Accelerate Math Processing Routines, Increase Application Performance

Intel® oneAPI Math Kernel Library (oneMKL)

Fastest & Most Used Math Library for Intel® Architecture-based Systems

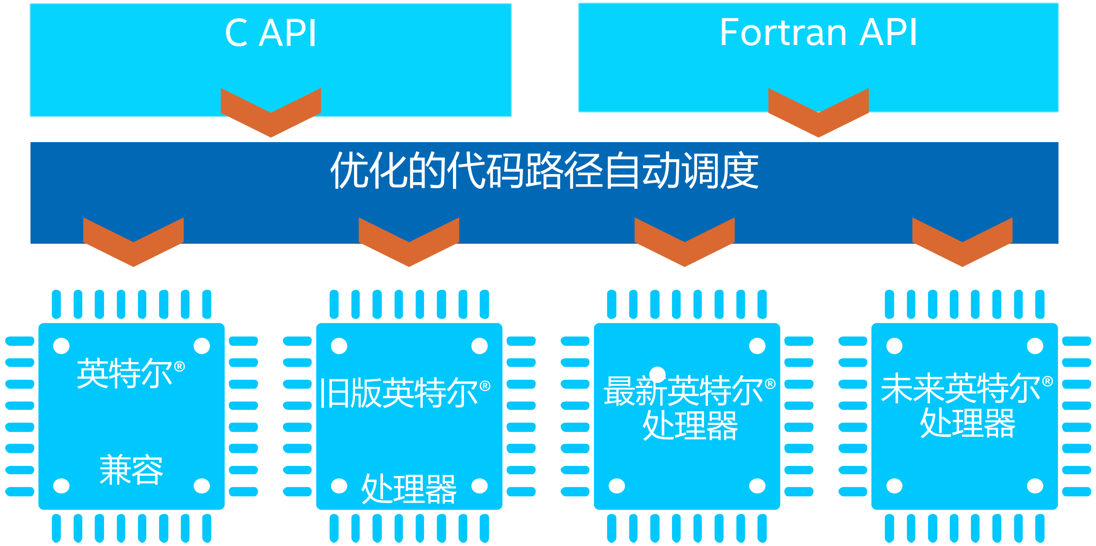
Cao Ruqiu
Intel Software & Technology Group (SATG)

Outline

- 英特尔数学核心函数库(oneMKL)介绍
- oneMKL 如何跨平台使用
- oneMKL 用例介绍
- oneMKL 基于Intel第4代 Xeon 服务器性能数据
- oneMKL 丰富的线上资源
- Q&A

传统英特尔® 数学核心函数库简介

- 高度优化的线程化数学例程
  - 性能至关重要！
- 行业领先的数学函数库
  - 广泛应用于科学、工程和数据处理领域
- 面向最新及下一代英特尔® 处理器进行调试



《2016 年 EDC 北美开发调查》第 I 卷

更多数学函数库用户依赖 MKL

感知多处理器

- 交叉平台支持
- 矢量化、线程化和能够感知分布式多处理器

英特尔® oneMKL 的内部结构

加速 HPC、企业、物联网和云应用等

线性代数
- BLAS
- LAPACK
- ScaLAPACK
- Extended Eigensolver

FFT
- 多维
- FFTW 接口
- 集群 FFT

稀疏矩阵求解器
- 稀疏 BLAS
- PARDISO*
- 迭代稀疏矩阵解算器
- 集群稀疏矩阵解算器

矢量随机数
- 同余
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neiderreiter
- 非确定性

汇总统计
- 峰态
- 变异系数
- 次序统计
- 最小值/最大值
- 方差-协方差

矢量数学
- 三角函数
- 双曲函数
- 指数函数
- log
- 幂
- 根

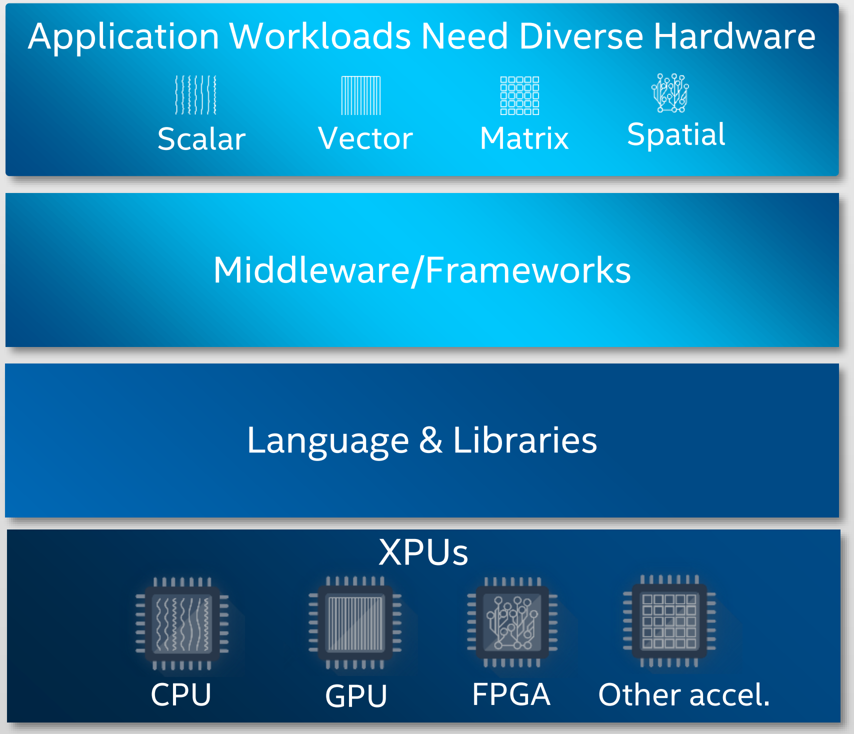
其他
- 样条函数
- 插值
- 置信域
- 快速泊松求解器

英特尔® 架构平台

操作系统：Windows*、Linux*、MacOS[1]*

Programming Challenges for Multiple Architectures

- Growth in specialized workloads
- No common programming language or APIs
- Inconsistent tool support across platforms
- Each platform requires unique software investment
- Diverse set of data-centric hardware required



Intel® oneAPI Tools for HPC
Intel® oneAPI HPC Toolkit

Deliver Fast Applications that Scale

What is it?

A toolkit that adds to the Intel® oneAPI Base Toolkit for building high-performance, scalable parallel code on C++, SYCL, Fortran, OpenMP & MPI from enterprise to cloud, and HPC to AI applications.

Who needs this product?

- OEMs/ISVs
- C++, Fortran, OpenMP, MPI Developers

Why is this important?

