A New, Open-Standards-Based, Open-Source Programming Model for All Accelerators

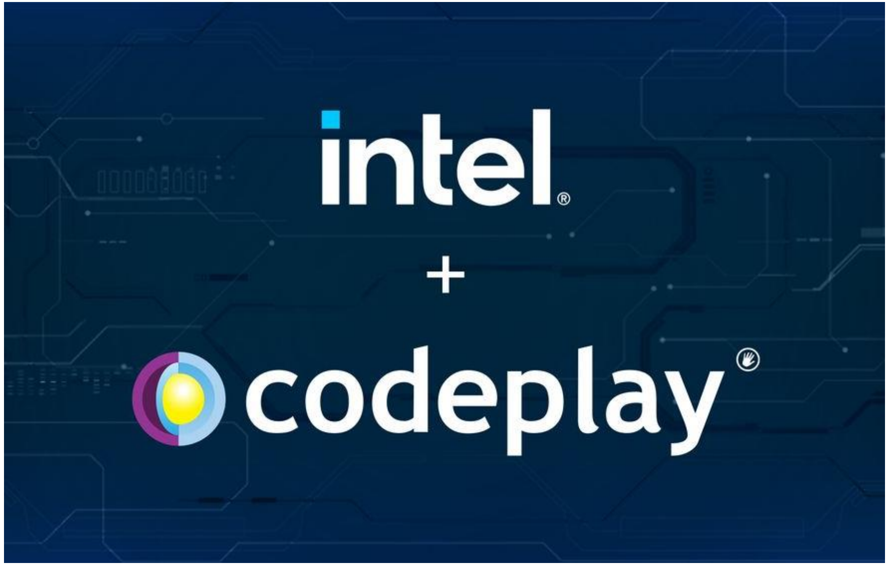
Charles Macfarlane

Chief Business Officer

Codeplay Software

Who Are We?

- Codeplay is a wholly owned subsidiary of Intel
- Focus on advancing and embracing SYCL and oneAPI



NVIDIA GPUs are Ubiquitous

- CUDA is proprietary
- Defined by NVIDIA for NVIDIA
- Locked to NVIDIA hardware
- Limited input into direction of CUDA
- Protected by NVIDIA legal terms

DISCRETE GPU MARKET SHARE (Q1 2022)

	Q1’21	Q4’21	Q1’22
AMD	19%	18%	17%
INTEL	n/a	5%	4%
NVIDIA	81%	78%	78%

https://wccftech.com/nvidia-amd-gain-gpu-market-share-while-overall-shipments-decrease-by-19-in-q1-2022/

The oneAPI and SYCL Approach

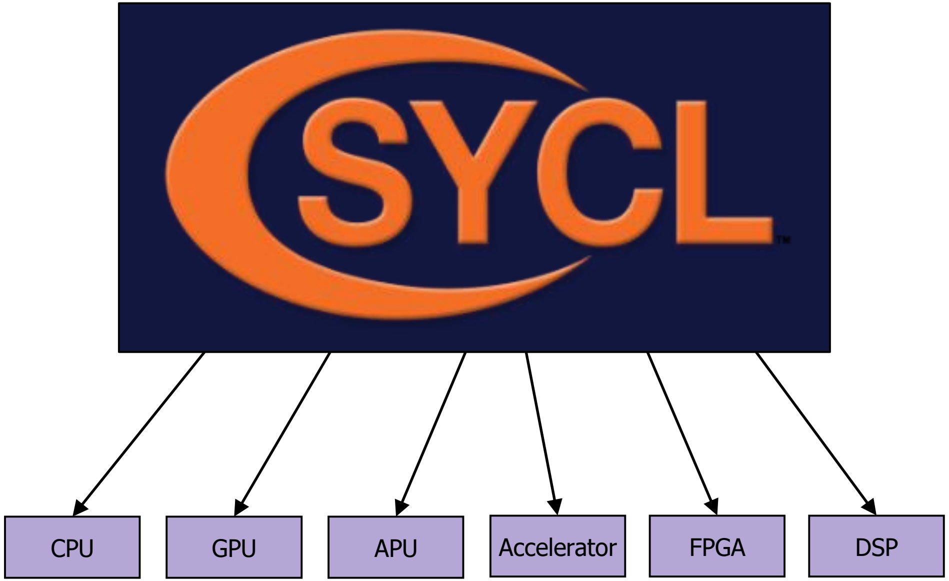
codeplay®

Open Standards Programming

- SYCL is an open, cross-platform standard programming model based on C++ 17 developed by The Khronos Group
- SYCL supports multiple types of hardware including GPUs, CPUs, and FPGAs from all major vendors
- SYCL is supported by multiple compilers

SYCL Is a Single-source, High-level, Standard C++ Programming Model

- SYCL can target any device supported by its backend
- SYCL can target a number of different backends

