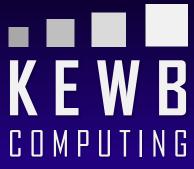
Adventures in SIMD Thinking (Part 1 of 2)

Bob Steagall CppCon 2020



Agenda



- Learn a little about Intel's SIMD facilities (disclaimer: I don't work for Intel)
- Create some useful functions in terms of AVX-512 intrinsics
- Try some SIMD-style thinking to tackle some interesting problems
 - Intra-register sorting
 - Fast linear median-of-seven filter
 - Fast small-kernel convolution
 - Faster (?) UTF-8 to UTF-32 conversion (with AVX2)
- No heavy code, but lots of pictures
 - Thinking "vertically"

Getting Started



```
#include <cstdio>
#include <cstdint>
#include <type_traits>
#ifdef __OPTIMIZE__
   #include <immintrin.h>
   #define KEWB_FORCE_INLINE inline __attribute__((__always_inline__))
#else
    #define OPTIMIZE
   #include <immintrin.h>
   #undef OPTIMIZE
   #define KEWB FORCE INLINE
                             inline
#endif
namespace simd {
using rf_512 = __m512;
using ri_512 = \__m512i;
using msk_512 = uint32_t;
. . .
```

Function load_value()



```
KEWB_FORCE_INLINE rf_512
load_value(float fill)
                                                  return _mm512_set1_ps(v);
   fill
                                                                                                 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2.3 | 2
```

CppCon 2020 - Adventures in SIMD Thinking Copyright © 2020 Bob Steagall

Function load_value()



```
KEWB_FORCE_INLINE ri_512
load_value(int32_t fill)
    return _mm512_set1_epi32(i);
fill
```

CppCon 2020 - Adventures in SIMD Thinking Copyright © 2020 Bob Steagall

Function load_from()



```
KEWB_FORCE_INLINE rf_512
load_from(float const* psrc)
              return _mm512_loadu_ps(psrc);
 psrç
                                                 \mathbf{m}_{\scriptscriptstyle 1}
                                                                    \mathbf{m}_{2}
                                                                                        \mathbf{M}_{3}
                                                                                                           \mathbf{m}_{4}
                                                                                                                              \mathbf{m}_{5}
                                                                                                                                                                                                                              \mathbf{m}_{10}
                                                                                                                                                                                                                                                 \mathbf{m}_{11}
                                                                                                                                                                                                                                                                     \mathbf{m}_{12}
                                                                                                                                                                                                                                                                                        \mathbf{m}_{13}
                                                                                                                                                                                                                                                                                                           \mathbf{m}_{14}
                              \mathbf{m}_{o}
                                                                                                                                                  \mathbf{m}_{6}
                                                                                                                                                                     \mathbf{m}_{7}
                                                                                                                                                                                        m<sub>8</sub>
                                                                                                                                                                                                            \mathbf{m}_9
                                                                                                                                                                                                                                                                                                                              \mathbf{m}_{15}
                                                                                                                                                                                                                                                                                      \mathbf{m}_{_{13}}
                                                                                        \mathbf{M}_{3}
                                                                                                                                                                                                                                                 \mathbf{m}_{11}
                              \mathsf{m}_{\mathsf{o}}
                                                 \mathsf{m}_{\scriptscriptstyle 1}
                                                                    \mathbf{m}_{2}
                                                                                                                                                                                                                              \mathsf{m}_{\scriptscriptstyle{10}}
                                                                                                                                                                                                                                                                    \mathbf{M}_{12}
                                                                                                                                                                                                                                                                                                           \mathbf{m}_{_{14}}
                                                                                                           \mathsf{m}_{\scriptscriptstyle{4}}
                                                                                                                              \mathbf{m}_{5}
                                                                                                                                                  \mathbf{m}_{6}
                                                                                                                                                                     \mathbf{m}_{7}
                                                                                                                                                                                        \mathsf{m}_{\mathsf{8}}
                                                                                                                                                                                                           \mathbf{m}_{9}
                                                                                                                                                                                                                                                                                                                              \mathbf{m}_{15}
```

Function masked_load_from()



```
KEWB FORCE INLINE rf 512
masked_load_from(float const* psrc, float fill, msk_512 mask)
        return _mm512_mask_loadu_ps(_mm512_set1_ps(fill), (__mmask16) mask, psrc);
psrç
                            \mathbf{m}_{1}
                                       \mathbf{m}_{2}
                                                  \mathbf{M}_{3}
                                                                                                                                                                                   \mathbf{m}_{15}
                 Ma
                                                            \mathbf{m}_{4}
                                                                       \mathbf{m}_{5}
                                                                                  \mathbf{m}_{6}
                                                                                             \mathbf{m}_{7}
                                                                                                        m<sub>8</sub>
                                                                                                                   \mathbf{m}_{9}
                                                                                                                             \mathbf{m}_{10}
                                                                                                                                        \mathbf{m}_{11}
                                                                                                                                                   \mathbf{m}_{12}
                                                                                                                                                              \mathbf{M}_{13}
                                                                                                                                                                         \mathbf{m}_{14}
 fill
                                                                                                                                         0
                             0
                                       1
                                                   1
                                                                        0
                                                                                   1
                                                             0
                                                                                                         0
                                                                                                                   0
                                                                                                                              0
                                                                                                                                                     1
                                                                                                                                                                          1
 mask
                                       \mathbf{m}_{2}
                                                  \mathbf{M}_{3}
                                                                                  \mathbf{m}_{6}
                                                                                             \mathbf{m}_{7}
                                                                                                                                                   \mathbf{m}_{12}
                                                                                                                                                              \mathbf{M}_{13}
                                                                                                                                                                         \mathbf{m}_{14}
                                                                                                                                                                                   \mathbf{m}_{15}
```

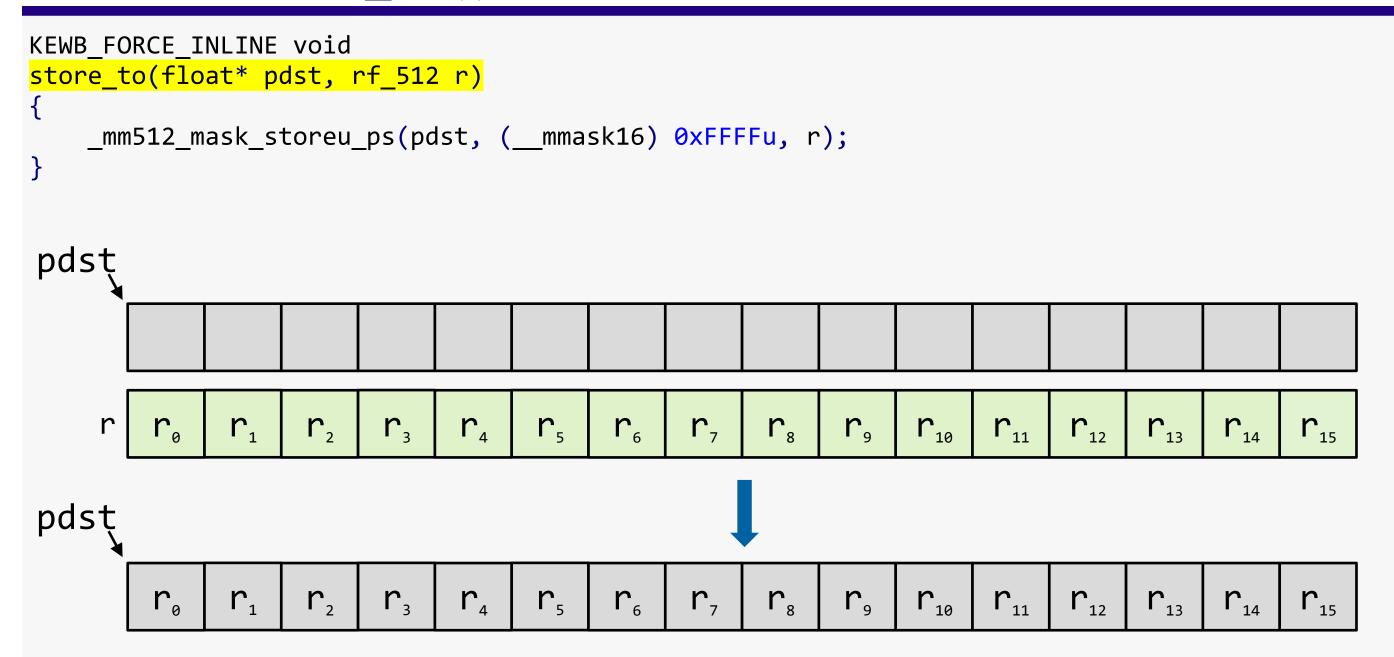
Function masked_load_from()



```
KEWB FORCE INLINE rf 512
masked_load_from(float const* psrc, rf_512 fill, msk_512 mask)
         return _mm512_mask_loadu_ps(fill, (__mmask16) mask, psrc);
 psrç
                             \mathbf{m}_{1}
                                        \mathbf{m}_{2}
                                                    \mathbf{M}_{3}
                                                                                                                                                                    \mathbf{m}_{13}
                                                                                                                                                                               \mathbf{m}_{14}
                  Ma
                                                               \mathbf{m}_{4}
                                                                          \mathbf{m}_{5}
                                                                                      \mathbf{m}_{6}
                                                                                                 \mathbf{m}_{7}
                                                                                                            m<sub>8</sub>
                                                                                                                        \mathbf{m}_{9}
                                                                                                                                  \mathbf{m}_{10}
                                                                                                                                              \mathbf{m}_{11}
                                                                                                                                                         \mathbf{m}_{12}
                                                                                                                                                                                           \mathbf{m}_{15}
                                         f_{2}
                                                    f_3
                                                               f_4
                                                                           f_{5}
                                                                                                 f_{7}
                                                                                                                                  f_{10}
                                                                                                                                              f<sub>11</sub>
                                                                                                                                                                     f_{13}
                                                                                                                        f_9
 fill
                              0
                                         1
                                                     1
                                                                                                                                               0
                                                                0
                                                                           0
                                                                                       1
                                                                                                  1
                                                                                                             0
                                                                                                                         0
                                                                                                                                    0
                                                                                                                                                           1
                                                                                                                                                                                 1
 mask
                                                               f_{a}
                                                                           f_{5}
                                                                                                                                              f_{11}
                                        \mathbf{m}_{2}
                                                    \mathbf{M}_{3}
                                                                                      \mathbf{m}_{6}
                                                                                                 \mathbf{m}_{7}
                                                                                                                                                         \mathbf{m}_{12}
                                                                                                                                                                    \mathbf{M}_{13}
                                                                                                                                                                                \mathbf{m}_{14}
                                                                                                                                                                                           \mathbf{m}_{15}
```

Function store_to()