- Accelerate performance on Intel® Xeon® and Core™ Processors and Intel® Accelerators
- Deliver fast, scalable, reliable parallel code with less effort built on industry standards

Learn More: intel.com/oneAPI-HPCKit

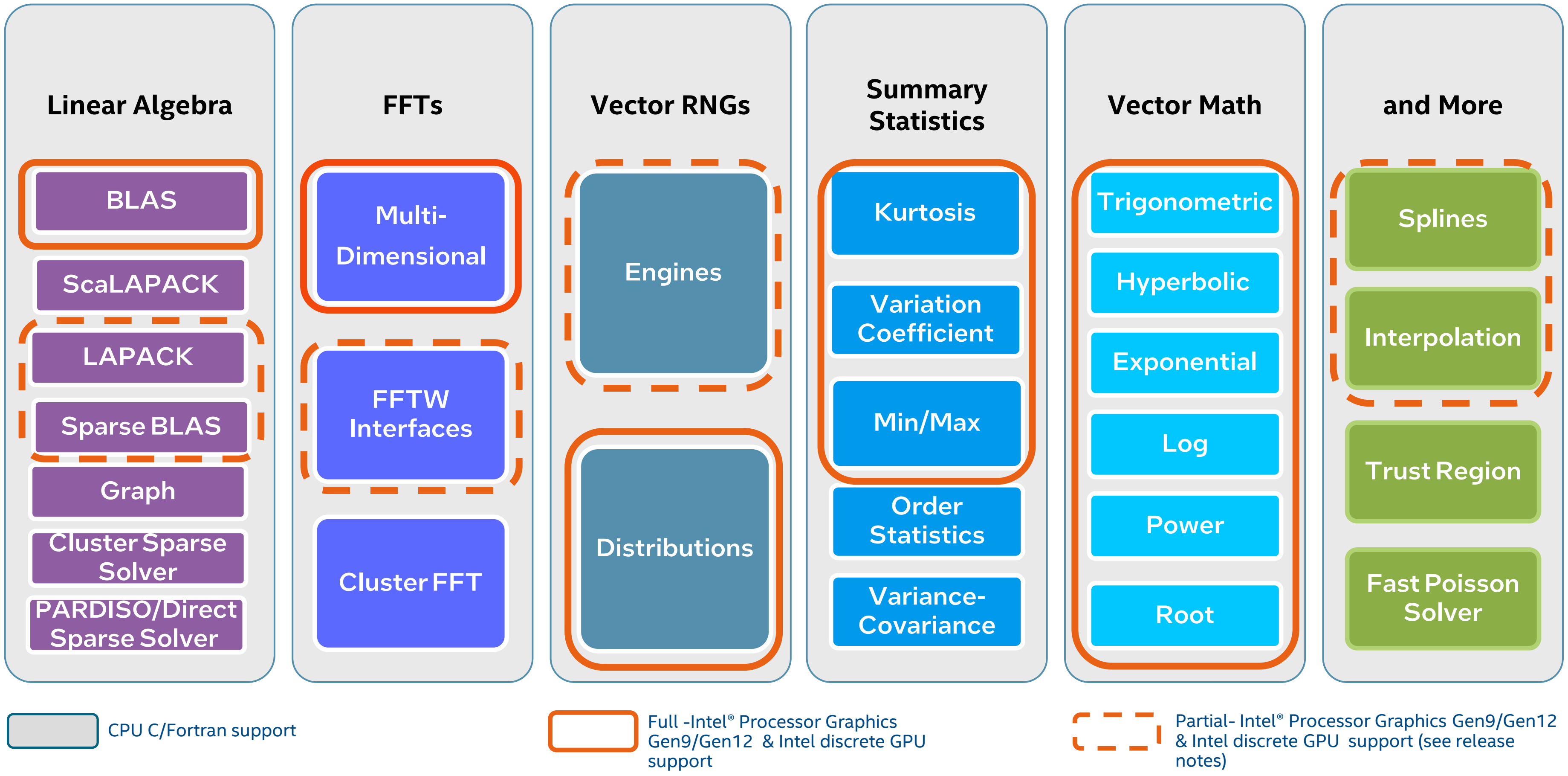
Intel® oneAPI Base & HPC Toolkits

Direct Programming	API-Based Programming	Analysis & debug Tools
Intel® C++ Compiler Classic	Intel® MPI Library	Intel® Inspector
Intel® Fortran Compiler Classic	Intel® oneAPI DPC++ Library oneDPL	Intel® Trace Analyzer & Collector
Intel® Fortran Compiler	Intel® oneAPI Math Kernel Library - oneMKL	Intel® Cluster Checker
Intel® oneAPI DPC++/C++ Compiler	Intel® oneAPI Data Analytics Library - oneDAL	Intel® VTune™ Profiler
Intel® DPC++ Compatibility Tool	Intel® oneAPI Threading Building Blocks - oneTBB	Intel® Advisor
Intel® Distribution for Python	Intel® oneAPI Video Processing Library - oneVPL	Intel® Distribution for GDB
Intel® FPGA Add-on for oneAPI Base Toolkit	Intel® oneAPI Collective Communications Library oneCCL	
	Intel® oneAPI Deep Neural Network Library - oneDNN	
	Intel® Integrated Performance Primitives – Intel® IPP	

Intel® oneAPI **HPC** Toolkit +
Intel® oneAPI **Base** Toolkit

intel.
1
oneAPI
HPC TOOLKIT

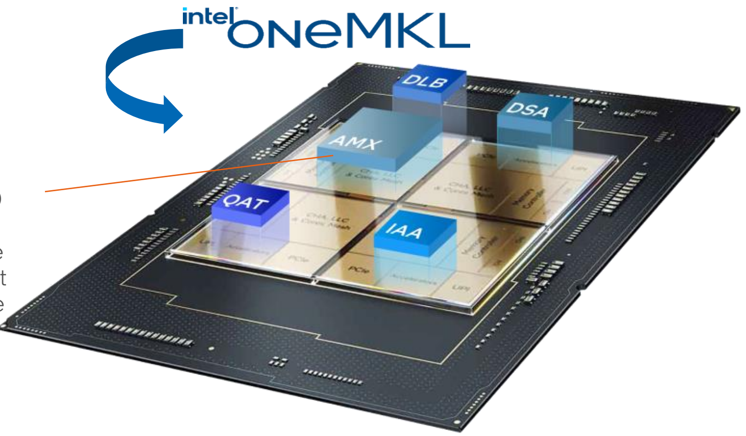
What’s Inside Intel® oneAPI Math Kernel Library (oneMKL)



What’s New for Intel® oneAPI Math Kernel Library (oneMKL) 2023.0

Better GPU Performance + Intel® Data Center GPU Max Series & 4th Gen Intel® Xeon® Scalable processors Support

Advanced Matrix Extensions (AMX)
Built-in AI acceleration engine delivers a significant leap in performance for deep learning inference and training.



