SYCL State of the Union Keynote
SYCLCon 2021 Specification Release

Michael Wong
SYCL WG Chair
Codeplay Distinguished Engineer
ISOCPP Director & VP
ISO C++ Directions Group
michael@codeplay.com | wongmichael.com/about
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
- **Expanded interoperability**
  - Efficient acceleration by diverse backend acceleration APIs
- **SYCL atomic operations are now more closely aligned to standard C++ atomics**
  - Enhances parallel programming freedom
Parallel Industry Initiatives

- **C++11**
  - OpenCL 1.2
  - OpenCL 2.1
  - SPIR-V

- **C++14**
  - OpenCL 1.2
  - OpenCL C Kernel Language

- **C++17**
  - SYCL 1.2
  - SYCL 1.2.1
  - SPIR-V

- **C++20**
  - SYCL 2020
  - Many backend options

- **C++23**
  - SYCL 202X
  - Many backend options

Years:
- 2011
- 2015
- 2017
- 2020
- 202X
SYCL Evolution

SYCL 2020 compared with SYCL 1.2.1

- Easier to integrate with C++17 (CTAD, Deduction Guides...)
- Less verbose, smaller code size, simplify patterns
- Backend independent
- Multiple object archives aka modules simplify interoperability
- Ease porting C++ applications to SYCL
- Enable capabilities to improve programmability
- Backwards compatible but minor API break based on user feedback

SYCL 2020 Features

- Unified Shared Memory
- Parallel Reductions adds a built in reduction operation
- Work-group and sub-group algorithms
- Improvements to atomic operations
- Class template argument deduction (CTAD) and deduction guides
- Simplification of accessors
- Expanded interoperability with different backends
- Extension mechanism
- Address spaces
- Vector rework
- Specialization Constants

SYCL Future Roadmap (MAY CHANGE)

- Improving Software Ecosystem
  - Books, Tutorials, Tool, libraries, GitHub
- Expanding Implementation
  - DPC++
  - ComputeCpp
  - trISYCL
  - hipSYCL
  - neoSYCL
- Regular Maintenance Updates
  - Spec clarifications, formatting and bug fixes
    (https://www.khronos.org/registry/SYCL/)
- Conformance Tests
  - Working on Implementations
- Future SYCL NEXT Proposals

Repeat The Cycle every 1.5-3 years

SYCL Con 2020 Talks and Events

- SYCL, DPC++, SPUs, oneAPI - a View from Intel by James Reinders
- oneAPI Developer Summit Monday Apr 26, Biagio Cosenza, Peter Zuzek, Steffen Larsen
- Hands on SYCL Tutorial Tuesday Apr 27 by Rod Burns and SYCL team
- Sylkan: Towards a Vulkan Compute Target Platform for SYCL by Peter Thorman
- Performance-Portable Distributed K-Nearest neighbours using Locality-Sensitive Hashing and SYCL by Marcel Breyer
- Toward Performance Portability of Highly Parametrizable TRSM Algorithm Using SYCL by Thales Sabino
- On Measuring the Maturity of SYCL implementations by Tracking Historical Performance improvements by Wei-Chen Lin
- Experiences Supporting DPC++ in AMReX by Sravanthi Konda
- Developing Medical Imaging Applications Across GPU, FPGA, and CPU using oneAPI
- hipSYCL in 2021: Peculiarities, Unique Features and SYCL 2020 by Aksel Alpay
- Experiences with Adding SYCL Support to GROMACS by Andrew Alexenko
- Extending DPC++ with SSupport for Huawei Ascend AI Chipset
- Toward a Better SYCL Memory Consistency Model by Ben Ashb
- Bringing SYCL to A100 Ampere Architecture on Perlmutter Steffen Larsen and LBNL
- SYCL and OpenCL Meet Challenges of Functional Safety by Ilya Rudkin
- Enabling OpenCL and SYCL for RISC-V processors by Colin Davidson, Aidan Dodds
- SYCL Panel Thursday Apr 29

SYCL Evolution - Improvements

- Easier to integrate with C++17 (CTAD, Deduction Guides...)
- Less verbose, smaller code size, simplify patterns
- Backend independent
- Multiple object archives aka modules simplify interoperability
- Ease porting C++ applications to SYCL
- Enable capabilities to improve programmability
- Backwards compatible but minor API break based on user feedback

Integration of successful
Extrnsions plus new Core functionality

Over 40 Selected
Features for SYCL 2020

- Unified Shared Memory
- Parallel Reductions adds a built in reduction operation
- Work-group and sub-group algorithms
- Improvements to atomic operations
- Class template argument deduction (CTAD) and deduction guides
- Simplification of accessors
- Expanded interoperability with different backends
- Extension mechanism
- Address spaces
- Vector rework
- Specialization Constants

Repeat The Cycle every 1.5-3 years
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.

