VW-SLP: Auto-Vectorization with Adaptive Vector Width

Vasileios Porpodas ¹ Rodrigo C. O. Rocha ² Luís F. W. Góes ³

¹Intel Santa Clara, USA

²University of Edinburgh, UK

³PUC Minas, Brazil

PACT 2018









Legal Disclaimer & Optimization Notice

Performance results are based on testing as of 10/16/2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Copyright 2018, Intel Corporation. All rights reserved. Intel, the Intel logo, Pentium, Xeon, Core, VTune, OpenVINO, Cilk, are trademarks of Intel Corporation or its subsidiaries in the U.S. and other countries.

Optimization Notice

Intels compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microprocessors of Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

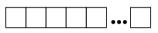
Notice revision #20110804



 Wikipedia: In Computer Science a vector is "A one-dimensional array"

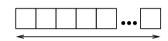


 Wikipedia: In Computer Science a vector is "A one-dimensional array"





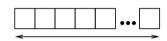
 Wikipedia: In Computer Science a vector is "A one-dimensional array"



Vector-Width is the size of the vector



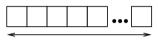
 Wikipedia: In Computer Science a vector is "A one-dimensional array"



- Vector-Width is the size of the vector
- Aka Vector Length (VL) or Vector Factor (VF)



- Wikipedia: In Computer Science a vector is "A one-dimensional array"
- Vector-Width is the size of the vector
- Aka Vector Length (VL) or Vector Factor (VF)
- VW == VL == VF



Vector Width Vector Length Vector Factor



• Superword Level Parallelism



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
for (i=0; i<N; i+=4)

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations



```
for (i=0; i<N; i+=4)

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations



```
for (i=0; i<N; i+=4)
A[i] = B[i]
A[i+1] = B[i+1]
A[i+2] = B[i+2]
A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4
for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4
for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4
for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i] = B[i]

A[i+1,i+5,i+9, i+13] = B[i+1,i+5,i+9, i+13]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4

for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]
for (i=0; i<N; i+=4)

A[i+1,i+5,i+9, i+13] = B[i+1,i+5,i+9, i+13]
A[i+1] = B[i+1]
A[i+1] = B[i+1]
A[i+2,i+6,i+10,i+14] = B[i+2,i+6,i+10,i+14]
A[i+3,i+7,i+11,i+15] = B[i+3,i+7,i+11,i+15]
A[i+3] = B[i+4]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations



```
for (i=0; i<N; i+=4)

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]
```

```
Loop Vectorization (LV) with VF = 4 for (i=0; i< N; i+=16)
```

```
A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i+1,i+5,i+9, i+13] = B[i+1,i+5,i+9, i+13]

A[i+2,i+6,i+10,i+14] = B[i+2,i+6,i+10,i+14]

A[i+3,i+7,i+11,i+15] = B[i+3,i+7,i+11,i+15]
```



```
SLP Vectorizer with VF = 4 for (i=0; i<N; i+=4)
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations



```
for (i=0; i<N; i+=4)
A[i] = B[i]
A[i+1] = B[i+1]
A[i+2] = B[i+2]
A[i+3] = B[i+4]
```

```
Loop Vectorization (LV) with VF = 4 for (i=0; i<N; i+=16)
```

```
A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i+1,i+5,i+9, i+13] = B[i+1,i+5,i+9, i+13]

A[i+2,i+6,i+10,i+14] = B[i+2,i+6,i+10,i+14]

A[i+3,i+7,i+11,i+15] = B[i+3,i+7,i+11,i+15]
```



SLP Vectorizer with VF = 4 for (i=0; i<N; i+=4)



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4

for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]

SLP Vectorizer with VF = 4

for (i=0; i<N; i+=4)

A[i:i+3] = B[i:i+3]
```



- Superword Level Parallelism
- Bottom-Up SLP implemented in GCC and LLVM
- Vectorizes across instructions, *NOT* iterations

```
Loop Vectorization (LV) with VF = 4

for (i=0; i<N; i+=16)

A[i, i+4,i+8, i+12] = B[i, i+4,i+8, i+12]

A[i] = B[i]

A[i+1] = B[i+1]

A[i+2] = B[i+2]

A[i+3] = B[i+4]

SLP Vectorizer with VF = 4

for (i=0; i<N; i+=4)

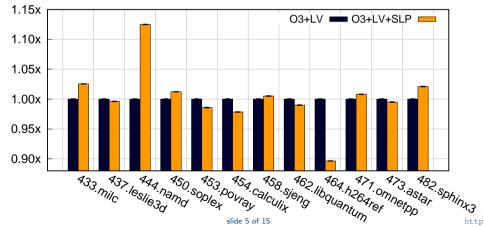
A[i:i+3] = B[i:i+3]
```

• Note: LV may be able to optimize the interleaved loads/stores



Isn't the Loop Vectorizer (LV) good enough?

