | im1     | 2048  | 2328  | 1.13       |
| im0     | im1     | 2048  | 2328  | 1.13       |
| im1     | im1     | 2048  | 4374  | 2.13       |
| im1     | im1     | 2048  | 4374  | 2.13       |
| im1     | im1     | 2048  | 4374  | 2.13       |
| im1     | im1     | 2048  | 4374  | 2.13       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2328  | 1.13       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4374  | 2.13       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 161   | 20.12      |
| im0     | im1     | 8     | 161   | 20.12      |
| im0     | im1     | 128   | 289   | 2.25       |
| im0     | im1     | 128   | 289   | 2.25       |
| im0     | im1     | 1024  | 1240  | 1.21       |
| im0     | im1     | 1024  | 1240  | 1.21       |
| im0     | im1     | 2048  | 2328  | 1.13       |
| im0     | im1     | 2048  | 2328  | 1.13       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 161   | 20.12      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2328  | 1.13       |

См. определение в файле vInit.h строка 629

#### 6.232.2.11 nmppsCopy\_8s()

```
void nmppsCopy_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 377   | 0.18       |
| im0     | im1     | 2048  | 377   | 0.18       |
| im0     | im1     | 2048  | 377   | 0.18       |
| im0     | im1     | 2048  | 377   | 0.18       |
| im1     | im1     | 2048  | 629   | 0.30       |
| im1     | im1     | 2048  | 629   | 0.30       |
| im1     | im1     | 2048  | 629   | 0.30       |
| im1     | im1     | 2048  | 629   | 0.30       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 377   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 629   | 0.30       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |
| im0     | im1     | 128   | 120   | 0.93       |
| im0     | im1     | 128   | 120   | 0.93       |
| im0     | im1     | 1024  | 240   | 0.23       |
| im0     | im1     | 1024  | 240   | 0.23       |
| im0     | im1     | 2048  | 377   | 0.18       |
| im0     | im1     | 2048  | 377   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 102   | 12.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 377   | 0.18       |

#### 6.232.2.12 nmppsCopy\_8u()

```
__INLINE__ void nmppsCopy_8u (
    const nm8u * pSrcVec,
    nm8u * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 426   | 0.20       |
| im0     | im1     | 2048  | 426   | 0.20       |
| im0     | im1     | 2048  | 426   | 0.20       |
| im0     | im1     | 2048  | 426   | 0.20       |
| im1     | im1     | 2048  | 678   | 0.33       |
| im1     | im1     | 2048  | 678   | 0.33       |
| im1     | im1     | 2048  | 678   | 0.33       |
| im1     | im1     | 2048  | 678   | 0.33       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 426   | 0.20       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 678   | 0.33       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 8     | 151   | 18.87      |
| im0     | im1     | 128   | 169   | 1.32       |
| im0     | im1     | 128   | 169   | 1.32       |
| im0     | im1     | 1024  | 289   | 0.28       |
| im0     | im1     | 1024  | 289   | 0.28       |
| im0     | im1     | 2048  | 426   | 0.20       |
| im0     | im1     | 2048  | 426   | 0.20       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 151   | 18.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 426   | 0.20       |

См. определение в файле vInit.h строка 626

## 6.233 nmppsCopyua\_

Копирование вектора с невыровненной байтовой позиции в выровненную.

Функции

- void `nmppsCopyua_8s` (const `nm8s` \*pSrcVec, int nSrcOffset, `nm8s` \*pDstVec, int nSize)

### 6.233.1 Подробное описание

Копирование вектора с невыровненной байтовой позиции в выровненную.

$$pDstVec[i] = pSrcVec[nSrcOffset + i]$$

$$i = \overline{0 \dots nSize - 1}$$

Позиция байта считается выровненной если она совпадает с границей 64р. слов в памяти.

\param pSrcVec

Входной вектор.

Аргументы

|            |                                                                                              |
|------------|----------------------------------------------------------------------------------------------|
| pDstVec    | Результирующий вектор.                                                                       |
| nSrcOffset | Смещение в элементах относительно начала вектора. nSrcOffset Может принимать любое значение. |
| nSize      | Кол-во копируемых элементов.                                                                 |

Возвращает

void

### 6.233.2 Функции



## 6.233.2.1 nmppsCopyua\_8s()

```
void nmppsCopyua_8s (
    const nm8s * pSrcVec,
    int nSrcOffset,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im0     | 3          | 10240 | 3967  | 0.38       |
| im0     | im0     | 3          | 10240 | 3967  | 0.38       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |
| im1     | im0     | 3          | 10240 | 3494  | 0.34       |
| im1     | im0     | 3          | 10240 | 3494  | 0.34       |
| im1     | im1     | 3          | 10240 | 4774  | 0.46       |
| im1     | im1     | 3          | 10240 | 4774  | 0.46       |

The best configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |

The worst configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im1     | im1     | 3          | 10240 | 4774  | 0.46       |

Perfomance table 1

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 0          | 10240 | 2804  | 0.27       |
| im0     | im1     | 0          | 10240 | 2804  | 0.27       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |
| im0     | im1     | 7          | 10240 | 2804  | 0.27       |
| im0     | im1     | 7          | 10240 | 2804  | 0.27       |

The best configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 0          | 10240 | 2804  | 0.27       |

The worst configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 0          | 10240 | 2804  | 0.27       |

Perfomance table 2

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 3          | 8     | 177   | 22.12      |
| im0     | im1     | 3          | 8     | 177   | 22.12      |
| im0     | im1     | 3          | 128   | 217   | 1.69       |
| im0     | im1     | 3          | 128   | 217   | 1.69       |
| im0     | im1     | 3          | 1024  | 464   | 0.45       |
| im0     | im1     | 3          | 1024  | 464   | 0.45       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |

The best configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 3          | 8     | 177   | 22.12      |

The worst configuration:

| pSrcVec | pDstVec | nSrcOffset | nSize | ticks | ticks/elem |
|---------|---------|------------|-------|-------|------------|
| im0     | im1     | 3          | 10240 | 2804  | 0.27       |

## 6.234 nmppsSwap\_

Перестановка двух векторов.

Функции

- void `nmppsSwap_64s` (`nm64s` \*pSrcVec1, `nm64s` \*pSrcVec2, int nSize)

### 6.234.1 Подробное описание

Перестановка двух векторов.

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращает

void

### 6.234.2 Функции

#### 6.234.2.1 nmppsSwap\_64s()

```
void nmppsSwap_64s (  
    nm64s * pSrcVec1,  
    nm64s * pSrcVec2,  
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im0      | 2048  | 8266  | 4.03       |
| im0      | im0      | 2048  | 8266  | 4.03       |
| im0      | im1      | 2048  | 4733  | 2.31       |
| im0      | im1      | 2048  | 4733  | 2.31       |
| im1      | im0      | 2048  | 4584  | 2.23       |
| im1      | im0      | 2048  | 4584  | 2.23       |
| im1      | im1      | 2048  | 8491  | 4.14       |
| im1      | im1      | 2048  | 8491  | 4.14       |

The best configuration:

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im0      | 2048  | 4584  | 2.23       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im1      | im1      | 2048  | 8491  | 4.14       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 133   | 16.62      |
| im0      | im1      | 8     | 133   | 16.62      |
| im0      | im1      | 128   | 413   | 3.22       |
| im0      | im1      | 128   | 413   | 3.22       |
| im0      | im1      | 1024  | 2429  | 2.37       |
| im0      | im1      | 1024  | 2429  | 2.37       |
| im0      | im1      | 2048  | 4733  | 2.31       |
| im0      | im1      | 2048  | 4733  | 2.31       |

The best configuration:

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 8     | 133   | 16.62      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | nSize | ticks | ticks/elem |
|----------|----------|-------|-------|------------|
| im0      | im1      | 2048  | 4733  | 2.31       |

## 6.235 nmppsMax\_

Поиск значения максимального элемента вектора.

## Функции

- void nmppsMax\_8s7b (const nm8s7b \*pSrcVec, int nSize, int8b \*nMaxValue)
- void nmppsMax\_16s15b (const nm16s15b \*pSrcVec, int nSize, int16b \*nMaxValue)
- void nmppsMax\_32s31b (const nm32s31b \*pSrcVec, int nSize, int \*nMaxValue)
- void nmppsMax\_64s63b (const nm64s63b \*pSrcVec, int nSize, int64b \*nMaxValue)
- int nmppsMax\_8sm (const nm8s \*srcVec, int size, int8b \*maxValue, nm16s \*tmp)
- int nmppsMax\_16sm (const nm16s \*srcVec, int size, int16b \*maxValue, nm32s \*tmp)
- int nmppsMax\_32sm (const nm32s \*srcVec, int size, int32b \*maxValue, nm64s \*tmp)

## 6.235.1 Подробное описание

Поиск значения максимального элемента вектора.

$$nMaxValue = \max_i (pSrcVec[i])$$

## Аргументы

|         |                             |
|---------|-----------------------------|
| pSrcVec | Входной вектор.             |
| nSize   | Размер вектора в элементах. |

## Возвращаемые значения

|           |                                         |
|-----------|-----------------------------------------|
| nMaxValue | значение максимального элемент вектора. |
|-----------|-----------------------------------------|

## Возвращает

void

## Restrictions:

Ограничения на параметры приводятся в описании каждой из функций.

## 6.235.2 Функции

## 6.235.2.1 nmppsMax\_16s15b()

```
void nmppsMax_16s15b (
    const nm16s15b * pSrcVec,
    int nSize,
    int16b * nMaxValue )
```

Restrictions:

Максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{15}-1$ .  
Примеры допустимых диапазонов входных чисел:

## 6.235.2.2 nmppsMax\_32s31b()

```
void nmppsMax_32s31b (
    const nm32s31b * pSrcVec,
    int nSize,
    int * nMaxValue )
```

Restrictions:

Максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{31}-1$ .  
Примеры допустимых диапазонов входных чисел:

$[00000000h..7FFFFFFFh] = [0..+2^{31}-1]$   $[FFFFFFFFh..7FFFFFFEh] = [-1..+2^{31}-2]$   $[C0000000h..3\leftarrow FFFFFFFFh] = [-2^{30}..+2^{30}-1]$   $[80000000h..00000000h] = [-2^{31}..0]$

Perfomance table 0

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 3394  | 1.65       |
| im0     | im1       | 2048  | 3394  | 1.65       |
| im1     | im1       | 2048  | 3401  | 1.66       |
| im1     | im1       | 2048  | 3401  | 1.66       |

The best configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 3394  | 1.65       |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 3401  | 1.66       |

Perfomance table 1

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 216   | 27.00      |
| im0     | im1       | 8     | 216   | 27.00      |
| im0     | im1       | 128   | 514   | 4.01       |
| im0     | im1       | 128   | 514   | 4.01       |
| im0     | im1       | 1024  | 1858  | 1.81       |
| im0     | im1       | 1024  | 1858  | 1.81       |
| im0     | im1       | 2048  | 3394  | 1.65       |
| im0     | im1       | 2048  | 3394  | 1.65       |

The best configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 216   | 27.00      |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 3394  | 1.65       |

### 6.235.2.3 nmppsMax\_64s63b()

```
void nmppsMax_64s63b (
    const nm64s63b * pSrcVec,
    int nSize,
    int64b * nMaxValue )
```

Perfomance table 0

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 6453  | 3.15       |
| im0     | im1       | 2048  | 6453  | 3.15       |
| im1     | im1       | 2048  | 6460  | 3.15       |
| im1     | im1       | 2048  | 6460  | 3.15       |

The best configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 6453  | 3.15       |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 6460  | 3.15       |

Perfomance table 1

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 274   | 34.25      |
| im0     | im1       | 8     | 274   | 34.25      |
| im0     | im1       | 128   | 693   | 5.41       |
| im0     | im1       | 128   | 693   | 5.41       |
| im0     | im1       | 1024  | 3381  | 3.30       |
| im0     | im1       | 1024  | 3381  | 3.30       |
| im0     | im1       | 2048  | 6453  | 3.15       |
| im0     | im1       | 2048  | 6453  | 3.15       |

The best configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 274   | 34.25      |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 6453  | 3.15       |

#### 6.235.2.4 nmppsMax\_8s7b()

```
void nmppsMax_8s7b (
    const nm8s7b * pSrcVec,
    int nSize,
    int8b * nMaxValue )
```

Restrictions:

Физический размер вектора должен быть кратен блоку из 32-х 64р. слов

Максимальный и минимальный элементы массива должны отличаться не более чем на 127.

Примеры допустимых диапазонов входных чисел:

[00h..7Fh]=[ 0..+127] [FFh..7Eh]=[ -1..+126] [C0h..3Fh]=[-64..+63] [80h..00h]=[-128..0 ]

Perfomance table 0

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 1152  | 0.56       |
| im0     | im1       | 2048  | 1152  | 0.56       |
| im1     | im1       | 2048  | 1159  | 0.56       |
| im1     | im1       | 2048  | 1159  | 0.56       |

The best configuration:



| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 1152  | 0.56       |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 1159  | 0.56       |

Perfomance table 1

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 224   | 28.00      |
| im0     | im1       | 8     | 224   | 28.00      |
| im0     | im1       | 128   | 539   | 4.21       |
| im0     | im1       | 128   | 539   | 4.21       |
| im0     | im1       | 1024  | 780   | 0.76       |
| im0     | im1       | 1024  | 780   | 0.76       |
| im0     | im1       | 2048  | 1152  | 0.56       |
| im0     | im1       | 2048  | 1152  | 0.56       |

The best configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 8     | 224   | 28.00      |

The worst configuration:

| pSrcVec | nMaxValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 1152  | 0.56       |

## 6.236 nmppsMin

Поиск значения минимального элемента вектора.

### Функции

- void nmppsMin\_8s7b (const nm8s7b \*pSrcVec, int nSize, int8b \*nMinValue)
- void nmppsMin\_16s15b (const nm16s15b \*pSrcVec, int nSize, int16b \*nMinValue)
- void nmppsMin\_32s31b (const nm32s31b \*pSrcVec, int nSize, int \*nMinValue)
- void nmppsMin\_64s63b (const nm64s63b \*pSrcVec, int nSize, int64b \*nMinValue)
- int nmppsMin\_8sm (const nm8s \*srcVec, int size, int8b \*minValue, nm16s \*tmp)
- int nmppsMin\_16sm (const nm16s \*srcVec, int size, int16b \*minValue, nm32s \*tmp)
- int nmppsMin\_32sm (const nm32s \*srcVec, int size, int32b \*minValue, nm64s \*tmp)

### 6.236.1 Подробное описание

Поиск значения минимального элемента вектора.

$$nMinValue = \min_i (pSrcVec[i])$$

### Аргументы

|         |                             |
|---------|-----------------------------|
| pSrcVec | Входной вектор.             |
| nSize   | Размер вектора в элементах. |

### Возвращаемые значения

|           |                                |
|-----------|--------------------------------|
| nMinValue | значение минимального элемент. |
|-----------|--------------------------------|

### Возвращает

void

### Restrictions:

Ограничения на параметры приводятся в описании каждой из функций.

### 6.236.2 Функции

## 6.236.2.1 nmppsMin\_16s15b()

```
void nmppsMin_16s15b (
    const nm16s15b * pSrcVec,
    int nSize,
    int16b * nMinValue )
```

## Restrictions:

Максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{15}-1$ .

Примеры допустимых диапазонов входных чисел:

[0000h..7FFFh]=[ 0..+32767 ] [FFFFh..7FFEh]=[ -1..+32766 ] [C000h..3FFFh]=[ -16384..+16383 ]  
[8000h..0000h]=[ -32768..0 ]

## 6.236.2.2 nmppsMin\_32s31b()

```
void nmppsMin_32s31b (
    const nm32s31b * pSrcVec,
    int nSize,
    int * nMinValue )
```

## Restrictions:

Максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{31}-1$ .

Примеры допустимых диапазонов входных чисел:

[00000000h..7FFFFFFFh]=[ 0..+ $2^{31}-1$  ] [FFFFFFFFh..7FFFFFFEh]=[ -1..+ $2^{31}-2$  ] [C0000000h..3←  
FFFFFFFh]=[ - $2^{30}$ ..+ $2^{30}-1$  ] [80000000h..00000000h]=[ - $2^{31}$ ..0 ]

## Perfomance table 0

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 3394  | 1.65       |
| im0     | im1       | 2048  | 3394  | 1.65       |
| im1     | im1       | 2048  | 3403  | 1.66       |
| im1     | im1       | 2048  | 3403  | 1.66       |

## The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 3394  | 1.65       |

## The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 3403  | 1.66       |

## Perfomance table 1

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 216   | 27.00      |
| im0     | im2       | 8     | 216   | 27.00      |
| im0     | im2       | 128   | 514   | 4.01       |
| im0     | im2       | 128   | 514   | 4.01       |
| im0     | im2       | 1024  | 1858  | 1.81       |
| im0     | im2       | 1024  | 1858  | 1.81       |
| im0     | im2       | 2048  | 3394  | 1.65       |
| im0     | im2       | 2048  | 3394  | 1.65       |

The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 216   | 27.00      |

The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 2048  | 3394  | 1.65       |

### 6.236.2.3 nmppsMin\_64s63b()

```
void nmppsMin_64s63b (
    const nm64s63b * pSrcVec,
    int nSize,
    int64b * nMinValue )
```

Perfomance table 0

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 6451  | 3.14       |
| im0     | im1       | 2048  | 6451  | 3.14       |
| im1     | im1       | 2048  | 6460  | 3.15       |
| im1     | im1       | 2048  | 6460  | 3.15       |

The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 6451  | 3.14       |

The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 6460  | 3.15       |

Perfomance table 1

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 274   | 34.25      |
| im0     | im2       | 8     | 274   | 34.25      |
| im0     | im2       | 128   | 691   | 5.39       |
| im0     | im2       | 128   | 691   | 5.39       |
| im0     | im2       | 1024  | 3379  | 3.29       |
| im0     | im2       | 1024  | 3379  | 3.29       |
| im0     | im2       | 2048  | 6451  | 3.14       |
| im0     | im2       | 2048  | 6451  | 3.14       |

The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 274   | 34.25      |

The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 2048  | 6451  | 3.14       |

#### 6.236.2.4 nmppsMin\_8s7b()

```
void nmppsMin_8s7b (
    const nm8s7b * pSrcVec,
    int nSize,
    int8b * nMinValue )
```

Restrictions:

Максимальный и минимальный элементы массива должны отличаться не более чем на 127.

Примеры допустимых диапазонов входных чисел:

[00h..7Fh]=[ 0..+127] [FFh..7Eh]=[ -1..+126] [C0h..3Fh]=[-64..+63] [80h..00h]=[-128..0 ]

Perfomance table 0

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 1150  | 0.56       |
| im0     | im1       | 2048  | 1150  | 0.56       |
| im1     | im1       | 2048  | 1159  | 0.56       |
| im1     | im1       | 2048  | 1159  | 0.56       |

The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im1       | 2048  | 1150  | 0.56       |

The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im1     | im1       | 2048  | 1159  | 0.56       |

Perfomance table 1

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 228   | 28.50      |
| im0     | im2       | 8     | 228   | 28.50      |
| im0     | im2       | 128   | 543   | 4.24       |
| im0     | im2       | 128   | 543   | 4.24       |
| im0     | im2       | 1024  | 782   | 0.76       |
| im0     | im2       | 1024  | 782   | 0.76       |
| im0     | im2       | 2048  | 1150  | 0.56       |
| im0     | im2       | 2048  | 1150  | 0.56       |

The best configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 8     | 228   | 28.50      |

The worst configuration:

| pSrcVec | nMinValue | nSize | ticks | ticks/elem |
|---------|-----------|-------|-------|------------|
| im0     | im2       | 2048  | 1150  | 0.56       |

## 6.237 nmppsMaxIndx\_

Поиск значения максимального элемента вектора и его положения (положений) в векторе.

## Функции

- void nmppsMaxIndx\_8s (nm8s7b \*pSrcVec, int nSize, int \*nIndex, int8b \*nMaxValue, void \*pL←TmpBuf, void \*pGTmpBuf, int nSearchDir)
- void nmppsMaxIndx\_16s (nm16s15b \*pSrcVec, int nSize, int \*nIndex, int16b \*nMaxValue, void \*pLTmpBuf, void \*pGTmpBuf, int nSearchDir)
- void nmppsMaxIndx\_32s (nm32s31b \*pSrcVec, int nSize, int \*nIndex, int32b \*nMaxValue, void \*pLTmpBuf, void \*pGTmpBuf, int nSearchDir)

## 6.237.1 Подробное описание

Поиск значения максимального элемента вектора и его положения (положений) в векторе.

$$nMaxValue = \max_i (pSrcVec[i])$$

## Аргументы

|            |                                                                                                                                                                                   |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pSrcVec    | Входной вектор.                                                                                                                                                                   |
| nSize      | Размер вектора в элементах. Занимаемый этим вектором объем памяти должен быть кратен 64 длинным словам (nm64s[64,128,...]).                                                       |
| pLTmpBuf   | Временный массив на локальной шине из nSize элементов.                                                                                                                            |
| pGTmpBuf   | Временный массив на глобальной шине .                                                                                                                                             |
| nSearchDir | Направление поиска максимума.<br>При nSearchDir = 1, поиск ведется от начала массива.<br>При nSearchDir = -1, поиск ведется от конца массива.<br>Значение максимального элемента. |

## Возвращаемые значения

|        |                                                   |
|--------|---------------------------------------------------|
| nIndex | Индекс первого найденного максимума среди равных. |
|--------|---------------------------------------------------|

## Возвращает

void

## Restrictions:

Ограничения на параметры приводятся в описании каждой из функций.

## Restrictions:

Диапазоны входных элементов могут быть "плавающими"  
Например для данных nm16s15b максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{15}-1$ . Примеры допустимых диапазонов входных чисел для типа nm16s15b :

Here are some examples of admissible ranges for input numbers: \n

[0000h..7FFFh]=[ 0..+32767 ] [FFFFh..7FFEh]=[ -1..+32766 ] [C000h..3FFFh]=[-16384..+16383 ]  
[8000h..0000h]=[-32768..0 ]

## 6.237.2 Функции

### 6.237.2.1 nmppsMaxIndx\_32s()

```
void nmppsMaxIndx_32s (
    nm32s31b * pSrcVec,
    int nSize,
    int * nIndex,
    int32b * nMaxValue,
    void * pLTmpBuf,
    void * pGTmpBuf,
    int nSearchDir )
```

Perfomance table 0

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 2271  | 35.48      |
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 2271  | 35.48      |
| im0     | im2           | im3           | im0            | im3    | 1               | 1024  | 3962  | 3.86       |
| im0     | im2           | im3           | im0            | im3    | 1               | 1024  | 3962  | 3.86       |
| im0     | im2           | im3           | im0            | im3    | 1               | 2048  | 5766  | 2.81       |
| im0     | im2           | im3           | im0            | im3    | 1               | 2048  | 5766  | 2.81       |
| im0     | im2           | im3           | im0            | im3    | -1              | 64    | 2288  | 35.75      |
| im0     | im2           | im3           | im0            | im3    | -1              | 64    | 2288  | 35.75      |
| im0     | im2           | im3           | im0            | im3    | -1              | 1024  | 3979  | 3.88       |
| im0     | im2           | im3           | im0            | im3    | -1              | 1024  | 3979  | 3.88       |
| im0     | im2           | im3           | im0            | im3    | -1              | 2048  | 5772  | 2.81       |
| im0     | im2           | im3           | im0            | im3    | -1              | 2048  | 5772  | 2.81       |
| im1     | im2           | im3           | im0            | im3    | 1               | 64    | 3427  | 53.54      |
| im1     | im2           | im3           | im0            | im3    | 1               | 64    | 3427  | 53.54      |
| im1     | im2           | im3           | im0            | im3    | 1               | 1024  | 5419  | 5.29       |
| im1     | im2           | im3           | im0            | im3    | 1               | 1024  | 5419  | 5.29       |
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 7511  | 3.66       |
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 7511  | 3.66       |
| im1     | im2           | im3           | im0            | im3    | -1              | 64    | 2320  | 36.25      |
| im1     | im2           | im3           | im0            | im3    | -1              | 64    | 2320  | 36.25      |
| im1     | im2           | im3           | im0            | im3    | -1              | 1024  | 4072  | 3.97       |
| im1     | im2           | im3           | im0            | im3    | -1              | 1024  | 4072  | 3.97       |
| im1     | im2           | im3           | im0            | im3    | -1              | 2048  | 5897  | 2.87       |
| im1     | im2           | im3           | im0            | im3    | -1              | 2048  | 5897  | 2.87       |



The best configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 2271  | 35.48      |

The worst configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 7511  | 3.66       |

### 6.237.2.2 nmppsMaxIndx\_8s()

```
void nmppsMaxIndx_8s (
    nm8s7b * pSrcVec,
    int nSize,
    int * nIndex,
    int8b * nMaxValue,
    void * pLTmpBuf,
    void * pGTmpBuf,
    int nSearchDir )
```

Perfomance table 0

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 1035  | 16.17      |
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 1035  | 16.17      |
| im0     | im2           | im3           | im0            | im3    | 1               | 1024  | 1721  | 1.68       |
| im0     | im2           | im3           | im0            | im3    | 1               | 1024  | 1721  | 1.68       |
| im0     | im2           | im3           | im0            | im3    | 1               | 2048  | 1989  | 0.97       |
| im0     | im2           | im3           | im0            | im3    | 1               | 2048  | 1989  | 0.97       |
| im0     | im2           | im3           | im0            | im3    | -1              | 64    | 1052  | 16.43      |
| im0     | im2           | im3           | im0            | im3    | -1              | 64    | 1052  | 16.43      |
| im0     | im2           | im3           | im0            | im3    | -1              | 1024  | 1738  | 1.69       |
| im0     | im2           | im3           | im0            | im3    | -1              | 1024  | 1738  | 1.69       |
| im0     | im2           | im3           | im0            | im3    | -1              | 2048  | 1995  | 0.97       |
| im0     | im2           | im3           | im0            | im3    | -1              | 2048  | 1995  | 0.97       |
| im1     | im2           | im3           | im0            | im3    | 1               | 64    | 2168  | 33.87      |
| im1     | im2           | im3           | im0            | im3    | 1               | 64    | 2168  | 33.87      |
| im1     | im2           | im3           | im0            | im3    | 1               | 1024  | 3152  | 3.07       |
| im1     | im2           | im3           | im0            | im3    | 1               | 1024  | 3152  | 3.07       |
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 3686  | 1.79       |
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 3686  | 1.79       |
| im1     | im2           | im3           | im0            | im3    | -1              | 64    | 1061  | 16.57      |
| im1     | im2           | im3           | im0            | im3    | -1              | 64    | 1061  | 16.57      |
| im1     | im2           | im3           | im0            | im3    | -1              | 1024  | 1805  | 1.76       |
| im1     | im2           | im3           | im0            | im3    | -1              | 1024  | 1805  | 1.76       |
| im1     | im2           | im3           | im0            | im3    | -1              | 2048  | 2072  | 1.01       |
| im1     | im2           | im3           | im0            | im3    | -1              | 2048  | 2072  | 1.01       |

The best configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im2           | im3           | im0            | im3    | 1               | 64    | 1035  | 16.17      |

The worst configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMax↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im1     | im2           | im3           | im0            | im3    | 1               | 2048  | 3686  | 1.79       |

## 6.238 nmppsMinIndx\_

Поиск значения минимального элемента вектора и его положения (положений) в векторе.

## Функции

- void nmppsMinIndx\_8s (nm8s7b \*pSrcVec, int nSize, int \*nIndex, int8b \*nMinValue, void \*pLTmpBuf, void \*pGTmpBuf, int nSearchDir)
- void nmppsMinIndx\_16s (nm16s15b \*pSrcVec, int nSize, int \*nIndex, int16b \*nMinValue, void \*pLTmpBuf, void \*pGTmpBuf, int nSearchDir)
- void nmppsMinIndx\_32s (nm32s31b \*pSrcVec, int nSize, int \*nIndex, int32b \*nMinValue, void \*pLTmpBuf, void \*pGTmpBuf, int nSearchDir)

## 6.238.1 Подробное описание

Поиск значения минимального элемента вектора и его положения (положений) в векторе.

$$nMinValue = \min_i (pSrcVec[i])$$

## Аргументы

|            |                                                                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pSrcVec    | Входной вектор.                                                                                                                                                                 |
| nSize      | Размер вектора в элементах. Занимаемый этим вектором объем памяти должен быть кратен 64 длинным словам (nm64s[64,128,...]).                                                     |
| pLTmpBuf   | Временный массив на локальной шине .                                                                                                                                            |
| pGTmpBuf   | Временный массив на глобальной шине из nSize элементов.                                                                                                                         |
| nSearchDir | Направление поиска минимума.<br>При nSearchDir = 1, поиск ведется от начала массива.<br>При nSearchDir = -1, поиск ведется от конца массива.<br>Значение минимального элемента. |

## Возвращаемые значения

|        |                                                  |
|--------|--------------------------------------------------|
| nIndex | Индекс первого найденного минимума среди равных. |
|--------|--------------------------------------------------|

## Возвращает

void

## Restrictions:

Ограничения на параметры приводятся в описании каждой из функций.

## Restrictions:

Диапазоны входных элементов могут быть "плавающими"  
Например для данных nm16s15b максимальный и минимальный элементы массива должны отличаться не более чем на  $2^{15}-1$ . Примеры допустимых диапазонов входных чисел для типа nm16s15b :

Here are some examples of admissible ranges for input numbers: \n

[0000h..7FFFh]=[ 0..+32767 ] [FFFFh..7FFEh]=[ -1..+32766 ] [C000h..3FFFh]=[-16384..+16383 ]  
[8000h..0000h]=[-32768..0 ]

## 6.238.2 Функции

### 6.238.2.1 nmppsMinIndx\_32s()

```
void nmppsMinIndx_32s (
    nm32s31b * pSrcVec,
    int nSize,
    int * nIndex,
    int32b * nMinValue,
    void * pLTmpBuf,
    void * pGTmpBuf,
    int nSearchDir )
```

Perfomance table 0

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im1           | im2           | 5              | 5      | 1               | 64    | 3397  | 53.07      |
| im0     | im1           | im2           | 5              | 5      | 1               | 64    | 3397  | 53.07      |
| im0     | im1           | im2           | 5              | 5      | 1               | 1024  | 5328  | 5.20       |
| im0     | im1           | im2           | 5              | 5      | 1               | 1024  | 5328  | 5.20       |
| im0     | im1           | im2           | 5              | 5      | 1               | 2048  | 7437  | 3.63       |
| im0     | im1           | im2           | 5              | 5      | 1               | 2048  | 7437  | 3.63       |
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 2290  | 35.78      |
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 2290  | 35.78      |
| im0     | im1           | im2           | 5              | 5      | -1              | 1024  | 3981  | 3.88       |
| im0     | im1           | im2           | 5              | 5      | -1              | 1024  | 3981  | 3.88       |
| im0     | im1           | im2           | 5              | 5      | -1              | 2048  | 5834  | 2.84       |
| im0     | im1           | im2           | 5              | 5      | -1              | 2048  | 5834  | 2.84       |
| im1     | im1           | im2           | 5              | 5      | 1               | 64    | 3429  | 53.57      |
| im1     | im1           | im2           | 5              | 5      | 1               | 64    | 3429  | 53.57      |
| im1     | im1           | im2           | 5              | 5      | 1               | 1024  | 5423  | 5.29       |
| im1     | im1           | im2           | 5              | 5      | 1               | 1024  | 5423  | 5.29       |
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 7564  | 3.69       |
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 7564  | 3.69       |
| im1     | im1           | im2           | 5              | 5      | -1              | 64    | 2322  | 36.28      |
| im1     | im1           | im2           | 5              | 5      | -1              | 64    | 2322  | 36.28      |
| im1     | im1           | im2           | 5              | 5      | -1              | 1024  | 4076  | 3.98       |
| im1     | im1           | im2           | 5              | 5      | -1              | 1024  | 4076  | 3.98       |
| im1     | im1           | im2           | 5              | 5      | -1              | 2048  | 5961  | 2.91       |
| im1     | im1           | im2           | 5              | 5      | -1              | 2048  | 5961  | 2.91       |

The best configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 2290  | 35.78      |

The worst configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 7564  | 3.69       |

### 6.238.2.2 nmppsMinIndx\_8s()

```
void nmppsMinIndx_8s (
    nm8s7b * pSrcVec,
    int nSize,
    int * nIndex,
    int8b * nMinValue,
    void * pLTmpBuf,
    void * pGTmpBuf,
    int nSearchDir )
```

Perfomance table 0

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im1           | im2           | 5              | 5      | 1               | 64    | 2161  | 33.76      |
| im0     | im1           | im2           | 5              | 5      | 1               | 64    | 2161  | 33.76      |
| im0     | im1           | im2           | 5              | 5      | 1               | 1024  | 3085  | 3.01       |
| im0     | im1           | im2           | 5              | 5      | 1               | 1024  | 3085  | 3.01       |
| im0     | im1           | im2           | 5              | 5      | 1               | 2048  | 3658  | 1.78       |
| im0     | im1           | im2           | 5              | 5      | 1               | 2048  | 3658  | 1.78       |
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 1054  | 16.46      |
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 1054  | 16.46      |
| im0     | im1           | im2           | 5              | 5      | -1              | 1024  | 1738  | 1.69       |
| im0     | im1           | im2           | 5              | 5      | -1              | 1024  | 1738  | 1.69       |
| im0     | im1           | im2           | 5              | 5      | -1              | 2048  | 2055  | 1.00       |
| im0     | im1           | im2           | 5              | 5      | -1              | 2048  | 2055  | 1.00       |
| im1     | im1           | im2           | 5              | 5      | 1               | 64    | 2170  | 33.90      |
| im1     | im1           | im2           | 5              | 5      | 1               | 64    | 2170  | 33.90      |
| im1     | im1           | im2           | 5              | 5      | 1               | 1024  | 3154  | 3.08       |
| im1     | im1           | im2           | 5              | 5      | 1               | 1024  | 3154  | 3.08       |
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 3737  | 1.82       |
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 3737  | 1.82       |
| im1     | im1           | im2           | 5              | 5      | -1              | 64    | 1063  | 16.60      |
| im1     | im1           | im2           | 5              | 5      | -1              | 64    | 1063  | 16.60      |
| im1     | im1           | im2           | 5              | 5      | -1              | 1024  | 1807  | 1.76       |
| im1     | im1           | im2           | 5              | 5      | -1              | 1024  | 1807  | 1.76       |
| im1     | im1           | im2           | 5              | 5      | -1              | 2048  | 2134  | 1.04       |
| im1     | im1           | im2           | 5              | 5      | -1              | 2048  | 2134  | 1.04       |

The best configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im0     | im1           | im2           | 5              | 5      | -1              | 64    | 1054  | 16.46      |

The worst configuration:

| pSrcVec | pL↔<br>TmpBuf | pG↔<br>TmpBuf | nMin↔<br>Value | nIndex | n↔<br>SearchDir | nSize | ticks | ticks/elem |
|---------|---------------|---------------|----------------|--------|-----------------|-------|-------|------------|
| im1     | im1           | im2           | 5              | 5      | 1               | 2048  | 3737  | 1.82       |

## 6.239 nmppsMinIndxVN\_

Поиск значения минимального элемента вектора длины N и его положения в векторе.

### Функции

- int [nmppsMinIndxV9\\_32s](#) (int \*pSrcVec, int nStride, int \*nPos)
- int [nmppsMinIndxV16\\_32s](#) (int \*pSrcVec, int nStride, int \*nPos)
- int [nmppsMinIndxV25\\_32s](#) (int \*pSrcVec, int nStride, int \*nPos)
- int [nmppsMinIndxV256\\_32s](#) (int \*pSrcVec, int nStride, int \*nPos)
- int [nmppsMinIndxV1024\\_32s](#) (int \*pSrcVec, int nStride, int \*nPos)

### 6.239.1 Подробное описание

Поиск значения минимального элемента вектора длины N и его положения в векторе.

#### Аргументы

|         |                                                                   |
|---------|-------------------------------------------------------------------|
| pSrcVec | Массив из N элементов, где N определяется числом в имени функции. |
| nStride | шаг между элементами в 32р. словах                                |

#### Возвращаемые значения

|        |                              |
|--------|------------------------------|
| Индекс | первого найденного минимума. |
|--------|------------------------------|

#### Возвращает

Значение минимального элемента.

#### Restrictions:

Вычисления проиводятся на скалярном ядре, поэтому указатель на данные pSrcVec может ссылаться на нечетный адрес.

### 6.239.2 Функции

#### 6.239.2.1 nmppsMinIndxV1024\_32s()

```
int nmppsMinIndxV1024_32s (  
    int * pSrcVec,  
    int nStride,  
    int * nPos )
```

#### Perfomance table 0

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 8397  | 8397.00    |
| im0     | 2       | im2  | 8397  | 8397.00    |
| im1     | 2       | im2  | 9408  | 9408.00    |
| im1     | 2       | im2  | 9408  | 9408.00    |

The best configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 8397  | 8397.00    |

The worst configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im1     | 2       | im2  | 9408  | 9408.00    |

#### 6.239.2.2 nmppsMinIndxV16\_32s()

```
int nmppsMinIndxV16_32s (
    int * pSrcVec,
    int nStride,
    int * nPos )
```

Perfomance table 0

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 210   | 210.00     |
| im0     | 2       | im2  | 210   | 210.00     |
| im1     | 2       | im2  | 226   | 226.00     |
| im1     | 2       | im2  | 226   | 226.00     |

The best configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 210   | 210.00     |

The worst configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im1     | 2       | im2  | 226   | 226.00     |



## 6.239.2.3 nmppsMinIndxV256\_32s()

```
int nmppsMinIndxV256_32s (
    int * pSrcVec,
    int nStride,
    int * nPos )
```

Perfomance table 0

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 2193  | 2193.00    |
| im0     | 2       | im2  | 2193  | 2193.00    |
| im1     | 2       | im2  | 2453  | 2453.00    |
| im1     | 2       | im2  | 2453  | 2453.00    |

The best configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 2193  | 2193.00    |

The worst configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im1     | 2       | im2  | 2453  | 2453.00    |

## 6.239.2.4 nmppsMinIndxV25\_32s()

```
int nmppsMinIndxV25_32s (
    int * pSrcVec,
    int nStride,
    int * nPos )
```

Perfomance table 0

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 282   | 282.00     |
| im0     | 2       | im2  | 282   | 282.00     |
| im1     | 2       | im2  | 308   | 308.00     |
| im1     | 2       | im2  | 308   | 308.00     |

The best configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 282   | 282.00     |

The worst configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im1     | 2       | im2  | 308   | 308.00     |

#### 6.239.2.5 nmppsMinIndxV9\_32s()

```
int nmppsMinIndxV9_32s (
    int * pSrcVec,
    int nStride,
    int * nPos )
```

Perfomance table 0

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 154   | 154.00     |
| im0     | 2       | im2  | 154   | 154.00     |
| im1     | 2       | im2  | 164   | 164.00     |
| im1     | 2       | im2  | 164   | 164.00     |

The best configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im0     | 2       | im2  | 154   | 154.00     |

The worst configuration:

| pSrcVec | nStride | nPos | ticks | ticks/elem |
|---------|---------|------|-------|------------|
| im1     | 2       | im2  | 164   | 164.00     |

## 6.240 nmppsFirstZeroIndx

Поиск позиции первого нулевого элемента в векторе .

Функции

- int [nmppsFirstZeroIndx\\_32s](#) (int \*pSrcVec, int nSize)

### 6.240.1 Подробное описание

Поиск позиции первого нулевого элемента в векторе .

Аргументы

|         |                 |
|---------|-----------------|
| pSrcVec | Входной массив. |
| nIndex  | Размер массива. |

Возвращает

Позиция нулевого элемента или -1 в случае если нулевой элемент не найден

Restrictions:

Вычисления производятся на скалярном ядре, поэтому указатель на данные pSrcVec может ссылаться на нечетный адрес. Поиск производится до первого нулевого элемента.

### 6.240.2 Функции

#### 6.240.2.1 nmppsFirstZeroIndx\_32s()

```
int nmppsFirstZeroIndx_32s (
    int * pSrcVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 142   | 17.75      |
| im0     | 8     | 142   | 17.75      |
| im0     | 64    | 590   | 9.21       |
| im0     | 64    | 590   | 9.21       |
| im0     | 512   | 4174  | 8.15       |
| im0     | 512   | 4174  | 8.15       |

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 8     | 149   | 18.62      |
| im1     | 8     | 149   | 18.62      |
| im1     | 64    | 653   | 10.20      |
| im1     | 64    | 653   | 10.20      |
| im1     | 512   | 4685  | 9.15       |
| im1     | 512   | 4685  | 9.15       |

The best configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 142   | 17.75      |

The worst configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 512   | 4685  | 9.15       |

## 6.241 nmppsFirstNonZeroIndx

Поиск позиции первого ненулевого элемента в векторе .

Функции

- int [nmppsFirstNonZeroIndx\\_32s](#) (int \*pSrcVec, int nSize)

### 6.241.1 Подробное описание

Поиск позиции первого ненулевого элемента в векторе .

Аргументы

|         |                 |
|---------|-----------------|
| pSrcVec | Входной массив. |
| nIndex  | Размер массива. |

Возвращает

Позиция ненулевого элемента или -1 в случае если ненулевой элемент не найден

Restrictions:

Вычисления производятся на скалярном ядре, поэтому указатель на данные pSrcVec может ссылаться на нечетный адрес. Поиск производится до первого ненулевого элемента.

### 6.241.2 Функции

#### 6.241.2.1 nmppsFirstNonZeroIndx\_32s()

```
int nmppsFirstNonZeroIndx_32s (
    int * pSrcVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 144   | 18.00      |
| im0     | 8     | 144   | 18.00      |
| im0     | 64    | 592   | 9.25       |
| im0     | 64    | 592   | 9.25       |
| im0     | 512   | 4176  | 8.15       |
| im0     | 512   | 4176  | 8.15       |

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 8     | 144   | 18.00      |
| im1     | 8     | 144   | 18.00      |
| im1     | 64    | 592   | 9.25       |
| im1     | 64    | 592   | 9.25       |
| im1     | 512   | 4176  | 8.15       |
| im1     | 512   | 4176  | 8.15       |

The best configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 144   | 18.00      |

The worst configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 512   | 4176  | 8.15       |

## 6.242 nmppsLastZeroIndx

Поиск позиции последнего нулевого элемента в векторе .

Функции

- `int nmppsLastZeroIndx_32s (int *pSrcVec, int nSize)`

### 6.242.1 Подробное описание

Поиск позиции последнего нулевого элемента в векторе .

Аргументы

|                      |                 |
|----------------------|-----------------|
| <code>pSrcVec</code> | Входной массив. |
| <code>nIndex</code>  | Размер массива. |

Возвращает

Позиция нулевого элемента или -1 в случае если нулевой элемент не найден

Restrictions:

Вычисления проиводятся на скалярном ядре, поэтому указатель на данные `pSrcVec` может ссылаться на нечетный адрес. Поиск производится с конца до первого нулевого элемента.

## 6.243 nmppsLastNonZeroIndx

Поиск позиции последнего ненулевого элемента в векторе .

Функции

- int [nmppsLastNonZeroIndx\\_32s](#) (int \*pSrcVec, int nSize)

### 6.243.1 Подробное описание

Поиск позиции последнего ненулевого элемента в векторе .

Аргументы

|         |                 |
|---------|-----------------|
| pSrcVec | Входной массив. |
| nIndex  | Размер массива. |

Возвращает

Позиция нулевого элемента или -1 в случае если нулевой элемент не найден

Restrictions:

Вычисления производятся на скалярном ядре, поэтому указатель на данные pSrcVec может ссылаться на нечетный адрес. Поиск производится с конца до первого ненулевого элемента.