Function masked_store_to()



```
KEWB FORCE INLINE void
masked_store_to(float* pdst, rf_512 r, msk_512 mask)
         _mm512_mask_storeu_ps(pdst, (__mmask16) mask, r);
 pdsţ
                              \mathbf{m}_{1}
                                          \mathbf{m}_{2}
                                                      \mathbf{M}_{3}
                                                                                                                                                                            \mathbf{M}_{13}
                                                                                                                                                                                                    \mathbf{m}_{15}
                   m<sub>o</sub>
                                                                  \mathbf{m}_{4}
                                                                              \mathbf{m}_{5}
                                                                                          \mathbf{m}_{6}
                                                                                                      \mathbf{m}_{7}
                                                                                                                  m<sub>8</sub>
                                                                                                                             \mathbf{m}_{9}
                                                                                                                                         \mathbf{m}_{10}
                                                                                                                                                     \mathbf{m}_{11}
                                                                                                                                                                \mathbf{m}_{12}
                                                                                                                                                                                        \mathbf{m}_{14}
                                                      r_{3}
                                                                                                                                                     r_{11}
                               r_{1}
                                          r_{2}
                                                                   r_4
                                                                              r_{5}
                                                                                                      r_{7}
                                                                                                                              r_{10}
                                                                                                                                                                r_{12}
                                                                                                                                                                             \mathbf{r}_{13}
                                                                                                                                                      0
                               0
                                           1
                                                        1
                                                                   0
                                                                               0
                                                                                                                                          0
                                                                                                                                                                  1
                                                                                                                                                                                          1
                                                                                           1
                                                                                                       1
                                                                                                                  0
                                                                                                                              0
                                                                                                                                                                                                      1
 mask
pdsţ
                                          r_2
                                                      r_{3}
                              \mathbf{m}_{1}
                                                                  \mathbf{m}_{4}
                                                                              \mathbf{m}_{5}
                                                                                                      r_7
                                                                                                                                         \mathbf{m}_{10}
                                                                                                                                                    \mathbf{m}_{11}
                                                                                                                                                                 r_{12}
                                                                                                                                                                             r_{13}
                                                                                                                                                                                                    r_{15}
                   m<sub>o</sub>
                                                                                                                             m<sub>9</sub>
                                                                                                                  m<sub>8</sub>
```

Function make_bit_mask()



```
template<unsigned A=0, unsigned B=0, unsigned C=0, unsigned D=0,
         unsigned E=0, unsigned F=0, unsigned G=0, unsigned H=0,
         unsigned I=0, unsigned J=0, unsigned K=0, unsigned L=0,
         unsigned M=0, unsigned N=0, unsigned O=0, unsigned P=0>
KEWB FORCE INLINE constexpr uint32 t
make bit mask()
    static_assert((A < 2) && (B < 2) && (C < 2) && (D < 2) &&
                  (E < 2) \&\& (F < 2) \&\& (G < 2) \&\& (H < 2) \&\&
                  (I < 2) \&\& (J < 2) \&\& (K < 2) \&\& (L < 2) \&\&
                  (M < 2) \&\& (N < 2) \&\& (O < 2) \&\& (P < 2));
    return ((A << 0) | (B << 1) | (C << 2) | (D << 3) |
            (E << 4) | (F << 5) | (G << 6) | (H << 7)
            (I << 8) | (J << 9) | (K << 10) | (L << 11) |
            (M << 12) \mid (N << 13) \mid (O << 14) \mid (P << 15));
                                                        ObPONM'LKJI'HGFE'DCBA
                              E
                                                                         M
                                                                               N
                                              Н
```

Function blend()



```
KEWB FORCE INLINE rf 512
blend(rf_512 a, rf_512 b, msk_512 mask)
        return _mm512_mask_blend_ps((__mmask16) mask, a, b);
         a
                                                                                                                                                                      a<sub>14</sub>
                                      a_2
                            a_{\scriptscriptstyle 1}
                                                 a_3
                                                            a_4
                                                                       a_{5}
                                                                                 a_{6}
                                                                                            a<sub>7</sub>
                                                                                                       a<sub>8</sub>
                                                                                                                 a_9
                                                                                                                           a<sub>10</sub>
                                                                                                                                      a<sub>11</sub>
                                                                                                                                                 a<sub>12</sub>
                                                                                                                                                            a<sub>13</sub>
                                                                                                                                                                                 a<sub>15</sub>
         b
                                      b_2
                                                 b_3
                                                                                 b_6
                                                                                                                 b_9
                                                                                                                                      b_{11}
                                                                                                                                                            b_{13}
                            b_1
                                                            b_4
                                                                       b_{5}
                                                                                            b_7
                                                                                                       b<sub>8</sub>
                                                                                                                           b_{10}
                                                                                                                                                 b_{12}
                                                                                                                                                                       b<sub>14</sub>
                                                                                                                                                                                 b<sub>15</sub>
                            0
                                       1
                                                                                                                             0
                                                                                                                                       0
 mask
                                                             0
                                                                       0
                                                                                                       0
                                                                                                                  0
                                      b_2
                                                 b_3
                                                                                 b_6
                                                                                                                                                            b_{13}
                                                                                            b_7
                                                                                                                                                 b_{12}
                                                                                                                                                                       b_{14}
                                                                                                                                                                                 b_{15}
                                                                                                                           a<sub>10</sub>
                            a_{\scriptscriptstyle 1}
                                                            a_4
                                                                       a<sub>5</sub>
                                                                                                       a<sub>8</sub>
                                                                                                                 a_9
                                                                                                                                      a<sub>11</sub>
                 a
```

Function permute()



```
KEWB_FORCE_INLINE rf_512
permute(rf_512 r, ri_512 perm)
     return _mm512_permutexvar_ps(perm, r);
                                 \mathbf{r}_{3}
                                               \mathbf{r}_{5}
                                                                                          r_{11}
                                                                                                 r_{12}
                                                                                                        \mathbf{r}_{13}
                                        r_4
                                                                            r_9
                 14
                         13
                                12
                                       11
                                               10
                                                       9
                                                              8
                                                                                   5
                                                                                                  3
                                                                                                                       0
                                r_{12}
                                                                     r_{7}
```

Function masked_permute()



```
KEWB FORCE INLINE rf 512
masked_permute(rf_512 a, rf_512 b, ri_512 perm, msk_512 mask)
     return _mm512_mask_permutexvar_ps(a, (__mmask16) mask, perm, b);
                             r_{3}
                                   r_4
                                          r_{5}
                                                                                r_{11}
                                                                                      \mathbf{r}_{12}
                                                                                             13
                             12
                                         10
                                                                                       3
                14
                                   11
                                                                                                          0
perm
                0
                                    0
                                          0
                                                             0
                                                                   0
                                                                          0
                                                                                0
mask
                r_{1}
                      \mathbf{r}_{13}
                             r_{12}
                                    r_{5}
                                                r<sub>8</sub>
                                                                   r_{11}
                                                                                       r_3
                                                                                             r_{2}
```

Function make_perm_map()



```
template<unsigned A, unsigned B, unsigned C, unsigned D,
         unsigned E, unsigned F, unsigned G, unsigned H,
         unsigned I, unsigned J, unsigned K, unsigned L,
         unsigned M, unsigned N, unsigned O, unsigned P>
KEWB_FORCE_INLINE ri 512
make_perm_map()
    static_assert((A < 16) && (B < 16) && (C < 16) && (D < 16) &&
                   (E < 16) && (F < 16) && (G < 16) && (H < 16) &&
                   (I < 16) \&\& (J < 16) \&\& (K < 16) \&\& (L < 16) \&\&
                   (M < 16) \&\& (N < 16) \&\& (O < 16) \&\& (P < 16));
    return mm512 setr epi32(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P);
                         D
                               E
                                               Н
                                                               K
                                                                          M
                                                                               Ν
                                                                                          P
              В
                                         G
```

CppCon 2020 - Adventures in SIMD Thinking

Copyright © 2020 Bob Steagall

Function rotate()

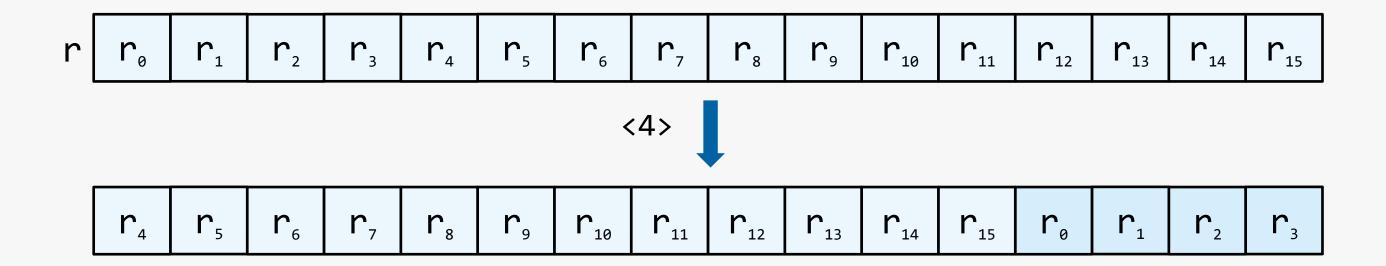


```
template<int R> KEWB_FORCE_INLINE rf_512
rotate(rf 512 r)
    if constexpr ((R % 16) == 0)
         return r;
    else
         constexpr int S = (R > 0) ? (16 - (R % 16)) : -R;
         constexpr int A = (S + 0) \% 16;
                           B = (S + 1) \% 16;
         constexpr int
         constexpr int
                           0 = (S + 14) \% 16;
         constexpr int P = (S + 15) \% 16;
         return _mm512_permutexvar_ps(_mm512_setr_epi32(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P), r);
                        r_{2}
                               r_{3}
                                      r_{11}
                                                                                           \mathbf{r}_{12}
                                            r_{5}
                                                   r_{6}
                                                          r_{7}
                                                                 r_{10}
                                                                                                   r_{13}
                                                        <3>
          \mathbf{r}_{13}
                 r_{15}
                               r
                                      r_{1}
                                             r_{2}
                                                   r_3
                                                          r_4
                                                                 r_{5}
                                                                               r_{7}
                                                                                     r<sub>9</sub>
                                                                                                   r_{10}
                                                                                                         r_{11}
                                                                                                                \mathbf{r}_{12}
```

Function rotate_down()



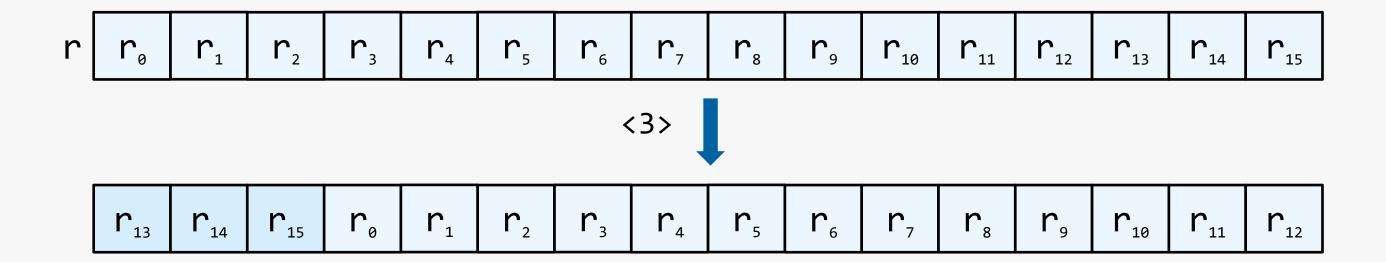
```
template<int R> KEWB_FORCE_INLINE rf_512
rotate_down(rf_512 r)
{
    static_assert(R >= 0);
    return rotate<-R>(r);
}
```