- SPEC 2006, 10 runs, Intel[®] Core[™] i7-4790
- LLVM runs both LV and SLP



http://vporpo.me



```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



How SLP Works uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0]) A[i+1]=B[i+1]-(C[i+1]&D[i+1]) A[i+2]=E[i+0]-F[i+0] A[i+3]=E[i+1]-F[i+1] B[i+0] & L B[i+0] & L SA[i+0] & SA[i+1] & SA[i+2] & SA[i+3]



How SLP Works | uint64_t A[],B[],C[],D[],E[],F[] A[i+0] = B[i+0]-(C[i+0]&D[i+0]) A[i+1] = B[i+1]-(C[i+1]&D[i+1]) A[i+2] = E[i+0]-F[i+0] A[i+2] = E[i+0]-F[i+0] B[i+1]-F[i+1] B[i+1]-F[i+1] B[i+1]-F[i+1] B[i+1]-F[i+1] B[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+0]-F[i+1]-F[i

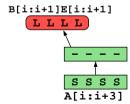
SSSS A[i:i+3]





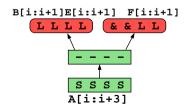


How SLP Works | uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0]) A[i+1]=B[i+1]-(C[i+1]&D[i+1]) A[i+2]=E[i+0]-F[i+0] A[i+2]=E[i+0]-F[i+0] A[i+3]=E[i+1]-F[i+1] A[i+3]=E[i+1]-F[i+1] A[i+3]=E[i+1]-F[i+1] A[i+3]=E[i+1]-F[i+1] A[i+3]=E[i+0] A[i+3]=E[i+1]-F[i+1] A[i+3]=E[i+1]-F[i+



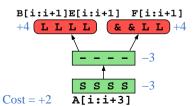


uint64_t A[],B[],C[],D[],E[],F[] How SLP Works A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]C[i+0]D[i+0]C[i+1]D[i+1]A[i+3]=E[i+1]-F[i+1]E[i+0]F[i+0] E[i+1]F[i+1]B[i+0] B[i+1] Œ **&** SA[i+0] S A[i+1] S A[i+2] \mathbf{S} $\mathbf{A}[\mathbf{i}+3]$





How SLP Works uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0]) A[i+1]=B[i+1]-(C[i+1]&D[i+1]) A[i+2]=E[i+0]- F[i+0] A[i+3]=E[i+1]- F[i+1] B[i+0] & L L L B[i+0] & L L SA[i+0] & SA[i+1] & SA[i+2] & SA[i+3]





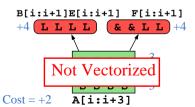
Cost = +2

A[i:i+3]

uint64_t A[],B[],C[],D[],E[],F[] How SIP Works A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]C[i+0]D[i+0]C[i+1]D[i+1]A[i+3]=E[i+1]-F[i+1]E[i+0]F[i+0] E[i+1]F[i+1] B[i+0] B[i+1] SA[i+0] S A[i+1] (S) A[i+2] \mathbf{S} $\mathbf{A}[\mathbf{i}+3]$ B[i:i+1]E[i:i+1] F[i:i+1] +4 (LLLL) **& & L L**)+4 Not Vectorized



```
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
B[i+0] B[i+1] E[i+0]F[i+0] E[i+1]F[i+1]
L L L L
SA[i+0] SA[i+1] SA[i+2] SA[i+3]
```



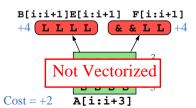
S S A[i:i+1]

uint64_t A[],B[],C[],D[],E[],F[]

A[i+0]=B[i+0]-(C[i+0]&D[i+0])



```
A[i+1]=B[i+1]=(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
B[i+0] B[i+1] E[i+0]F[i+0] E[i+1]F[i+1]
L L L L
SA[i+0] SA[i+1] SA[i+2] SA[i+3]
```





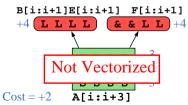
uint64_t A[],B[],C[],D[],E[],F[]

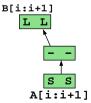
A[i+0]=B[i+0]=(C[i+0]&D[i+0])

slide 6 of 15



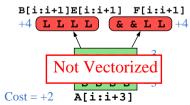
```
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
B[i+0] B[i+1] E[i+0]F[i+0] E[i+1]F[i+1]
L L L L
SA[i+0] SA[i+1] SA[i+2] SA[i+3]
```

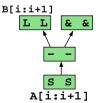




uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0])



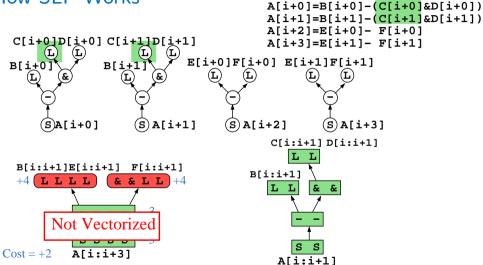




uint64_t A[],B[],C[],D[],E[],F[]

A[i+0]=B[i+0]-(C[i+0]&D[i+0])





slide 6 of 15

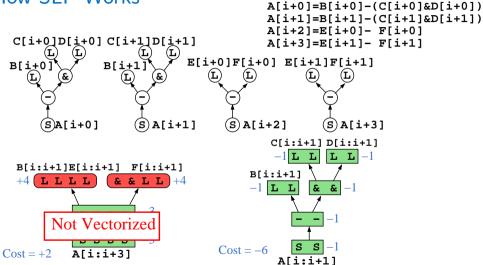
uint64_t A[],B[],C[],D[],E[],F[]



