SYCL Webinar | Fall 2021

December 7, 2021 / VIRTUAL
Khronos SYCL: Open Standard modern Heterogeneous C++

Michael Wong
Distinguished Engineer
SYCL WG Chair
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 with ~80 employees

Products

Acoran
Integrates all the industry standard technologies needed to support a very wide range of AI and HPC

ComputeAorta
The heart of Codeplay's compute technology enabling OpenCL™, SPIR-V™, HSA™ and Vulkan™

ComputeCpp
C++ platform via the SYCL™ open standard, enabling vision & machine learning e.g. TensorFlow™

Partners

Markets

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

Technologies: Artificial Intelligence
Vision Processing
Machine Learning
Big Data Compute
Distinguished Engineer

- Chair of SYCL Heterogeneous Programming Language
- ISO C++ Directions Group past Chair
- Past CEO OpenMP
- ISOCPP.org Director, VP
- Head of Delegation for C++ Standard for Canada
- Chair of Programming Languages for Standards Council of Canada
- Chair of WG21 SG19 Machine Learning
- Chair of WG21 SG14 Games Dev/Low Latency/Financial Trading/Embedded
- Editor: C++ SG5 Transactional Memory Technical Specification
- Editor: C++ SG1 Concurrency Technical Specification
- MISRA C++ and AUTOSAR
- Chair of Standards Council Canada TC22/SC32 Electrical and electronic components (SOTIF)
- Chair of UL4600 Object Tracking
- http://wongmichael.com/about
- C++11 book in Chinese:
  https://www.amazon.cn/dp/B00ETOV2OQ

Michael Wong

Argonne and Oak Ridge National Laboratories Award
Codeplay® Software to Further Strengthen SYCL™
Support Extending the Open Standard Software for
AMD GPUs
17 June 2021

NSITEXE, Kyoto Microcomputer and Codeplay
Software are bringing open standards programming to
RISC-V Vector processor for HPC and AI systems
29 October 2020

We build GPU compilers for some of the most powerful
supercomputers in the world
Acknowledgement and Disclaimer

Numerous people internal and external to the original C++/Khronos group/OpenMP, in industry and academia, have made contributions, influenced ideas, written part of this presentations, and offered feedbacks to form part of this talk. These include Bjarne Stroustrup, Joe Hummel, Botond Ballo, Simon McIntosh-Smith, Rod Burns, Ronan Keryell, Mike Kinsner, James Brodman, Colin Davidson, Fraser Cormack, Mehdi Goli, Aidan Dodds, SYCL WG, ISO C++, OpenMP, Codeplay Research as well as many others.

But I claim all credit for errors, and stupid mistakes. These are mine, all mine! You can’t have them.
THIS WORK REPRESENTS THE VIEW OF THE AUTHOR AND DOES NOT NECESSARILY REPRESENT THE VIEW OF CODEPLAY.

OTHER COMPANY, PRODUCT, AND SERVICE NAMES MAY BE TRADEMARKS OR SERVICE MARKS OF OTHERS.
SYCL Single Source C++ Parallel Programming

- C++ Libraries
- Standard C++ Application Code
- ML Frameworks
- C++ Template Libraries
- SYCL Compiler
- CPU Compiler
- CPU

- Accelerated code passed into device OpenCL compilers
- Complex ML frameworks can be directly compiled and accelerated
- SYCL is ideal for accelerating larger C++-based engines and applications with performance portability

- C++ Kernel Fusion can give better performance on complex apps and libraries than hand-coding
- C++ templates and lambda functions separate host & accelerated device code

- One-MKL
- One-DNN
- SYCL-Eigen
- SYCL-DNN
- SYCL Parallel STL

- GPU
- FPGA
- DSP
- Custom Hardware

- CPU
- GPU
- FPGA
- DSP
- AI/Tensor HW
- Custom Hardware

- Other Backends
SYCL 2020 is here!
Open Standard for Single Source C++ Parallel Heterogeneous Programming

SYCL 2020 is released after 3 years of intense work
Significant adoption in Embedded, Desktop and HPC markets
Improved programmability, smaller code size, faster performance
Based on C++17, backwards compatible with SYCL 1.2.1
Simplify porting of standard C++ applications to SYCL
Closer alignment and integration with ISO C++
Multiple Backend acceleration and API independent

SYCL 2020 increases expressiveness and simplicity for modern C++ heterogeneous programming
SYCL 2020 Industry Momentum

SYCL support growing from Embedded Systems through Desktops to Supercomputers
SYCL 2020 Major Features

• Unified Shared Memory (USM)
  • Code with pointers can work naturally without buffers or accessors
  • Simplifies porting from most code (e.g. CUDA, C++)

• Parallel Reductions
  • Added built-in reduction operation to avoid boilerplate code and achieve maximum performance on hardware with built-in reduction operation acceleration.