Function rotate_up()



```
template<int R> KEWB_FORCE_INLINE rf_512
rotate_up(rf_512 r)
{
    static_assert(R >= 0);
    return rotate<R>(r);
}
```



Function shift_down()



```
template<int S> KEWB_FORCE_INLINE rf_512
shift_down(rf_512 r)
     static_assert(S >= 0 && S <= 16);
     return blend(rotate_down<S>(r), load_value(0.0f), shift_down_blend_mask<S>());
                                                                                                       \mathbf{r}_{12}
                                   \mathbf{r}_{3}
                           r_{2}
                                           r_4
                                                                                        \Gamma_{10}
                                                                                                r_{11}
                                                                                                                \mathbf{r}_{13}
                                                              <4>
                                                                                                                                0
                                                                                                                0
                                                                                                         0
                                   \mathbf{r}_{7}
                                           r<sub>8</sub>
                                                  r_9
                                                                         \mathbf{r}_{12}
                                                                                                r<sub>15</sub>
```

Function shift_down_with_carry()



```
template<int S> KEWB FORCE INLINE rf 512
shift_down_with_carry(rf_512 a, rf_512 b)
       static_assert(S >= 0 && S <= 16);
       return blend(rotate_down<S>(a), rotate_down<S>(b), shift_down_blend_mask<S>());
                                    b_2
                                                                            b_6
         b
                         b_1
                                              b_3
                                                        b_{4}
                                                                  b_{5}
                                                                                      b_7
                                                                                                b<sub>8</sub>
                                                                                                          b_9
                                                                                                                   b_{10}
                                                                                                                             b_{11}
                                                                                                                                       b_{12}
                                                                                                                                                 b_{13}
                                                                                                                                                           b_{14}
                                                                                                                                                                     b_{15}
                                                                                                                  a<sub>10</sub>
        a
                                    a_2
                                              a_3
                                                                                      a<sub>7</sub>
                                                                                                          a
                                                                                                                             a<sub>11</sub>
                                                                                                                                       a<sub>12</sub>
                                                                                                                                                           a<sub>14</sub>
                         a_{\scriptscriptstyle 1}
                                                        a<sub>4</sub>
                                                                  a<sub>5</sub>
                                                                            a_{6}
                                                                                                a<sub>8</sub>
                                                                                                                                                 a<sub>13</sub>
                                                                                                                                                                     a<sub>15</sub>
                                                                                 <4>
                                                                                                                                                            b_2
                                                                                                                                        b_{\alpha}
                                                                                                                                                  b_1
                                                                                                                                                                      b_3
                                                                                                                   a<sub>14</sub>
                a<sub>4</sub>
                         a_{5}
                                    a_{6}
                                              a,
                                                        a<sub>8</sub>
                                                                  a
                                                                           a<sub>10</sub>
                                                                                     a<sub>11</sub>
                                                                                               a<sub>12</sub>
                                                                                                         a<sub>13</sub>
                                                                                                                             a<sub>15</sub>
```

Function shift_up()



21

```
template<int S> KEWB_FORCE_INLINE rf_512
shift_up(rf_512 r0)
     static_assert(S >= 0 && S <= 16);
     return blend(rotate_up<S>(r0), load_value(0.0f), shift_up_blend_mask<S>());
                                \mathbf{r}_{3}
                         r_{2}
                                       r_{7}
                                                                                 r_{10}
                                                                                        r_{11}
                                                                                               \mathbf{r}_{12}
                                                         <3>
                         0
           0
                  0
                                r
                                       r_{\scriptscriptstyle 1}
                                              \mathbf{r}_{2}
                                                                                                r_9
                                                                                                      r_{10}
```

Function shift_up_with_carry()



```
template<int S> KEWB FORCE INLINE rf 512
shift_up_with_carry(rf_512 lo, rf_512 hi)
       static_assert(S >= 0 && S <= 16);
       return blend(rotate_up<S>(lo), rotate_up<S>(hi), shift_up_blend_mask<S>());
                                   b_2
                                                                          b_6
                         b_1
                                             b_3
                                                       b_{4}
                                                                 b_{5}
                                                                                    b_7
                                                                                              b<sub>8</sub>
                                                                                                        b_9
                                                                                                                 b_{10}
                                                                                                                           b_{11}
                                                                                                                                     b_{12}
                                                                                                                                               b_{13}
                                                                                                                                                         b_{14}
                                                                                                                                                                  b_{15}
        b
                                                                                                                a<sub>10</sub>
        a
                                             a_3
                                   a_2
                                                                                    a<sub>7</sub>
                                                                                                        a<sub>9</sub>
                                                                                                                           a<sub>11</sub>
                                                                                                                                     a<sub>12</sub>
                                                                                                                                                         a<sub>14</sub>
                         a_{\scriptscriptstyle 1}
                                                       a<sub>4</sub>
                                                                 a<sub>5</sub>
                                                                          a_{6}
                                                                                              a<sub>8</sub>
                                                                                                                                               a<sub>13</sub>
                                                                                                                                                                  a<sub>15</sub>
                                                                               <3>
                                                                                                                 b_7
                                                                                                                                               b_{10}
                                                                 b_2
                                                                          b_3
                                                                                                        b_6
                                                                                                                            b<sub>8</sub>
                                                                                                                                     b<sub>9</sub>
                                                                                                                                                         b_{11}
                                             b_{\alpha}
                                                       b_1
                                                                                    b_4
                                                                                              b_{5}
                                                                                                                                                                   b_{12}
                        a<sub>14</sub>
                                  a<sub>15</sub>
               a<sub>13</sub>
```

Function in_place_shift_down_with_carry()



```
template<int S> KEWB FORCE INLINE void
in_place_shift_down_with_carry(rf_512& a, rf_512& b)
      static assert(S \ge 0 \&\& S <= 16);
      constexpr msk_512  zmask = (0xFFFFu >> (unsigned) S);
      constexpr msk_512 bmask = ~zmask & 0xFFFFu;
                                    perm = make_shift_permutation<S, bmask>();
      ri_512
      a = _mm512_permutex2var_ps(a, perm, hi);
      b = mm512_maskz_permutex2var_ps((__mmask16) zmask, b, perm, b);
                            b<sub>2</sub>
                                     b_3
                                                             b_6
                    b_1
                                             b_{A}
                                                     b_{5}
                                                                     b_{7}
                                                                             b<sub>8</sub>
                                                                                     b
                                                                                            b_{10}
                                                                                                     b_{11}
                                                                                                            b_{12}
                                                                                                                     b_{13}
                                                                                                                             b_{14}
                                                                                                                                     b_{15}
       b
                                                                                            a<sub>10</sub>
       a
            a
                    a_{\scriptscriptstyle 1}
                            a_2
                                    a_3
                                             a<sub>4</sub>
                                                     a_{5}
                                                            a_{6}
                                                                     a,
                                                                             a_8
                                                                                     a_9
                                                                                                     a<sub>11</sub>
                                                                                                            a<sub>12</sub>
                                                                                                                     a<sub>13</sub>
                                                                                                                             a<sub>14</sub>
                                                                                                                                     a<sub>15</sub>
                                                                  <4>
       b
                            b_6
                    b<sub>5</sub>
                                    b_7
                                             b
                                                     b
                                                            b_{10}
                                                                    b_{11}
                                                                                    b_{13}
                                                                                            b_{14}
                                                                                                     b_{15}
                                                                                                             0
                                                                                                                     0
                                                                                                                                     0
            b_{A}
                                                                            b_{12}
                                                                                                                             b_2
                                                                                                                     b_1
                                                                                                                                     b_3
                                                           a<sub>10</sub>
                                                                            a<sub>12</sub>
                                                                                            a<sub>14</sub>
                                                                                                             b
       a
                                    a<sub>7</sub>
                            a_{6}
                                             a
                                                     a<sub>9</sub>
                                                                                    a<sub>13</sub>
                                                                                                     a<sub>15</sub>
                    a_{5}
                                                                    a_{11}
```

Function fused_multiply_add()



```
KEWB FORCE INLINE rf 512
fused_multiply_add(rf_512 a, rf_512 b, rf_512 c)
         return _mm512_fmadd_ps(a, b, c);
                                                                                                                                          a<sub>10</sub>
                                                                                                                                                                                          a<sub>14</sub>
          a
                               a_{\scriptscriptstyle 1}
                                           a_2
                                                       a_3
                                                                               a_{5}
                                                                                                       a,
                                                                                                                               a<sub>9</sub>
                                                                                                                                                       a<sub>11</sub>
                                                                                                                                                                   a<sub>12</sub>
                                                                   a<sub>4</sub>
                                                                                           a_{6}
                                                                                                                   a<sub>8</sub>
                                                                                                                                                                               a<sub>13</sub>
                                                                                                                                                                                                       a<sub>15</sub>
           b
                               b_1
                                           b_2
                                                       b_3
                                                                   b_4
                                                                               b_{5}
                                                                                           b_6
                                                                                                       b_7
                                                                                                                   b<sub>8</sub>
                                                                                                                               b_9
                                                                                                                                          b_{10}
                                                                                                                                                      b_{11}
                                                                                                                                                                  b_{12}
                                                                                                                                                                               b_{13}
                                                                                                                                                                                           b_{14}
                                                                                                                                                                                                       b_{15}
                   b
           C
                               \mathsf{C}_{\scriptscriptstyle 1}
                                           \mathbf{C}_2
                                                       C<sub>3</sub>
                                                                   C_4
                                                                               C<sub>5</sub>
                                                                                           C_6
                                                                                                       C<sub>7</sub>
                                                                                                                                          C<sub>10</sub>
                                                                                                                                                       C<sub>11</sub>
                                                                                                                                                                  C<sub>12</sub>
                                                                                                                                                                               C<sub>13</sub>
                                                                                                                   C<sub>8</sub>
                                                                                                                               C_9
                 d_n = a_n b_n + c_n
                                                                                                                                                      d_{11}
                                                                                                                                                                  d_{12}
                                                       d_3
                                                                   d_{4}
                                                                               d_{5}
                                                                                           d_{6}
                                                                                                       d_{7}
                                                                                                                               d
                                                                                                                                          d_{10}
                                                                                                                                                                               d_{13}
                   d_{\alpha}
                               d_{\scriptscriptstyle 1}
                                           d_2
                                                                                                                   d<sub>8</sub>
```