```
C[i+0]D[i+0]C[i+1]D[i+1]
                            E[i+0]F[i+0] E[i+1]F[i+1]
 B[i+0]
               B[i+1]
   (L)
      SA[i+0]
                    S A[i+1]
                                  (S) A[i+2]
  B[i:i+1]E[i:i+1] F[i:i+1]
                                      B[i:i+1]
  +4 (LLLL)
                & & L L +4
                                               & &
       Not Vectorized
                                            S
Cost = +2
          A[i:i+3]
                                          A[\overline{i:i+1}]
```

uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]A[i+3]=E[i+1]-F[i+1](S)A[i+3]C[i:i+1] D[i:i+1] L L

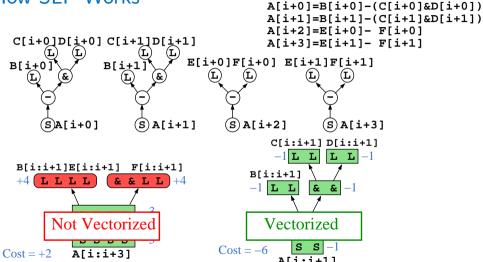




slide 6 of 15

uint64_t A[],B[],C[],D[],E[],F[]



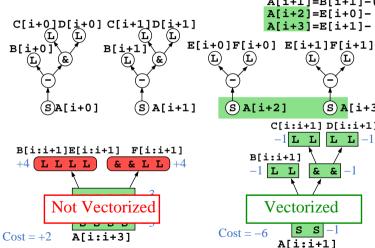


A[i:i+1] slide 6 of 15

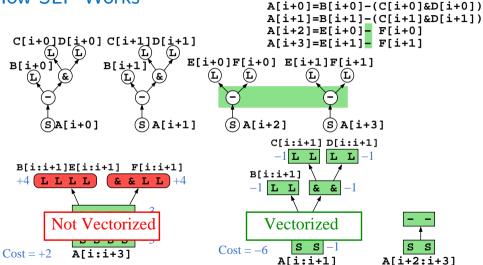
uint64_t A[],B[],C[],D[],E[],F[]

http://vporpo.me



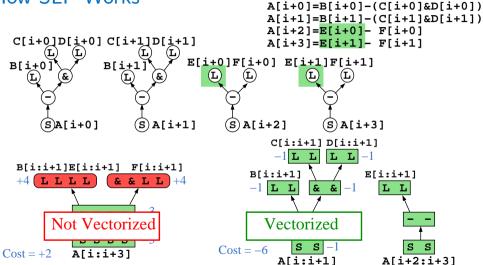






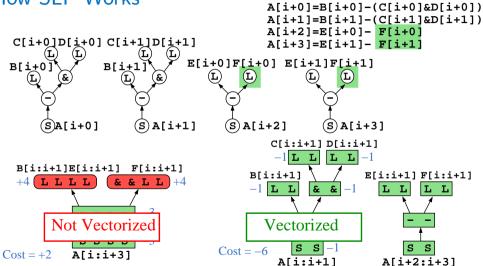
uint64_t A[],B[],C[],D[],E[],F[]





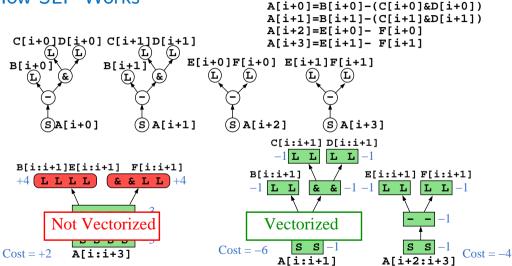
uint64_t A[],B[],C[],D[],E[],F[]





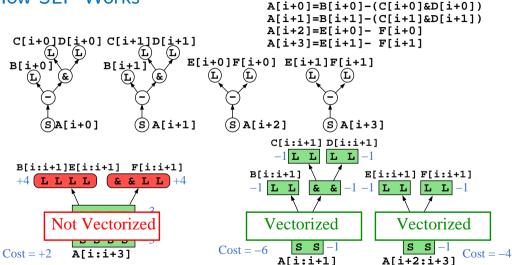
uint64_t A[],B[],C[],D[],E[],F[]





uint64_t A[],B[],C[],D[],E[],F[]





uint64_t A[],B[],C[],D[],E[],F[]



```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Find seed instructions

```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Find seed instructions

Get next seed chain

```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Find seed instructions

Get next seed chain

Success?

```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Find seed instructions Get next seed chain



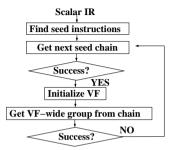
```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Scalar IR Find seed instructions Get next seed chain Success? YES Initialize VF Get VF-wide group from chain

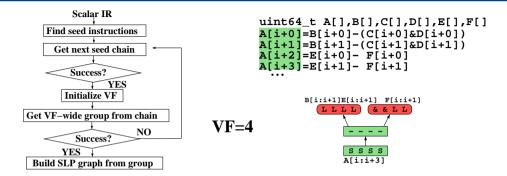
```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



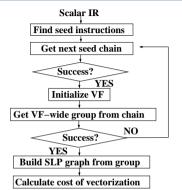


```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```







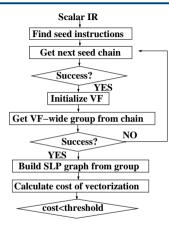


```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]-F[i+0]
A[i+3]=E[i+1]-F[i+1]
...

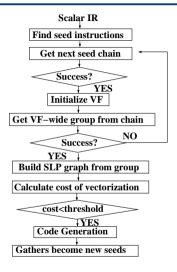
VF=4

VF=4
```



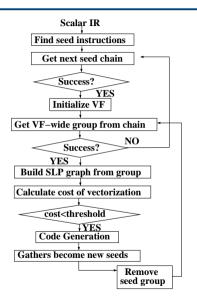




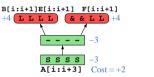


```
uint64 t A[],B[],C[],D[],E[],F[]
         A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
          A[i+3]=E[i+1]-F[i+1]
                   B[i:i+1]E[i:i+1] F[i:i+1]
                   +4 LLL & & LL +4
VF=4
                            A[i:i+3]
                                      Cost = \pm 2
```

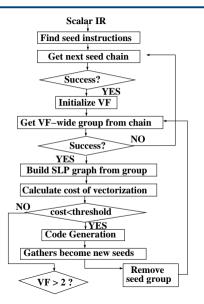




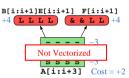
```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



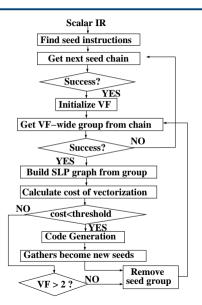




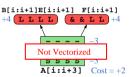
```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



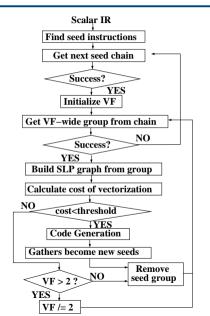




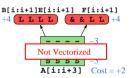
```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



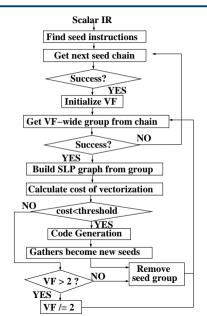




