

performance/compiler_optionen/gcc/example_vec_report_stream

Example: GCC compiler optimization report for benchmark stream

- 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/gnu/12

```

- Compile benchmark with vectorization report enabled

```

gcc -std=c11 -Ofast -march=native -flto -fopenmp \
    -fopt-info-vec \
    stream.OpenMP.c -o stream

```

- Output

```

...
stream.OpenMP.c:349:30: optimized: loop vectorized using 32 byte vectors
stream.OpenMP.c:349:30: optimized: loop vectorized using 16 byte vectors
stream.OpenMP.c:369:21: optimized: loop vectorized using 32 byte vectors
stream.OpenMP.c:369:21: optimized: loop vectorized using 16 byte vectors
...

```

- Report on successful vectorization
- Report on vector length (loop + remainder loop)