Function minimum()



```
KEWB_FORCE_INLINE rf_512
minimum(rf_512 a, rf_512 b)
        return _mm512_min_ps(a, b);
         a
                           a_{\scriptscriptstyle 1}
                                      a_2
                                                 a_3
                                                            a_4
                                                                      a<sub>5</sub>
                                                                                 a_{6}
                                                                                           a,
                                                                                                                 a<sub>9</sub>
                                                                                                                           a<sub>10</sub>
                                                                                                                                     a<sub>11</sub>
                                                                                                                                                a<sub>12</sub>
                                                                                                                                                                     a<sub>14</sub>
                                                                                                      a<sub>8</sub>
                                                                                                                                                           a<sub>13</sub>
                                                                                                                                                                                a<sub>15</sub>
         b
                           b_{1}
                                      b_2
                                                 b_3
                                                           b_4
                                                                      b_{5}
                                                                                 b_6
                                                                                           b_7
                                                                                                      b<sub>8</sub>
                                                                                                                 b_9
                                                                                                                           b_{10}
                                                                                                                                     b_{11}
                                                                                                                                                b_{12}
                                                                                                                                                           b_{13}
                                                                                                                                                                     b_{14}
                                                                                                                                                                                b_{15}
                 b<sub>o</sub>
               d_n = min(a_n, b_n)
                                                                                                                                               d_{12}
                d_{o}
                                      d_2
                                                                      d_{5}
                                                                                d_{6}
                                                                                           d_{7}
                                                                                                                          d_{10}
                                                                                                                                     d_{11}
                                                                                                                                                          d_{13}
                                                                                                                                                                     d_{14}
                           d_{1}
                                                 d_3
                                                           d_4
                                                                                                      d<sub>8</sub>
                                                                                                                d_9
```

Function maximum()



```
KEWB_FORCE_INLINE rf_512
maximum(rf_512 a, rf_512 a)
        return _mm512_max_ps(a, b);
         a
                           a_{\scriptscriptstyle 1}
                                      a_2
                                                 a_3
                                                            a_4
                                                                      a<sub>5</sub>
                                                                                 a_{6}
                                                                                           a,
                                                                                                                 a<sub>9</sub>
                                                                                                                           a<sub>10</sub>
                                                                                                                                      a<sub>11</sub>
                                                                                                                                                a<sub>12</sub>
                                                                                                                                                                      a<sub>14</sub>
                                                                                                      a<sub>8</sub>
                                                                                                                                                           a<sub>13</sub>
                                                                                                                                                                                a<sub>15</sub>
         b
                                                                                 b_6
                           b_{1}
                                      b_2
                                                 b_3
                                                            b_4
                                                                      b_{5}
                                                                                           b_7
                                                                                                      b<sub>8</sub>
                                                                                                                 b_9
                                                                                                                           b_{10}
                                                                                                                                     b_{11}
                                                                                                                                                b_{12}
                                                                                                                                                           b_{13}
                                                                                                                                                                      b_{14}
                                                                                                                                                                                b_{15}
                 b<sub>o</sub>
               d_n = \max(a_n, b_n)
                                                                                                                                                d_{12}
                d_{o}
                                      d_2
                                                                      d_{5}
                                                                                 d_{6}
                                                                                           d_{7}
                                                                                                                           d_{10}
                                                                                                                                     d_{11}
                                                                                                                                                           d_{13}
                                                                                                                                                                     d_{14}
                           d_{1}
                                                 d_3
                                                           d_4
                                                                                                      d<sub>8</sub>
                                                                                                                d_9
```



```
KEWB_FORCE_INLINE rf_512
compare_with_exchange(rf_512 vals, ri_512 perm, msk_512 mask)
{
    rf_512 exch = permute(vals, perm);
    rf_512 vmin = minimum(vals, exch);
    rf_512 vmax = maximum(vals, exch);
    return blend(vmin, vmax, mask);
}
```



vals	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
perm	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15
mask	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0



```
KEWB_FORCE_INLINE rf_512
compare_with_exchange(rf_512 vals, ri_512 perm, msk_512 mask)
{
    rf_512    exch = permute(vals, perm);
    rf_512    vmin = minimum(vals, exch);
    rf_512    vmax = maximum(vals, exch);
    return blend(vmin, vmax, mask);
}
```



r <mark>vals</mark>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
- <mark>perm</mark>	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15
mask	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
→ <mark>exch</mark>	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0



```
KEWB_FORCE_INLINE rf_512
compare_with_exchange(rf_512 vals, ri_512 perm, msk_512 mask)
{
    rf_512    exch = permute(vals, perm);
    rf_512    vmin = minimum(vals, exch);
    rf_512    vmax = maximum(vals, exch);
    return blend(vmin, vmax, mask);
}
```



┌ <mark>vals</mark>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
perm	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	1 5
mask	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
<pre>- exch</pre>	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
 <mark>∨min</mark>	14	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0



```
KEWB_FORCE_INLINE rf_512
compare_with_exchange(rf_512 vals, ri_512 perm, msk_512 mask)
{
    rf_512 exch = permute(vals, perm);
    rf_512 vmin = minimum(vals, exch);
    rf_512 vmax = maximum(vals, exch);
    return blend(vmin, vmax, mask);
}
```



<pre>rals</pre>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
perm	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15
mask	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
						<u>,</u>										
– <mark>exch</mark>	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vmin	14	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
→ <mark>vmax</mark>	15	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0



```
KEWB_FORCE_INLINE rf_512
compare_with_exchange(rf_512 vals, ri_512 perm, msk_512 mask)
{
    rf_512    exch = permute(vals, perm);
    rf_512    vmin = minimum(vals, exch);
    rf_512    vmax = maximum(vals, exch);
    return blend(vmin, vmax, mask);
}
```



vals	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
perm	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15
- <mark>mask</mark>	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
exch	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
- <mark>vmin</mark>	14	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
- <mark>vmax</mark>	15	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Function compare_with_exchange()

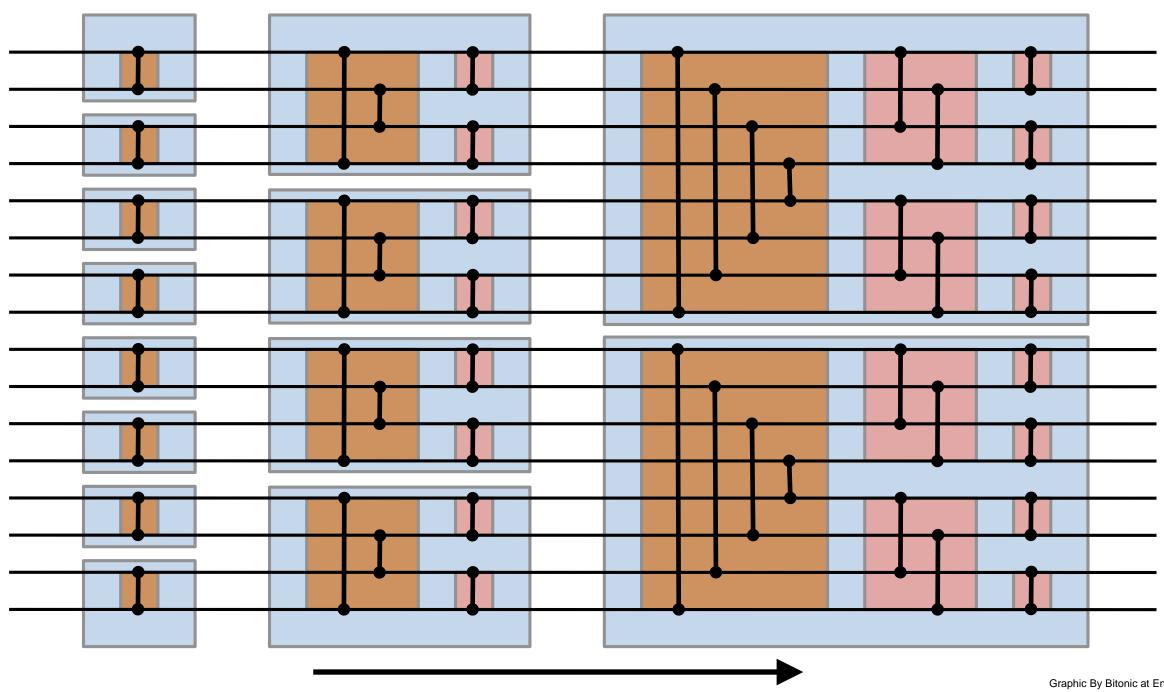


vals	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
perm	1	0	2	3	4	5	6	7	8	9	10	11	12	13	14	15
mask	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
exch	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vmin	14	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vmax	15	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	14	15	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Intra-Register Sorting

Sorting Network for Two Lanes of Eight

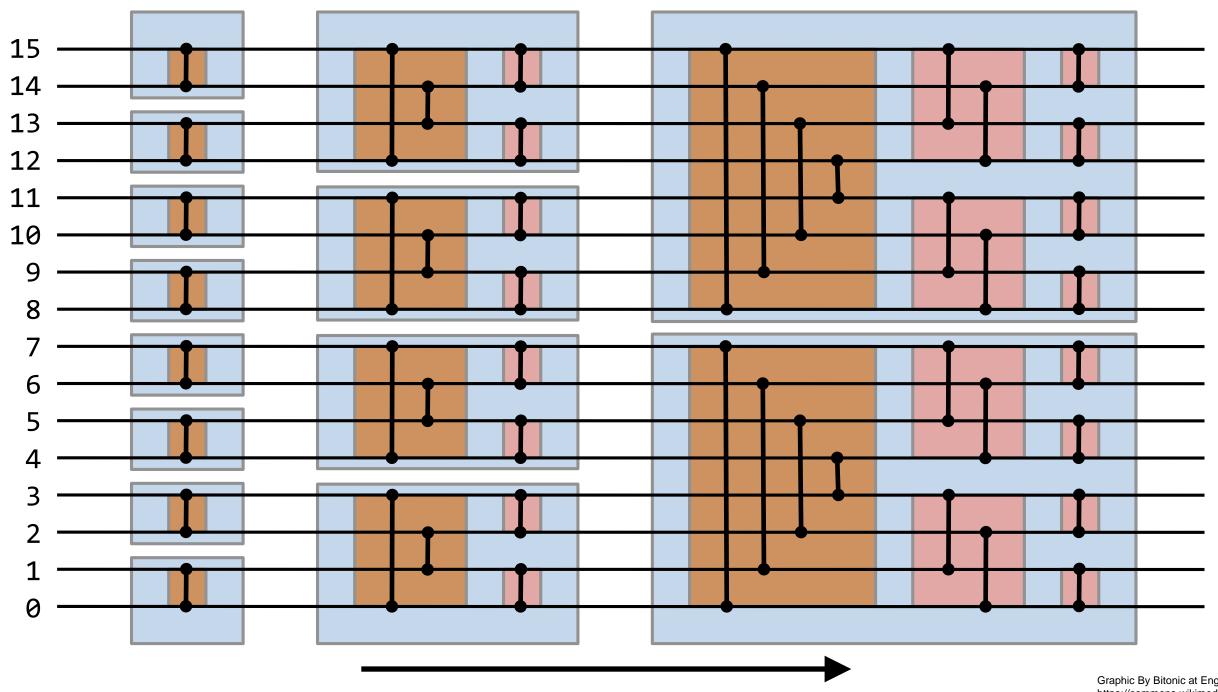




Graphic By Bitonic at English Wikipedia - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=21961917

Sorting Network for Two Lanes of Eight





Graphic By Bitonic at English Wikipedia - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=21961917