4th Gen Intel® Xeon® Scalable Processors with Intel® Advanced Matrix Extensions, Quick assist Technology, Intel® AVX-512, bfloat16, and more built-in accelerators

Intel® Data Center GPUs with hardware AV1 encode and Max with datatype flexibility, Intel® X^e Matrix Extensions, vector engine, XE-Link, and other features

- Intel® oneAPI Math Kernel Library increases CUDA library function API compatibility coverage for BLAS, LAPACK, sparse BLAS, FFT, vector math, summary statistics, splines, and more; easing code migration to oneAPI and Intel GPUs.
- On 4th Gen Intel® Xeon® Scalable processors, oneMKL leverages Intel® XMX to optimize matrix multiply computations for TF32, FP16, BF16, and INT8 data types; and provides interfaces for SYCL and C/Fortran OpenMP offload programming.
- Enabled C OpenMP offload functionality for 1/2/3D real Fast Fourier Transform (FFT) using FFTW3 APIs. Improved performance for 1/2D complex scaled FFTs on CPU. Improved performance for 1/2/3D real single precision and 1/2/3D complex FFT performance on Intel® Data Center GPU Max Series.
- Broadened oneMKL's scope by including optimizations for next. gen CPU and GPU including DGEMM, SGEMM, Systolic GEMM, DGETRF, DPOTRF, FFT SP/DP and RNG functions.
- Expanded support for AMX/bfloat16 & AVX512/float16 for 4th Gen. Intel® Xeon® Scalable Processor.
- Improved performance for BLAS Level-2 and Level-3 routines on Intel GPUs including Intel® Data Center GPU Max Series.

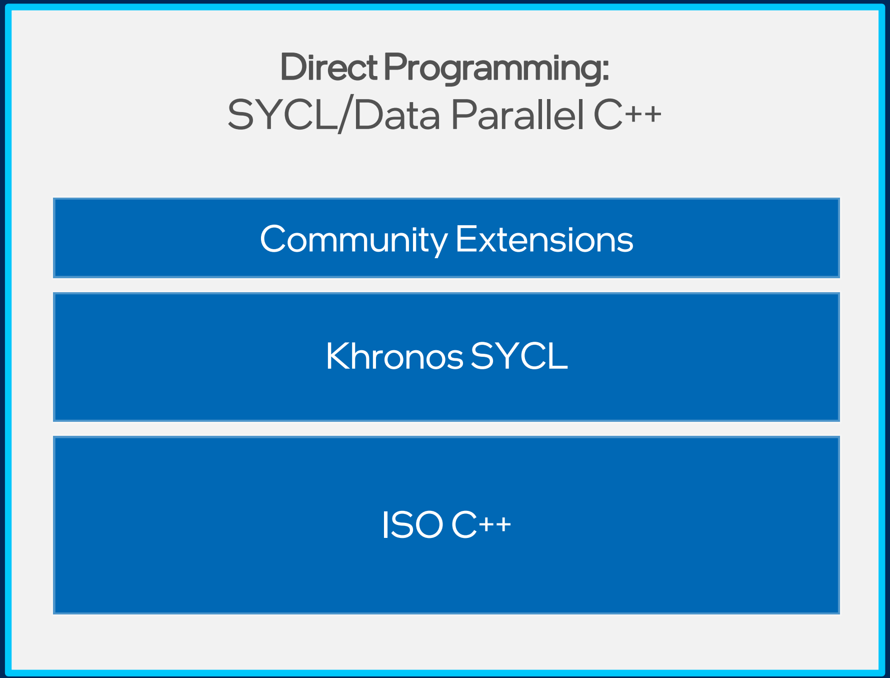
For more details, review oneMKL Release Notes

Data Parallel C++ (DPC++) Introduction

- SYCL is a C++-based, single-source programming language for heterogeneous computing.
- DPC++ is SYCL + many new extensions.
  - *e.g.* pointer-based programming (Unified Shared Memory)
- Open, standards-based, multi-vendor.

DPC++ = ISO C++ and Khronos SYCL and community extensions

https://software.intel.com/en-us/oneapi



oneMKL Traditional C API: GEMM Example

$$C \leftarrow \alpha AB + \beta C$$

```
int main() {

    int64_t m = 10, n = 6, k = 8, lda = 12, ldb = 8, ldc = 10;
    int64_t sizea = lda * k, sizeb = ldb * n, sizec = ldc * n;
    double alpha = 1.0, beta = 0.0;

    // Allocate matrices
    double *A = (double *) mkl_malloc(sizeof(double) * sizea);
    double *B = (double *) mkl_malloc(sizeof(double) * sizeb);
    double *C = (double *) mkl_malloc(sizeof(double) * sizec);

    // Initialize matrices here
    …
    cblas_dgemm(CblasColMajor, CblasNoTrans, CblasNoTrans, m, n, k,
                alpha, A, lda, B, ldb, beta, C, ldc);
    …
}
```

oneMKL SYCL API: GEMM Example

```
int main() {
    using namespace oneapi::mkl;

    int64_t m = 10, n = 6, k = 8, lda = 12, ldb = 8, ldc = 10;
    int64_t sizea = lda * k, sizeb = ldb * n, sizec = ldc * n;
    double alpha = 1.0, beta = 0.0;

    sycl::queue Q{sycl::gpu_selector{}};

    // Allocate matrices
    double *A = malloc_shared<double>(sizea, Q);
    double *B = malloc_shared<double>(sizeb, Q);
    double *C = malloc_shared<double>(sizec, Q);

    // Initialize matrices here
    …
    auto e = blas::gemm(Q, transpose::N, transpose::N, m, n, k,
                        alpha, A, lda, B, ldb, beta, C, ldc);
    …
}
```

$C \leftarrow \alpha AB + \beta C$

Set up GPU queue

Allocate CPU/GPU-accessible shared memory

New oneMKL SYCL API

Computation is performed on given queue

Output **e** is a sycl::event object representing command completion

Call `e.wait()` to wait for completion

oneMKL C OpenMP Offload: GEMM

```
int main() {
    long m = 10, n = 6, k = 8, lda = 12, ldb = 8, ldc = 10;
    long sizea = lda * k, sizeb = ldb * n, sizec = ldc * n;
    double alpha = 1.0, beta = 0.0;

    // Allocate matrices
    double *A = (double *) mkl_malloc(sizeof(double) * sizea, 64);
    double *B = (double *) mkl_malloc(sizeof(double) * sizeb, 64);
    double *C = (double *) mkl_malloc(sizeof(double) * sizec, 64);

    // Initialize matrices here
    …
#pragma omp target data map(to:A[0:sizea],B[0:sizeb]) map(tofrom:C[0:sizec])
    {
#pragma omp dispatch nowait
        {
            // Compute C = A * B on GPU
            cblas_dgemm(CblasColMajor, CblasNoTrans, CblasNoTrans, m, n, k,
                        alpha, A, lda, B, ldb, beta, C, ldc);
        }
    }
    …
}
```

$C \leftarrow \alpha AB + \beta C$

Use `target data map` to send matrices to the device

Use `omp dispatch` to request GPU execution for cblas_dgemm

Optional `nowait` clause for asynchronous execution
Use `#pragma omp taskwait` for synchronization