• Work group and subgroup algorithms
  • Efficient parallel operations between work items

• Class template argument deduction (CTAD) and template deduction guides
  • Simplified class template instantiation

• Simplified use of Accessors with a built-in reduction operation
  • Reduces boilerplate code and streamlines the use of C++ software design patterns
Parallel Industry Initiatives

- **OpenCL 1.2**: C++11 Single source programming
- **OpenCL 2.1**: SPIR-V in Core
- **OpenCL 2.2**: Many backend options
- **OpenCL 3.0**: Many backend options
- **SYCL 1.2**: C++11 Single source programming
- **SYCL 1.2.1**: C++11 Single source programming
- **SYCL 2020**: C++17 Single source programming
- **SYCL 202X**: C++20 Single source programming

- **C++11**
- **C++14**
- **C++17**
- **C++20**
- **C++23**
SYCL Implementations in 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.

**Source Code**

- **DPC++**
  - Uses LLVM/Clang
  - Part of oneAPI
  - Any CPU
  - Intel CPUs
  - Intel GPUs
  - Intel FPGAs
  - Level Zero

- **ComputeCp**
  - Multiple Backends
  - Any CPU
  - Intel CPUs
  - Intel GPUs
  - Intel FPGAs
  - AMD GPUs
  - Arm Mali
  - IMG PowerVR
  - Renesas R-Car

- **trisYCL**
  - Open source test bed
  - Any CPU
  - Intel CPUs
  - Intel GPUs
  - Intel FPGAs
  - XILINX FPGAs
  - POCL

- **hipSYCL**
  - CUDA and HIP/ROCm
  - Any CPU
  - Intel CPUs
  - Intel GPUs
  - XILINX FPGAs
  - AMD GPUs

- **neoSYCL**
  - SX-AURORA TSUBASA
  - Any CPU
  - AMD GPUs
  - Intel GPUs
  - NEC VEs

- **Bisheng SYCL/C++**
  - Any CPU
  - Intel CPUs
  - Intel GPUs
  - Intel FPGAs
  - Arm Mali
  - IMG PowerVR
  - Renesas R-Car

**Multiple Backends in Development**

There is development on supporting SYCL on even more low-level frameworks.

For more information: [http://sycl.tech](http://sycl.tech)
SYCL Ecosystem, Research and Benchmarks

**Implementations**
- neoSYCL SX-AURORA TSUBASA
- triSYCL
- Celerity
- kokkos
- DATA PARALLEL C++
- Field
- ComputeCpp
- hipSYCL
- GROMACS
- alaka
- Argonne National Laboratory
- Qualcomm
- University of Bristol
- AMD
- Arm
- University of Innsbruck
- Intel
- Xilinx
- SYCL

**Research**
- Taskflow: A General-purpose Parallel and Heterogeneous Task Programming System using Modern C++
- RAJA
- ATLAS
- STRIPE
- Exascale Computing Project
- ECP

**Benchmarks/Books**
- Direct Programming Benchmark
- SYCL-Bench
- free eBook
- 360k downloads

**Linear Algebra Libraries**
- BLAS
- FFT
- Math
- RAND
- SYCLBLAS
- oneMKL
- oneMKL
- oneMKL
- oneMKL
- SOLVER
- SPARSE
- TENSOR
- STL
- oneMKL
- oneMKL
- SYCL-DNN
- Eigen
- oneDNN

**Machine Learning Libraries and Parallel Acceleration Frameworks**
- SYCL-Parallel
- STL
- oneDPL
- TensorFlow

**Working Group Members**
- Working Group Members

**Syntactic Trees**
- Implementations
- Research
- Benchmarks/Books
- Linear Algebra Libraries
- Machine Learning Libraries and Parallel Acceleration Frameworks
- Working Group Members
Programming Models Must Persist

US National Laboratory Supercomputers 2021-2023

- HPC and now exascale computing requires programming models that endure for future workloads, > 20 years
- But Hardware changes frequently, constant improvement
- Programming models, have to be stable but also support latest HW,

Requires an open interface, across architectures with multiple implementations
**SYCL in HPC/Supercomputers**

### Simulation
- HPC Languages
- Solver Libraries, Parallel RT

### Data
- Productivity Languages
- Big Data Stack, Stats Lib, Databases

### Learning
- Productivity Languages
- Deep Learning, Linear Alg, ML

#### Three Pillars of Science Problem

- Need Languages that allow control of these Data Issues
  - Set Data affinity, Data Layout, Data movement, Data Locality, highly parameterized Code and dynamically compose the algorithms (C++ templates, parallel STL, inlining and fusion, abstractions)