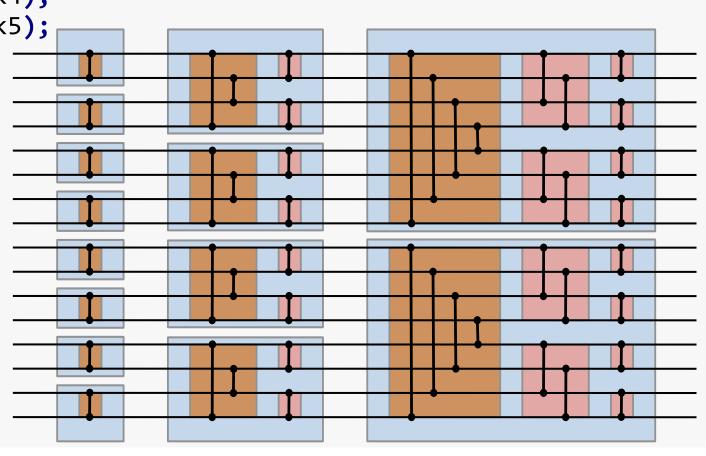
```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   //- Precompute the permutations and bitmasks for the 6 stages of this bitonic sorting sequence.
                                        1 2 3 4 5 6 7
                                                                 0 1 2 3 4 5 6 7
   ri_512 const perm0 = make_perm_map<1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14>();
   constexpr msk_512 mask0 = make_bit_mask<0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1>();
   ri_512 const perm1 = make_perm_map<3, 2, 1, 0, 7, 6, 5, 4, 11, 10, 9, 8, 15, 14, 13, 12>();
   constexpr msk_512 mask1 = make_bit_mask<0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1>();
   ri_512 const perm2 = make_perm_map<1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14>();
   constexpr msk_512 mask2 = make_bit_mask<0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1>();
   ri_512 const perm3 = make_perm_map<7, 6, 5, 4, 3, 2, 1, 0, 15, 14, 13, 12, 11, 10, 9, 8>();
   constexpr msk_512 mask3 = make_bit_mask<0, 0, 0, 0, 1, 1, 1,
                                                              0, 0, 0, 0, 1, 1, 1, 1>();
   ri_512 const perm4 = make_perm_map<2, 3, 0, 1, 6, 7, 4, 5, 10, 11, 8, 9, 14, 15, 12, 13>();
   constexpr msk_512 mask4 = make_bit_mask<0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1>();
   ri 512 const
                  perm5 = make_perm_map<1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14>();
   constexpr msk_512 mask5 = make_bit_mask<0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1>();
   . . .
```



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   //- Precompute the permutations and bitmasks for the 6 stages of this bitonic sorting sequence.
                    perm0 = make_perm_map<1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14>();
   ri 512 const
   constexpr msk_512 mask0 = make_bit_mask<0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1>();
   constexpr msk 512 mask2 = make bit ma
   constexpr msk 512 mask4 = make bit mask<@</pre>
   ri_512 const perm5 = make_perm_map<1, 0, 3, 2, 5, 4, 7, 6, 9, 8, 11, 10, 13, 12, 15, 14>();
```



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   vals = compare_with_exchange(vals, perm0, mask0);
   vals = compare_with_exchange(vals, perm1, mask1);
   vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
   vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```

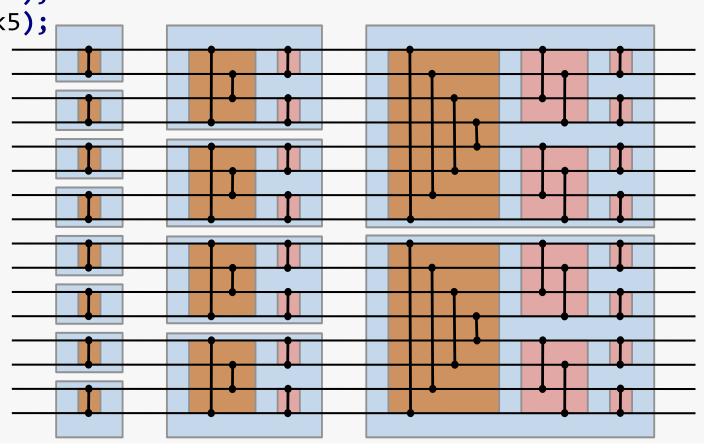




vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
------	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	--



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
    vals = compare_with_exchange(vals, perm0, mask0);
    vals = compare_with_exchange(vals, perm1, mask1);
   vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
   vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```

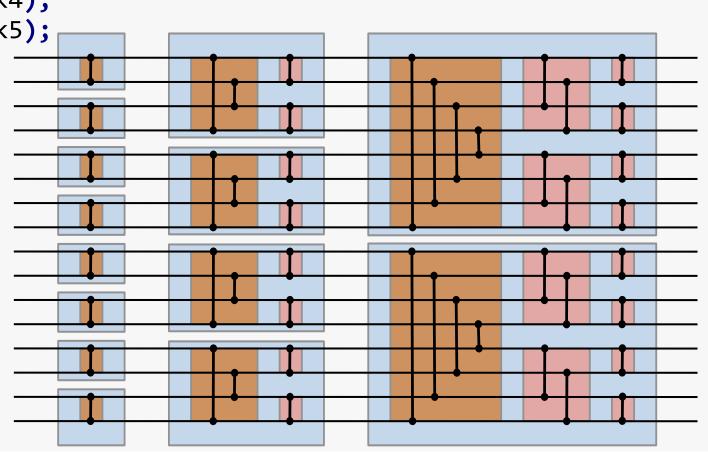




vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
    vals = compare_with_exchange(vals, perm0, mask0);
    vals = compare_with_exchange(vals, perm1, mask1);
    vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
   vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```

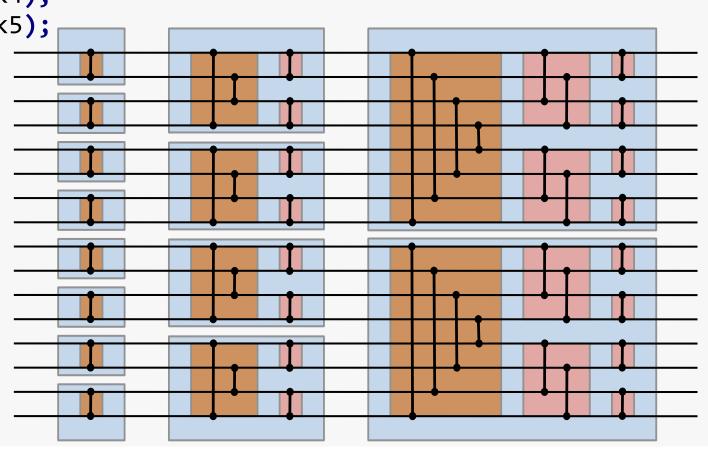




vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2
vals ₁	14	13	16	15	10	9	12	11	6	5	8	7	2	1	4	3



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
    vals = compare_with_exchange(vals, perm0, mask0);
    vals = compare_with_exchange(vals, perm1, mask1);
    vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
   vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```





vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2
$vals_1$	14	13	16	15	10	9	12	11	6	5	8	7	2	1	4	3
vals ₂	13	14	15	16	9	10	11	12	5	6	7	8	1	2	3	4



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   vals = compare_with_exchange(vals, perm0, mask0);
   vals = compare_with_exchange(vals, perm1, mask1);
    vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
    vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```



vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2
vals ₁	14	13	16	15	10	9	12	11	6	5	8	7	2	1	4	3
vals ₂	13	14	15	16	9	10	11	12	5	6	7	8	1	2	3	4
vals ₃	12	11	10	9	16	15	14	13	4	3	2	1	8	7	6	5



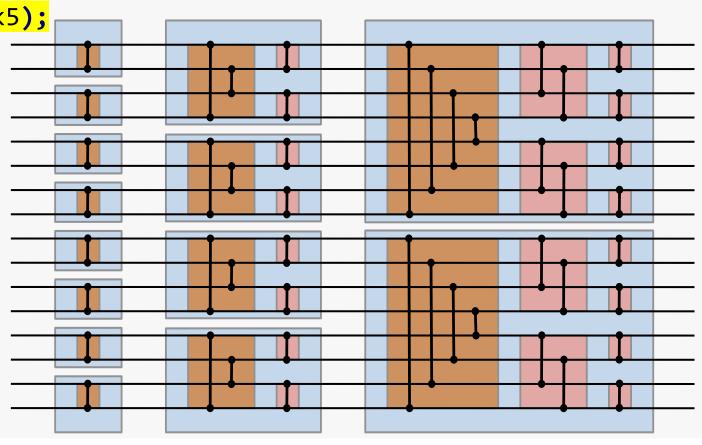
```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   vals = compare_with_exchange(vals, perm0, mask0);
   vals = compare_with_exchange(vals, perm1, mask1);
   vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
   vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```



vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2
vals ₁	14	13	16	15	10	9	12	11	6	5	8	7	2	1	4	3
vals ₂	13	14	15	16	9	10	11	12	5	6	7	8	1	2	3	4
vals ₃	12	11	10	9	16	15	14	13	4	3	2	1	8	7	6	5
vals ₄	10	9	12	11	14	13	16	15	2	1	4	3	6	5	8	7



```
KEWB FORCE INLINE rf 512
sort_two_lanes_of_8(rf_512 vals)
   vals = compare_with_exchange(vals, perm0, mask0);
   vals = compare_with_exchange(vals, perm1, mask1);
   vals = compare_with_exchange(vals, perm2, mask2);
   vals = compare_with_exchange(vals, perm3, mask3);
    vals = compare_with_exchange(vals, perm4, mask4);
    vals = compare_with_exchange(vals, perm5, mask5);
    return vals;
```





vals	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
vals ₀	15	16	13	14	11	12	9	10	7	8	5	6	3	4	1	2
vals ₁	14	13	16	15	10	9	12	11	6	5	8	7	2	1	4	3
vals ₂	13	14	15	16	9	10	11	12	5	6	7	8	1	2	3	4
vals ₃	12	11	10	9	16	15	14	13	4	3	2	1	8	7	6	5
$vals_4 \Big $	10	9	12	11	14	13	16	15	2	1	4	3	6	5	8	7
vals ₅	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8

Fast Linear Median Filter

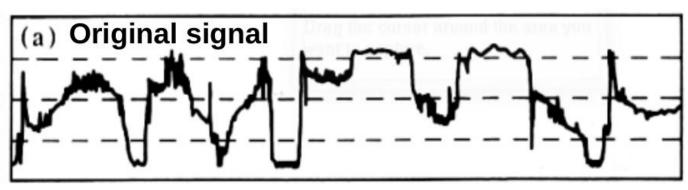
Fast Median Filter?

