SIMD demo

1) **Set number of threads:**
```
export OMP_NUM_THREADS=4
```

2) **Set CPU affinity:**
```
export GOMP_CPU_AFFINITY=1,2,3,4
```

3) **Check:**
```
cpuinfo
```

here is the output:
```
simd> cpuinfo
Intel(R) processor family information utility, Version 2021.1-
beta10 Build 20201007 (id: 81f9c8f47)
Copyright (C) 2005-2020 Intel Corporation.  All rights reserved.

=====  Processor composition  =====
Processor name    : Intel(R) Core(TM) i7-7700HQ
Packages(sockets) : 1
Cores             : 4
Processors(CPUs)  : 8
Cores per package : 4
Threads per core  : 2

=====  Processor identification  =====
Processor Thread Id.     Core Id.  Package Id.
0         0         0         0
1         0         1         0
2         0         2         0
3         0         3         0
4         1         0         0
5         1         1         0
6         1         2         0
7         1         3         0
=====  Placement on packages  =====
Package Id.    Core Id.  Processors
0         0,1,2,3        (0,4)(1,5)(2,6)(3,7)

=====  Cache sharing  =====
Cache     Size      Processors
L1   32  KB         (0,4)(1,5)(2,6)(3,7)
L2   256 KB    (0,4)(1,5)(2,6)(3,7)
L3   6   MB         (0,1,2,3,4,5,6,7)
simd>
```

4) **Prepare the compilation environment**

```
simd> source ~/science/opt/intel/oneapi/setvars.sh
:: initializing oneAPI environment ...
   BASH version = 5.0.17(1)-release
:: vpl -- latest
:: mkl -- latest
:: inspector -- latest
:: dpcpp-ct -- latest
:: vtune -- latest
:: debugger -- latest
:: mpi -- latest
:: clck -- latest
:: dpl -- latest
:: intelpython -- latest
:: itac -- latest
:: ipp -- latest
:: ippcp -- latest
:: tbb -- latest
:: compiler -- latest
:: dev-utilities -- latest
:: ccl -- latest
:: dal -- latest
:: dnnl -- latest
:: advisor -- latest
:: oneAPI environment initialized ::
```

5) **Compile:**

```
simd> icc -O3 -qopt-report -qopenmp -o ./simd_example
./simd_example.c
icc: remark #10397: optimization reports are generated in *.optrpt
files in the output location
```

6) **Execute:**

./simd_example

simd> ./simd_example
serial computation
compute time=0.007758

Parallel loop
compute time=0.009401

Parallel loop+simd
compute time=0.008552

7) **Check the optimization report**

```
simd> code ./simd_example.optrpt &
```

```
Intel(R) Advisor can now assist with vectorization and show optimization
  report messages with your source code.
See "https://software.intel.com/en-us/intel-advisor-xe" for details.


    Report from: Interprocedural optimizations [ipo]

INLINING OPTION VALUES:
  -inline-factor: 100
  -inline-min-size: 30
  -inline-max-size: 230
  -inline-max-total-size: 2000
  -inline-max-per-routine: 10000
  -inline-max-per-compile: 500000


Begin optimization report for: main()

    Report from: Interprocedural optimizations [ipo]

INLINE REPORT: (main()) [1] ./simd_example.c(19,12)


    Report from: OpenMP optimizations [openmp]

OpenMP Construct at ./simd_example.c(50,5)
remark #16204: OpenMP multithreaded code generation for SINGLE was successful
OpenMP Construct at ./simd_example.c(49,5)
remark #16201: OpenMP DEFINED REGION WAS PARALLELIZED
OpenMP Construct at ./simd_example.c(69,5)
remark #16204: OpenMP multithreaded code generation for SINGLE was successful
OpenMP Construct at ./simd_example.c(68,5)
remark #16201: OpenMP DEFINED REGION WAS PARALLELIZED

    Report from: Loop nest, Vector & Auto-parallelization optimizations [loop,
vec, par]


LOOP BEGIN at ./simd_example.c(32,5)
<Peeled loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(32,5)
   remark #25084: Preprocess Loopnests: Moving Out Store    [
./simd_example.c(32,18) ]
   remark #15300: LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(32,5)
<Alternate Alignment Vectorized Loop>
LOOP END

LOOP BEGIN at ./simd_example.c(32,5)
<Remainder loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(37,5)
<Peeled loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(37,5)
```