- Libraries augment compiler optimizations for Performance Portable programs

- Use open standards to run Performance Portable code on new generation, or different vendor’s, hardware with compiler optimization, explicit parametrization and dynamically composed algorithm

---

Based on Hal Finkel’s IWOCL 2020 keynote
SYCL in Embedded Systems, Automotive, and AI

Networks trained on high-end desktop and cloud systems

Applications link to compiled inferencing code or call vision/inferencing API

Diverse Embedded Hardware
- Multi-core CPUs, GPUs
- DSPs, FPGAs, Tensor Cores
  * Vulkan only runs on GPUs

Open industry standards, enable flexible integration and deployment of multiple acceleration technologies

Hardware Acceleration APIs
- OpenCL
- Vulkan

Sensor Data
Safety Critical API Evolution

OpenCL and SYCL SC work will minimize API surface area, reduce ambiguity, UB, increase determinism.

New Generation Safety Critical APIs for Graphics, Compute and Display

Industry Need for GPU Acceleration APIs designed to ease system safety certification is increasing ISO 26262 / ASIL-D

UNECE WP.29

ISO 26262

ISO/PAS 21448

UL 4600

Rendering  Compute  Display
oneAPI and SYCL

- SYCL sits at the heart of oneAPI
- Provides an open standard interface for developers
- Defined by the industry
"this work supports the productivity of scientific application developers and users through performance portability of applications between Aurora and Perlmutter."

Codeplay works in partnership with US National Laboratories to enable SYCL on exascale supercomputers.
Nvidia and AMD Support in oneAPI

- Extending DPC++ to target Nvidia and AMD GPUs
- Supporting Perlmutter, Polaris and Frontier supercomputers
- Open source and available to everyone

Different targets using a simple compiler flag

clang++ -fsycl -fsycl-targets=nvptx64-nvidia-cuda
clang++ -fsycl -fsycl-targets=amdgcn-amd-amdhsa

https://www.codeplay.com/oneapiforcuda
Resources for AMD coming soon
SYCL Modern C++ Heterogeneous Model

• SYCL is becoming part of a standard programming model for HPC machines across the world
• SYCL is a global standards group with multiple company contributions
• SYCL moves with ISO C++, updates every 1.5-3 years
• SYCL is at the heart of oneAPI
• SYCL can adapt to the latest HPC hardware changes
• SYCL is being used in multiple top500 supercomputers
Enabling Industry Engagement

- SYCL working group values industry feedback
  - [https://community.khronos.org/c/sycl](https://community.khronos.org/c/sycl)
  - [https://sycl.tech](https://sycl.tech)

- SYCL FAQ
  - [https://www.khronos.org/blog/sycl-2020-what-do-you-need-to-know](https://www.khronos.org/blog/sycl-2020-what-do-you-need-to-know)

- What features would you like in future SYCL versions?

- Advisory Panel Chaired by Tom Deakin of U of Bristol

- SYCL Advisory Panel meeting here at IWOCL/SYCLCon

- Regular meetings to give feedback on roadmap and draft specifications

Open to all!
[https://community.khronos.org/](https://community.khronos.org/)
[https://app.slack.com/client/TDMDFS87M/CE9UX4CHG](https://app.slack.com/client/TDMDFS87M/CE9UX4CHG)
[https://community.khronos.org/c/sycl](https://community.khronos.org/c/sycl)
[https://stackoverflow.com/questions/tagged/sycl](https://stackoverflow.com/questions/tagged/sycl)
[https://www.reddit.com/r/sycl](https://www.reddit.com/r/sycl)
[https://github.com/codeplaysoftware/syclacademy](https://github.com/codeplaysoftware/syclacademy)
[https://sycl.tech/](https://sycl.tech/)

Khirnos SYCL Forums, Slack Channels, Stackoverflow, reddit, and SYCL.tech

Khirnos GitHub
Contribute to SYCL open source specs, CTS, tools and ecosystem

SYCL Working Group

SYCL Advisory Panels

Public contributions to Specification, Conformance Tests and software
[https://github.com/KhronosGroup/SYCL-CTS](https://github.com/KhronosGroup/SYCL-CTS)
[https://github.com/KhronosGroup/SYCL-Docs](https://github.com/KhronosGroup/SYCL-Docs)
[https://github.com/KhronosGroup/SYCL-Shared](https://github.com/KhronosGroup/SYCL-Shared)
[https://github.com/KhronosGroup/SYCL-Registry](https://github.com/KhronosGroup/SyIlParallelSTL)

Invited Experts
[https://www.khronos.org/advisors/](https://www.khronos.org/advisors/)

Khirnos members
[https://www.khronos.org/members/](https://www.khronos.org/members/)
[https://www.khronos.org/registry/SYCL/](https://www.khronos.org/registry/SYCL/)