### 6.243.2 Функции

#### 6.243.2.1 nmppsLastNonZeroIndx\_32s()

```
int nmppsLastNonZeroIndx_32s (
    int * pSrcVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 368   | 46.00      |
| im0     | 8     | 368   | 46.00      |
| im0     | 64    | 2328  | 36.37      |
| im0     | 64    | 2328  | 36.37      |
| im0     | 512   | 18008 | 35.17      |
| im0     | 512   | 18008 | 35.17      |



| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im1     | 8     | 368   | 46.00      |
| im1     | 8     | 368   | 46.00      |
| im1     | 64    | 2328  | 36.37      |
| im1     | 64    | 2328  | 36.37      |
| im1     | 512   | 18008 | 35.17      |
| im1     | 512   | 18008 | 35.17      |

The best configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 8     | 368   | 46.00      |

The worst configuration:

| pSrcVec | nSize | ticks | ticks/elem |
|---------|-------|-------|------------|
| im0     | 512   | 18008 | 35.17      |

## 6.244 nmppsMinEvery\_

Поэлементный минимум из двух векторов.

## Функции

- void nmppsMinEvery\_8s (nm8s7b \*pSrcVec1, nm8s7b \*pSrcVec2, nm8s7b \*pDstMinVec, int nSize)
- void nmppsMinEvery\_16s (nm16s15b \*pSrcVec1, nm16s15b \*pSrcVec2, nm16s15b \*pDstMinVec, int nSize)
- void nmppsMinEvery\_32s (nm32s31b \*pSrcVec1, nm32s31b \*pSrcVec2, nm32s31b \*pDstMinVec, int nSize)
- void nmppsMinEvery\_64s (nm64s63b \*pSrcVec1, nm64s63b \*pSrcVec2, nm64s63b \*pDstMinVec, int nSize)

## 6.244.1 Подробное описание

Поэлементный минимум из двух векторов.

$$pDstMinVec[i] = \min(pSrcVec1[i], pSrcVec2[i])$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| nSize    | Размер векторов в элементах. |

## Возвращаемые значения

|            |                                                     |
|------------|-----------------------------------------------------|
| pDstMinVec | Массив поэлементных минимумов для входных массивов. |
|------------|-----------------------------------------------------|

## Возвращает

void

## Restrictions:

Сравниваемые пары чисел двух массивов должны отличаться не более чем на  $2^{15} - 1$ .

Примеры допустимых диапазонов сравниваемых пар чисел:

should not be more than  $2^{15} - 1$ .

Here are some examples of admissible ranges for comparable pairs of numbers:

[0000h..7FFFh]=[ 0..+32767 ]

[FFFFh..7FFEh]=[ -1..+32766 ]

[C000h..3FFFh]=[-16384..+16383 ]

[8000h..0000h]=[-32768..0 ]

## 6.244.2 Функции

## 6.244.2.1 nmppsMinEvery\_16s()

```
void nmppsMinEvery_16s (
    nm16s15b * pSrcVec1,
    nm16s15b * pSrcVec2,
    nm16s15b * pDstMinVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 2116  | 1.03       |
| im0      | im0      | im2        | 2048  | 2116  | 1.03       |
| im0      | im0      | im3        | 2048  | 2116  | 1.03       |
| im0      | im0      | im3        | 2048  | 2116  | 1.03       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im3        | 2048  | 1706  | 0.83       |
| im0      | im1      | im3        | 2048  | 1706  | 0.83       |
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |
| im1      | im0      | im3        | 2048  | 1643  | 0.80       |
| im1      | im0      | im3        | 2048  | 1643  | 0.80       |
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |
| im1      | im1      | im3        | 2048  | 2187  | 1.06       |
| im1      | im1      | im3        | 2048  | 2187  | 1.06       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 128   | 16.00      |
| im0      | im1      | im2        | 8     | 128   | 16.00      |
| im0      | im1      | im2        | 128   | 219   | 1.71       |
| im0      | im1      | im2        | 128   | 219   | 1.71       |

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 1024  | 914   | 0.89       |
| im0      | im1      | im2        | 1024  | 914   | 0.89       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 128   | 16.00      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |

#### 6.244.2.2 nmppsMinEvery\_32s()

```
void nmppsMinEvery_32s (
    nm32s31b * pSrcVec1,
    nm32s31b * pSrcVec2,
    nm32s31b * pDstMinVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 4180  | 2.04       |
| im0      | im0      | im2        | 2048  | 4180  | 2.04       |
| im0      | im0      | im3        | 2048  | 4180  | 2.04       |
| im0      | im0      | im3        | 2048  | 4180  | 2.04       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im3        | 2048  | 3290  | 1.60       |
| im0      | im1      | im3        | 2048  | 3290  | 1.60       |
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |
| im1      | im0      | im3        | 2048  | 3227  | 1.57       |
| im1      | im0      | im3        | 2048  | 3227  | 1.57       |
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |
| im1      | im1      | im3        | 2048  | 4251  | 2.07       |
| im1      | im1      | im3        | 2048  | 4251  | 2.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 138   | 17.25      |
| im0      | im1      | im2        | 8     | 138   | 17.25      |
| im0      | im1      | im2        | 128   | 320   | 2.50       |
| im0      | im1      | im2        | 128   | 320   | 2.50       |
| im0      | im1      | im2        | 1024  | 1706  | 1.66       |
| im0      | im1      | im2        | 1024  | 1706  | 1.66       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 138   | 17.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |

### 6.244.2.3 nmppsMinEvery\_64s()

```
void nmppsMinEvery_64s (
    nm64s63b * pSrcVec1,
    nm64s63b * pSrcVec2,
    nm64s63b * pDstMinVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 8308  | 4.05       |
| im0      | im0      | im2        | 2048  | 8308  | 4.05       |
| im0      | im0      | im3        | 2048  | 8308  | 4.05       |
| im0      | im0      | im3        | 2048  | 8308  | 4.05       |

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im3        | 2048  | 6458  | 3.15       |
| im0      | im1      | im3        | 2048  | 6458  | 3.15       |
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |
| im1      | im0      | im3        | 2048  | 6395  | 3.12       |
| im1      | im0      | im3        | 2048  | 6395  | 3.12       |
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |
| im1      | im1      | im3        | 2048  | 8379  | 4.09       |
| im1      | im1      | im3        | 2048  | 8379  | 4.09       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |

Performance table 1

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 146   | 18.25      |
| im0      | im1      | im2        | 8     | 146   | 18.25      |
| im0      | im1      | im2        | 128   | 518   | 4.04       |
| im0      | im1      | im2        | 128   | 518   | 4.04       |
| im0      | im1      | im2        | 1024  | 3290  | 3.21       |
| im0      | im1      | im2        | 1024  | 3290  | 3.21       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 146   | 18.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |

## 6.244.2.4 nmppsMinEvery\_8s()

```
void nmppsMinEvery_8s (
    nm8s7b * pSrcVec1,
    nm8s7b * pSrcVec2,
    nm8s7b * pDstMinVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 1084  | 0.52       |
| im0      | im0      | im2        | 2048  | 1084  | 0.52       |
| im0      | im0      | im3        | 2048  | 1084  | 0.52       |
| im0      | im0      | im3        | 2048  | 1084  | 0.52       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |
| im0      | im1      | im3        | 2048  | 914   | 0.44       |
| im0      | im1      | im3        | 2048  | 914   | 0.44       |
| im1      | im0      | im2        | 2048  | 851   | 0.41       |
| im1      | im0      | im2        | 2048  | 851   | 0.41       |
| im1      | im0      | im3        | 2048  | 851   | 0.41       |
| im1      | im0      | im3        | 2048  | 851   | 0.41       |
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |
| im1      | im1      | im3        | 2048  | 1155  | 0.56       |
| im1      | im1      | im3        | 2048  | 1155  | 0.56       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 851   | 0.41       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 126   | 15.75      |
| im0      | im1      | im2        | 8     | 126   | 15.75      |
| im0      | im1      | im2        | 128   | 167   | 1.30       |
| im0      | im1      | im2        | 128   | 167   | 1.30       |
| im0      | im1      | im2        | 1024  | 518   | 0.50       |
| im0      | im1      | im2        | 1024  | 518   | 0.50       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMinVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 914   | 0.44       |



## 6.245 nmppsMaxEvery\_

Поэлементный максимум из двух векторов.

## Функции

- void `nmppsMaxEvery_8s` (`nm8s7b` \*pSrcVec1, `nm8s7b` \*pSrcVec2, `nm8s7b` \*pDstMaxVec, int nSize)
- void `nmppsMaxEvery_16s` (`nm16s15b` \*pSrcVec1, `nm16s15b` \*pSrcVec2, `nm16s15b` \*pDstMaxVec, int nSize)
- void `nmppsMaxEvery_32s` (`nm32s31b` \*pSrcVec1, `nm32s31b` \*pSrcVec2, `nm32s31b` \*pDstMaxVec, int nSize)
- void `nmppsMaxEvery_64s` (`nm64s63b` \*pSrcVec1, `nm64s63b` \*pSrcVec2, `nm64s63b` \*pDstMaxVec, int nSize)

## 6.245.1 Подробное описание

Поэлементный максимум из двух векторов.

$$pDstMaxVec[i] = \max(pSrcVec1[i], pSrcVec2[i])$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| nSize    | Размер векторов в элементах. |

## Возвращаемые значения

|            |                                                     |
|------------|-----------------------------------------------------|
| pDstMaxVec | Массив поэлементных минимумов для входных массивов. |
|------------|-----------------------------------------------------|

## Возвращает

void

## Restrictions:

Сравниваемые пары чисел двух массивов должны отличаться не более чем на  $2^{15} - 1$ .

Примеры допустимых диапазонов сравниваемых пар чисел:

should not be more than  $2^{15} - 1$ .

Here are some examples of admissible ranges for comparable pairs of numbers:

[0000h..7FFFh]=[ 0..+32767 ]

[FFFFh..7FFEh]=[ -1..+32766 ]

[C000h..3FFFh]=[ -16384..+16383 ]

[8000h..0000h]=[ -32768..0 ]

## 6.245.2 Функции

## 6.245.2.1 nmppsMaxEvery\_16s()

```
void nmppsMaxEvery_16s (
    nm16s15b * pSrcVec1,
    nm16s15b * pSrcVec2,
    nm16s15b * pDstMaxVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 2116  | 1.03       |
| im0      | im0      | im2        | 2048  | 2116  | 1.03       |
| im0      | im0      | im3        | 2048  | 2116  | 1.03       |
| im0      | im0      | im3        | 2048  | 2116  | 1.03       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im3        | 2048  | 1706  | 0.83       |
| im0      | im1      | im3        | 2048  | 1706  | 0.83       |
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |
| im1      | im0      | im3        | 2048  | 1643  | 0.80       |
| im1      | im0      | im3        | 2048  | 1643  | 0.80       |
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |
| im1      | im1      | im3        | 2048  | 2187  | 1.06       |
| im1      | im1      | im3        | 2048  | 2187  | 1.06       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 1643  | 0.80       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 2187  | 1.06       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 128   | 16.00      |

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 128   | 16.00      |
| im0      | im1      | im2        | 128   | 219   | 1.71       |
| im0      | im1      | im2        | 128   | 219   | 1.71       |
| im0      | im1      | im2        | 1024  | 914   | 0.89       |
| im0      | im1      | im2        | 1024  | 914   | 0.89       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 128   | 16.00      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 1706  | 0.83       |

#### 6.245.2.2 nmppsMaxEvery\_32s()

```
void nmppsMaxEvery_32s (
    nm32s31b * pSrcVec1,
    nm32s31b * pSrcVec2,
    nm32s31b * pDstMaxVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 4180  | 2.04       |
| im0      | im0      | im2        | 2048  | 4180  | 2.04       |
| im0      | im0      | im3        | 2048  | 4180  | 2.04       |
| im0      | im0      | im3        | 2048  | 4180  | 2.04       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im3        | 2048  | 3290  | 1.60       |
| im0      | im1      | im3        | 2048  | 3290  | 1.60       |
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |
| im1      | im0      | im3        | 2048  | 3227  | 1.57       |
| im1      | im0      | im3        | 2048  | 3227  | 1.57       |
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |
| im1      | im1      | im3        | 2048  | 4251  | 2.07       |
| im1      | im1      | im3        | 2048  | 4251  | 2.07       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 3227  | 1.57       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 4251  | 2.07       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 138   | 17.25      |
| im0      | im1      | im2        | 8     | 138   | 17.25      |
| im0      | im1      | im2        | 128   | 320   | 2.50       |
| im0      | im1      | im2        | 128   | 320   | 2.50       |
| im0      | im1      | im2        | 1024  | 1706  | 1.66       |
| im0      | im1      | im2        | 1024  | 1706  | 1.66       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 138   | 17.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 3290  | 1.60       |

### 6.245.2.3 nmppsMaxEvery\_64s()

```
void nmppsMaxEvery_64s (
    nm64s63b * pSrcVec1,
    nm64s63b * pSrcVec2,
    nm64s63b * pDstMaxVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 8308  | 4.05       |
| im0      | im0      | im2        | 2048  | 8308  | 4.05       |

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im3        | 2048  | 8308  | 4.05       |
| im0      | im0      | im3        | 2048  | 8308  | 4.05       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im3        | 2048  | 6458  | 3.15       |
| im0      | im1      | im3        | 2048  | 6458  | 3.15       |
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |
| im1      | im0      | im3        | 2048  | 6395  | 3.12       |
| im1      | im0      | im3        | 2048  | 6395  | 3.12       |
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |
| im1      | im1      | im3        | 2048  | 8379  | 4.09       |
| im1      | im1      | im3        | 2048  | 8379  | 4.09       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 6395  | 3.12       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 8379  | 4.09       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 146   | 18.25      |
| im0      | im1      | im2        | 8     | 146   | 18.25      |
| im0      | im1      | im2        | 128   | 518   | 4.04       |
| im0      | im1      | im2        | 128   | 518   | 4.04       |
| im0      | im1      | im2        | 1024  | 3290  | 3.21       |
| im0      | im1      | im2        | 1024  | 3290  | 3.21       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 146   | 18.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 6458  | 3.15       |

## 6.245.2.4 nmppsMaxEvery\_8s()

```
void nmppsMaxEvery_8s (
    nm8s7b * pSrcVec1,
    nm8s7b * pSrcVec2,
    nm8s7b * pDstMaxVec,
    int nSize )
```

Performace table 0

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im0      | im2        | 2048  | 1084  | 0.52       |
| im0      | im0      | im2        | 2048  | 1084  | 0.52       |
| im0      | im0      | im3        | 2048  | 1084  | 0.52       |
| im0      | im0      | im3        | 2048  | 1084  | 0.52       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |
| im0      | im1      | im3        | 2048  | 914   | 0.44       |
| im0      | im1      | im3        | 2048  | 914   | 0.44       |
| im1      | im0      | im2        | 2048  | 851   | 0.41       |
| im1      | im0      | im2        | 2048  | 851   | 0.41       |
| im1      | im0      | im3        | 2048  | 851   | 0.41       |
| im1      | im0      | im3        | 2048  | 851   | 0.41       |
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |
| im1      | im1      | im3        | 2048  | 1155  | 0.56       |
| im1      | im1      | im3        | 2048  | 1155  | 0.56       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im0      | im2        | 2048  | 851   | 0.41       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im1      | im1      | im2        | 2048  | 1155  | 0.56       |

Performace table 1

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 126   | 15.75      |
| im0      | im1      | im2        | 8     | 126   | 15.75      |
| im0      | im1      | im2        | 128   | 167   | 1.30       |
| im0      | im1      | im2        | 128   | 167   | 1.30       |
| im0      | im1      | im2        | 1024  | 518   | 0.50       |
| im0      | im1      | im2        | 1024  | 518   | 0.50       |
| im0      | im1      | im2        | 2048  | 914   | 0.44       |

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 914   | 0.44       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMaxVec | nSize | ticks | ticks/elem |
|----------|----------|------------|-------|-------|------------|
| im0      | im1      | im2        | 2048  | 914   | 0.44       |

## 6.246 nmppsMinCmpLtV\_

Поэлементный минимум из двух векторов.

Функции

- void `nmppsMinCmpLtV_16s` (`nm16s15b` \*pSrcVec1, `nm16s15b` \*pSrcVec2, `nm16s15b` \*pDstMin, `nm16s15b` \*pDstSignMask, int nSize)

## 6.246.1 Подробное описание

Поэлементный минимум из двух векторов.

$$pDstMin[i] = \min(pSrcVec1[i], pSrcVec2[i])$$

$$pDstSignMask[i] = \begin{cases} 11 \dots 1b, & pSrcVec1[i] < pSrcVec2[i] \\ 00 \dots 0b, & pSrcVec1[i] \leq pSrcVec2[i] \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| nSize    | Размер векторов в элементах. |

Возвращаемые значения

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| pDstMin      | Массив поэлементных минимумов для входных массивов.              |
| pDstSignMask | Массив знаков поэлементных разностей первого и второго векторов. |

Возвращает

void

Restrictions:

Сравниваемые пары чисел двух массивов должны отличаться не более чем на  $2^{15} - 1$ .

Примеры допустимых диапазонов сравниваемых пар чисел:

should not be more than  $2^{15} - 1$ .

Here are some examples of admissible ranges for comparable pairs of numbers:

[0000h..7FFFh]=[ 0..+32767 ]

[FFFFh..7FFEh]=[ -1..+32766 ]

[C000h..3FFFh]=[ -16384..+16383 ]

[8000h..0000h]=[ -32768..0 ]



## 6.246.2 Функции

## 6.246.2.1 nmppsMinCmpLtV\_16s()

```
void nmppsMinCmpLtV_16s (
    nm16s15b * pSrcVec1,
    nm16s15b * pSrcVec2,
    nm16s15b * pDstMin,
    nm16s15b * pDstSignMask,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im0      | im0      | im2     | im0          | 4096  | 5426  | 1.32       |
| im0      | im0      | im2     | im0          | 4096  | 5426  | 1.32       |
| im0      | im0      | im2     | im1          | 4096  | 5243  | 1.28       |
| im0      | im0      | im2     | im1          | 4096  | 5243  | 1.28       |
| im0      | im0      | im3     | im0          | 4096  | 5426  | 1.32       |
| im0      | im0      | im3     | im0          | 4096  | 5426  | 1.32       |
| im0      | im0      | im3     | im1          | 4096  | 5243  | 1.28       |
| im0      | im0      | im3     | im1          | 4096  | 5243  | 1.28       |
| im0      | im1      | im2     | im0          | 4096  | 4440  | 1.08       |
| im0      | im1      | im2     | im0          | 4096  | 4440  | 1.08       |
| im0      | im1      | im2     | im1          | 4096  | 4602  | 1.12       |
| im0      | im1      | im2     | im1          | 4096  | 4602  | 1.12       |
| im0      | im1      | im3     | im0          | 4096  | 4440  | 1.08       |
| im0      | im1      | im3     | im0          | 4096  | 4440  | 1.08       |
| im0      | im1      | im3     | im1          | 4096  | 4602  | 1.12       |
| im0      | im1      | im3     | im1          | 4096  | 4602  | 1.12       |
| im1      | im0      | im2     | im0          | 4096  | 4468  | 1.09       |
| im1      | im0      | im2     | im0          | 4096  | 4468  | 1.09       |
| im1      | im0      | im2     | im1          | 4096  | 4286  | 1.04       |
| im1      | im0      | im2     | im1          | 4096  | 4286  | 1.04       |
| im1      | im0      | im3     | im0          | 4096  | 4468  | 1.09       |
| im1      | im0      | im3     | im0          | 4096  | 4468  | 1.09       |
| im1      | im0      | im3     | im1          | 4096  | 4286  | 1.04       |
| im1      | im0      | im3     | im1          | 4096  | 4286  | 1.04       |
| im1      | im1      | im2     | im0          | 4096  | 5398  | 1.31       |
| im1      | im1      | im2     | im0          | 4096  | 5398  | 1.31       |
| im1      | im1      | im2     | im1          | 4096  | 5561  | 1.35       |
| im1      | im1      | im2     | im1          | 4096  | 5561  | 1.35       |
| im1      | im1      | im3     | im0          | 4096  | 5398  | 1.31       |
| im1      | im1      | im3     | im0          | 4096  | 5398  | 1.31       |
| im1      | im1      | im3     | im1          | 4096  | 5561  | 1.35       |
| im1      | im1      | im3     | im1          | 4096  | 5561  | 1.35       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im1      | im0      | im2     | im1          | 4096  | 4286  | 1.04       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im1      | im1      | im2     | im1          | 4096  | 5561  | 1.35       |

Perfomance table 1

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im0      | im1      | im2     | im3          | 8     | 170   | 21.25      |
| im0      | im1      | im2     | im3          | 8     | 170   | 21.25      |
| im0      | im1      | im2     | im3          | 128   | 257   | 2.00       |
| im0      | im1      | im2     | im3          | 128   | 257   | 2.00       |
| im0      | im1      | im2     | im3          | 1024  | 1200  | 1.17       |
| im0      | im1      | im2     | im3          | 1024  | 1200  | 1.17       |
| im0      | im1      | im2     | im3          | 4096  | 4440  | 1.08       |
| im0      | im1      | im2     | im3          | 4096  | 4440  | 1.08       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im0      | im1      | im2     | im3          | 8     | 170   | 21.25      |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstSignMask | nSize | ticks | ticks/elem |
|----------|----------|---------|--------------|-------|-------|------------|
| im0      | im1      | im2     | im3          | 4096  | 4440  | 1.08       |

## 6.247 nmppsCmpLt0

Сравнивает элементы массива на меньше нуля.

### Функции

- void `nmppsCmpLt0_8s` (const `nm8s` \*pSrcVec, `nm8s` \*pDstVec, int nSize)
- void `nmppsCmpLt0_16s` (const `nm16s` \*pSrcVec, `nm16s` \*pDstVec, int nSize)
- void `nmppsCmpLt0_32s` (const `nm32s` \*pSrcVec, `nm32s` \*pDstVec, int nSize)
- void `nmppsCmpLt0_64s` (const `nm64s` \*pSrcVec, `nm64s` \*pDstVec, int nSize)

### 6.247.1 Подробное описание

Сравнивает элементы массива на меньше нуля.

$$pDstVec(i) = \begin{cases} 11 \dots 1b, & \text{if } pSrcVec(i) < 0 \\ 00 \dots 0b, & \text{if } pSrcVec(i) \geq 0 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Если элемент входного вектора меньше 0, во все биты соответствующего элемента выходного вектора записывается 1.

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.247.2 Функции

## 6.247.2.1 nmppsCmpLt0\_16s()

```
void nmppsCmpLt0_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 1064  | 0.51       |
| im0     | im0     | 2048  | 1064  | 0.51       |
| im0     | im1     | 2048  | 653   | 0.31       |
| im0     | im1     | 2048  | 653   | 0.31       |
| im1     | im0     | 2048  | 651   | 0.31       |
| im1     | im0     | 2048  | 651   | 0.31       |
| im1     | im1     | 2048  | 1163  | 0.56       |
| im1     | im1     | 2048  | 1163  | 0.56       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 651   | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1163  | 0.56       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 109   | 13.62      |
| im0     | im1     | 8     | 109   | 13.62      |
| im0     | im1     | 128   | 144   | 1.12       |
| im0     | im1     | 128   | 144   | 1.12       |
| im0     | im1     | 1024  | 383   | 0.37       |
| im0     | im1     | 1024  | 383   | 0.37       |
| im0     | im1     | 2048  | 653   | 0.31       |
| im0     | im1     | 2048  | 653   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 109   | 13.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 653   | 0.31       |

## 6.247.2.2 nmppsCmpLt0\_32s()

```
void nmppsCmpLt0_32s (
    const nm32s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 2088  | 1.01       |
| im0     | im0     | 2048  | 2088  | 1.01       |
| im0     | im1     | 2048  | 1197  | 0.58       |
| im0     | im1     | 2048  | 1197  | 0.58       |
| im1     | im0     | 2048  | 1195  | 0.58       |
| im1     | im0     | 2048  | 1195  | 0.58       |
| im1     | im1     | 2048  | 2219  | 1.08       |
| im1     | im1     | 2048  | 2219  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 1195  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 2219  | 1.08       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 111   | 13.87      |
| im0     | im1     | 8     | 111   | 13.87      |
| im0     | im1     | 128   | 179   | 1.39       |
| im0     | im1     | 128   | 179   | 1.39       |
| im0     | im1     | 1024  | 653   | 0.63       |
| im0     | im1     | 1024  | 653   | 0.63       |
| im0     | im1     | 2048  | 1197  | 0.58       |
| im0     | im1     | 2048  | 1197  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 111   | 13.87      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 1197  | 0.58       |

### 6.247.2.3 nmppsCmpLt0\_64s()

```
void nmppsCmpLt0_64s (
    const nm64s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 4136  | 2.01       |
| im0     | im0     | 2048  | 4136  | 2.01       |
| im0     | im1     | 2048  | 2285  | 1.11       |
| im0     | im1     | 2048  | 2285  | 1.11       |
| im1     | im0     | 2048  | 2283  | 1.11       |
| im1     | im0     | 2048  | 2283  | 1.11       |
| im1     | im1     | 2048  | 4331  | 2.11       |
| im1     | im1     | 2048  | 4331  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 2283  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 4331  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 128   | 246   | 1.92       |
| im0     | im1     | 128   | 246   | 1.92       |
| im0     | im1     | 1024  | 1197  | 1.16       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 1024  | 1197  | 1.16       |
| im0     | im1     | 2048  | 2285  | 1.11       |
| im0     | im1     | 2048  | 2285  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 118   | 14.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 2285  | 1.11       |

#### 6.247.2.4 nmppsCmpLt0\_8s()

```
void nmppsCmpLt0_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 2048  | 552   | 0.26       |
| im0     | im0     | 2048  | 552   | 0.26       |
| im0     | im1     | 2048  | 383   | 0.18       |
| im0     | im1     | 2048  | 383   | 0.18       |
| im1     | im0     | 2048  | 379   | 0.18       |
| im1     | im0     | 2048  | 379   | 0.18       |
| im1     | im1     | 2048  | 635   | 0.31       |
| im1     | im1     | 2048  | 635   | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 379   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 635   | 0.31       |

Perfomance table 1

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 8     | 108   | 13.50      |
| im0     | im1     | 128   | 126   | 0.98       |
| im0     | im1     | 128   | 126   | 0.98       |
| im0     | im1     | 1024  | 246   | 0.24       |
| im0     | im1     | 1024  | 246   | 0.24       |
| im0     | im1     | 2048  | 383   | 0.18       |
| im0     | im1     | 2048  | 383   | 0.18       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 8     | 108   | 13.50      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 2048  | 383   | 0.18       |



## 6.248 nmppsCmpLteC

Функция сравнения элементов массива с константой (меньше или равно)

Функции

- void nmppsCmpLteC\_v2nm32f (const v2nm32f \*pSrcVec, const v2nm32f \*C, nm1 \*evenFlags, nm1 \*oddFlags, int step, int nSize)

### 6.248.1 Подробное описание

Функция сравнения элементов массива с константой (меньше или равно)

Аргументы

|    |         |                                                                                                      |
|----|---------|------------------------------------------------------------------------------------------------------|
| in | pSrcVec | входной массив (элемент массива представляет собой структура v2nm32f, состоящая из двух чисел float) |
| in | C       | константа (два числа float)                                                                          |

Возвращаемые значения

|       |                                                                              |
|-------|------------------------------------------------------------------------------|
| [out] | evenFlags маска полученная после сравнения первого поля из структуры v2nm32f |
| [out] | oddFlags маска полученная после сравнения второго поля из структуры v2nm32f  |

Аргументы

|    |       |                                                  |
|----|-------|--------------------------------------------------|
| in | step  | шаг, с которым будут браться элементы из pSrcVec |
| in | nSize | размер массива (в v2nm32f)                       |

В структуре v2nm32f два поля. Первое - v0. Второе - v1.

## 6.249 nmppsCmpLtC

Функция сравнения элементов массива с константой (меньше или равно)

Функции

- void nmppsCmpLtC\_v2nm32f (const v2nm32f \*pSrcVec, const v2nm32f \*C, nm1 \*evenFlags, nm1 \*oddFlags, int step, int nSize)

### 6.249.1 Подробное описание

Функция сравнения элементов массива с константой (меньше или равно)

Аргументы

|    |         |                                                                                                      |
|----|---------|------------------------------------------------------------------------------------------------------|
| in | pSrcVec | входной массив (элемент массива представляет собой структура v2nm32f, состоящая из двух чисел float) |
| in | C       | константа (два числа float)                                                                          |

Возвращаемые значения

|       |                                                                              |
|-------|------------------------------------------------------------------------------|
| [out] | evenFlags маска полученная после сравнения первого поля из структуры v2nm32f |
| [out] | oddFlags маска полученная после сравнения второго поля из структуры v2nm32f  |

Аргументы

|    |       |                                                  |
|----|-------|--------------------------------------------------|
| in | step  | шаг, с которым будут браться элементы из pSrcVec |
| in | nSize | размер массива (в v2nm32f)                       |

В структуре v2nm32f два поля. Первое - v0. Второе - v1.

## 6.250 nmppsCmpEq0

Сравнивает элементы массива на признак равенства нулю.

## Функции

- void nmppsCmpEq0\_8u7b (nm8u7b \*pSrcVec, nm1 \*pDstVec, int nSize, int nTrueFlag)
- void nmppsCmpEq0\_16u15b (nm16u15b \*pSrcVec, nm1 \*pDstVec, int nSize, int nTrueFlag)
- void nmppsCmpEq0\_32u31b (nm32u31b \*pSrcVec, nm1 \*pDstVec, int nSize, int nTrueFlag)
- void nmppsCmpEq0\_32s (const nm32s \*pSrcVec, nm32s \*pDstVec, int nSize)

## 6.250.1 Подробное описание

Сравнивает элементы массива на признак равенства нулю.

$$pDstVec(i) = \begin{cases} 1, & \text{if } pSrcVec(i) = 0 \\ 0, & \text{if } pSrcVec(i) \neq 0 \end{cases}, \text{if } nTrueFlag = 1$$

$$pDstVec(i) = \begin{cases} 0, & \text{if } pSrcVec(i) = 0 \\ 1, & \text{if } pSrcVec(i) \neq 0 \end{cases}, \text{if } nTrueFlag = 0$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|           |                                                                                  |
|-----------|----------------------------------------------------------------------------------|
| pSrcVec   | Входной вектор.                                                                  |
| nSize     | Размер векторов в элементах. Vector size in elements.                            |
| nTrueFlag | Младший бит nTrueFlag определяет значение выходного бита при выполнении условия. |

## Возвращаемые значения

|         |                                 |
|---------|---------------------------------|
| pDstVec | Результирующий бинарный вектор. |
|---------|---------------------------------|

## Возвращает

void

$$pDstVec(i) = \begin{cases} -1, & \text{if } pSrcVec(i) = 0 \\ 0, & \text{if } pSrcVec(i) \neq 0 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|         |                 |
|---------|-----------------|
| pSrcVec | Входной вектор. |
|---------|-----------------|

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Аргументы

|       |                              |
|-------|------------------------------|
| nSize | Размер векторов в элементах. |
|-------|------------------------------|

Возвращает

void

## 6.250.2 Функции

### 6.250.2.1 nmppsCmpEq0\_16u15b()

```
void nmppsCmpEq0_16u15b (  
    nm16u15b * pSrcVec,  
    nm1 * pDstVec,  
    int nSize,  
    int nTrueFlag )
```

Заметки

nSize =[1,2,3,4...]

### 6.250.2.2 nmppsCmpEq0\_32u31b()

```
void nmppsCmpEq0_32u31b (  
    nm32u31b * pSrcVec,  
    nm1 * pDstVec,  
    int nSize,  
    int nTrueFlag )
```

Заметки

nSize =[1,2,3,4...]

## 6.250.2.3 nmppsCmpEq0\_8u7b()

```
void nmppsCmpEq0_8u7b (  
    nm8u7b * pSrcVec,  
    nm1 * pDstVec,  
    int nSize,  
    int nTrueFlag )
```

## Заметки

nSize =[1,2,3,4...]

## 6.251 nmppsCmpMinMaxV\_

Поэлементное сравнение двух векторов.

## Функции

- void `nmppsCmpMinMaxV_8s` (`nm8s` \*pSrcVec1, `nm8s` \*pSrcVec2, `nm8s` \*pDstMin, `nm8s` \*pDstMax, int nSize)
- void `nmppsCmpMinMaxV_16s` (`nm16s` \*pSrcVec1, `nm16s` \*pSrcVec2, `nm16s` \*pDstMin, `nm16s` \*pDstMax, int nSize)
- void `nmppsCmpMinMaxV_32s` (`nm32s` \*pSrcVec1, `nm32s` \*pSrcVec2, `nm32s` \*pDstMin, `nm32s` \*pDstMax, int nSize)

## 6.251.1 Подробное описание

Поэлементное сравнение двух векторов.

$$pDstMin[i] = \min(pSrcVec1[i], pSrcVec2[i])$$

$$pDstMax[i] = \max(pSrcVec1[i], pSrcVec2[i])$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|          |                              |
|----------|------------------------------|
| pSrcVec1 | Первый входной вектор.       |
| pSrcVec2 | Второй Входной вектор.       |
| nSize    | Размер векторов в элементах. |

## Возвращаемые значения

|         |                                                      |
|---------|------------------------------------------------------|
| pDstMin | Массив поэлементных минимумов для входных массивов.  |
| pDstMax | Массив поэлементных максимумов для входных массивов. |

## Возвращает

void

## Restrictions:

Сравниваемые пары чисел двух массивов должны отличаться не более чем на  $2^{15} - 1$ .

Примеры допустимых диапазонов сравниваемых пар чисел:

should not be more than  $2^{15} - 1$ .

Here are some examples of admissible ranges for comparable pairs of numbers:

[0000h..7FFFh]=[ 0..+32767 ]

```
[FFFFh..7FFEh]=[ -1..+32766 ]
[C000h..3FFFh]=[-16384..+16383 ]
[8000h..0000h]=[-32768..0 ]
```

## 6.251.2 Функции

### 6.251.2.1 nmppsCmpMinMaxV\_16s()

```
void nmppsCmpMinMaxV_16s (
    nm16s * pSrcVec1,
    nm16s * pSrcVec2,
    nm16s * pDstMin,
    nm16s * pDstMax,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im0      | im0      | im2     | im2     | 4096  | 6268  | 1.53       |
| im0      | im0      | im2     | im2     | 4096  | 6268  | 1.53       |
| im0      | im0      | im2     | im3     | 4096  | 6268  | 1.53       |
| im0      | im0      | im2     | im3     | 4096  | 6268  | 1.53       |
| im0      | im0      | im3     | im2     | 4096  | 6268  | 1.53       |
| im0      | im0      | im3     | im2     | 4096  | 6268  | 1.53       |
| im0      | im0      | im3     | im3     | 4096  | 6268  | 1.53       |
| im0      | im0      | im3     | im3     | 4096  | 6268  | 1.53       |
| im0      | im1      | im2     | im2     | 4096  | 5405  | 1.31       |
| im0      | im1      | im2     | im2     | 4096  | 5405  | 1.31       |
| im0      | im1      | im2     | im3     | 4096  | 5405  | 1.31       |
| im0      | im1      | im2     | im3     | 4096  | 5405  | 1.31       |
| im0      | im1      | im3     | im2     | 4096  | 5405  | 1.31       |
| im0      | im1      | im3     | im2     | 4096  | 5405  | 1.31       |
| im0      | im1      | im3     | im3     | 4096  | 5405  | 1.31       |
| im0      | im1      | im3     | im3     | 4096  | 5405  | 1.31       |
| im1      | im0      | im2     | im2     | 4096  | 5400  | 1.31       |
| im1      | im0      | im2     | im2     | 4096  | 5400  | 1.31       |
| im1      | im0      | im2     | im3     | 4096  | 5400  | 1.31       |
| im1      | im0      | im2     | im3     | 4096  | 5400  | 1.31       |
| im1      | im0      | im3     | im2     | 4096  | 5400  | 1.31       |
| im1      | im0      | im3     | im2     | 4096  | 5400  | 1.31       |
| im1      | im0      | im3     | im3     | 4096  | 5400  | 1.31       |
| im1      | im0      | im3     | im3     | 4096  | 5400  | 1.31       |
| im1      | im1      | im2     | im2     | 4096  | 6334  | 1.54       |
| im1      | im1      | im2     | im2     | 4096  | 6334  | 1.54       |
| im1      | im1      | im2     | im3     | 4096  | 6334  | 1.54       |
| im1      | im1      | im2     | im3     | 4096  | 6334  | 1.54       |
| im1      | im1      | im3     | im2     | 4096  | 6334  | 1.54       |
| im1      | im1      | im3     | im2     | 4096  | 6334  | 1.54       |
| im1      | im1      | im3     | im3     | 4096  | 6334  | 1.54       |
| im1      | im1      | im3     | im3     | 4096  | 6334  | 1.54       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im0      | im2     | im2     | 4096  | 5400  | 1.31       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im1      | im2     | im2     | 4096  | 6334  | 1.54       |

#### 6.251.2.2 nmppsCmpMinMaxV\_32s()

```
void nmppsCmpMinMaxV_32s (
    nm32s * pSrcVec1,
    nm32s * pSrcVec2,
    nm32s * pDstMin,
    nm32s * pDstMax,
    int nSize )
```

Performance table 0

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im0      | im0      | im2     | im2     | 4096  | 12444 | 3.03       |
| im0      | im0      | im2     | im2     | 4096  | 12444 | 3.03       |
| im0      | im0      | im2     | im3     | 4096  | 12444 | 3.03       |
| im0      | im0      | im2     | im3     | 4096  | 12444 | 3.03       |
| im0      | im0      | im3     | im2     | 4096  | 12444 | 3.03       |
| im0      | im0      | im3     | im2     | 4096  | 12444 | 3.03       |
| im0      | im0      | im3     | im3     | 4096  | 12444 | 3.03       |
| im0      | im0      | im3     | im3     | 4096  | 12444 | 3.03       |
| im0      | im1      | im2     | im2     | 4096  | 10685 | 2.60       |
| im0      | im1      | im2     | im2     | 4096  | 10685 | 2.60       |
| im0      | im1      | im2     | im3     | 4096  | 10685 | 2.60       |
| im0      | im1      | im2     | im3     | 4096  | 10685 | 2.60       |
| im0      | im1      | im3     | im2     | 4096  | 10685 | 2.60       |
| im0      | im1      | im3     | im2     | 4096  | 10685 | 2.60       |
| im0      | im1      | im3     | im3     | 4096  | 10685 | 2.60       |
| im0      | im1      | im3     | im3     | 4096  | 10685 | 2.60       |
| im1      | im0      | im2     | im2     | 4096  | 10712 | 2.61       |
| im1      | im0      | im2     | im2     | 4096  | 10712 | 2.61       |
| im1      | im0      | im2     | im3     | 4096  | 10712 | 2.61       |
| im1      | im0      | im2     | im3     | 4096  | 10712 | 2.61       |
| im1      | im0      | im3     | im2     | 4096  | 10712 | 2.61       |
| im1      | im0      | im3     | im2     | 4096  | 10712 | 2.61       |
| im1      | im0      | im3     | im3     | 4096  | 10712 | 2.61       |
| im1      | im0      | im3     | im3     | 4096  | 10712 | 2.61       |
| im1      | im1      | im2     | im2     | 4096  | 12542 | 3.06       |
| im1      | im1      | im2     | im2     | 4096  | 12542 | 3.06       |
| im1      | im1      | im2     | im3     | 4096  | 12542 | 3.06       |



| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im1      | im2     | im3     | 4096  | 12542 | 3.06       |
| im1      | im1      | im3     | im2     | 4096  | 12542 | 3.06       |
| im1      | im1      | im3     | im2     | 4096  | 12542 | 3.06       |
| im1      | im1      | im3     | im3     | 4096  | 12542 | 3.06       |
| im1      | im1      | im3     | im3     | 4096  | 12542 | 3.06       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im0      | im1      | im2     | im2     | 4096  | 10685 | 2.60       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im1      | im2     | im2     | 4096  | 12542 | 3.06       |

### 6.251.2.3 nmppsCmpMinMaxV\_8s()

```
void nmppsCmpMinMaxV_8s (
    nm8s * pSrcVec1,
    nm8s * pSrcVec2,
    nm8s * pDstMin,
    nm8s * pDstMax,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im0      | im0      | im2     | im2     | 4096  | 3180  | 0.77       |
| im0      | im0      | im2     | im2     | 4096  | 3180  | 0.77       |
| im0      | im0      | im2     | im3     | 4096  | 3180  | 0.77       |
| im0      | im0      | im2     | im3     | 4096  | 3180  | 0.77       |
| im0      | im0      | im3     | im2     | 4096  | 3180  | 0.77       |
| im0      | im0      | im3     | im2     | 4096  | 3180  | 0.77       |
| im0      | im0      | im3     | im3     | 4096  | 3180  | 0.77       |
| im0      | im0      | im3     | im3     | 4096  | 3180  | 0.77       |
| im0      | im1      | im2     | im2     | 4096  | 2765  | 0.67       |
| im0      | im1      | im2     | im2     | 4096  | 2765  | 0.67       |
| im0      | im1      | im2     | im3     | 4096  | 2765  | 0.67       |
| im0      | im1      | im2     | im3     | 4096  | 2765  | 0.67       |
| im0      | im1      | im3     | im2     | 4096  | 2765  | 0.67       |
| im0      | im1      | im3     | im2     | 4096  | 2765  | 0.67       |
| im0      | im1      | im3     | im3     | 4096  | 2765  | 0.67       |
| im0      | im1      | im3     | im3     | 4096  | 2765  | 0.67       |
| im1      | im0      | im2     | im2     | 4096  | 2744  | 0.66       |
| im1      | im0      | im2     | im2     | 4096  | 2744  | 0.66       |

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im0      | im2     | im3     | 4096  | 2744  | 0.66       |
| im1      | im0      | im2     | im3     | 4096  | 2744  | 0.66       |
| im1      | im0      | im3     | im2     | 4096  | 2744  | 0.66       |
| im1      | im0      | im3     | im2     | 4096  | 2744  | 0.66       |
| im1      | im0      | im3     | im3     | 4096  | 2744  | 0.66       |
| im1      | im0      | im3     | im3     | 4096  | 2744  | 0.66       |
| im1      | im1      | im2     | im2     | 4096  | 3230  | 0.78       |
| im1      | im1      | im2     | im2     | 4096  | 3230  | 0.78       |
| im1      | im1      | im2     | im3     | 4096  | 3230  | 0.78       |
| im1      | im1      | im2     | im3     | 4096  | 3230  | 0.78       |
| im1      | im1      | im3     | im2     | 4096  | 3230  | 0.78       |
| im1      | im1      | im3     | im2     | 4096  | 3230  | 0.78       |
| im1      | im1      | im3     | im3     | 4096  | 3230  | 0.78       |
| im1      | im1      | im3     | im3     | 4096  | 3230  | 0.78       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im0      | im2     | im2     | 4096  | 2744  | 0.66       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstMin | pDstMax | nSize | ticks | ticks/elem |
|----------|----------|---------|---------|-------|-------|------------|
| im1      | im1      | im2     | im2     | 4096  | 3230  | 0.78       |

## 6.252 nmppsClipPowC\_

Функция насыщения.

Функции

- void nmppsClipPowC\_8s (nm8s \*pSrcVec, int nClipFactor, nm8s \*pDstVec, int nSize)
- void nmppsClipPowC\_16s (nm16s \*pSrcVec, int nClipFactor, nm16s \*pDstVec, int nSize)
- void nmppsClipPowC\_32s (nm32s \*pSrcVec, int nClipFactor, nm32s \*pDstVec, int nSize)
- void nmppsClipPowC\_64s (nm64s \*pSrcVec, int nClipFactor, nm64s \*pDstVec, int nSize)

## 6.252.1 Подробное описание

Функция насыщения.

$$pDstVec[i] = \begin{cases} -2^{nClipFactor}, & \text{if } pSrcVec[i] < -2^{nClipFactor} \\ pSrcVec[i], & \text{if } -2^{nClipFactor} \leq pSrcVec[i] \leq 2^{nClipFactor} - 1 \\ 2^{nClipFactor} - 1, & \text{if } pSrcVec[i] > 2^{nClipFactor} - 1 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|             |                                                                                      |
|-------------|--------------------------------------------------------------------------------------|
| pSrcVec     | Входной вектор.                                                                      |
| nClipFactor | Показатель степени, определяющий верхний и нижний пороги насыщения.<br>nClipFactor>0 |
| nSize       | Размер вектора в элементах.                                                          |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.252.2 Функции

## 6.252.2.1 nmppsClipPowC\_16s()

```
void nmppsClipPowC_16s (
    nm16s * pSrcVec,
    int nClipFactor,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im0     | 2           | 2048  | 1072  | 0.52       |
| im0     | im0     | 2           | 2048  | 1072  | 0.52       |
| im0     | im1     | 2           | 2048  | 661   | 0.32       |
| im0     | im1     | 2           | 2048  | 661   | 0.32       |
| im1     | im0     | 2           | 2048  | 659   | 0.32       |
| im1     | im0     | 2           | 2048  | 659   | 0.32       |
| im1     | im1     | 2           | 2048  | 1171  | 0.57       |
| im1     | im1     | 2           | 2048  | 1171  | 0.57       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im0     | 2           | 2048  | 659   | 0.32       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im1     | 2           | 2048  | 1171  | 0.57       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 661   | 0.32       |
| im0     | im1     | 2           | 2048  | 661   | 0.32       |
| im0     | im1     | 3           | 2048  | 661   | 0.32       |
| im0     | im1     | 3           | 2048  | 661   | 0.32       |
| im0     | im1     | 5           | 2048  | 661   | 0.32       |
| im0     | im1     | 5           | 2048  | 661   | 0.32       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 661   | 0.32       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 661   | 0.32       |

Perfomance table 2

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 117   | 14.62      |
| im0     | im1     | 2           | 8     | 117   | 14.62      |
| im0     | im1     | 2           | 128   | 152   | 1.18       |
| im0     | im1     | 2           | 128   | 152   | 1.18       |
| im0     | im1     | 2           | 1024  | 391   | 0.38       |
| im0     | im1     | 2           | 1024  | 391   | 0.38       |
| im0     | im1     | 2           | 2048  | 661   | 0.32       |
| im0     | im1     | 2           | 2048  | 661   | 0.32       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 117   | 14.62      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 661   | 0.32       |

#### 6.252.2.2 nmppsClipPowC\_32s()

```
void nmppsClipPowC_32s (
    nm32s * pSrcVec,
    int nClipFactor,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im0     | 2           | 2048  | 2096  | 1.02       |
| im0     | im0     | 2           | 2048  | 2096  | 1.02       |
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |
| im1     | im0     | 2           | 2048  | 1203  | 0.58       |
| im1     | im0     | 2           | 2048  | 1203  | 0.58       |
| im1     | im1     | 2           | 2048  | 2227  | 1.08       |
| im1     | im1     | 2           | 2048  | 2227  | 1.08       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im0     | 2           | 2048  | 1203  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im1     | 2           | 2048  | 2227  | 1.08       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |
| im0     | im1     | 3           | 2048  | 1205  | 0.58       |
| im0     | im1     | 3           | 2048  | 1205  | 0.58       |
| im0     | im1     | 5           | 2048  | 1205  | 0.58       |
| im0     | im1     | 5           | 2048  | 1205  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |

Perfomance table 2

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 119   | 14.87      |
| im0     | im1     | 2           | 8     | 119   | 14.87      |
| im0     | im1     | 2           | 128   | 187   | 1.46       |
| im0     | im1     | 2           | 128   | 187   | 1.46       |
| im0     | im1     | 2           | 1024  | 661   | 0.64       |
| im0     | im1     | 2           | 1024  | 661   | 0.64       |
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 119   | 14.87      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 1205  | 0.58       |

## 6.252.2.3 nmppsClipPowC\_64s()

```
void nmppsClipPowC_64s (
    nm64s * pSrcVec,
    int nClipFactor,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im0     | 2           | 2048  | 4144  | 2.02       |
| im0     | im0     | 2           | 2048  | 4144  | 2.02       |
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |
| im1     | im0     | 2           | 2048  | 2291  | 1.11       |
| im1     | im0     | 2           | 2048  | 2291  | 1.11       |
| im1     | im1     | 2           | 2048  | 4339  | 2.11       |
| im1     | im1     | 2           | 2048  | 4339  | 2.11       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im0     | 2           | 2048  | 2291  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im1     | 2           | 2048  | 4339  | 2.11       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |
| im0     | im1     | 3           | 2048  | 2293  | 1.11       |
| im0     | im1     | 3           | 2048  | 2293  | 1.11       |
| im0     | im1     | 5           | 2048  | 2293  | 1.11       |
| im0     | im1     | 5           | 2048  | 2293  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |

Perfomance table 2

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 126   | 15.75      |
| im0     | im1     | 2           | 8     | 126   | 15.75      |
| im0     | im1     | 2           | 128   | 254   | 1.98       |
| im0     | im1     | 2           | 128   | 254   | 1.98       |
| im0     | im1     | 2           | 1024  | 1205  | 1.17       |
| im0     | im1     | 2           | 1024  | 1205  | 1.17       |
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 126   | 15.75      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 2293  | 1.11       |

#### 6.252.2.4 nmppsClipPowC\_8s()

```
void nmppsClipPowC_8s (
    nm8s * pSrcVec,
    int nClipFactor,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im0     | 2           | 2048  | 560   | 0.27       |
| im0     | im0     | 2           | 2048  | 560   | 0.27       |
| im0     | im1     | 2           | 2048  | 391   | 0.19       |
| im0     | im1     | 2           | 2048  | 391   | 0.19       |
| im1     | im0     | 2           | 2048  | 387   | 0.18       |
| im1     | im0     | 2           | 2048  | 387   | 0.18       |
| im1     | im1     | 2           | 2048  | 643   | 0.31       |
| im1     | im1     | 2           | 2048  | 643   | 0.31       |

The best configuration:



| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im0     | 2           | 2048  | 387   | 0.18       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im1     | im1     | 2           | 2048  | 643   | 0.31       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 391   | 0.19       |
| im0     | im1     | 2           | 2048  | 391   | 0.19       |
| im0     | im1     | 3           | 2048  | 391   | 0.19       |
| im0     | im1     | 3           | 2048  | 391   | 0.19       |
| im0     | im1     | 5           | 2048  | 391   | 0.19       |
| im0     | im1     | 5           | 2048  | 391   | 0.19       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 391   | 0.19       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 391   | 0.19       |

Perfomance table 2

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 116   | 14.50      |
| im0     | im1     | 2           | 8     | 116   | 14.50      |
| im0     | im1     | 2           | 128   | 134   | 1.04       |
| im0     | im1     | 2           | 128   | 134   | 1.04       |
| im0     | im1     | 2           | 1024  | 254   | 0.24       |
| im0     | im1     | 2           | 1024  | 254   | 0.24       |
| im0     | im1     | 2           | 2048  | 391   | 0.19       |
| im0     | im1     | 2           | 2048  | 391   | 0.19       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 8     | 116   | 14.50      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nSize | ticks | ticks/elem |
|---------|---------|-------------|-------|-------|------------|
| im0     | im1     | 2           | 2048  | 391   | 0.19       |

## 6.253 nmppsClipCC\_

Функция насыщения с произвольными порогами.

Функции

- void nmppsClipCC\_32s (nm32s30b \*pSrcVec, int30b nNegThresh, int30b nPosThresh, nm32s30b \*pDstVec, int nSize)

## 6.253.1 Подробное описание

Функция насыщения с произвольными порогами.

$$pDstVec[i] = \begin{cases} nPosThresh, & \text{if } pSrcVec[i] > nPosThresh \\ pSrcVec[i], & \text{if } nNegThresh \leq pSrcVec[i] \leq nPosThresh \\ nNegThresh, & \text{if } pSrcVec[i] < nNegThresh \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|            |                             |
|------------|-----------------------------|
| pSrcVec    | Входной вектор.             |
| nNegThresh | Нижний порог насыщения.     |
| nPosThresh | Верхний порог насыщения.    |
| nSize      | Размер вектора в элементах. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

Restrictions:

- Диапазон изменения для \form#67.
- Диапазон изменения для \form#68. \n

## 6.254 nmppsClipRShiftConvert\_AddC\_

Сокращение разрядности данных с предварительной их обработкой.

### Функции

- void `nmppsClipRShiftConvertAddC_16s8s` (`nm16s` \*pSrcVec, int nClipFactor, int nShift, `int8b` n←AddValue, `nm8s` \*pDstVec, int nSize)
- void `nmppsClipRShiftConvertAddC_32s8s` (`nm32s` \*pSrcVec, int nClipFactor, int nShift, `int8b` n←AddValue, `nm8s` \*pDstVec, int nSize)

### 6.254.1 Подробное описание

Сокращение разрядности данных с предварительной их обработкой.

$$pDstVec[i] = Convert(Clip(pSrcVec[i], nClipFactor) >> nShift) + nAddValue$$

$$i = \overline{0 \dots nSize - 1}$$

Сокращение разрядности данных выполняется после предварительной обработки и осуществляется путем отбрасывания старших битов.

### Аргументы

|             |                                                                                             |
|-------------|---------------------------------------------------------------------------------------------|
| pSrcVec     | Входной вектор.                                                                             |
| nClipFactor | Показатель степени, определяющий верхний и нижний пороги насыщения.<br>=[1,2,3...15]        |
| nShift      | Параметр определяет на сколько позиций =[2,4,6...14] нужно сдвинуть биты элементов вектора; |
| nAddValue   | Добавляемая константа.                                                                      |
| nSize       | Размер векторов в элементах.                                                                |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.254.2 Функции

## 6.254.2.1 nmppsClipRShiftConvertAddC\_16s8s()

```
void nmppsClipRShiftConvertAddC_16s8s (
    nm16s * pSrcVec,
    int nClipFactor,
    int nShift,
    int8b nAddValue,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im0     | 4           | 2      | 127       | 4096  | 1696  | 0.41       |
| im0     | im0     | 4           | 2      | 127       | 4096  | 1696  | 0.41       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im1     | im0     | 4           | 2      | 127       | 4096  | 1552  | 0.37       |
| im1     | im0     | 4           | 2      | 127       | 4096  | 1552  | 0.37       |
| im1     | im1     | 4           | 2      | 127       | 4096  | 2020  | 0.49       |
| im1     | im1     | 4           | 2      | 127       | 4096  | 2020  | 0.49       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im1     | im1     | 4           | 2      | 127       | 4096  | 2020  | 0.49       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 1           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 15          | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 15          | 2      | 127       | 4096  | 1282  | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 1282  | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 1282  | 0.31       |

Performance table 2

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 4      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 4      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 6      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 6      | 127       | 4096  | 1282  | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

Performance table 3

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 0         | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 64        | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 64        | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 1282  | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 1282  | 0.31       |

Performance table 4

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 8     | 183   | 22.87      |
| im0     | im1     | 4           | 2      | 127       | 8     | 183   | 22.87      |
| im0     | im1     | 4           | 2      | 127       | 128   | 226   | 1.76       |
| im0     | im1     | 4           | 2      | 127       | 128   | 226   | 1.76       |
| im0     | im1     | 4           | 2      | 127       | 1024  | 478   | 0.46       |
| im0     | im1     | 4           | 2      | 127       | 1024  | 478   | 0.46       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 8     | 183   | 22.87      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 1282  | 0.31       |