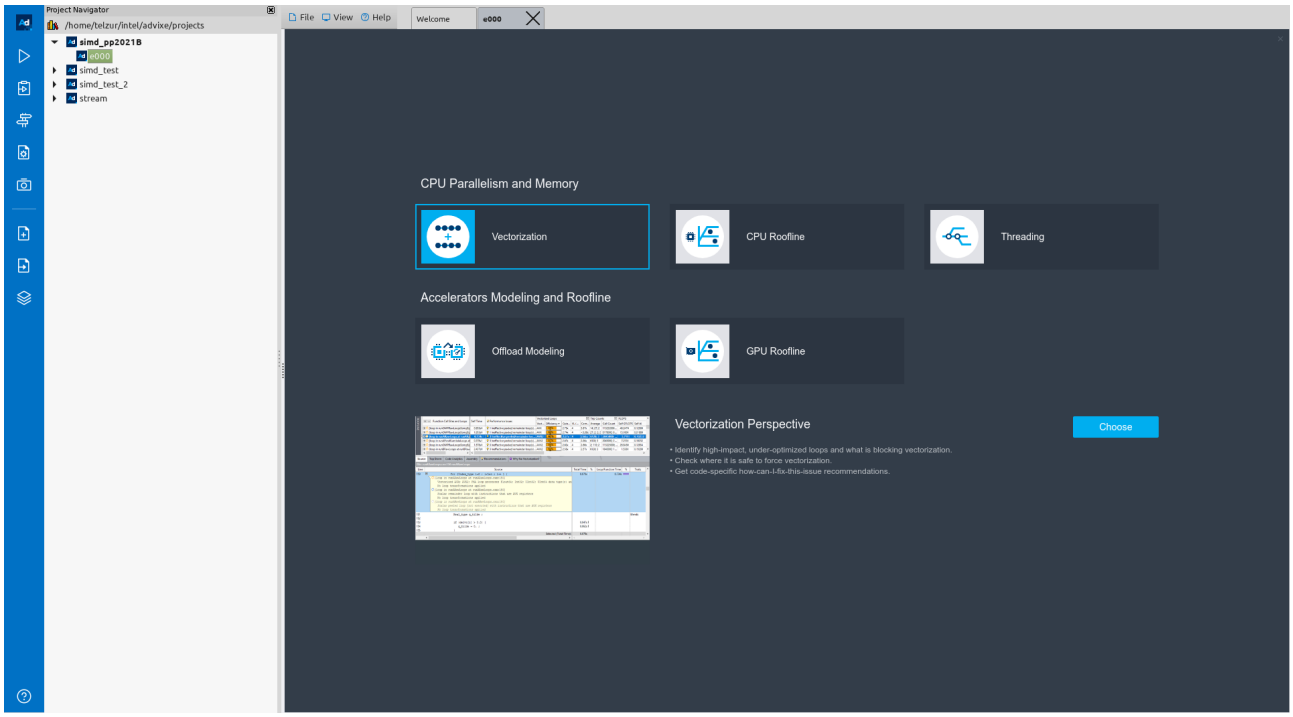
```
   remark #25084: Preprocess Loopnests: Moving Out Store    [
./simd_example.c(37,23) ]
   remark #15300: LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(37,5)
<Alternate Alignment Vectorized Loop>
LOOP END

LOOP BEGIN at ./simd_example.c(37,5)
<Remainder loop for vectorization>
   remark #15301: REMAINDER LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(37,5)
<Remainder loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(44,5)
<Peeled loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(44,5)
   remark #25084: Preprocess Loopnests: Moving Out Store    [
./simd_example.c(44,18) ]
   remark #15300: LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(44,5)
<Alternate Alignment Vectorized Loop>
LOOP END

LOOP BEGIN at ./simd_example.c(44,5)
<Remainder loop for vectorization>
   remark #15335: remainder loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to override
LOOP END

LOOP BEGIN at ./simd_example.c(63,5)
<Distributed chunk1>
   remark #25084: Preprocess Loopnests: Moving Out Store    [
./simd_example.c(63,18) ]
   remark #25426: Loop Distributed (2 way)
   remark #15301: PARTIAL LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(63,5)
<Remainder loop for vectorization, Distributed chunk1>
   remark #15335: remainder loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to override
LOOP END

LOOP BEGIN at ./simd_example.c(63,5)
<Distributed chunk2>
   remark #15301: PARTIAL LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(63,5)
<Remainder loop for vectorization, Distributed chunk2>
   remark #15335: remainder loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to override
LOOP END

LOOP BEGIN at ./simd_example.c(49,5)
```