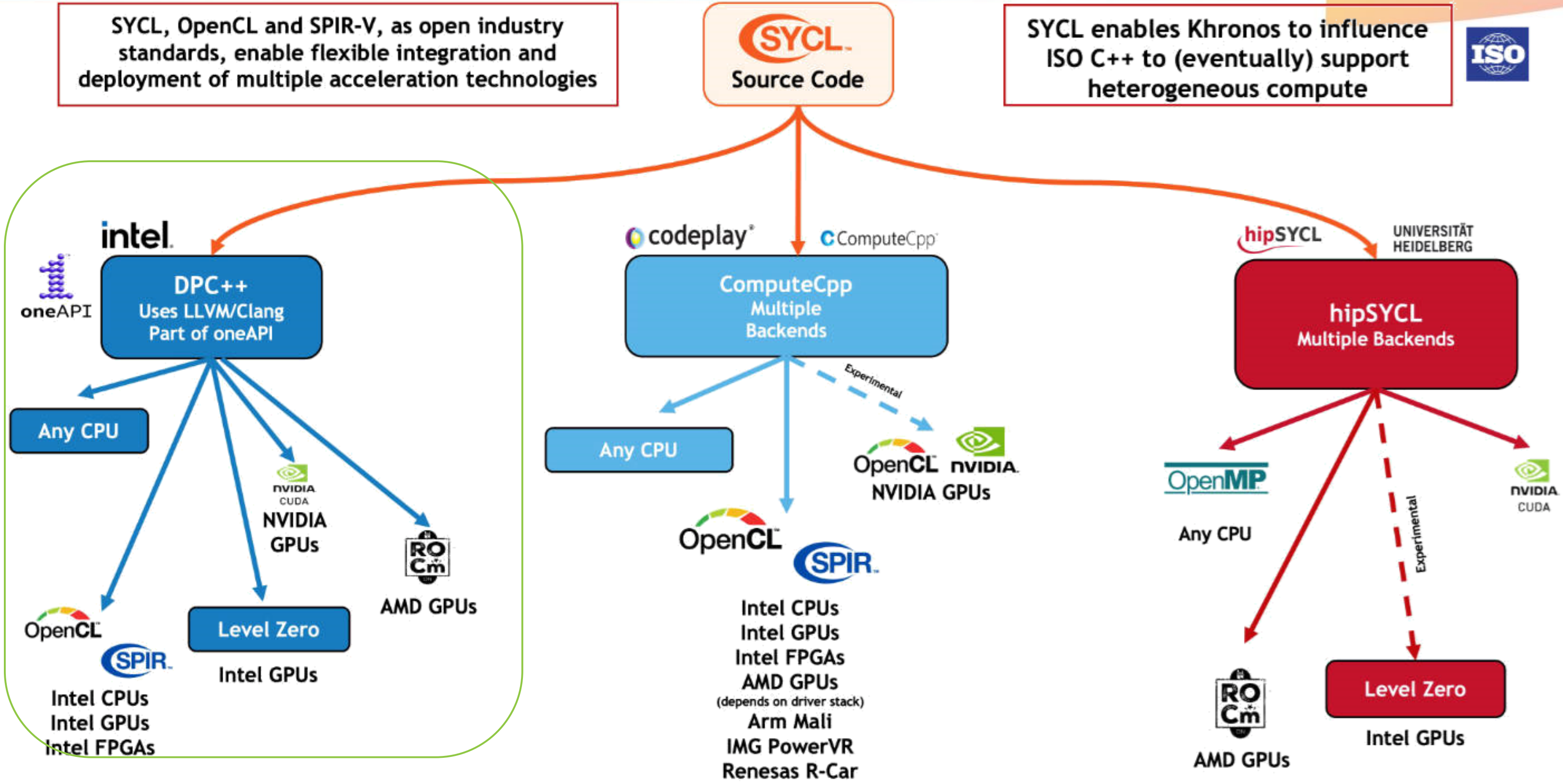


SYCL can target a range of heterogeneous platforms

SYCL Implementations Under Development

SYCL, OpenCL and SPIR-V, as open industry standards, enable flexible integration and deployment of multiple acceleration technologies

SYCL enables Khronos to influence ISO C++ to (eventually) support heterogeneous compute



SYCL Source Code

intel

oneAPI

DPC++
Uses LLVM/Clang
Part of oneAPI

Any CPU

OpenCL SPIR
Intel CPUs
Intel GPUs
Intel FPGAs

NVIDIA CUDA
NVIDIA GPUs

Level Zero
Intel GPUs

ROCm
AMD GPUs

codeplay ComputeCpp

ComputeCpp
Multiple
Backends

Any CPU

OpenCL SPIR
Intel CPUs
Intel GPUs
Intel FPGAs
AMD GPUs
(depends on driver stack)
Arm Mali
IMG PowerVR
Renesas R-Car

Experimental

OpenCL NVIDIA
NVIDIA GPUs

hipSYCL UNIVERSITÄT HEIDELBERG

hipSYCL
Multiple Backends

OpenMP
Any CPU

NVIDIA CUDA

Experimental

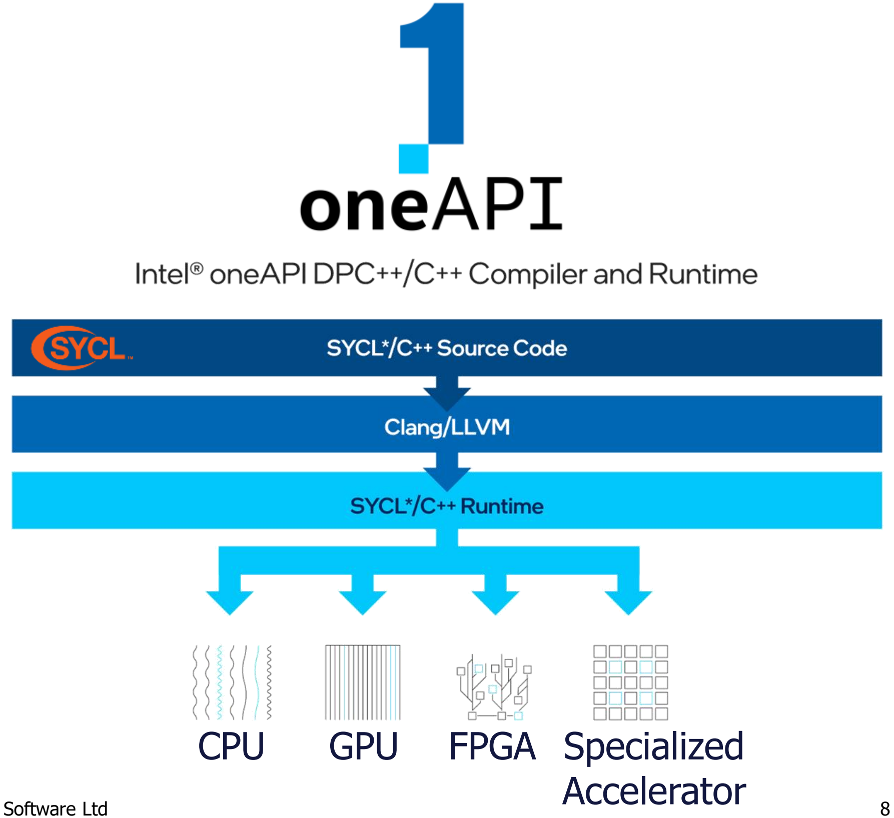
ROCm
AMD GPUs

Level Zero
Intel GPUs

Source: https://www.khronos.org/sycl/

oneAPI and SYCL

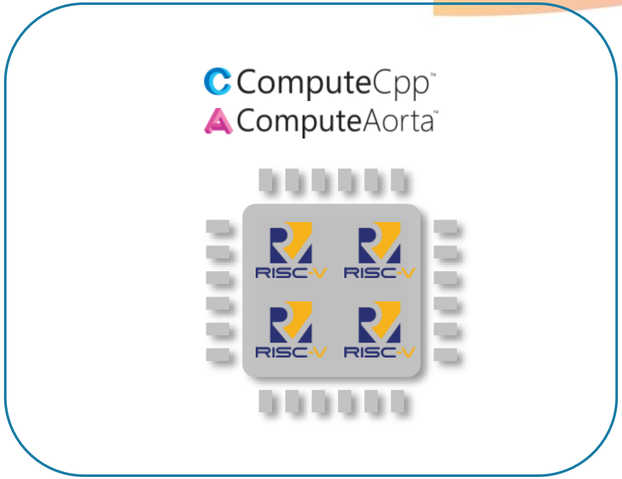
- SYCL sits at the heart of oneAPI
  - Compare with CUDA at the heart of NVIDIA software
- The DPC++ SYCL compiler is open source and based on the LLVM Compiler Infrastructure project