#### 6.254.2.2 nmppsClipRShiftConvertAddC\_32s8s()

```
void nmppsClipRShiftConvertAddC_32s8s (
    nm32s * pSrcVec,
    int nClipFactor,
    int nShift,
    int8b nAddValue,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im0     | 4           | 2      | 127       | 4096  | 2724  | 0.66       |
| im0     | im0     | 4           | 2      | 127       | 4096  | 2724  | 0.66       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im1     | im0     | 4           | 2      | 127       | 4096  | 2547  | 0.62       |
| im1     | im0     | 4           | 2      | 127       | 4096  | 2547  | 0.62       |
| im1     | im1     | 4           | 2      | 127       | 4096  | 3056  | 0.74       |
| im1     | im1     | 4           | 2      | 127       | 4096  | 3056  | 0.74       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im1     | im1     | 4           | 2      | 127       | 4096  | 3056  | 0.74       |

Perfomance table 1

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 1           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 15          | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 15          | 2      | 127       | 4096  | 2287  | 0.55       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 2287  | 0.55       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 1           | 2      | 127       | 4096  | 2287  | 0.55       |

Perfomance table 2

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 4      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 4      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 6      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 6      | 127       | 4096  | 2287  | 0.55       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

Perfomance table 3



| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 0         | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 64        | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 64        | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 2287  | 0.55       |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 0         | 4096  | 2287  | 0.55       |

Perfomance table 4

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 8     | 180   | 22.50      |
| im0     | im1     | 4           | 2      | 127       | 8     | 180   | 22.50      |
| im0     | im1     | 4           | 2      | 127       | 128   | 246   | 1.92       |
| im0     | im1     | 4           | 2      | 127       | 128   | 246   | 1.92       |
| im0     | im1     | 4           | 2      | 127       | 1024  | 715   | 0.69       |
| im0     | im1     | 4           | 2      | 127       | 1024  | 715   | 0.69       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

The best configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 8     | 180   | 22.50      |

The worst configuration:

| pSrcVec | pDstVec | nClipFactor | nShift | nAddValue | nSize | ticks | ticks/elem |
|---------|---------|-------------|--------|-----------|-------|-------|------------|
| im0     | im1     | 4           | 2      | 127       | 4096  | 2287  | 0.55       |

## 6.255 nmppsClipConvert\_AddC\_

Сокращение разрядности данных с предварительной их обработкой.

Определения типов

- typedef [nm64u](#) NmppsWeightState

Функции

- void nmppsClipConvertAddCInitAlloc\_16s8s (NmppsWeightState \*\*ppState)
- void nmppsClipConvertAddC\_16s8s ([nm16s](#) \*pSrcVec, int nClipFactor, [int8b](#) nAddValue, [nm8s](#) \*pDstVec, int nSize, NmppsWeightState \*pState)
- void nmppsClipConvertAddCFree (NmppsWeightState \*pState)

### 6.255.1 Подробное описание

Сокращение разрядности данных с предварительной их обработкой.

$$pDstVec[i] = Convert(Clip(pSrcVec[i], nClipFactor)) + nAddValue$$

$$i = \overline{0 \dots nSize - 1}$$

Сокращение разрядности данных выполняется после предварительной обработки и осуществляется путем отбрасывания старших битов.

\param pSrcVec

Входной вектор.

Аргументы

|             |                                                                                                           |
|-------------|-----------------------------------------------------------------------------------------------------------|
| nClipFactor | Показатель степени, определяющий верхний и нижний пороги насыщения.<br>=[1,2,3...15]                      |
| nAddValue   | Добавляемая константа.                                                                                    |
| nSize       | Размер векторов в элементах. указатель на матрицу коэффициентов для векторного умножителя. Для ускорения. |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.256 nmppsCmpEqC

Сравнивает элементы массива на признак равенства константе.

### Функции

- void `nmppsCmpEqC_16u15b` (`nm16u15b` \*pSrcVec, `uint15b` nCmpVal, `nm16s` \*pDstVec, int nSize, `int16b` nTrueFlag)
- void `nmppsCmpEqC_8u7b` (`nm8u7b` \*pSrcVec, `uint7b` nCmpVal, `nm8s` \*pDstVec, int nSize, `int8b` nTrueFlag)
- void `nmppsCmpEqC_4u3b` (`nm4u3b` \*pSrcVec, `uint3b` nCmpVal, `nm4s` \*pDstVec, int nSize, `int4b` nTrueFlag)

### 6.256.1 Подробное описание

Сравнивает элементы массива на признак равенства константе.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) = nCmpVal \\ 0, & \text{if } pSrcVec(i) \neq nCmpVal \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|           |                                                         |
|-----------|---------------------------------------------------------|
| pSrcVec   | Входной вектор.                                         |
| nCmpVal   | Значение константы для сравнения                        |
| nSize     | Размер векторов в элементах.                            |
| nTrueFlag | Значение флага, устанавливаемого при выполнении условия |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.256.2 Функции

## 6.256.2.1 nmppsCmpEqC\_16u15b()

```
void nmppsCmpEqC_16u15b (
    nm16u15b * pSrcVec,
    uint15b nCmpVal,
    nm16s * pDstVec,
    int nSize,
    int16b nTrueFlag )
```

Performance table 0

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im0     | im0     | 0       | 1         | 4096  | 3161  | 0.77       |
| im0     | im0     | 0       | 1         | 4096  | 3161  | 0.77       |
| im0     | im1     | 0       | 1         | 4096  | 2324  | 0.56       |
| im0     | im1     | 0       | 1         | 4096  | 2324  | 0.56       |
| im1     | im0     | 0       | 1         | 4096  | 2293  | 0.55       |
| im1     | im0     | 0       | 1         | 4096  | 2293  | 0.55       |
| im1     | im1     | 0       | 1         | 4096  | 3286  | 0.80       |
| im1     | im1     | 0       | 1         | 4096  | 3286  | 0.80       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im0     | 0       | 1         | 4096  | 2293  | 0.55       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im1     | 0       | 1         | 4096  | 3286  | 0.80       |

## 6.256.2.2 nmppsCmpEqC\_4u3b()

```
void nmppsCmpEqC_4u3b (
    nm4u3b * pSrcVec,
    uint3b nCmpVal,
    nm4s * pDstVec,
    int nSize,
    int4b nTrueFlag )
```

Performance table 0

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im0     | im0     | 0       | 1         | 4096  | 938   | 0.22       |
| im0     | im0     | 0       | 1         | 4096  | 938   | 0.22       |
| im0     | im1     | 0       | 1         | 4096  | 821   | 0.20       |
| im0     | im1     | 0       | 1         | 4096  | 821   | 0.20       |
| im1     | im0     | 0       | 1         | 4096  | 790   | 0.19       |
| im1     | im0     | 0       | 1         | 4096  | 790   | 0.19       |
| im1     | im1     | 0       | 1         | 4096  | 1063  | 0.25       |
| im1     | im1     | 0       | 1         | 4096  | 1063  | 0.25       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im0     | 0       | 1         | 4096  | 790   | 0.19       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im1     | 0       | 1         | 4096  | 1063  | 0.25       |

### 6.256.2.3 nmppsCmpEqC\_8u7b()

```
void nmppsCmpEqC_8u7b (
    nm8u7b * pSrcVec,
    uint7b nCmpVal,
    nm8s * pDstVec,
    int nSize,
    int8b nTrueFlag )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im0     | im0     | 0       | 1         | 4096  | 1657  | 0.40       |
| im0     | im0     | 0       | 1         | 4096  | 1657  | 0.40       |
| im0     | im1     | 0       | 1         | 4096  | 1300  | 0.31       |
| im0     | im1     | 0       | 1         | 4096  | 1300  | 0.31       |
| im1     | im0     | 0       | 1         | 4096  | 1269  | 0.30       |
| im1     | im0     | 0       | 1         | 4096  | 1269  | 0.30       |
| im1     | im1     | 0       | 1         | 4096  | 1782  | 0.43       |
| im1     | im1     | 0       | 1         | 4096  | 1782  | 0.43       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im0     | 0       | 1         | 4096  | 1269  | 0.30       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nTrueFlag | nSize | ticks | ticks/elem |
|---------|---------|---------|-----------|-------|-------|------------|
| im1     | im1     | 0       | 1         | 4096  | 1782  | 0.43       |

## 6.257 nmppsCmpNe0

Сравнивает элементы массива на признак неравенства нулю.

### Функции

- void `nmppsCmpNe0_8s` (const `nm8s` \*pSrcVec, `nm8s` \*pDstVec, int nSize)
- void `nmppsCmpNe0_16s` (const `nm16s` \*pSrcVec, `nm16s` \*pDstVec, int nSize)
- void `nmppsCmpNe0_32s` (const `nm32s` \*pSrcVec, `nm32s` \*pDstVec, int nSize)
- void `nmppsCmpNe0_64s` (const `nm64s` \*pSrcVec, `nm64s` \*pDstVec, int nSize)

### 6.257.1 Подробное описание

Сравнивает элементы массива на признак неравенства нулю.

$$pDstVec(i) = \begin{cases} -1, & \text{if } pSrcVec(i) \neq 0 \\ 0, & \text{if } pSrcVec(i) = 0 \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

## 6.257.2 Функции

### 6.257.2.1 nmppsCmpNe0\_16s()

```
void nmppsCmpNe0_16s (
    const nm16s * pSrcVec,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 128   | 121   | 0.94       |
| im0     | im0     | 128   | 121   | 0.94       |
| im0     | im0     | 512   | 398   | 0.77       |
| im0     | im0     | 512   | 398   | 0.77       |
| im0     | im0     | 1024  | 778   | 0.75       |
| im0     | im0     | 1024  | 778   | 0.75       |
| im0     | im0     | 2048  | 1538  | 0.75       |
| im0     | im0     | 2048  | 1538  | 0.75       |
| im0     | im1     | 8     | 120   | 15.00      |
| im0     | im1     | 8     | 120   | 15.00      |
| im0     | im1     | 128   | 208   | 1.62       |
| im0     | im1     | 128   | 208   | 1.62       |
| im0     | im1     | 512   | 401   | 0.78       |
| im0     | im1     | 512   | 401   | 0.78       |
| im0     | im1     | 1024  | 661   | 0.64       |
| im0     | im1     | 1024  | 661   | 0.64       |
| im0     | im1     | 2048  | 1181  | 0.57       |
| im0     | im1     | 2048  | 1181  | 0.57       |
| im1     | im0     | 8     | 120   | 15.00      |
| im1     | im0     | 8     | 120   | 15.00      |
| im1     | im0     | 128   | 147   | 1.14       |
| im1     | im0     | 128   | 147   | 1.14       |
| im1     | im0     | 512   | 345   | 0.67       |
| im1     | im0     | 512   | 345   | 0.67       |
| im1     | im0     | 1024  | 605   | 0.59       |
| im1     | im0     | 1024  | 605   | 0.59       |
| im1     | im0     | 2048  | 1125  | 0.54       |
| im1     | im0     | 2048  | 1125  | 0.54       |
| im1     | im1     | 8     | 121   | 15.12      |
| im1     | im1     | 8     | 121   | 15.12      |
| im1     | im1     | 128   | 210   | 1.64       |
| im1     | im1     | 128   | 210   | 1.64       |
| im1     | im1     | 512   | 498   | 0.97       |
| im1     | im1     | 512   | 498   | 0.97       |
| im1     | im1     | 1024  | 878   | 0.85       |
| im1     | im1     | 1024  | 878   | 0.85       |
| im1     | im1     | 2048  | 1638  | 0.79       |
| im1     | im1     | 2048  | 1638  | 0.79       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 1638  | 0.79       |



## 6.257.2.2 nmppsCmpNe0\_32s()

```
void nmppsCmpNe0_32s (
    const nm32s * pSrcVec,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 128   | 208   | 1.62       |
| im0     | im0     | 128   | 208   | 1.62       |
| im0     | im0     | 512   | 778   | 1.51       |
| im0     | im0     | 512   | 778   | 1.51       |
| im0     | im0     | 1024  | 1538  | 1.50       |
| im0     | im0     | 1024  | 1538  | 1.50       |
| im0     | im0     | 2048  | 3058  | 1.49       |
| im0     | im0     | 2048  | 3058  | 1.49       |
| im0     | im1     | 8     | 124   | 15.50      |
| im0     | im1     | 8     | 124   | 15.50      |
| im0     | im1     | 128   | 271   | 2.11       |
| im0     | im1     | 128   | 271   | 2.11       |
| im0     | im1     | 512   | 661   | 1.29       |
| im0     | im1     | 512   | 661   | 1.29       |
| im0     | im1     | 1024  | 1181  | 1.15       |
| im0     | im1     | 1024  | 1181  | 1.15       |
| im0     | im1     | 2048  | 2221  | 1.08       |
| im0     | im1     | 2048  | 2221  | 1.08       |
| im1     | im0     | 8     | 121   | 15.12      |
| im1     | im0     | 8     | 121   | 15.12      |
| im1     | im0     | 128   | 215   | 1.67       |
| im1     | im0     | 128   | 215   | 1.67       |
| im1     | im0     | 512   | 605   | 1.18       |
| im1     | im0     | 512   | 605   | 1.18       |
| im1     | im0     | 1024  | 1125  | 1.09       |
| im1     | im0     | 1024  | 1125  | 1.09       |
| im1     | im0     | 2048  | 2165  | 1.05       |
| im1     | im0     | 2048  | 2165  | 1.05       |
| im1     | im1     | 8     | 127   | 15.87      |
| im1     | im1     | 8     | 127   | 15.87      |
| im1     | im1     | 128   | 308   | 2.40       |
| im1     | im1     | 128   | 308   | 2.40       |
| im1     | im1     | 512   | 878   | 1.71       |
| im1     | im1     | 512   | 878   | 1.71       |
| im1     | im1     | 1024  | 1638  | 1.59       |
| im1     | im1     | 1024  | 1638  | 1.59       |
| im1     | im1     | 2048  | 3158  | 1.54       |
| im1     | im1     | 2048  | 3158  | 1.54       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 3158  | 1.54       |

### 6.257.2.3 nmppsCmpNe0\_64s()

```
void nmppsCmpNe0_64s (
    const nm64s * pSrcVec,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 125   | 15.62      |
| im0     | im0     | 8     | 125   | 15.62      |
| im0     | im0     | 128   | 405   | 3.16       |
| im0     | im0     | 128   | 405   | 3.16       |
| im0     | im0     | 512   | 1545  | 3.01       |
| im0     | im0     | 512   | 1545  | 3.01       |
| im0     | im0     | 1024  | 3065  | 2.99       |
| im0     | im0     | 1024  | 3065  | 2.99       |
| im0     | im0     | 2048  | 6105  | 2.98       |
| im0     | im0     | 2048  | 6105  | 2.98       |
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 8     | 144   | 18.00      |
| im0     | im1     | 128   | 408   | 3.18       |
| im0     | im1     | 128   | 408   | 3.18       |
| im0     | im1     | 512   | 1188  | 2.32       |
| im0     | im1     | 512   | 1188  | 2.32       |
| im0     | im1     | 1024  | 2228  | 2.17       |
| im0     | im1     | 1024  | 2228  | 2.17       |
| im0     | im1     | 2048  | 4308  | 2.10       |
| im0     | im1     | 2048  | 4308  | 2.10       |
| im1     | im0     | 8     | 132   | 16.50      |
| im1     | im0     | 8     | 132   | 16.50      |
| im1     | im0     | 128   | 352   | 2.75       |
| im1     | im0     | 128   | 352   | 2.75       |
| im1     | im0     | 512   | 1132  | 2.21       |
| im1     | im0     | 512   | 1132  | 2.21       |
| im1     | im0     | 1024  | 2172  | 2.12       |
| im1     | im0     | 1024  | 2172  | 2.12       |
| im1     | im0     | 2048  | 4252  | 2.07       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im0     | 2048  | 4252  | 2.07       |
| im1     | im1     | 8     | 147   | 18.37      |
| im1     | im1     | 8     | 147   | 18.37      |
| im1     | im1     | 128   | 505   | 3.94       |
| im1     | im1     | 128   | 505   | 3.94       |
| im1     | im1     | 512   | 1645  | 3.21       |
| im1     | im1     | 512   | 1645  | 3.21       |
| im1     | im1     | 1024  | 3165  | 3.09       |
| im1     | im1     | 1024  | 3165  | 3.09       |
| im1     | im1     | 2048  | 6205  | 3.02       |
| im1     | im1     | 2048  | 6205  | 3.02       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 125   | 15.62      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 6205  | 3.02       |

#### 6.257.2.4 nmppsCmpNe0\_8s()

```
void nmppsCmpNe0_8s (
    const nm8s * pSrcVec,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 8     | 118   | 14.75      |
| im0     | im0     | 128   | 118   | 0.92       |
| im0     | im0     | 128   | 118   | 0.92       |
| im0     | im0     | 512   | 208   | 0.40       |
| im0     | im0     | 512   | 208   | 0.40       |
| im0     | im0     | 1024  | 398   | 0.38       |
| im0     | im0     | 1024  | 398   | 0.38       |
| im0     | im0     | 2048  | 778   | 0.37       |
| im0     | im0     | 2048  | 778   | 0.37       |
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 8     | 118   | 14.75      |
| im0     | im1     | 128   | 161   | 1.25       |
| im0     | im1     | 128   | 161   | 1.25       |

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im1     | 512   | 271   | 0.52       |
| im0     | im1     | 512   | 271   | 0.52       |
| im0     | im1     | 1024  | 401   | 0.39       |
| im0     | im1     | 1024  | 401   | 0.39       |
| im0     | im1     | 2048  | 661   | 0.32       |
| im0     | im1     | 2048  | 661   | 0.32       |
| im1     | im0     | 8     | 118   | 14.75      |
| im1     | im0     | 8     | 118   | 14.75      |
| im1     | im0     | 128   | 133   | 1.03       |
| im1     | im0     | 128   | 133   | 1.03       |
| im1     | im0     | 512   | 215   | 0.41       |
| im1     | im0     | 512   | 215   | 0.41       |
| im1     | im0     | 1024  | 345   | 0.33       |
| im1     | im0     | 1024  | 345   | 0.33       |
| im1     | im0     | 2048  | 605   | 0.29       |
| im1     | im0     | 2048  | 605   | 0.29       |
| im1     | im1     | 8     | 120   | 15.00      |
| im1     | im1     | 8     | 120   | 15.00      |
| im1     | im1     | 128   | 164   | 1.28       |
| im1     | im1     | 128   | 164   | 1.28       |
| im1     | im1     | 512   | 308   | 0.60       |
| im1     | im1     | 512   | 308   | 0.60       |
| im1     | im1     | 1024  | 498   | 0.48       |
| im1     | im1     | 1024  | 498   | 0.48       |
| im1     | im1     | 2048  | 878   | 0.42       |
| im1     | im1     | 2048  | 878   | 0.42       |

The best configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im0     | im0     | 8     | 118   | 14.75      |

The worst configuration:

| pSrcVec | pDstVec | nSize | ticks | ticks/elem |
|---------|---------|-------|-------|------------|
| im1     | im1     | 2048  | 878   | 0.42       |

## 6.258 nmppsCmpNeC

Сравнивает элементы массива на признак неравенства константе.

### Функции

- void `nmppsCmpNeC_8s` (const `nm8s` \*pSrcVec, `int8b` nCmpVal, `nm8s` \*pDstVec, int nSize)
- void `nmppsCmpNeC_16s` (const `nm16s` \*pSrcVec, `int16b` nCmpVal, `nm16s` \*pDstVec, int nSize)
- void `nmppsCmpNeC_32s` (const `nm32s` \*pSrcVec, `int32b` nCmpVal, `nm32s` \*pDstVec, int nSize)
- void `nmppsCmpNeC_64s` (const `nm64s` \*pSrcVec, `int64b` nCmpVal, `nm64s` \*pDstVec, int nSize)

### 6.258.1 Подробное описание

Сравнивает элементы массива на признак неравенства константе.

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|           |                                                         |
|-----------|---------------------------------------------------------|
| pSrcVec   | Входной вектор.                                         |
| nCmpVal   | Значение константы для сравнения                        |
| nSize     | Размер векторов в элементах.                            |
| nTrueFlag | Значение флага, устанавливаемого при выполнении условия |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

## 6.258.2 Функции

### 6.258.2.1 nmppsCmpNeC\_16s()

```
void nmppsCmpNeC_16s (
    const nm16s * pSrcVec,
    int16b nCmpVal,
    nm16s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 135   | 16.87      |
| im0     | im0     | 0       | 8     | 135   | 16.87      |
| im0     | im0     | 0       | 128   | 190   | 1.48       |
| im0     | im0     | 0       | 128   | 190   | 1.48       |
| im0     | im0     | 0       | 512   | 571   | 1.11       |
| im0     | im0     | 0       | 512   | 571   | 1.11       |
| im0     | im0     | 0       | 1024  | 1079  | 1.05       |
| im0     | im0     | 0       | 1024  | 1079  | 1.05       |
| im0     | im0     | 0       | 2048  | 2095  | 1.02       |
| im0     | im0     | 0       | 2048  | 2095  | 1.02       |
| im0     | im1     | 0       | 8     | 137   | 17.12      |
| im0     | im1     | 0       | 8     | 137   | 17.12      |
| im0     | im1     | 0       | 128   | 252   | 1.96       |
| im0     | im1     | 0       | 128   | 252   | 1.96       |
| im0     | im1     | 0       | 512   | 543   | 1.06       |
| im0     | im1     | 0       | 512   | 543   | 1.06       |
| im0     | im1     | 0       | 1024  | 931   | 0.90       |
| im0     | im1     | 0       | 1024  | 931   | 0.90       |
| im0     | im1     | 0       | 2048  | 1707  | 0.83       |
| im0     | im1     | 0       | 2048  | 1707  | 0.83       |
| im1     | im0     | 0       | 8     | 136   | 17.00      |
| im1     | im0     | 0       | 8     | 136   | 17.00      |
| im1     | im0     | 0       | 128   | 192   | 1.50       |
| im1     | im0     | 0       | 128   | 192   | 1.50       |
| im1     | im0     | 0       | 512   | 489   | 0.95       |
| im1     | im0     | 0       | 512   | 489   | 0.95       |
| im1     | im0     | 0       | 1024  | 881   | 0.86       |
| im1     | im0     | 0       | 1024  | 881   | 0.86       |
| im1     | im0     | 0       | 2048  | 1665  | 0.81       |
| im1     | im0     | 0       | 2048  | 1665  | 0.81       |
| im1     | im1     | 0       | 8     | 138   | 17.25      |
| im1     | im1     | 0       | 8     | 138   | 17.25      |
| im1     | im1     | 0       | 128   | 254   | 1.98       |
| im1     | im1     | 0       | 128   | 254   | 1.98       |
| im1     | im1     | 0       | 512   | 643   | 1.25       |
| im1     | im1     | 0       | 512   | 643   | 1.25       |
| im1     | im1     | 0       | 1024  | 1159  | 1.13       |
| im1     | im1     | 0       | 1024  | 1159  | 1.13       |
| im1     | im1     | 0       | 2048  | 2191  | 1.06       |
| im1     | im1     | 0       | 2048  | 2191  | 1.06       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 135   | 16.87      |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 2191  | 1.06       |

## 6.258.2.2 nmppsCmpNeC\_32s()

```
void nmppsCmpNeC_32s (
    const nm32s * pSrcVec,
    int32b nCmpVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 133   | 16.62      |
| im0     | im0     | 0       | 8     | 133   | 16.62      |
| im0     | im0     | 0       | 128   | 313   | 2.44       |
| im0     | im0     | 0       | 128   | 313   | 2.44       |
| im0     | im0     | 0       | 512   | 1075  | 2.09       |
| im0     | im0     | 0       | 512   | 1075  | 2.09       |
| im0     | im0     | 0       | 1024  | 2091  | 2.04       |
| im0     | im0     | 0       | 1024  | 2091  | 2.04       |
| im0     | im0     | 0       | 2048  | 4123  | 2.01       |
| im0     | im0     | 0       | 2048  | 4123  | 2.01       |
| im0     | im1     | 0       | 8     | 138   | 17.25      |
| im0     | im1     | 0       | 8     | 138   | 17.25      |
| im0     | im1     | 0       | 128   | 345   | 2.69       |
| im0     | im1     | 0       | 128   | 345   | 2.69       |
| im0     | im1     | 0       | 512   | 927   | 1.81       |
| im0     | im1     | 0       | 512   | 927   | 1.81       |
| im0     | im1     | 0       | 1024  | 1703  | 1.66       |
| im0     | im1     | 0       | 1024  | 1703  | 1.66       |
| im0     | im1     | 0       | 2048  | 3255  | 1.58       |
| im0     | im1     | 0       | 2048  | 3255  | 1.58       |
| im1     | im0     | 0       | 8     | 136   | 17.00      |
| im1     | im0     | 0       | 8     | 136   | 17.00      |
| im1     | im0     | 0       | 128   | 289   | 2.25       |
| im1     | im0     | 0       | 128   | 289   | 2.25       |
| im1     | im0     | 0       | 512   | 877   | 1.71       |
| im1     | im0     | 0       | 512   | 877   | 1.71       |
| im1     | im0     | 0       | 1024  | 1661  | 1.62       |
| im1     | im0     | 0       | 1024  | 1661  | 1.62       |
| im1     | im0     | 0       | 2048  | 3229  | 1.57       |
| im1     | im0     | 0       | 2048  | 3229  | 1.57       |
| im1     | im1     | 0       | 8     | 139   | 17.37      |
| im1     | im1     | 0       | 8     | 139   | 17.37      |
| im1     | im1     | 0       | 128   | 381   | 2.97       |
| im1     | im1     | 0       | 128   | 381   | 2.97       |
| im1     | im1     | 0       | 512   | 1155  | 2.25       |
| im1     | im1     | 0       | 512   | 1155  | 2.25       |
| im1     | im1     | 0       | 1024  | 2187  | 2.13       |
| im1     | im1     | 0       | 1024  | 2187  | 2.13       |
| im1     | im1     | 0       | 2048  | 4251  | 2.07       |
| im1     | im1     | 0       | 2048  | 4251  | 2.07       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 133   | 16.62      |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 4251  | 2.07       |

### 6.258.2.3 nmppsCmpNeC\_64s()

```
void nmppsCmpNeC_64s (
    const nm64s * pSrcVec,
    int64b nCmpVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 152   | 19.00      |
| im0     | im0     | 0       | 8     | 152   | 19.00      |
| im0     | im0     | 0       | 128   | 574   | 4.48       |
| im0     | im0     | 0       | 128   | 574   | 4.48       |
| im0     | im0     | 0       | 512   | 2098  | 4.09       |
| im0     | im0     | 0       | 512   | 2098  | 4.09       |
| im0     | im0     | 0       | 1024  | 4130  | 4.03       |
| im0     | im0     | 0       | 1024  | 4130  | 4.03       |
| im0     | im0     | 0       | 2048  | 8194  | 4.00       |
| im0     | im0     | 0       | 2048  | 8194  | 4.00       |
| im0     | im1     | 0       | 8     | 161   | 20.12      |
| im0     | im1     | 0       | 8     | 161   | 20.12      |
| im0     | im1     | 0       | 128   | 546   | 4.26       |
| im0     | im1     | 0       | 128   | 546   | 4.26       |
| im0     | im1     | 0       | 512   | 1710  | 3.33       |
| im0     | im1     | 0       | 512   | 1710  | 3.33       |
| im0     | im1     | 0       | 1024  | 3262  | 3.18       |
| im0     | im1     | 0       | 1024  | 3262  | 3.18       |
| im0     | im1     | 0       | 2048  | 6366  | 3.10       |
| im0     | im1     | 0       | 2048  | 6366  | 3.10       |
| im1     | im0     | 0       | 8     | 152   | 19.00      |
| im1     | im0     | 0       | 8     | 152   | 19.00      |
| im1     | im0     | 0       | 128   | 492   | 3.84       |
| im1     | im0     | 0       | 128   | 492   | 3.84       |
| im1     | im0     | 0       | 512   | 1668  | 3.25       |
| im1     | im0     | 0       | 512   | 1668  | 3.25       |
| im1     | im0     | 0       | 1024  | 3236  | 3.16       |
| im1     | im0     | 0       | 1024  | 3236  | 3.16       |



| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im0     | 0       | 2048  | 6372  | 3.11       |
| im1     | im0     | 0       | 2048  | 6372  | 3.11       |
| im1     | im1     | 0       | 8     | 161   | 20.12      |
| im1     | im1     | 0       | 8     | 161   | 20.12      |
| im1     | im1     | 0       | 128   | 646   | 5.04       |
| im1     | im1     | 0       | 128   | 646   | 5.04       |
| im1     | im1     | 0       | 512   | 2194  | 4.28       |
| im1     | im1     | 0       | 512   | 2194  | 4.28       |
| im1     | im1     | 0       | 1024  | 4258  | 4.15       |
| im1     | im1     | 0       | 1024  | 4258  | 4.15       |
| im1     | im1     | 0       | 2048  | 8386  | 4.09       |
| im1     | im1     | 0       | 2048  | 8386  | 4.09       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 152   | 19.00      |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 8386  | 4.09       |

#### 6.258.2.4 nmppsCmpNeC\_8s()

```
void nmppsCmpNeC_8s (
    const nm8s * pSrcVec,
    int8b nCmpVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 140   | 17.50      |
| im0     | im0     | 0       | 8     | 140   | 17.50      |
| im0     | im0     | 0       | 128   | 170   | 1.32       |
| im0     | im0     | 0       | 128   | 170   | 1.32       |
| im0     | im0     | 0       | 512   | 322   | 0.62       |
| im0     | im0     | 0       | 512   | 322   | 0.62       |
| im0     | im0     | 0       | 1024  | 576   | 0.56       |
| im0     | im0     | 0       | 1024  | 576   | 0.56       |
| im0     | im0     | 0       | 2048  | 1084  | 0.52       |
| im0     | im0     | 0       | 2048  | 1084  | 0.52       |
| im0     | im1     | 0       | 8     | 140   | 17.50      |
| im0     | im1     | 0       | 8     | 140   | 17.50      |

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im1     | 0       | 128   | 195   | 1.52       |
| im0     | im1     | 0       | 128   | 195   | 1.52       |
| im0     | im1     | 0       | 512   | 354   | 0.69       |
| im0     | im1     | 0       | 512   | 354   | 0.69       |
| im0     | im1     | 0       | 1024  | 548   | 0.53       |
| im0     | im1     | 0       | 1024  | 548   | 0.53       |
| im0     | im1     | 0       | 2048  | 936   | 0.45       |
| im0     | im1     | 0       | 2048  | 936   | 0.45       |
| im1     | im0     | 0       | 8     | 140   | 17.50      |
| im1     | im0     | 0       | 8     | 140   | 17.50      |
| im1     | im0     | 0       | 128   | 170   | 1.32       |
| im1     | im0     | 0       | 128   | 170   | 1.32       |
| im1     | im0     | 0       | 512   | 298   | 0.58       |
| im1     | im0     | 0       | 512   | 298   | 0.58       |
| im1     | im0     | 0       | 1024  | 494   | 0.48       |
| im1     | im0     | 0       | 1024  | 494   | 0.48       |
| im1     | im0     | 0       | 2048  | 886   | 0.43       |
| im1     | im0     | 0       | 2048  | 886   | 0.43       |
| im1     | im1     | 0       | 8     | 140   | 17.50      |
| im1     | im1     | 0       | 8     | 140   | 17.50      |
| im1     | im1     | 0       | 128   | 195   | 1.52       |
| im1     | im1     | 0       | 128   | 195   | 1.52       |
| im1     | im1     | 0       | 512   | 390   | 0.76       |
| im1     | im1     | 0       | 512   | 390   | 0.76       |
| im1     | im1     | 0       | 1024  | 648   | 0.63       |
| im1     | im1     | 0       | 1024  | 648   | 0.63       |
| im1     | im1     | 0       | 2048  | 1164  | 0.56       |
| im1     | im1     | 0       | 2048  | 1164  | 0.56       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 8     | 140   | 17.50      |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 1164  | 0.56       |

## 6.259 nmppsCmpNeC\_flag

Сравнивает элементы массива на признак неравенства константе.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) \neq nCmpVal \\ 0, & \text{if } pSrcVec(i) = nCmpVal \end{cases}$$

.

## Функции

- void nmppsCmpNeC\_8u7b (nm8u7b \*pSrcVec, uint7b nCmpVal, nm8s \*pDstVec, int nSize, int8b nTrueFlag)
- void nmppsCmpNeC\_16u15b (nm16u15b \*pSrcVec, uint15b nCmpVal, nm16s \*pDstVec, int nSize, int16b nTrueFlag)

## 6.259.1 Подробное описание

Сравнивает элементы массива на признак неравенства константе.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) \neq nCmpVal \\ 0, & \text{if } pSrcVec(i) = nCmpVal \end{cases}$$

.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) \neq nCmpVal \\ 0, & \text{if } pSrcVec(i) = nCmpVal \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|           |                                                         |
|-----------|---------------------------------------------------------|
| pSrcVec   | Входной вектор.                                         |
| nCmpVal   | Значение константы для сравнения                        |
| nSize     | Размер векторов в элементах.                            |
| nTrueFlag | Значение флага, устанавливаемого при выполнении условия |

## Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

## Возвращает

void

## 6.260 nmppsCmpLtC

Функция сравнения элементов массива с константой (меньше)

Функции

- void [nmppsCmpLtC\\_8s7b](#) (const [nm8s7b](#) \*pSrcVec, [int8b](#) nCmpVal, [nm8s](#) \*pDstVec, int nSize)
- void [nmppsCmpLtC\\_16s15b](#) (const [nm16s15b](#) \*pSrcVec, [int16b](#) nCmpVal, [nm16s](#) \*pDstVec, int nSize)
- void [nmppsCmpLtC\\_32s31b](#) (const [nm32s31b](#) \*pSrcVec, [int32b](#) nCmpVal, [nm32s](#) \*pDstVec, int nSize)
- void [nmppsCmpLtC\\_64s63b](#) (const [nm64s63b](#) \*pSrcVec, [int64b](#) nCmpVal, [nm64s](#) \*pDstVec, int nSize)

### 6.260.1 Подробное описание

Функция сравнения элементов массива с константой (меньше)

Аргументы

|         |                                  |
|---------|----------------------------------|
| pSrcVec | Входной вектор.                  |
| nCmpVal | Значение константы для сравнения |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Аргументы

|       |                              |
|-------|------------------------------|
| nSize | Размер векторов в элементах. |
|-------|------------------------------|

Возвращает

void

### 6.260.2 Функции

## 6.260.2.1 nmppsCmpLtC\_16s15b()

```
void nmppsCmpLtC_16s15b (  
    const nm16s15b * pSrcVec,  
    int16b nCmpVal,  
    nm16s * pDstVec,  
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 2048  | 1560  | 0.76       |
| im0     | im0     | 0       | 2048  | 1560  | 0.76       |
| im0     | im1     | 0       | 2048  | 1203  | 0.58       |
| im0     | im1     | 0       | 2048  | 1203  | 0.58       |
| im1     | im0     | 0       | 2048  | 1170  | 0.57       |
| im1     | im0     | 0       | 2048  | 1170  | 0.57       |
| im1     | im1     | 0       | 2048  | 1683  | 0.82       |
| im1     | im1     | 0       | 2048  | 1683  | 0.82       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im0     | 0       | 2048  | 1170  | 0.57       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 1683  | 0.82       |

#### 6.260.2.2 nmppsCmpLtC\_32s31b()

```
void nmppsCmpLtC_32s31b (
    const nm32s31b * pSrcVec,
    int32b nCmpVal,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 2048  | 3073  | 1.50       |
| im0     | im0     | 0       | 2048  | 3073  | 1.50       |
| im0     | im1     | 0       | 2048  | 2236  | 1.09       |
| im0     | im1     | 0       | 2048  | 2236  | 1.09       |
| im1     | im0     | 0       | 2048  | 2203  | 1.07       |
| im1     | im0     | 0       | 2048  | 2203  | 1.07       |
| im1     | im1     | 0       | 2048  | 3196  | 1.56       |
| im1     | im1     | 0       | 2048  | 3196  | 1.56       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im0     | 0       | 2048  | 2203  | 1.07       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 3196  | 1.56       |

## 6.260.2.3 nmppsCmpLtC\_64s63b()

```
void nmppsCmpLtC_64s63b (
    const nm64s63b * pSrcVec,
    int64b nCmpVal,
    nm64s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 2048  | 6120  | 2.98       |
| im0     | im0     | 0       | 2048  | 6120  | 2.98       |
| im0     | im1     | 0       | 2048  | 4323  | 2.11       |
| im0     | im1     | 0       | 2048  | 4323  | 2.11       |
| im1     | im0     | 0       | 2048  | 4290  | 2.09       |
| im1     | im0     | 0       | 2048  | 4290  | 2.09       |
| im1     | im1     | 0       | 2048  | 6243  | 3.04       |
| im1     | im1     | 0       | 2048  | 6243  | 3.04       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im0     | 0       | 2048  | 4290  | 2.09       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 6243  | 3.04       |

## 6.260.2.4 nmppsCmpLtC\_8s7b()

```
void nmppsCmpLtC_8s7b (
    const nm8s7b * pSrcVec,
    int8b nCmpVal,
    nm8s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im0     | im0     | 0       | 2048  | 802   | 0.39       |
| im0     | im0     | 0       | 2048  | 802   | 0.39       |
| im0     | im1     | 0       | 2048  | 685   | 0.33       |
| im0     | im1     | 0       | 2048  | 685   | 0.33       |
| im1     | im0     | 0       | 2048  | 652   | 0.31       |
| im1     | im0     | 0       | 2048  | 652   | 0.31       |
| im1     | im1     | 0       | 2048  | 925   | 0.45       |
| im1     | im1     | 0       | 2048  | 925   | 0.45       |

The best configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im0     | 0       | 2048  | 652   | 0.31       |

The worst configuration:

| pSrcVec | pDstVec | nCmpVal | nSize | ticks | ticks/elem |
|---------|---------|---------|-------|-------|------------|
| im1     | im1     | 0       | 2048  | 925   | 0.45       |



## 6.261 nmppsCmpGtC

Функция сравнения элементов массива с константой (больше)

Функции

- void nmppsCmpGtC\_8s7b (const nm8s7b \*pSrcVec, int8b nCmpVal, nm8s \*pDstVec, int nSize)
- void nmppsCmpGtC\_16s15b (const nm16s15b \*pSrcVec, int16b nCmpVal, nm16s \*pDstVec, int nSize)
- void nmppsCmpGtC\_32s31b (const nm32s31b \*pSrcVec, int32b nCmpVal, nm32s \*pDstVec, int nSize)
- void nmppsCmpGtC\_64s63b (const nm64s63b \*pSrcVec, int64b nCmpVal, nm64s \*pDstVec, int nSize)
- void nmppsCmpGtC\_v2nm32f (const v2nm32f \*pSrcVec, const v2nm32f \*C, nm1 \*evenFlags, nm1 \*oddFlags, int step, int nSize)

### 6.261.1 Подробное описание

Функция сравнения элементов массива с константой (больше)

Функция сравнения элементов массива с константой (больше или равно)

Аргументы

|         |                                  |
|---------|----------------------------------|
| pSrcVec | Входной вектор.                  |
| nCmpVal | Значение константы для сравнения |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Аргументы

|       |                              |
|-------|------------------------------|
| nSize | Размер векторов в элементах. |
|-------|------------------------------|

Возвращает

void

Аргументы

|    |         |                                                                                                      |
|----|---------|------------------------------------------------------------------------------------------------------|
| in | pSrcVec | входной массив (элемент массива представляет собой структура v2nm32f, состоящая из двух чисел float) |
| in | C       | константа (два числа float)                                                                          |

Возвращаемые значения

|       |                                                                              |
|-------|------------------------------------------------------------------------------|
| [out] | evenFlags маска полученная после сравнения первого поля из структуры v2nm32f |
| [out] | oddFlags маска полученная после сравнения второго поля из структуры v2nm32f  |

Аргументы

|    |       |                                                  |
|----|-------|--------------------------------------------------|
| in | step  | шаг, с которым будут браться элементы из pSrcVec |
| in | nSize | размер массива (в v2nm32f)                       |

В структуре v2nm32f два поля. Первое - v0. Второе - v1.

## 6.262 nmppsCmpGteC

Функция сравнения элементов массива с константой (больше или равно)

Функции

- void nmppsCmpGteC\_v2nm32f (const v2nm32f \*pSrcVec, const v2nm32f \*C, nm1 \*evenFlags, nm1 \*oddFlags, int step, int nSize)

### 6.262.1 Подробное описание

Функция сравнения элементов массива с константой (больше или равно)

Аргументы

|    |         |                                                                                                      |
|----|---------|------------------------------------------------------------------------------------------------------|
| in | pSrcVec | входной массив (элемент массива представляет собой структура v2nm32f, состоящая из двух чисел float) |
| in | C       | константа (два числа float)                                                                          |

Возвращаемые значения

|       |                                                                              |
|-------|------------------------------------------------------------------------------|
| [out] | evenFlags маска полученная после сравнения первого поля из структуры v2nm32f |
| [out] | oddFlags маска полученная после сравнения второго поля из структуры v2nm32f  |

Аргументы

|    |       |                                                  |
|----|-------|--------------------------------------------------|
| in | step  | шаг, с которым будут браться элементы из pSrcVec |
| in | nSize | размер массива (в v2nm32f)                       |

В структуре v2nm32f два поля. Первое - v0. Второе - v1.

## 6.263 nmppsCmpEqV\_

Поэлементное сравнение элементов двух векторов на признак равенства.

Функции

- void nmppsCmpEqV\_16u15b (nm16u15b \*pSrcVec1, nm16u15b \*pSrcVec2, nm16s \*pDstVec, int nSize, int16b nTrueFlag)
- void nmppsCmpEqV\_8u7b (nm8u7b \*pSrcVec1, nm8u7b \*pSrcVec2, nm8s \*pDstVec, int nSize, int8b nTrueFlag)

### 6.263.1 Подробное описание

Поэлементное сравнение элементов двух векторов на признак равенства.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) = nCmpVal \\ 0, & \text{if } pSrcVec(i) \neq nCmpVal \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

Аргументы

|           |                                                         |
|-----------|---------------------------------------------------------|
| pSrcVec1  | Первый входной вектор.                                  |
| pSrcVec2  | Второй входной вектор.                                  |
| nSize     | Размер векторов в элементах.                            |
| nTrueFlag | Значение флага, устанавливаемого при выполнении условия |

Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

Возвращает

void

## 6.264 nmppsCmpNe

Сравнивает элементы массива на признак неравенства константе.

### Функции

- void `nmppsCmpNe_8s` (const `nm8s` \*pSrcVec1, const `nm8s` \*pSrcVec2, `nm8s` \*pDstVec, int nSize)
- void `nmppsCmpNe_16s` (const `nm16s` \*pSrcVec1, const `nm16s` \*pSrcVec2, `nm16s` \*pDstVec, int nSize)
- void `nmppsCmpNe_32s` (const `nm32s` \*pSrcVec1, const `nm32s` \*pSrcVec2, `nm32s` \*pDstVec, int nSize)
- void `nmppsCmpNe_64s` (const `nm64s` \*pSrcVec1, const `nm64s` \*pSrcVec2, `nm64s` \*pDstVec, int nSize)

### 6.264.1 Подробное описание

Сравнивает элементы массива на признак неравенства константе.

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|          |                             |
|----------|-----------------------------|
| pSrcVec1 | Входной вектор1             |
| pSrcVec2 | Входной вектор2             |
| nSize    | Размер векторов в элементах |

### Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

### Возвращает

void

### 6.264.2 Функции

#### 6.264.2.1 nmppsCmpNe\_16s()

```
void nmppsCmpNe_16s (
    const nm16s * pSrcVec1,
```

```
const nm16s * pSrcVec2,  
nm16s * pDstVec,  
int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2190  | 1.06       |
| im0      | im1      | im2     | 2048  | 2190  | 1.06       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2190  | 1.06       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 2190  | 1.06       |

### 6.264.2.2 nmppsCmpNe\_32s()

```
void nmppsCmpNe_32s (
    const nm32s * pSrcVec1,
    const nm32s * pSrcVec2,
    nm32s * pDstVec,
    int nSize )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4302  | 2.10       |
| im0      | im1      | im2     | 2048  | 4302  | 2.10       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4302  | 2.10       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 4302  | 2.10       |

### 6.264.2.3 nmppsCmpNe\_64s()

```
void nmppsCmpNe_64s (
    const nm64s * pSrcVec1,
```

```
const nm64s * pSrcVec2,
nm64s * pDstVec,
int nSize )
```

Performance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 8529  | 4.16       |
| im0      | im1      | im2     | 2048  | 8529  | 4.16       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 8529  | 4.16       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 8529  | 4.16       |

#### 6.264.2.4 nmppsCmpNe\_8s()

```
void nmppsCmpNe_8s (
    const nm8s * pSrcVec1,
    const nm8s * pSrcVec2,
    nm8s * pDstVec,
    int nSize )
```

Performance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1134  | 0.55       |
| im0      | im1      | im2     | 2048  | 1134  | 0.55       |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1134  | 0.55       |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | ticks | ticks/elem |
|----------|----------|---------|-------|-------|------------|
| im0      | im1      | im2     | 2048  | 1134  | 0.55       |



## 6.265 nmppsCmpLt

Функция сравнения элементов массива с константой (меньше)

Функции

- void nmppsCmpLt\_8s7b (const nm8s \*pSrcVec1, const nm8s \*pSrcVec2, nm8s \*pDstVec, int nSize)
- void nmppsCmpLt\_16s15b (const nm16s \*pSrcVec1, const nm16s \*pSrcVec2, nm16s \*pDstVec, int nSize)
- void nmppsCmpLt\_32s31b (const nm32s \*pSrcVec1, const nm32s \*pSrcVec2, nm32s \*pDstVec, int nSize)
- void nmppsCmpLt\_64s63b (const nm64s \*pSrcVec1, const nm64s \*pSrcVec2, nm64s \*pDstVec, int nSize)

### 6.265.1 Подробное описание

Функция сравнения элементов массива с константой (меньше)

Аргументы

|          |                 |
|----------|-----------------|
| pSrcVec1 | Входной вектор1 |
| pSrcVec2 | Входной вектор2 |

Возвращаемые значения

|         |                       |
|---------|-----------------------|
| pDstVec | Результирующий вектор |
|---------|-----------------------|

Аргументы

|       |                             |
|-------|-----------------------------|
| nSize | Размер векторов в элементах |
|-------|-----------------------------|

Возвращает

void

## 6.266 nmppsCmpNeV\_

Поэлементное сравнение элементов двух векторов на признак неравенства.