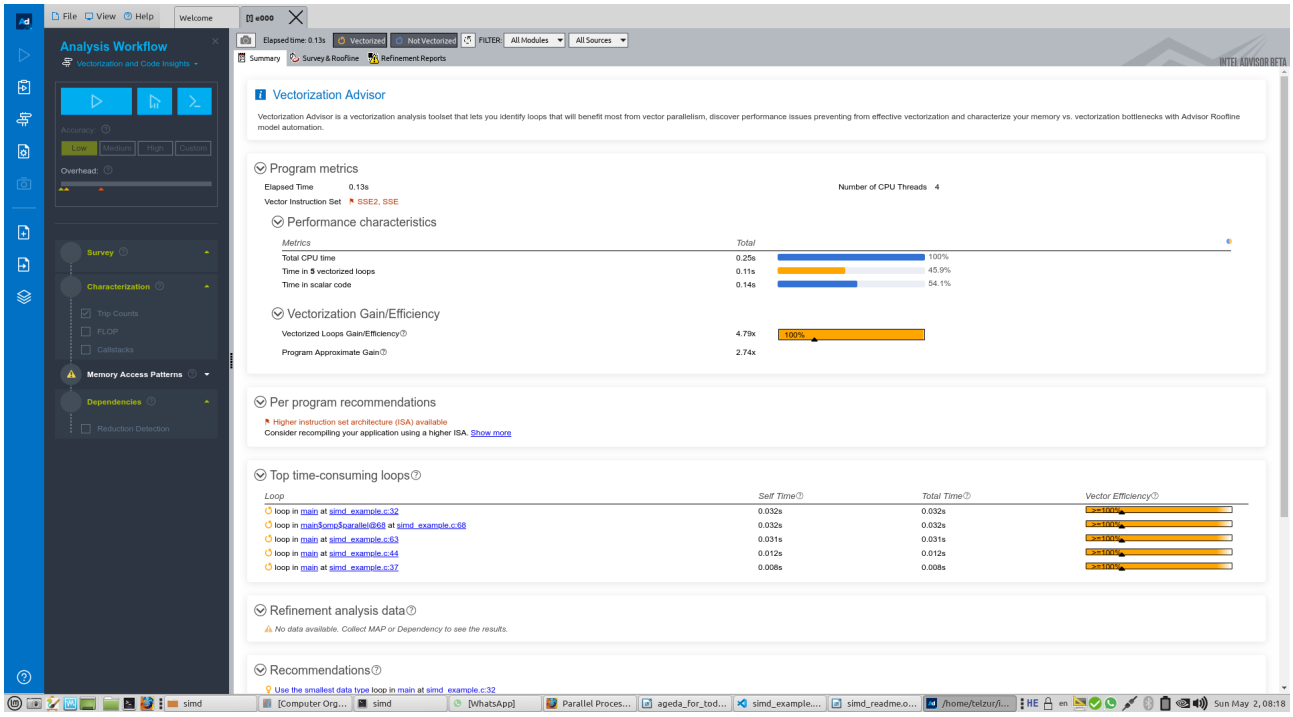
```
<Peeled loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(49,5)
   remark #15300: LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(49,5)
<Alternate Alignment Vectorized Loop>
LOOP END

LOOP BEGIN at ./simd_example.c(49,5)
<Remainder loop for vectorization>
   remark #15335: remainder loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to override
LOOP END

LOOP BEGIN at ./simd_example.c(68,5)
<Peeled loop for vectorization>
LOOP END

LOOP BEGIN at ./simd_example.c(68,5)
   remark #15301: SIMD LOOP WAS VECTORIZED
LOOP END

LOOP BEGIN at ./simd_example.c(68,5)
<Remainder loop for vectorization>
   remark #15335: remainder loop was not vectorized: vectorization possible but
seems inefficient. Use vector always directive or -vec-threshold0 to override
LOOP END

    Report from: Code generation optimizations [cg]

./simd_example.c(19,12):remark #34051: REGISTER ALLOCATION : [main]
./simd_example.c:19

    Hardware registers
        Reserved     :    2[ rsp rip]
        Available    :   39[ rax rdx rcx rbx rbp rsi rdi r8-r15 mm0-mm7 zmm0-
zmm15]
        Callee-save  :    6[ rbx rbp r12-r15]
        Assigned     :   20[ rax rdx rcx rbx rsi rdi r8-r15 zmm0-zmm5]

    Routine temporaries
        Total         :     556
            Global    :     160
            Local     :     396
        Regenerable   :     137
        Spilled       :       8

    Routine stack
        Variables     :      72 bytes*
            Reads     :      24 [7.71e-01 ~ 0.8%]
            Writes    :      25 [1.83e+00 ~ 1.8%]
        Spills        :     104 bytes*
            Reads     :      23 [6.33e-01 ~ 0.6%]
            Writes    :      23 [1.04e+00 ~ 1.0%]

    Notes

        *Non-overlapping variables and spills may share stack space,
         so the total stack size might be less than this.
```