© 2023 Codeplay Software Ltd

SYCL for RISC-V

- Bring industry leading AI and HPC software to the growing range of RISC-V solutions
- Industry-standard compilers & libraries
- Open-source libs and frameworks supported
- Fast migration path of scientific and AI software from NVIDIA GPUs



“By applying Codeplay's ComputeAorta and ComputeCpp technology, we expect that we can bring state-of-art technology to RISC-V community with our research results.”
Hideki Sugimoto, CTO NSITEXE Inc, Oct 30th, 2020

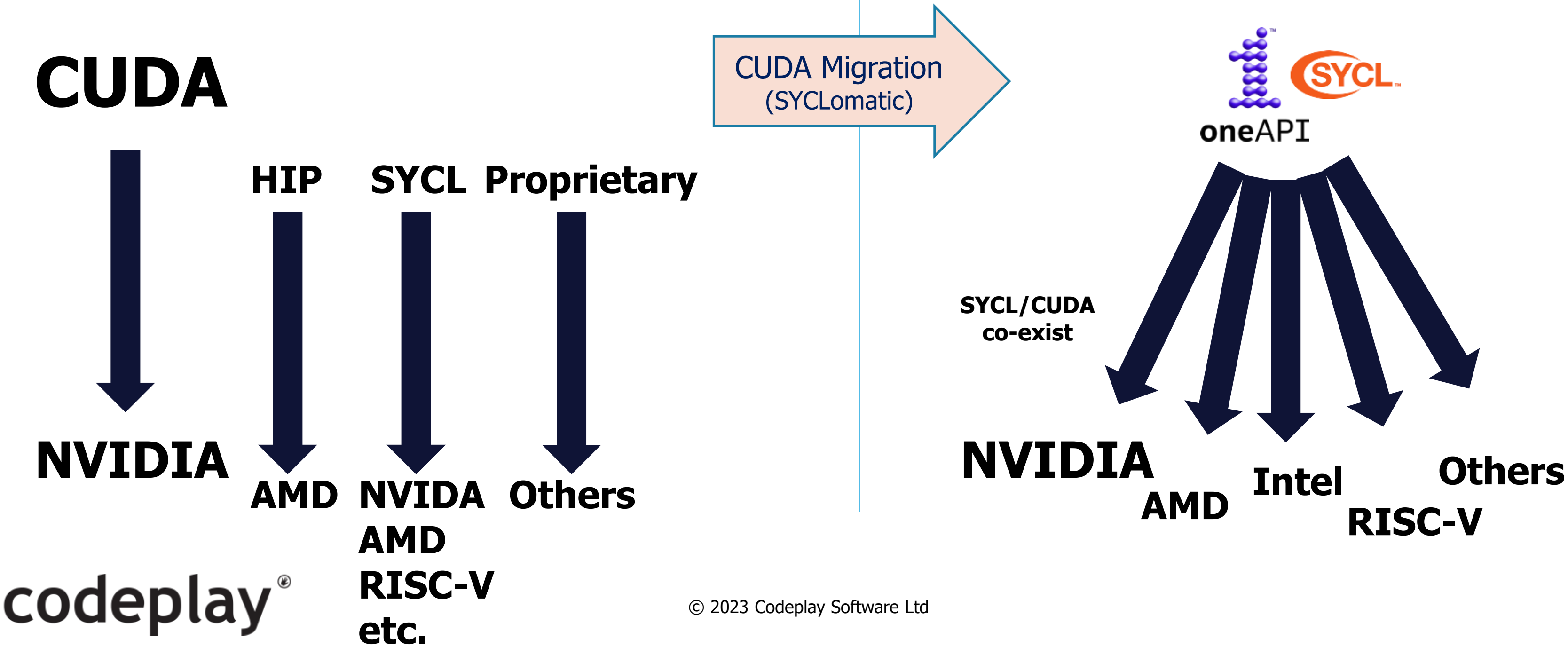
NSI-TEXE NEDO KMC Kyoto Microcomputer Co.,Ltd.

Migrating from CUDA to Open Standards

Achieving Multi-Platform Support

Today many programming platforms supported

Future only one software platform needed



CUDA Migration (SYCLomatic)

CUDA → NVIDIA

HIP → AMD

SYCL → NVIDA AMD RISC-V etc.

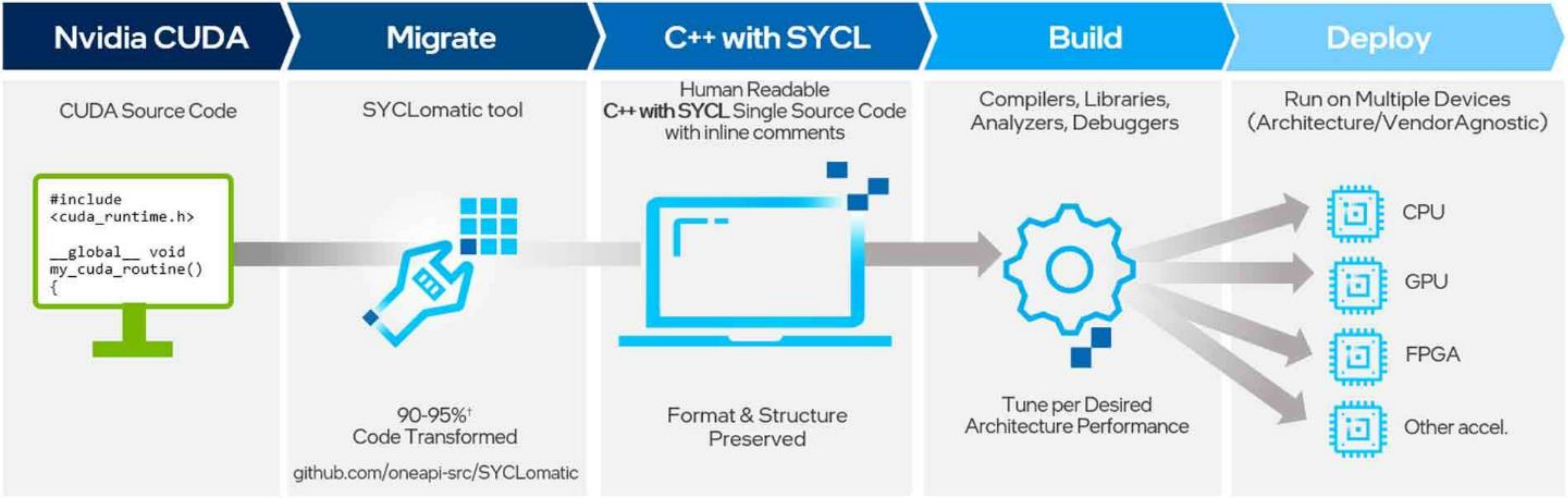
Proprietary → Others

oneAPI SYCL → NVIDIA, AMD, Intel, RISC-V, Others

SYCL/CUDA co-exist

CUDA to SYCL Code Migration Workflow

SYCLomatic / Intel® DPC++ Compatibility Tool assists the migration of code written in CUDA to SYCL once, generating **human readable** code wherever possible



Nvidia CUDA → **Migrate** → **C++ with SYCL** → **Build** → **Deploy**

CUDA Source Code

```
#include
<cuda_runtime.h>

__global__ void
my_cuda_routine()
{
```

SYCLomatic tool

90-95%[†]
Code Transformed

github.com/oneapi-src/SYCLomatic

Human Readable
C++ with SYCL Single Source Code
with inline comments

Format & Structure
Preserved

Compilers, Libraries,
Analyzers, Debuggers

Tune per Desired
Architecture Performance

Run on Multiple Devices
(Architecture/Vendor Agnostic)

[†] Intel estimates as of September 2021. Based on measurements on a set of 70 HPC benchmarks and samples, with examples like Rodinia, SHOC, PENNANT. Results may vary.

