2020 IXPUG Annual Meeting

Intel® DPC++ Compatibility Tool Porting SPECFEM3D_GLOBE to DPC++

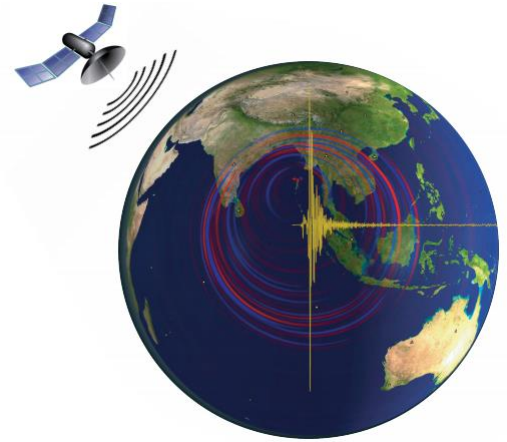
Sunny Gogar, Konstantinos Krommydas, Rama Kishan V Malladi and Philippe Thierry
Intel Architecture, Graphics, and Software

intel®

Agenda

- Introduction: SPECFEM3D_GLOBE
- Motivation – Why Data Parallel C++ (DPC++) for SPECFEM3D_GLOBE
- Introduction: Intel® DPC++ Compatibility Tool
- Migration Workflow
- Performance Comparison
- Summary

SPECFEM3D_GLOBE



- SPECFEM3D_GLOBE simulates global and regional (continental-scale) seismic wave propagation
- Use of BOAST automated-tool to generate OpenCL*/CUDA* kernels
- Official repo: https://github.com/geodynamics/specfem3d_globe
- Workloads analyzed based on

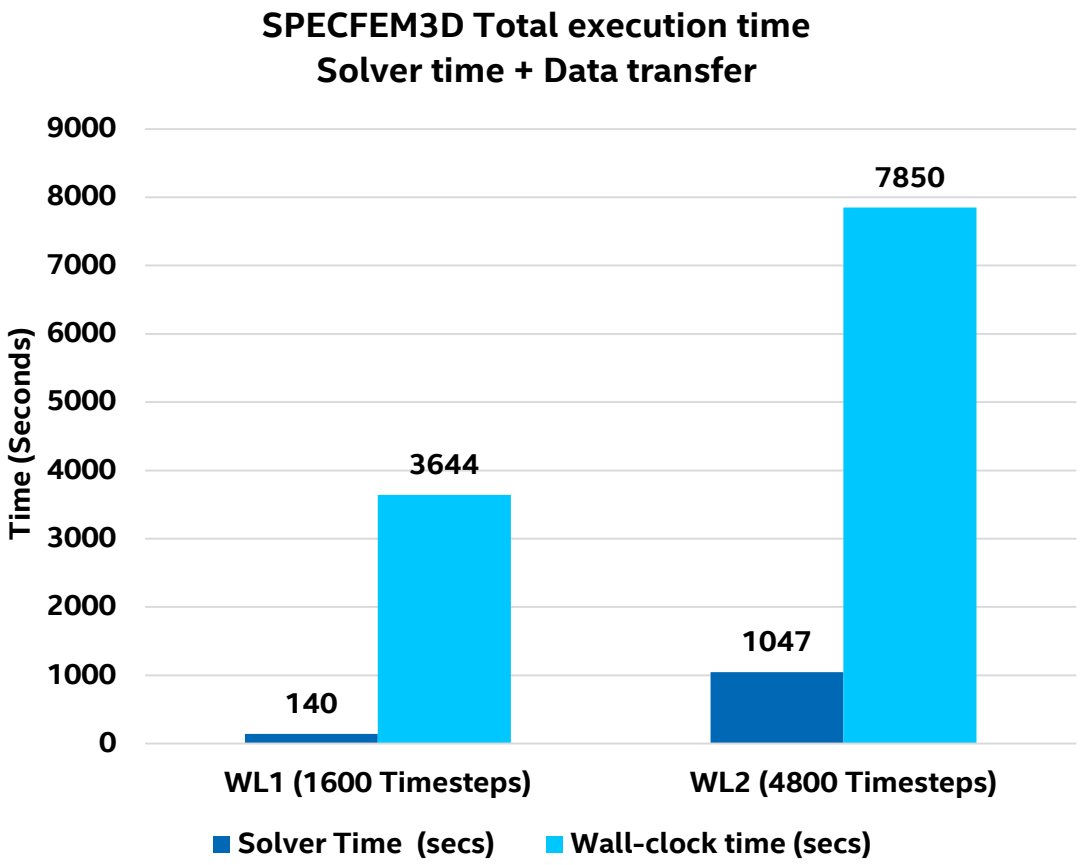
  competitive assessment, problem size:

  - WL1: 1D_isotropic_prem [64x64]
  - WL2: s362ani [80x80]

```
github.com/AlDanial/cloc v 1.74  T=1.44 s (370.8 files/s, 156306.8 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Fortran 90                     279          27677          41716         100021
C                               81           3145           5405          20851
CUDA                            88           1410           2286          10841
Ruby                            61            554            192           4365
make                            17            532            817           1887
C/C++ Header                     5            284            370            995
C++                              1            196            229            773
Markdown                         1             31              0            102
-------------------------------------------------------------------------------
SUM:                           533          33829          51015         139835
-------------------------------------------------------------------------------
```

Image source: specfem3d_globe-manual.pdf

Motivation – Why DPC++?



OpenCL Limitations: Inefficient copy of many small buffers

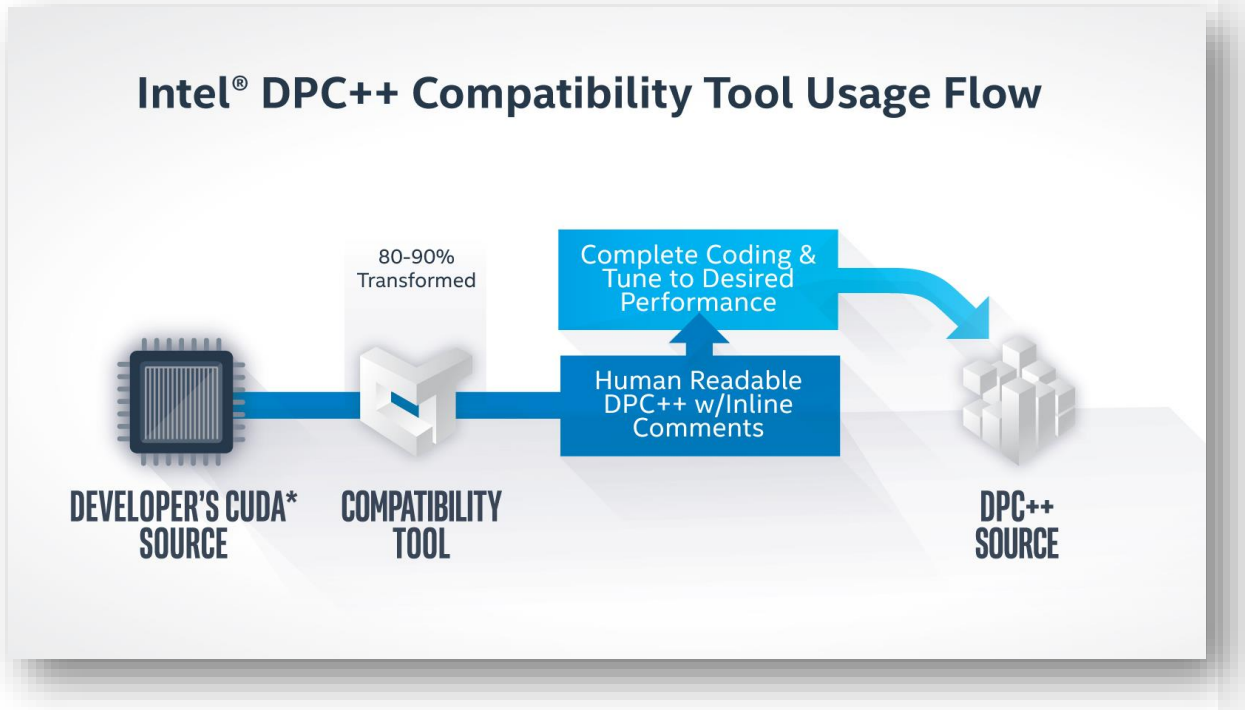
```
// mp->NSPEC_CRUST_MANTLE – order of several thousands

#ifdef USE_OPENCL
if (run_opencl) {
 int i;
 for (i = 0; i < mp->NSPEC_CRUST_MANTLE; i++) {
   int offset = i * NGLL3_PADDED * sizeof (realw);
   clCheck (clEnqueueWriteBuffer (mocl.command_queue,
         mp->d_xix_crust_mantle.ocl, CL_FALSE, offset,
         NGLL3 * sizeof (realw), &h_xix[i*NGLL3], 0,
         NULL, NULL));
.
.
#endif
#ifdef USE_CUDA
if (run_cuda) {
 // faster (small memcpy have low bandwidth...)
 print_CUDA_error_if_any(cudaMemcpy2D(mp->d_xix_crust_mantle.cuda,
                  NGLL3_PADDED*sizeof(realw), h_xix,
                  NGLL3*sizeof(realw), NGLL3*sizeof(realw),
                  mp->NSPEC_CRUST_MANTLE,
                  cudaMemcpyHostToDevice),1501);
.
.
#endif
```