```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



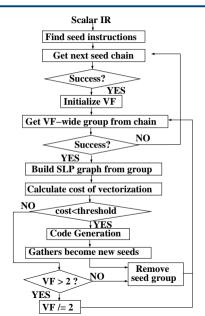




```
uint64 t A[],B[],C[],D[],E[],F[]
         A[i+0]=B[i+0]-(C[i+0]&D[i+0])

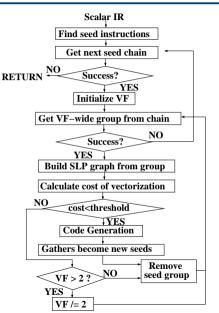
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
         A[i+2]=E[i+0]-F[i+0]
         A[i+3]=E[i+1]-F[i+1]
                  B[i:i+1]E[i:i+1] F[i:i+1]
                  +4 LLL & & LL +4
VF=4
                          Not Vectorized
                          A[i:i+3]
                                    Cost = \pm 2
               C[i:i+1] D[i:i+1]
               -1 L L L L -1
VF=2
            B[i:i+1]
                Vectorized
         Cost = -6 | 5 5 | -1
                A[\overline{i:i+1}]
```





```
uint64 t A[],B[],C[],D[],E[],F[]
        A[i+0]=B[i+0]-(C[i+0]&D[i+0])
        A[i+1]=B[i+1]-(C[i+1]&D[i+1])
         A[i+2]=E[i+0]-F[i+0]
        A[i+3]=E[i+1]-F[i+1]
                 B[i:i+1]E[i:i+1] F[i:i+1]
                 +4 LLL & & LL +4
VF=4
                        Not Vectorized
                         A[i:i+31] Cost = +2
              C[i:i+1] D[i:i+1]
              -1 L L L L -1
VF=2
           B[i:i+1]
                             E[i:i+1] F[i:i+1]
                            -1 T. T.
              Vectorized
                                 Vectorized
        Cost = -6 | 5 5 | -1
                           Cost = -4 | s | s | -1
              A[\overline{i:i+1}]
                                A[i+2:i+3]
```





```
uint64 t A[],B[],C[],D[],E[],F[]
         A[i+0]=B[i+0]-(C[i+0]&D[i+0])
         A[i+1]=B[i+1]-(C[i+1]&D[i+1])
         A[i+2]=E[i+0]-F[i+0]
         A[i+3]=E[i+1]-F[i+1]
                  B[i:i+1]E[i:i+1] F[i:i+1]
                  +4 LLL & & LL +4
VF=4
                         Not Vectorized
                          A[i:i+31] Cost = +2
              C[i:i+1] D[i:i+1]
               -1 L L L L -1
VF=2
           B[i:i+1]
                               E[i:i+1] F[i:i+1]
               Vectorized
                                  Vectorized
         Cost = -6 | 5 5 | -1
                             Cost = -4 | \mathbf{S} \cdot \mathbf{S} | -1
               A[\overline{i:i+1}]
                                 A[i+2:i+3]
```



Variable Width SLP

```
uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+0]&D[i+0])
A[i+1]=B[i+1]-(C[i+1]&D[i+1])
A[i+2]=E[i+0]- F[i+0]
A[i+3]=E[i+1]- F[i+1]
```



Variable Width SLP

```
A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+1]-(C[i+1]&D[i+1])

A[i+2]=E[i+0]-F[i+0]

A[i+2]=E[i+0]-F[i+0]

A[i+3]=E[i+1]-F[i+1]

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+1]-(C[i+1]&D[i+1])

A[i+2]=E[i+0]-F[i+0]

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+2]=E[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+1]&D[i+1])

A[i+2]=E[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+1]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]&D[i+0])

A[i+0]=B[i+0]-(C[i+0]
```

uint64 t A[],B[],C[],D[],E[],F[]



S S S S A[i:i+3]















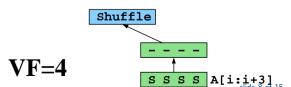
VF=2



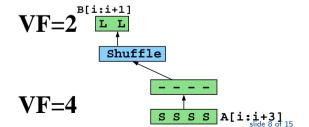




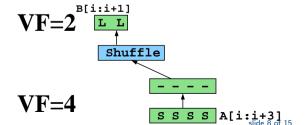
VF=2





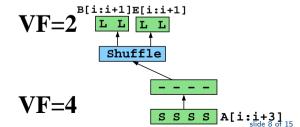






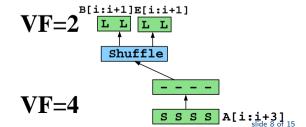


Variable Width SLP | Uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+0]&D[i+0]) A[i+1]=B[i+1]-(C[i+1]&D[i+1]) A[i+2]=E[i+0]-F[i+0] A[i+3]=E[i+1]-F[i+1] B[i+0] & D[i+1] B[i+1] B[i+1]-F[i+1] D[i+1] B[i+0] & D[i+1] B[i+1]-F[i+1] D[i+0] B[i+1] & D[i+1] B[i+1] D[i+1] D[i+1]



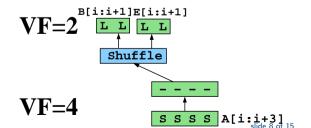


uint64 t A[],B[],C[],D[],E[],F[] Variable Width SI P A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]C[i+0]D[i+0]C[i+1]D[i+1]A[i+3]=E[i+1]-F[i+1]E[i+0]F[i+0] E[i+1]F[i+1]B[i+0] B[i+1] (&) & (L)S A[i+0] S A[i+1] (S)A[i+2] $(\mathbf{S})\mathbf{A}[\mathbf{i}+3]$



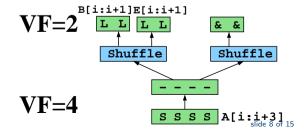


uint64 t A[],B[],C[],D[],E[],F[] Variable Width SI P A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]C[i+0]D[i+0]C[i+1]D[i+1]A[i+3]=E[i+1]-F[i+1]E[i+0]F[i+0] E[i+1]F[i+1]B[i+1] B[i+0] (3) (&) (L)S A[i+0] S A[i+1] (S) A[i+2] $(\mathbf{S})\mathbf{A}[\mathbf{i}+3]$

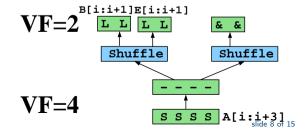




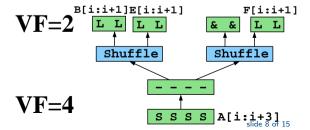
uint64 t A[],B[],C[],D[],E[],F[] Variable Width SI P A[i+0]=B[i+0]-(C[i+0]&D[i+0])A[i+1]=B[i+1]-(C[i+1]&D[i+1])A[i+2]=E[i+0]-F[i+0]C[i+0]D[i+0]C[i+1]D[i+1]A[i+3]=E[i+1]-F[i+1]E[i+0]F[i+0] E[i+1]F[i+1]B[i+1] B[i+0] (&) (æ) (L)S A[i+0] S A[i+1] (S)A[i+2] $(\mathbf{S})\mathbf{A}[\mathbf{i}+3]$