Migrate and Deploy Code in 5 Easy Steps

Migration Approaches

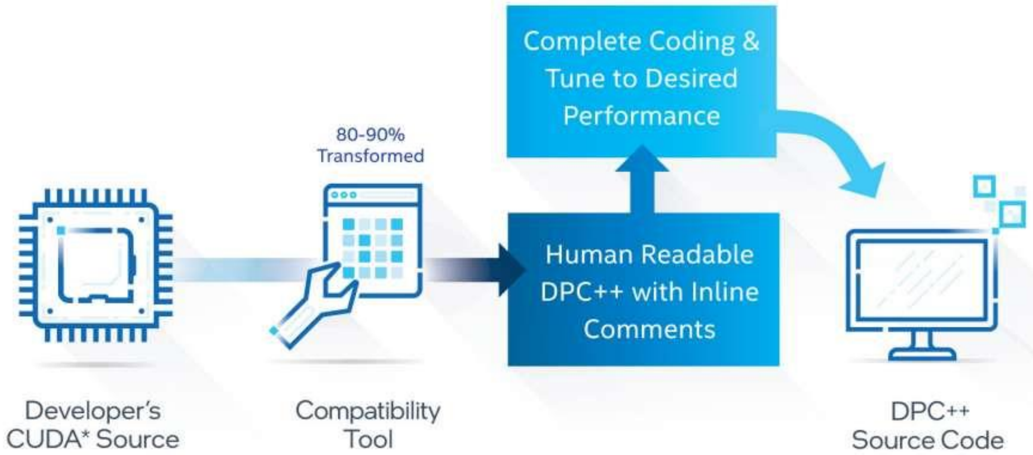
Semi-Automatic	Incremental Porting
Use conversion tools	Port your kernels alongside existing CUDA code
Some engineering work to complete migration	Run CUDA and SYCL code together

Semi-Automatic

- **DPCT**
  - Intel released tool
- **SYCLomatic**
  - Open source

- Migrates CUDA code to SYCL
- ~90% of code is migrated

Intel® DPC++ Compatibility Tool Usage Flow



Developer's CUDA* Source

80-90% Transformed

Compatibility Tool

Human Readable DPC++ with Inline Comments

Complete Coding & Tune to Desired Performance

DPC++ Source Code

Incremental Porting

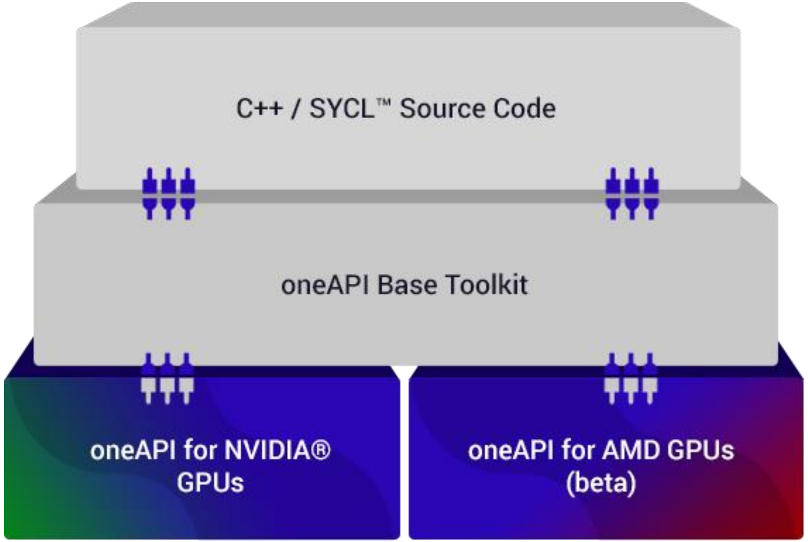
- Migrating large codebases is a major effort
- It is possible to incrementally migrate CUDA kernels to SYCL
- Run SYCL and CUDA co-existing in same application on NVIDIA GPU

(CUDA + SYCL) ➔ NVIDIA GPU

Evaluate and transition application code to SYCL and oneAPI

oneAPI for NVIDIA GPUs and AMD GPUs

- Codeplay contributes plugins
  - Application developers can continue to execute SYCL and oneAPI software on NVIDIA and AMD GPUs
  - Adds support for NVIDIA and AMD GPUs to the oneAPI Base Toolkit

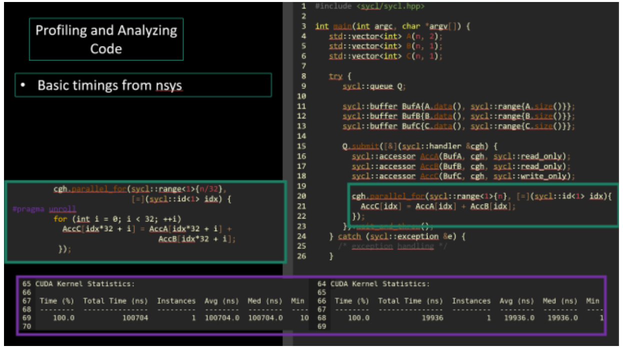


Uses existing NVIDIA and AMD tools and libraries

Download from developer.codeplay.com

Use Familiar NVIDIA GPU Tools

- Developers can profile code on NVIDIA GPUs with nsys and ncu
- Developers can debug on NVIDIA GPUs with CUDA-gdb
- All of these tools are used with oneAPI in the same way as an application written in CUDA