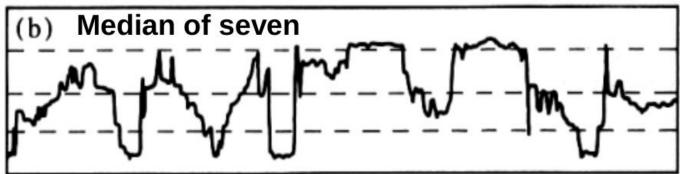


- If we can sort two lanes of eight, why not two lanes of seven?
- With a fast sort, we could implement a fast median-of-seven linear filter
 - median: a value separating the higher half from the lower half of a data sample
 - If I have an array a of seven integers, and they are sorted, the median is a [3]
- Median filters are good at
 - Preserving edge features in a signal
 - Eliminating outliers without blur
 - Preserving large discontinuities
 - De-noising











```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
   rf 512 prev; //- Bottom of the input data window
   rf_512 curr; //- Middle of the input data window
   rf_512 next; //- Top of the input data window
   rf_512 lo; //- Primary work register
   rf 512 hi; //- Upper work data register; feeds values into the top of 'lo'
   rf 512 data; //- Holds output prior to store operation
    rf 512 work; //- Accumulator
   rf 512 const first = load_value(psrc[0]);
   //- This permutation specifies how to load the two lanes of 7.
    //
    ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
   //- This permutation specifies which elements to save.
    ri_512 const save_perm = make_perm_map<3,11,3,11,3,11,3,11,3,11,3,11,3,11>();
    • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
   rf 512 prev; //- Bottom of the input data window
   rf_512 curr; //- Middle of the input data window
   rf_512 next; //- Top of the input data window
   rf_512 lo; //- Primary work register
   rf 512 hi; //- Upper work data register; feeds values into the top of 'lo'
   rf 512 data; //- Holds output prior to store operation
    rf 512 work; //- Accumulator
   rf 512 const first = load value(psrc[0]);
   //- This permutation specifies how to load the two lanes of 7.
    //
    ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
   //- This permutation specifies which elements to save.
    ri_512 const save_perm = make_perm_map<3,11,3,11,3,11,3,11,3,11,3,11,3,11>();
    • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    rf 512 prev; //- Bottom of the input data window
   rf_512 curr; //- Middle of the input data window
   rf_512 next; //- Top of the input data window
   rf_512 lo; //- Primary work register
   rf 512 hi; //- Upper work data register; feeds values into the top of 'lo'
   rf 512 data; //- Holds output prior to store operation
    rf 512 work; //- Accumulator
    rf_512 const first = load_value(psrc[0]);
   //- This permutation specifies how to load the two lanes of 7.
    ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
   //- This permutation specifies which elements to save.
    ri_512 const save_perm = make_perm_map<3,11,3,11,3,11,3,11,3,11,3,11,3,11>();
    • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
   rf 512 prev; //- Bottom of the input data window
   rf_512 curr; //- Middle of the input data window
   rf_512 next; //- Top of the input data window
   rf_512 lo; //- Primary work register
   rf 512 hi; //- Upper work data register; feeds values into the top of 'lo'
   rf 512 data; //- Holds output prior to store operation
   rf 512 work; //- Accumulator
   rf_512 const first = load_value(psrc[0]);
   //- This permutation specifies how to load the two lanes of 7.
   ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
   //- This permutation specifies which elements to save.
   • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    . . .
    //- This permutation specifies how to load the two lanes of 7.
    ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
    //- This permutation specifies which elements to save.
    ri_512 const save_perm = make_perm_map<3,11,3,11,3,11,3,11,3,11,3,11,3,11>();
    //- This is a bitmask pattern for picking out adjacent elements.
    constexpr msk_512 save = make_bit_mask<1,1>();
    //- This array of bitmasks specifies which pair of elements to blend into the result.
    constexpr msk_512 save_mask[8] = {save << 0, save << 2, save << 4, save << 6,
                                       save << 8, save << 10, save << 12, save << 14};
    • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    . . .
    //- This permutation specifies how to load the two lanes of 7.
    ri_512 const load_perm = make_perm_map<0,1,2,3,4,5,6,7,1,2,3,4,5,6,7,8>();
    //- This permutation specifies which elements to save.
    ri_512 const save_perm = make_perm_map<3,11,3,11,3,11,3,11,3,11,3,11,3,11>();
    //- This is a bitmask pattern for picking out adjacent elements.
    constexpr msk_512 save = make_bit_mask<1,1>();
    //- This array of bitmasks specifies which pair of elements to blend into the result.
    constexpr msk_512 save_mask[8] = {save << 0, save << 2, save << 4, save << 6,
                                       save << 8, save << 10, save << 12, save << 14};
    • • •
```



```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    • • •
    //- Preload the initial input data window
    size_t read = 0;
    size_t wrote = 0;
    curr = first;
    next = load_from(psrc);
    read += 16;
```

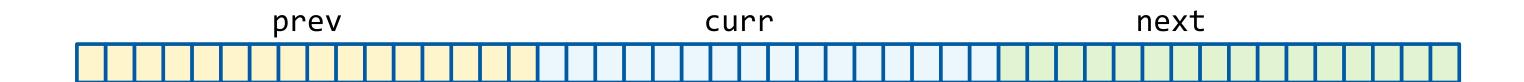






```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf_len + 16))</pre>
        prev = curr;
        curr = next;
        next = load_from(psrc + read);
        read += 16;
        lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
        hi = shift up with carry<3>(curr, next); // correct offset in the input data window
        for (int i = 0; i < 8; ++i)
                                      //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
            data = masked_permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
        store to(pdst + wrote, data);
        wrote += 16;
```

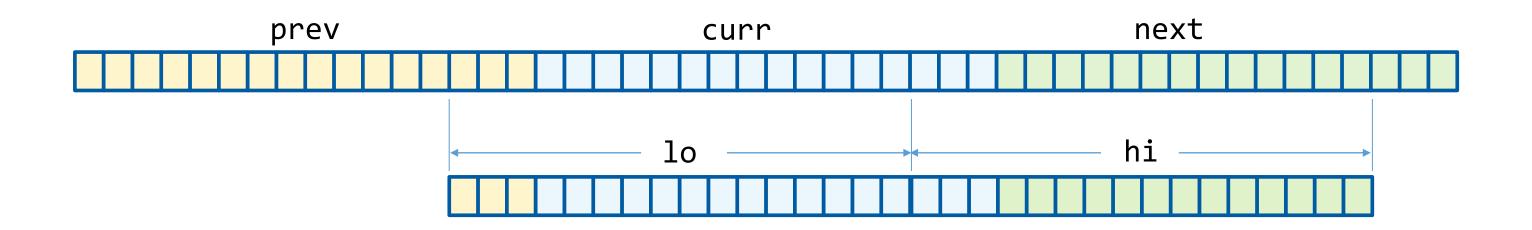






```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift_up_with_carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift_up_with_carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked_permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
        store to(pdst + wrote, data);
       wrote += 16;
```

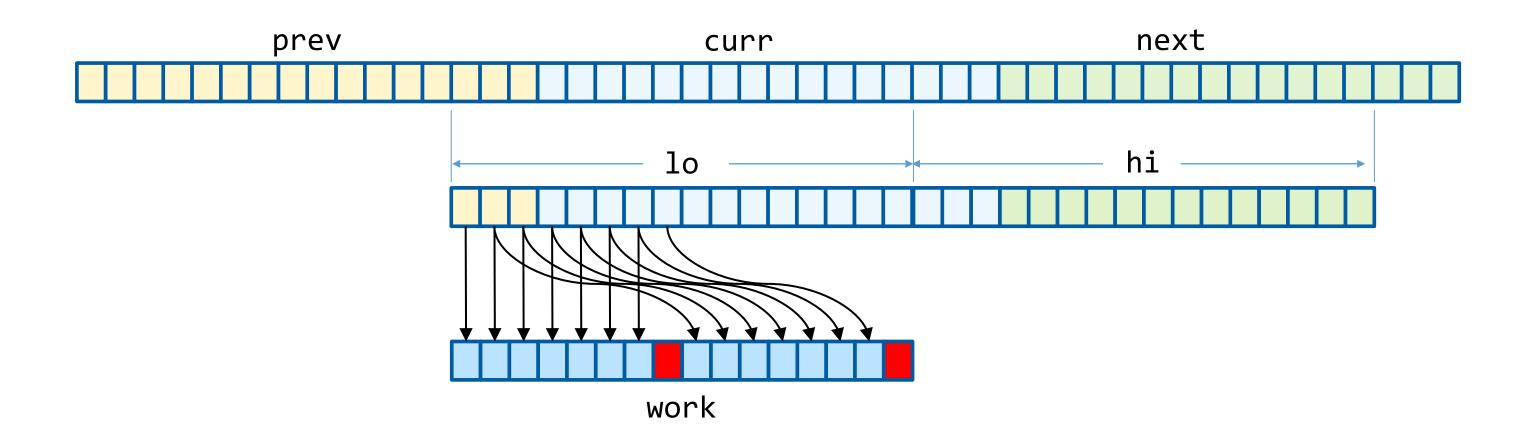






```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift up with carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked_permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
        store to(pdst + wrote, data);
       wrote += 16;
```

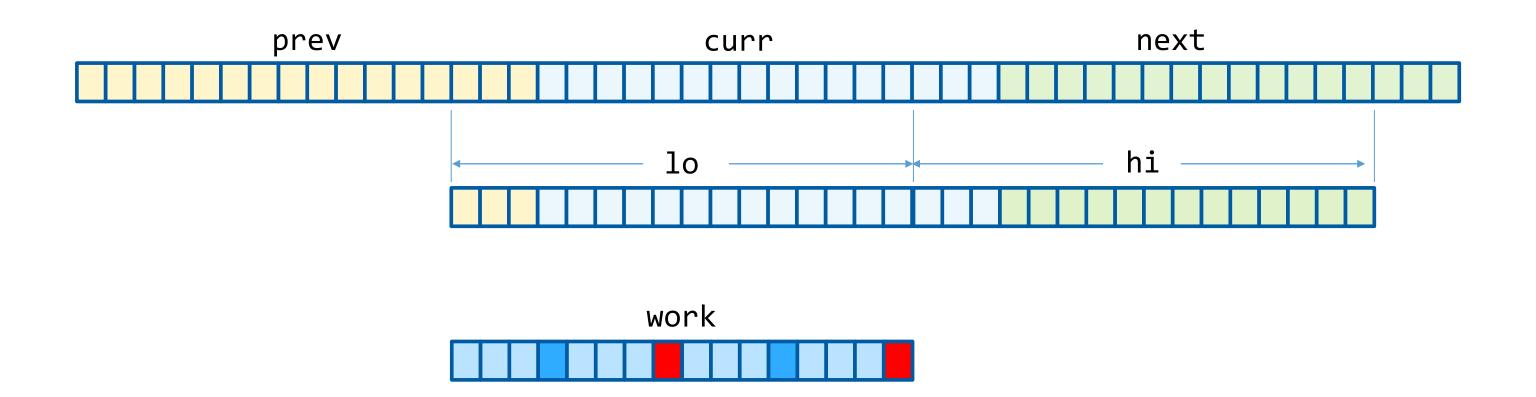






```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift up with carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked_permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
        store to(pdst + wrote, data);
       wrote += 16;
```