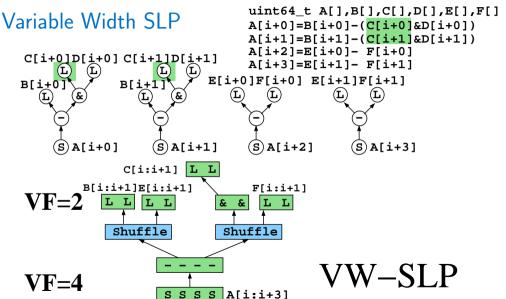




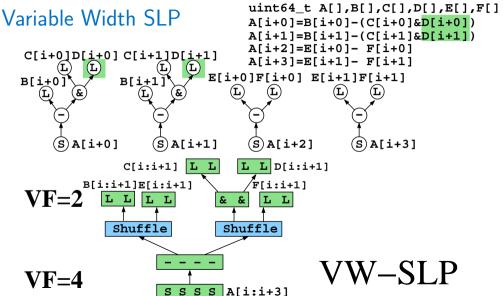




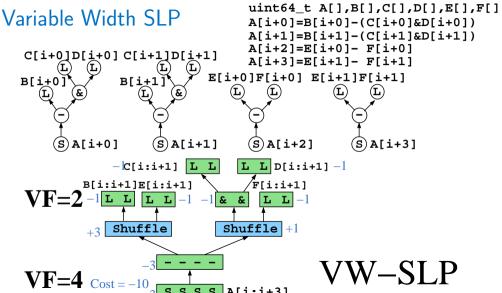




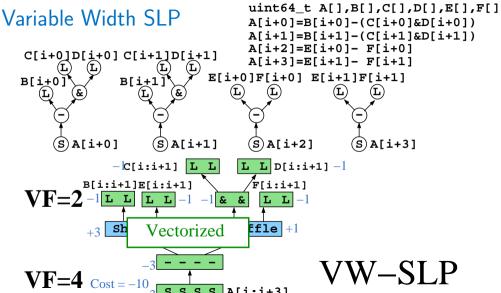














```
uint64_t A[],B[],C[],D[],E[],F[]
                                              A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                              A[i+1]=B[i+1]-F[i+1)
                                              A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                              A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
           B[i+1]F[i+1]E[i+0]
B[i+0]
                                      E[i+1]F[i+0]
    (s) A[i+0]
                  (s) A[i+1]
                              (s) A[i+2]
                                            (s) A[i+3]
```



A[i:i+3]

```
uint64_t A[],B[],C[],D[],E[],F[]
                                             A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                             A[i+1]=B[i+1]-F[i+1)
                                             A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                             A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
           B[i+1]F[i+1]E[i+0]
B[i+0]
                                      E[i+1]F[i+0]
    (s) A[i+0]
                 S A[i+1]
                              (s) A[i+2]
                                           (s) A[i+3]
     E[i:i+1]
 B[i:i+1]
   LLLL
        SSSS
```



A[i:i+3]

```
uint64_t A[],B[],C[],D[],E[],F[]
                                              A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                              A[i+1]=B[i+1]-F[i+1)
                                              A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                              A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
           B[i+1]F[i+1]E[i+0]
B[i+0]
                                      E[i+1]F[i+0]
                                 æ
    (s) A[i+0]
                 (s) A[i+1]
                              (s)A[i+2]
                                            (s) A[i+3]
     E[i:i+1]
                  F[i]
 B[i:i+1]
            F[i+1]
   LLLL
             & L & L
        SSSS
```



A[i:i+3]

Cost = +2

```
uint64_t A[],B[],C[],D[],E[],F[]
                                              A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                              A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
           B[i+1]F[i+1]E[i+0]
B[i+0]
                                      E[i+1]F[i+0]
    (s) A[i+0]
                  (s) A[i+1]
                              (s)A[i+2]
                                            (s) A[i+3]
      E[i:i+1]
                  F[i]
  B[i:i+1]
            F[i+1]
   LLLL
             & L & L
  \pm 4
        SSSS
```



A[i:i+3]

Cost = +2

```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                           C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
    (s) A[i+0]
                  (s) A[i+1]
                               (s)A[i+2]
                                             (s) A[i+3]
      E[i:i+1]
                  F[i]
  B[i:i+1]
            F[i+1]
   LLLL
             & L & L
  \pm 4
       Not Vectorized
```



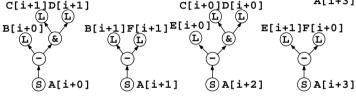
```
uint64_t A[],B[],C[],D[],E[],F[]
                                                A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                                A[i+1]=B[i+1]-F[i+1)
                                                A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                                A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                           C[i+0]D[i+0]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                        E[i+1]F[i+0]
  (L)
    (s) A[i+0]
                  (s) A[i+1]
                                S A[i+2]
                                              (s) A[i+3]
      E[i:i+1]
                  F[i]
                             F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                   E[i:i+1]
                                             F[i]
   LLLL
              & L & L
                        L L
                                     L L
                        -1
  \pm 4
       Not Vectorized
                                         s s -1
                                     A[i+2:i+3]
        A[i:i+3]
                         A[i:i+1]
    Cost = +2
                          Cost = -1
                                      Cost = -1
```

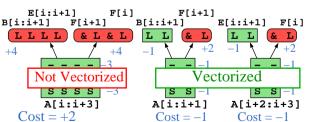


```
uint64_t A[],B[],C[],D[],E[],F[]
                                                A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                                A[i+1]=B[i+1]-F[i+1)
                                                A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                                A[i+3]=E[i+1]-F[i+0]
                           C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                        E[i+1]F[i+0]
  (L)
    (s) A[i+0]
                  (s) A[i+1]
                               S A[i+2]
                                              (s) A[i+3]
      E[i:i+1]
                  F[i]
                             F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                   E[i:i+1]
                                             F[i]
   LLLL
              & L & L
                       L L
                                     L L
  \pm 4
                               Vectorized
       Not Vectorized
                                     A[i+2:i+3]
        A[i:i+3]
                         A[i:i+1]
    Cost = +2
                          Cost = -1
                                      Cost = -1
```



uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+1]&D[i+1])
A[i+1]=B[i+1]- F[i+1)
A[i+2]=E[i+0]-(C[i+0]&D[i+0])
A[i+3]=E[i+1]- F[i+0]