Profiling and Analyzing Code

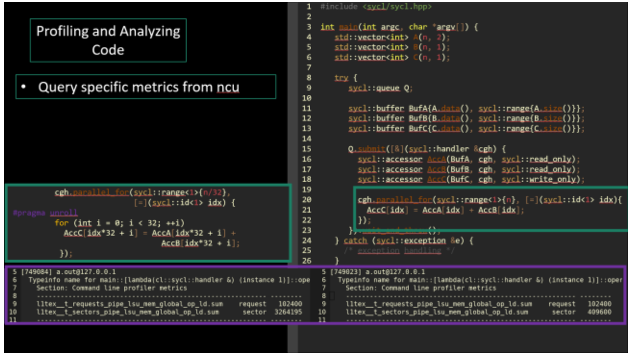
- Basic timings from nsys

```
        cgh.parallel_for(sycl::range<1>{n/32},
                         [=](sycl::id<1> idx) {
#pragma unroll
        for (int i = 0; i < 32; ++i)
          AccC[idx*32 + i] = AccA[idx*32 + i] +
                             AccB[idx*32 + i];
         });
```

```
#include <sycl/sycl.hpp>

int main(int argc, char *argv[]) {
  std::vector<int> A(n, 2);
  std::vector<int> B(n, 1);
  std::vector<int> C(n, 1);

  try {
    sycl::queue Q;

    sycl::buffer BufA{A.data(), sycl::range{A.size()}};
    sycl::buffer BufB{B.data(), sycl::range{B.size()}};
    sycl::buffer BufC{C.data(), sycl::range{C.size()}};

    Q.submit([&](sycl::handler &cgh) {
      sycl::accessor AccA(BufA, cgh, sycl::read_only);
      sycl::accessor AccB(BufB, cgh, sycl::read_only);
      sycl::accessor AccC(BufC, cgh, sycl::write_only);

      cgh.parallel_for(sycl::range<1>{n}, [=](sycl::id<1> idx){
        AccC[idx] = AccA[idx] + AccB[idx];
      });
    }).wait_and_throw();
  } catch (sycl::exception &e) {
    /* exception handling */
  }
```

```
CUDA Kernel Statistics:

 Time (%)  Total Time (ns)  Instances  Avg (ns)  Med (ns)  Min
 --------  ---------------  ---------  --------  --------  ----
    100.0           100704          1  100704.0  100704.0    10
```

```
CUDA Kernel Statistics:

 Time (%)  Total Time (ns)  Instances  Avg (ns)  Med (ns)  Min
 --------  ---------------  ---------  --------  --------  ----
    100.0            19936          1   19936.0   19936.0     1
```



Profiling and Analyzing Code

- Query specific metrics from ncu

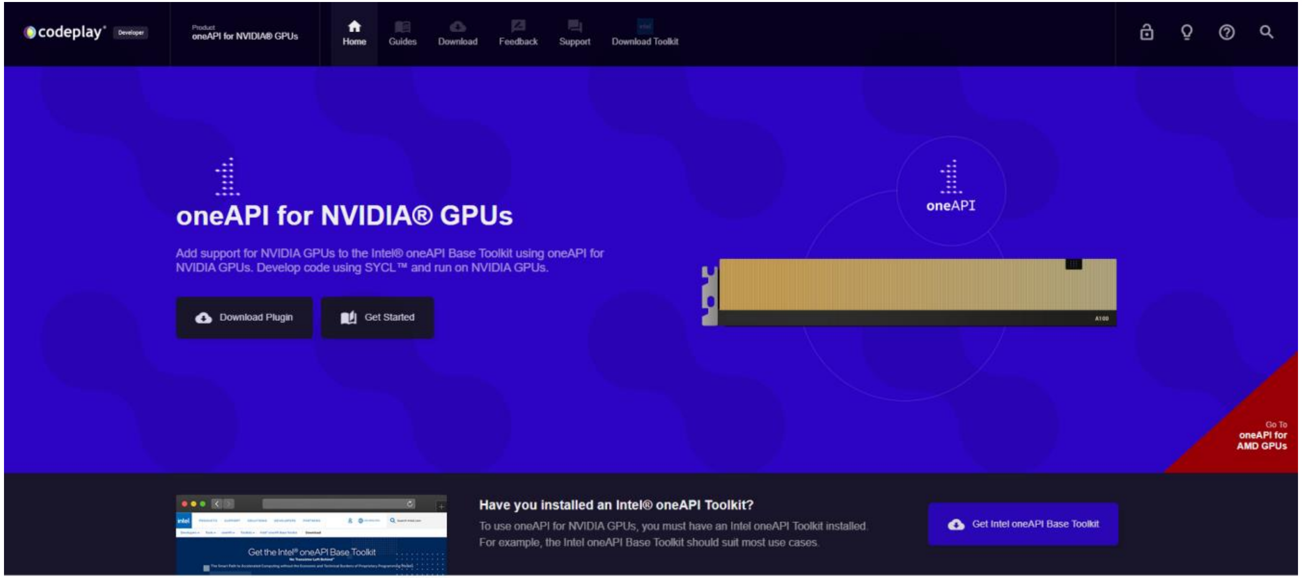
```
        cgh.parallel_for(sycl::range<1>{n/32},
                         [=](sycl::id<1> idx) {
#pragma unroll
        for (int i = 0; i < 32; ++i)
          AccC[idx*32 + i] = AccA[idx*32 + i] +
                             AccB[idx*32 + i];
         });
```

```
#include <sycl/sycl.hpp>

int main(int argc, char *argv[]) {
  std::vector<int> A(n, 2);
  std::vector<int> B(n, 1);
  std::vector<int> C(n, 1);

  try {
    sycl::queue Q;

    sycl::buffer BufA{A.data(), sycl::range{A.size()}};
    sycl::buffer BufB{B.data(), sycl::range{B.size()}};
    sycl::buffer BufC{C.data(), sycl::range{C.size()}};

    Q.submit([&](sycl::handler &cgh) {
      sycl::accessor AccA(BufA, cgh, sycl::read_only);
      sycl::accessor AccB(BufB, cgh, sycl::read_only);
      sycl::accessor AccC(BufC, cgh, sycl::write_only);

      cgh.parallel_for(sycl::range<1>{n}, [=](sycl::id<1> idx){
        AccC[idx] = AccA[idx] + AccB[idx];
      });
    }).wait_and_throw();
  } catch (sycl::exception &e) {
    /* exception handling */
  }
```

```
[749084] a.out@127.0.0.1
  Typeinfo name for main::[lambda(cl::sycl::handler &) (instance 1)]::ope
    Section: Command line profiler metrics
    ---------------------------------------------------------- --------
    l1tex__t_requests_pipe_lsu_mem_global_op_ld.sum    request   102400
    l1tex__t_sectors_pipe_lsu_mem_global_op_ld.sum      sector  3264195
    ---------------------------------------------------------- --------
```

```
[749823] a.out@127.0.0.1
  Typeinfo name for main::[lambda(cl::sycl::handler &) (instance 1)]::oper
    Section: Command line profiler metrics
    ---------------------------------------------------------- --------
    l1tex__t_requests_pipe_lsu_mem_global_op_ld.sum    request   102400
    l1tex__t_sectors_pipe_lsu_mem_global_op_ld.sum      sector   409600
    ---------------------------------------------------------- --------
```

How to Get the Plugins



oneAPI for NVIDIA® GPUs

Add support for NVIDIA GPUs to the Intel® oneAPI Base Toolkit using oneAPI for NVIDIA GPUs. Develop code using SYCL™ and run on NVIDIA GPUs.