### Функции

- void `nmppsCmpNeV_16u` (`nm16u15b` \*pSrcVec1, `nm16u15b` \*pSrcVec2, `nm16s` \*pDstVec, int nSize, `int16b` nTrueFlag)
- void `nmppsCmpNeV_8u` (`nm8u7b` \*pSrcVec1, `nm8u7b` \*pSrcVec2, `nm8s` \*pDstVec, int nSize, `int8b` nTrueFlag)

### 6.266.1 Подробное описание

Поэлементное сравнение элементов двух векторов на признак неравенства.

$$pDstVec(i) = \begin{cases} nTrueFlag, & \text{if } pSrcVec(i) \neq nCmpVal \\ 0, & \text{if } pSrcVec(i) = nCmpVal \end{cases}$$

$$i = \overline{0 \dots nSize - 1}$$

### Аргументы

|           |                                                         |
|-----------|---------------------------------------------------------|
| pSrcVec1  | Первый входной вектор.                                  |
| pSrcVec2  | Второй входной вектор.                                  |
| nSize     | Размер векторов в элементах.                            |
| nTrueFlag | Значение флага, устанавливаемого при выполнении условия |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

### 6.266.2 Функции

## 6.266.2.1 nmppsCmpNeV\_16u()

```
void nmppsCmpNeV_16u (
    nm16u15b * pSrcVec1,
    nm16u15b * pSrcVec2,
    nm16s * pDstVec,
    int nSize,
    int16b nTrueFlag )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im0      | im0      | im2     | 2048  | 1         | 1200  | 1200.00    |
| im0      | im0      | im2     | 2048  | 1         | 1200  | 1200.00    |
| im0      | im0      | im3     | 2048  | 1         | 1200  | 1200.00    |
| im0      | im0      | im3     | 2048  | 1         | 1200  | 1200.00    |
| im0      | im1      | im2     | 2048  | 1         | 1179  | 1179.00    |
| im0      | im1      | im2     | 2048  | 1         | 1179  | 1179.00    |
| im0      | im1      | im3     | 2048  | 1         | 1179  | 1179.00    |
| im0      | im1      | im3     | 2048  | 1         | 1179  | 1179.00    |
| im1      | im0      | im2     | 2048  | 1         | 1177  | 1177.00    |
| im1      | im0      | im2     | 2048  | 1         | 1177  | 1177.00    |
| im1      | im0      | im3     | 2048  | 1         | 1177  | 1177.00    |
| im1      | im0      | im3     | 2048  | 1         | 1177  | 1177.00    |
| im1      | im1      | im2     | 2048  | 1         | 1271  | 1271.00    |
| im1      | im1      | im2     | 2048  | 1         | 1271  | 1271.00    |
| im1      | im1      | im3     | 2048  | 1         | 1271  | 1271.00    |
| im1      | im1      | im3     | 2048  | 1         | 1271  | 1271.00    |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im1      | im0      | im2     | 2048  | 1         | 1177  | 1177.00    |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im1      | im1      | im2     | 2048  | 1         | 1271  | 1271.00    |

## 6.266.2.2 nmppsCmpNeV\_8u()

```
void nmppsCmpNeV_8u (
    nm8u7b * pSrcVec1,
    nm8u7b * pSrcVec2,
    nm8s * pDstVec,
    int nSize,
    int8b nTrueFlag )
```

Perfomance table 0

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im0      | im0      | im2     | 2048  | 1         | 1213  | 1213.00    |
| im0      | im0      | im2     | 2048  | 1         | 1213  | 1213.00    |
| im0      | im0      | im3     | 2048  | 1         | 1213  | 1213.00    |
| im0      | im0      | im3     | 2048  | 1         | 1213  | 1213.00    |
| im0      | im1      | im2     | 2048  | 1         | 1192  | 1192.00    |
| im0      | im1      | im2     | 2048  | 1         | 1192  | 1192.00    |
| im0      | im1      | im3     | 2048  | 1         | 1192  | 1192.00    |
| im0      | im1      | im3     | 2048  | 1         | 1192  | 1192.00    |
| im1      | im0      | im2     | 2048  | 1         | 1190  | 1190.00    |
| im1      | im0      | im2     | 2048  | 1         | 1190  | 1190.00    |
| im1      | im0      | im3     | 2048  | 1         | 1190  | 1190.00    |
| im1      | im0      | im3     | 2048  | 1         | 1190  | 1190.00    |
| im1      | im1      | im2     | 2048  | 1         | 1284  | 1284.00    |
| im1      | im1      | im2     | 2048  | 1         | 1284  | 1284.00    |
| im1      | im1      | im3     | 2048  | 1         | 1284  | 1284.00    |
| im1      | im1      | im3     | 2048  | 1         | 1284  | 1284.00    |

The best configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im1      | im0      | im2     | 2048  | 1         | 1190  | 1190.00    |

The worst configuration:

| pSrcVec1 | pSrcVec2 | pDstVec | nSize | nTrueFlag | ticks | ticks/elem |
|----------|----------|---------|-------|-----------|-------|------------|
| im1      | im1      | im2     | 2048  | 1         | 1284  | 1284.00    |

## 6.267 Vec\_ClipRShiftConvert\_AddC

Изменение разрядности элементов вектора с клиппированием

Преобразование 32p знаковых данных к восьмиразрядным осуществляется вырезанием битов 16..31., затем клиппированием и вырезанием восьми младших бит результата. При клиппировании используется глобальная константа типа nm64s с именем `_F1CR_16_to_x`. По умолчанию, там лежит `ff00_ff00_ff00_ff00` соответствующая диапазону результата [-255..256]. К каждому полученному 64p слову результата прибавляется значение глобальной переменной `_VR_16_to_x`, которая по умолчанию равна нулю.

### Функции

- `void Vec_ClipRShiftConvert_AddC (nm32s *pSrcVec, nm8u *pDstVec, int nSize)`

### 6.267.1 Подробное описание

Изменение разрядности элементов вектора с клиппированием

Преобразование 32p знаковых данных к восьмиразрядным осуществляется вырезанием битов 16..31., затем клиппированием и вырезанием восьми младших бит результата. При клиппировании используется глобальная константа типа nm64s с именем `_F1CR_16_to_x`. По умолчанию, там лежит `ff00_ff00_ff00_ff00` соответствующая диапазону результата [-255..256]. К каждому полученному 64p слову результата прибавляется значение глобальной переменной `_VR_16_to_x`, которая по умолчанию равна нулю.

### Аргументы

|         |                              |
|---------|------------------------------|
| pSrcVec | Входной вектор.              |
| nSize   | Размер векторов в элементах. |

### Возвращаемые значения

|         |                        |
|---------|------------------------|
| pDstVec | Результирующий вектор. |
|---------|------------------------|

### Возвращает

void

## 6.268 nmppsAddr\_

Возвращает адрес ячейки памяти, содержащей указанный элемент.

Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.

### Функции

- `__INLINE__ nm1 * nmppsAddr_1 (const nm1 *pVec, int nIndex)`
- `nm2s * nmppsAddr_2s (const nm2s *pVec, int nIndex)`
- `nm4s * nmppsAddr_4s (const nm4s *pVec, int nIndex)`
- `__INLINE__ nm8s * nmppsAddr_8s (const nm8s *pVec, int nIndex)`
- `__INLINE__ nm16s * nmppsAddr_16s (const nm16s *pVec, int nIndex)`
- `__INLINE__ nm32s * nmppsAddr_32s (const nm32s *pVec, int nIndex)`
- `__INLINE__ nm32f * nmppsAddr_32f (const nm32f *pVec, int nIndex)`
- `__INLINE__ nm64s * nmppsAddr_64s (const nm64s *pVec, int nIndex)`
- `__INLINE__ nm64f * nmppsAddr_64f (const nm64f *pVec, int nIndex)`
- `nm2u * nmppsAddr_2u (const nm2u *pVec, int nIndex)`
- `nm4u * nmppsAddr_4u (const nm4u *pVec, int nIndex)`
- `__INLINE__ nm8u * nmppsAddr_8u (const nm8u *pVec, int nIndex)`
- `__INLINE__ nm16u * nmppsAddr_16u (const nm16u *pVec, int nIndex)`
- `__INLINE__ nm32u * nmppsAddr_32u (const nm32u *pVec, int nIndex)`
- `__INLINE__ nm64u * nmppsAddr_64u (const nm64u *pVec, int nIndex)`

### 6.268.1 Подробное описание

Возвращает адрес ячейки памяти, содержащей указанный элемент.

Реализация для процессора NeuroMatrix возвращает адрес, выровненный в памяти на 32 бита.

#### Аргументы

|        |                  |
|--------|------------------|
| pVec   | Входной вектор.  |
| nIndex | Индекс элемента. |

Возвращает

Адрес ячейки памяти.

#### Заметки

Для ускорения работы на PC возможно использование макроса ADDR(ptr, index), который раскрывается на PC как (ptr+index), а на NM как вызов функции nmppsAddr\_.

### 6.268.2 Функции

## 6.268.2.1 nmppsAddr\_1()

```
__INLINE__ nm1 * nmppsAddr_1 (
    const nm1 * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 108

## 6.268.2.2 nmppsAddr\_16s()

```
__INLINE__ nm16s * nmppsAddr_16s (
    const nm16s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 112

#### 6.268.2.3 nmppsAddr\_16u()

```
__INLINE__ nm16u * nmppsAddr_16u (
    const nm16u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 122

#### 6.268.2.4 nmppsAddr\_2s()

```
nm2s * nmppsAddr_2s (
    const nm2s * pVec,
    int nIndex ) [inline]
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |



The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 109

#### 6.268.2.5 nmppsAddr\_2u()

```
nm2u * nmppsAddr_2u (
    const nm2u * pVec,
    int nIndex ) [inline]
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 119

#### 6.268.2.6 nmppsAddr\_32f()

```
__INLINE__ nm32f * nmppsAddr_32f (
    const nm32f * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 163   | 162.99     |
| im1  | 0      | 163   | 162.99     |
| im2  | 0      | 163   | 162.99     |
| im2  | 0      | 163   | 162.99     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 163   | 162.99     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 163   | 162.99     |

См. определение в файле vSupport.h строка 114

#### 6.268.2.7 nmppsAddr\_32s()

```
__INLINE__ nm32s * nmppsAddr_32s (
    const nm32s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 113

## 6.268.2.8 nmppsAddr\_32u()

```
__INLINE__ nm32u * nmppsAddr_32u (
    const nm32u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 123

## 6.268.2.9 nmppsAddr\_4s()

```
nm4s * nmppsAddr_4s (
    const nm4s * pVec,
    int nIndex ) [inline]
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 110

6.268.2.10 nmppsAddr\_4u()

```
nm4u * nmppsAddr_4u (
    const nm4u * pVec,
    int nIndex ) [inline]
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 120

6.268.2.11 nmppsAddr\_64f()

```
__INLINE__ nm64f * nmppsAddr_64f (
    const nm64f * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 127   | 126.99     |
| im1  | 0      | 127   | 126.99     |
| im2  | 0      | 127   | 126.99     |
| im2  | 0      | 127   | 126.99     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 127   | 126.99     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 127   | 126.99     |

См. определение в файле vSupport.h строка 116

#### 6.268.2.12 nmppsAddr\_64s()

```
__INLINE__ nm64s * nmppsAddr_64s (
    const nm64s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |
| im1  | 0      | 135   | 135.00     |
| im2  | 0      | 135   | 135.00     |
| im2  | 0      | 135   | 135.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |

См. определение в файле vSupport.h строка 115

#### 6.268.2.13 nmppsAddr\_64u()

```
__INLINE__ nm64u * nmppsAddr_64u (
    const nm64u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |
| im1  | 0      | 135   | 135.00     |
| im2  | 0      | 135   | 135.00     |
| im2  | 0      | 135   | 135.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 135   | 135.00     |

См. определение в файле vSupport.h строка 124

6.268.2.14 nmppsAddr\_8s()

```
__INLINE__ nm8s * nmppsAddr_8s (
    const nm8s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 111

## 6.268.2.15 nmppsAddr\_8u()

```
__INLINE__ nm8u * nmppsAddr_8u (
    const nm8u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |
| im1  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |
| im2  | 0      | 174   | 174.00     |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im1  | 0      | 174   | 174.00     |

См. определение в файле vSupport.h строка 121

## 6.269 nmppsSetVal\_

Модификация элемента вектора.

Функции

- void nmppsPut\_1 (nm1 \*pVec, int nIndex, int1b nVal)
- void nmppsPut\_2s (nm2s \*pVec, int nIndex, int2b nVal)
- void nmppsPut\_4s (nm4s \*pVec, int nIndex, int4b nVal)
- void nmppsPut\_8s (nm8s \*pVec, int nIndex, int8b nVal)
- void nmppsPut\_16s (nm16s \*pVec, int nIndex, int16b nVal)
- \_\_INLINE\_\_ void nmppsPut\_32s (nm32s \*pVec, int nIndex, int32b nVal)
- \_\_INLINE\_\_ void nmppsPut\_64s (nm64s \*pVec, int nIndex, int64b nVal)
- \_\_INLINE\_\_ void nmppsPut\_2u (nm2u \*pVec, int nIndex, uint2b nVal)
- \_\_INLINE\_\_ void nmppsPut\_4u (nm4u \*pVec, int nIndex, uint4b nVal)
- \_\_INLINE\_\_ void nmppsPut\_8u (nm8u \*pVec, int nIndex, uint8b nVal)
- \_\_INLINE\_\_ void nmppsPut\_16u (nm16u \*pVec, int nIndex, uint16b nVal)
- \_\_INLINE\_\_ void nmppsPut\_32u (nm32u \*pVec, int nIndex, uint32b nVal)
- \_\_INLINE\_\_ void nmppsPut\_64u (nm64u \*pVec, int nIndex, uint64b nVal)

### 6.269.1 Подробное описание

Модификация элемента вектора.

$$pVec(nIndex) = Val$$

Аргументы

|        |                   |
|--------|-------------------|
| pVec   | Вектор.           |
| nIndex | Позиция элемента  |
| nVal   | Значение элемента |

Возвращает

void

### 6.269.2 Функции



## 6.269.2.1 nmppsPut\_1()

```
void nmppsPut_1 (
    nm1 * pVec,
    int nIndex,
    int1b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |
| im0  | 3    | 0      | 89    | 89.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 90    | 90.00      |

#### 6.269.2.2 nmppsPut\_16s()

```
void nmppsPut_16s (
    nm16s * pVec,
    int nIndex,
    int16b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |
| im0  | 3    | 0      | 89    | 89.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 90    | 90.00      |

## 6.269.2.3 nmppsPut\_16u()

```
__INLINE__ void nmppsPut_16u (
    nm16u * pVec,
    int nIndex,
    uint16b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |
| im0  | 3    | 0      | 142   | 142.00     |
| im1  | 3    | 0      | 143   | 143.00     |
| im1  | 3    | 0      | 143   | 143.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 143   | 143.00     |

См. определение в файле vSupport.h строка 180

## 6.269.2.4 nmppsPut\_2s()

```
void nmppsPut_2s (
    nm2s * pVec,
    int nIndex,
    int2b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |
| im0  | 3    | 0      | 89    | 89.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 90    | 90.00      |

6.269.2.5 nmppsPut\_2u()

```
__INLINE__ void nmppsPut_2u (
    nm2u * pVec,
    int nIndex,
    uint2b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |
| im0  | 3    | 0      | 142   | 142.00     |
| im1  | 3    | 0      | 143   | 143.00     |
| im1  | 3    | 0      | 143   | 143.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 143   | 143.00     |

См. определение в файле vSupport.h строка 177

6.269.2.6 nmppsPut\_32s()

```
__INLINE__ void nmppsPut_32s (
    nm32s * pVec,
    int nIndex,
    int32b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |
| im0  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |

См. определение в файле vSupport.h строка 174

#### 6.269.2.7 nmppsPut\_32u()

```
__INLINE__ void nmppsPut_32u (
    nm32u * pVec,
    int nIndex,
    uint32b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |
| im0  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 90    | 90.00      |

См. определение в файле vSupport.h строка 181

#### 6.269.2.8 nmppsPut\_4s()

```
void nmppsPut_4s (
    nm4s * pVec,
```

```
int nIndex,
uint4b nVal )
```

Performance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |
| im0  | 3    | 0      | 89    | 89.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 90    | 90.00      |

#### 6.269.2.9 nmppsPut\_4u()

```
__INLINE__ void nmppsPut_4u (
    nm4u * pVec,
    int nIndex,
    uint4b nVal )
```

Performance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |
| im0  | 3    | 0      | 142   | 142.00     |
| im1  | 3    | 0      | 143   | 143.00     |
| im1  | 3    | 0      | 143   | 143.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 143   | 143.00     |

См. определение в файле vSupport.h строка 178

#### 6.269.2.10 nmppsPut\_64s()

```
__INLINE__ void nmppsPut_64s (
    nm64s * pVec,
    int nIndex,
    int64b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |
| im0  | 3    | 0      | 151   | 151.00     |
| im1  | 3    | 0      | 151   | 151.00     |
| im1  | 3    | 0      | 151   | 151.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |

См. определение в файле vSupport.h строка 175

#### 6.269.2.11 nmppsPut\_64u()

```
__INLINE__ void nmppsPut_64u (
    nm64u * pVec,
    int nIndex,
    uint64b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |
| im0  | 3    | 0      | 151   | 151.00     |
| im1  | 3    | 0      | 151   | 151.00     |
| im1  | 3    | 0      | 151   | 151.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 151   | 151.00     |

См. определение в файле vSupport.h строка 182

#### 6.269.2.12 nmppsPut\_8s()

```
void nmppsPut_8s (
    nm8s * pVec,
    int nIndex,
    int8b nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |
| im0  | 3    | 0      | 89    | 89.00      |
| im1  | 3    | 0      | 90    | 90.00      |
| im1  | 3    | 0      | 90    | 90.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 89    | 89.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 90    | 90.00      |

#### 6.269.2.13 nmppsPut\_8u()

```
__INLINE__ void nmppsPut_8u (
    nm8u * pVec,
    int nIndex,
    uint8b nVal )
```



Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |
| im0  | 3    | 0      | 142   | 142.00     |
| im1  | 3    | 0      | 143   | 143.00     |
| im1  | 3    | 0      | 143   | 143.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | 3    | 0      | 142   | 142.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im1  | 3    | 0      | 143   | 143.00     |

См. определение в файле vSupport.h строка 179

## 6.270 nmppsGetVal\_

Извлекает значение элемента вектора.

### Функции

- void [nmppsGetVal\\_1](#) (const [nm1](#) \*pVec, int nIndex, [int1b](#) \*nVal)
- void [nmppsGetVal\\_2s](#) (const [nm2s](#) \*pVec, int nIndex, [int2b](#) \*nVal)
- void [nmppsGetVal\\_4s](#) (const [nm4s](#) \*pVec, int nIndex, [int4b](#) \*nVal)
- void [nmppsGetVal\\_8s](#) (const [nm8s](#) \*pVec, int nIndex, [int8b](#) \*nVal)
- void [nmppsGetVal\\_16s](#) (const [nm16s](#) \*pVec, int nIndex, [int16b](#) \*nVal)
- `__INLINE__` void [nmppsGetVal\\_32s](#) (const [nm32s](#) \*pVec, int nIndex, [int32b](#) \*nVal)
- `__INLINE__` void [nmppsGetVal\\_64s](#) (const [nm64s](#) \*pVec, int nIndex, [int64b](#) \*nVal)
- void [nmppsGetVal\\_2u](#) (const [nm2u](#) \*pVec, int nIndex, [uint2b](#) \*nVal)
- void [nmppsGetVal\\_4u](#) (const [nm4u](#) \*pVec, int nIndex, [uint4b](#) \*nVal)
- void [nmppsGetVal\\_8u](#) (const [nm8u](#) \*pVec, int nIndex, [uint8b](#) \*nVal)
- void [nmppsGetVal\\_16u](#) (const [nm16u](#) \*pVec, int nIndex, [uint16b](#) \*nVal)
- `__INLINE__` void [nmppsGetVal\\_32u](#) (const [nm32u](#) \*pVec, int nIndex, [uint32b](#) \*nVal)
- `__INLINE__` void [nmppsGetVal\\_64u](#) (const [nm64u](#) \*pVec, int nIndex, [uint64b](#) \*nVal)

### 6.270.1 Подробное описание

Извлекает значение элемента вектора.

Аргументы

|        |                   |
|--------|-------------------|
| pVec   | Вектор.           |
| nIndex | Позиция элемента. |

Возвращаемые значения

|      |                    |
|------|--------------------|
| nVal | Значение элемента. |
|------|--------------------|

Возвращает

void

### 6.270.2 Функции

## 6.270.2.1 nmppsGetVal\_1()

```
void nmppsGetVal_1 (
    const nm1 * pVec,
    int nIndex,
    int1b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

#### 6.270.2.2 nmppsGetVal\_16s()

```
void nmppsGetVal_16s (
    const nm16s * pVec,
    int nIndex,
    int16b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.270.2.3 nmppsGetVal\_16u()

```
void nmppsGetVal_16u (
    const nm16u * pVec,
    int nIndex,
    uint16b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.270.2.4 nmppsGetVal\_2s()

```
void nmppsGetVal_2s (
    const nm2s * pVec,
    int nIndex,
    int2b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.270.2.5 nmppsGetVal\_2u()

```
void nmppsGetVal_2u (
    const nm2u * pVec,
    int nIndex,
    uint2b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.270.2.6 nmppsGetVal\_32s()

```
__INLINE__ void nmppsGetVal_32s (
    const nm32s * pVec,
    int nIndex,
    int32b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |
| im0  | im2  | 0      | 95    | 95.00      |
| im1  | im2  | 0      | 95    | 95.00      |
| im1  | im2  | 0      | 95    | 95.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |

См. определение в файле vSupport.h строка 227

#### 6.270.2.7 nmppsGetVal\_32u()

```
__INLINE__ void nmppsGetVal_32u (
    const nm32u * pVec,
    int nIndex,
    uint32b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |
| im0  | im2  | 0      | 95    | 95.00      |
| im1  | im2  | 0      | 95    | 95.00      |
| im1  | im2  | 0      | 95    | 95.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 95    | 95.00      |

См. определение в файле vSupport.h строка 234

#### 6.270.2.8 nmppsGetVal\_4s()

```
void nmppsGetVal_4s (
    const nm4s * pVec,
```



```
int nIndex,
uint4b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

#### 6.270.2.9 nmppsGetVal\_4u()

```
void nmppsGetVal_4u (
    const nm4u * pVec,
    int nIndex,
    uint4b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.270.2.10 nmppsGetVal\_64s()

```
__INLINE__ void nmppsGetVal_64s (
    const nm64s * pVec,
    int nIndex,
    int64b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |
| im0  | im2  | 0      | 157   | 157.00     |
| im1  | im2  | 0      | 157   | 157.00     |
| im1  | im2  | 0      | 157   | 157.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |

См. определение в файле vSupport.h строка 228

## 6.270.2.11 nmppsGetVal\_64u()

```
__INLINE__ void nmppsGetVal_64u (
    const nm64u * pVec,
    int nIndex,
    uint64b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |
| im0  | im2  | 0      | 157   | 157.00     |
| im1  | im2  | 0      | 157   | 157.00     |
| im1  | im2  | 0      | 157   | 157.00     |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 157   | 157.00     |

См. определение в файле vSupport.h строка 235

#### 6.270.2.12 nmppsGetVal\_8s()

```
void nmppsGetVal_8s (
    const nm8s * pVec,
    int nIndex,
    int8b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

#### 6.270.2.13 nmppsGetVal\_8u()

```
void nmppsGetVal_8u (
    const nm8u * pVec,
    int nIndex,
    uint8b * nVal )
```

Perfomance table 0

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |
| im0  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |
| im1  | im2  | 0      | 86    | 86.00      |

The best configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

The worst configuration:

| pVec | nVal | nIndex | ticks | ticks/elem |
|------|------|--------|-------|------------|
| im0  | im2  | 0      | 86    | 86.00      |

## 6.271 nmppsGetVal\_(return)

Извлекает значение элемента вектора.

### Функции

- `int2b nmppsGet_2s` (const `nm2s` \*pVec, int nIndex)
- `int4b nmppsGet_4s` (const `nm4s` \*pVec, int nIndex)
- `int8b nmppsGet_8s` (const `nm8s` \*pVec, int nIndex)
- `int16b nmppsGet_16s` (const `nm16s` \*pVec, int nIndex)
- `__INLINE__ int32b nmppsGet_32s` (const `nm32s` \*pVec, int nIndex)
- `uint1b nmppsGet_1` (const `nm1` \*pVec, int nIndex)
- `uint2b nmppsGet_2u` (const `nm2u` \*pVec, int nIndex)
- `uint4b nmppsGet_4u` (const `nm4u` \*pVec, int nIndex)
- `uint8b nmppsGet_8u` (const `nm8u` \*pVec, int nIndex)
- `uint16b nmppsGet_16u` (const `nm16u` \*pVec, int nIndex)
- `__INLINE__ uint32b nmppsGet_32u` (const `nm32u` \*pVec, int nIndex)

### 6.271.1 Подробное описание

Извлекает значение элемента вектора.

#### Аргументы

|        |                   |
|--------|-------------------|
| pVec   | Вектор.           |
| nIndex | Позиция элемента. |

Возвращает

Значение элемента.

### 6.271.2 Функции

#### 6.271.2.1 nmppsGet\_1()

```
uint1b nmppsGet_1 (
    const nm1 * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

#### 6.271.2.2 nmppsGet\_16s()

```
int16b nmppsGet_16s (
    const nm16s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

## 6.271.2.3 nmppsGet\_16u()

```
uint16b nmppsGet_16u (
    const nm16u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

## 6.271.2.4 nmppsGet\_2s()

```
int2b nmppsGet_2s (
    const nm2s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

#### 6.271.2.5 nmppsGet\_2u()

```
uint2b nmppsGet_2u (
    const nm2u * pVec,
    int nIndex )
```

#### Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

#### 6.271.2.6 nmppsGet\_32s()

```
__INLINE__ int32b nmppsGet_32s (
    const nm32s * pVec,
    int nIndex )
```

#### Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |
| im0  | 0      | 87    | 87.00      |
| im1  | 0      | 87    | 87.00      |
| im1  | 0      | 87    | 87.00      |

The best configuration:



| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |

См. определение в файле vSupport.h строка 275

#### 6.271.2.7 nmppsGet\_32u()

```
__INLINE__ uint32b nmppsGet_32u (
    const nm32u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |
| im0  | 0      | 87    | 87.00      |
| im1  | 0      | 87    | 87.00      |
| im1  | 0      | 87    | 87.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 87    | 87.00      |

См. определение в файле vSupport.h строка 282

#### 6.271.2.8 nmppsGet\_4s()

```
int4b nmppsGet_4s (
    const nm4s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

#### 6.271.2.9 nmppsGet\_4u()

```
uint4b nmppsGet_4u (
    const nm4u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

## 6.271.2.10 nmppsGet\_8s()

```
int8b nmppsGet_8s (
    const nm8s * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

## 6.271.2.11 nmppsGet\_8u()

```
uint8b nmppsGet_8u (
    const nm8u * pVec,
    int nIndex )
```

Perfomance table 0

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |
| im0  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |
| im1  | 0      | 82    | 82.00      |

The best configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

The worst configuration:

| pVec | nIndex | ticks | ticks/elem |
|------|--------|-------|------------|
| im0  | 0      | 82    | 82.00      |

## 6.272 VEC\_QSort

Сортировка массива по убыванию.

Функции

- void `nmppsQSort_32s` (`nm32s` \*pSrcDstVec, int nSize)

### 6.272.1 Подробное описание

Сортировка массива по убыванию.

Аргументы

|            |                                  |
|------------|----------------------------------|
| pSrcDstVec | Входной и результирующий вектор. |
| nSize      | Размер вектора в элементах.      |

Возвращает

void

Restrictions:

Функция работает рекурсивно, передавая параметры через стек, поэтому размер стека должен быть больше  $4 \cdot \log_2(nSize)$  32-битных слов в лучшем случае (элементы массива расположены беспорядочно) и больше  $6 \cdot nSize$  32-битных слов в худшем случае (элементы массива уже упорядочены)

### 6.272.2 Функции

#### 6.272.2.1 nmppsQSort\_32s()

```
void nmppsQSort_32s (
    nm32s * pSrcDstVec,
    int nSize )
```

Perfomance table 0

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im0        | 256   | 59895 | 233.96     |
| im0        | 256   | 59895 | 233.96     |
| im1        | 256   | 61299 | 239.44     |
| im1        | 256   | 61299 | 239.44     |

The best configuration:

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im0        | 256   | 59895 | 233.96     |

The worst configuration:

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im1        | 256   | 61299 | 239.44     |

Perfomance table 1

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im1        | 8     | 987   | 123.37     |
| im1        | 8     | 987   | 123.37     |
| im1        | 128   | 29143 | 227.67     |
| im1        | 128   | 29143 | 227.67     |
| im1        | 256   | 83871 | 327.62     |
| im1        | 256   | 83871 | 327.62     |

The best configuration:

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im1        | 8     | 987   | 123.37     |

The worst configuration:

| pSrcDstVec | nSize | ticks | ticks/elem |
|------------|-------|-------|------------|
| im1        | 256   | 83871 | 327.62     |

## 6.273 nmppsRemap\_

Переупорядочивание элементов вектора по таблице.

## Функции

- void nmppsRemap\_32u (nm32u \*pSrcVec, nm32u \*pDstVec, nm32s \*pRemapTable, int nDstVecSize)
- void nmppsRemap\_8u (nm8u \*pSrcVec, nm8u \*pDstVec, nm32s \*pRemapTable, int nSrcVecSize, int nDstVecSize, void \*pTmpBuf1, void \*pTmpBuf2)

## 6.273.1 Подробное описание

Переупорядочивание элементов вектора по таблице.

$$pDstVec[i] = pSrcVec[pRemapTable[i]],$$

$$i = \overline{0 \dots nSize - 1}$$

## Аргументы

|             |                                                                                                |
|-------------|------------------------------------------------------------------------------------------------|
| pSrcVec     | Входной вектор.                                                                                |
| pRemapTable | Таблица новых индексов для переупорядочивания.                                                 |
| nDstVecSize | Размер результирующего вектора в элементах.                                                    |
| pTmpBuf1    | Временный массив nm32u pTmpBuf1[nSrcVecSize].                                                  |
| pTmpBuf2    | Временный массив nm32u pTmpBuf2[nDstVecSize]. Результирующий вектор nm8u pDstVec[nDstVecSize]. |

## Возвращает

void

```
// Функция
// void nmppsRemap_8u(nm8u* pSrcVec, nm8u* pDstVec, nm32s* pRemapTable, int nSrcVecSize, int nDstVecSize,
// void* pTmpBuf1, void* pTmpBuf2);
// выполняет следующие действия:
nmppsConvert_8u((nm8u*) pSrcVec, (nm32u*)pTmpBuf1,nSrcVecSize);
nmppsRemap_32u((nm32u*)pTmpBuf1,(nm32u*)pTmpBuf2,RemapTable,DstVecSize);
nmppsConvert_32s((nm32s*)pTmpBuf2,(nm8s*) pDstVec, DstVecSize);
```

## Заметки

Возможность использования inplace параметров определяется исходя из последовательности процессов чтения/записи:

the folowing sequence of actions :

- pSrcVec => pTmpBuf1 - inplace запрещен;

- $pTmpBuf1 \Rightarrow pTmpBuf2$  - inplace запрещен;
- $pTmpBuf2 \Rightarrow pDstVec$  - inplace разрешен;



## 6.274 nmppSplitTmp

Расцепляет массив на два, группируя по четным и нечетным элементам

### Функции

- void nmppsSplitTmp\_8s (const nm8s \*src, nm8s \*dst1, nm8s \*dst2, int size, nm8s \*tmpSizeOfDst)

### 6.274.1 Подробное описание

Расцепляет массив на два, группируя по четным и нечетным элементам

#### Аргументы

|     |              |                                                                                                                  |
|-----|--------------|------------------------------------------------------------------------------------------------------------------|
| in  | src          | Входной массив                                                                                                   |
| out | dst1         | Выходной массив размера size/2                                                                                   |
| out | dst2         | Выходной массив размера size/2                                                                                   |
| in  | size         | Размер исходного массива в элементах. Кратность параметра size должна соответствовать двум длинным 64-р. словам. |
| in  | tmpSizeOfDst | Временный массив размера size/2                                                                                  |

#### Возвращает

Details Максимальная производительность достигается при размещении входных, выходных и временных массивов в разных банках памяти. Массивы dst1 и dst2 могут находиться в одном банке. Макс производительность на 64-р. слово результата = 2.1 такта (при size=10240 байт) и 2.5 такта (при size=4096 байт)

## 6.275 nmppsSplit

Расщепляет массив на два массива, группируя по четным и нечетным элементам

### Функции

- void nmppsSplit\_8s (const nm8s \*src, nm8s \*dst1, nm8s \*dst2, int size)
- void nmppsSplit\_16s (const nm16s \*src, nm16s \*dst1, nm16s \*dst2, int size)
- void nmppsSplit\_32s (const nm32s \*src, nm32s \*dst1, nm32s \*dst2, int size)

### 6.275.1 Подробное описание

Расщепляет массив на два массива, группируя по четным и нечетным элементам

#### Аргументы

|     |      |                                                                                                                  |
|-----|------|------------------------------------------------------------------------------------------------------------------|
| in  | src  | Входной массив                                                                                                   |
| out | dst1 | Выходной массив размера size/2                                                                                   |
| out | dst2 | Выходной массив размера size/2                                                                                   |
| in  | size | Размер исходного массива в элементах. Кратность параметра size должна соответствовать двум длинным 64-р. словам. |

#### Возвращает

Details Максимальная производительность достигается при размещении входных, выходных массивов в разных банках памяти. Массивы dst1 и dst2 могут находиться в одном банке. Макс производительность на 64-р. слово результата = 2.14 такта (при size=10240 байт) и 2.6 такта (при size=4096 байт)

## 6.276 nmppMerge

Собирает массив из двух, чередуя элементы из каждого. Функция обратная nmppsSplit.

### Функции

- void nmppsMerge\_8s (const nm8s \*src0, const nm8s \*src1, nm8s \*dst, int sizeSrc)
- void nmppsMerge\_16s (const nm16s \*src0, const nm16s \*src1, nm16s \*dst, int sizeSrc)
- void nmppsMerge\_32s (const nm32s \*src0, const nm32s \*src1, nm32s \*dst, int sizeSrc)

### 6.276.1 Подробное описание

Собирает массив из двух, чередуя элементы из каждого. Функция обратная nmppsSplit.

#### Аргументы

|     |         |                                                                                                       |
|-----|---------|-------------------------------------------------------------------------------------------------------|
| in  | src0    | Входной массив размера sizeSrc                                                                        |
| in  | src1    | Входной массив размера sizeSrc                                                                        |
| out | dst     | Выходной массив размера 2*sizeSrc                                                                     |
| in  | sizeSrc | Размер выходного массива в элементах. Кратность параметра sizeSrc должна соответствовать 64-р. слову. |

Возвращает

#### Details

## 6.277 nmppsSplit\_32fcr

Расщепляет массив на два, группируя по четным и нечетным элементам

### Функции

- void nmppsSplit\_32fcr (const nm32fcr \*pSrcVec, nm32fcr \*pDstVec1, nm32fcr \*pDstVec, int size↵ Src)

### 6.277.1 Подробное описание

Расщепляет массив на два, группируя по четным и нечетным элементам

#### Аргументы

|     |          |                                                           |
|-----|----------|-----------------------------------------------------------|
| in  | pSrcVec  | Входной массив                                            |
| out | pDstVec1 | Выходной массив размера size/2                            |
| out | pDstVec2 | Выходной массив размера size/2                            |
| in  | sizeSrc  | Размер исходного массива в элементах (должен быть четным) |

#### Возвращает

Details Максимальная производительность достигается при размещении входных, выходных массивов в разных банках памяти. Массивы dst1 и dst2 могут находиться в одном банке. Макс производительность на 64-р. слово результата = 1 такт

## 6.278 nmppsDecimate

Делает выборку элементов из массива с некоторым шагом

### Функции

- void nmppsDecimate\_16s (nm16s \*pSrc, int startPos, int step, nm16s \*pDst, int nSizeDst)
- void nmppsDecimate\_32s (nm32s \*pSrc, int startPos, int step, nm32s \*pDst, int nSizeDst)

### 6.278.1 Подробное описание

Делает выборку элементов из массива с некоторым шагом

#### Аргументы

|     |          |                                                                                                                   |
|-----|----------|-------------------------------------------------------------------------------------------------------------------|
| in  | pSrc     | Входной массив                                                                                                    |
| in  | startPos | Положение элемента в 64-р. слове                                                                                  |
| out | step     | Шаг выборки. Кратность параметра step должна соответствовать длинному 64-р. слову.                                |
| out | pDst     | Выходной массив                                                                                                   |
| in  | nSizeDst | Размер результирующего массива в элементах. Кратность параметра size должна соответствовать длинному 64-р. слову. |

Возвращает

Details Максимальная производительность достигается при размещении входных, выходных массивов в разных банках памяти.

## 6.279 Целочисленные функции

### Группы

- [Векторные функции](#)
- [Матричные функции](#)
- [Функции обработки сигналов](#)
- [Функции обработки изображений](#)
- [Скалярные функции](#)
- [Базовые регистровые функции библиотеки](#)

### 6.279.1 Подробное описание

## 6.280 Функции с плавающей точкой

### Группы

- [Векторные функции](#)
- [Матричные функции](#)
- [Функции обработки сигналов](#)
- [nmblas](#)

### 6.280.1 Подробное описание

## 6.281 Типы данных

### Группы

- [Типы векторных данных](#)
- [Типы скалярных данных](#)

#### 6.281.1 Подробное описание



## 6.282 Векторные функции

### Группы

- [Функции поддержки](#)
- [Инициализация и копирование](#)
- [Арифметические операции](#)
- [Логические и бинарные операции](#)
- [Операции сравнения](#)
- [Переупорядочивание и сортировка](#)

### 6.282.1 Подробное описание

## 6.283 Векторные функции

### Группы

- [Инициализация и копирование](#)
- [Арифметические операции](#)
- [Операции сравнения](#)
- [Элементарные математические функции](#)

### 6.283.1 Подробное описание

## 6.284 Матричные функции

### Группы

- [Инициализация и копирование](#)
- [Функции поддержки](#)
- [Векторно-матричные операции](#)

### 6.284.1 Подробное описание

## 6.285 Матричные функции

### Группы

- [Обращение матрицы](#)

#### 6.285.1 Подробное описание

## 6.286 Функции обработки сигналов

### Группы

- [Свертка](#)
- [Масочная фильтрация](#)
- [Изменение размеров](#)
- [Быстрое преобразование Фурье](#)

### 6.286.1 Подробное описание

## 6.287 Функции обработки сигналов

### Группы

- [Быстрое преобразование Фурье](#)

#### 6.287.1 Подробное описание

## 6.288 Функции обработки изображений

### Группы

- [Floodfill](#)

Исполняет разделение бинарной картинки на односвязные области. Пример вызова: `no=VL_↵ FloodFill32b(pSrcImage, Tetr,Image, pTmpBuff, nSrcWidth, nSrcHeight);`.

- [Переупорядочивание изображений](#)
- [Арифметические действия](#)
- [Масочная фильтрация](#)
- [Инициализация и копирование](#)
- [Функции поддержки](#)
- [Функции графического вывода текста](#)

### 6.288.1 Подробное описание

## 6.289 Скалярные функции

### Группы

- [Инициализация](#)
- [Integer operations](#)
- [Fix-point 64](#)
- [Fix-point 32](#)
- [Арифметические операции](#)
- [Функции деинтерлейсинга](#)

### 6.289.1 Подробное описание

#### 6.289.1.1 Введение

Назначением данной библиотеки является предоставление базовых операций по работе со скалярными данными для процессора NM6403, NM6404, NM6405.

В состав библиотеки входят арифметические, тригонометрические функции, функции для работы с данными в формате с фиксированной точкой.

Библиотека предназначена для быстрой разработки эффективных пользовательских программ на языке высокого уровня(C++).

Назначением данной библиотеки является предоставление базовых операций обработки изображений для процессора NM6403, NM6404, NM6405. В состав библиотеки входят функции двумерной фильтрации, арифметические действия и цветовые преобразования. Библиотека предназначена для быстрой разработки эффективных пользовательских программ на языке высокого уровня с использованием преимуществ архитектуры данного процессора.

Функции библиотеки имеют C++ интерфейс. Большинство функций библиотеки реализованы на языке ассемблера с использованием векторных инструкций и оптимизированы под архитектуру процессора NM6403.

Для удобства разработки прикладных программ библиотека содержит аналогичные реализации функций для процессоров серии x86, выполненных на языке C++. Данные реализации позволяют выполнять написанные с использованием данной библиотеки прикладные программы на персональном компьютере.

Назначением данной библиотеки является предоставление базовых операций по обработке матриц для процессорах NM6403, NM6404, NM6405. В состав библиотеки входят арифметические операции над матрицами. Библиотека предназначена для быстрой разработки эффективных пользовательских программ как на языке высокого уровня(C++).

Функции библиотеки имеют C++ интерфейс. Большинство функций библиотеки реализованы на языке ассемблера с использованием векторных инструкций и оптимизированы под архитектуру процессора NM6403.

Для удобства разработки прикладных программ библиотека содержит аналогичные реализации функций для процессоров серии x86, выполненных на языке C++. Данные реализации позволяют выполнять написанные с использованием данной библиотеки прикладные программы на персональном компьютере.

Назначением данной библиотеки является предоставление базовых функций по обработке сигналов для процессоров NM6403,NM6404,NM6405. В состав библиотеки входят функции одномерной КИХ



фильтрации, нелинейной фильтрации, передискретизации. Библиотека предназначена для быстрой разработки эффективных пользовательских программ как на языке высокого уровня(C++).

Функции библиотеки имеют C++ интерфейс. Большинство функций библиотеки реализованы на языке ассемблера с использованием векторных инструкций и оптимизированы под архитектуру процессоров NM6403.

Для удобства разработки прикладных программ библиотека содержит аналогичные реализации функций для процессоров серии x86, выполненных на языке C++. Данные реализации позволяют выполнять написанные с использованием данной библиотеки прикладные программы на персональном компьютере.

Назначением данной библиотеки является предоставление базовых операций по обработке одномерных массивов (векторов) для процессоров NM6405, NM6406, систем на кристалле с ядром NMC.

В состав библиотеки входят логические и арифметические функции, операции сравнения, инициализации, копирования, преобразования разрядностей и т.п. Библиотека предназначена для быстрой разработки эффективных пользовательских программ как на языке высокого уровня(C++), так и на языке ассемблера с помощью прилагаемой библиотеки ядра низкоуровневых функций. Функции библиотеки имеют C++ интерфейс.

Большинство функций библиотеки реализованы на языке ассемблера с использованием векторных инструкций и оптимизированы под архитектуру процессоров NMC. Для удобства разработки прикладных программ библиотека содержит аналогичные реализации функций для процессоров серии x86, выполненных на языке C++. Данные реализации позволяют выполнять написанные с использованием данной библиотеки прикладные программы на персональном компьютере.

Функции векторного ядра библиотеки

Функции различных библиотек: nmplv, nmpls, nmpli, nmplm и др. , имеющие C++ интерфейсы, в своей реализации используют вызовы функций ядра. Функции ядра не имеют C++ интерфейса. Их вызов возможен только из ассемблера процессора NeuroMatrix. Передача параметров и настройка функций производится через регистры.

Одна и та же функция ядра может использоваться при реализации одной

или нескольких функций библиотеки. Функции ядра также могут быть использованы для реализации пользовательских функций. Использование функций ядра позволяет минимизировать время разработки, уменьшить размер кода и получить максимальную производительность.

## 6.290 Базовые регистровые функции библиотеки

### Группы

- [Элементарные функции](#)
- [функции взвешенного суммирования](#)
- [Целевые функции](#)

### 6.290.1 Подробное описание

6.291 nmblas

## 6.292 контроль переполнения

### Классы

- class `nmmtr< T >`
- class `nmvecpack< T >`
- class `vec< T >`

### Макросы

- `#define GetVec getvec`

### Функции

- `__INLINE__ ostream & operator<< (ostream &s, mtr< unsigned char > &mtr)`

### 6.292.1 Подробное описание

определяет классы, предназначенные для контроля переполнения при реализации библиотеки на РС.

—реализации библиотеки на РС производит контроль переполнения с выдачей сообщения об ошибке пользователю библиотеки. Для этой цели определены шаблонные классы для вектора, матрицы и скалярного числа, позволяющие производить базовые операции над их элементами.

### 6.292.2 Макросы

#### 6.292.2.1 GetVec

```
#define GetVec getvec
```

Класс матриц.

Примеры:

```
int Test[10]={1,125,3,4,5,6,7,8,9,10};
int Res [10];

mtr<int> AA0(3,3);
vec<int> A0(3);
scalar<int> a0(2);

mtr<int> BB0(3,3);
mtr<int> CC0(3,3);
vec<int> C0(3);

BB0.SetDataa(Test);

BB0=AA0;
A0 =AA0[2];
a0 =AA0[2][2];

CC0=AA0+BB0;
CC0=AA0*a0;
C0 =AA0*A0;
CC0=AA0*BB0;
```

См. определение в файле `tmatrix.h` строка 60

### 6.292.3 Функции

#### 6.292.3.1 operator<<()

```
__INLINE__ ostream& operator<< (  
    ostream & s,  
    mtr< unsigned char > & mtr )
```

Класс матриц.

Примеры:

```
int Test[10]={1,125,3,4,5,6,7,8,9,10};  
int Res [10];  
  
mtr<int> AA0(3,3);  
vec<int> A0(3);  
scalar<int> a0(2);  
  
mtr<int> BB0(3,3);  
mtr<int> CC0(3,3);  
vec<int> C0(3);  
  
BB0.SetData( Test);  
  
BB0=AA0;  
A0 =AA0[2];  
a0 =AA0[2][2];  
  
CC0=AA0+BB0;  
CC0=AA0*a0;  
C0 =AA0*A0;  
CC0=AA0*BB0;
```

См. определение в файле nmtlio.h строка 64

## 6.293 Элементарные функции

### Группы

- [Vec\\_0\\_sub\\_data](#)
- [Vec\\_activate\\_data](#)
- [Vec\\_activate\\_data\\_add\\_0](#)
- [Vec\\_activate\\_data\\_xor\\_data](#)
- [Vec\\_activate\\_data\\_add\\_ram](#)
- [Vec\\_affo](#)
- [Vec\\_data](#)
- [Vec\\_data\\_add\\_ram](#)
- [Vec\\_data\\_and\\_ram](#)
- [Vec\\_data\\_or\\_ram](#)
- [Vec\\_data\\_sub\\_ram](#)
- [Vec\\_data\\_xor\\_ram](#)
- [Vec\\_And](#)
- [Vec\\_Mask](#)
- [Vec\\_Or](#)
- [Vec\\_Xor](#)
- [Vec\\_Add](#)
- [Vec\\_Sub](#)
- [Vec\\_not\\_data](#)
- [Vec\\_ram](#)
- [Vec\\_ram\\_sub\\_data](#)
- [Vec\\_vsum\\_activate\\_data\\_0](#)

### 6.293.1 Подробное описание

## 6.294 функции взвешенного суммирования

### Группы

- [Vec\\_ClipMul2D2W8\\_AddVr](#)
- [Vec\\_ClipMulNDNW2\\_AddVr](#)
- [Vec\\_ClipMulNDNW4\\_AddVr](#)
- [Vec\\_ClipMulNDNW8\\_AddVr](#)
- [Vec\\_Mul2D2W1\\_AddVr](#)
- [Vec\\_Mul2D2W2\\_AddVr](#)
- [Vec\\_Mul2D2W4\\_AddVr](#)
- [Vec\\_Mul2D2W8\\_AddVr](#)
- [Vec\\_Mul3D3W2\\_AddVr](#)
- [Vec\\_Mul3D3W8\\_AddVr](#)
- [Vec\\_Mul4D4W2\\_AddVr](#)
- [Vec\\_MulVN\\_AddVN](#)
- [Vec\\_vsum\\_data\\_0](#)
- [Vec\\_vsum\\_data\\_vr](#)
- [Vec\\_vsum\\_shift\\_data\\_0](#)
- [Vec\\_vsum\\_shift\\_data\\_vr](#)
- [Vec\\_vsum\\_shift\\_data\\_afifo](#)

### 6.294.1 Подробное описание

## 6.295 Целевые функции

### Группы

- [Vec\\_Add\\_VV\\_shift](#)
- [Vec\\_data\\_add\\_afifo](#)
- [Vec\\_FilterCoreRow2](#)
- [Vec\\_FilterCoreRow4](#)
- [Vec\\_FilterCoreRow8](#)
- [Vec\\_Abs](#)
- [Vec\\_ClipExt](#)
- [Vec\\_IncNeg](#)
- [Vec\\_SubAbs](#)
- [Vec\\_SubVN\\_Abs](#)
- [Vec\\_Swap](#)
- [Vec\\_MUL\\_2V4toW8\\_shift](#)
- [Vec\\_MUL\\_2V8toW16\\_shift](#)
- [Vec\\_vsum\\_data\\_afifo](#)
- [Vec\\_CompareMinV](#)

Поэлементный поиск минимального

Действие функции эквивалентно следующим псевдоинструкциям:

- [Vec\\_CompareMaxV](#)

Поэлементный поиск максимального

Действие функции эквивалентно следующим псевдоинструкциям:

- [Vec\\_DupValueInVector8](#)

Размножение 8-ми битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

- [Vec\\_DupValueInVector16](#)

Размножение 16-ти битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

- [Vec\\_BuildDiagWeights8](#)

Построение диагональной матрицы весовых коэффициентов (8x8).

- [Vec\\_BuildDiagWeights16](#)

Построение диагональной матрицы весовых коэффициентов (16x16).

- [Vec\\_MaxVal\\_v8nm8s](#)

Поиск максимума в 8 байтах

- [Vec\\_MaxVal\\_v4nm16s](#)

Поиск максимума в 4-х 16р. элементах

- [Vec\\_MaxVal](#)

Поиск максимумов в колонках матрицы SrcMtr1.

- [Vec\\_MinVal\\_v8nm8s](#)

Поиск минимума в 8 байтах

- [Vec\\_MinVal\\_v4nm16s](#)

Поиск минимума в 4-х 16р. элементах

- [Vec\\_MinVal](#)

Поиск минимумов в колонках матрицы SrcMtr1.

- [Vec\\_AccMul1D1W32\\_AddVr](#)

Умножение с накоплением

Действие функции эквивалентно следующим псевдоинструкциям:

### 6.295.1 Подробное описание



## 6.296 Vec\_0\_sub\_data

## Функции

- void vec\_0\_sub\_data (nmreg nb1, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.296.1 Подробное описание

Ядро функции nmppsNeg().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with 0-data;  
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr1                 |
| gr0 | SrcMtr1 stride                               |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6 | указатель на столбец DstMtr                  |
| gr6 | DstMtr stride                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.297 Vec\_activate\_data

### Функции

- void vec\_activate\_data (nmreg f1cr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.297.1 Подробное описание

Ядро функции nmppsCmpLt0\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with activate data;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|      |                                          |
|------|------------------------------------------|
| f1cr | задает функцию активации                 |
| ar0  | указатель на столбец SrcMtr1             |
| gr0  | SrcMtr1 stride                           |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6  | указатель на столбец DstMtr              |
| gr6  | DstMtr stride                            |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.298 Vec\_activate\_data\_add\_0

## Функции

- void vec\_activate\_data\_add\_0 (nmreg flcr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.298.1 Подробное описание

Функция производит арифметическую активацию.

Ядро функции nmppsClipPowC\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with activate data + 0;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|      |                                          |
|------|------------------------------------------|
| flcr | задает функцию активации                 |
| ar0  | указатель на столбец SrcMtr1             |
| gr0  | SrcMtr1 stride                           |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6  | указатель на столбец DstMtr              |
| gr6  | DstMtr stride                            |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.299 Vec\_activate\_data\_xor\_data

### Функции

- void vec\_activate\_data\_xor\_data (nmreg flcr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.299.1 Подробное описание

Функция позволяет вычислить приближенное значение модуля.

Ядро функции nmppsAbs1().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with activate data xor data;
rep N [ar6++gr6]=aifo;
```

### Аргументы

|      |                                            |
|------|--------------------------------------------|
| flcr | задает функцию активации                   |
| ar0  | указатель на столбец SrcMtr1               |
| gr0  | SrcMtr1 stride                             |
| gr5  | Высота матриц $N = [0,1,2...31,32,33,...]$ |
| ar6  | указатель на столбец DstMtr                |
| gr6  | DstMtr stride                              |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.300 Vec\_activate\_data\_add\_ram

## Функции

- void vec\_activate\_data\_add\_ram (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.300.1 Подробное описание

Функция выполняет арифметическую активацию с прибавлением константы.

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar1]
rep N data = [ar0 ++ gr0] with activate data + ram;
rep N [ar6 ++ gr6] = afifo;
```

## Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| flcr | задает функцию активации                     |
| ar0  | указатель на столбец SrcMtr1                 |
| gr0  | SrcMtr1 stride                               |
| ar1  | указатель на 64р. слово                      |
| gr5  | Высота матриц $N = [0,1,2...31,32,33,...]$   |
| ar6  | указатель на столбец DstMtr                  |
| gr6  | DstMtr stride                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0, ar6.