DFTI interface routines

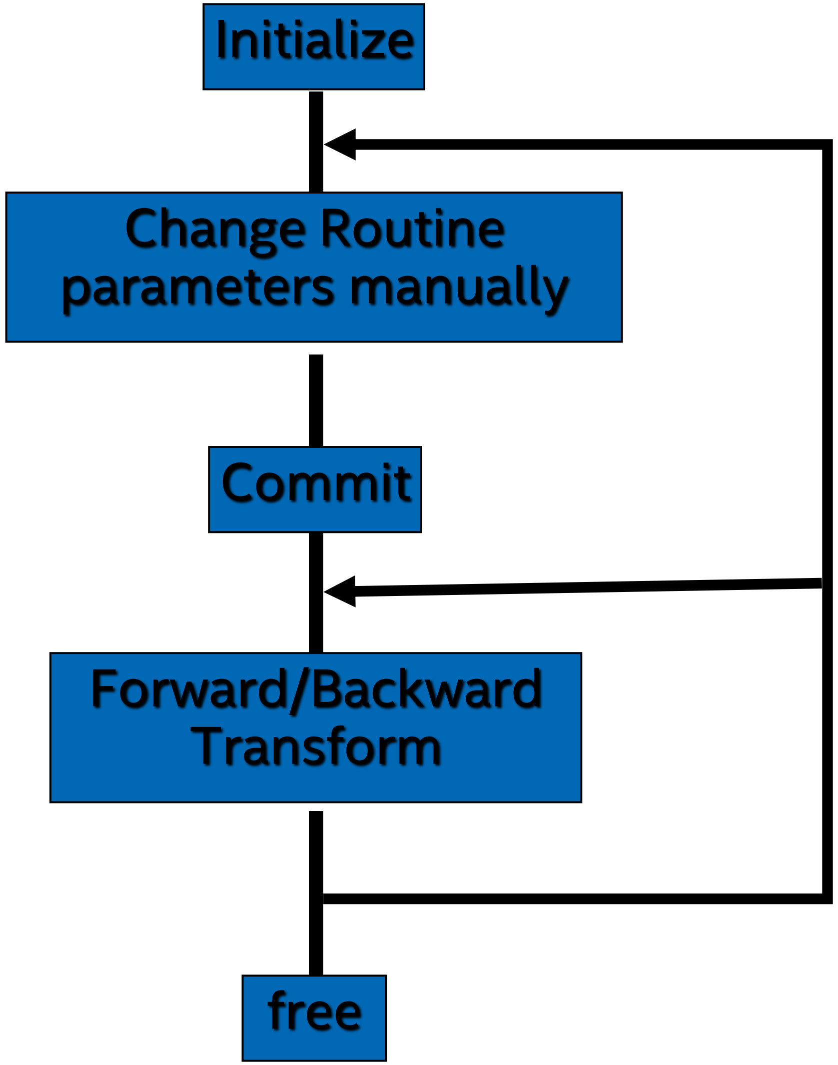
DftiCreateDescriptor

DftiSetValue

DftiCommitDescriptor

DftiComputeForward
DftiComputeBackward

DftiFreeDescriptor

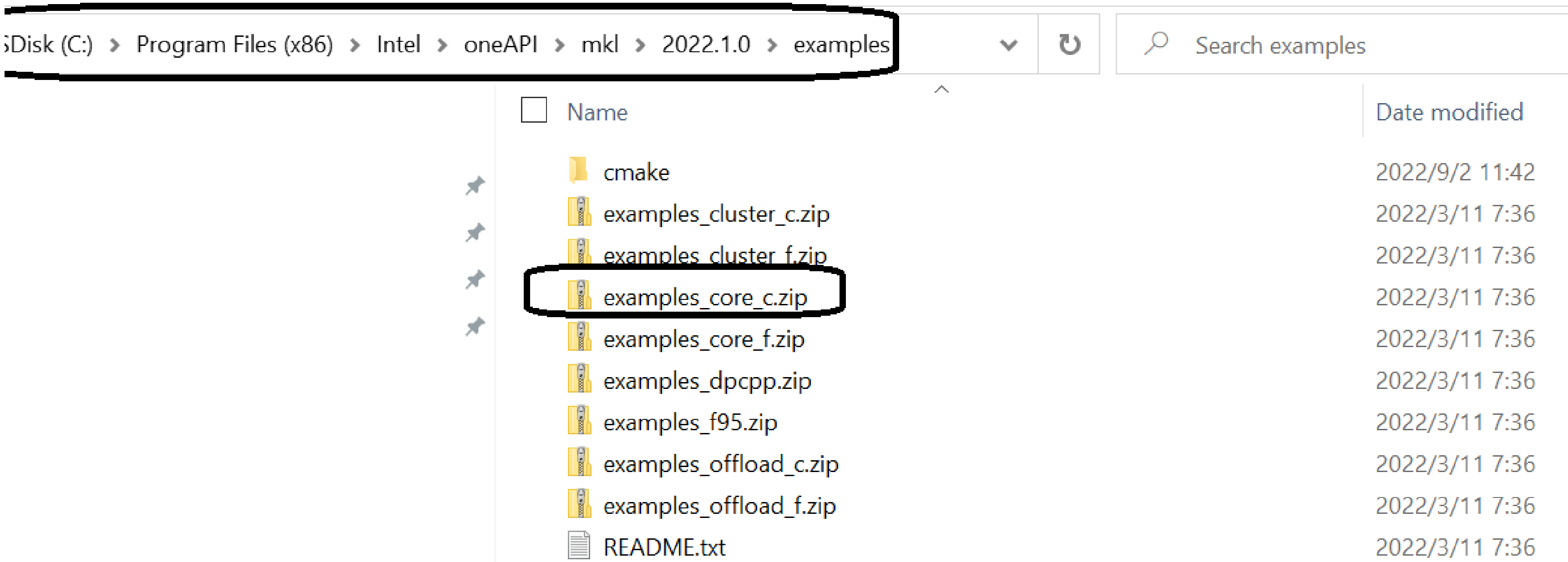


DFTI interface example program

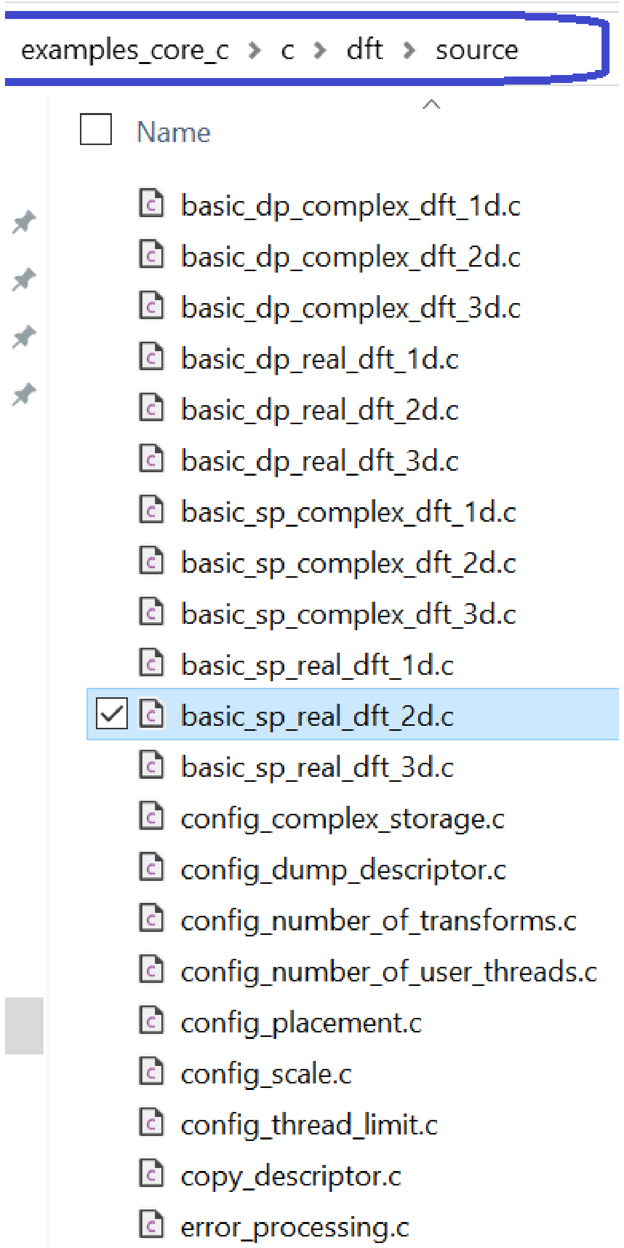
- Complex-to-complex 1D transform for double precision data not inplace.
- /* Create Dfti descriptor for 1D double precision transform */
  Status = DftiCreateDescriptor(&Desc_Handle, DFTI_DOUBLE, DFTI_COMPLEX, 1, n);
- /* Set placement of result DFTI_NOT_INPLACE */
  Status = DftiSetValue(Desc_Handle, DFTI_PLACEMENT, DFTI_NOT_INPLACE);
- /* Commit Dfti descriptor */
  Status = DftiCommitDescriptor(Desc_Handle);
- /* Compute Forward transform */
  Status = DftiComputeForward(Desc_Handle, x_in, x_out); //实际计算耗时就这一步
- /* Free DFTI descriptor */
  Status = DftiFreeDescriptor(&Desc_Handle);

oneMKL examples location

)\Intel\oneAPI\mkl\2022.1.0\examples




```
[rcao8@ortce-skl examples]$ pwd
/opt/intel/oneapi/mkl/2023.1.0/examples
[rcao8@ortce-skl examples]$ ll
total 4052
drwxr-xr-x 2 tmefford cthorsec      82 Apr  6 15:36 cmake
-rw-r--r-- 1 tmefford cthorsec   31529 Mar  7 17:32 examples_cluster_c.tgz
-rw-r--r-- 1 tmefford cthorsec   37228 Mar  7 17:32 examples_cluster_f.tgz
-rw-r--r-- 1 tmefford cthorsec 1378364 Mar  7 17:32 examples_core_c.tgz
-rw-r--r-- 1 tmefford cthorsec 1371536 Mar  7 17:32 examples_core_f.tgz