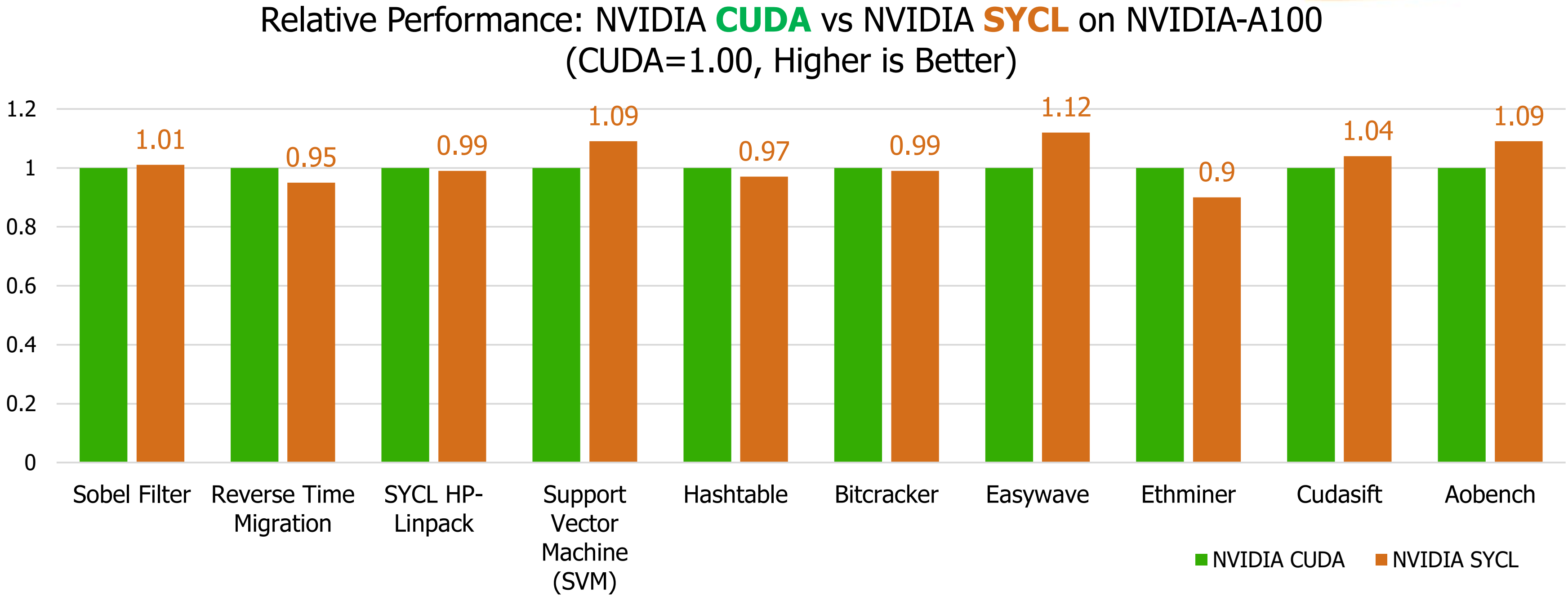
Have you installed an Intel® oneAPI Toolkit?

To use oneAPI for NVIDIA GPUs, you must have an Intel oneAPI Toolkit installed. For example, the Intel oneAPI Base Toolkit should suit most use cases.

Download for free from **developer.codeplay.com**

Performance

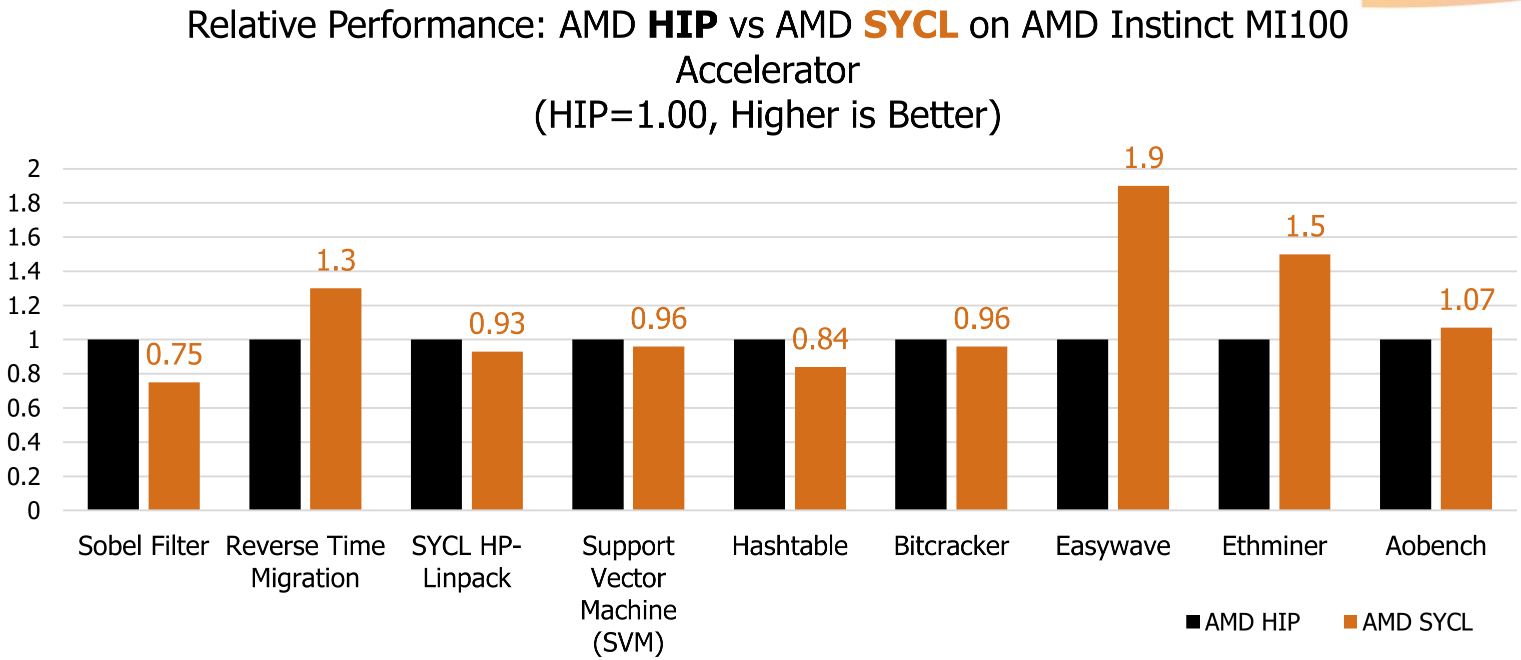
Relative Performance
Nvidia SYCL vs Nvidia CUDA on Nvidia GPU