## 6.301 Vec\_Add\_VV\_shift

### Функции

- void vec\_Add\_VV\_shift (nmreg nb1, nmreg sb, nmreg woper, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.301.1 Подробное описание

Функция служит для суммирования двух массивов со сдвигом результата на 1 бит вправо.

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar4];
rep N data = [ar0++gr0] with data + 0;
rep N data = [ar1++gr1] with data + aifo;
rep N with mask ram,shift aifo,0;
rep N [ar6++gr6] = aifo;
```

### Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на строки                                   |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr1                                 |
| gr0   | SrcMtr1 stride                                               |
| ar1   | указатель на столбец SrcMtr2                                 |
| gr1   | SrcMtr2 stride                                               |
| ar4   | указатель на 64р. слово (маска)                              |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar6,gr2,gr5.

## 6.302 Vec\_afifo

### Функции

- void vec\_afifo (nmreg ar0, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.302.1 Подробное описание

Функция служит для заполнения массива константой.

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 1 data=[ar0] with data;  
rep 1*N [ar6++gr6]=afifo with afifo;
```

### Аргументы

|     |                                            |
|-----|--------------------------------------------|
| ar0 | указатель на 64р. слово                    |
| gr5 | Высота матриц $N = [0,1,2...31,32,33,...]$ |
| ar6 | указатель на столбец DstMtr                |
| gr6 | DstMtr stride                              |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.303 Vec\_data

### Функции

- void vec\_data (nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

#### 6.303.1 Подробное описание

Ядро функции nmppsCopy\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;  
rep N [ar6++gr6]=aifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.



## 6.304 Vec\_data\_add\_afifo

## Функции

- void vec\_data\_add\_afifo (nmreg nb1, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6)

## 6.304.1 Подробное описание

Ядро функции nmppsSum().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 1 data=vfalse;
.repeat N;
rep 1 data=[ar0++gr0] with data + afifo; (rep1 N times)
.endrepeat;
rep 1 [ar6]=afifo;
```

## Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr                  |
| gr0 | SrcMtr stride                                |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6 | указатель на 64р. слово                      |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0.

## 6.305 Vec\_data\_add\_ram

### Функции

- void vec\_data\_add\_ram (nmreg nb1, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.305.1 Подробное описание

Ядро функции nmppsAddC().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1];
rep N data=[ar0++gr0] with data+ram;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr1                 |
| gr0 | SrcMtr stride                                |
| ar1 | указатель на 64р. слово-константу            |
| gr5 | Высота матриц $N = [0,1,2...31,32,33,...]$   |
| ar6 | указатель на столбец DstMtr                  |
| gr6 | DstMtr stride                                |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.306 Vec\_data\_and\_ram

## Функции

- void vec\_data\_and\_ram (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.306.1 Подробное описание

Ядро функции nmppsAndC\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1];
rep N data=[ar0++gr0] with data and ram;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на 64р. слово-константу        |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.307 Vec\_data\_or\_ram

### Функции

- void vec\_data\_or\_ram (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.307.1 Подробное описание

Ядро функции nmppsOrC\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1];
rep N data=[ar0++gr0] with data or ram;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на 64-е слово-константу        |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.308 Vec\_data\_sub\_ram

## Функции

- void vec\_data\_sub\_ram (nmreg nb1, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.308.1 Подробное описание

Ядро функции nmppsSubC().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1];
rep N data=[ar0++gr0] with data-ram;
rep N [ar6++gr6]=aiffo;
```

## Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr                  |
| gr0 | SrcMtr stride                                |
| ar1 | указатель на 64р. слово-константу            |
| gr5 | Высота матриц $N = [0,1,2...31,32,33,...]$   |
| ar6 | указатель на столбец DstMtr                  |
| gr6 | DstMtr stride                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.309 Vec\_data\_xor\_ram

### Функции

- void vec\_data\_xor\_ram (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.309.1 Подробное описание

Ядро функции nmppsXorC\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar1];
rep N data = [ar0++gr0] with data xor ram;
rep N [ar6++gr6] = afifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на 64р. слово-константу        |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.310 Vec\_FilterCoreRow2

### Функции

- void vec\_FilterCoreRow2 (nmreg ar0, nmreg ar4, nmreg ar6, nmreg gr1, nmreg gr4, nmreg gr6)

#### 6.310.1 Подробное описание

## 6.311 Vec\_FilterCoreRow4

### Функции

- void vec\_FilterCoreRow4 (nmreg ar0, nmreg ar4, nmreg ar6, nmreg gr1, nmreg gr4, nmreg gr6)

#### 6.311.1 Подробное описание



## 6.312 Vec\_FilterCoreRow8

### Функции

- void vec\_FilterCoreRow8 (nmreg ar0, nmreg ar4, nmreg ar6, nmreg gr1, nmreg gr4, nmreg gr6)

#### 6.312.1 Подробное описание

## 6.313 Vec\_And

### Функции

- void vec\_And (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.313.1 Подробное описание

Ядро функции nmppsAnd\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;
rep N data=[ar1++gr1] with data and aifo;
rep N [ar6++gr6]=aifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на столбец SrcMtr2             |
| gr1 | SrcMtr2 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на результирующий столбец      |
| gr6 | межстрочный шаг для ar6                  |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.314 Vec\_Mask

### Функции

- void vec\_Mask (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar2, nmreg gr2, nmreg gr5, nmreg ar6, nmreg gr6)

#### 6.314.1 Подробное описание

Ядро функции nmppsMaskV\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar0++gr0];
rep N data=[ar1++gr1] with data;
rep N data=[ar2++gr2] with mask data,ram,afifo;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на столбец SrcMtr2             |
| gr1 | SrcMtr2 stride                           |
| ar2 | указатель на столбец SrcMtr2 (маска)     |
| gr2 | SrcMtr3 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на результирующий столбец      |
| gr6 | межстрочный шаг для ar6                  |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar6.

## 6.315 Vec\_Or

### Функции

- void vec\_Or (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.315.1 Подробное описание

Ядро функции nmppsOr\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;
rep N data=[ar1++gr1] with data or aifo;
rep N [ar6++gr6]=aifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на столбец SrcMtr2             |
| gr1 | SrcMtr2 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на результирующий столбец      |
| gr6 | межстрочный шаг для ar6                  |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.316 Vec\_Xor

## Функции

- void vec\_Xor (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.316.1 Подробное описание

Ядро функции nmppsXor\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;
rep N data=[ar1++gr1] with data xor affo;
rep N [ar6++gr6]=affo;
```

## Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на столбец SrcMtr2             |
| gr1 | SrcMtr2 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на результирующий столбец      |
| gr6 | межстрочный шаг для ar6                  |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.317 Vec\_Abs

### Функции

- void vec\_Abs (nmreg nb1, nmreg sb, nmreg flcr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.317.1 Подробное описание

Ядро функции nmppsAbs().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar0++gr0] with activate data;
rep N          with vsum afifo,ram,ram;
rep N [ar6++gr6] = afifo;
```

### Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| sb   | задает разбиение на 8 строк                  |
| flcr | задает функцию активации                     |
| ar0  | указатель на столбец SrcMtr                  |
| gr0  | SrcMtr stride                                |
| gr5  | Высота матриц $N = [0,1,2...31,32,33,...]$   |
| ar6  | указатель на результирующий столбец          |
| gr6  | межстрочный шаг для ar6                      |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.318 Vec\_Add

## Функции

- void vec\_Add (nmreg nb1, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.318.1 Подробное описание

Ядро функции nmppsAdd().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;
rep N data=[ar1++gr1] with data + aifo;
rep N [ar6++gr6]=aifo;
```

## Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr1                 |
| gr0 | SrcMtr1 stride                               |
| ar1 | указатель на столбец SrcMtr2                 |
| gr1 | SrcMtr2 stride                               |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6 | указатель на результирующий столбец          |
| gr6 | межстрочный шаг для ar6                      |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.319 Vec\_ClipExt

### Функции

- `void vec_ClipExt (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg ar2, nmreg ar3, nmreg gr5, nmreg ar6, nmreg gr6)`

### 6.319.1 Подробное описание

Ядро функции `nmppsClipCC_()`.

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar1];
rep N data = [ar0++gr0] with data-ram;
rep N data = [ar2]          with activate afifo+data;
rep N data = [ar3]          with activate afifo-data;
rep N [ar6++gr6] = afifo;
```

### Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| flcr | задает функцию активации                     |
| ar0  | указатель на столбец SrcMtr                  |
| gr0  | SrcMtr stride                                |
| ar1  | указатель на 64р. слово-константу            |
| ar2  | указатель на 64р. слово-константу            |
| ar3  | указатель на 64р. слово-константу            |
| gr5  | Высота матриц $N = [0,1,2...31,32,33,...]$   |
| ar6  | указатель на результирующий столбец          |
| gr6  | межстрочный шаг для ar6                      |

Restrictions:

При выходе из функции изменяется содержимое регистров: `ar0,ar6`.

`nmppsClipCC_`



## 6.320 Vec\_ClipMul2D2W8\_AddVr

## Функции

- void vec\_ClipMul2D2W8\_AddVr (nmreg nb1, nmreg sb, nmreg flcr, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.320.1 Подробное описание

Взвешенное умножение двух массивов с накоплением и активацией

Ядро функции nmppsClipPowC\_RShift\_Convert\_AddC\_().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 8 wfifo=[ar4++],ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,vr;
rep 8 wfifo=[ar4++],ftw,wtw;
rep N data =[ar1++gr1] with vsum ,activate data,afifo;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|      |                                                |
|------|------------------------------------------------|
| nb1  | задает разбиение на колонки                    |
| sb   | задает разбиение на 8 строк                    |
| flcr | задает функцию активации                       |
| vr   | константа для суммирования                     |
| ar0  | SrcMtr1                                        |
| gr0  | SrcMtr1 stride                                 |
| ar1  | SrcMtr2                                        |
| gr1  | SrcMtr2 stride                                 |
| ar4  | 2 матрицы весовых коэффициентов по 8 64р. слов |
| gr4  | дублирует nb1                                  |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]       |
| ar6  | указатель на результирующий столбец            |
| gr6  | межстрочный шаг для ar6                        |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6,gr7.

## 6.321 Vec\_ClipMulNDNW2\_AddVr

### Функции

- void vec\_ClipMulNDNW2\_AddVr (nmreg nb1, nmreg sb, nmreg flcr, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.321.1 Подробное описание

Взвешенное умножение нескольких массивов с накоплением и активацией

Ядро функции SIG\_Filter().

Действие функции эквивалентно следующим псевдоинструкциям:

```
ar2=ar0;

gr2=[ar1++];
ar0=ar2+gr2;
rep 2 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,vr;

.repeat K-1;
gr2=[ar1++];
ar0=ar2+gr2;
rep 2 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,afifo;
.endrepeat;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|      |                                                                               |
|------|-------------------------------------------------------------------------------|
| nb1  | задает разбиение на колонки                                                   |
| sb   | задает разбиение на 2 строки (sb=2)                                           |
| flcr | задает функцию активации                                                      |
| vr   | константа для суммирования                                                    |
| ar0  | задает базовый адрес для входных массивов (как правило адрес первого массива) |
| gr0  | шаг чтения входного массива stride for input arrays                           |
| ar1  | массив адресных смещений входных массивов относительно ar0                    |
| gr1  | количество массив - K                                                         |
| ar4  | массив из K матриц весовых коэффициентов по 2 64р. слов                       |
| gr4  | дублирует nb1                                                                 |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]                                      |
| ar6  | указатель на результирующий столбец                                           |
| gr6  | межстрочный шаг для ar6                                                       |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,gr1,ar2,gr2,ar3,gr3,ar4,ar6,gr7.

## 6.322 Vec\_ClipMulNDNW4\_AddVr

## Функции

- void vec\_ClipMulNDNW4\_AddVr (nmreg nb1, nmreg sb, nmreg flcr, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.322.1 Подробное описание

Взвешенное умножение нескольких массивов с накоплением и активацией

Ядро функции SIG\_Filter().

Действие функции эквивалентно следующим псевдоинструкциям:

```
ar2=ar0;

gr2=[ar1++];
ar0=ar2+gr2;
rep 4 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,vr;

.repeat K-1;
gr2=[ar1++];
ar0=ar2+gr2;
rep 4 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,afifo;
.endrepeat;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|      |                                                                               |
|------|-------------------------------------------------------------------------------|
| nb1  | задает разбиение на колонки                                                   |
| sb   | задает разбиение на 4 строки (sb=20002h)                                      |
| flcr | задает функцию активации                                                      |
| vr   | константа для суммирования                                                    |
| ar0  | задает базовый адрес для входных массивов (как правило адрес первого массива) |
| gr0  | stride for input arrays                                                       |
| ar1  | массив адресных смещений входных массивов относительно ar0                    |
| gr1  | количество массив - K                                                         |
| ar4  | массив из K матриц весовых коэффициентов по 4 64р. слов                       |
| gr4  | дублирует nb1                                                                 |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]                                      |
| ar6  | указатель на результирующий столбец                                           |
| gr6  | межстрочный шаг для ar6                                                       |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,gr1,ar2,gr2,ar3,gr3,ar4,ar6,gr7.

## 6.323 Vec\_ClipMulNDNW8\_AddVr

### Функции

- void vec\_ClipMulNDNW8\_AddVr (nmreg nb1, nmreg sb, nmreg flcr, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.323.1 Подробное описание

Взвешенное умножение нескольких массивов с накоплением и активацией

Ядро функции SIG\_Filter().

Действие функции эквивалентно следующим псевдоинструкциям:

```
ar2=ar0;

gr2=[ar1++];
ar0=ar2+gr2;
rep 8 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,vr;

.repeat K-1;
gr2=[ar1++];
ar0=ar2+gr2;
rep 8 wfifo=[ar4++].ftw,wtw;
rep N data =[ar0++gr0] with vsum ,activate data,affo;
.endrepeat;
rep N [ar6++gr6]=affo;
```

### Аргументы

|      |                                                                                                                     |
|------|---------------------------------------------------------------------------------------------------------------------|
| nb1  | задает разбиение на колонки                                                                                         |
| sb   | задает разбиение на 8 строки (sb=2020202h)                                                                          |
| flcr | задает функцию активации                                                                                            |
| vr   | константа для суммирования                                                                                          |
| ar0  | \ задает базовый адрес для входных массивов (как правило адрес первого массива) \ set base address for input arrays |
| gr0  | stride for input arrays                                                                                             |
| ar1  | массив адресных смещений входных массивов относительно ar0                                                          |
| gr1  | \ количество массив - K \ number of arrays - K                                                                      |
| ar4  | массив из K матриц весовых коэффициентов по 8 64р. слов                                                             |
| gr4  | дублирует nb1                                                                                                       |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]                                                                            |
| ar6  | указатель на результирующий столбец                                                                                 |
| gr6  | межстрочный шаг для ar6                                                                                             |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,gr1,ar2,gr2,ar3,gr3,ar4,ar6,gr7.

## 6.324 Vec\_IncNeg

### Функции

- void vec\_IncNeg (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

#### 6.324.1 Подробное описание

Увеличивает отрицательные числа на 1.

Применяется в nmppsDivC().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram = [ar0++gr0] with activate data;
rep N          with ram - aifo;
rep N [ar6++gr6] = aifo;
```

### Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| flcr | задает функцию активации                     |
| ar0  | указатель на столбец SrcMtr                  |
| gr0  | SrcMtr stride                                |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6  | указатель на столбец DstMtr                  |
| gr6  | DstMtr stride                                |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.325 Vec\_Mul2D2W1\_AddVr

## Функции

- void vec\_Mul2D2W1\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.325.1 Подробное описание

Ядро функции nmppsConvert\_64s(nm64s\* ,nm32s\* ,int ).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 1 wfifo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 1 wfifo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,afifo;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                                                   |
|-----|-------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                      |
| sb  | задает разбиение в 1 строку                                       |
| vr  | константа для суммирования                                        |
| ar0 | указатель на столбец SrcMtr1                                      |
| gr0 | SrcMtr1 stride                                                    |
| ar1 | указатель на столбец SrcMtr2                                      |
| gr1 | SrcMtr2 stride                                                    |
| ar4 | указатель на 2 матрицы весовых коэффициентов по 2 строки в каждой |
| gr4 | дублирует nb1                                                     |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                          |
| ar6 | указатель на столбец DstMtr                                       |
| gr6 | DstMtr stride                                                     |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6,gr7.

## 6.326 Vec\_Mul2D2W2\_AddVr

## Функции

- void vec\_Mul2D2W2\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.326.1 Подробное описание

Ядро функции nmppsConvert\_32s(nm32s\* ,nm16s\* ,int ).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 2 wffo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 2 wffo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,affo;
rep N [ar6++gr6]=affo;
```

## Аргументы

|     |                                                                   |
|-----|-------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                      |
| sb  | задает разбиение на 2 строки                                      |
| vr  | константа для суммирования                                        |
| ar0 | указатель на столбец SrcMtr1                                      |
| gr0 | SrcMtr1 stride                                                    |
| ar1 | указатель на столбец SrcMtr2                                      |
| gr1 | SrcMtr2 stride                                                    |
| ar4 | указатель на 2 матрицы весовых коэффициентов по 2 строки в каждой |
| gr4 | дублирует nb1                                                     |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                          |
| ar6 | указатель на столбец DstMtr                                       |
| gr6 | DstMtr stride                                                     |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6,gr7.

## 6.327 Vec\_Mul2D2W4\_AddVr

## Функции

- void vec\_Mul2D2W4\_AddVr (nmreg nb1, nmreg sb, nmreg flcr, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.327.1 Подробное описание

Ядро функции nmppsConvert\_16s(nm16s\* pSrcVec, nm8s\* pDstVec, int nSize).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 4 wffo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 4 wffo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,afifo;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|      |                                                                   |
|------|-------------------------------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw)                      |
| sb   | задает разбиение на 4 строки                                      |
| flcr | задает функцию активации                                          |
| vr   | константа для суммирования                                        |
| ar0  | указатель на столбец SrcMtr1                                      |
| gr0  | SrcMtr1 stride                                                    |
| ar1  | указатель на столбец SrcMtr2                                      |
| gr1  | SrcMtr2 stride                                                    |
| ar4  | указатель на 2 матрицы весовых коэффициентов по 4 строки в каждой |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]                          |
| ar6  | указатель на столбец DstMtr                                       |
| gr6  | DstMtr stride                                                     |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6,gr7.



## 6.328 Vec\_Mul2D2W8\_AddVr

## Функции

- void vec\_Mul2D2W8\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.328.1 Подробное описание

Применяется в MTR\_Cоруua().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 8 wffo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 8 wffo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,affo;
rep N [ar6++gr6]=affo;
```

## Аргументы

|     |                                                                  |
|-----|------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                     |
| sb  | задает разбиение на 8 строк                                      |
| vr  | константа для суммирования                                       |
| ar0 | указатель на столбец SrcMtr1                                     |
| gr0 | SrcMtr1 stride                                                   |
| ar1 | указатель на столбец SrcMtr2                                     |
| gr1 | SrcMtr2 stride                                                   |
| ar4 | указатель на 2 матрицы весовых коэффициентов по 8 строк в каждой |
| gr4 | дублирует nb1                                                    |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                         |
| ar6 | указатель на столбец DstMtr                                      |
| gr6 | DstMtr stride                                                    |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6,gr7.

## 6.329 Vec\_Mul3D3W2\_AddVr

### Функции

- void vec\_Mul3D3W2\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar2, nmreg gr2, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.329.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,afifo;
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar2++gr2] with vsum ,data,afifo;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|     |                                                                   |
|-----|-------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                      |
| sb  | задает разбиение на 2 строки                                      |
| vr  | константа для суммирования                                        |
| ar0 | указатель на столбец SrcMtr1                                      |
| gr0 | SrcMtr1 stride                                                    |
| ar1 | указатель на столбец SrcMtr2                                      |
| gr1 | SrcMtr2 stride                                                    |
| ar2 | указатель на столбец SrcMtr3                                      |
| gr2 | SrcMtr3 stride                                                    |
| ar4 | указатель на 3 матрицы весовых коэффициентов по 2 строки в каждой |
| gr4 | дублирует nb1                                                     |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                          |
| ar6 | указатель на столбец DstMtr                                       |
| gr6 | DstMtr stride                                                     |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar6,gr7.

## 6.330 Vec\_Mul3D3W8\_AddVr

## Функции

- void vec\_Mul3D3W8\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar2, nmreg gr2, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.330.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 8 wfifo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 8 wfifo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,afifo;
rep 8 wfifo=[ar4++],ftw,wtw;
rep N data=[ar2++gr2] with vsum ,data,afifo;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                                                  |
|-----|------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                     |
| sb  | задает разбиение на 8 строк                                      |
| vr  | константа для суммирования                                       |
| ar0 | указатель на столбец SrcMtr1                                     |
| gr0 | SrcMtr1 stride                                                   |
| ar1 | указатель на столбец SrcMtr2                                     |
| gr1 | SrcMtr2 stride                                                   |
| ar2 | указатель на столбец SrcMtr3                                     |
| gr2 | SrcMtr3 stride                                                   |
| ar4 | указатель на 3 матрицы весовых коэффициентов по 8 строк в каждой |
| gr4 | дублирует nb1                                                    |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                         |
| ar6 | указатель на столбец DstMtr                                      |
| gr6 | DstMtr stride                                                    |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar6,gr7.

## 6.331 Vec\_Mul4D4W2\_AddVr

### Функции

- void vec\_Mul4D4W2\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar2, nmreg gr2, nmreg ar3, nmreg gr3, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.331.1 Подробное описание

Ядро функции nmppsConvert\_32s(nm32s\* pSrcVec, nm8s\* pDstVec, int nSize).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar0++gr0] with vsum ,data,vr;
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar1++gr1] with vsum ,data,afifo;
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar2++gr2] with vsum ,data,afifo;
rep 2 wfifo=[ar4++],ftw,wtw;
rep N data=[ar3++gr3] with vsum ,data,afifo;
rep N [ar6++gr6]=afifo;
```

### Аргументы

|     |                                                                   |
|-----|-------------------------------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw)                      |
| sb  | задает разбиение на 2 строки                                      |
| vr  | константа для суммирования                                        |
| ar0 | указатель на столбец SrcMtr1                                      |
| gr0 | SrcMtr1 stride                                                    |
| ar1 | указатель на столбец SrcMtr2                                      |
| gr1 | SrcMtr2 stride                                                    |
| ar2 | указатель на столбец SrcMtr3                                      |
| gr2 | SrcMtr3 stride                                                    |
| ar3 | указатель на столбец SrcMtr4                                      |
| gr3 | SrcMtr4 stride                                                    |
| ar4 | указатель на 4 матрицы весовых коэффициентов по 2 строки в каждой |
| gr4 | дублирует nb1                                                     |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                          |
| ar6 | указатель на столбец DstMtr                                       |
| gr6 | DstMtr stride                                                     |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar3,ar6,gr7.

## 6.332 Vec\_MulVN\_AddVN

## Функции

- void vec\_MulVN\_AddVN (nmreg nb1, nmreg sb, nmreg flcr, nmreg woper, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.332.1 Подробное описание

Ядро функции MTR\_MulC\_AddVsVc( ).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,data, vr;
rep N data=[ar1++gr1] with affo+data;
rep N [ar6++gr6]=affo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| flcr  | задает функцию активации                                     |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr1                                 |
| gr0   | SrcMtr1 stride                                               |
| ar1   | указатель на столбец SrcMtr2                                 |
| gr1   | SrcMtr2 stride                                               |
| gr5   | Высота матриц $N = [0,1,2...31,32,33,...]$                   |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

### 6.333 Vec\_Sub

#### Функции

- void vec\_Sub (nmreg nb1, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

#### 6.333.1 Подробное описание

Ядро функции nmppsSub().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data;
rep N data=[ar1++gr1] with data - affo;
rep N [ar6++gr6]=affo;
```

#### Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr1                 |
| gr0 | SrcMtr1 stride                               |
| ar1 | указатель на столбец SrcMtr2                 |
| gr1 | SrcMtr2 stride                               |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6 | указатель на столбец DstMtr                  |
| gr6 | DstMtr stride                                |

#### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.334 Vec\_SubAbs

## Функции

- void vec\_SubAbs (nmreg nb1, nmreg sb, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.334.1 Подробное описание

Ядро функции nmppsAbsDiff().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with data
rep N data=[ar1++gr1] with afifo-data;
rep N [ar4],ram =afifo with activate afifo;
rep N with vsum afifo,ram,ram;
rep N [ar6++gr6]= afifo;
```

## Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| sb   | задает разбиение на 8 строк                  |
| flcr | задает функцию активации                     |
| ar0  | указатель на столбец SrcMtr1                 |
| gr0  | SrcMtr1 stride                               |
| ar1  | указатель на столбец SrcMtr2                 |
| gr1  | SrcMtr2 stride                               |
| ar4  | указатель на временный буфер (1 64р. слово)  |
| gr5  | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6  | указатель на столбец DstMtr                  |
| gr6  | DstMtr stride                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.335 Vec\_SubVN\_Abs

## Функции

- void vec\_SubVN\_Abs (nmreg nb1, nmreg sb, nmreg flcr, nmreg woper, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.335.1 Подробное описание

Ядро функции mtr\_SubMV\_Abs().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1]
rep N data=[ar0++gr0] with data-ram;
rep N [ar2++]=afifo with activate afifo;
// ar2,ar5- internal pointers to temporary buffer size of long[32]
rep N data=[ar5++] with vsum afifo,data,data;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| flcr  | задает функцию активации                                     |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr                                  |
| gr0   | SrcMtr stride                                                |
| ar1   | указатель на маску (1 64p. слово)                            |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.



## 6.336 Vec\_Swap

## Функции

- void vec\_Swap (nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.336.1 Подробное описание

Ядро функции mtr\_SubVN\_Abs().

Функция осуществляет два одновременных копирования:

[ar0++gr0] => [ar4++gr4]

[ar1++gr1] => [ar6++gr6]

если ar6=ar0,gr6=gr0, ar4=ar1,gr4=gr1

то выполняется перестановка двух векторов

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar0++gr0];
rep N data=[ar1++gr1] with data;
rep N [ar6++gr6]=afifo with ram;
rep N [ar4++gr4]=afifo;
```

## Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| ar1 | указатель на столбец SrcMtr2             |
| gr1 | SrcMtr2 stride                           |
| ar4 | указатель на столбец DstMtr1             |
| gr4 | DstVec1 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr2             |
| gr6 | DstVec2 stride                           |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar4,ar6.

## 6.337 Vec\_MUL\_2V4toW8\_shift

## Функции

- void vec\_MUL\_2V4toW8\_shift (nmreg nb1, nmreg sb, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar3, nmreg gr4, nmreg ar5, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.337.1 Подробное описание

Ядро функции SIG\_ResizeDown2(nm8u7b\* pSrcVec, nm8u7b\* pDstVec, int nSize).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar3];
rep 8 wifo=[ar5++],ftw,wtw;
rep N data =[ar0++gr0],ftw,wtw with vsum ,data,0;
rep N data =[ar1++gr1]          with vsum ,data,afifo;
rep N          with mask ram,shift afifo,0;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                                          |
|-----|----------------------------------------------------------|
| nb1 | задает разбиение на колонки                              |
| sb  | задает разбиение на 4 строки                             |
| ar0 | указатель на столбец SrcMtr                              |
| gr0 | SrcMtr stride                                            |
| ar1 | указатель на столбец SrcMtr2                             |
| gr1 | SrcMtr2 stride                                           |
| ar3 | указатель на 64р. маску                                  |
| gr4 | дублирует nb1                                            |
| ar5 | указатель на матрицу весовых коэффициентов (16 64р.слов) |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                 |
| ar6 | указатель на столбец DstMtr                              |
| gr6 | DstMtr stride                                            |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar4,ar6,gr2,gr5.

## 6.338 Vec\_MUL\_2V8toW16\_shift

## Функции

- void vec\_MUL\_2V8toW16\_shift (nmreg nb1, nmreg sb, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar3, nmreg gr4, nmreg ar5, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.338.1 Подробное описание

Ядро функции SIG\_ResizeDown2(nm16u15b\* pSrcVec, nm16u15b\* pDstVec, int nSize).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar3];
rep 16 wifo=[ar5++],ftw,wtw;
rep N data=[ar0++gr0],ftw,wtw with vsum ,data,0;
rep N data=[ar1++gr1]          with vsum ,data,afifo;
rep N                               with mask ram,shift afifo,0;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                                          |
|-----|----------------------------------------------------------|
| nb1 | задает разбиение на колонки                              |
| sb  | задает разбиение на 2 строки                             |
| ar0 | указатель на столбец SrcMtr1                             |
| gr0 | SrcMtr1 stride                                           |
| ar1 | указатель на столбец SrcMtr2                             |
| gr1 | SrcMtr2 stride                                           |
| ar3 | указатель на 64р. маску                                  |
| gr4 | дублирует nb1                                            |
| ar5 | указатель на матрицу весовых коэффициентов (16 64р.слов) |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]                 |
| ar6 | указатель на столбец DstMtr                              |
| gr6 | DstMtr stride                                            |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar2,ar4,ar6,gr2,gr5.

## 6.339 Vec\_not\_data

## Функции

- void vec\_not\_data (nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.339.1 Подробное описание

Ядро функции nmppsNot\_ ( ).

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with not data;
rep N [ar6++gr6]=aifo;
```

## Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr0 | SrcMtr1 stride                           |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar6.

## 6.340 Vec\_ram

### Функции

- void vec\_ram (nmreg ar0, nmreg gr5, nmreg ar6, nmreg gr6)

#### 6.340.1 Подробное описание

Ядро функции nmppsSet\_().

Функция служит для заполнения массива константой.

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar0] with data;  
rep N [ar6++gr6]=aifo;
```

### Аргументы

|     |                                          |
|-----|------------------------------------------|
| ar0 | указатель на столбец SrcMtr1             |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6 | указатель на столбец DstMtr              |
| gr6 | DstMtr stride                            |

### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.341 Vec\_ram\_sub\_data

## Функции

- void vec\_ram\_sub\_data (nmreg nb1, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.341.1 Подробное описание

Ядро функции nmppsSubCRev().

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar1];
rep N data=[ar0++gr0] with ram-data;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|     |                                              |
|-----|----------------------------------------------|
| nb1 | задает разбиение на колонки (необходимо wtw) |
| ar0 | указатель на столбец SrcMtr                  |
| gr0 | SrcMtr stride                                |
| gr5 | Высота матриц N = [0,1,2...31,32,33,...]     |
| ar6 | указатель на столбец DstMtr                  |
| gr6 | DstMtr stride                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.342 Vec\_vsum\_activate\_data\_0

## Функции

- void vec\_vsum\_activate\_data\_0 (nmreg nb1, nmreg sb, nmreg flcr, nmreg woper, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.342.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,activate,0;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| flcr  | задает функцию активации                                     |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr1                                 |
| gr0   | SrcMtr1 stride                                               |
| ar1   | указатель на столбец SrcMtr2                                 |
| gr1   | SrcMtr2 stride                                               |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.343 Vec\_vsum\_data\_0

## Функции

- void vec\_vsum\_data\_0 (nmreg nb1, nmreg sb, nmreg woper, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.343.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,data,0;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr                                  |
| gr0   | SrcMtr stride                                                |
| gr5   | Высота матриц $N = [0,1,2...31,32,33,...]$                   |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.



## 6.344 Vec\_vsum\_data\_afifo

## Функции

- void vec\_vsum\_data\_afifo (nmreg nb1, nmreg sb, nmreg woper, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6)

## 6.344.1 Подробное описание

Используется в nmppsSum(nm1\* pSrcVec, void\* pTmpBuf, int nSize)

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 1  data=vfalse;
.repeat N;
rep 1  data=[ar0++gr0] with vsum ,data,afifo;
.endrepeat;
rep 1  [ar6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на строки                                   |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr                                  |
| gr0   | SrcMtr stride                                                |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0.

## 6.345 Vec\_vsum\_data\_vr

## Функции

- void vec\_vsum\_data\_vr (nmreg nb1, nmreg sb, nmreg woper, nmreg vr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.345.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,data,vr;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| flcr  | задает функцию активации                                     |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| vr    | константа для суммирования                                   |
| ar0   | указатель на столбец SrcMtr                                  |
| gr0   | SrcMtr stride                                                |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.346 Vec\_vsum\_shift\_data\_0

## Функции

- void vec\_vsum\_shift\_data\_0 (nmreg nb1, nmreg sb, nmreg woper, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.346.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,shift data,0;  
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr1                                 |
| gr0   | SrcMtr1 stride                                               |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.347 Vec\_vsum\_shift\_data\_vr

## Функции

- void vec\_vsum\_shift\_data\_vr (nmreg nb1, nmreg sb, nmreg woper, nmreg vr, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6, nmreg gr6)

## 6.347.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N data=[ar0++gr0] with vsum ,shift data, vr;
rep N [ar6++gr6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| vr    | константа для суммирования                                   |
| ar0   | указатель на столбец SrcMtr1                                 |
| gr0   | SrcMtr1 stride                                               |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |
| gr6   | DstMtr stride                                                |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar6.

## 6.348 Vec\_vsum\_shift\_data\_afifo

## Функции

- void vec\_vsum\_shift\_data\_afifo (nmreg nb1, nmreg sb, nmreg flcr, nmreg woper, nmreg ar0, nmreg gr0, nmreg gr5, nmreg ar6)

## 6.348.1 Подробное описание

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 1  data=vfalse;
rep 1*N data=[ar0++gr0] with vsum , shift data,afifo; (rep1 N times)
rep 1  [ar6]=afifo;
```

## Аргументы

|       |                                                              |
|-------|--------------------------------------------------------------|
| nb1   | задает разбиение на колонки (необходимо wtw)                 |
| sb    | задает разбиение на 8 строк                                  |
| flcr  | задает функцию активации                                     |
| woper | в рабочей матрице должны быть загружены весовые коэффициенты |
| ar0   | указатель на столбец SrcMtr                                  |
| gr0   | SrcMtr stride                                                |
| gr5   | Высота матриц N = [0,1,2...31,32,33,...]                     |
| ar6   | указатель на столбец DstMtr                                  |

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0.

## 6.349 Vec\_CompareMinV

Поэлементный поиск минимального

Действие функции эквивалентно следующим псевдоинструкциям:

Функции

- void vec\_CompareMinV (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar3, nmreg gr3, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.349.1 Подробное описание

Поэлементный поиск минимального

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar0++gr0];
rep N data=[ar1++gr1] with ram-data;
rep N      with activate aifo;
rep N data=[ar3++gr3] with mask aifo, ram, data;
rep N [ar6++gr6]=aifo;
```

\param nb1

задает разбиение на колонки (необходимо wtw)

Аргументы

|         |                                          |
|---------|------------------------------------------|
| flcr    | задает функцию активации                 |
| ar0     | указатель на столбец SrcMtr1             |
| gr0     | SrcMtr1 stride                           |
| ar1,ar3 | указатель на столбец SrcMtr2             |
| gr1,gr3 | SrcMtr2 stride                           |
| gr5     | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6     | указатель на столбец DstMtr              |
| gr6     | DstMtr stride                            |

Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar3,ar6.

## 6.350 Vec\_CompareMaxV

Поэлементный поиск максимального

Действие функции эквивалентно следующим псевдоинструкциям:

Функции

- void vec\_CompareMaxV (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar1, nmreg gr1, nmreg ar3, nmreg gr3, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.350.1 Подробное описание

Поэлементный поиск максимального

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep N ram=[ar0++gr0];
rep N data=[ar1++gr1] with ram-data;
rep N      with activate aifo;
rep N data=[ar3++gr3] with mask aifo, data, ram;
rep N [ar6++gr6]=aifo;
```

\param nb1

задаёт разбиение на колонки (необходимо wtw)

Аргументы

|         |                                          |
|---------|------------------------------------------|
| flcr    | задаёт функцию активации                 |
| ar0     | указатель на столбец SrcMtr1             |
| gr0     | SrcMtr1 stride                           |
| ar1,ar3 | указатель на столбец SrcMtr2             |
| gr1,gr3 | SrcMtr2 stride                           |
| gr5     | Высота матриц N = [0,1,2...31,32,33,...] |
| ar6     | указатель на столбец DstMtr              |
| gr6     | DstMtr stride                            |

Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,ar1,ar3,ar6.

## 6.351 Vec\_DupValueInVector8

Размножение 8-ми битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

Функции

- void vec\_DupValueInVector8 (nmreg ar1, nmreg gr1)

### 6.351.1 Подробное описание

Размножение 8-ми битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

```
gr1 = gr1 and 0FFh;  
[ar1] = gr1 + (gr1 << 8) + (gr1 << 16) + (gr1 << 24) +  
        (gr1 << 32) + (gr1 << 40) + (gr1 << 48) + (gr1 << 56).  
ar1 += 2;
```

\param ar1

Адрес 64-х битового вектора.

Аргументы

|     |                   |
|-----|-------------------|
| gr1 | Значение (8 бит). |
|-----|-------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: ar1, gr1.



## 6.352 Vec\_DupValueInVector16

Размножение 16-ти битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

Функции

- void vec\_DupValueInVector16 ([nmreg](#) ar1, [nmreg](#) gr1)

### 6.352.1 Подробное описание

Размножение 16-ти битового значения по всему вектору.

Действие функции эквивалентно следующим псевдоинструкциям:

```
gr1 = gr1 and 0FFFFh;
[ar1] = gr1 + (gr1 << 16) + (gr1 << 32) + (gr1 << 48).
ar1 += 2;
```

\param ar1

Адрес 64-х битового вектора.

Аргументы

|     |                   |
|-----|-------------------|
| gr1 | Значение (8 бит). |
|-----|-------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: ar1, gr1.

## 6.353 Vec\_BuildDiagWeights8

Построение диагональной матрицы весовых коэффициентов (8x8).

Функции

- void vec\_BuildDiagWeights8 (nmreg ar1, nmreg gr1)

### 6.353.1 Подробное описание

Построение диагональной матрицы весовых коэффициентов (8x8).

Аргументы

|     |                                                     |
|-----|-----------------------------------------------------|
| ar1 | Адрес 64-х буфера весовых коэффициентов (8x64 бит). |
| gr1 | Значение (8 бит).                                   |

Restrictions:

При выходе из функции изменяется содержимое регистров: ar1, gr1.

## 6.354 Vec\_BuildDiagWeights16

Построение диагональной матрицы весовых коэффициентов (16x16).

Функции

- void vec\_BuildDiagWeights16 (nmreg ar1, nmreg gr1)

### 6.354.1 Подробное описание

Построение диагональной матрицы весовых коэффициентов (16x16).

Аргументы

|     |                                                     |
|-----|-----------------------------------------------------|
| ar1 | Адрес 64-х буфера весовых коэффициентов (4x64 бит). |
| gr1 | Значение (16 бит).                                  |

Restrictions:

При выходе из функции изменяется содержимое регистров: ar1, gr1.

## 6.355 Vec\_MaxVal\_v8nm8s

Поиск максимума в 8 байтах

Функции

- void vec\_MaxVal\_v8nm8s ([nmreg](#) ar0, [nmreg](#) gr7)

### 6.355.1 Подробное описание

Поиск максимума в 8 байтах

Аргументы

|     |                  |
|-----|------------------|
| ar0 | Адрес 64р. слова |
|-----|------------------|

Возвращаемые значения

|     |                    |
|-----|--------------------|
| gr7 | Максимум из 8 байт |
|-----|--------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: gr0, gr1, gr2, gr3, ar5, gr5, gr7.

## 6.356 Vec\_MaxVal\_v4nm16s

Поиск максимума в 4-х 16р. элементах

Функции

- void vec\_MaxVal\_v4nm16s ([nmreg](#) ar0, [nmreg](#) gr7)

### 6.356.1 Подробное описание

Поиск максимума в 4-х 16р. элементах

Аргументы

|     |                  |
|-----|------------------|
| ar0 | Адрес 64р. слова |
|-----|------------------|

Возвращаемые значения

|     |                    |
|-----|--------------------|
| gr7 | Максимум из 8 байт |
|-----|--------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: gr0, gr1, gr2, gr3, ar5, gr5, gr7.

## 6.357 Vec\_MaxVal

Поиск максимумов в колонках матрицы SrcMtr1.

## Функции

- void vec\_MaxVal (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar4, nmreg gr5, nmreg ar6)

## 6.357.1 Подробное описание

Поиск максимумов в колонках матрицы SrcMtr1.

## Аргументы

|      |                                              |
|------|----------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw) |
| flcr | задает функцию активации                     |

\param ar0

указатель на столбец SrcMtr1

## Аргументы

|     |                                                  |
|-----|--------------------------------------------------|
| gr0 | SrcMtr stride                                    |
| ar4 | указатель на временный массив размером nm64s[64] |
| gr5 | Высота матрицы SrcMtr1 N = [32,64,...]           |

## Возвращаемые значения

|     |                                                  |
|-----|--------------------------------------------------|
| ar6 | указатель на 64р. слово результатов (максимумов) |
|-----|--------------------------------------------------|

## Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,gr0,ar4,ar3,ar6,gr7.

## 6.358 Vec\_MinVal\_v8nm8s

Поиск минимума в 8 байтах

Функции

- void vec\_MinVal\_v8nm8s (nmreg ar0, nmreg gr7)

### 6.358.1 Подробное описание

Поиск минимума в 8 байтах

Аргументы

|     |                  |
|-----|------------------|
| ar0 | Адрес 64р. слова |
|-----|------------------|

Возвращаемые значения

|     |                    |
|-----|--------------------|
| gr7 | Максимум из 8 байт |
|-----|--------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: gr0, gr1, gr2, gr3, ar5, gr5, gr7.

## 6.359 Vec\_MinVal\_v4nm16s

Поиск минимума в 4-х 16р. элементах

Функции

- void vec\_MinVal\_v4nm16s ([nmreg](#) ar0, [nmreg](#) gr7)

### 6.359.1 Подробное описание

Поиск минимума в 4-х 16р. элементах

Аргументы

|     |                  |
|-----|------------------|
| ar0 | Адрес 64р. слова |
|-----|------------------|

Возвращаемые значения

|     |                    |
|-----|--------------------|
| gr7 | Максимум из 8 байт |
|-----|--------------------|

Restrictions:

При выходе из функции изменяется содержимое регистров: gr0, gr1, gr2, gr3, ar5, gr5, gr7.



## 6.360 Vec\_MinVal

Поиск минимумов в колонках матрицы SrcMtr1.

### Функции

- void vec\_MinVal (nmreg nb1, nmreg flcr, nmreg ar0, nmreg gr0, nmreg ar4, nmreg gr5, nmreg ar6)

### 6.360.1 Подробное описание

Поиск минимумов в колонках матрицы SrcMtr1.

#### Аргументы

|      |                                                  |
|------|--------------------------------------------------|
| nb1  | задает разбиение на колонки (необходимо wtw)     |
| flcr | задает функцию активации                         |
| ar0  | указатель на столбец SrcMtr1                     |
| gr0  | SrcMtr stride                                    |
| ar4  | указатель на временный массив размером nm64s[64] |
| gr5  | Высота матрицы SrcMtr1 N = [32,64,...]           |

#### Возвращаемые значения

|     |                                                  |
|-----|--------------------------------------------------|
| ar6 | указатель на 64р. слово результатов (максимумов) |
|-----|--------------------------------------------------|

#### Restrictions:

При выходе из функции изменяется содержимое регистров: ar0,gr0,ar4,ar3,ar6,gr7.

