

performance/compiler\_optionen/llvm/example\_loop\_vectorizer\_dia

## Example: LLVM compiler loop vectorizer diagnostics for benchmark stream

- LLVM: Loop Vectorizer: Diagnostics
- `stream` source code snippet

```
/* ---
Tuned vector scale:  b[] = scalar * c[]

In:  STREAM_ARRAY_SIZE_thread, scalar, c[]
Out: b[]
--- */
void static inline tuned_STREAM_Scale(const STREAM_TYPE scalar) {
    #pragma omp parallel default(none) shared(scalar, STREAM_ARRAY_SIZE_thread)
    {
        #ifdef __INTEL_COMPILER
            // Instructs the compiler to use non-temporal (that is, streaming) stores
            #pragma vector nontemporal
        #endif
        #pragma omp simd aligned(b, c : alignment_bytes)
        for (long int j = 0; j < STREAM_ARRAY_SIZE_thread; j++) {
            b[j] = scalar * c[j]; // Line: 349
        }
    }
}

/* ---
Tuned vector add:  c[] = a[] + b[]

In:  STREAM_ARRAY_SIZE_thread, a[], b[]
Out: c[]
--- */
void static inline tuned_STREAM_Add() {
```

```

#pragma omp parallel default(none) shared(STREAM_ARRAY_SIZE_thread)
{
    #ifdef __INTEL_COMPILER
        // Instructs the compiler to use non-temporal (that is, streaming) stores
        #pragma vector nontemporal
    #endif
    #pragma omp simd aligned(a, b, c : alignment_bytes)
    for (long int j = 0; j < STREAM_ARRAY_SIZE_thread; j++) {
        c[j] = a[j] + b[j]; // Line: 369
    }
}
}

```

- Prepare environment

```

module purge
module add compiler/llvm

```

- Compile benchmark with optimization report enabled

```

# -Rpass=loop-vectorize
# -> identifies loops that were successfully vectorized
# -Rpass-missed=loop-vectorize
# -> identifies loops that failed vectorization and indicates if vectorization was spee
# -Rpass-analysis=loop-vectorize
# -> identifies the statements that caused vectorization to fail.
# If in addition -fsave-optimization-record is provided, multiple causes of vectoriz
clang -std=c11 -Ofast -march=native -flto -fopenmp \
    -Rpass=loop-vectorize \
    -Rpass-missed=loop-vectorize \
    -Rpass-analysis=loop-vectorize \
    stream.OpenMP.c -o stream

```

- Output

```

stream.OpenMP.c:347:9: remark: vectorized loop (vectorization width: 4, interleaved cou
    #pragma omp simd aligned(b, c : alignment_bytes)
    ^
stream.OpenMP.c:367:9: remark: vectorized loop (vectorization width: 4, interleaved cou
    #pragma omp simd aligned(a, b, c : alignment_bytes)
    ^
...
LLVM gold plugin: stream.OpenMP.c:347:9: loop not vectorized: vectorization and interle
LLVM gold plugin: stream.OpenMP.c:347:9: loop not vectorized: vectorization and interle
LLVM gold plugin: stream.OpenMP.c:367:9: loop not vectorized: vectorization and interle
LLVM gold plugin: stream.OpenMP.c:367:9: loop not vectorized: vectorization and interle
...

```

- Report on successful vectorization

- Report on vector length