-rw-r--r-- 1 tmefford cthorsec 1030902 Mar  7 17:32 examples_dpcpp.tgz
-rw-r--r-- 1 tmefford cthorsec   30944 Mar  7 17:32 examples_f95.tgz
-rw-r--r-- 1 tmefford cthorsec  126528 Mar  7 17:32 examples_offload_c.tgz
-rw-r--r-- 1 tmefford cthorsec  112263 Mar  7 17:32 examples_offload_f.tgz
-rw-r--r-- 1 tmefford cthorsec   12545 Mar  7 17:32 README.txt
```



oneMKL Link Line Advisor

[Intel® oneMKL Link Line Advisor]

Intel® oneAPI Math Kernel Library (oneMKL) Link Line Advisor v6.20

Reset

Select Intel® product:	oneMKL 2023
Select OS:	Linux*
Select programming language:	C/C++
Select compiler:	Intel(R) C/C++ Classic
Select architecture:	Intel(R) 64
Select dynamic or static linking:	Static
Select interface layer:	C API with 64-bit integer
Select threading layer:	OpenMP threading
Select OpenMP library:	Intel(R) (libiomp5)
Enable OpenMP offload feature to GPU:	☐
Select cluster library:	☐ Parallel Direct Sparse Solver for Clusters (BLACS required) ☐ Cluster Discrete Fast Fourier Transform (BLACS required) ☐ ScaLAPACK (BLACS required) ☐ BLACS
Select MPI library:	<Select MPI>
Select the Fortran 95 interfaces:	☐ BLAS95 ☐ LAPACK95
Link with Intel® oneMKL libraries explicitly:	☑
Link with DPC++ debug runtime compatible libraries:	☐

Use this link line:

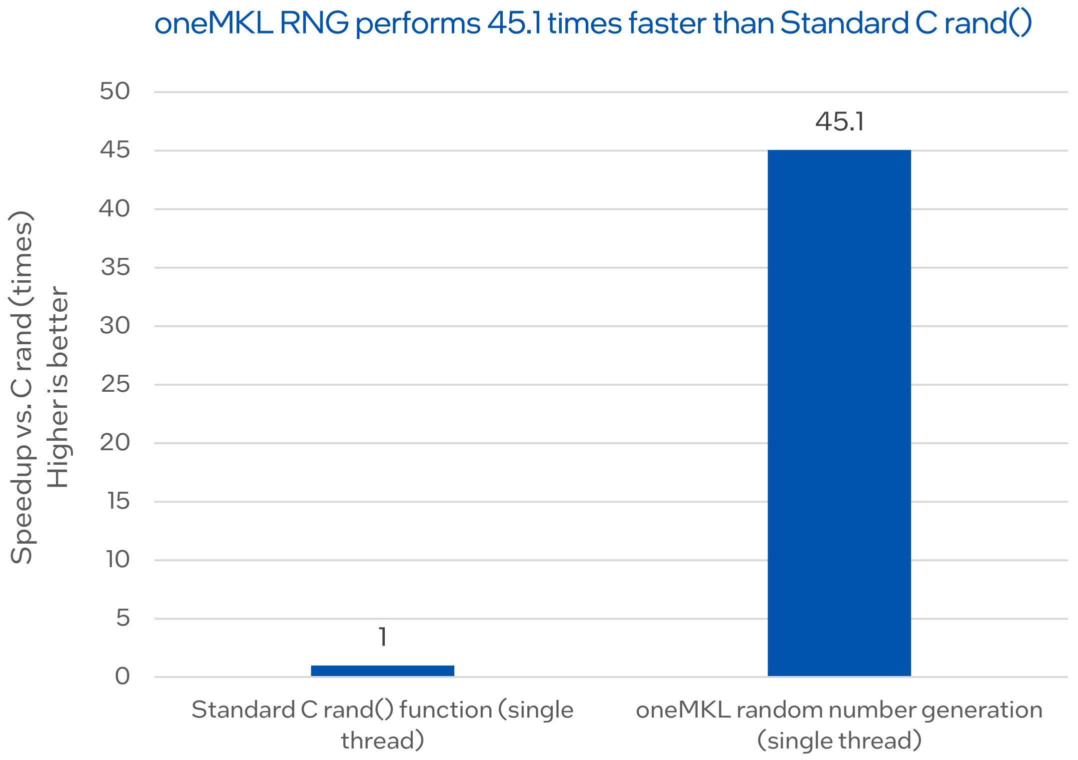
```
 -Wl,--start-group ${MKLROOT}/lib/intel64/libmkl_intel_ilp64.a
${MKLROOT}/lib/intel64/libmkl_intel_thread.a ${MKLROOT}/lib/intel64/libmkl_core.a
-Wl,--end-group -liomp5 -lpthread -lm -ldl
```

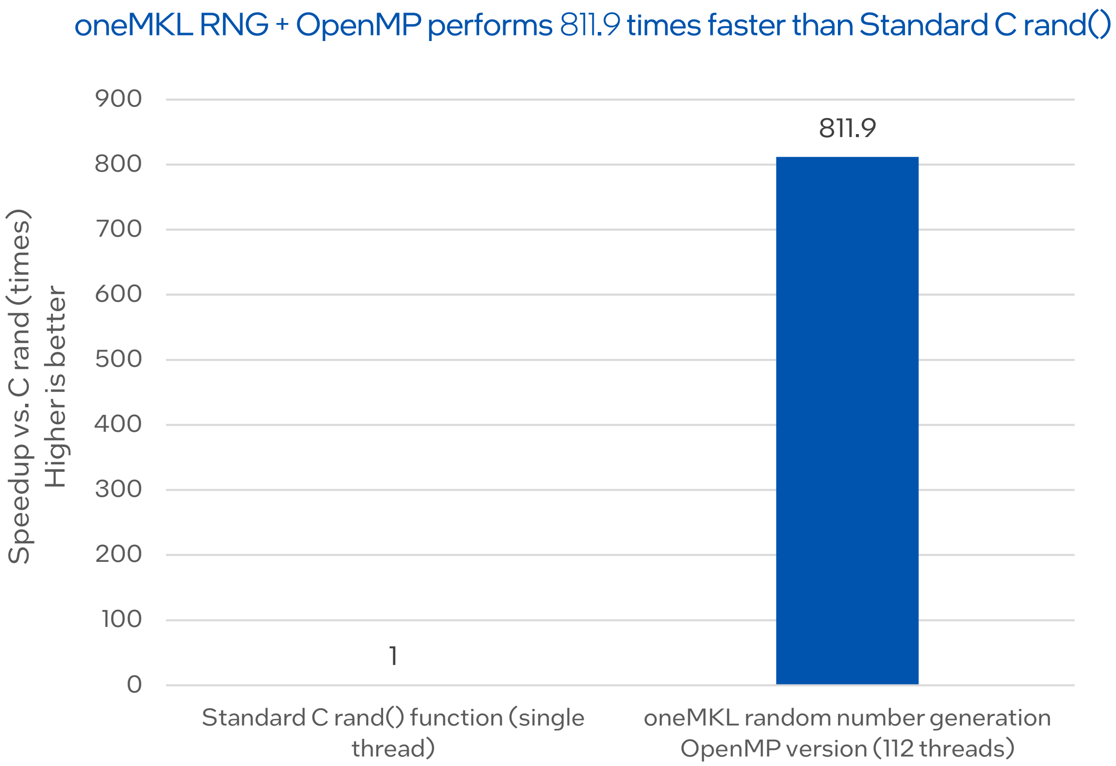
Compiler options:

```
 -DMKL_ILP64  -I"${MKLROOT}/include"
```

oneMKL Random Number Generator (RNG) Performance

Intel® oneAPI Math Kernel Library (oneMKL) 2023.0 RNG vs. C rand() on 4th Gen Intel® Xeon® Scalable Processor





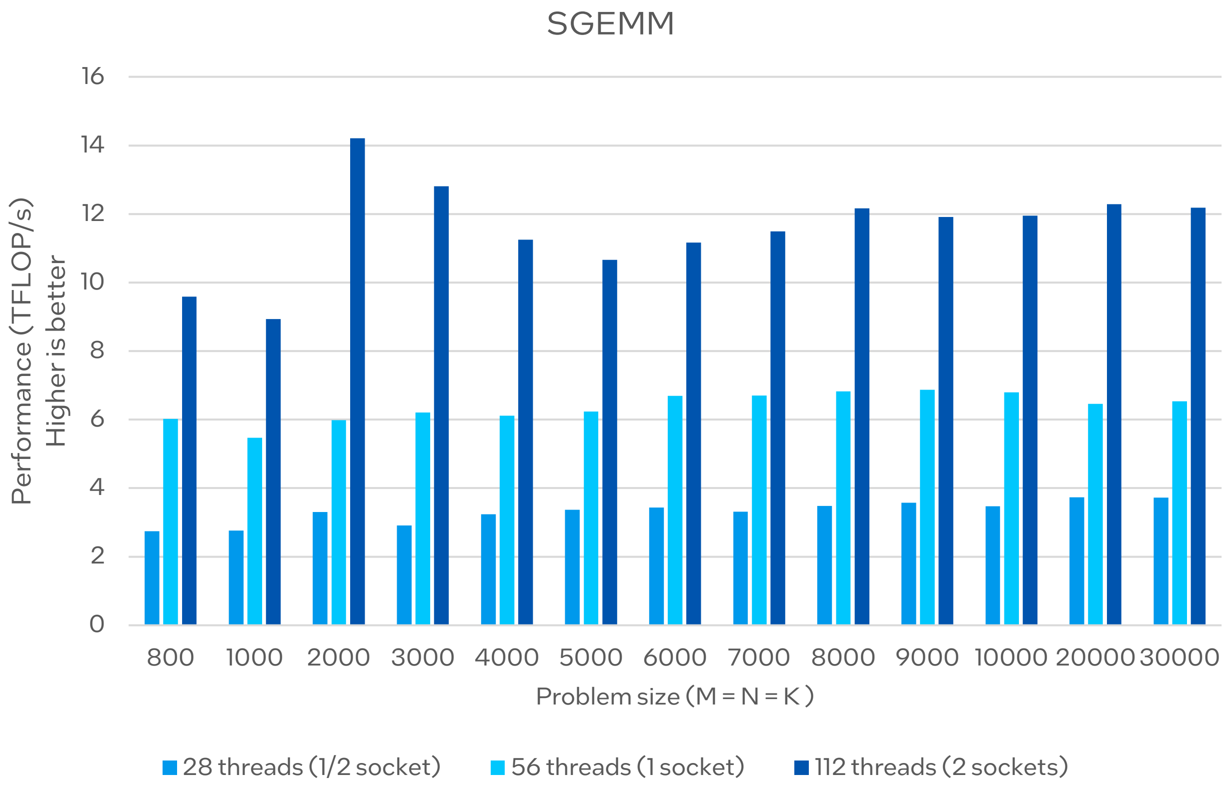
Testing Date: Performance results are **based on testing by Intel as of February 6, 2023** and may not reflect all publicly available security updates.