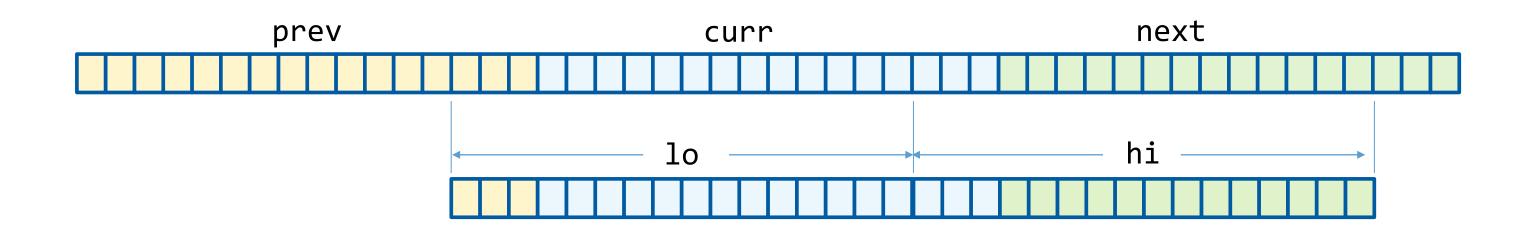


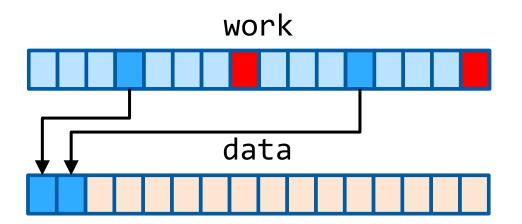




```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift up with carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked_permute(data, work, save_perm, save_mask[i]);
           in place shift down with carry<2>(lo, hi);
        store to(pdst + wrote, data);
       wrote += 16;
```



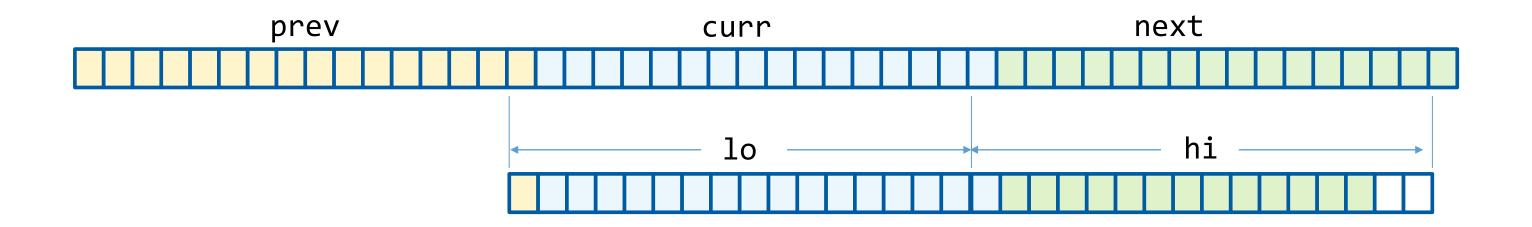




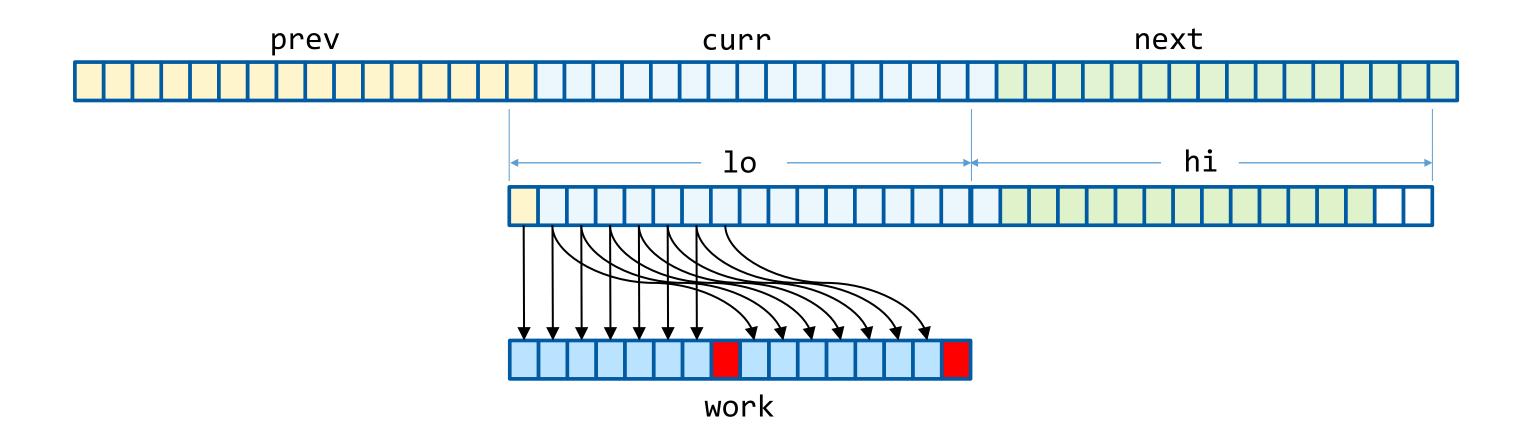


```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift up with carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
        store to(pdst + wrote, data);
       wrote += 16;
```

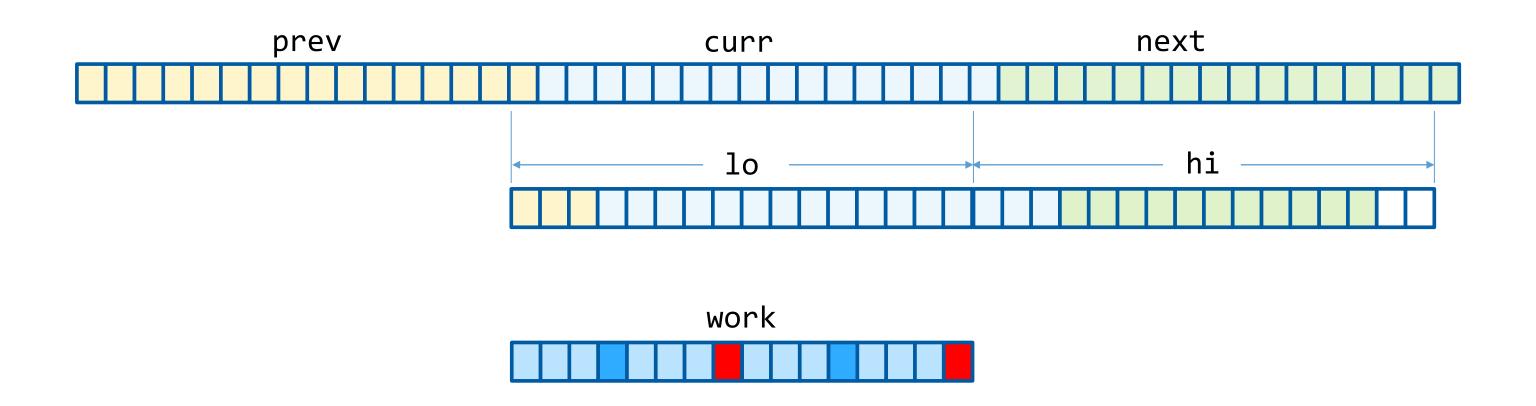




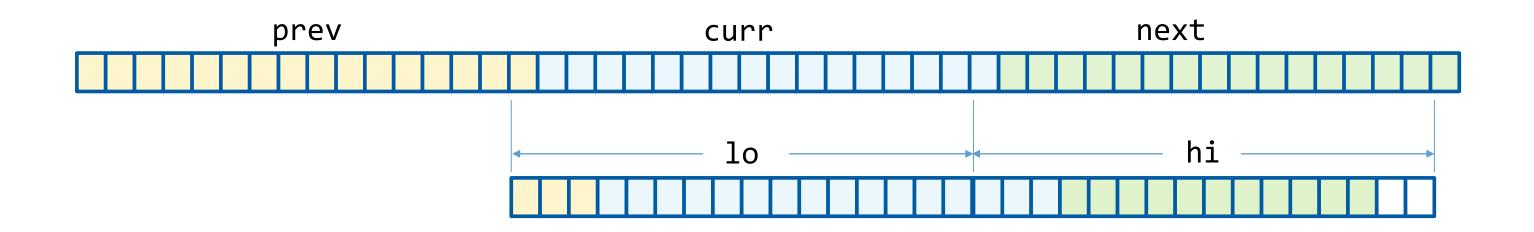


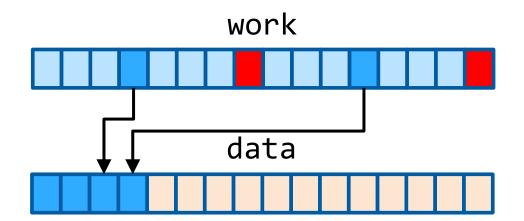














```
void
avx_median_of_7(float* pdst, float const* psrc, size_t const buf_len)
    while (used < (buf len + 16))
       prev = curr;
       curr = next;
       next = load from(psrc + read);
       read += 16;
       lo = shift up with carry<3>(prev, curr); //- Init the work data registers to the
       hi = shift up with carry<3>(curr, next); // correct offset in the input data window
       for (int i = 0; i < 8; ++i)
                                     //- Perform two sorts of 7 at a time, in lanes of 8
           work = permute(lo, load perm);
           work = sort two lanes of 7(work);
           data = masked_permute(data, work, save_perm, save_mask[i]);
           in_place_shift_down_with_carry<2>(lo, hi);
       store_to(pdst + wrote, data);
       wrote += 16;
```