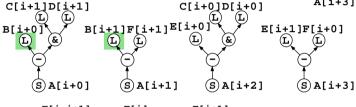


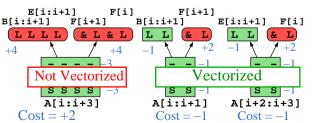




uint64_t A[],B[],C[],D[],E[],F[]
A[i+0]=B[i+0]-(C[i+1]&D[i+1])
A[i+1]=B[i+1]- F[i+1)
A[i+2]=E[i+0]-(C[i+0]&D[i+0])
A[i+3]=E[i+1]- F[i+0]

(L)









```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
                                                            VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                              (s)A[i+2]
                                            (s) A[i+3]
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                           F[i]
                      L
   LLLL
             & L & L
                                    L L
                                                        Shuffle
  \pm 4
                              Vectorized
       Not Vectorized
                                        S S -
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                             A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0](L)
B[i+0]
                                       E[i+1]F[i+0]
                                                            VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                               (s)A[i+2]
                                             (s) A[i+3]
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
                       L
   LLLL
             & L & L
                                    L L
                                                        Shuffle
  \pm 4
                               Vectorized
       Not Vectorized
                                        S S -
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                          C[i+0]D[i+0]
            B[i+1]F[i+1]E[i+0](L)
B[i+0]
                                       E[i+1]F[i+0]
                                                            VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                                             (s) A[i+3]
                               (S)A[i+2]
                                                         E[i:i+1]
                                                    B[i:i+1]
      E[i:i+1]
                  F[i]
                            F[i+1]
                                                           L L
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
   LLLL
             & L & L
                       L L
                                    L L
                                                        Shuffle
  \pm 4
                               Vectorized
       Not Vectorized
        SSSS
                                        S S -
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                           C[i+0]D[i+0]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
       (&)
                                                            VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                                             (s) A[i+3]
                               (S)A[i+2]
                                                          E[i:i+1]
                                                    B[i:i+1]
      E[i:i+1]
                  F[i]
                             F[i+1]
                                                            L L
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                   E[i:i+1]
                                            F[i]
   LLLL
              & L & L
                       L L
                                    L L
                                                        Shuffle
  \pm 4
                               Vectorized
       Not Vectorized
                                        S S -
                                                               SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                         A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                          C[i+0]D[i+0]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
       (&)
                                                           VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                               (s)A[i+2]
                                            (s) A[i+3]
                                                         E[i:i+1]
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
                                                                    & &
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
   LLLL
             & L & L
                       L L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
                               Vectorized
       Not Vectorized
        SSSS
                                        S S -
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



uint64_t A[],B[],C[],D[],E[],F[] A[i+0]=B[i+0]-(C[i+1]&D[i+1])A[i+1]=B[i+1]-F[i+1)A[i+2]=E[i+0]-(C[i+0]&D[i+0])A[i+3]=E[i+1]-F[i+0]C[i+1]D[i+1]C[i+0]D[i+0]B[i+1]F[i+1]E[i+v] E[i+1]F[i+0] B[i+0]VW–SLP (s) A[i+0] (s) A[i+1] (s) A[i+3] (S)A[i+2] E[i:i+1] B[i:i+1] E[i:i+1] F[i] F[i+1] & & B[i:i+1] F[i+1] B[i:i+1] E[i:i+1] F[i] L LLLL & L & L L L Shuffle Shuffle ± 4 Vectorized Not Vectorized SSSS S S -SSSS A[i+2:i+3]A[i:i+3] A[i:i+1] A[i:i+1] Cost = +2Cost = -1Cost = -1



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                          C[i+0]D[i+0]
           B[i+1]F[i+1]E[i+0]
B[i+0]
                                      E[i+1]F[i+0]
  (L)
                                                           VW–SLP
    (s) A[i+0]
                  (s) A[i+1]
                                            (s) A[i+3]
                              (S)A[i+2]
                                                         E[i:i+1]
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
                                                                   & &
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
                      L
   LLLL
             & L & L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
                              Vectorized
       Not Vectorized
        SSSS
                                        S S -
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                             A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
C[i+1]D[i+1]
                          C[i+0]D[i+0]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
                                                            VW–SLP
                                                             C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  (s) A[i+1]
                                            (s) A[i+3]
                               (S)A[i+2]
                                                         E[i:i+1] L L
                                                                      L L
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
   LLLL
             & L & L
                       L L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
       Not Vectorized
                               Vectorized
        SSSS
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
                                                            VW–SLP
                                                             C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  (s) A[i+1]
                                            (s) A[i+3]
                               (S)A[i+2]
                                                         E[i:i+1] L L
                                                                      L L
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
   LLLL
             & L & L
                       L L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
       Not Vectorized
                               Vectorized
        SSSS
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
                                                            VW–SLP
                                                             C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  (s) A[i+1]
                               S A[i+2]
                                            (s) A[i+3]
                                                         E[i:i+1] L L
                                                                      L L
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
   LLLL
             & L & L
                       L L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
       Not Vectorized
                               Vectorized
        SSSS
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
uint64_t A[],B[],C[],D[],E[],F[]
                                               A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                               A[i+1]=B[i+1]-F[i+1)
                                               A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                               A[i+3]=E[i+1]-F[i+0]
                          C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
                                                            VW–SLP
                                                             C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  (s) A[i+1]
                               S A[i+2]
                                             (s) A[i+3]
                                                                        F[i:i+1]
                                                    B[i:i+1]
     E[i:i+1]
                  F[i]
                            F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                  E[i:i+1]
                                            F[i]
                       L
   LLLL
             & L & L
                                    L L
                                                        Shuffle
                                                                     Shuffle
  \pm 4
       Not Vectorized
                               Vectorized
        SSSS
                                                              SSSS
                                    A[i+2:i+3]
        A[i:i+3]
                        A[i:i+1]
                                                              A[i:i+1]
    Cost = +2
                         Cost = -1
                                     Cost = -1
```



```
A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                                A[i+1]=B[i+1]-F[i+1)
                                                A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                                A[i+3]=E[i+1]-F[i+0]
                           C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                        E[i+1]F[i+0]
  (L)
                                                              VW–SLP
                                                               C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  (s) A[i+1]
                                              (s) A[i+3]
                               (S)A[i+2]
                                                                           F[i:i+1]
                                                     B[i:i+1]
      E[i:i+1]
                   F[i]
                             F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                   E[i:i+1]
                                             F[i]
                       L
   LLLL
              & L & L
                                     L L
                                                         Shuffle
                                                                       Shuffle +3
  \pm 4
                                                         \pm 1
       Not Vectorized
                               Vectorized
                                         S S -
                                     A[i+2:i+3]
        A[i:i+3]
                         A[i:i+1]
                                                     Cost = -8
    Cost = +2
                          Cost = -1
                                      Cost = -1
```

uint64_t A[],B[],C[],D[],E[],F[]

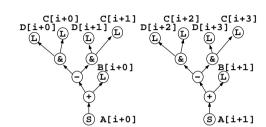


```
A[i+0]=B[i+0]-(C[i+1]&D[i+1])
                                                A[i+1]=B[i+1]-F[i+1)
                                                A[i+2]=E[i+0]-(C[i+0]&D[i+0])
                                                A[i+3]=E[i+1]-F[i+0]
                           C[i+0]D[i+0]
C[i+1]D[i+1]
            B[i+1]F[i+1]E[i+0]
B[i+0]
                                       E[i+1]F[i+0]
  (L)
                                                             VW–SLP
                                                              C[i:i+1] D[i:i+1]
    (s) A[i+0]
                  S A[i+1]
                                             (s) A[i+3]
                               (S)A[i+2]
                                                                          F[i:i+1]
                                                     B[i:i+1]
      E[i:i+1]
                  F[i]
                             F[i+1]
  B[i:i+1]
            F[i+1]
                      B[i:i+1]
                                   E[i:i+1]
                                             F[i]
                       L
   LLLL
              & L & L
                                     L L
                                                         Shuffle
                                                                      Shuffle +3
  \pm 4
                                                             Vectorized
       Not Vectorized
                               Vectorized
                                         S S -
                                     A[i+2:i+3]
        A[i:i+3]
                         A[i:i+1]
                                                    Cost = -8
    Cost = +2
                         Cost = -1
                                      Cost = -1
```

uint64_t A[],B[],C[],D[],E[],F[]

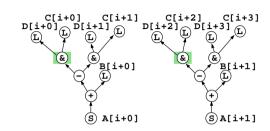


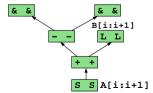
```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```





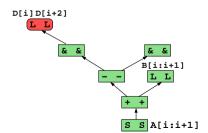
```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

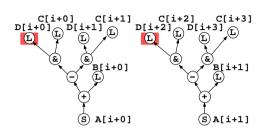






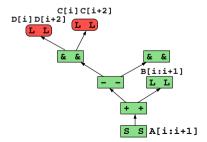
```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

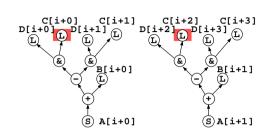






```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

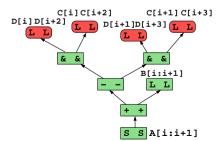


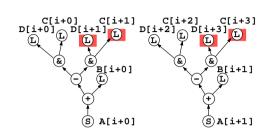


slide 10 of 15 http://vporpo.me



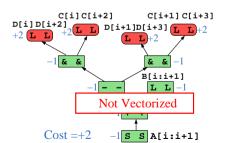
```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

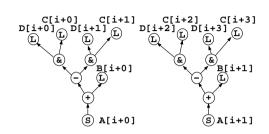






```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

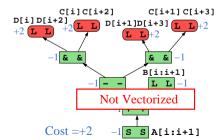


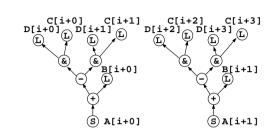


slide 10 of 15 http://vporpo.me

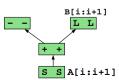


```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```



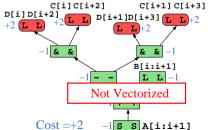


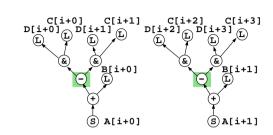




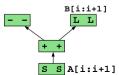


```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```





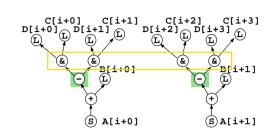




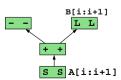
VF=2



```
uint64 t A[],B[],C[],D[]
uint64 t tmp0=C[i+0]&D[i+0]
uint64 t tmp1=C[i+1]&D[i+1]
uint64 t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
        C[i]C[i+2]
                       C[i+1]C[i+3]
D[i]D[i+2]
               D[i+1]D[i+3] T. 1.
+2 L L
      −1 & &
                      B[i:i+1]
               Not Vectorized
     Cost = +2
                 -1 S S A[i:i+1]
```







VF=2

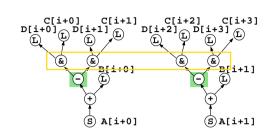


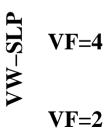
Not Vectorized

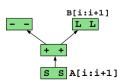
-1 S S A[i:i+1]

```
uint64 t A[],B[],C[],D[]
uint64 t tmp0=C[i+0]&D[i+0]
uint64 t tmp1=C[i+1]&D[i+1]
uint64 t tmp2=C[i+2]&D[i+2]
uint64 t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
       C[i]C[i+2]
                       C[i+1]C[i+3]
D[i]D[i+2]
               D[i+1]D[i+3] T. +2
+2 L L
      −1 & &
                      B[i:i+1]
                       L L -
```

Cost = +2

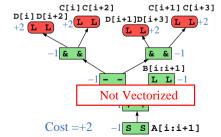


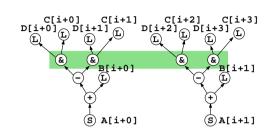


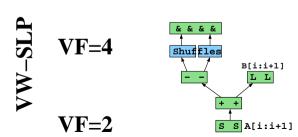




```
uint64_t A[],B[],C[],D[]
uint64_t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64_t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
```