Configuration Details and Workload Setup: Intel® Xeon® Platinum 8360Y CPU@2.4GHz, 2 socket, yper Thread On, Turbo On, 256GB Hynix DDR4-3200, ucode 0x000363, GPU NVIDIA A100 PCIe 80GB GPU memory. Software SYCL open source/CLANG 15.0.0, CUDA SDK 11.7 with NVIDIA-NVCC 11.7.64, cuMath 11.7, cuDNN 11.7, Ubuntu 22.04.1, SYCL open source/CLANG compiler switches -fsycl-targets=nvptx64-nvidia-cuda, NVIDIA NVCC compiler switches: -O3 -gencode=arch=compute_80,code=sm_80. Represented workloads with Intel optimizations
Performance results are based on testing as of dates shown in configurations and may not reflect

Performance results are based on testing by Intel as of August 15th, 2022 and may not reflect all publicly available updates. See configuration disclosure for details. No product or component can be absolutely secure. Performance varies by use, configuration and other factors. Learn more at www.intel.com/performanceindex. Your costs and results may vary.

Relative Performance
AMD SYCL vs AMD HIP on AMD GPU



Relative Performance: AMD **HIP** vs AMD **SYCL** on AMD Instinct MI100 Accelerator
(HIP=1.00, Higher is Better)

Configuration Details and Workload Setup: Intel® Xeon® Gold 6330 CPU @2.0GHz, 2 socket, Hyper Thread Off, Turbo On, 256GB Hynix DDR4-3200, ucode 0xd000363, GPU: AMD Instinct MI100, 32GB GPU memory. Software SYCL open source/CLANG 15.0.0, AMD RoCm 5.2.1 with AMD-HIPCC 5.2.21152-4b155a06, hipSolver 5.2.1, rocBLAS 5.2.1, Ubuntu 20.04.4. SYCL open source/CLANG compiler switches -fsycl-targets=amdgcn-amd-amdhsa –Xsycl-target-backend –offload-arch-gfx908, AMD-HIPCC compiler switches –03. Represented workloads with Intel optimizations

Performance results are based on testing by Intel as of August 15th, 2022 and may not reflect all publicly available updates. See configuration disclosure for details. No product or component can be absolutely secure. Performance varies by use, configuration and other factors. Learn more at www.intel.com/performanceindex. Your costs and results may vary.

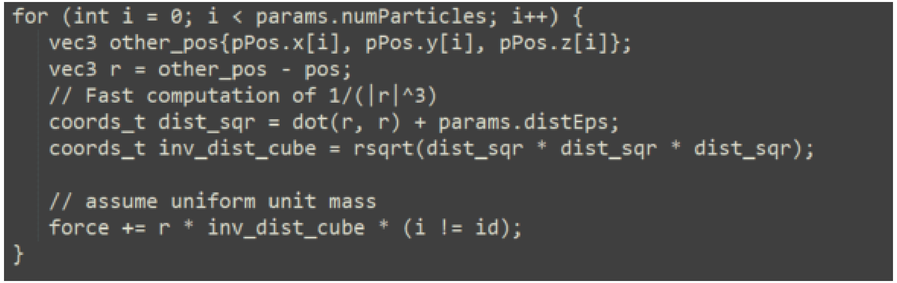
Example Conversion

Example Conversion : N-Body

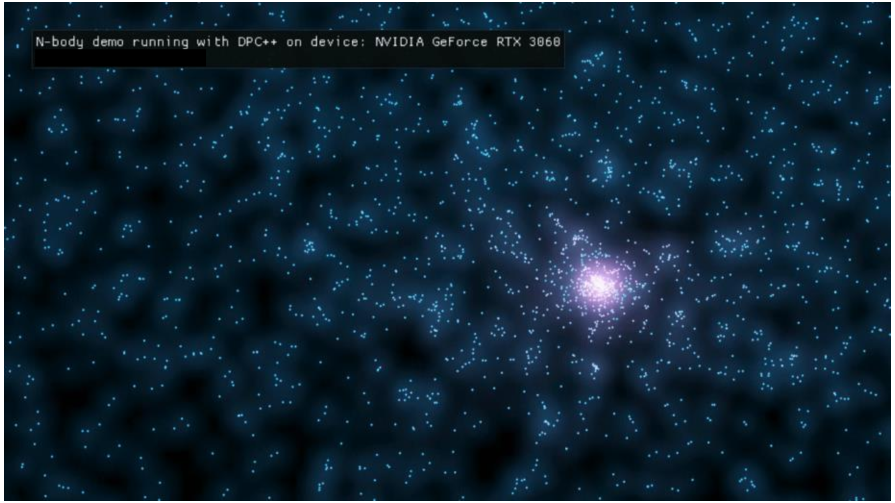
- Simulates gravitational interaction in a fictional galaxy

$$\vec{F}_i = -\sum_{i \neq j} G \frac{(\vec{r}_i - \vec{r}_j)}{|\vec{r}_i - \vec{r}_j|^3}$$

- Intentionally simple kernel
- OpenGL for graphics



```
for (int i = 0; i < params.numParticles; i++) {
    vec3 other_pos{pPos.x[i], pPos.y[i], pPos.z[i]};
    vec3 r = other_pos - pos;
    // Fast computation of 1/(|r|^3)
    coords_t dist_sqr = dot(r, r) + params.distEps;
    coords_t inv_dist_cube = rsqrt(dist_sqr * dist_sqr * dist_sqr);

    // assume uniform unit mass
    force += r * inv_dist_cube * (i != id);
}
```



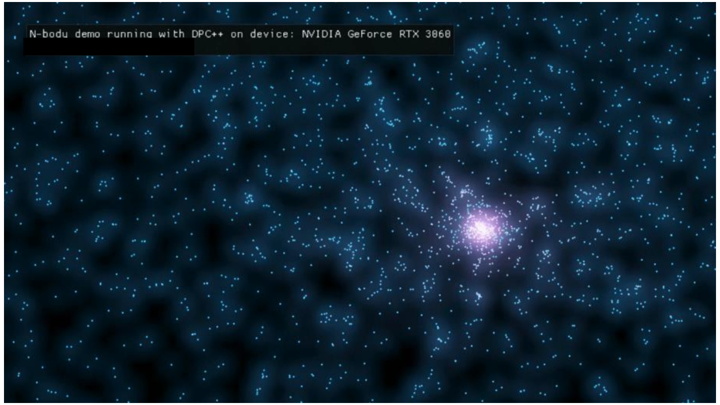
N-body demo running with DPC++ on device: NVIDIA GeForce RTX 3060

Try It Out for Yourself

- https://github.com/codeplaysoftware/cuda-to-sycl-nbody

- Run it on your own hardware
- Raise issues
- Contribute

- Visit the demo at the Codeplay booth



```
N-body demo running with DPC++ on device: NVIDIA GeForce RTX 3060
```

oneAPI Community Forum

codeplay®

What is the oneAPI Community Forum?