Configuration Details and Workload Setup: 1-node, 2x Intel® Xeon® Platinum 8480+ processor on Denali Pass platform with 1024 GB (16 slots/ 64GB/ 4800) total DDR5 memory, ucode 0x2b000161, HT off, Turbo on, Ubuntu 22.04.1 LTS, 5.17.0-051700-generic, 1x Intel® SSD 3.5TB OS Drive; Intel® oneAPI Math Kernel Library 2023.0 (oneMKL). Pi-number Evaluation by Monte Carlo Simulations Benchmark; Basic random number generator is mcg59, random number distribution is double precision uniform distribution; Intel® oneAPI Math Kernel Library 2023.0 (oneMKL).

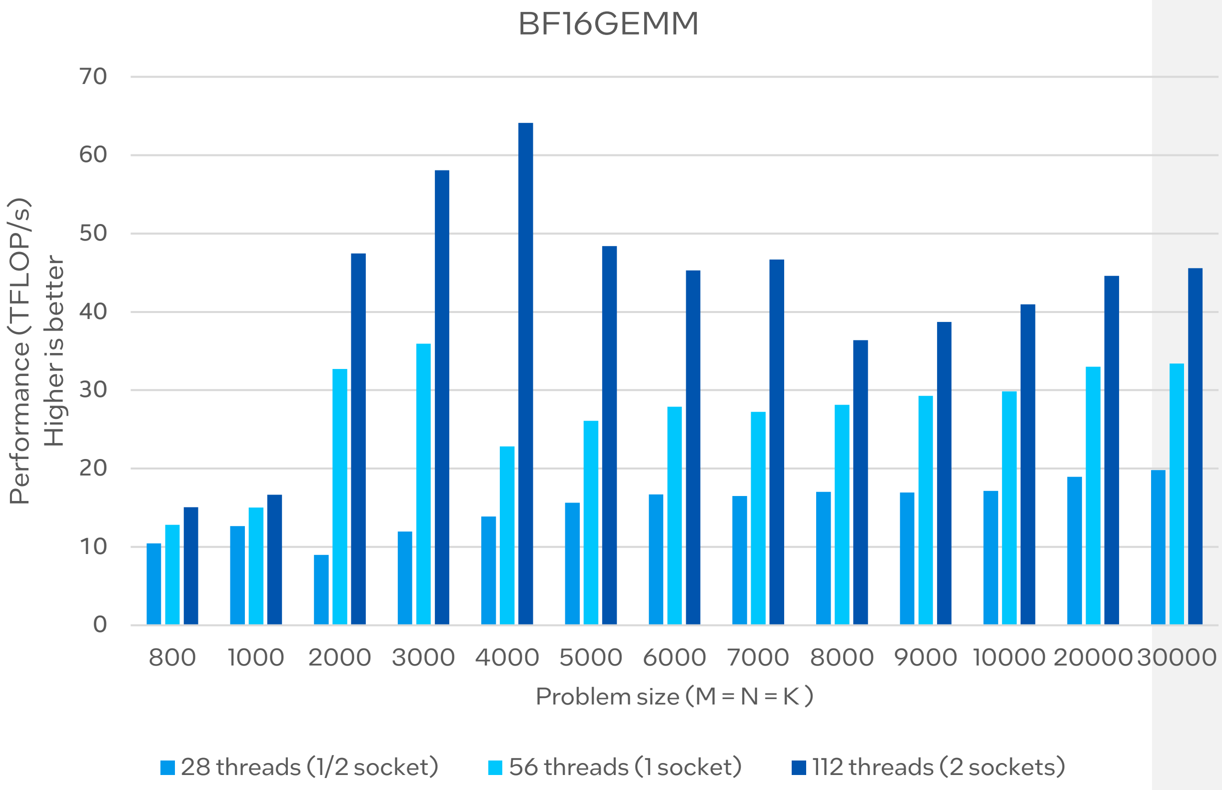
oneMKL General Matrix Multiplication (GEMM) Performance

Intel® oneAPI Math Kernel Library (oneMKL) 2023.0 GEMM on 4th Gen Intel® Xeon® Scalable Processor

oneMKL SGEMM shows up to 14.2 TFLOP/s performance on 2 sockets 4th Gen Intel® Xeon® Scalable Processor



oneMKL BF16GEMM shows up to 64.1 TFLOP/s performance on 2 sockets 4th Gen Intel® Xeon® Scalable Processor

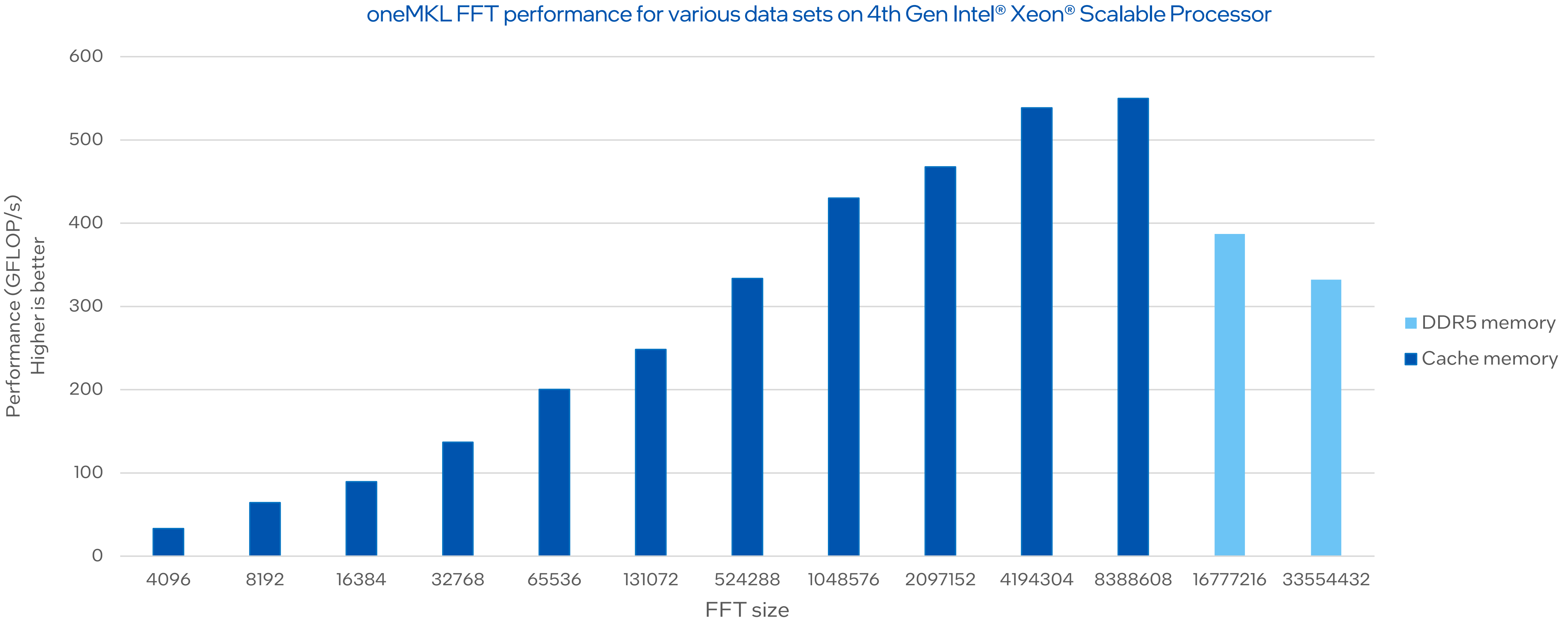


Testing Date: Performance results are **based on testing by Intel as of February 6, 2023** and may not reflect all publicly available security updates.