OpenCL may not be the ideal choice to achieve portability with performance

See backup for configuration details on slide 13. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

Intel® DPC++ Compatibility Tool

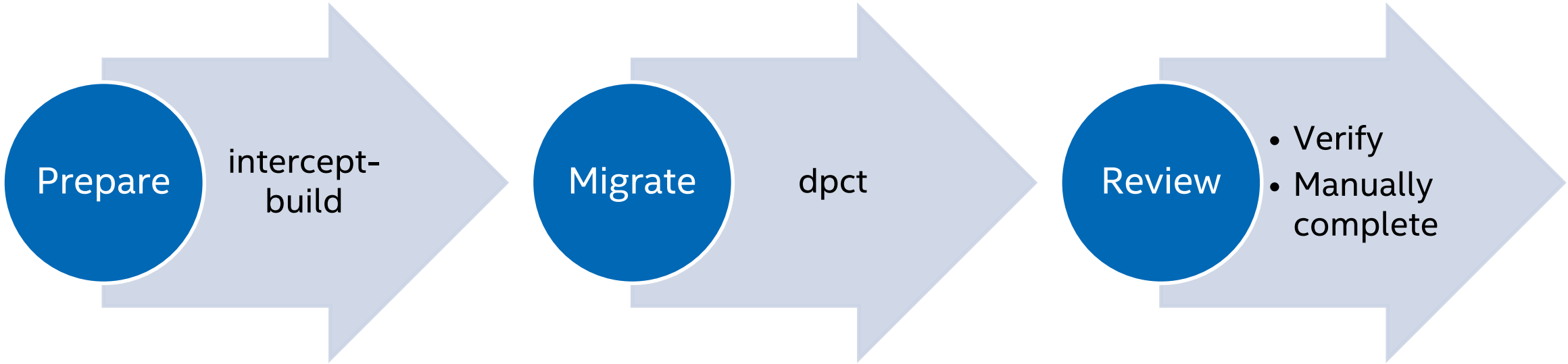
- Assists developers migrating code written in CUDA to DPC++ by generating DPC++ code wherever possible
- Expect up to 80-90% of code to migrate automatically
- Inline comments are provided to help developer complete code



Intel® DPC++ Compatibility Tool minimizes code migration time

Migration Workflow

Typical migration steps for simple to complex projects:



see Compatibility Tool – Diagnostics Reference

SPECFEM3D_GLOBE – Migration to DPC++

Prepare

```
$ git clone --recursive --branch devel
  https://github.com/geodynamics/specfem3d_globe.git
$ ./configure --with-cuda=cuda9    CUDA_LIB=${CUDA_ROOT}/lib64/ \
          CUDA_INC=${CUDA_ROOT}/include/    MPI_INC=${I_MPI_ROOT}/include/
$ intercept-build make -i
```

Migrate

```
$ dpct -p compile_commands.json
```

Review

Review diagnostics messages using reference and manually edit

Address other not-so-obvious issues

Diagnostics Reference

Compatibility Tool highlights issues with migration and code comments

/path/to/file:20:1: warning: DPCT10XX:0: text of the warning

//source code line for which warning was generated

ID	Message	Detailed Help	Suggestions to Fix
DPCT1000	An error handling `if-stmt` was detected but could not be rewritten. See the details in the resulting file comments.	The CUDA* API return error codes that are consumed by the program logic. SYCL* uses exceptions to report errors and does not return the error code. When the error handling logic in the original code is simple (for example, a print error message and exit), the code is removed in the resulting Data Parallel C++ (DPC++) application. The expectation is that SYCL throws an exception, which is handled with the printing of an exception message and exiting (the exception handler is generated automatically by the Intel® DPC++ Compatibility Tool). This warning is generated when the Intel® DPC++ Compatibility Tool detects more complex error handling than it considers safe to remove.	Review the error handling `if-statement` and try to rewrite it to use an exception handler instead.
DPCT1001	The statement could not be removed. See the details in the resulting file comments.	The Intel® DPC++ Compatibility Tool was not able to remove the code in the `then` clause of `if-stmt`. See `DPCT1000`.	See `DPCT1000`.
DPCT1002	A special case error handling `if-stmt` was detected. You may need to rewrite this code.	See `DPCT1000`	See `DPCT1000`.

see Compatibility Tool – Diagnostics Reference

Diagnostics Messages Breakdown

DPCT{diagnostics#} (count)	Summary
DPCT1000 (6), DPCT1001(6), DPCT1003 (111) , DPCT1009 (8), DPCT1010 (3) , DPCT1024 (2)	Different scenarios for error handling
DPCT1005 (4), DPCT1012 (4), DPCT1017 (10), DPCT1019 (1), DPCT1022 (1), DPCT1026 (5), DPCT1027 (3), DPCT1051 (4)	Unavailable equivalent API's in SYCL* (e.g. device versions, certain device properties, timing logic)
DPCT1039 (9)	Handling atomics (global atomics by default, local will need intervention)
DPCT1049 (59)	Validating use of work-group sizes

Other Issues (Solutions)

- Identified during compile-time or runtime

Function Pointers
(Rewrite)

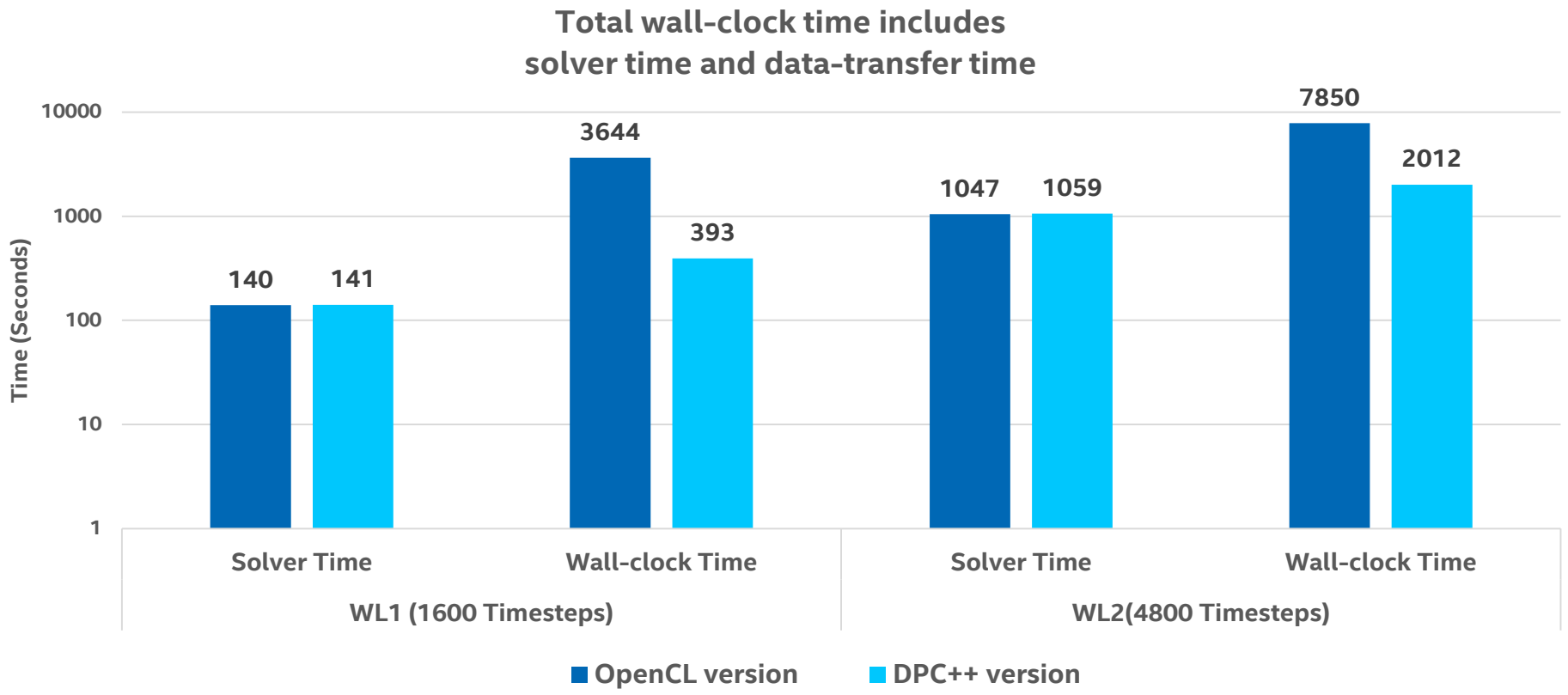
Texture Memory
API
(Disable)

GPU_ASYNC_COPY
(Disable)

Elapsed Time Logic
(Review and Fix)

Performance Comparison

Comparing DPC++ (Level Zero) and OpenCL on Intel® Gen9 HD Graphics



DPC++ with Level Zero backend improves the small buffer data transfer times without impacting solver execution time

See backup for configuration details on slide 13. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

Summary

- Intel® DPC++ Compatibility Tool can minimize migration time for source code written in CUDA to DPC++
- Using the Compatibility Tool we were able to port an application with 100K+ lines of source code
  - From [MPI + FORTRAN + CUDA] to [MPI + FORTRAN + DPC++]
- SYCL enables developers to tap-in performance and features enabled with primary backends like Level Zero, CUDA or ROCm* using the same source code

System Configuration

- Tested by Intel as of 10/08/2020. 1 socket of Intel® Core(TM) i7-7567U CPU @ 3.50GHz, 2 cores/socket, 2 Threads/core. Total Memory 32 GB (2 slots/ 16GB/ 2133 MT/s). BIOS: BNKBL357.86A.0062.2018.0222.1644, Ubuntu 18.04.4 LTS, 5.4.0-47-generic, SPECFEM3D_GLOBE (https://github.com/geodynamics/specfem3d_globe.git) –branch devel, ea6e5f6ccec884c8aab8a097115b99173a592e2, Intel® oneAPI Base Toolkit (Beta09), Intel® oneAPI HPC Toolkit (Beta09)
- GPU: Intel(R) OpenCL HD Graphics, 1150MHz, 48 compute units, global memory size: 25.05GiB, Max memory allocation 4GiB, Max work group size:256
- Commands:
  - git clone --recursive --branch devel https://github.com/geodynamics/specfem3d_globe.git
  - For OpenCL
    - ./configure --with-opencl OCL_LIBS=/opt/intel/oneapi/compiler/latest/linux/lib/libOpenCL.so OCL_INC=/opt/intel/oneapi/compiler/latest/linux/lib/oclfpga/host/include/ MPI_INC=/opt/intel/oneapi/mpi/latest/include/ --
  - Test Workloads:
    - WL1: MODEL=1D_isotropic_prem; NEX_XI=NEX_ETA=64; NPROC_XI=NPROC_ETA=1; NCHUNKS=1 ; RECORD_LENGTH_IN_MINUTES = 2.5d0 STEPS=1600
    - WL2 – Small_benchmark_run_to_test_more_complex_Earth: MODEL=s362ani; NEX_XI=NEX_ETA=80; NPROC_XI=NPROC_ETA=1; NCHUNKS=1; RECORD_LENGTH_IN_MINUTES = 8.0d0 STEPS=4700)
  - time bin/xmeshfem3d
  - time bin/xspecfem3d

References

- SPECFEM3D_GLOBE: Komatitsch, D.; Vilotte, J.-P.; Tromp, J.; Afanasiev, M.; Bozdag, E.; Charles, J.; Chen, M.; Goddeke, D.; Hjorleifsdottir, V.; Labarta, J.; Le Goff, N.; Le Loher, P.; Liu, Q.; Maggi, A.; Martin, R.; McRitchie, D.; Messmer, P.; Michea, D.; Nissen-Meyer, T.; Peter, D.; Rietmann, M.; de Andrade, S.; Savage, B.; Schuberth, B.; Siemenski, A.; Strand, L.; Tape, C.; Xie, Z.; Zhu, H. (2020), SPECFEM3D GLOBE [software], Computational Infrastructure for Geodynamics, doi: GITHASH8, url: https://geodynamics.org/cig/software/specfem3d_globe/
- Intel® oneAPI: A Unified X-Architecture Programming Model
- Intel® DPC++ Compatibility Tool
- oneAPI Level Zero Specification
- Migrating Your Existing CUDA Code to DPC++

Notices and Disclaimers

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.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

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.


intel