## 6.361 Vec\_AccMul1D1W32\_AddVr

Умножение с накоплением

Действие функции эквивалентно следующим псевдоинструкциям:

Функции

- void vec\_AccMul1D1W32\_AddVr (nmreg nb1, nmreg sb, nmreg vr, nmreg ar0, nmreg gr0, nmreg ar4, nmreg gr4, nmreg gr5, nmreg ar6, nmreg gr6)

### 6.361.1 Подробное описание

Умножение с накоплением

Действие функции эквивалентно следующим псевдоинструкциям:

```
rep 32 wfifo=[ar4++],ftw,wtw;
rep 32 wfifo data = [ar0++gr0] with vsum ,data,vr;
with gr5--;
with gr5--;
<Loop>
    rep 32 wfifo=[ar4++],ftw,wtw;
    rep 32 data=[ar0++gr0] with vsum ,data,afifo;
if <>0 goto Loop with gr5--;
rep 32 [ar6++gr6]=afifo;
```

\param nb1

задает разбиение на колонки (необходимо wtw)

Аргументы

|     |                                         |
|-----|-----------------------------------------|
| sb  | задает разбиение на 32 строки           |
| vr  | константа для суммирования              |
| ar0 | указатель на столбец SrcMtr1            |
| gr0 | SrcMtr stride                           |
| ar4 | матрицы весовых коэффициентов           |
| gr4 | дублирует nb1                           |
| gr5 | кол-во итераций умножений с накоплением |

Возвращаемые значения

|     |                                                           |
|-----|-----------------------------------------------------------|
| ar6 | указатель на столбец DstMtr, состоящий из 32 длинных слов |
|-----|-----------------------------------------------------------|

Аргументы

|     |               |
|-----|---------------|
| gr6 | DstMtr stride |
|-----|---------------|

## Глава 7

# Классы

### 7.1 Класс C\_2DSubPixelMinPosition

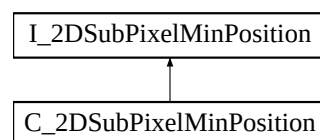
Открытые члены

- virtual void Find (float \*S9, float &dx, float &dy)
- virtual void Release ()

#### 7.1.1 Подробное описание

См. определение в файле isubpixel2dimpl.h строка 7

Граф наследования: C\_2DSubPixelMinPosition:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/isubpixel2dimpl.h

### 7.2 Класс C\_2DTrigSubPixelMinPosition

Открытые члены

- virtual void Find (float \*S9, float &dx, float &dy)
- virtual void Release ()

### Защищенные члены

- float S9Interpolation (float x, float y, float \*S9)

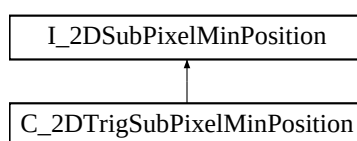
### Защищенные данные

- float Teta [8]

#### 7.2.1 Подробное описание

См. определение в файле isubpixel2dimpl.h строка 20

Граф наследования: C\_2DTrigSubPixelMinPosition:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/isubpixel2dimpl.h

## 7.3 Класс C\_Allocator32

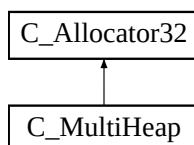
### Открытые члены

- void \* Allocate ()
- int Release ()

#### 7.3.1 Подробное описание

См. определение в файле multiheap.h строка 330

Граф наследования: C\_Allocator32:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/multiheap.h

## 7.4 Шаблон класса C\_BoxImg< T >

### Открытые члены

- C\_BoxImg (C\_MultiHeap &MultiHeap, int Width, int Height, int BorderHeight=0, int FillMode=BOX\_nmppiFILL\_FF)
- void Lock ()
- void Unlock ()
- void Fill (int FillMode)
- C\_BoxImg (T \*Data, int Width, int Height, int BorderHeight=0)
- void Init (T \*Data, int Width, int Height, int BorderHeight=0)
- T \* Addr (int y, int x)
- T \* Allocate (C\_MultiHeap &MultiHeap, int Width, int Height, int BorderHeight=0)
- T \* Allocate (C\_MultiHeap &MultiHeap)
- int Release ()
- \_\_INLINE\_\_ T \* operator[] (int idx)

### Открытые атрибуты

- int nWidth
- int nHeight
- int sizeBox
- int sizeData
- int nBorder
- T \* pBox
- T \* pData
- C\_MultiHeap \* pHeap

#### 7.4.1 Подробное описание

```
template<class T>
class C_BoxImg< T >
```

См. определение в файле multiheap.h строка 717

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/multiheap.h

## 7.5 Шаблон класса C\_BoxVec< T >

### Открытые члены

- C\_BoxVec (T \*Data, int SizeData, int SizeBorder=0)
- C\_BoxVec (C\_MultiHeap &MultiHeap, int SizeData, int Border=0)
- T \* Addr (int idx)
- int Assign (T \*Data, int SizeData, int SizeBorder=0)
- T \* Allocate (C\_MultiHeap &MultiHeap, int SizeData, int Border=0)
- T \* Allocate (C\_MultiHeap &MultiHeap)
- int Release ()

## Открытые атрибуты

- int sizeData
- int sizeBox
- int nBorder
- T \* pBox
- T \* pData
- [C\\_MultiHeap](#) \* pHeap

## 7.5.1 Подробное описание

```
template<class T>
class C_BoxVec< T >
```

См. определение в файле multiheap.h строка 634

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/multiheap.h

## 7.6 Класс C\_Heap

класс - куча

```
#include <multiheap.h>
```

## Открытые члены

- [C\\_Heap](#) (void \*addrHeap, size\_t size32Heap)  
конструктор - создает кучу в указанной памяти
- void [Create](#) (void \*addrHeap, size\_t size32Heap)  
создает кучу в указанной памяти
- int [IsMine](#) (void \*addr)  
устанавливает принадлежность к куче
- size\_t [AllocateMaxAvail](#) ()  
Возвращает объем свободной памяти в пуле в 32-х словах
- int \* [Allocate](#) (size\_t size32Buffer)  
Выделяет буфер в куче
- int [ReleaseBuffer](#) ([S\\_BufferInfo](#) \*pDelBuffer)  
удаляет структуру буфера из списка
- int [Release](#) (void \*p)  
освобождает память по адресу
- void [Lock](#) (void \*p)  
блокирует указатель от удаления через Release.
- void [LockAll](#) ()  
блокирует все указатели от удаления через Release.
- void [UnlockAll](#) ()  
разблокирует все указатели для удаления через Release.
- void [Unlock](#) (void \*p)  
разблокирует все указатели для удаления через Release.
- void [ReleaseAll](#) ()  
удаляет все указатели из кучи
- void [LockHeap](#) ()  
Запрещает операции с кучей
- void [UnlockHeap](#) ()  
Разрешает операции с кучей
- int [Check](#) ()

## Открытые атрибуты

- [S\\_BufferInfo \\* pZeroBuffer](#)  
< указатель на нулевой буфер в списке (с нулевым размером)
- [int \\* pHeapEnd](#)  
< указатель на слово следующее за концом кучи
- [int size32HeapAvail](#)  
< размер общей свободной памяти в куче
- [bool isHeapLocked](#)  
< запрещает операции с кучей
- [int status](#)

## 7.6.1 Подробное описание

класс - куча

См. определение в файле multiheap.h строка 64

## 7.6.2 Методы

## 7.6.2.1 AllocateMaxAvail()

```
size_t32 C_Heap::AllocateMaxAvail ( ) [inline]
```

Возвращает объем свободной памяти в пуле в 32р. словах

Возвращает максимальный размер буфера в 32р. словах, который можно выделить в куче

См. определение в файле multiheap.h строка 115

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/multiheap.h

## 7.7 Шаблон класса C\_Img&lt; T &gt;

## Открытые члены

- [C\\_Img \(int nWidth, int nHeight, int nStride, int nBorder, void \\*\(\\*allocator32\)\(int\)\)](#)
- [C\\_Img \(T \\*pData, int nWidth, int nHeight, int nStride, int nBorder\)](#)
- [void Fill \(T color\)](#)

## Открытые атрибуты

- int m\_nBorder
- T \* m\_pContainer
- T \* m\_pData
- int m\_nWidth
- int m\_nHeight
- int m\_nStride
- int m\_nSize

## 7.7.1 Подробное описание

```
template<class T>
class C_Img< T >
```

См. определение в файле iSupport.h строка 52

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/iSupport.h

## 7.8 Класс C\_MultiHeap

## Открытые члены

- C\_MultiHeap (int Mode=ALLOCATE\_FORWARD)
- int Error ()
- void Mode (int mode, void \*\*legend=0)
- unsigned Rand ()
  - Генератор случайных чисел
- unsigned Rand (unsigned min, unsigned max)
- C\_Heap & operator[] (int idxHeap)
- int CreateHeap (void \*addrHeap, size\_t size32Heap)
  - создает кучу по адресууказанного размера (полный размер со служебными данными)
- void \* Allocate (size\_t size32Buffer)
  - обходит кучи в заднном в AllocateMode порядке и выделяет память заданного размера
- void \* Allocate (size\_t size32Buffer, int nPriorHeap0, int nPriorHeap1=-1, int nPriorHeap2=-1, int nPriorHeap3=-1, int nPriorHeap4=-1, int nPriorHeap5=-1)
  - обходит кучи в заднном порядке и выделяет память заданного размера
- void \* AllocateWith (size\_t size32Buffer, void \*addrInTheSameHeap)
  - выделяет массив в той же куче где и указатель
- int Which (void \*addr)
  - Возвращает номер кучи к которой принадлежит адрес
- void Lock (void \*addr)
- int Unlock (void \*addr)
- int LockAll ()
- int UnlockAll ()
- int Release (void \*addr)
- void ReleaseAll ()
  - Удвляет все назаблокированные указатели из куч
- void LockHeap (int idxHeap)
  - Запрещает операции Allocate и Release с кучей
- void UnlockHeap (int idxHeap)
  - Разрешает операции Allocate и Release с кучей
- int Check ()



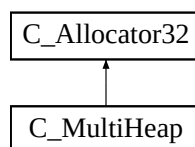
## Открытые атрибуты

- `C_Heap pHeap [MAX_NUM_BANKS]`  
массив куч
- `unsigned numHeaps`  
число проинициализированных куч
- `unsigned numAllocateFails`  
число ошибок выделения куч
- `unsigned AllocateMode`  
порядок обхода куч при поиске свободного места
- `void ** pAllocateLegend`  
история номеров куч использованных в последних 8 Allocate.
- `unsigned idxAllocateLegend`
- `long long allocateHistory`

## 7.8.1 Подробное описание

См. определение в файле `multiheap.h` строка 336

Граф наследования: C\_MultiHeap:



Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/multiheap.h`

## 7.9 Класс C\_PlessyCornerDetector

## Открытые члены

- `virtual void Allocate (int w, int h, int ww)`
- `virtual void DeAllocate ()`
- `virtual void FindCorners (unsigned char *Picture, int w, int h, int ww, float *px, float *py, int &nc)`
- `virtual void Release ()`
- `virtual void SetThreshold (float _threshold)=0`

## Защищенные члены

- `virtual void CountDer (unsigned char *Picture, int w, int h, int ww)`
- `virtual void CountPlessy (int w, int h, int ww)`

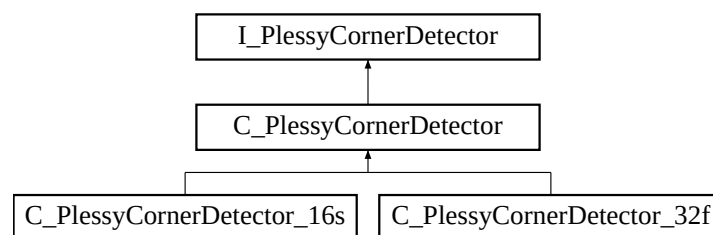
## Защищенные данные

- `I_2DSubPixelMinPosition` \* SubPixelMinPosition
- short \* fxi
- short \* fyi
- short \* Picture16
- float \* SumSxxSyy
- float \* Subxy
- float S9 [9]
- float threshold
- unsigned char \* cres

### 7.9.1 Подробное описание

См. определение в файле `iPlessyDetector.h` строка 8

Граф наследования: `C_PlessyCornerDetector`:



Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmpli/iPlessyDetector.h`

### 7.10 Класс `C_PlessyCornerDetector_16s`

#### Открытые члены

- `virtual void Allocate (int w, int h, int ww)`
- `virtual void DeAllocate ()`
- `virtual void SetThreshold (float _threshold)`

#### Защищенные члены

- `virtual void CountPlessy (int w, int h, int ww)`

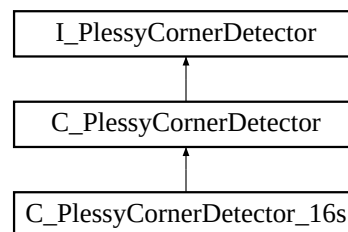
## Защищенные данные

- short \* sxxi
- short \* sxyi
- short \* syyi
- short \* Sxxi
- short \* Sxyi
- short \* Syyi
- short \* Sxxit
- short \* Syyit
- short \* SumSxxSyyi
- short \* MulSxxSyyi
- short \* MulSxySxyi
- short \* Subxyi
- short \* cSubxyi
- short threshold

## 7.10.1 Подробное описание

См. определение в файле iPlessyDetector.h строка 56

Граф наследования: C\_PlessyCornerDetector\_16s:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/iPlessyDetector.h

## 7.11 Класс C\_PlessyCornerDetector\_32f

## Открытые члены

- virtual void Allocate (int w, int h, int ww)
- virtual void DeAllocate ()
- virtual void SetThreshold (float \_threshold)

## Защищенные члены

- virtual void CountPlessy (int w, int h, int ww)
- virtual void CountDer (unsigned char \*Picture, int w, int h, int ww)

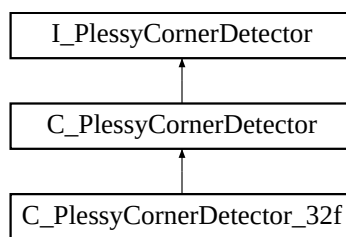
## Защищенные данные

- float \* fx
- float \* fy
- float \* sxx
- float \* sxy
- float \* syy
- float \* Sxx
- float \* Sxy
- float \* Syy
- float \* MulSxxSyy
- float \* MulSxySxy
- float \* cSubxy
- float threshold

### 7.11.1 Подробное описание

См. определение в файле iPlessyDetector.h строка 34

Граф наследования: C\_PlessyCornerDetector\_32f:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/iPlessyDetector.h

## 7.12 Шаблон класса C\_RingBufferRemote< T >

### Открытые члены

- C\_RingBufferRemote (size\_t ringbuffer\_addr, t\_bytecpy push\_memcpy, t\_bytecpy pop\_memcpy)
- size\_t GetHead ()
- size\_t GetTail ()
- void SetHead ()
- void SetTail ()
- bool Init (size\_t ringbuffer\_addr, t\_bytecpy push\_memcpy, t\_bytecpy pop\_memcpy)
- \_\_INLINE\_\_ bool IsFull ()
- \_\_INLINE\_\_ bool IsEmpty ()
- \_\_INLINE\_\_ size\_t GetWriteAvail ()
- \_\_INLINE\_\_ size\_t GetReadAvail ()
- bool Push (int numElements)
- bool Pop (int numElements)
- size\_t Push (T \*pSrcElements, size\_t numElements, int ExitMode=EXIT\_ON\_COMPLETED)
- size\_t Pop (T \*pDstElements, size\_t numElements, int ExitMode=EXIT\_ON\_COMPLETED)
- size\_t View (T \*pDstElements, size\_t numElements, int ExitMode=EXIT\_ON\_COMPLETED)

## Открытые атрибуты

- `size_t data_addr`  
физический адрес кольцевого буфера входных данных
- `size_t head_addr`  
сколько элементов ОТ НАЧАЛА ПОТОКА код MASTER уже записал в буфер входных данных [заполняется MASTER].
- `size_t tail_addr`  
сколько элементов ОТ НАЧАЛА ПОТОКА код SLAVE уже прочитал (обработал) [заполняется SLAVE].
- `size_t size`  
размер кольцевого буфера входных данных (в элементах; гарантируется что это степень двойки)
- `size_t head`  
сколько элементов ОТ НАЧАЛА ПОТОКА код MASTER уже записал в буфер входных данных [заполняется MASTER].
- `size_t tail`  
сколько элементов ОТ НАЧАЛА ПОТОКА код SLAVE уже прочитал (обработал) [заполняется SLAVE].
- `size_t id`  
сколько элементов ОТ НАЧАЛА ПОТОКА код SLAVE уже прочитал (обработал) [заполняется SLAVE].
- `bool isConnected`
- `t_bytecpy push_memcpy`
- `t_bytecpy pop_memcpy`
- `t_memcpy dma_init`
- `size_t(* dma_check )()`
- `T * dma_ptr`
- `size_t dma_left`
- `size_t dma_size`
- `unsigned timeout`
- `unsigned time2sleep`
- `int pad [16-3-5 *sizeof(t_bytecpy)/sizeof(int)]`  
резервные поля

## 7.12.1 Подробное описание

```
template<class T>
class C_RingBufferRemote< T >
```

См. определение в файле ringremote.h строка 39

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/ringremote.h`

## 7.13 Структура ds\_struct

## Открытые атрибуты

- `int nnSpot`  
общее число отбракованных пятен (по данному признаку)
- `int nnPxl`  
суммарное число пикселей в отбракованных пятнах
- `int dtSpot`  
общее время обработки отбракованных пятен (в тактах процессора)

### 7.13.1 Подробное описание

См. определение в файле iFloodFill.h строка 94

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iFloodFill.h

## 7.14 Класс EnterHardMode

### 7.14.1 Подробное описание

См. определение в файле macros\_fpu.h строка 21

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/macros\_fpu.h

## 7.15 Класс I\_2DSubPixelMinPosition

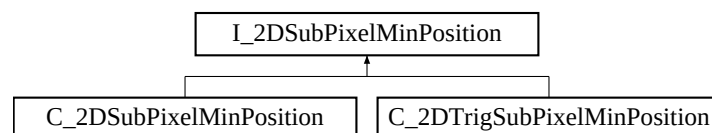
Открытые члены

- virtual void Find (float \*S9, float &dx, float &dy)=0
- virtual void Release ()=0

### 7.15.1 Подробное описание

См. определение в файле isubpixel2d.h строка 7

Граф наследования:I\_2DSubPixelMinPosition:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/isubpixel2d.h

## 7.16 Класс I\_PlessyCornerDetector

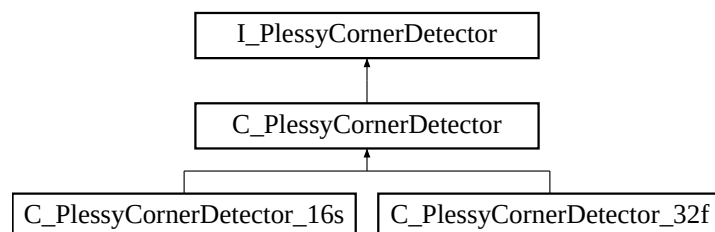
### Открытые члены

- virtual void Allocate (int w, int h, int ww)=0
- virtual void DeAllocate ()=0
- virtual void FindCorners (unsigned char \*Picture, int w, int h, int ww, float \*px, float \*py, int &nc)=0
- virtual void Release ()=0
- virtual void SetThreshold (float \_threshold)=0

### 7.16.1 Подробное описание

См. определение в файле iPlessy.h строка 10

Граф наследования:I\_PlessyCornerDetector:



Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmpli/iPlessy.h

## 7.17 Структура int15in16x4

### Открытые атрибуты

- unsigned long items

### 7.17.1 Подробное описание

См. определение в файле nmtype.h строка 235

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtype.h](#)

## 7.18 Структура int30in32x2

### Открытые атрибуты

- unsigned long items

### 7.18.1 Подробное описание

См. определение в файле `nmtype.h` строка 321

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.19 Структура `int31in32x2`

Открытые атрибуты

- `unsigned long items`

### 7.19.1 Подробное описание

См. определение в файле `nmtype.h` строка 301

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.20 Шаблон класса `mtr< T >`

Открытые члены

- `void origin (int y, int x)`
- `void setval (int y, int x, const T &val)`
- `void setvalx (int y, int x, const T &val)`
- `T getval (int y, int x) const`
- `T getvalx (int y, int x) const`
- `mtr (int nHeight, int nWidth, int nBorder=0)`
- `void resize (int nHeight, int nWidth, int nBorder=0)`
- `mtr (const mtr< T > &matr)`
- `void assign (T *Data, int nHeight, int nWidth, int nStride=0)`
- `mtr (T *Data, int nHeight, int nWidth, int nStride=0)`
- `mtr< T > & operator= (mtr< T > &matr)`
- `mtr< T > & operator= (T &val)`
- `T * operator[] (int row) const`
- `T & index (int idx)`
- `mtr< T > & operator*= (const T val)`
- `mtr< T > operator* (const T &val) const`
- `vec< T > operator* (const vec< T > &vect)`
- `mtr< T > operator* (const mtr< T > &matr)`
- `mtr< T > & operator+= (const T &val)`
- `mtr< T > & operator+= (const mtr< T > &matr)`
- `mtr< T > operator+ (const mtr< T > &matr) const`
- `mtr< T > & operator-= (const mtr< T > &matr)`



- `mtr< T > & operator-= (const T &val)`
- `mtr< T > operator- (const mtr< T > &matr) const`
- `mtr< T > operator- () const`
- `mtr< T > & operator/= (const T val)`
- `mtr< T > operator/ (const T val) const`
- `mtr< T > & operator>>= (const int shr)`
- `mtr< T > operator>> (const int shr) const`
- `mtr< T > & operator<<= (const int shl)`
- `mtr< T > operator<< (const int shl) const`
- `mtr< T > & operator &= (const T &val)`
- `mtr< T > & operator &= (const mtr< T > &matr)`
- `mtr< T > operator & (const mtr< T > &matr) const`
- `mtr< T > & operator|= (const mtr< T > &matr)`
- `mtr< T > operator| (const mtr< T > &matr) const`
- `mtr< T > & operator^= (const mtr< T > &matr)`
- `mtr< T > & operator^= (const T &val)`
- `mtr< T > operator^ (const mtr< T > &matr) const`
- `mtr< T > operator^ (const T &val) const`
- `mtr< T > operator~ ()`
- `void set (const T val)`
- `mtr< T > transpose ()`
- `mtr< T > & diag (T val)`
- `T * addr (int y, int x)`
- `void reset ()`
- `int sum ()`
- `vec< T > getvec (int y) const`
- `vec< T > getcol (int x) const`
- `T minpos (int &ypos, int &xpos)`
- `T maxpos (int &ypos, int &xpos)`
- `void CopyTo (T *pData)`
- `void CopyFrom (T *pData)`
- `template<class T2 >`  
`void ConvertTo (T2 *pData)`
- `template<class T2 >`  
`void ConvertFrom (T2 *pData)`
- `template<>`  
`mtr< double > & operator<<= (const int Shl)`

## Открытые атрибуты

- `int m_border`
- `int m_stride`
- `int m_height`
- `int m_width`
- `int m_size`
- `int m_x0`
- `int m_y0`
- `T * m_data`

## Защищенные данные

- `T * m_container`

### 7.20.1 Подробное описание

```
template<class T>  
class mtr< T >
```

См. определение в файле `tmatrix.h` строка 87

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmtl/tmatrix.h`

## 7.21 Структура `nm16sc`

Открытые атрибуты

- signed short `r`
- signed short `c`

### 7.21.1 Подробное описание

См. определение в файле `nmtype.h` строка 1240

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.22 Класс `nmchar`

Открытые члены

- `nmchar (unsigned int *p, int offset)`
- `nmchar (nmchar &ch)`
- `nmchar & operator= (nmchar ch)`
- `unsigned int operator+ (nmchar &ch)`
- `nmchar & operator&= (unsigned int val)`
- `nmchar & operator|= (unsigned int val)`
- `nmchar & operator= (unsigned int val)`
- `operator unsigned char ()`
- `uint8ptr operator& ()`

Открытые атрибуты

- unsigned int \* `adr`
- int `idx`

### 7.22.1 Подробное описание

См. определение в файле nmchar.h строка 13

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmchar.h

## 7.23 Шаблон класса nmchar1D< N >

Открытые члены

- [nmchar](#) & operator[] (int idx)
- operator uint8ptr ()
- unsigned int \* ptr ()

Открытые атрибуты

- [nmchar](#) deref
- unsigned int data [(N+3)/4]

### 7.23.1 Подробное описание

```
template<int N>  
class nmchar1D< N >
```

См. определение в файле nmchar.h строка 363

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmchar.h

## 7.24 Шаблон класса nmchar2D< Y, X >

Открытые члены

- [uint8ptr](#) & operator[] (int idx)
- unsigned int \* ptr ()

Открытые атрибуты

- [uint8ptr](#) arr
- unsigned int data [Y \*X/4]

### 7.24.1 Подробное описание

```
template<int Y, int X>
class nmchar2D< Y, X >
```

См. определение в файле nmchar.h строка 345

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmchar.h

## 7.25 Шаблон класса nmintpack< T >

### Открытые члены

- nmintpack (T \*base, int idx)
- nmintpack< T > & operator= (const nmintpack< T > &val)
- nmintpack< T > & operator= (const int &val)
- operator int (void) const
- \_\_INLINE\_\_ int intdisp (int indx)
- \_\_INLINE\_\_ int bitdisp (int indx)

### Открытые атрибуты

- T \* m\_container
- int m\_disp

### 7.25.1 Подробное описание

```
template<class T>
class nmintpack< T >
```

См. определение в файле tnmvespack.h строка 28

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmtl/tnmvespack.h

## 7.26 Шаблон класса nmmtr< T >

```
#include <tnmmtr.h>
```

## Открытые члены

- `nmmtr (int Height, int Width, int Border=0)`
- `nmvec< T > operator[] (int y) const`
- `nmmtr (nmmtr< T > &mtr)`
- `nmmtr (T *Data, int Height, int Width, int Stride=0)`
- `nmmtr (const T *Data, int Height, int Width, int Stride=0)`
- `nmmtr< T > & operator= (const nmmtr< T > &mtr)`
- `nmmtr< T > & operator*= (const nmint< T > &val)`
- `template<class T2 >`  
`nmmtr< T2 > operator* (const nmint< T2 > &val)`
- `template<class T2 >`  
`nmvec< T2 > operator* (const nmvec< T2 > &vec)`
- `template<class T2 >`  
`nmmtr< T2 > operator* (const nmmtr< T2 > &mtr)`
- `nmvec< T > & operator+= (const nmint< T > &val)`
- `nmmtr< T > & operator+= (const nmmtr< T > &mtr)`
- `nmmtr< T > operator+ (const nmmtr< T > &mtr) const`
- `nmmtr< T > & operator-= (const nmmtr< T > &mtr)`
- `nmmtr< T > operator- (const nmmtr< T > &mtr) const`
- `nmmtr< T > operator- () const`
- `nmmtr< T > & operator/= (const T val)`
- `nmmtr< T > operator/ (const nmint< T > val) const`
- `nmmtr< T > & operator>>= (const int shr)`
- `nmmtr< T > operator>> (const int shr) const`
- `nmmtr< T > & operator<<= (const int shl)`
- `nmmtr< T > operator<< (const int shl) const`
- `nmmtr< T > & operator|= (const nmmtr< T > &mtr)`
- `nmmtr< T > operator| (const nmmtr< T > &mtr) const`
- `nmmtr< T > & operator &= (const nmmtr< T > &mtr)`
- `nmmtr< T > operator & (const nmmtr< T > &mtr) const`
- `nmmtr< T > & operator^= (const nmint< T > &val)`
- `nmmtr< T > operator^ (const nmint< T > &val) const`
- `nmmtr< T > operator~ () const`
- `T * addr (int y, int x)`
- `nmvec< T > vec (int y)`
- `void fill (nmint< T > &nVal)`
- `nmmtr< T > transpose ()`
- `template<class T2 >`  
`void set (nmmtr< T2 > &mSrcMtr) const`
- `template<class T2 >`  
`void set (mtr< T2 > &Mtr)`
- `void set (const T val)`
- `void reset ()`

## Открытые атрибуты

- `int m_height`
- `int m_width`
- `int m_size`
- `int m_stride`
- `int m_border`
- `T * m_data`

## Защищенные данные

- `T * m_container`

### 7.26.1 Подробное описание

```
template<class T>
class nmmtr< T >
```

класс матриц.

См. определение в файле `tnmmtr.h` строка 37

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmtl/tnmmtr.h`

## 7.27 Структура NmppsFFTSpec

### Открытые атрибуты

- `nm32sc * buffer [FFT_SPEC_NUM_BUFFERS]`
- `void * fftTable [FFT_SPEC_NUM_TABLES]`
- `RoundShift norm [4]`
- `int shift [FFT_SPEC_NUM_SHIFTS]`
- `int amp [FFT_SPEC_NUM_AMPLITUDES]`
- `int round [8]`
- `Free32Func * free`

### 7.27.1 Подробное описание

См. определение в файле `fft.h` строка 89

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmpls/fft.h`

## 7.28 Структура NmppsFFTSpec\_32fcr

### Открытые атрибуты

- `nm32fcr * Buffers [NUMBUFF1]`
- `nm32fcr * Bufts [NUMBUFF2]`
- `int order`

### 7.28.1 Подробное описание

См. определение в файле `fft_32fcr.h` строка 25

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/fft_32fcr.h`

## 7.29 Структура NmppsFrame\_16s

Открытые атрибуты

- `void * pull`
- `nm16s * data`

### 7.29.1 Подробное описание

См. определение в файле `malloc32.h` строка 97

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/malloc32.h`

## 7.30 Структура NmppsFrame\_16u

Открытые атрибуты

- `void * pull`
- `nm16u * data`

### 7.30.1 Подробное описание

См. определение в файле `malloc32.h` строка 92

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/malloc32.h`

## 7.31 Структура NmppsFrame\_32s

Открытые атрибуты

- `void * pull`
- `nm32s * data`

### 7.31.1 Подробное описание

См. определение в файле malloc32.h строка 107

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.32 Структура NmppsFrame\_32u

Открытые атрибуты

- void \* pull
- [nm32u](#) \* data

### 7.32.1 Подробное описание

См. определение в файле malloc32.h строка 102

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.33 Структура NmppsFrame\_64s

Открытые атрибуты

- void \* pull
- [nm64s](#) \* data

### 7.33.1 Подробное описание

См. определение в файле malloc32.h строка 117

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.34 Структура NmppsFrame\_64u

Открытые атрибуты

- void \* pull
- [nm64u](#) \* data



#### 7.34.1 Подробное описание

См. определение в файле malloc32.h строка 112

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

### 7.35 Структура NmppsFrame\_8s

Открытые атрибуты

- void \* pull
- nm8s \* data

#### 7.35.1 Подробное описание

См. определение в файле malloc32.h строка 87

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

### 7.36 Структура NmppsFrame\_8u

Открытые атрибуты

- void \* pull
- nm8u \* data

#### 7.36.1 Подробное описание

См. определение в файле malloc32.h строка 82

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.37 Структура NmppsMallocSpec

### Открытые атрибуты

- Malloc32Func \* allocator [4]
- enum MALLOC32\_MODE mode
- uint32 random
- fseq32 priority
- uint32 status
- uint32 time
- uint32 timeBest
- uint32 routePos
- fseq64 route [NMPPS\_MALLOC\_LIMIT/16]
- fseq64 bestRoute [NMPPS\_MALLOC\_LIMIT/16]
- void \* allocHistory [NMPPS\_MALLOC\_LIMIT]
- void \* freeHistory [NMPPS\_MALLOC\_LIMIT]
- uint32 allocHistoryPos
- uint32 freeHistoryPos
- uint32 firstPass

### 7.37.1 Подробное описание

См. определение в файле malloc32.h строка 42

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.38 Структура NmppsTmpSpec

### Открытые атрибуты

- void \* buffer0
- void \* buffer1

### 7.38.1 Подробное описание

См. определение в файле malloc32.h строка 60

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.39 Структура nmreg

```
#include <nmtypes.h>
```

## Открытые атрибуты

- int nVal

## 7.39.1 Подробное описание

NM регистр.

См. определение в файле nmtype.h строка 55

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmtype.h

## 7.40 Класс nmshort

## Открытые члены

- `__INLINE__ nmshort (nmshort &ch)`
- `__INLINE__ nmshort & operator= (nmshort ch)`
- `__INLINE__ unsigned int operator+ (nmshort &ch)`
- `__INLINE__ nmshort & operator= (unsigned int val)`
- `__INLINE__ operator unsigned char ()`
- `__INLINE__ uint16ptr operator& ()`

## Открытые атрибуты

- unsigned int \* adr
- int idx

## 7.40.1 Подробное описание

См. определение в файле nmshort.h строка 8

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmshort.h

## 7.41 Шаблон класса nmshort2D&lt; Y, X &gt;

## Открытые члены

- `uint16ptr & operator[] (int idx)`
- `unsigned int * ptr ()`

## Открытые атрибуты

- `uint16ptr` arr
- `unsigned int` data [Y \*X/2]

## 7.41.1 Подробное описание

```
template<int Y, int X>
class nmshort2D< Y, X >
```

См. определение в файле `nmshort.h` строка 292

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmshort.h`

7.42 Шаблон класса `nmvecpack< T >`

```
#include <tnmvecpack.h>
```

## Открытые члены

- `nmvecpack` (void \*Data, int Size, int Border=0)
- `nmvecpack` (int Size, int Border=0)
- `nmvecpack` (const `nmvecpack`< T > &vec)
- `nmvecpack`< T > & operator= (const `nmvecpack`< T > &vec)
- `__INLINE__ nmintpack`< T > operator[] (int idx)
- `template<class T2 >`  
`nmvecpack`< T2 > & operator\*= (const `nmint`< T2 > val)
- `template<class T2 >`  
`nmvecpack`< T2 > operator\* (const `nmint`< T2 > &val) const
- `template<class T2 >`  
`nmint`< T2 > operator\* (const `nmvecpack`< T2 > &vec) const
- `nmvecpack`< T > & operator+= (const `nmint`< T > &val)
- `nmvecpack`< T > & operator+= (const `nmvecpack`< T > &vec)
- `nmvecpack`< T > operator+ (const `nmint`< T > &val) const
- `nmvecpack`< T > operator+ (const `nmvecpack`< T > &vec) const
- `nmvecpack`< T > & operator-= (const `nmint`< T > &val)
- `nmvecpack`< T > & operator-= (const `nmvecpack`< T > &vec)
- `nmvecpack`< T > operator- () const
- `nmvecpack`< T > operator- (const `nmint`< T > &val) const
- `nmvecpack`< T > operator- (const `nmvecpack`< T > &vec) const
- `nmvecpack`< T > & operator/= (const `nmint`< T > val)
- `nmvecpack`< T > operator/ (const T val) const
- `nmvecpack`< T > & operator>>= (const int shr)
- `nmvecpack`< T > operator>> (const int shr) const
- `nmvecpack`< T > & operator<<= (const int shl)
- `nmvecpack`< T > operator<< (const int shl) const
- `nmvecpack`< T > & operator|= (const `nmint`< T > &val)
- `nmvecpack`< T > & operator|= (const `nmvecpack`< T > &vec)

- `nmvecpack< T > operator| (const nmint< T > &val) const`
- `nmvecpack< T > operator| (const nmvecpack< T > &vec) const`
- `nmvecpack< T > & operator &= (const nmint< T > &val)`
- `nmvecpack< T > & operator &= (const nmvecpack< T > &vec)`
- `nmvecpack< T > operator & (const nmint< T > &val) const`
- `nmvecpack< T > operator & (const nmvecpack< T > &vec) const`
- `nmvecpack< T > & operator^= (const nmint< T > &val)`
- `nmvecpack< T > & operator^= (const nmvecpack< T > &vec)`
- `nmvecpack< T > operator^ (const nmint< T > &val) const`
- `nmvecpack< T > operator^ (const nmvecpack< T > &vec) const`
- `nmvecpack< T > & operator~ () const`
- `bool operator== (const nmvecpack< T > &vec) const`
- `bool operator!= (const nmvecpack< T > &vec) const`
- `template<class T2 >`  
`void SetData (T2 *Data)`
- `template<class T2 >`  
`void GetData (T2 *Data)`
- `void reset ()`

### Открытые атрибуты

- `int m_size`
- `T * m_data`

### Защищенные данные

- `T * m_container`
- `int m_border`

#### 7.42.1 Подробное описание

```
template<class T>
class nmvecpack< T >
```

Класс векторов.

Примеры:

```
int Test[10]={1,125,3,4,5,6,7,8,9,10};
int Res [10];
nmvecpack<int> A0(3);
nmvecpack<int> B0(3);
nmvecpack<int> C0(3);
```

См. определение в файле `tnmvecpack.h` строка 137

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmtl/tnmvecpack.h`

## 7.43 Структура RGB32\_nm10s

Открытые атрибуты

- int nB:10
- int nG:10
- int nR:10
- int nA:2

### 7.43.1 Подробное описание

См. определение в файле iDef.h строка 59

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iDef.h

## 7.44 Структура RGB32\_nm10u

Открытые атрибуты

- unsigned int nB:10
- unsigned int nG:10
- unsigned int nR:10
- unsigned int nA:2

### 7.44.1 Подробное описание

См. определение в файле iDef.h строка 51

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iDef.h

## 7.45 Структура RGB32\_nm8s

Открытые атрибуты

- int nB:8
- int nG:8
- int nR:8
- int nA:8

### 7.45.1 Подробное описание

См. определение в файле iDef.h строка 43

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iDef.h

## 7.46 Структура RGB32\_nm8u

Открытые атрибуты

- unsigned int nB:8
- unsigned int nG:8
- unsigned int nR:8
- unsigned int nA:8

### 7.46.1 Подробное описание

См. определение в файле iDef.h строка 35

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iDef.h

## 7.47 Структура RGB64\_nm16u

Открытые атрибуты

- unsigned int nB:16
- unsigned int nG:16
- unsigned int nR:16
- unsigned int nA:16

### 7.47.1 Подробное описание

См. определение в файле iDef.h строка 67

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iDef.h

## 7.48 Структура RoundShift

Открытые атрибуты

- int round
- int shift

### 7.48.1 Подробное описание

См. определение в файле `fft.h` строка 84

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmpls/fft.h`

## 7.49 Класс RPoint

Открытые члены

- `RPoint (const RPoint &p)`
- `RPoint (double _x, double _y)`
- `RPoint & operator= (const RPoint &p)`

Открытые атрибуты

- double x
- double y

### 7.49.1 Подробное описание

См. определение в файле `iCellTexture.h` строка 25

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmpli/iCellTexture.h`

## 7.50 Структура S\_BufferInfo

класс буфер - заголовок в начале выделяемой динамической памяти

```
#include <multiheap.h>
```



## Открытые члены

- `int * DataBegin ()`  
Возвращает указатель на данные в буфере
- `int * DataEnd ()`  
Возвращает указатель на конец данных в буфере (следу)
- `int * EndGuardBits ()`  
Возвращает указатель на конечные защитные поля (2 слова)
- `unsigned CheckGuardBits ()`

## Открытые атрибуты

- `int guardInfoBits0`
- `size_t size32Buffer`  
< размер массива данных в буфере в 32-х словах
- `S\_BufferInfo * pPrevBuffer`  
< указатель на предыдущий буфер в списке
- `S\_BufferInfo * pNextBuffer`  
< указатель на следующий буфер в списке
- `bool isLocked`  
< флаг блокировки буфера, запрещающий его удаление с помощью `Release()`
- `int guardInfoBits1`

## 7.50.1 Подробное описание

класс буфер - заголовок в начале выделяемой динамической памяти

См. определение в файле `multiheap.h` строка 14

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/multiheap.h`

## 7.51 Структура s\_int32x2

## Открытые атрибуты

- `int hi`
- `int lo`

## 7.51.1 Подробное описание

См. определение в файле `nmtype.h` строка 259

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.52 Структура `s_nm32fc`

Открытые атрибуты

- `float re`
- `float im`

### 7.52.1 Подробное описание

См. определение в файле `nmtype.h` строка 1291

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.53 Структура `s_nm32fer`

Открытые атрибуты

- `float im`
- `float re`

### 7.53.1 Подробное описание

См. определение в файле `nmtype.h` строка 1298

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.54 Структура `s_nm32sc`

Открытые атрибуты

- `nm32s re`
- `nm32s im`

### 7.54.1 Подробное описание

См. определение в файле `nmtype.h` строка 1264

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.55 Структура s\_nm64sc

### Открытые атрибуты

- long long [re](#)
- long long [im](#)

#### 7.55.1 Подробное описание

См. определение в файле nmtype.h строка 1327

#### 7.55.2 Данные класса

##### 7.55.2.1 im

long long s\_nm64sc::im

Мнимая часть комплексного числа.

См. определение в файле nmtype.h строка 1340

##### 7.55.2.2 re

long long s\_nm64sc::re

Вещественная часть комплексного числа.

См. определение в файле nmtype.h строка 1334

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtype.h](#)

## 7.56 Структура S\_nmppiFilterKernel

### Открытые атрибуты

- [nm32s](#) \* pDispArray
- [nm32s](#) \* pWeightMatrix

### 7.56.1 Подробное описание

См. определение в файле iFilter.h строка 27

Объявления и описания членов структуры находятся в файле:

- [D:/GIT/nmpp/include/nmpli/iFilter.h](#)

## 7.57 Структура S\_nmppiFilterKernel\_32s32s

Открытые атрибуты

- [nm32s](#) \* pDispArray
- [nm32s](#) \* pWeightMatrix
- int nKerWidth
- int nKerHeight

### 7.57.1 Подробное описание

См. определение в файле iFilter.h строка 40

Объявления и описания членов структуры находятся в файле:

- [D:/GIT/nmpp/include/nmpli/iFilter.h](#)

## 7.58 Структура s\_v16nm16s

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [4]

### 7.58.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 16р. чисел со знаком.

См. определение в файле nmtypes.h строка 1024

Объявления и описания членов структуры находятся в файле:

- [D:/GIT/nmpp/include/nmtypes.h](#)

## 7.59 Структура s\_v16nm16u

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [4]

### 7.59.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 16р. чисел без знака.

См. определение в файле nmtypes.h строка 1161

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.60 Структура s\_v16nm32s

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [8]

### 7.60.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 32р. чисел со знаком.

См. определение в файле nmtypes.h строка 1069

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.61 Структура s\_v16nm32u

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [8]

### 7.61.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 32-р. чисел без знака.

См. определение в файле `nmtype.h` строка 1205

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.62 Структура `s_v16nm4u`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [1]`

### 7.62.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 4-р. чисел без знака.

См. определение в файле `nmtype.h` строка 1095

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.63 Структура `s_v16nm8s`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [2]`

### 7.63.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 8-р. чисел со знаком.

См. определение в файле `nmtype.h` строка 989

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.64 Структура s\_v16nm8u

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [2]

### 7.64.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 8-б. чисел без знака.

См. определение в файле nmtypes.h строка 1128

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.65 Структура s\_v2nm32f

Открытые атрибуты

- float v0
- float v1

### 7.65.1 Подробное описание

См. определение в файле nmtypes.h строка 1274

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.66 Структура s\_v2nm32s

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [1]

### 7.66.1 Подробное описание

Тип векторной структуры, состоящей из 2-х 32р. чисел со знаком.

См. определение в файле `nmtype.h` строка 1036

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.67 Структура `s_v2nm32u`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [1]`

### 7.67.1 Подробное описание

Тип векторной структуры, состоящей из 2-х 32р. чисел без знака.

См. определение в файле `nmtype.h` строка 1172

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.68 Структура `s_v4nm16s`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [1]`

### 7.68.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 16р. чисел со знаком.

См. определение в файле `nmtype.h` строка 1000

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`



## 7.69 Структура s\_v4nm16u

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [1]

### 7.69.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 16р. чисел без знака.

См. определение в файле nmtypes.h строка 1139

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.70 Структура s\_v4nm32f

Открытые атрибуты

- float vec [4]

### 7.70.1 Подробное описание

См. определение в файле nmtypes.h строка 1280

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.71 Структура s\_v4nm32s

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [2]

### 7.71.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 32р. чисел со знаком.

См. определение в файле `nmtype.h` строка 1047

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.72 Структура `s_v4nm32u`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [2]`

### 7.72.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 32р. чисел без знака.

См. определение в файле `nmtype.h` строка 1183

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.73 Структура `s_v4nm64f`

Открытые атрибуты

- `double vec [4]`

### 7.73.1 Подробное описание

См. определение в файле `nmtype.h` строка 1285

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.74 Структура `s_v4nm8u`

```
#include <nmtype.h>
```

Открытые атрибуты

- unsigned long long vec [1]

#### 7.74.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 8р. чисел без знака.

См. определение в файле nmtype.h строка 1106

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtype.h](#)

## 7.75 Структура s\_v8nm16s

```
#include <nmtype.h>
```

Открытые атрибуты

- unsigned long long vec [2]

#### 7.75.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 16р. чисел со знаком.

См. определение в файле nmtype.h строка 1013

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtype.h](#)

## 7.76 Структура s\_v8nm16u

```
#include <nmtype.h>
```

Открытые атрибуты

- unsigned long long vec [2]

### 7.76.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 16р. чисел без знака.

См. определение в файле `nmtype.h` строка 1150

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.77 Структура `s_v8nm32s`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [4]`

### 7.77.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 32р. чисел со знаком.

См. определение в файле `nmtype.h` строка 1058

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.78 Структура `s_v8nm32u`

```
#include <nmtype.h>
```

Открытые атрибуты

- `unsigned long long vec [4]`

### 7.78.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 32р. чисел без знака.

См. определение в файле `nmtype.h` строка 1194

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

## 7.79 Структура s\_v8nm8s

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [1]

### 7.79.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 8р. чисел со знаком.

См. определение в файле nmtypes.h строка 978

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.80 Структура s\_v8nm8u

```
#include <nmtypes.h>
```

Открытые атрибуты

- unsigned long long vec [1]

### 7.80.1 Подробное описание

Тип векторной структуры, состоящей из 8-ми 8р. чисел без знака.

См. определение в файле nmtypes.h строка 1117

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.81 Структура SpecTmp1

Открытые атрибуты

- void \* buffer
- int status
- int mode
- fseq64 route

### 7.81.1 Подробное описание

См. определение в файле malloc32.h строка 153

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/malloc32.h

## 7.82 Структура spot\_struct

Открытые атрибуты

- int [Xmin](#)  
координаты минимального прямоугольника, содержащего пятно
- int [Ymin](#)  
координаты минимального прямоугольника, содержащего пятно
- int [Xmax](#)  
координаты минимального прямоугольника, содержащего пятно
- int [Ymax](#)  
координаты минимального прямоугольника, содержащего пятно
- int [noPxl](#)  
номер начального пиксела следующего пятна в массиве pixels.
- int [dtSpot](#)  
время обработки пятна (в тактах процессора)

### 7.82.1 Подробное описание

См. определение в файле iFloodFill.h строка 84

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/nmpli/iFloodFill.h

## 7.83 Структура tagSegmentInfo

Открытые атрибуты

- int xMin
- int yMin
- int xMax
- int yMax
- int N

## 7.83.1 Подробное описание

См. определение в файле `iFloodFill.h` строка 22

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmpli/iFloodFill.h`

7.84 Шаблон класса `tfixpoint< T, point >`

Открытые члены

- `INLINE tfixpoint (const int val)`
- `INLINE tfixpoint (const float val)`
- `INLINE tfixpoint (const double val)`
- `INLINE tfixpoint< T, point > & operator= (const int val)`
- `INLINE tfixpoint< T, point > & operator= (const float val)`
- `tfixpoint< T, point > & operator= (const double val)`
- `INLINE tfixpoint< T, point > & operator= (const tfixpoint< T, point > &val)`
- `INLINE tfixpoint< T, point > & operator-= (const tfixpoint< T, point > &val)`
- `INLINE tfixpoint< T, point > & operator+= (const tfixpoint< T, point > &val)`
- `INLINE tfixpoint< T, point > operator- (const tfixpoint< T, point > &val) const`
- `INLINE tfixpoint< T, point > operator+ (const tfixpoint< T, point > &val) const`
- `INLINE tfixpoint< T, point > & operator++ (int)`
- `INLINE tfixpoint< T, point > & operator-- (int)`
- `template<class T2 , int point2>`  
`INLINE tfixpoint< T, point > & operator*= (const tfixpoint< T2, point2 > val)`
- `INLINE tfixpoint< T, point > & operator*= (const int val)`
- `INLINE tfixpoint< T, point > & operator*= (const float val)`
- `INLINE tfixpoint< T, point > & operator*= (const double val)`
- `template<int point2>`  
`INLINE tfixpoint< T, point > operator* (const tfixpoint< T, point2 > val) const`
- `INLINE tfixpoint< T, point > operator* (const int val) const`
- `INLINE tfixpoint< T, point > & operator/= (const tfixpoint< T, point > val)`
- `INLINE tfixpoint< T, point > operator/ (const tfixpoint< T, point > val) const`
- `INLINE tfixpoint< T, point > & operator>= (const int y)`
- `INLINE tfixpoint< T, point > operator>> (const int y) const`
- `INLINE tfixpoint< T, point > & operator<= (const int y)`
- `INLINE tfixpoint< T, point > operator<< (const int n) const`
- `INLINE tfixpoint< T, point > & operator^= (tfixpoint< T, point > &val)`
- `INLINE tfixpoint< T, point > operator^ (tfixpoint< T, point > &val) const`
- `INLINE tfixpoint< T, point > operator- () const`
- `bool operator> (const tfixpoint< T, point > &y) const`
- `bool operator>= (const tfixpoint< T, point > &y) const`
- `bool operator< (const tfixpoint< T, point > &y) const`
- `bool operator<= (const tfixpoint< T, point > &y) const`
- `bool operator== (const tfixpoint< T, point > &y) const`
- `bool operator!= (const tfixpoint< T, point > &y) const`
- `float flt ()`

## Открытые атрибуты

- `T m_value`

### 7.84.1 Подробное описание

```
template<class T, int point>
class tfixpoint< T, point >
```

См. определение в файле `tfixpoint.h` строка 52

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmtl/tfixpoint.h`

## 7.85 Структура `Tmp2BuffSpec`

### Открытые атрибуты

- `void * buffer0`
- `void * buffer1`
- `fseq64 route`
- `int mode`
- `int status`

### 7.85.1 Подробное описание

См. определение в файле `nmtypе.h` строка 1366

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtypе.h`

## 7.86 Класс `uint16ptr`

### Открытые члены

- `__INLINE__ unsigned char * x86addr ()`
- `__INLINE__ uint16ptr (void *p)`
- `__INLINE__ uint16ptr (unsigned short *p)`
- `__INLINE__ uint16ptr (short *p)`
- `__INLINE__ uint16ptr (const uint16ptr &p)`
- `__INLINE__ uint16ptr (unsigned int *p)`
- `__INLINE__ uint16ptr (unsigned int *p, int offset)`
- `__INLINE__ nmshort & operator[] (int idx)`
- `__INLINE__ uint16ptr & operator= (unsigned int *ptr)`
- `__INLINE__ bool operator< (uint16ptr ptr)`



- `__INLINE__ bool operator>= (uint16ptr ptr)`
- `__INLINE__ int operator- (uint16ptr ptr)`
- `__INLINE__ uint16ptr & operator= (const uint16ptr &p)`
- `__INLINE__ unsigned int * ptr ()`
- `__INLINE__ nmshort & operator* ()`
- `__INLINE__ uint16ptr operator+ (int idx)`
- `__INLINE__ uint16ptr operator- (int idx)`
- `__INLINE__ uint16ptr & operator+= (int idx)`
- `__INLINE__ uint16ptr & operator-= (int idx)`
- `__INLINE__ uint16ptr operator++ (int)`
- `__INLINE__ uint16ptr & operator++ ()`
- `__INLINE__ unsigned int operator== (unsigned int N)`
- `__INLINE__ bool operator== (uint16ptr ptr)`

#### Открытые атрибуты

- `unsigned int * addr`
- `int idx`
- `nmshort arref`

#### 7.86.1 Подробное описание

См. определение в файле `nmshort.h` строка 57

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmshort.h`

## 7.87 Класс uint8ptr

#### Открытые члены

- `unsigned char * x86addr ()`
- `uint8ptr (const void *p)`
- `uint8ptr (const unsigned char *p)`
- `uint8ptr (const char *p)`
- `uint8ptr (const uint8ptr &p)`
- `uint8ptr (unsigned int *p)`
- `uint8ptr (unsigned int *p, int offset)`
- `nmchar & operator[] (int idx)`
- `uint8ptr & operator= (unsigned int *ptr)`
- `bool operator< (uint8ptr ptr)`
- `bool operator> (uint8ptr ptr)`
- `bool operator>= (uint8ptr ptr)`
- `int operator- (uint8ptr ptr)`
- `uint8ptr & operator= (const uint8ptr &p)`
- `unsigned int * ptr ()`
- `nmchar & operator* ()`
- `uint8ptr operator+ (int idx)`
- `uint8ptr operator- (int idx)`
- `uint8ptr & operator+= (int idx)`
- `uint8ptr & operator-= (int idx)`
- `uint8ptr operator++ (int)`
- `uint8ptr & operator++ ()`
- `bool operator== (uint8ptr ptr)`

## Открытые атрибуты

- unsigned int \* addr
- int indx
- [nmchar](#) arref

### 7.87.1 Подробное описание

См. определение в файле nmchar.h строка 79

Объявления и описания членов класса находятся в файле:

- D:/GIT/nmpp/include/nmchar.h

## 7.88 Структура v16nm4s

```
#include <nmtypes.h>
```

## Открытые атрибуты

- unsigned long long vec [1]

### 7.88.1 Подробное описание

Тип векторной структуры, состоящей из 16-ти 4-р. чисел со знаком.

См. определение в файле nmtypes.h строка 955

Объявления и описания членов структуры находятся в файле:

- D:/GIT/nmpp/include/[nmtypes.h](#)

## 7.89 Структура v4nm8s

```
#include <nmtypes.h>
```

## Открытые атрибуты

- unsigned long long vec [1]

## 7.89.1 Подробное описание

Тип векторной структуры, состоящей из 4-х 8-б. чисел со знаком.

См. определение в файле `nmtype.h` строка 967

Объявления и описания членов структуры находятся в файле:

- `D:/GIT/nmpp/include/nmtype.h`

7.90 Шаблон класса `vec< T >`

```
#include <tvector.h>
```

Открытые члены

- `void resize (int Size, int Border=0)`
- `vec (T *Data, int Size, int Border=0)`
- `vec (int Size, int Border=0)`
- `vec (const vec< T > &vect)`
- `void reset ()`
- `vec< T > & InitRamp (T StartValue, T Increment)`
- `int MaxPos (T &maxval)`
- `int MinPos (T &minval)`
- `double Mean ()`
- `T & CustomMax ()`
- `vec< T > & operator= (const T &val)`
- `vec< T > & operator= (const vec< T > &vect)`
- `T * addr (int idx)`
- `T & operator[] (size_t idx)`
- `vec< T > & operator*= (const T &val)`
- `vec< T > operator* (const T &val) const`
- `T operator* (const vec< T > &vect) const`
- `vec< T > operator* (const mtr< T > matr) const`
- `vec< T > & operator+= (const T &val)`
- `vec< T > & operator+= (const vec< T > &vect)`
- `vec< T > operator+ (const T &val) const`
- `vec< T > operator+ (const vec< T > &vect) const`
- `vec< T > & operator-= (const T &val)`
- `vec< T > & operator-= (const vec< T > &vect)`
- `vec< T > operator- (const vec< T > &vect) const`
- `vec< T > operator- () const`
- `vec< T > & operator/= (const T val)`
- `vec< T > operator/ (const T val) const`
- `vec< T > & operator>>= (const int shr)`
- `vec< T > operator>> (const int shr) const`
- `vec< T > & operator<<= (const int shl)`
- `vec< T > operator<< (const int shl) const`
- `vec< T > & operator &= (const T &val)`
- `vec< T > & operator &= (const vec< T > &vect)`
- `vec< T > operator & (const T &val) const`

- `vec< T > operator & (const vec< T > &vect) const`
- `vec< T > & operator|= (const T &val)`
- `vec< T > & operator|= (const vec< T > &vect)`
- `vec< T > operator| (const T &val) const`
- `vec< T > operator| (const vec< T > &vect) const`
- `vec< T > & operator^= (const T &val)`
- `vec< T > & operator^= (const vec< T > &vect)`
- `vec< T > operator^ (const T &val) const`
- `vec< T > operator^ (const vec< T > &vect) const`
- `int sum ()`
- `bool operator== (const vec< T > &vect)`
- `bool operator!= (const vec< T > &vect)`

### Открытые атрибуты

- `int m_border`
- `int size`
- `T * m_data`

### Защищенные данные

- `T * m_container`

### 7.90.1 Подробное описание

```
template<class T>
class vec< T >
```

Класс векторов.

Примеры:

```
int Test[10]={1,125,3,4,5,6,7,8,9,10};
int Res [10];
vec<int> A0(3);
vec<int> B0(3);
vec<int> C0(3);
scalar<int> a0(2);
A0[0]=1;
A0[1]=2;
A0[2]=3;
B0.SetData(Test);
B0=A0;
a0=A0[1];

C0=A0+B0;
C0=A0*a0;
a0=A0*A0;
```

См. определение в файле `tvector.h` строка 79

Объявления и описания членов класса находятся в файле:

- `D:/GIT/nmpp/include/nmtl/tvector.h`

## Глава 8

### Файлы

#### 8.1 Файл D:/GIT/nmpp/include/nmblas.h

nmblas (NeuroMatrix Basic Linear Algebra Subroutines)

#### Перечисления

- enum nm\_trans { nm\_n =0, nm\_t =1 }

#### Функции

- double nmblas\_dasum (const int N, const double \*X, const int INCX)
- void nmblas\_daxpy (const int N, const double A, const double \*X, const int INCX, double \*Y, const int INCY)
- void nmblas\_dcopy (const int N, const double \*X, const int INCX, double \*Y, const int INCY)
- double nmblas\_ddot (const int N, const double \*X, const int INCX, const double \*Y, const int INCY)
- double nmblas\_dnorm2 (const int N, const double \*x, const int INCX)
- void nmblas\_drot (const int N, double \*X, const int INCX, double \*Y, const int INCY, const double C, const double S)
- void nmblas\_drotg (double \*A, double \*B, double \*C, double \*S)
- void nmblas\_drotm (const int N, double \*X, const int INCX, double \*Y, const int INCY, double \*param)
- void nmblas\_dscal (const int N, const double ALPHA, double \*X, const int INCX)
- double nmblas\_dsdot (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)
- void nmblas\_dswap (const int N, double \*X, const int INCX, double \*Y, const int INCY)
- double nmblas\_dznrm2 (const int N, const double \*X, const int INCX)
- int nmblas\_idamax (const int N, const double \*X, const int INCX)
- int nmblas\_isamax (const int N, const float \*X, const int INCX)
- float nmblas\_sasum (const int N, const float \*X, const int INCX)
- void nmblas\_saxpy (int N, const float A, const float \*X, const int INCX, float \*Y, const int INCY)
- float nmblas\_scnrm2 (const int N, const float \*X, const int INCX)
- void nmblas\_scopy (const int N, const float \*X, const int INCX, float \*Y, const int INCY)
- float nmblas\_sdot (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)
- float nmblas\_sdsdot (const int N, const float B, const float \*X, const int INCX, const float \*Y, const int INCY)
- float nmblas\_snrm2 (const int N, const float \*X, const int INCX)

- void nmblas\_srot (const int N, float \*X, const int INCX, float \*Y, const int INCY, const float C, const float S)
- void nmblas\_srotm (const int N, float \*X, const int INCX, float \*Y, const int INCY, float \*param)
- void [nmblas\\_sscal](#) (const int N, const float A, const float \*X, const int INCX)
- void [nmblas\\_sswap](#) (const int N, const float \*X, const int INCX, const float \*Y, const int INCY)
- void nmblas\_sgemv (const enum nm\_trans TRANS, const int M, const int N, const float ALPHA, const float \*A, const int LDA, const float \*X, const int INCX, const float BETA, float \*Y, const int INCY)
- void nmblas\_dgemv (const enum nm\_trans TRANS, const int M, const int N, const double ALPHA, const double \*A, const int LDA, const double \*X, const int INCX, const double BETA, double \*Y, const int INCY)
- void MullMatrix\_f (void \*A, int pI, int ldA, void \*B, int pK, int ldB, void \*C, int pJ, int ldC, bool plusC)

### 8.1.1 Подробное описание

nmblas (NeuroMatrix Basic Linear Algebra Subroutines)

This class is used to demonstrate a number of section commands.

Автор

Alexandr Bolotnikov

Версия

1.0

Дата

2017-05-23T15:13:13

[Ошибка](#)

Предупреждения

Авторство

(c) RC Module Inc.

## 8.2 Файл D:/GIT/nmpp/include/nmblas/nmblas\_sgemm.h

part of nmblas library, sgemm function implementation

Макросы

- `#define ALL_FPU(instr)`

### 8.2.1 Подробное описание

part of nmblas library, sgemm function implementation

single precision matrix multiplication implementation

$C = \alpha * A * B + \beta * C$

see blas docs for detailed description

limitations: M, N, K, ld\* must be all even, all matrix rows must be even aligned

limitations: row major (C order, not Fortran) only

limitations: TransA is not supported yet

Автор

leshabirukov

Версия

1.0

Дата

2018-07-26

Аргументы

|        |        |                                                               |
|--------|--------|---------------------------------------------------------------|
| in     | TransA | transpose first multiplicand (A) flag                         |
| in     | TransB | transpose second multiplicand (B) flag                        |
| in     | M      | number of rows of the matrix A and C                          |
| in     | N      | number of columns of the matrix B and C                       |
| in     | K      | number of rows of the matrix B and of columns of the matrix C |
| in     | alpha  | specifies the scalar alpha                                    |
| in     | A      | pointer to the first multiplicand                             |
| in     | lda    | A stride, first dimension of A holder                         |
| in     | B      | pointer to the second multiplicand                            |
| in     | ldb    | B stride, first dimension of B holder                         |
| in     | beta   | specifies the scalar beta                                     |
| in,out | C      | pointer to accumulator matrix                                 |
| in     | ldc    | C stride, first dimension of C holder                         |

[Ошибка](#)

Предупреждения

Авторство

(c) RC Module Inc.

## 8.2.2 Макросы

### 8.2.2.1 ALL\_FPU

```
#define ALL_FPU(  
    instr )
```

Макроопределение:

```
"fpu 0 " instr "\n\t" \  
                                "fpu 1 " instr "\n\t" \  
                                "fpu 2 " instr "\n\t" \  
                                "fpu 3 " instr "\n\t"
```

См. определение в файле `nmblas_sgemv.h` строка 42

## 8.3 Файл D:/GIT/nmpp/include/nmtype.h

Классы

- struct `nmreg`
- struct `int15in16x4`
- struct `s_int32x2`
- struct `int31in32x2`
- struct `int30in32x2`
- struct `v16nm4s`
- struct `v4nm8s`
- struct `s_v8nm8s`
- struct `s_v16nm8s`
- struct `s_v4nm16s`
- struct `s_v8nm16s`
- struct `s_v16nm16s`
- struct `s_v2nm32s`
- struct `s_v4nm32s`
- struct `s_v8nm32s`
- struct `s_v16nm32s`
- struct `s_v16nm4u`
- struct `s_v4nm8u`
- struct `s_v8nm8u`
- struct `s_v16nm8u`
- struct `s_v4nm16u`
- struct `s_v8nm16u`
- struct `s_v16nm16u`
- struct `s_v2nm32u`
- struct `s_v4nm32u`



- struct [s\\_v8nm32u](#)
- struct [s\\_v16nm32u](#)
- struct [nm16sc](#)
- struct [s\\_nm32sc](#)
- struct [s\\_v2nm32f](#)
- struct [s\\_v4nm32f](#)
- struct [s\\_v4nm64f](#)
- struct [s\\_nm32fc](#)
- struct [s\\_nm32fcr](#)
- struct [s\\_nm64sc](#)
- struct [Tmp2BuffSpec](#)

## Макросы

- `#define MEM_LOCAL 0`
- `#define MEM_GLOBAL 1`
- `#define HEAP0 0`
- `#define HEAP1 1`
- `#define HEAP2 2`
- `#define HEAP3 3`
- `#define sizeof32(t) sizeof(t)`
- `#define VEC_NM1(X) unsigned data[(X)/32];`
- `#define VEC_NM2U(X) unsigned data[(X)/16];`
- `#define VEC_NM2S(X) int data[(X)/16];`
- `#define VEC_NM4U(X)`
- `#define VEC_NM4S(X) int data[(X)/8];`
- `#define VEC_NM8U(X) unsigned int data[(X)/4];`
- `#define VEC_NM8S(X)`
- `#define VEC_NM16U(X)`
- `#define VEC_NM16S(X)`
- `#define VEC_NM32U(X) nm32u data[(X)];`
- `#define VEC_NM32S(X)`
- `#define CAPACITY\_nm64s 1`
- `#define CAPACITY\_nm32s 2`
- `#define CAPACITY\_nm16s 4`
- `#define CAPACITY\_nm8s 8`
- `#define CAPACITY\_nm4s 16`
- `#define CAPACITY\_nm2s 32`
- `#define CAPACITY\_nm1 64`
- `#define NM16S(vec, size) nm64s vec##__64s[size/4]; nm16s* vec=(nm16s*)vec##__64s`
- `#define NM32S(vec, size) nm64s vec##__64s[size/2]; nm32s* vec=(nm32s*)vec##__64s`
- `#define __INLINE__ static inline`
- `#define NM16Sx4(x0, x1, x2, x3)`
- `#define NM32Sx2(x0, x1)`
- `#define NM32Sx4(x0, x1, x2, x3) NM32Sx2(x0,x1),NM32Sx2(x2,x3)`

## Определения типов

- typedef signed long long INT64
- typedef unsigned long long UINT64
- typedef void nm1
- typedef void nm2s
- typedef void nm4s
- typedef void nm8s
- typedef signed char nm8s7b
- typedef void nm16s
- typedef nm16s nm16s15b
- typedef int nm32s
- typedef struct s\_int32x2 int32x2
- typedef int nm32s31b
- typedef int nm32s30b
- typedef long long nm64s
- typedef nm64s nm64s63b
- typedef void nm2u
- typedef void nm4u
- typedef nm4u nm4u3b
- typedef void nm8u
- typedef unsigned char nm8u7b
- typedef void nm16u
- typedef nm16u nm16u15b
- typedef unsigned int nm32u
- typedef unsigned int nm32u31b
- typedef unsigned long long nm64u
- typedef int int1b
- typedef int int2b
- typedef int int3b
- typedef int int4b
- typedef int int7b
- typedef int int8b
- typedef int int15b
- typedef int int16b
- typedef int int30b
- typedef int int31b
- typedef int int32b
- typedef INT64 int63b
- typedef INT64 int64b
- typedef unsigned int uint1b
- typedef unsigned int uint2b
- typedef unsigned int uint3b
- typedef unsigned int uint4b
- typedef unsigned int uint7b
- typedef unsigned int uint8b
- typedef unsigned int uint15b
- typedef unsigned int uint16b
- typedef unsigned int uint31b
- typedef unsigned int uint32b
- typedef UINT64 uint63b
- typedef nm64u uint64b
- typedef struct s\_v8nm8s v8nm8s
- typedef struct s\_v16nm8s v16nm8s
- typedef struct s\_v4nm16s v4nm16s

- typedef struct s\_v8nm16s v8nm16s
- typedef struct s\_v16nm16s v16nm16s
- typedef struct s\_v2nm32s v2nm32s
- typedef struct s\_v4nm32s v4nm32s
- typedef struct s\_v8nm32s v8nm32s
- typedef struct s\_v16nm32s v16nm32s
- typedef v16nm8s v16nm8s7b
- typedef struct s\_v16nm4u v16nm4u
- typedef struct s\_v4nm8u v4nm8u
- typedef struct s\_v8nm8u v8nm8u
- typedef struct s\_v16nm8u v16nm8u
- typedef struct s\_v4nm16u v4nm16u
- typedef struct s\_v8nm16u v8nm16u
- typedef struct s\_v16nm16u v16nm16u
- typedef struct s\_v2nm32u v2nm32u
- typedef struct s\_v4nm32u v4nm32u
- typedef struct s\_v8nm32u v8nm32u
- typedef struct s\_v16nm32u v16nm32u
- typedef v16nm4u v16nm4b3u
- typedef struct s\_nm32sc nm32sc
- typedef struct s\_v2nm32f v2nm32f
- typedef struct s\_v4nm32f v4nm32f
- typedef struct s\_v4nm64f v4nm64f
- typedef struct s\_nm32fc nm32fc
- typedef struct s\_nm32fcr nm32fcr
- typedef float nm32f
- typedef double nm64f
- typedef struct s\_nm64sc nm64sc
- typedef unsigned long long uint64
- typedef unsigned int uint32
- typedef uint64 fifo64
- typedef uint32 fseq32
- typedef uint64 seq64
- typedef uint64 fseq64

### 8.3.1 Подробное описание

Файл, содержащий определения типов упакованных данных.

### 8.3.2 Макросы

#### 8.3.2.1 CAPACITY\_nm64s

```
#define CAPACITY_nm64s 1
```

•

См. определение в файле nmtype.h строка 1392

## 8.3.2.2 NM16Sx4

```
#define NM16Sx4(
    x0,
    x1,
    x2,
    x3 )
```

Макроопределение:

```
((long long)(x3&0xFFFFF1)<<48)| \
                                     ((long long)(x2&0xFFFFF1)<<32)| \
                                     ((long long)(x1&0xFFFFF1)<<16)| \
                                     ((long long)(x0&0xFFFFF1))
```

См. определение в файле nmtype.h строка 1432

## 8.3.2.3 NM32Sx2

```
#define NM32Sx2(
    x0,
    x1 )
```

Макроопределение:

```
((long long)(unsigned(x1)<<32)| \
                                     ((long long)(unsigned(x0)))
```

См. определение в файле nmtype.h строка 1436

## 8.3.2.4 VEC\_NM16S

```
#define VEC_NM16S(
    X )
```

Макроопределение:

```
int data[(X)/2]; \
void set(int i, int val){ ((unsigned short*)data)[i]=val;} \
short operator[] (int index){ return ((unsigned short*)data)[index];}
```

См. определение в файле nmtype.h строка 934

## 8.3.2.5 VEC\_NM16U

```
#define VEC_NM16U(
    X )
```

Макроопределение:

```
unsigned int data[(X)/2]; \
void set(int i, int val){ ((unsigned short*)data)[i]=val;} \
unsigned short operator[] (int index){ return ((unsigned short*)data)[index];}
```

См. определение в файле nmtype.h строка 929

## 8.3.2.6 VEC\_NM32S

```
#define VEC_NM32S(
    X )
```

Макроопределение:

```
int data[(X)]; \
void set(int i, int val){ ((int*)data)[i]=val;} \
int operator[] (int index){ return ((int*)data)[index];}
```

См. определение в файле nmtype.h строка 940

## 8.3.2.7 VEC\_NM4U

```
#define VEC_NM4U(
    X )
```

Макроопределение:

```
unsigned data[(X)/8]; \
uint4b get(int nIndex){return nmget((nm4u*)data,nIndex); } \
uint4b operator[] (int index){ return nmget((nm4u*)data,index);}
```

См. определение в файле nmtype.h строка 916

## 8.3.2.8 VEC\_NM8S

```
#define VEC_NM8S(
    X )
```

Макроопределение:

```
int data[(X)/4]; \
void set(int i, int val){ ((char*)data)[i]=val;} \
char operator[] (int index){ return ((char*)data)[index];}
```

См. определение в файле nmtype.h строка 924



# Предметный указатель

- Арифметические действия, [198](#)
- Арифметические операции, [178](#), [361](#), [362](#)
- Базовые регистровые функции библиотеки, [888](#)
- Быстрое преобразование Фурье, [320](#), [321](#)
  - `nmppsFFTFree_32fcr`, [322](#)
- Блочное переупорядочивание, [191](#)
- Целевые функции, [894](#)
- Целочисленные функции, [876](#)
- Экспонента, [371](#)
- Элементарные функции, [892](#)
- Элементарные математические функции, [368](#)
- Функции деинтерлейсинга, [179](#)
  - `nmppiDeinterlaceBlend`, [179](#)
  - `nmppiDeinterlaceSplit`, [179](#)
- Функции графического вывода текста, [202](#)
  - `hex2ascii`, [202](#)
  - `nmppiPrint8x15`, [202](#)
- Функции обработки изображений, [885](#)
- Функции обработки сигналов, [883](#), [884](#)
- Функции поддержки, [201](#), [286](#), [358](#)
- Функции с плавающей точкой, [877](#)
- Функция деления векторов, [370](#)
- Инициализация, [174](#)
- Инициализация и копирование, [200](#), [204](#), [359](#), [360](#)
- Изменение размеров, [319](#)
- КИХ-фильтрация, [181](#), [327](#)
- Логические и бинарные операции, [363](#)
- Масочная фильтрация, [199](#), [318](#)
- Матричные функции, [881](#), [882](#)
- Натуральный логарифм, [372](#)
- Обращение матрицы, [288](#)
- Операции сравнения, [364](#), [366](#)
- Переупорядочивание и сортировка, [367](#)
- Переупорядочивание изображений, [190](#)
- Скалярные функции, [886](#)
- Свертка, [317](#)
- Типы данных, [878](#)
- Типы скалярных данных, [351](#)
  - `int15b`, [351](#)
  - `int16b`, [352](#)
  - `int1b`, [352](#)
  - `int2b`, [352](#)
  - `int30b`, [352](#)
  - `int31b`, [353](#)
  - `int32b`, [353](#)
  - `int3b`, [353](#)
  - `int4b`, [353](#)
  - `int63b`, [354](#)
  - `int64b`, [354](#)
  - `int7b`, [354](#)
  - `int8b`, [354](#)
  - `uint15b`, [355](#)
  - `uint16b`, [355](#)
  - `uint1b`, [355](#)
  - `uint2b`, [355](#)
  - `uint31b`, [356](#)
  - `uint32b`, [356](#)
  - `uint3b`, [356](#)
  - `uint4b`, [356](#)
  - `uint63b`, [357](#)
  - `uint64b`, [357](#)
  - `uint7b`, [357](#)
  - `uint8b`, [357](#)
- Типы векторных данных, [338](#)
  - `nm1`, [340](#)
  - `nm16s`, [340](#)
  - `nm16s15b`, [340](#)
  - `nm16u`, [341](#)
  - `nm16u15b`, [341](#)
  - `nm2s`, [341](#)
  - `nm2u`, [342](#)
  - `nm32s`, [342](#)
  - `nm32s30b`, [342](#)
  - `nm32s31b`, [343](#)
  - `nm32u`, [343](#)
  - `nm32u31b`, [343](#)
  - `nm4s`, [344](#)
  - `nm4u`, [344](#)
  - `nm4u3b`, [344](#)
  - `nm64s`, [345](#)
  - `nm64s63b`, [345](#)
  - `nm64u`, [345](#)
  - `nm8s`, [346](#)
  - `nm8s7b`, [346](#)
  - `nm8u`, [346](#)
  - `nm8u7b`, [347](#)
  - `v16nm16s`, [347](#)
  - `v16nm16u`, [347](#)
  - `v16nm32s`, [347](#)
  - `v16nm32u`, [347](#)
  - `v16nm4b3u`, [348](#)
  - `v16nm4u`, [348](#)
  - `v16nm8s`, [348](#)
  - `v16nm8s7b`, [348](#)
  - `v16nm8u`, [348](#)
  - `v2nm32s`, [348](#)
  - `v2nm32u`, [349](#)
  - `v4nm16s`, [349](#)

- v4nm16u, [349](#)
- v4nm32s, [349](#)
- v4nm32u, [349](#)
- v4nm8u, [349](#)
- v8nm16s, [349](#)
- v8nm16u, [349](#)
- v8nm32s, [350](#)
- v8nm32u, [350](#)
- v8nm8s, [350](#)
- v8nm8u, [350](#)
- Тригонометрические функции, [369](#)
- Векторные функции, [879](#), [880](#)
- Векторно-матричные операции, [287](#)
- Возведение в степень, [373](#)
- функции взвешенного суммирования, [893](#)
- контроль переполнения, [890](#)
  - GetVec, [890](#)
  - operator<<, [891](#)
- ALL\_FPU
  - nmblas\_sgemm.h, [1014](#)
- AllocateMaxAvail
  - C\_Heap, [965](#)
- BLASS-LEVEL1, [82](#)
- BLASS-LEVEL3, [81](#)
- C\_2DSubPixelMinPosition, [961](#)
- C\_2DTrigSubPixelMinPosition, [961](#)
- C\_Allocator32, [962](#)
- C\_BoxImg< T >, [963](#)
- C\_BoxVec< T >, [963](#)
- C\_Heap, [964](#)
  - AllocateMaxAvail, [965](#)
- C\_Img< T >, [965](#)
- C\_MultiHeap, [966](#)
- C\_PlessyCornerDetector, [967](#)
- C\_PlessyCornerDetector\_16s, [968](#)
- C\_PlessyCornerDetector\_32f, [969](#)
- C\_RingBufferRemote< T >, [970](#)
- CAPACITY\_nm64s
  - nmtype.h, [1017](#)
- D:/GIT/nmpp/include/nmblas.h, [1011](#)
- D:/GIT/nmpp/include/nmblas/nmblas\_sgemm.h, [1012](#)
- D:/GIT/nmpp/include/nmtype.h, [1014](#)
- DFT-8, [27](#)
  - nmppsDFT8Fwd\_32fcr, [27](#)
- ds\_struct, [971](#)
- EnterHardMode, [972](#)
- FFT-1024, [44](#), [299](#)
  - FFT\_Fwd1024, [299](#)
  - nmppsFFT1024Fwd\_32fcr, [44](#)
- FFT-128, [35](#)
  - nmppsFFT128Fwd\_32fcr, [35](#)
- FFT-16, [28](#)
- FFT-2048, [47](#), [304](#)
  - FFT\_Fwd2048, [304](#)
  - nmppsFFT2048Fwd\_32fcr, [47](#)
- FFT-256, [38](#), [289](#)
  - FFT\_Fwd256, [289](#)
  - nmppsFFT256Fwd\_32fcr, [38](#)
- FFT-32, [29](#)
  - nmppsFFT32Fwd\_32fcr, [29](#)
- FFT-4096, [50](#), [309](#)
  - FFT\_Fwd4096, [309](#)
  - nmppsFFT4096Fwd\_32fcr, [50](#)
- FFT-512, [41](#), [294](#)
  - FFT\_Fwd512, [294](#)
  - nmppsFFT512Fwd\_32fcr, [41](#)
- FFT-64, [32](#)
  - nmppsFFT64Fwd\_32fcr, [32](#)
- FFT-8192, [313](#)
  - FFT\_Fwd8192, [313](#)
- FFT-Common, [77](#)
- FFT\_Fwd1024
  - FFT-1024, [299](#)
- FFT\_Fwd2048
  - FFT-2048, [304](#)
- FFT\_Fwd256
  - FFT-256, [289](#)
- FFT\_Fwd4096
  - FFT-4096, [309](#)
- FFT\_Fwd512
  - FFT-512, [294](#)
- FFT\_Fwd8192
  - FFT-8192, [313](#)
- FFT\_Inv1024
  - IFFT-1024, [301](#)
- FFT\_Inv2048
  - IFFT-2048, [306](#)
- FFT\_Inv256
  - IFFT-256, [291](#)
- FFT\_Inv4096
  - IFFT-4096, [311](#)
- FFT\_Inv512
  - IFFT-512, [296](#)
- FFT\_Inv8192
  - IFFT-8192, [315](#)
- FFTFwdInitAlloc, [25](#)
- FFTInvInitAlloc, [26](#)
- Fix-point 32, [177](#)
- Fix-point 64, [176](#)
- FloodFill8
  - Floodfill, [182](#)
- Floodfill, [182](#)
  - FloodFill8, [182](#)
- GetVec
  - контроль переполнения, [890](#)
- hex2ascii
  - Функции графического вывода текста, [202](#)
- I\_2DSubPixelMinPosition, [972](#)
- I\_PlessyCornerDetector, [973](#)



- IDFT-8, [52](#)
- IFFT-1024, [69](#), [301](#)
  - FFT\_Inv1024, [301](#)
  - nmppsFFT1024Inv\_32fcr, [69](#)
- IFFT-128, [60](#)
  - nmppsFFT128Inv\_32fcr, [60](#)
- IFFT-16, [53](#)
- IFFT-2048, [72](#), [306](#)
  - FFT\_Inv2048, [306](#)
  - nmppsFFT2048Inv\_32fcr, [72](#)
- IFFT-256, [63](#), [291](#)
  - FFT\_Inv256, [291](#)
  - nmppsFFT256Inv\_32fcr, [63](#)
- IFFT-32, [54](#)
  - nmppsFFT32Inv\_32fcr, [54](#)
- IFFT-4096, [75](#), [311](#)
  - FFT\_Inv4096, [311](#)
  - nmppsFFT4096Inv\_32fcr, [75](#)
- IFFT-512, [66](#), [296](#)
  - FFT\_Inv512, [296](#)
  - nmppsFFT512Inv\_32fcr, [66](#)
- IFFT-64, [57](#)
  - nmppsFFT64Inv\_32fcr, [57](#)
- IFFT-8192, [315](#)
  - FFT\_Inv8192, [315](#)
- IFFT-Common, [78](#)
- im
  - s\_nm64sc, [993](#)
- int15b
  - Типы скалярных данных, [351](#)
- int15in16x4, [973](#)
- int16b
  - Типы скалярных данных, [352](#)
- int1b
  - Типы скалярных данных, [352](#)
- int2b
  - Типы скалярных данных, [352](#)
- int30b
  - Типы скалярных данных, [352](#)
- int30in32x2, [973](#)
- int31b
  - Типы скалярных данных, [353](#)
- int31in32x2, [974](#)
- int32b
  - Типы скалярных данных, [353](#)
- int3b
  - Типы скалярных данных, [353](#)
- int4b
  - Типы скалярных данных, [353](#)
- int63b
  - Типы скалярных данных, [354](#)
- int64b
  - Типы скалярных данных, [354](#)
- int7b
  - Типы скалярных данных, [354](#)
- int8b
  - Типы скалярных данных, [354](#)
- Integer operations, [175](#)
- MTR\_Addr, [259](#)
  - MTR\_Addr\_1, [259](#)
  - MTR\_Addr\_16s, [260](#)
  - MTR\_Addr\_16u, [261](#)
  - MTR\_Addr\_2s, [261](#)
  - MTR\_Addr\_2u, [262](#)
  - MTR\_Addr\_32s, [262](#)
  - MTR\_Addr\_32u, [263](#)
  - MTR\_Addr\_4s, [264](#)
  - MTR\_Addr\_4u, [264](#)
  - MTR\_Addr\_64s, [265](#)
  - MTR\_Addr\_64u, [265](#)
  - MTR\_Addr\_8s, [266](#)
  - MTR\_Addr\_8u, [266](#)
- MTR\_Addr\_1
  - MTR\_Addr, [259](#)
- MTR\_Addr\_16s
  - MTR\_Addr, [260](#)
- MTR\_Addr\_16u
  - MTR\_Addr, [261](#)
- MTR\_Addr\_2s
  - MTR\_Addr, [261](#)
- MTR\_Addr\_2u
  - MTR\_Addr, [262](#)
- MTR\_Addr\_32s
  - MTR\_Addr, [262](#)
- MTR\_Addr\_32u
  - MTR\_Addr, [263](#)
- MTR\_Addr\_4s
  - MTR\_Addr, [264](#)
- MTR\_Addr\_4u
  - MTR\_Addr, [264](#)
- MTR\_Addr\_64s
  - MTR\_Addr, [265](#)
- MTR\_Addr\_64u
  - MTR\_Addr, [265](#)
- MTR\_Addr\_8s
  - MTR\_Addr, [266](#)
- MTR\_Addr\_8u
  - MTR\_Addr, [266](#)
- MTR\_Copy, [216](#)
  - nmppmCopy\_1, [217](#)
  - nmppmCopy\_16s, [218](#)
  - nmppmCopy\_2s, [219](#)
  - nmppmCopy\_32s, [220](#)
  - nmppmCopy\_4s, [222](#)
  - nmppmCopy\_64s, [223](#)
  - nmppmCopy\_8s, [224](#)
- MTR\_Copyau, [213](#)
  - nmppmCopyau\_8s, [214](#)
- MTR\_Free, [258](#)
- MTR\_GetVal, [277](#)
  - MTR\_GetVal\_1, [277](#)
  - MTR\_GetVal\_16s, [278](#)
  - MTR\_GetVal\_16u, [279](#)
  - MTR\_GetVal\_2s, [279](#)
  - MTR\_GetVal\_2u, [280](#)
  - MTR\_GetVal\_32s, [280](#)

- MTR\_GetVal\_32u, [281](#)
- MTR\_GetVal\_4s, [282](#)
- MTR\_GetVal\_4u, [282](#)
- MTR\_GetVal\_64s, [283](#)
- MTR\_GetVal\_64u, [283](#)
- MTR\_GetVal\_8s, [284](#)
- MTR\_GetVal\_8u, [285](#)
- MTR\_GetVal\_1
  - MTR\_GetVal, [277](#)
- MTR\_GetVal\_16s
  - MTR\_GetVal, [278](#)
- MTR\_GetVal\_16u
  - MTR\_GetVal, [279](#)
- MTR\_GetVal\_2s
  - MTR\_GetVal, [279](#)
- MTR\_GetVal\_2u
  - MTR\_GetVal, [280](#)
- MTR\_GetVal\_32s
  - MTR\_GetVal, [280](#)
- MTR\_GetVal\_32u
  - MTR\_GetVal, [281](#)
- MTR\_GetVal\_4s
  - MTR\_GetVal, [282](#)
- MTR\_GetVal\_4u
  - MTR\_GetVal, [282](#)
- MTR\_GetVal\_64s
  - MTR\_GetVal, [283](#)
- MTR\_GetVal\_64u
  - MTR\_GetVal, [283](#)
- MTR\_GetVal\_8s
  - MTR\_GetVal, [284](#)
- MTR\_GetVal\_8u
  - MTR\_GetVal, [285](#)
- MTR\_Malloc, [257](#)
- MTR\_ProdUnitV, [256](#)
- MTR\_SetVal, [268](#)
  - MTR\_SetVal\_1, [268](#)
  - MTR\_SetVal\_16s, [269](#)
  - MTR\_SetVal\_16u, [270](#)
  - MTR\_SetVal\_2s, [270](#)
  - MTR\_SetVal\_2u, [271](#)
  - MTR\_SetVal\_32s, [271](#)
  - MTR\_SetVal\_32u, [272](#)
  - MTR\_SetVal\_4s, [273](#)
  - MTR\_SetVal\_4u, [273](#)
  - MTR\_SetVal\_64s, [274](#)
  - MTR\_SetVal\_64u, [274](#)
  - MTR\_SetVal\_8s, [275](#)
  - MTR\_SetVal\_8u, [276](#)
- MTR\_SetVal\_1
  - MTR\_SetVal, [268](#)
- MTR\_SetVal\_16s
  - MTR\_SetVal, [269](#)
- MTR\_SetVal\_16u
  - MTR\_SetVal, [270](#)
- MTR\_SetVal\_2s
  - MTR\_SetVal, [270](#)
- MTR\_SetVal\_2u
  - MTR\_SetVal, [271](#)
- MTR\_SetVal\_32s, [271](#)
- MTR\_SetVal\_32u, [272](#)
- MTR\_SetVal\_4s, [273](#)
- MTR\_SetVal\_4u, [273](#)
- MTR\_SetVal\_64s, [274](#)
- MTR\_SetVal\_64u, [274](#)
- MTR\_SetVal\_8s, [275](#)
- MTR\_SetVal\_8u, [276](#)
- MTR\_SetVal, [271](#)
- MTR\_SetVal\_32s, [271](#)
- MTR\_SetVal\_32u, [272](#)
- MTR\_SetVal\_4s, [273](#)
- MTR\_SetVal\_4u, [273](#)
- MTR\_SetVal\_64s, [274](#)
- MTR\_SetVal\_64u, [274](#)
- MTR\_SetVal\_8s, [275](#)
- MTR\_SetVal\_8u, [276](#)
- MTR\_fpResolve\_Gauss, [226](#)
- MTR\_fpResolve\_PivotGauss, [227](#)
- mtr< T >, [974](#)
- NM16Sx4
  - nmtypes.h, [1017](#)
- NM32Sx2
  - nmtypes.h, [1018](#)
- nm1
  - Типы векторных данных, [340](#)
- nm16s
  - Типы векторных данных, [340](#)
- nm16s15b
  - Типы векторных данных, [340](#)
- nm16sc, [976](#)
- nm16u
  - Типы векторных данных, [341](#)
- nm16u15b
  - Типы векторных данных, [341](#)
- nm2s
  - Типы векторных данных, [341](#)
- nm2u
  - Типы векторных данных, [342](#)
- nm32s
  - Типы векторных данных, [342](#)
- nm32s30b
  - Типы векторных данных, [342](#)
- nm32s31b
  - Типы векторных данных, [343](#)
- nm32u
  - Типы векторных данных, [343](#)
- nm32u31b
  - Типы векторных данных, [343](#)
- nm4s
  - Типы векторных данных, [344](#)
- nm4u
  - Типы векторных данных, [344](#)
- nm4u3b
  - Типы векторных данных, [344](#)
- nm64s
  - Типы векторных данных, [345](#)
- nm64s63b

Типы векторных данных, 345  
 nm64u  
   Типы векторных данных, 345  
 nm8s  
   Типы векторных данных, 346  
 nm8s7b  
   Типы векторных данных, 346  
 nm8u  
   Типы векторных данных, 346  
 nm8u7b  
   Типы векторных данных, 347  
 nmblas, 889  
 nmblas\_dasum, 84  
   nmblas\_dasum, 84  
 nmblas\_daxpy, 86  
   nmblas\_daxpy, 86  
 nmblas\_dcopy, 88  
   nmblas\_dcopy, 88  
 nmblas\_ddot, 90  
   nmblas\_ddot, 90  
 nmblas\_dgemv, 129  
 nmblas\_dnrm2, 92  
   nmblas\_dnrm2, 92  
 nmblas\_drot, 94  
   nmblas\_drot, 94  
 nmblas\_drotg, 96  
 nmblas\_drotm, 97  
 nmblas\_dscal, 98  
   nmblas\_dscal, 98  
 nmblas\_dsdot, 100  
   nmblas\_dsdot, 100  
 nmblas\_dswap, 102  
 nmblas\_dznrm2, 103  
 nmblas\_idamax, 104  
   nmblas\_idamax, 104  
 nmblas\_isamax, 106  
   nmblas\_isamax, 106  
 nmblas\_sasum, 108  
   nmblas\_sasum, 108  
 nmblas\_saxpy, 110  
   nmblas\_saxpy, 110  
 nmblas\_scnrm2, 112  
   nmblas\_scnrm2, 112  
 nmblas\_scopy, 114  
   nmblas\_scopy, 114  
 nmblas\_sdot, 116  
   nmblas\_sdot, 116  
 nmblas\_sdsdot, 118  
 nmblas\_sgemm.h  
   ALL\_FPU, 1014  
 nmblas\_sgemv, 127  
 nmblas\_snrm2, 119  
   nmblas\_snrm2, 119  
 nmblas\_srot, 121  
 nmblas\_srotm, 122  
 nmblas\_sscal, 123  
   nmblas\_sscal, 123  
 nmblas\_sswap, 125

  nmblas\_sswap, 125  
 nmchar, 976  
 nmchar1D< N >, 977  
 nmchar2D< Y, X >, 977  
 nmintpack< T >, 978  
 nmmtr< T >, 978  
 nmppMerge, 873  
 nmppSplit, 872  
 nmppSplit\_32fer, 874  
 nmppSplitTmp, 871  
 nmppcDivC, 131  
 nmppcDoubleToFix32, 139  
 nmppcDoubleToFix64, 161  
 nmppcFix32ToDouble, 140  
   nmppcFix32ToDouble, 140  
 nmppcFix64Exp01, 169  
   nmppcFix64Exp01, 169  
 nmppcFix64ToDouble, 162  
   nmppcFix64ToDouble, 162  
 nmppcFixArcTan32, 137  
   nmppcFixArcTan32, 137  
 nmppcFixArcTan64, 168  
   nmppcFixArcTan64, 168  
 nmppcFixDiv32  
   nmppcFixMul32, 143  
 nmppcFixDiv64, 164  
   nmppcFixDiv64, 164  
 nmppcFixDivMod32, 158  
   nmppcFixDivMod32, 158  
 nmppcFixExp32, 133  
   nmppcFixExp32, 133  
 nmppcFixInv32, 148  
   nmppcFixInv32, 148  
 nmppcFixMul32, 143  
   nmppcFixDiv32, 143  
   nmppcFixMul32, 145  
 nmppcFixSinCos32, 135  
   nmppcFixSinCos32, 135  
 nmppcFixSinCos64, 166  
   nmppcFixSinCos64, 166  
 nmppcFixSqrt32, 142  
 nmppcFixSqrt64, 160  
 nmppcProdC, 132  
 nmppcRandMinMaxDiv  
   nmppsRand, 171  
 nmppcSqrt, 173  
 nmppcTblFixArcCos32, 152  
   nmppcTblFixArcCos32, 152  
 nmppcTblFixArcSin32, 150  
   nmppcTblFixArcSin32, 150  
 nmppcTblFixCos32, 154  
   nmppcTblFixCos32, 154  
 nmppcTblFixSin32, 156  
   nmppcTblFixSin32, 156  
 nmppiConvert, 188  
 nmppiDeinterlaceBlend  
   Функции деинтерлейсинга, 179  
 nmppiDeinterlaceSplit

- Функции деинтерлейсинга, 179
- nmppiFree, 196
- nmppiMergeFromBlocks, 194
  - nmppiMergeFromBlocks8x8, 194
- nmppiMergeFromBlocks8x8
  - nmppiMergeFromBlocks, 194
- nmppiPrint8x15
  - Функции графического вывода текста, 202
- nmppiRGB32ToGray, 189
- nmppiRelease, 197
- nmppiSplitIntoBlocks, 192
  - nmppiSplitIntoBlocks8x8, 192
- nmppiSplitIntoBlocks8x8
  - nmppiSplitIntoBlocks, 192
- nmppmCopy\_1
  - MTR\_Copy, 217
- nmppmCopy\_16s
  - MTR\_Copy, 218
- nmppmCopy\_2s
  - MTR\_Copy, 219
- nmppmCopy\_32s
  - MTR\_Copy, 220
- nmppmCopy\_4s
  - MTR\_Copy, 222
- nmppmCopy\_64s
  - MTR\_Copy, 223
- nmppmCopy\_8s
  - MTR\_Copy, 224
- nmppmCopyau\_8s
  - MTR\_Copyau, 214
- nmppmCopyua, 205
  - nmppmCopyua\_16s, 205
  - nmppmCopyua\_32s, 208
  - nmppmCopyua\_8s, 210
- nmppmCopyua\_16s
  - nmppmCopyua, 205
- nmppmCopyua\_32s
  - nmppmCopyua, 208
- nmppmCopyua\_8s
  - nmppmCopyua, 210
- nmppmLUDecomp, 228
- nmppmLUResolve, 229
- nmppmMul\_mm, 230
  - nmppmMul\_mm\_16s32s, 231
  - nmppmMul\_mm\_16s64s, 233
  - nmppmMul\_mm\_2s64s, 235
  - nmppmMul\_mm\_32s32s, 237
  - nmppmMul\_mm\_32s64s, 239
  - nmppmMul\_mm\_4s64s, 241
  - nmppmMul\_mm\_64s64s, 244
  - nmppmMul\_mm\_8s64s, 246
- nmppmMul\_mm\_16s32s
  - nmppmMul\_mm, 231
- nmppmMul\_mm\_16s64s
  - nmppmMul\_mm, 233
- nmppmMul\_mm\_2s64s
  - nmppmMul\_mm, 235
- nmppmMul\_mm\_32s32s
  - nmppmMul\_mm, 237
- nmppmMul\_mm\_32s64s
  - nmppmMul\_mm, 239
- nmppmMul\_mm\_4s64s
  - nmppmMul\_mm, 241
- nmppmMul\_mm\_64s64s
  - nmppmMul\_mm, 244
- nmppmMul\_mm\_8s64s
  - nmppmMul\_mm, 246
- nmppmMul\_mv\_, 249
  - nmppmMul\_mv\_16s64s, 249
  - nmppmMul\_mv\_32s64s, 251
  - nmppmMul\_mv\_8s64s, 252
- nmppmMul\_mv\_16s64s
  - nmppmMul\_mv\_, 249
- nmppmMul\_mv\_32s64s
  - nmppmMul\_mv\_, 251
- nmppmMul\_mv\_8s64s
  - nmppmMul\_mv\_, 252
- nmppmMul\_mv\_8xH, 254
- nmppmMul\_mv\_\_AddC, 255
- nmppsAbs, 374
  - nmppsAbs\_16s, 374
  - nmppsAbs\_32s, 376
  - nmppsAbs\_4s, 377
  - nmppsAbs\_64s, 378
  - nmppsAbs\_8s, 379
- nmppsAbs1, 381
  - nmppsAbs1\_16s, 381
  - nmppsAbs1\_32s, 383
  - nmppsAbs1\_64s, 384
  - nmppsAbs1\_8s, 385
- nmppsAbs1\_16s
  - nmppsAbs1, 381
- nmppsAbs1\_32s
  - nmppsAbs1, 383
- nmppsAbs1\_64s
  - nmppsAbs1, 384
- nmppsAbs1\_8s
  - nmppsAbs1, 385
- nmppsAbs\_16s
  - nmppsAbs, 374
- nmppsAbs\_32s
  - nmppsAbs, 376
- nmppsAbs\_4s
  - nmppsAbs, 377
- nmppsAbs\_64s
  - nmppsAbs, 378
- nmppsAbs\_8s
  - nmppsAbs, 379
- nmppsAbsDiff, 439
  - nmppsAbsDiff\_16s, 439
  - nmppsAbsDiff\_32s, 441
  - nmppsAbsDiff\_64s, 442
  - nmppsAbsDiff\_8s, 443
- nmppsAbsDiff1, 445
  - nmppsAbsDiff1\_8s, 445
- nmppsAbsDiff1\_8s

- nmppsAbsDiff1, [445](#)
- nmppsAbsDiff\_16s
  - nmppsAbsDiff, [439](#)
- nmppsAbsDiff\_32s
  - nmppsAbsDiff, [441](#)
- nmppsAbsDiff\_64s
  - nmppsAbsDiff, [442](#)
- nmppsAbsDiff\_8s
  - nmppsAbsDiff, [443](#)
- nmppsAdd, [403](#), [411](#)
  - nmppsAdd\_16s, [403](#)
  - nmppsAdd\_32f, [411](#)
  - nmppsAdd\_32s, [405](#)
  - nmppsAdd\_4s, [406](#)
  - nmppsAdd\_64s, [407](#)
  - nmppsAdd\_8s, [409](#)
- nmppsAdd\_16s
  - nmppsAdd, [403](#)
- nmppsAdd\_32f
  - nmppsAdd, [411](#)
- nmppsAdd\_32s
  - nmppsAdd, [405](#)
- nmppsAdd\_4s
  - nmppsAdd, [406](#)
- nmppsAdd\_64s
  - nmppsAdd, [407](#)
- nmppsAdd\_8s
  - nmppsAdd, [409](#)
- nmppsAdd\_AddC\_32s
  - nmppsAdd\_AddC, [414](#)
- nmppsAdd\_AddC, [414](#)
  - nmppsAdd\_AddC\_32s, [414](#)
- nmppsAddC\_16s
  - nmppsAddC, [393](#)
- nmppsAddC\_32fcr
  - nmppsAddC, [401](#)
- nmppsAddC\_32s
  - nmppsAddC, [395](#)
- nmppsAddC\_64s
  - nmppsAddC, [396](#)
- nmppsAddC\_8s
  - nmppsAddC, [397](#)
- nmppsAddC\_p, [399](#)
  - nmppsAddC\_p64s, [399](#)
- nmppsAddC\_p64s
  - nmppsAddC\_p, [399](#)
- nmppsAddC, [393](#), [401](#)
  - nmppsAddC\_16s, [393](#)
  - nmppsAddC\_32fcr, [401](#)
  - nmppsAddC\_32s, [395](#)
  - nmppsAddC\_64s, [396](#)
  - nmppsAddC\_8s, [397](#)
- nmppsAddr\_, [828](#)
  - nmppsAddr\_1, [828](#)
  - nmppsAddr\_16s, [829](#)
  - nmppsAddr\_16u, [830](#)
  - nmppsAddr\_2s, [830](#)
  - nmppsAddr\_2u, [831](#)
- nmppsAddr\_32f, [831](#)
- nmppsAddr\_32s, [832](#)
- nmppsAddr\_32u, [832](#)
- nmppsAddr\_4s, [833](#)
- nmppsAddr\_4u, [834](#)
- nmppsAddr\_64f, [834](#)
- nmppsAddr\_64s, [835](#)
- nmppsAddr\_64u, [835](#)
- nmppsAddr\_8s, [836](#)
- nmppsAddr\_8u, [836](#)
- nmppsAddr\_1
  - nmppsAddr\_, [828](#)
- nmppsAddr\_16s
  - nmppsAddr\_, [829](#)
- nmppsAddr\_16u
  - nmppsAddr\_, [830](#)
- nmppsAddr\_2s
  - nmppsAddr\_, [830](#)
- nmppsAddr\_2u
  - nmppsAddr\_, [831](#)
- nmppsAddr\_32f
  - nmppsAddr\_, [831](#)
- nmppsAddr\_32s
  - nmppsAddr\_, [832](#)
- nmppsAddr\_32u
  - nmppsAddr\_, [832](#)
- nmppsAddr\_4s
  - nmppsAddr\_, [833](#)
- nmppsAddr\_4u
  - nmppsAddr\_, [834](#)
- nmppsAddr\_64f
  - nmppsAddr\_, [834](#)
- nmppsAddr\_64s
  - nmppsAddr\_, [835](#)
- nmppsAddr\_64u
  - nmppsAddr\_, [835](#)
- nmppsAddr\_8s
  - nmppsAddr\_, [836](#)
- nmppsAddr\_8u
  - nmppsAddr\_, [836](#)
- nmppsAnd, [536](#)
  - nmppsAnd\_1, [536](#)
  - nmppsAnd\_16u, [538](#)
  - nmppsAnd\_2u, [539](#)
  - nmppsAnd\_32u, [540](#)
  - nmppsAnd\_4u, [542](#)
  - nmppsAnd\_64u, [543](#)
  - nmppsAnd\_8u, [544](#)
- nmppsAnd4V\_, [546](#)
- nmppsAnd\_1
  - nmppsAnd, [536](#)
- nmppsAnd\_16u
  - nmppsAnd, [538](#)
- nmppsAnd\_2u
  - nmppsAnd, [539](#)
- nmppsAnd\_32u
  - nmppsAnd, [540](#)
- nmppsAnd\_4u

- nmppsAnd, [542](#)
- nmppsAnd\_64u
  - nmppsAnd, [543](#)
- nmppsAnd\_8u
  - nmppsAnd, [544](#)
- nmppsAndNotV\_, [547](#)
  - nmppsAndNotV\_64u, [547](#)
- nmppsAndNotV\_64u
  - nmppsAndNotV\_, [547](#)
- nmppsAndC, [535](#)
- nmppsClipCC\_, [785](#)
- nmppsClipConvert\_AddC\_, [792](#)
- nmppsClipPowC\_, [777](#)
  - nmppsClipPowC\_16s, [777](#)
  - nmppsClipPowC\_32s, [779](#)
  - nmppsClipPowC\_64s, [781](#)
  - nmppsClipPowC\_8s, [782](#)
- nmppsClipPowC\_16s
  - nmppsClipPowC\_, [777](#)
- nmppsClipPowC\_32s
  - nmppsClipPowC\_, [779](#)
- nmppsClipPowC\_64s
  - nmppsClipPowC\_, [781](#)
- nmppsClipPowC\_8s
  - nmppsClipPowC\_, [782](#)
- nmppsClipRShiftConvert\_AddC\_, [786](#)
  - nmppsClipRShiftConvertAddC\_16s8s, [786](#)
  - nmppsClipRShiftConvertAddC\_32s8s, [789](#)
- nmppsClipRShiftConvertAddC\_16s8s
  - nmppsClipRShiftConvert\_AddC\_, [786](#)
- nmppsClipRShiftConvertAddC\_32s8s
  - nmppsClipRShiftConvert\_AddC\_, [789](#)
- nmppsCmpEq0, [769](#)
  - nmppsCmpEq0\_16u15b, [770](#)
  - nmppsCmpEq0\_32u31b, [770](#)
  - nmppsCmpEq0\_8u7b, [770](#)
- nmppsCmpEq0\_16u15b
  - nmppsCmpEq0, [770](#)
- nmppsCmpEq0\_32u31b
  - nmppsCmpEq0, [770](#)
- nmppsCmpEq0\_8u7b
  - nmppsCmpEq0, [770](#)
- nmppsCmpEqC\_16u15b
  - nmppsCmpEqC, [794](#)
- nmppsCmpEqC\_4u3b
  - nmppsCmpEqC, [795](#)
- nmppsCmpEqC\_8u7b
  - nmppsCmpEqC, [796](#)
- nmppsCmpEqV\_, [818](#)
- nmppsCmpEqC, [794](#)
  - nmppsCmpEqC\_16u15b, [794](#)
  - nmppsCmpEqC\_4u3b, [795](#)
  - nmppsCmpEqC\_8u7b, [796](#)
- nmppsCmpGtC, [815](#)
- nmppsCmpGteC, [817](#)
- nmppsCmpLt, [823](#)
- nmppsCmpLt0, [761](#)
  - nmppsCmpLt0\_16s, [761](#)
  - nmppsCmpLt0\_32s, [763](#)
  - nmppsCmpLt0\_64s, [764](#)
  - nmppsCmpLt0\_8s, [765](#)
- nmppsCmpLt0\_16s
  - nmppsCmpLt0, [761](#)
- nmppsCmpLt0\_32s
  - nmppsCmpLt0, [763](#)
- nmppsCmpLt0\_64s
  - nmppsCmpLt0, [764](#)
- nmppsCmpLt0\_8s
  - nmppsCmpLt0, [765](#)
- nmppsCmpLtC\_16s15b
  - nmppsCmpLtC, [810](#)
- nmppsCmpLtC\_32s31b
  - nmppsCmpLtC, [812](#)
- nmppsCmpLtC\_64s63b
  - nmppsCmpLtC, [813](#)
- nmppsCmpLtC\_8s7b
  - nmppsCmpLtC, [813](#)
- nmppsCmpLtC, [768](#), [810](#)
  - nmppsCmpLtC\_16s15b, [810](#)
  - nmppsCmpLtC\_32s31b, [812](#)
  - nmppsCmpLtC\_64s63b, [813](#)
  - nmppsCmpLtC\_8s7b, [813](#)
- nmppsCmpLteC, [767](#)
- nmppsCmpMinMaxV\_, [772](#)
  - nmppsCmpMinMaxV\_16s, [773](#)
  - nmppsCmpMinMaxV\_32s, [774](#)
  - nmppsCmpMinMaxV\_8s, [775](#)
- nmppsCmpMinMaxV\_16s
  - nmppsCmpMinMaxV\_, [773](#)
- nmppsCmpMinMaxV\_32s
  - nmppsCmpMinMaxV\_, [774](#)
- nmppsCmpMinMaxV\_8s
  - nmppsCmpMinMaxV\_, [775](#)
- nmppsCmpNe, [819](#)
  - nmppsCmpNe\_16s, [819](#)
  - nmppsCmpNe\_32s, [821](#)
  - nmppsCmpNe\_64s, [821](#)
  - nmppsCmpNe\_8s, [822](#)
- nmppsCmpNe0, [797](#)
  - nmppsCmpNe0\_16s, [797](#)
  - nmppsCmpNe0\_32s, [799](#)
  - nmppsCmpNe0\_64s, [800](#)
  - nmppsCmpNe0\_8s, [801](#)
- nmppsCmpNe0\_16s
  - nmppsCmpNe0, [797](#)
- nmppsCmpNe0\_32s
  - nmppsCmpNe0, [799](#)
- nmppsCmpNe0\_64s
  - nmppsCmpNe0, [800](#)
- nmppsCmpNe0\_8s
  - nmppsCmpNe0, [801](#)
- nmppsCmpNe\_16s
  - nmppsCmpNe, [819](#)
- nmppsCmpNe\_32s
  - nmppsCmpNe, [821](#)
- nmppsCmpNe\_64s

- nmppsCmpNe, [821](#)
- nmppsCmpNe\_8s
  - nmppsCmpNe, [822](#)
- nmppsCmpNeC\_16s
  - nmppsCmpNeC, [803](#)
- nmppsCmpNeC\_32s
  - nmppsCmpNeC, [805](#)
- nmppsCmpNeC\_64s
  - nmppsCmpNeC, [806](#)
- nmppsCmpNeC\_8s
  - nmppsCmpNeC, [807](#)
- nmppsCmpNeC\_flag, [809](#)
- nmppsCmpNeV\_, [824](#)
  - nmppsCmpNeV\_16u, [824](#)
  - nmppsCmpNeV\_8u, [825](#)
- nmppsCmpNeV\_16u
  - nmppsCmpNeV\_, [824](#)
- nmppsCmpNeV\_8u
  - nmppsCmpNeV\_, [825](#)
- nmppsCmpNeC, [803](#)
  - nmppsCmpNeC\_16s, [803](#)
  - nmppsCmpNeC\_32s, [805](#)
  - nmppsCmpNeC\_64s, [806](#)
  - nmppsCmpNeC\_8s, [807](#)
- nmppsConjMul, [477](#)
  - nmppsConjMul\_32fcr, [477](#)
- nmppsConjMul\_32fcr
  - nmppsConjMul, [477](#)
- nmppsConvert, [638](#), [674](#)
  - nmppsConvert\_16s32s, [639](#)
  - nmppsConvert\_16s4s, [640](#)
  - nmppsConvert\_16s64s, [641](#)
  - nmppsConvert\_16s8s, [642](#)
  - nmppsConvert\_16u32u, [643](#)
  - nmppsConvert\_1s2s, [645](#)
  - nmppsConvert\_1u2u, [647](#)
  - nmppsConvert\_1u4u, [648](#)
  - nmppsConvert\_2s1s, [649](#)
  - nmppsConvert\_2s4s, [650](#)
  - nmppsConvert\_2u4u, [651](#)
  - nmppsConvert\_32f32fcr, [674](#)
  - nmppsConvert\_32f64f, [675](#)
  - nmppsConvert\_32s16s, [652](#)
  - nmppsConvert\_32s32f, [677](#)
  - nmppsConvert\_32s32fcr, [678](#)
  - nmppsConvert\_32s64s, [653](#)
  - nmppsConvert\_32s8s, [655](#)
  - nmppsConvert\_32sc32fcr, [679](#)
  - nmppsConvert\_32u32fcr, [680](#)
  - nmppsConvert\_32u64u, [657](#)
  - nmppsConvert\_4s1s, [658](#)
  - nmppsConvert\_4s2s, [659](#)
  - nmppsConvert\_4s8s, [660](#)
  - nmppsConvert\_4u8u, [661](#)
  - nmppsConvert\_64f32f, [681](#)
  - nmppsConvert\_64s16s, [662](#)
  - nmppsConvert\_64s32s, [663](#)
  - nmppsConvert\_8s16s, [664](#)
  - nmppsConvert\_8s32s, [666](#)
  - nmppsConvert\_8s4s, [668](#)
  - nmppsConvert\_8s64s, [669](#)
  - nmppsConvert\_8u16u, [670](#)
  - nmppsConvert\_8u32u, [671](#)
  - nmppsConvert\_8u64u, [672](#)
  - nmppsConvert\_16s32s
    - nmppsConvert, [639](#)
  - nmppsConvert\_16s4s
    - nmppsConvert, [640](#)
  - nmppsConvert\_16s64s
    - nmppsConvert, [641](#)
  - nmppsConvert\_16s8s
    - nmppsConvert, [642](#)
  - nmppsConvert\_16u32u
    - nmppsConvert, [643](#)
  - nmppsConvert\_1s2s
    - nmppsConvert, [645](#)
  - nmppsConvert\_1u2u
    - nmppsConvert, [647](#)
  - nmppsConvert\_1u4u
    - nmppsConvert, [648](#)
  - nmppsConvert\_2s1s
    - nmppsConvert, [649](#)
  - nmppsConvert\_2s4s
    - nmppsConvert, [650](#)
  - nmppsConvert\_2u4u
    - nmppsConvert, [651](#)
  - nmppsConvert\_32f32fcr
    - nmppsConvert, [674](#)
  - nmppsConvert\_32f32s\_ceiling
    - nmppsConvert\_rounding, [683](#)
  - nmppsConvert\_32f32s\_floor
    - nmppsConvert\_rounding, [684](#)
  - nmppsConvert\_32f32s\_rounding
    - nmppsConvert\_rounding, [686](#)
  - nmppsConvert\_32f32s\_truncate
    - nmppsConvert\_rounding, [687](#)
  - nmppsConvert\_32f64f
    - nmppsConvert, [675](#)
  - nmppsConvert\_32s16s
    - nmppsConvert, [652](#)
  - nmppsConvert\_32s32f
    - nmppsConvert, [677](#)
  - nmppsConvert\_32s32fcr
    - nmppsConvert, [678](#)
  - nmppsConvert\_32s64s
    - nmppsConvert, [653](#)
  - nmppsConvert\_32s8s
    - nmppsConvert, [655](#)
  - nmppsConvert\_32sc32fcr
    - nmppsConvert, [679](#)
  - nmppsConvert\_32u32fcr
    - nmppsConvert, [680](#)
  - nmppsConvert\_32u64u
    - nmppsConvert, [657](#)
  - nmppsConvert\_4s1s
    - nmppsConvert, [658](#)

- nmppsConvert\_4s2s
  - nmppsConvert, [659](#)
- nmppsConvert\_4s8s
  - nmppsConvert, [660](#)
- nmppsConvert\_4u8u
  - nmppsConvert, [661](#)
- nmppsConvert\_64f32f
  - nmppsConvert, [681](#)
- nmppsConvert\_64s16s
  - nmppsConvert, [662](#)
- nmppsConvert\_64s32s
  - nmppsConvert, [663](#)
- nmppsConvert\_8s16s
  - nmppsConvert, [664](#)
- nmppsConvert\_8s32s
  - nmppsConvert, [666](#)
- nmppsConvert\_8s4s
  - nmppsConvert, [668](#)
- nmppsConvert\_8s64s
  - nmppsConvert, [669](#)
- nmppsConvert\_8u16u
  - nmppsConvert, [670](#)
- nmppsConvert\_8u32u
  - nmppsConvert, [671](#)
- nmppsConvert\_8u64u
  - nmppsConvert, [672](#)
- nmppsConvert\_rounding, [683](#)
  - nmppsConvert\_32f32s\_ceiling, [683](#)
  - nmppsConvert\_32f32s\_floor, [684](#)
  - nmppsConvert\_32f32s\_rounding, [686](#)
  - nmppsConvert\_32f32s\_truncate, [687](#)
- nmppsConvertRisc, [692](#)
  - nmppsConvertRisc\_32u8u, [692](#)
  - nmppsConvertRisc\_8u32u, [693](#)
- nmppsConvertRisc\_32u8u
  - nmppsConvertRisc, [692](#)
- nmppsConvertRisc\_8u32u
  - nmppsConvertRisc, [693](#)
- nmppsCopy\_, [695](#)
  - nmppsCopy\_16s, [695](#)
  - nmppsCopy\_16u, [697](#)
  - nmppsCopy\_2s, [698](#)
  - nmppsCopy\_32s, [699](#)
  - nmppsCopy\_32u, [700](#)
  - nmppsCopy\_4s, [701](#)
  - nmppsCopy\_4u, [702](#)
  - nmppsCopy\_64s, [703](#)
  - nmppsCopy\_64sc, [704](#)
  - nmppsCopy\_64u, [706](#)
  - nmppsCopy\_8s, [707](#)
  - nmppsCopy\_8u, [708](#)
- nmppsCopy\_16s
  - nmppsCopy\_, [695](#)
- nmppsCopy\_16u
  - nmppsCopy\_, [697](#)
- nmppsCopy\_2s
  - nmppsCopy\_, [698](#)
- nmppsCopy\_32s
  - nmppsCopy\_, [699](#)
- nmppsCopy\_, [699](#)
  - nmppsCopy\_32u, [700](#)
  - nmppsCopy\_4s, [701](#)
  - nmppsCopy\_4u, [702](#)
  - nmppsCopy\_64s, [703](#)
  - nmppsCopy\_64sc, [704](#)
  - nmppsCopy\_64u, [706](#)
  - nmppsCopy\_8s, [707](#)
  - nmppsCopy\_8u, [708](#)
- nmppsCopy\_32u
  - nmppsCopy\_, [700](#)
- nmppsCopy\_4s
  - nmppsCopy\_, [701](#)
- nmppsCopy\_4u
  - nmppsCopy\_, [702](#)
- nmppsCopy\_64s
  - nmppsCopy\_, [703](#)
- nmppsCopy\_64sc
  - nmppsCopy\_, [704](#)
- nmppsCopy\_64u
  - nmppsCopy\_, [706](#)
- nmppsCopy\_8s
  - nmppsCopy\_, [707](#)
- nmppsCopy\_8u
  - nmppsCopy\_, [708](#)
- nmppsCopyua\_, [710](#)
  - nmppsCopyua\_8s, [710](#)
- nmppsCopyua\_8s
  - nmppsCopyua\_, [710](#)
- nmppsDFT8Fwd\_32fcr
  - DFT-8, [27](#)
- nmppsDecimate, [875](#)
- nmppsDisplaceBits, [598](#)
  - nmppsFwdShiftBitstream, [599](#)
- nmppsDivC\_32s
  - nmppsDivC, [497](#)
- nmppsDivC, [496](#)
  - nmppsDivC\_32s, [497](#)
- nmppsDotProd, [514](#)
  - nmppsDotProd\_16s64s, [514](#)
  - nmppsDotProd\_32s64s, [516](#)
  - nmppsDotProd\_64s64s, [517](#)
  - nmppsDotProd\_8s64s, [518](#)
- nmppsDotProd\_16s16sm
  - nmppsDotProd\_sm, [505](#)
- nmppsDotProd\_16s32sm
  - nmppsDotProd\_sm, [507](#)
- nmppsDotProd\_16s64s
  - nmppsDotProd, [514](#)
- nmppsDotProd\_32s32sm
  - nmppsDotProd\_sm, [508](#)
- nmppsDotProd\_32s64s
  - nmppsDotProd, [516](#)
- nmppsDotProd\_64s64s
  - nmppsDotProd, [517](#)
- nmppsDotProd\_8s16sm
  - nmppsDotProd\_sm, [509](#)
- nmppsDotProd\_8s32sm
  - nmppsDotProd\_sm, [511](#)
- nmppsDotProd\_8s64s
  - nmppsDotProd, [518](#)
- nmppsDotProd\_8s8sm
  - nmppsDotProd\_sm, [512](#)
- nmppsDotProd\_sm, [505](#)
  - nmppsDotProd\_16s16sm, [505](#)
  - nmppsDotProd\_16s32sm, [507](#)



- nmppsDotProd\_32s32sm, [508](#)
- nmppsDotProd\_8s16sm, [509](#)
- nmppsDotProd\_8s32sm, [511](#)
- nmppsDotProd\_8s8sm, [512](#)
- nmppsFFT1024Fwd\_32fcr
  - FFT-1024, [44](#)
- nmppsFFT1024Inv\_32fcr
  - IFFT-1024, [69](#)
- nmppsFFT128Fwd\_32fcr
  - FFT-128, [35](#)
- nmppsFFT128Inv\_32fcr
  - IFFT-128, [60](#)
- nmppsFFT2048Fwd\_32fcr
  - FFT-2048, [47](#)
- nmppsFFT2048Inv\_32fcr
  - IFFT-2048, [72](#)
- nmppsFFT256Fwd\_32fcr
  - FFT-256, [38](#)
- nmppsFFT256Inv\_32fcr
  - IFFT-256, [63](#)
- nmppsFFT32Fwd\_32fcr
  - FFT-32, [29](#)
- nmppsFFT32Inv\_32fcr
  - IFFT-32, [54](#)
- nmppsFFT4096Fwd\_32fcr
  - FFT-4096, [50](#)
- nmppsFFT4096Inv\_32fcr
  - IFFT-4096, [75](#)
- nmppsFFT512Fwd\_32fcr
  - FFT-512, [41](#)
- nmppsFFT512Inv\_32fcr
  - IFFT-512, [66](#)
- nmppsFFT64Fwd\_32fcr
  - FFT-64, [32](#)
- nmppsFFT64Inv\_32fcr
  - IFFT-64, [57](#)
- nmppsFFTFree\_32fcr
  - Быстрое преобразование Фурье, [322](#)
- NmppsFFTSpec, [980](#)
- NmppsFFTSpec\_32fcr, [980](#)
- nmppsFIR\_Xs, [328](#)
- nmppsFIRFree, [332](#)
- nmppsFIRGetStateSize\_Xs, [331](#)
- nmppsFIRInit\_Xs, [329](#)
- nmppsFIRInitAlloc\_Xs, [330](#)
- nmppsFirstNonZeroIdx, [739](#)
  - nmppsFirstNonZeroIdx\_32s, [739](#)
- nmppsFirstNonZeroIdx\_32s
  - nmppsFirstNonZeroIdx, [739](#)
- nmppsFirstZeroIdx, [737](#)
  - nmppsFirstZeroIdx\_32s, [737](#)
- nmppsFirstZeroIdx\_32s
  - nmppsFirstZeroIdx, [737](#)
- NmppsFrame\_16s, [981](#)
- NmppsFrame\_16u, [981](#)
- NmppsFrame\_32s, [981](#)
- NmppsFrame\_32u, [982](#)
- NmppsFrame\_64s, [982](#)
- NmppsFrame\_64u, [982](#)
- NmppsFrame\_8s, [983](#)
- NmppsFrame\_8u, [983](#)
- nmppsFree, [80](#)
- nmppsFwdShiftBitstream
  - nmppsDisplaceBits, [599](#)
- nmppsGet\_1
  - nmppsGetVal\_(return), [859](#)
- nmppsGet\_16s
  - nmppsGetVal\_(return), [860](#)
- nmppsGet\_16u
  - nmppsGetVal\_(return), [860](#)
- nmppsGet\_2s
  - nmppsGetVal\_(return), [861](#)
- nmppsGet\_2u
  - nmppsGetVal\_(return), [862](#)
- nmppsGet\_32s
  - nmppsGetVal\_(return), [862](#)
- nmppsGet\_32u
  - nmppsGetVal\_(return), [863](#)
- nmppsGet\_4s
  - nmppsGetVal\_(return), [863](#)
- nmppsGet\_4u
  - nmppsGetVal\_(return), [864](#)
- nmppsGet\_8s
  - nmppsGetVal\_(return), [864](#)
- nmppsGet\_8u
  - nmppsGetVal\_(return), [865](#)
- nmppsGetVal\_, [849](#)
  - nmppsGetVal\_1, [849](#)
  - nmppsGetVal\_16s, [851](#)
  - nmppsGetVal\_16u, [851](#)
  - nmppsGetVal\_2s, [852](#)
  - nmppsGetVal\_2u, [853](#)
  - nmppsGetVal\_32s, [853](#)
  - nmppsGetVal\_32u, [854](#)
  - nmppsGetVal\_4s, [854](#)
  - nmppsGetVal\_4u, [855](#)
  - nmppsGetVal\_64s, [856](#)
  - nmppsGetVal\_64u, [856](#)
  - nmppsGetVal\_8s, [857](#)
  - nmppsGetVal\_8u, [857](#)
- nmppsGetVal\_(return), [859](#)
  - nmppsGet\_1, [859](#)
  - nmppsGet\_16s, [860](#)
  - nmppsGet\_16u, [860](#)
  - nmppsGet\_2s, [861](#)
  - nmppsGet\_2u, [862](#)
  - nmppsGet\_32s, [862](#)
  - nmppsGet\_32u, [863](#)
  - nmppsGet\_4s, [863](#)
  - nmppsGet\_4u, [864](#)
  - nmppsGet\_8s, [864](#)
  - nmppsGet\_8u, [865](#)
- nmppsGetVal\_1
  - nmppsGetVal\_, [849](#)
- nmppsGetVal\_16s
  - nmppsGetVal\_, [851](#)

- nmppsGetVal\_16u
  - nmppsGetVal\_, 851
- nmppsGetVal\_2s
  - nmppsGetVal\_, 852
- nmppsGetVal\_2u
  - nmppsGetVal\_, 853
- nmppsGetVal\_32s
  - nmppsGetVal\_, 853
- nmppsGetVal\_32u
  - nmppsGetVal\_, 854
- nmppsGetVal\_4s
  - nmppsGetVal\_, 854
- nmppsGetVal\_4u
  - nmppsGetVal\_, 855
- nmppsGetVal\_64s
  - nmppsGetVal\_, 856
- nmppsGetVal\_64u
  - nmppsGetVal\_, 856
- nmppsGetVal\_8s
  - nmppsGetVal\_, 857
- nmppsGetVal\_8u
  - nmppsGetVal\_, 857
- nmppsLastNonZeroIndx, 742
  - nmppsLastNonZeroIndx\_32s, 742
- nmppsLastNonZeroIndx\_32s
  - nmppsLastNonZeroIndx, 742
- nmppsLastZeroIndx, 741
- nmppsMalloc, 79
- NmppsMallocSpec, 984
- nmppsMaskV\_, 577
  - nmppsMaskV\_64u, 577
- nmppsMaskV\_64u
  - nmppsMaskV\_, 577
- nmppsMax\_, 715
  - nmppsMax\_16s15b, 715
  - nmppsMax\_32s31b, 716
  - nmppsMax\_64s63b, 717
  - nmppsMax\_8s7b, 718
- nmppsMax\_16s15b
  - nmppsMax\_, 715
- nmppsMax\_32s31b
  - nmppsMax\_, 716
- nmppsMax\_64s63b
  - nmppsMax\_, 717
- nmppsMax\_8s7b
  - nmppsMax\_, 718
- nmppsMaxEvery\_, 751
  - nmppsMaxEvery\_16s, 752
  - nmppsMaxEvery\_32s, 753
  - nmppsMaxEvery\_64s, 754
  - nmppsMaxEvery\_8s, 756
- nmppsMaxEvery\_16s
  - nmppsMaxEvery\_, 752
- nmppsMaxEvery\_32s
  - nmppsMaxEvery\_, 753
- nmppsMaxEvery\_64s
  - nmppsMaxEvery\_, 754
- nmppsMaxEvery\_8s
  - nmppsMaxEvery\_, 756
- nmppsMaxEvery\_, 756
- nmppsMaxIndx\_, 725
  - nmppsMaxIndx\_32s, 726
  - nmppsMaxIndx\_8s, 727
- nmppsMaxIndx\_32s
  - nmppsMaxIndx\_, 726
- nmppsMaxIndx\_8s
  - nmppsMaxIndx\_, 727
- nmppsMerge, 689
  - nmppsMerge\_32f, 689
- nmppsMerge\_32f
  - nmppsMerge, 689
- nmppsMin, 720
  - nmppsMin\_16s15b, 720
  - nmppsMin\_32s31b, 721
  - nmppsMin\_64s63b, 722
  - nmppsMin\_8s7b, 723
- nmppsMin\_16s15b
  - nmppsMin, 720
- nmppsMin\_32s31b
  - nmppsMin, 721
- nmppsMin\_64s63b
  - nmppsMin, 722
- nmppsMin\_8s7b
  - nmppsMin, 723
- nmppsMinCmpLtV\_, 758
  - nmppsMinCmpLtV\_16s, 759
- nmppsMinCmpLtV\_16s
  - nmppsMinCmpLtV\_, 759
- nmppsMinEvery\_, 744
  - nmppsMinEvery\_16s, 745
  - nmppsMinEvery\_32s, 746
  - nmppsMinEvery\_64s, 747
  - nmppsMinEvery\_8s, 748
- nmppsMinEvery\_16s
  - nmppsMinEvery\_, 745
- nmppsMinEvery\_32s
  - nmppsMinEvery\_, 746
- nmppsMinEvery\_64s
  - nmppsMinEvery\_, 747
- nmppsMinEvery\_8s
  - nmppsMinEvery\_, 748
- nmppsMinIndx\_, 729
  - nmppsMinIndx\_32s, 730
  - nmppsMinIndx\_8s, 731
- nmppsMinIndx\_32s
  - nmppsMinIndx\_, 730
- nmppsMinIndx\_8s
  - nmppsMinIndx\_, 731
- nmppsMinIndxV1024\_32s
  - nmppsMinIndxVN\_, 733
- nmppsMinIndxV16\_32s
  - nmppsMinIndxVN\_, 734
- nmppsMinIndxV256\_32s
  - nmppsMinIndxVN\_, 734
- nmppsMinIndxV25\_32s
  - nmppsMinIndxVN\_, 735
- nmppsMinIndxV9\_32s
  - nmppsMinIndxVN\_, 735

- nmppsMinIndxVN\_, 736
- nmppsMinIndxVN\_, 733
  - nmppsMinIndxV1024\_32s, 733
  - nmppsMinIndxV16\_32s, 734
  - nmppsMinIndxV256\_32s, 734
  - nmppsMinIndxV25\_32s, 735
  - nmppsMinIndxV9\_32s, 736
- nmppsMul, 474
  - nmppsMul\_32fcr, 474
- nmppsMul\_32fcr
  - nmppsMul, 474
- nmppsMul\_Add, 464
  - nmppsMul\_Add\_32f, 464
  - nmppsMul\_Add\_32fcr, 465
  - nmppsMul\_Add\_64f, 467
- nmppsMul\_Add\_32f
  - nmppsMul\_Add, 464
- nmppsMul\_Add\_32fcr
  - nmppsMul\_Add, 465
- nmppsMul\_Add\_64f
  - nmppsMul\_Add, 467
- nmppsMul\_AddC\_32f
  - nmppsMul\_AddC, 486
- nmppsMul\_AddC, 485, 486
  - nmppsMul\_AddC\_32f, 486
- nmppsMul\_ConjMul\_Add, 469
  - nmppsMul\_ConjMul\_Add\_32fcr, 469
- nmppsMul\_ConjMul\_Add\_32fcr
  - nmppsMul\_ConjMul\_Add, 469
- nmppsMul\_Mul\_Add, 459
  - nmppsMul\_Mul\_Add\_32f, 459
  - nmppsMul\_Mul\_Add\_32fcr, 460
  - nmppsMul\_Mul\_Add\_64f, 462
- nmppsMul\_Mul\_Add\_32f
  - nmppsMul\_Mul\_Add, 459
- nmppsMul\_Mul\_Add\_32fcr
  - nmppsMul\_Mul\_Add, 460
- nmppsMul\_Mul\_Add\_64f
  - nmppsMul\_Mul\_Add, 462
- nmppsMul\_Mul\_Sub, 480
  - nmppsMul\_Mul\_Sub\_32f, 480
- nmppsMul\_Mul\_Sub\_32f
  - nmppsMul\_Mul\_Sub, 480
- nmppsMulC\_16s
  - nmppsMulC, 448
- nmppsMulC\_16s32s
  - nmppsMulC, 450
- nmppsMulC\_32f
  - nmppsMulC, 457
- nmppsMulC\_32s
  - nmppsMulC, 451
- nmppsMulC\_32s64s
  - nmppsMulC, 452
- nmppsMulC\_64s
  - nmppsMulC, 453
- nmppsMulC\_8s
  - nmppsMulC, 454
- nmppsMulC\_8s16s
  - nmppsMulC, 455
- nmppsMulC\_Add, 483
  - nmppsMulC\_Add\_32fcr, 483
- nmppsMulC\_Add\_32fcr
  - nmppsMulC\_Add, 483
- nmppsMulC\_AddC\_2x32s
  - nmppsMulC\_AddC, 489
- nmppsMulC\_AddC\_32f
  - nmppsMulC\_AddC, 472
- nmppsMulC\_AddV\_AddC\_32s
  - nmppsMulC\_AddV\_AddC, 492
- nmppsMulC\_AddV\_AddC, 492
  - nmppsMulC\_AddV\_AddC\_32s, 492
- nmppsMulC\_AddC, 472, 489
  - nmppsMulC\_AddC\_2x32s, 489
  - nmppsMulC\_AddC\_32f, 472
- nmppsMulC, 448, 457
  - nmppsMulC\_16s, 448
  - nmppsMulC\_16s32s, 450
  - nmppsMulC\_32f, 457
  - nmppsMulC\_32s, 451
  - nmppsMulC\_32s64s, 452
  - nmppsMulC\_64s, 453
  - nmppsMulC\_8s, 454
  - nmppsMulC\_8s16s, 455
- nmppsNeg, 387
  - nmppsNeg\_16s, 387
  - nmppsNeg\_32s, 388
  - nmppsNeg\_64s, 390
  - nmppsNeg\_8s, 391
- nmppsNeg\_16s
  - nmppsNeg, 387
- nmppsNeg\_32s
  - nmppsNeg, 388
- nmppsNeg\_64s
  - nmppsNeg, 390
- nmppsNeg\_8s
  - nmppsNeg, 391
- nmppsNot\_, 527
  - nmppsNot\_16u, 527
  - nmppsNot\_2u, 529
  - nmppsNot\_32u, 530
  - nmppsNot\_4u, 531
  - nmppsNot\_64u, 532
  - nmppsNot\_8u, 533
- nmppsNot\_16u
  - nmppsNot\_, 527
- nmppsNot\_2u
  - nmppsNot\_, 529
- nmppsNot\_32u
  - nmppsNot\_, 530
- nmppsNot\_4u
  - nmppsNot\_, 531
- nmppsNot\_64u
  - nmppsNot\_, 532
- nmppsNot\_8u
  - nmppsNot\_, 533
- nmppsOr, 551

- nmppsOr\_1, [551](#)
- nmppsOr\_16u, [553](#)
- nmppsOr\_2u, [554](#)
- nmppsOr\_32u, [555](#)
- nmppsOr\_4u, [557](#)
- nmppsOr\_64u, [558](#)
- nmppsOr\_8u, [559](#)
- nmppsOr3V\_, [561](#)
- nmppsOr4V\_, [562](#)
- nmppsOr\_1
  - nmppsOr, [551](#)
- nmppsOr\_16u
  - nmppsOr, [553](#)
- nmppsOr\_2u
  - nmppsOr, [554](#)
- nmppsOr\_32u
  - nmppsOr, [555](#)
- nmppsOr\_4u
  - nmppsOr, [557](#)
- nmppsOr\_64u
  - nmppsOr, [558](#)
- nmppsOr\_8u
  - nmppsOr, [559](#)
- nmppsOrC, [550](#)
- nmppsPut\_1
  - nmppsSetVal\_, [838](#)
- nmppsPut\_16s
  - nmppsSetVal\_, [840](#)
- nmppsPut\_16u
  - nmppsSetVal\_, [840](#)
- nmppsPut\_2s
  - nmppsSetVal\_, [841](#)
- nmppsPut\_2u
  - nmppsSetVal\_, [842](#)
- nmppsPut\_32s
  - nmppsSetVal\_, [842](#)
- nmppsPut\_32u
  - nmppsSetVal\_, [843](#)
- nmppsPut\_4s
  - nmppsSetVal\_, [843](#)
- nmppsPut\_4u
  - nmppsSetVal\_, [844](#)
- nmppsPut\_64s
  - nmppsSetVal\_, [845](#)
- nmppsPut\_64u
  - nmppsSetVal\_, [845](#)
- nmppsPut\_8s
  - nmppsSetVal\_, [846](#)
- nmppsPut\_8u
  - nmppsSetVal\_, [846](#)
- nmppsQSort\_32s
  - VEC\_QSort, [867](#)
- nmppsRShiftC\_, [588](#)
  - nmppsRShiftC\_16u, [588](#)
  - nmppsRShiftC\_32u, [590](#)
  - nmppsRShiftC\_4u, [591](#)
  - nmppsRShiftC\_64u, [593](#)
  - nmppsRShiftC\_8u, [594](#)
- nmppsRShiftC\_16s
  - nmppsRShiftC, [580](#)
- nmppsRShiftC\_16u
  - nmppsRShiftC\_, [588](#)
- nmppsRShiftC\_32s
  - nmppsRShiftC, [582](#)
- nmppsRShiftC\_32u
  - nmppsRShiftC\_, [590](#)
- nmppsRShiftC\_4u
  - nmppsRShiftC\_, [591](#)
- nmppsRShiftC\_64s
  - nmppsRShiftC, [584](#)
- nmppsRShiftC\_64u
  - nmppsRShiftC\_, [593](#)
- nmppsRShiftC\_8s
  - nmppsRShiftC, [585](#)
- nmppsRShiftC\_8u
  - nmppsRShiftC\_, [594](#)
- nmppsRShiftC\_AddC\_, [597](#)
- nmppsRShiftC\_MulC\_AddC, [491](#)
- nmppsRShiftC, [580](#)
  - nmppsRShiftC\_16s, [580](#)
  - nmppsRShiftC\_32s, [582](#)
  - nmppsRShiftC\_64s, [584](#)
  - nmppsRShiftC\_8s, [585](#)
- nmppsRamp\_, [629](#)
  - nmppsRamp\_16s, [629](#)
  - nmppsRamp\_32s, [631](#)
  - nmppsRamp\_64s, [633](#)
  - nmppsRamp\_8s, [635](#)
- nmppsRamp\_16s
  - nmppsRamp\_, [629](#)
- nmppsRamp\_32s
  - nmppsRamp\_, [631](#)
- nmppsRamp\_64s
  - nmppsRamp\_, [633](#)
- nmppsRamp\_8s
  - nmppsRamp\_, [635](#)
- nmppsRand, [171](#)
  - nmppsRandMinMaxDiv, [171](#)
- nmppsRand\_32f
  - nmppsRandUniform, [624](#)
- nmppsRandUniform, [614](#), [624](#)
  - nmppsRand\_32f, [624](#)
  - nmppsRandUniform\_16s, [614](#)
  - nmppsRandUniform\_16u, [615](#)
  - nmppsRandUniform\_32f\_integer, [625](#)
  - nmppsRandUniform\_32s, [616](#)
  - nmppsRandUniform\_32u, [617](#)
  - nmppsRandUniform\_64f, [626](#)
  - nmppsRandUniform\_64s, [618](#)
  - nmppsRandUniform\_64u, [619](#)
  - nmppsRandUniform\_8s, [620](#)
  - nmppsRandUniform\_8u, [622](#)
- nmppsRandUniform\_, [628](#)
- nmppsRandUniform\_16s
  - nmppsRandUniform, [614](#)
- nmppsRandUniform\_16u

- nmppsRandUniform, [615](#)
- nmppsRandUniform\_32f\_integer
  - nmppsRandUniform, [625](#)
- nmppsRandUniform\_32s
  - nmppsRandUniform, [616](#)
- nmppsRandUniform\_32u
  - nmppsRandUniform, [617](#)
- nmppsRandUniform\_64f
  - nmppsRandUniform, [626](#)
- nmppsRandUniform\_64s
  - nmppsRandUniform, [618](#)
- nmppsRandUniform\_64u
  - nmppsRandUniform, [619](#)
- nmppsRandUniform\_8s
  - nmppsRandUniform, [620](#)
- nmppsRandUniform\_8u
  - nmppsRandUniform, [622](#)
- nmppsRemap\_, [869](#)
- nmppsSet-инициализация, [602](#)
  - nmppsSet\_16s, [602](#)
  - nmppsSet\_16u, [603](#)
  - nmppsSet\_32s, [605](#)
  - nmppsSet\_32u, [606](#)
  - nmppsSet\_64s, [607](#)
  - nmppsSet\_64sp, [608](#)
  - nmppsSet\_64u, [609](#)
  - nmppsSet\_64up, [610](#)
  - nmppsSet\_8s, [611](#)
  - nmppsSet\_8u, [612](#)
- nmppsSet\_16s
  - nmppsSet-инициализация, [602](#)
- nmppsSet\_16u
  - nmppsSet-инициализация, [603](#)
- nmppsSet\_32s
  - nmppsSet-инициализация, [605](#)
- nmppsSet\_32u
  - nmppsSet-инициализация, [606](#)
- nmppsSet\_64s
  - nmppsSet-инициализация, [607](#)
- nmppsSet\_64sp
  - nmppsSet-инициализация, [608](#)
- nmppsSet\_64u
  - nmppsSet-инициализация, [609](#)
- nmppsSet\_64up
  - nmppsSet-инициализация, [610](#)
- nmppsSet\_8s
  - nmppsSet-инициализация, [611](#)
- nmppsSet\_8u
  - nmppsSet-инициализация, [612](#)
- nmppsSetVal\_, [838](#)
  - nmppsPut\_1, [838](#)
  - nmppsPut\_16s, [840](#)
  - nmppsPut\_16u, [840](#)
  - nmppsPut\_2s, [841](#)
  - nmppsPut\_2u, [842](#)
  - nmppsPut\_32s, [842](#)
  - nmppsPut\_32u, [843](#)
  - nmppsPut\_4s, [843](#)
  - nmppsPut\_4u, [844](#)
  - nmppsPut\_64s, [845](#)
  - nmppsPut\_64u, [845](#)
  - nmppsPut\_8s, [846](#)
  - nmppsPut\_8u, [846](#)
- nmppsSub, [430](#), [436](#)
  - nmppsSub\_16s, [430](#)
  - nmppsSub\_32f, [436](#)
  - nmppsSub\_32s, [432](#)
  - nmppsSub\_64s, [433](#)
  - nmppsSub\_8s, [434](#)
- nmppsSub\_16s
  - nmppsSub, [430](#)
- nmppsSub\_32f
  - nmppsSub, [436](#)
- nmppsSub\_32s
  - nmppsSub, [432](#)
- nmppsSub\_64s
  - nmppsSub, [433](#)
- nmppsSub\_8s
  - nmppsSub, [434](#)
- nmppsSubC\_16s
  - nmppsSubC, [417](#)
- nmppsSubC\_32s
  - nmppsSubC, [419](#)
- nmppsSubC\_4s
  - nmppsSubC, [420](#)
- nmppsSubC\_64s
  - nmppsSubC, [421](#)
- nmppsSubC\_8s
  - nmppsSubC, [422](#)
- nmppsSubCRev, [424](#)
  - nmppsSubCRev\_16s, [424](#)
  - nmppsSubCRev\_32s, [426](#)
  - nmppsSubCRev\_64s, [427](#)
  - nmppsSubCRev\_8s, [428](#)
- nmppsSubCRev\_16s
  - nmppsSubCRev, [424](#)
- nmppsSubCRev\_32s
  - nmppsSubCRev, [426](#)
- nmppsSubCRev\_64s
  - nmppsSubCRev, [427](#)
- nmppsSubCRev\_8s
  - nmppsSubCRev, [428](#)
- nmppsSubC, [417](#)
  - nmppsSubC\_16s, [417](#)
  - nmppsSubC\_32s, [419](#)
  - nmppsSubC\_4s, [420](#)
  - nmppsSubC\_64s, [421](#)
  - nmppsSubC\_8s, [422](#)
- nmppsSum, [499](#)
  - nmppsSum\_16s, [499](#)
  - nmppsSum\_32s, [500](#)
  - nmppsSum\_64s, [501](#)
  - nmppsSum\_8s, [502](#)
- nmppsSum\_1, [504](#)
- nmppsSum\_16s
  - nmppsSum, [499](#)

- nmppsSum\_32s
  - nmppsSum, [500](#)
- nmppsSum\_64s
  - nmppsSum, [501](#)
- nmppsSum\_8s
  - nmppsSum, [502](#)
- nmppsSumN, [495](#)
- nmppsSwap\_, [713](#)
  - nmppsSwap\_64s, [713](#)
- nmppsSwap\_64s
  - nmppsSwap\_, [713](#)
- NmppsTmpSpec, [984](#)
- nmppsWeightedSum, [520](#)
  - nmppsWeightedSum\_16s32s, [521](#)
  - nmppsWeightedSum\_32s64s, [522](#)
  - nmppsWeightedSum\_8s16s, [524](#)
- nmppsWeightedSum\_16s32s
  - nmppsWeightedSum, [521](#)
- nmppsWeightedSum\_32s64s
  - nmppsWeightedSum, [522](#)
- nmppsWeightedSum\_8s16s
  - nmppsWeightedSum, [524](#)
- nmppsXor, [569](#)
  - nmppsXor\_16u, [569](#)
  - nmppsXor\_32u, [571](#)
  - nmppsXor\_4u, [572](#)
  - nmppsXor\_64u, [573](#)
  - nmppsXor\_8u, [575](#)
- nmppsXor\_16u
  - nmppsXor, [569](#)
- nmppsXor\_32u
  - nmppsXor, [571](#)
- nmppsXor\_4u
  - nmppsXor, [572](#)
- nmppsXor\_64u
  - nmppsXor, [573](#)
- nmppsXor\_8u
  - nmppsXor, [575](#)
- nmppsXorC\_16u
  - nmppsXorC, [563](#)
- nmppsXorC\_32u
  - nmppsXorC, [565](#)
- nmppsXorC\_64u
  - nmppsXorC, [566](#)
- nmppsXorC\_8u
  - nmppsXorC, [567](#)
- nmppsXorC, [563](#)
  - nmppsXorC\_16u, [563](#)
  - nmppsXorC\_32u, [565](#)
  - nmppsXorC\_64u, [566](#)
  - nmppsXorC\_8u, [567](#)
- nmreg, [984](#)
- nmshort, [985](#)
- nmshort2D< Y, X >, [985](#)
- nmtype.h
  - CAPACITY\_nm64s, [1017](#)
  - NM16Sx4, [1017](#)
  - NM32Sx2, [1018](#)
  - VEC\_NM16S, [1018](#)
  - VEC\_NM16U, [1018](#)
  - VEC\_NM32S, [1019](#)
  - VEC\_NM4U, [1019](#)
  - VEC\_NM8S, [1019](#)
- nmvecpack< T >, [986](#)
- operator<<
  - контроль переполнения, [891](#)
- RGB32\_nm10s, [988](#)
- RGB32\_nm10u, [988](#)
- RGB32\_nm8s, [988](#)
- RGB32\_nm8u, [989](#)
- RGB64\_nm16u, [989](#)
- RPoint, [990](#)
- re
  - s\_nm64sc, [993](#)
- RoundShift, [990](#)
- S\_BufferInfo, [990](#)
- s\_int32x2, [991](#)
- s\_nm32fc, [992](#)
- s\_nm32fcr, [992](#)
- s\_nm32sc, [992](#)
- s\_nm64sc, [993](#)
  - im, [993](#)
  - re, [993](#)
- S\_nmppiFilterKernel, [993](#)
- S\_nmppiFilterKernel\_32s32s, [994](#)
- s\_v16nm16s, [994](#)
- s\_v16nm16u, [995](#)
- s\_v16nm32s, [995](#)
- s\_v16nm32u, [995](#)
- s\_v16nm4u, [996](#)
- s\_v16nm8s, [996](#)
- s\_v16nm8u, [997](#)
- s\_v2nm32f, [997](#)
- s\_v2nm32s, [997](#)
- s\_v2nm32u, [998](#)
- s\_v4nm16s, [998](#)
- s\_v4nm16u, [999](#)
- s\_v4nm32f, [999](#)
- s\_v4nm32s, [999](#)
- s\_v4nm32u, [1000](#)
- s\_v4nm64f, [1000](#)
- s\_v4nm8u, [1000](#)
- s\_v8nm16s, [1001](#)
- s\_v8nm16u, [1001](#)
- s\_v8nm32s, [1002](#)
- s\_v8nm32u, [1002](#)
- s\_v8nm8s, [1003](#)
- s\_v8nm8u, [1003](#)
- SIG\_CreateResample, [335](#)
- SIG\_Median3, [326](#)
- SIG\_Resample\_perf, [337](#)
- SIG\_ResampleDown2, [333](#)
- SIG\_ResampleUp3Down2, [334](#)
- SIG\_SetResample, [336](#)

- [SIG\\_XCorr](#), [324](#)  
[SpecTmp1](#), [1003](#)  
[spot\\_struct](#), [1004](#)  
  
[tagSegmentInfo](#), [1004](#)  
[tfixpoint< T, point >](#), [1005](#)  
[Tmp2BuffSpec](#), [1006](#)  
  
[uint15b](#)  
        Типы скалярных данных, [355](#)  
[uint16b](#)  
        Типы скалярных данных, [355](#)  
[uint16ptr](#), [1006](#)  
[uint1b](#)  
        Типы скалярных данных, [355](#)  
[uint2b](#)  
        Типы скалярных данных, [355](#)  
[uint31b](#)  
        Типы скалярных данных, [356](#)  
[uint32b](#)  
        Типы скалярных данных, [356](#)  
[uint3b](#)  
        Типы скалярных данных, [356](#)  
[uint4b](#)  
        Типы скалярных данных, [356](#)  
[uint63b](#)  
        Типы скалярных данных, [357](#)  
[uint64b](#)  
        Типы скалярных данных, [357](#)  
[uint7b](#)  
        Типы скалярных данных, [357](#)  
[uint8b](#)  
        Типы скалярных данных, [357](#)  
[uint8ptr](#), [1007](#)  
  
[v16nm16s](#)  
        Типы векторных данных, [347](#)  
[v16nm16u](#)  
        Типы векторных данных, [347](#)  
[v16nm32s](#)  
        Типы векторных данных, [347](#)  
[v16nm32u](#)  
        Типы векторных данных, [347](#)  
[v16nm4b3u](#)  
        Типы векторных данных, [348](#)  
[v16nm4s](#), [1008](#)  
[v16nm4u](#)  
        Типы векторных данных, [348](#)  
[v16nm8s](#)  
        Типы векторных данных, [348](#)  
[v16nm8s7b](#)  
        Типы векторных данных, [348](#)  
[v16nm8u](#)  
        Типы векторных данных, [348](#)  
[v2nm32s](#)  
        Типы векторных данных, [348](#)  
[v2nm32u](#)  
        Типы векторных данных, [349](#)  
[v4nm16s](#)  
        Типы векторных данных, [349](#)  
[v4nm16u](#)  
        Типы векторных данных, [349](#)  
[v4nm32s](#)  
        Типы векторных данных, [349](#)  
[v4nm32u](#)  
        Типы векторных данных, [349](#)  
[v4nm8s](#), [1008](#)  
[v4nm8u](#)  
        Типы векторных данных, [349](#)  
[v8nm16s](#)  
        Типы векторных данных, [349](#)  
[v8nm16u](#)  
        Типы векторных данных, [349](#)  
[v8nm32s](#)  
        Типы векторных данных, [350](#)  
[v8nm32u](#)  
        Типы векторных данных, [350](#)  
[v8nm8s](#)  
        Типы векторных данных, [350](#)  
[v8nm8u](#)  
        Типы векторных данных, [350](#)  
[VEC\\_NM16S](#)  
        nmtype.h, [1018](#)  
[VEC\\_NM16U](#)  
        nmtype.h, [1018](#)  
[VEC\\_NM32S](#)  
        nmtype.h, [1019](#)  
[VEC\\_NM4U](#)  
        nmtype.h, [1019](#)  
[VEC\\_NM8S](#)  
        nmtype.h, [1019](#)  
[VEC\\_QSort](#), [867](#)  
        nmppsQSort\_32s, [867](#)  
[vec< T >](#), [1009](#)  
[Vec\\_0\\_sub\\_data](#), [895](#)  
[Vec\\_Abs](#), [916](#)  
[Vec\\_AccMul1D1W32\\_AddVr](#), [960](#)  
[Vec\\_Add](#), [917](#)  
[Vec\\_Add\\_VV\\_shift](#), [900](#)  
[Vec\\_And](#), [912](#)  
[Vec\\_BuildDiagWeights16](#), [953](#)  
[Vec\\_BuildDiagWeights8](#), [952](#)  
[Vec\\_ClipExt](#), [918](#)  
[Vec\\_ClipMul2D2W8\\_AddVr](#), [919](#)  
[Vec\\_ClipMulNDNW2\\_AddVr](#), [920](#)  
[Vec\\_ClipMulNDNW4\\_AddVr](#), [921](#)  
[Vec\\_ClipMulNDNW8\\_AddVr](#), [922](#)  
[Vec\\_ClipRShiftConvert\\_AddC](#), [827](#)  
[Vec\\_CompareMaxV](#), [949](#)  
[Vec\\_CompareMinV](#), [948](#)  
[Vec\\_DupValueInVector16](#), [951](#)  
[Vec\\_DupValueInVector8](#), [950](#)  
[Vec\\_FilterCoreRow2](#), [909](#)  
[Vec\\_FilterCoreRow4](#), [910](#)  
[Vec\\_FilterCoreRow8](#), [911](#)  
[Vec\\_IncNeg](#), [923](#)  
[Vec\\_MUL\\_2V4toW8\\_shift](#), [936](#)

Vec\_MUL\_2V8toW16\_shift, [937](#)  
Vec\_Mask, [913](#)  
Vec\_MaxVal, [956](#)  
Vec\_MaxVal\_v4nm16s, [955](#)  
Vec\_MaxVal\_v8nm8s, [954](#)  
Vec\_MinVal, [959](#)  
Vec\_MinVal\_v4nm16s, [958](#)  
Vec\_MinVal\_v8nm8s, [957](#)  
Vec\_Mul2D2W1\_AddVr, [924](#)  
Vec\_Mul2D2W2\_AddVr, [925](#)  
Vec\_Mul2D2W4\_AddVr, [926](#)  
Vec\_Mul2D2W8\_AddVr, [927](#)  
Vec\_Mul3D3W2\_AddVr, [928](#)  
Vec\_Mul3D3W8\_AddVr, [929](#)  
Vec\_Mul4D4W2\_AddVr, [930](#)  
Vec\_MulVN\_AddVN, [931](#)  
Vec\_Or, [914](#)  
Vec\_Sub, [932](#)  
Vec\_SubAbs, [933](#)  
Vec\_SubVN\_Abs, [934](#)  
Vec\_Swap, [935](#)  
Vec\_Xor, [915](#)  
Vec\_activate\_data, [896](#)  
Vec\_activate\_data\_add\_0, [897](#)  
Vec\_activate\_data\_add\_ram, [899](#)  
Vec\_activate\_data\_xor\_data, [898](#)  
Vec\_afifo, [901](#)  
Vec\_data, [902](#)  
Vec\_data\_add\_afifo, [903](#)  
Vec\_data\_add\_ram, [904](#)  
Vec\_data\_and\_ram, [905](#)  
Vec\_data\_or\_ram, [906](#)  
Vec\_data\_sub\_ram, [907](#)  
Vec\_data\_xor\_ram, [908](#)  
Vec\_not\_data, [938](#)  
Vec\_ram, [939](#)  
Vec\_ram\_sub\_data, [940](#)  
Vec\_vsum\_activate\_data\_0, [941](#)  
Vec\_vsum\_data\_0, [942](#)  
Vec\_vsum\_data\_afifo, [943](#)  
Vec\_vsum\_data\_vr, [944](#)  
Vec\_vsum\_shift\_data\_0, [945](#)  
Vec\_vsum\_shift\_data\_afifo, [947](#)  
Vec\_vsum\_shift\_data\_vr, [946](#)