Configuration Details and Workload Setup: 1-node, 2x Intel® Xeon® Platinum 8480+ processor on Denali Pass platform with 1024 GB (16 slots/ 64GB/ 4800) total DDR5 memory, ucode 0x2b000161, HT off, Turbo on, Ubuntu 22.04.1 LTS, 5.17.0-051700-generic, 1x Intel® SSD 3.5TB OS Drive; Intel® oneAPI Math Kernel Library 2023.0 (oneMKL). SGEMM & BF16GEMM performance for square matrix dimensions between 800 and 30,000.

oneMKL Fast Fourier Transforms (FFT) Performance

Intel® oneAPI Math Kernel Library (oneMKL) 2023.0 FFT on 4th Gen Intel® Xeon® Scalable Processor



Testing Date: Performance results are based on testing by Intel as of February 6, 2023 and may not reflect all publicly available security updates.

Configuration Details and Workload Setup: 1-node, 2x Intel® Xeon® Platinum 8480+ processor on Denali Pass platform with 1024 GB (16 slots/ 64GB/ 4800) total DDR5 memory, ucode 0x2b000161, HT off, Turbo on, Ubuntu 22.04.1 LTS, 5.17.0-051700-generic, 1x Intel® SSD 3.5TB OS Drive; Intel® oneAPI Math Kernel Library 2023.0 (oneMKL). Performance measured for forward out-of-place 1 Dimensional FFT with double precision data sets. In general, size of input data and size of L1, L2 and L3 caches, affect performance of FFT algorithm.

Intel® oneAPI Math Kernel Library Resources

software.intel.com/oneAPI/mkl

Get Started

- software.intel.com/oneAPI/mkl
- oneMKL - Get Started Guide
- oneMKL code samples for DPC++ interface
- oneMKL how-to's
- Migrating the MonteCarloMultiGPU from CUDA* to SYCL*

OneMKL Developer References & Guides

Developer References:
C | Fortran | DPC++

Developer Guides:
Windows* | Linux* | macOS*

Learn

- Training: Webinars & courses
- oneMKL Essentials Training
- Base toolkit on Intel® DevCloud
- oneMKL documentation
- Intel® oneMKL Link Line Advisor

Ecosystem & Support

Rich active developer ecosystem eases adoption

- oneMKL Community Forum
- Intel® DevMesh Innovator oneMKL Projects
- oneMKL Academic Programs: oneAPI Centers of Excellence: research, enabling code, curriculum, teaching
- Online Service Center (paid support)

Intel® oneAPI Math Kernel Library (oneMKL) available on Intel® DevCloud

Implement and test your applications on Intel® DevCloud today

intel. DevCloud

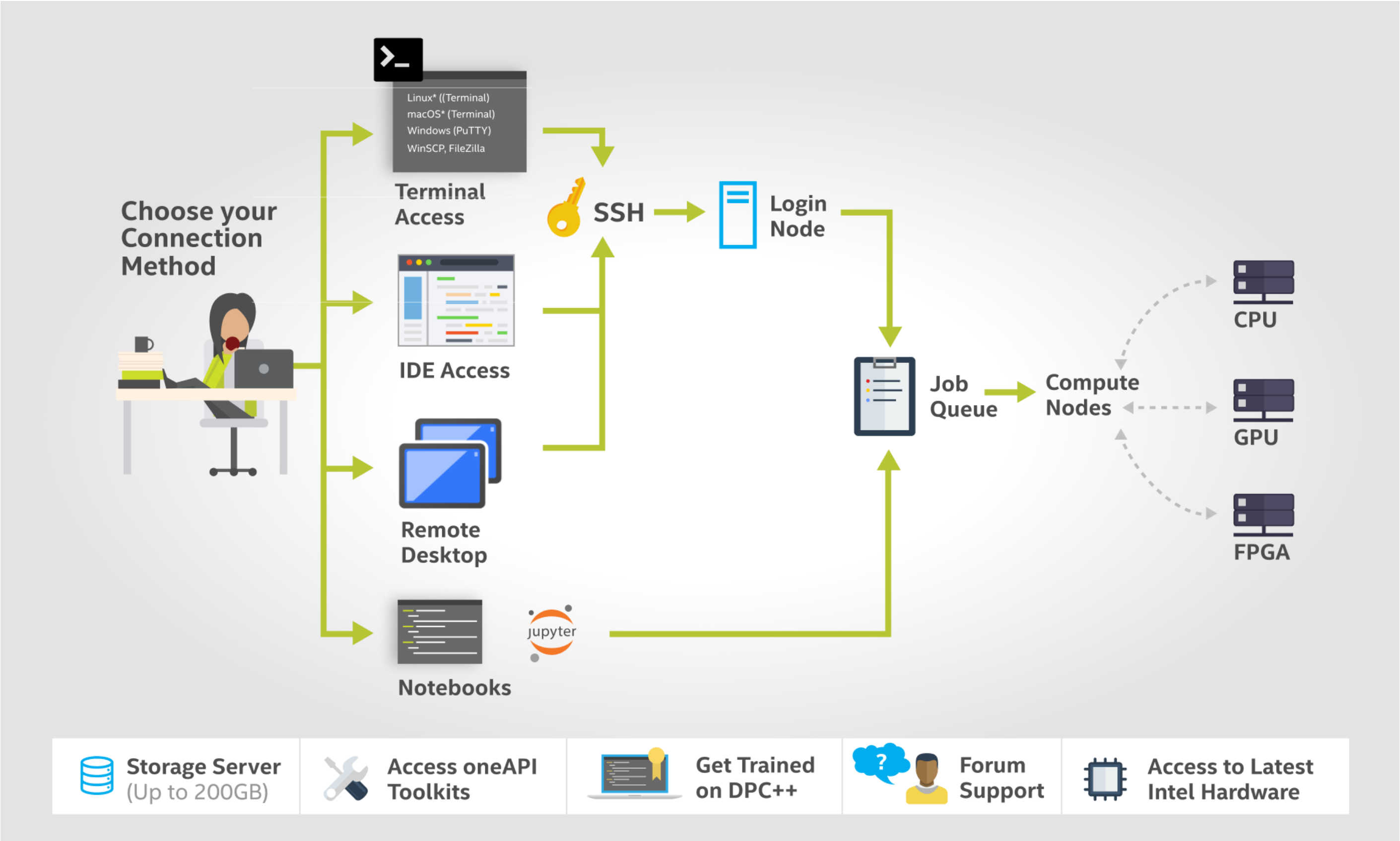
1 Minute to Code

No Hardware Acquisition

No Download, Install or Configuration

Easy Access to Samples & Tutorials

Support for Jupyter Notebooks, Visual Studio Code



software.intel.com/devcloud/oneapi

Q & A

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

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.

Your costs and results may vary.

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

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.