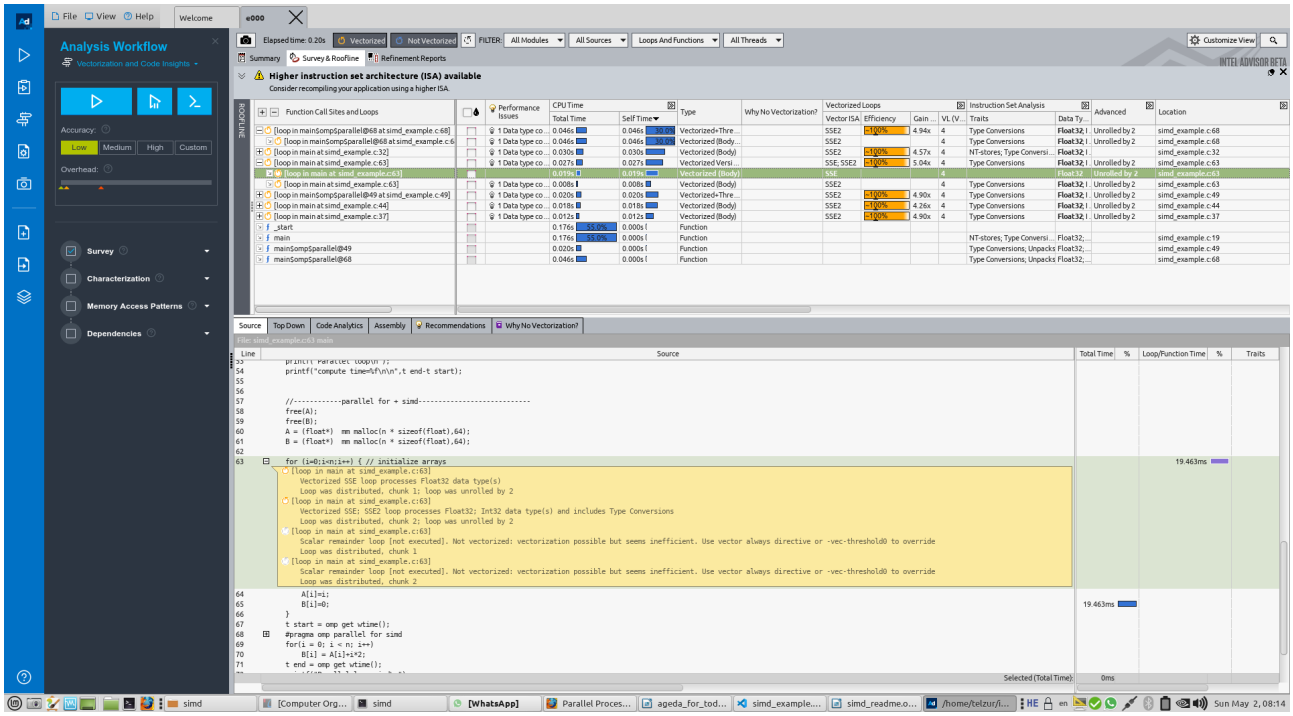
==========

8) **Profile**

Intel Advisor:
~/science/opt/intel/oneapi/advisor/2021.1-beta10/bin64/advisor-gui







Roofline report: