Intel® Composer XE

Vectorization:
Pragma/Directive SIMD
User-Mandated Vectorization

User-mandated vectorization is based on a new **SIMD Directive** (or “pragma”)

- The SIMD directive provides additional information to compiler to enable vectorization of loops (at this time only inner loop)
- Supplements automatic vectorization but differently to what traditional directives like IVDEP, VECTOR ALWAYS do, the SIMD directive is more a command than a hint or an assertion: The compiler heuristics are completely overwritten as long as a clear logical fault is not being introduced

Relationship similar to OpenMP versus automatic parallelization:

- User Mandated Vectorization
- Pure Automatic Vectorization
- OpenMP
- Automatic Parallelization
Positioning of SIMD Vectorization

- Fully automatic vectorization
- Auto vectorization hints (#pragma ivdep)
- User Mandated Vectorization (SIMD Directive)
- SIMD intrinsic class (F32vec4 add)
- Vector intrinsic (mm_add_ps())
- ASM code (addps)

Ease of use

Programmer control
SIMD Directive Notation

C/C++:  #pragma simd [clause [,clause] ...]

Fortran:  !DIR$ SIMD [clause [,clause] ...]

Without any clause, the directive enforces vectorization of the 
(innermost) loop

Sample:

```c
void add_fl(float *a, float *b, float *c, float *d, float *e, int n)
{
    #pragma simd
    for (int i=0; i<n; i++)
        a[i] = a[i] + b[i] + c[i] + d[i] + e[i];
}
```

Without SIMD directive, vectorization will fail since there are too many 
pointer references to do a run-time check for overlapping (compiler 
heuristic)
Clauses of SIMD Directive

vectorlength(n1 [,n2] ...)  
- n1, n2, ... must be 2,4,8 or 16: The compiler can assume a  
vectorization for a vector length of n1, n2, ... to be save

private(v1, v2, ...)  
- variables private to each iteration; initial value is broadcast to all  
private instances, and the last value is copied out from the last  
iteration instance.

linear(v1:step1, v2:step2, ...)  
- for every iteration of original scalar loop, v1 is incremented by  
step1, ... etc. Therefore it is incremented by step1 *(vector length)  
for the vectorized loop.

reduction(operator:v1, v2, ...)  
- v1 etc are reduction variables for operation “operator”

[no]assert  
- reaction in case vectorization fails: Print a warning only ( noassert,  
the default) or treat failure as error and stop compilation
Sample: SIMD Directive Vectorlength Clause

Void foo(float *a, float *b, float *c, int n)
{
    for (int k=0; k<n; k++)
        c[k] = a[k] + b[k];
}

Due to the overlapping nature of array accesses from the different call sites, it might not be semantically correct to use restrict keyword or IVDEP directive (there are dependencies between iterations for one call)

But it might be true for all calls, that e.g. 4 consecutive iterations can be executed in parallel without violating any dependencies

void foo(float *a, float *b, float *c, int n)
{
    #pragma simd vectorlength(4)
    for (int k=0; k<n; k++)
        c[k] = a[k] + b[k];
}
Sample: SIMD Directive Linear Clause

do 10 i=n1,n,n3
   k = k + j
   a(i) = a(i) + b(n-k+1)
 10 continue

Vectorization fails: “Existence of vector dependence”

!DIR$ SIMD linear(k:j)
do 10 i=n1,n,n3
   k = k + j
   a(i) = a(i) + b(n-k+1)
 10 continue

Vectorization succeeds now: The compiler receives the additional information, that k is an induction variable being incremented by j in each iteration. This is sufficient to enable vectorization
Optimization Notice

Intel’s 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 microarchitecture are reserved for 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
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.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference [www.intel.com/software/products](http://www.intel.com/software/products).


*Other names and brands may be claimed as the property of others.

Copyright © 2011. Intel Corporation.

http://intel.com/software/products