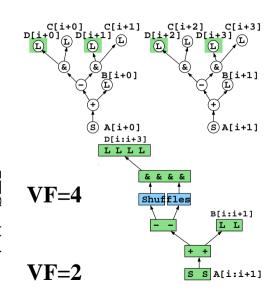






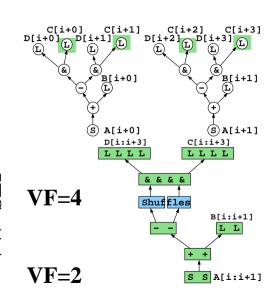


```
uint64 t A[],B[],C[],D[]
uint64 t tmp0=C[i+0]&D[i+0]
uint64 t tmp1=C[i+1]&D[i+1]
uint64 t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
        C[i]C[i+2]
                       C[i+1]C[i+3]
D[i]D[i+2]
               D[i+1]D[i+3] T. +2
+2 L L
      −1 & &
                      B[i:i+1]
                        L L -
               Not Vectorized
     Cost = +2
                 -1 S S A[i:i+1]
```



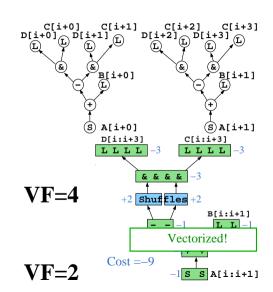


```
uint64 t A[],B[],C[],D[]
uint64 t tmp0=C[i+0]&D[i+0]
uint64 t tmp1=C[i+1]&D[i+1]
uint64 t tmp2=C[i+2]&D[i+2]
uint64_t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
        C[i]C[i+2]
                       C[i+1]C[i+3]
D[i]D[i+2]
               D[i+1]D[i+3] T. +2
+2 L L
      −1 & &
                      B[i:i+1]
                        L L -
               Not Vectorized
     Cost = +2
                 -1 S S A[i:i+1]
```





```
uint64 t A[],B[],C[],D[]
uint64 t tmp0=C[i+0]&D[i+0]
uint64_t tmp1=C[i+1]&D[i+1]
uint64 t tmp2=C[i+2]&D[i+2]
uint64 t tmp3=C[i+3]&D[i+3]
A[i+0]=B[i+0]+(tmp0-tmp1)
A[i+1]=B[i+1]+(tmp2-tmp3)
       C[i]C[i+2]
                       C[i+1]C[i+3]
D[i]D[i+2]
               D[i+1]D[i+3] T. +2
+2 L L
      −1 & &
                      B[i:i+1]
                       L L -
               Not Vectorized
     Cost = +2
                 -1SSA[i:i+1]
```



slide 10 of 15

http://vporpo.me



• Modified buildTree_rec() to handle variable-width



- Modified buildTree_rec() to handle variable-width
- Spawn localized buildTree_rec(NewOp) searches with shorter/wider/permuted NewOp



- Modified buildTree_rec() to handle variable-width
- Spawn localized buildTree_rec(NewOp) searches with shorter/wider/permuted NewOp
- Cap the buildTree_rec() to a maximum exploration depth.



- Modified buildTree_rec() to handle variable-width
- Spawn localized buildTree_rec(NewOp) searches with shorter/wider/permuted NewOp
- Cap the buildTree_rec() to a maximum exploration depth.
- Evaluate the cost of each new sub-tree using getEntryCost()
 and pick the NewOp with the best cost



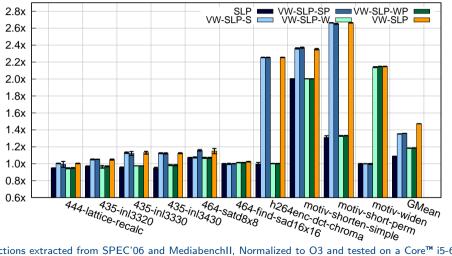
- Modified buildTree_rec() to handle variable-width
- Spawn localized buildTree_rec(NewOp) searches with shorter/wider/permuted NewOp
- Cap the buildTree_rec() to a maximum exploration depth.
- Evaluate the cost of each new sub-tree using getEntryCost() and pick the NewOp with the best cost
- Added scheduler support for roll-back and replay



- Modified buildTree_rec() to handle variable-width
- Spawn localized buildTree_rec(NewOp) searches with shorter/wider/permuted NewOp
- Cap the buildTree_rec() to a maximum exploration depth.
- Evaluate the cost of each new sub-tree using getEntryCost() and pick the NewOp with the best cost
- Added scheduler support for roll-back and replay
- Code generation changes in vectorizeTree() for variable-width support and the shufflevector instructions



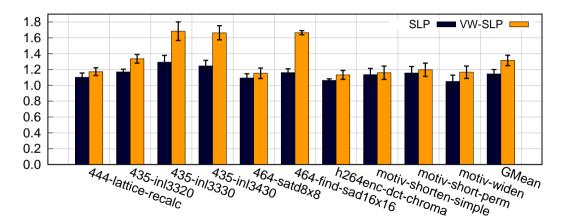
Performance



Functions extracted from SPEC'06 and MediabenchII, Normalized to O3 and tested on a Core™ i5-6440HQ slide 12 of 15 http://vporpo.me



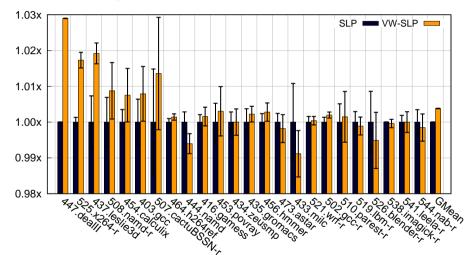
Compilation Time



Core[™] i5-6440HQ



Performance on SPEC



PSPEC 2006/2017 SLP(No LV), VW-SLP(No LV), on Core™ i5-6440HQ slide 14 of 15



• Presented SLP with variable vector length



- Presented SLP with variable vector length
- More effective SLP graph



- Presented SLP with variable vector length
- More effective SLP graph
- Better performance and coverage



- Presented SLP with variable vector length
- More effective SLP graph
- Better performance and coverage
- Very small increase in compilation time