1

A cross industry group of hardware and software experts

2

Defines standard interfaces for accelerator computing

3

Multiple specialist technical working groups

4

Drives the future of open-standard accelerator computing

Benefits

For Software Developers

- Develop with open standards for accelerator computing
- Single code base for multiple processors targets
- Standards and industry defined libraries
- Future proof your software

For Processor Developers

- Adopt an open standard with existing open-source implementations
- Enable an existing ecosystem of software and educational resources
- Leverage an existing tested and optimized toolchain

Free and based on open standards

Accrad
Accelerated Radiology

AIBLE

AI SINGAPORE

allegro.ai

ANACONDA

Argonne NATIONAL LABORATORY

@rchanan

AsiaInfo 亚信科技

AUTODESK® ARNOLD

Bai百度 飞桨 PaddlePaddle

BENTLEY

ATLAS EXPERIMENT

Atos

Alibaba Cloud

Ansys

BioDataAnalysis GmbH
Next Generation Bio-Medical Image Analysis

GeoEast

Berkeley UNIVERSITY OF CALIFORNIA

BittWare a molex company

Brightskies technologies

中国石油集团东方地球物理勘探有限责任公司
BGP INC.,CHINA NATIONAL PETROLEUM CORPORATION

CHAOSGROUP

codeplay®

CONVERGE CFD SOFTWARE

CANONICAL

CERN openlab

CERTI FACE

FOUNDRY. MODO

CGG

FACULTY OF MATHEMATICS AND PHYSICS Charles University

C-DAC

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

大势智慧 DASPATIAL

DataRobot

DELL Technologies

Durham University

EUREKOM

OLIGO ARCHIVE

Soda
Inria: National Institute for Research in Digital Science & Technology

GE

GIGASPACES

CINECA

CINESITE

HCL

Google Cloud

E4 COMPUTER ENGINEERING

inesc id lisboa

TÉCNICO LISBOA

KFBIO

kt

Laboratório Nacional de Computação Científica

Lenovo

ILLUMINATION MACGUFF

lrz

Hewlett Packard Enterprise

IISER PUNE

KATANA GRAPH

URZ HEIDELBERG UNIVERSITY COMPUTING CENTRE

ILLINOIS

NAVER CLOVA

NIU Northern Illinois University

UT-BATTELLE

LAIKA

PHILIPS

Peraton Labs

LOBACHEVSKY UNIVERSITY

MAXON

MAX PLANCK COMPUTING & DATA FACILITY

MPCDF

Microsoft Azure

MEGWARE

mercenaries engineering Guerilla

SDSC SAN DIEGO SUPERCOMPUTER CENTER

UKRI Science and Technology Facilities Council
Scientific Computing

Oak Ridge National Laboratory

OLD DOMINION UNIVERSITY

QZ quantumz.io

rENIAC

SAP

SAMSUNG MEDISON

MEGH COMPUTING

SiPearl

SUSE

TANGENTSTUDIOS

Taboola

Red Hat OpenShift Data Science

TECHNION Israel Institute of Technology

SANKHYA SUTRA LABS

sas

twitter

SENAI CIMATEC FIEB
Federação das Indústrias do Estado da Bahia

Ospirent

TACC

北京大学软件与微电子学院
School of Software & Microelectronics

Tencent腾讯

TensorFlow

UNITED IMAGING 联影

ZIB

Indian Institute of Technology Delhi

Tech Mahindra

TECHNISCHE UNIVERSITÄT DARMSTADT

UCDAVIS UNIVERSITY OF CALIFORNIA

UNIVERSITY OF CAMBRIDGE

ctg

UNIVERSITY OF OREGON

UNIVERSIDAD DE MÁLAGA

Ben-Gurion University of the Negev

INDIAN INSTITUTE OF SCIENCE

Stockholm University

KTH VETENSKAP OCH KONST

UNIVERSIDAD COMPLUTENSE MADRID

THE UNIVERSITY OF TENNESSEE KNOXVILLE

WeBank

University of Bristol

University College London

University of Stuttgart Germany

THE UNIVERSITY OF UTAH

Verizon

YUAN

These organizations support the oneAPI initiative for a single, unified programming model for cross-architecture development.
It does not indicate any agreement to purchase or use of Intel's products. *Other names and brands may be claimed as the property of others.

Conclusions

- NVIDIA with CUDA is dominant and starting place for most AI applications, but locks into one supplier
- SYCL is the best alternative and provides platform independence for heterogeneous processor programming
- oneAPI, based on SYCL, will provide the ecosystem and tools needed
- Start now with oneAPI
  - Experimenting with existing solutions and evolving your own
  - Join oneAPI Community Forum

Other Performance Research

- Excellent published papers and presentations
  - "State of SYCL – ECP BOF Showcases Progress and Performance"
    by John Russell, February 28, 2023
    - https://www.hpcwire.com/2023/02/28/state-of-sycl-ecp-bof-showcases-progress-and-performance/
  - "SYCL's impact on algorithms, data structures and implementations"
    by Tom Deakin and Tobias Weinzierl, February 27, 2023
    - https://tobiasweinzierl.webspace.durham.ac.uk/research/workshops/siam-cse-23-sycl/ (SeisSol project)
  - "Evaluation of Intel's DPC++ Compatibility Tool in heterogeneous computing"
    by Germán Castaño a, Youssef Faqir-Rhazoui a, Carlos García a b, Manuel Prieto-Matías
    July, 2022
    - https://www.sciencedirect.com/science/article/pii/S0743731522000727?via%3Dihub
- Intel's list of CUDA to SYCL resources
    - https://www.intel.com/content/www/us/en/developer/tools/oneapi/training/migrate-from-cuda-to-cpp-with-sycl.html

Codeplay Software

Company

Leaders in enabling high-performance software solutions for new AI processing systems

Enabling the toughest processors with tools and middleware based on open standards

Established 2002 in Scotland, acquired by Intel in 2022 and now ~90 employees.

Collaborations

Synopsys

Broadcom

CEVA

Imagination

Renesas

KMC Kyoto Microcomputer Co., Ltd.

NSI-TEXE

Berkeley Lab

Oak Ridge National Laboratory

Argonne National Laboratory

And many more!

codeplay

Enabling AI & HPC to be Open, Safe & Accessible to All

Supported Solutions

oneAPI

An open, cross-industry, SYCL based, unified, multiarchitecture, multi-vendor programming model that delivers a common developer experience across accelerator architectures

Markets

High Performance Compute (HPC)
Automotive ADAS, IoT, Cloud Compute
Smartphones & Tablets
Medical & Industrial

Technologies: Artificial Intelligence
Vision Processing
Machine Learning
Big Data Compute

codeplay®

Notices & Disclaimers

Performance varies by use, configuration and other factors.

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.