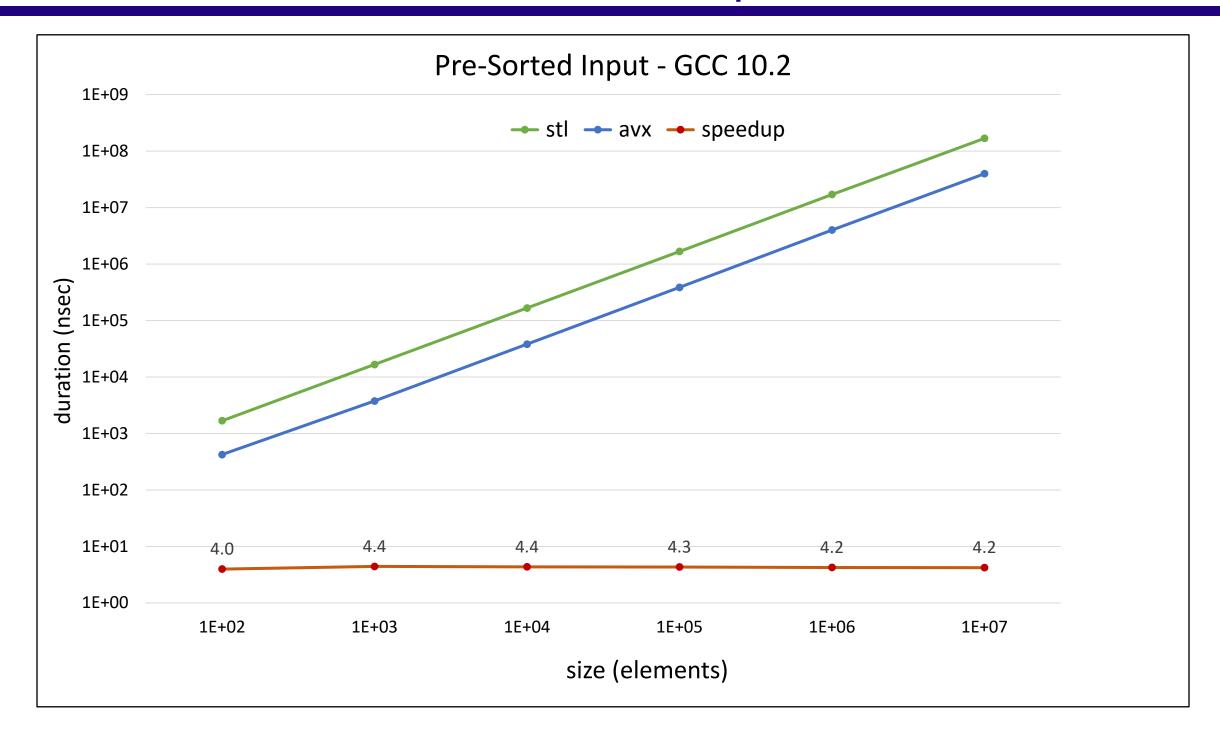
Testing Methodology – Platforms



- Ubuntu 18.04 on Cascade Lake
 - GCC 10.2, all code compiled with -O3 -mavx512 -march=skylake
 - Clang 10.0.1, all code compiled with -03 -mavx512 -march=skylake
- Element counts of 1E02 through 1E07 (by 10s)
 - Pre-sorted integers increasing monotonically
 - Randomly generated signed integers [-50, +50]
 - Always generated from the same seed
- Collect timings for each combination using two approaches
 - STL-based, performing std::sort() for every sequence of seven elements
 - Small-Kernel AVX

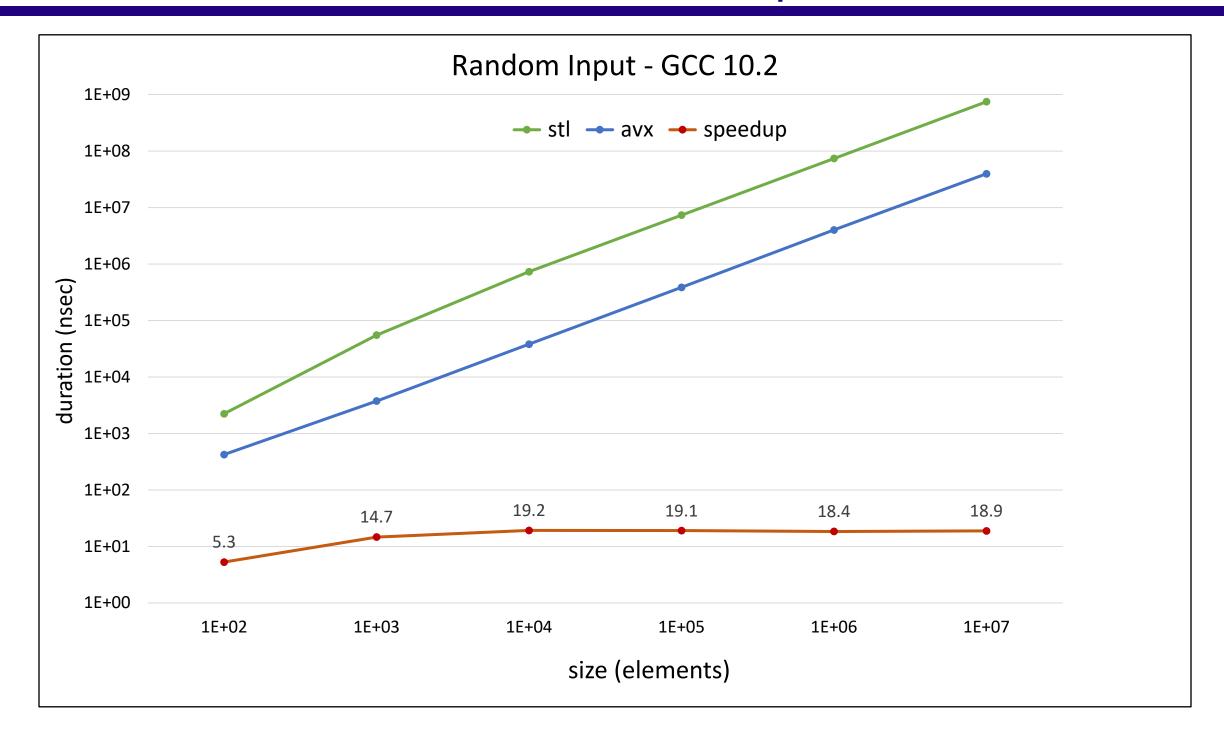
Results – Median-of-Seven – Sorted Input – GCC





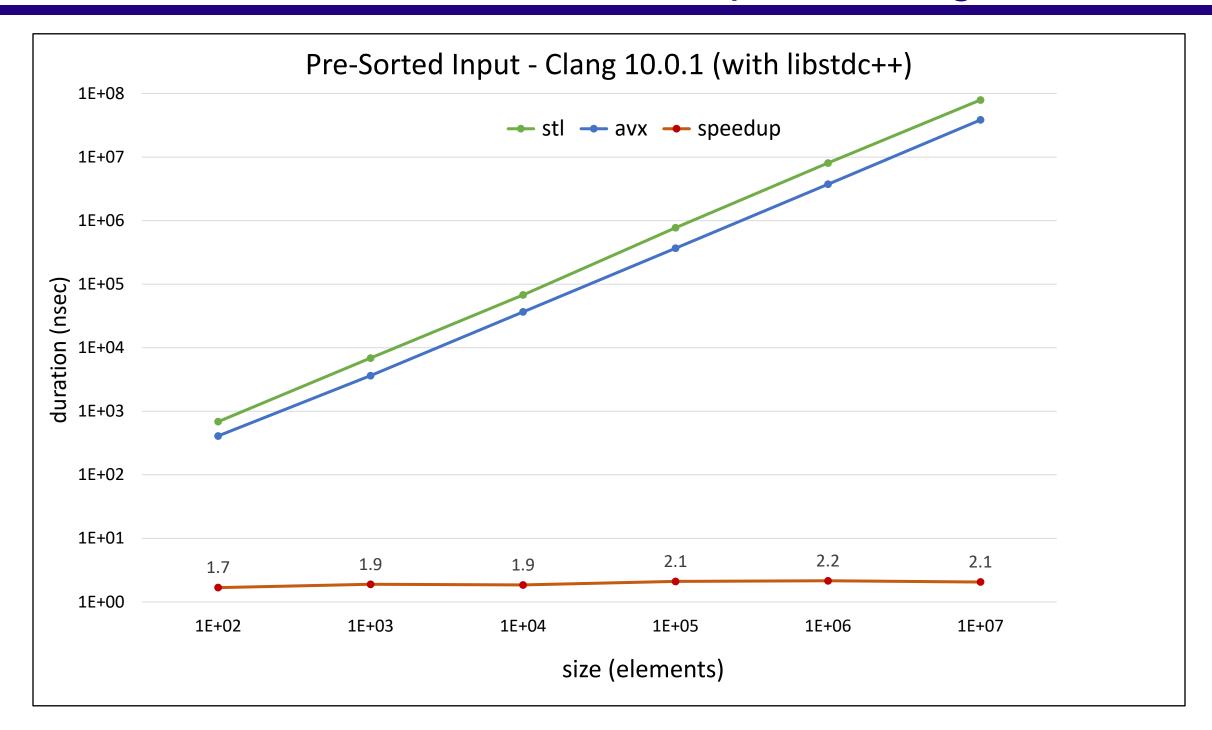
Results – Median-of-Seven – Random Input – GCC





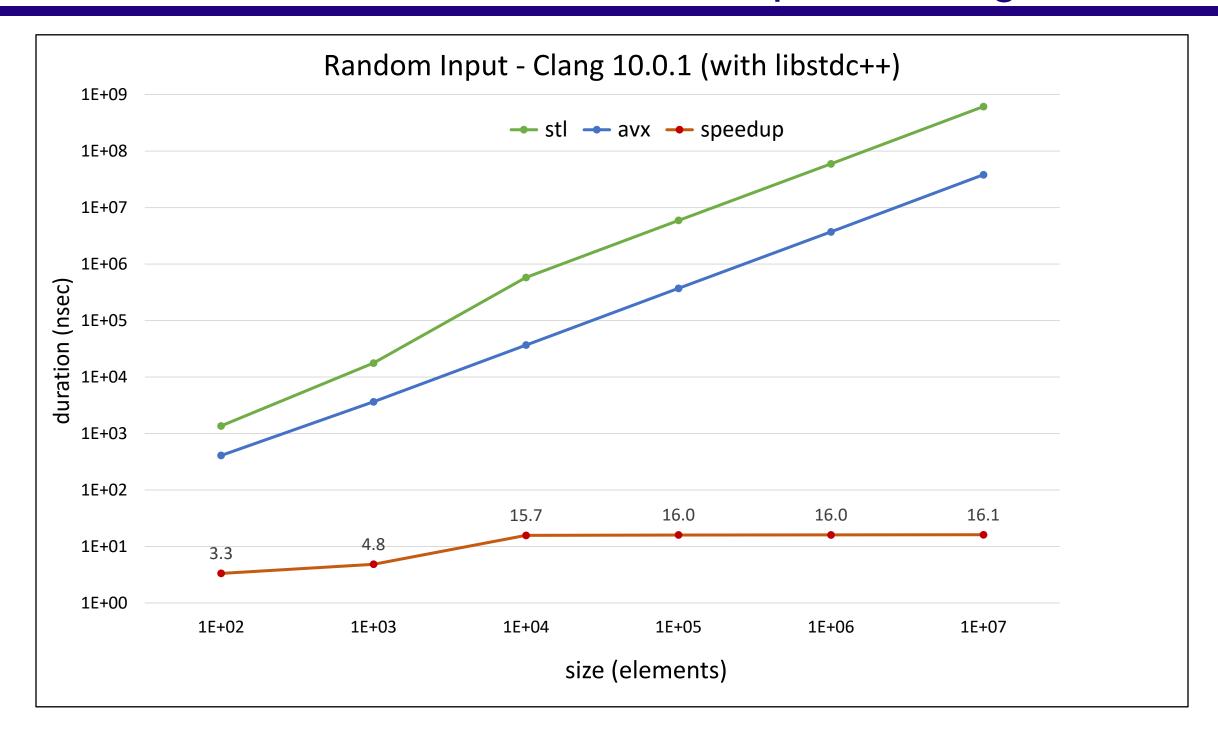
Results - Median-of-Seven - Sorted Input - Clang





Results – Median-of-Seven – Random Input – Clang





Thank You for Attending!

Talk: github.com/BobSteagall/CppCon2020

Blog: bobsteagall.com