Multiple Backends in Development
There is development on supporting SYCL on even more low-level frameworks.
For more information: http://sycl.tech
SYCL user and developer Growth

Stack Overflow Questions

10X growth over 6 years
SYCL Ecosystem, Research and Benchmarks

Implementations
- neoSYCL
- SX-AURORA TSUBASA

Research
- Celerity
- ECP
- Kokkos
- RAJA

Benchmarks/Books
- Benchmark
- Direct
- Programming
- Benchmark
- Data Parallel C++
- GROMACS
- ALPACA
- SYCL-Bench
- FREE eBook

Linear Algebra Libraries
- BLAS
- FFT
- Math
- RAND

Machine Learning Libraries and Parallel Acceleration Frameworks
- SYCLBLAS
- oneMKL
- oneMKL
- oneMKL
- SYCL-DNN
- Eigen
- oneDNN
- SYCL
- Parallel STL
- oneDPL

Working Group Members
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

Sensor Data

Compilation Ingestion

Hardware Acceleration APIs

Dedicated Hardware

GPU
FPGA
DSP

Compiled Code

Vision / Inferencing Engine

C++ Application Code

Neural Network Training

NNEF

Trained Networks

Training 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

UNECE WP.29

ISO/IEC JTC 1/SC 42

UL 4600

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

“Xilinx is excited about the progress achieved with SYCL 2020,” said Ralph Wittig, fellow, Xilinx.

“For Renesas, SYCL is a key enabler for automotive ADAS/AD software developers …,” said Cyril Cordoba, Director of ADAS Segment Marketing Department, Renesas.

“NSITEXE supports the SYCL 2020 technology, which is gaining attention in embedded applications,” said Hideki Sugimoto, CTO, NSITEXE, Inc. “

“Imagination recognises the benefit of SYCL across multiple markets. Our software stacks have been designed to improve SYCL performance, enabling a straightforward path to exploit the teraflops of compute performance in our latest IP,” said Mark Butler, Vice President of Software Engineering, Imagination Technologies.

“Xilinx is excited about the progress achieved with SYCL 2020,” said Ralph Wittig, fellow, Xilinx.

“For Renesas, SYCL is a key enabler for automotive ADAS/AD software developers …,” said Cyril Cordoba, Director of ADAS Segment Marketing Department, Renesas.

“NSITEXE supports the SYCL 2020 technology, which is gaining attention in embedded applications,” said Hideki Sugimoto, CTO, NSITEXE, Inc. “

“Imagination recognises the benefit of SYCL across multiple markets. Our software stacks have been designed to improve SYCL performance, enabling a straightforward path to exploit the teraflops of compute performance in our latest IP,” said Mark Butler, Vice President of Software Engineering, Imagination Technologies.

SYCL support from embedded systems, through desktops to supercomputers
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

Choose Algorithm for target

Implement and Test Algorithm

Optimize Algorithm

Today’s Supercomputing Development Workflow needs knowledge of system architecture and tools that control data

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

Math, ML, Data Libraries; C++ Std, C, Python Libraries

OpenMP for C and Fortran
CUDA/pthreads/OpenACC/OpenCL
C++ Application uses SYCL, Kokkos, Raja

SYCL in HPC/Supercomputers


This work is licensed under a Creative Commons Attribution 4.0 International License
“Our users will benefit from features in the SYCL 2020 specification. New features, such as support for unified memory (USM) and reductions, are important capabilities for programming high-performance-computing hardware. ...” said Nevin Liber, computer scientist, Argonne National Laboratory’s Leadership Computing Facility.

“At Cineca, based on our experience, we confirm the value that SYCL is bringing to the development of high-performance computing in a hybrid environment. ...” said Sanzio Bassini, director of supercomputing, Application Innovation Dept, Cineca.

SYCL support from embedded systems, through desktops to supercomputers
"...As co-developers of the Celerity project, together with the University of Salerno, we are welcoming these changes and look forward to applying them within distributed-memory research and industry applications, for example as part of the recently launched EuroHPC LIGATE project," said Thomas Fahringer, head of the Distributed and Parallel Systems Group at the University of Innsbruck.

"...we see modern C++ language-based approaches to accelerator programming, such as SYCL, as an important component of our programming environment offering for users of Perlmutter," said Brandon Cook, application performance specialist at NERSC.

"The SYCL 2020 final specification brings significant features to the industry that enable C++ developers to more productively build high-performance heterogeneous applications with unified programming across XPU architectures," said Jeff McVeigh, Intel vice president, Datacenter XPU Products and Solutions.

SYCL support from embedded systems, through desktops to supercomputers
What now?

Deep Dive into HPC future
When I was OpenMP CEO, I learned

- HPC, exascale computing requires programming models that endure for future workloads, last 20 years
- But Hardware change frequently, constant improvement
- Mandate Sharing diverse hardware across a consortium
- Programming models, have to be stable but also support latest HW, open, covers multiple architecture with multiple implementations

OpenMP is great for C and Fortran

ISO base languages
OpenMP for C and Fortran
Open Acceleration Languages

SYCL is great for modern C++, AI, Automotive

Here are some opportunities for HPC growth across Europe, Asia
What about Europe? EPI, ARM and RISC-V RVV
SYCL as a universal programming model for HPC

Starting with US National Labs

Across Europe, Asia are many Petascale and pre-exascale systems

- With many variety of CPUs GPUs FPGAs, custom devices
- Often with interconnected usage agreements
- Let’s look at Europe first:
HPCAsia 2021: neoSYCL thanks to Hiroyuki Takizawa

Open standard for offload programming = SYCL

BFS using Rodina Benchmark at HPC Asia 2021

No loss in performance between using SYCL and VEO

Programming with SYCL
Leads to lower Code Complexity

<table>
<thead>
<tr>
<th>Application</th>
<th>VERSION</th>
<th>NLOC</th>
<th>AvgCCN</th>
<th>Avg.token</th>
</tr>
</thead>
<tbody>
<tr>
<td>STREAM</td>
<td>SYCL</td>
<td>148</td>
<td>1.2</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>VE-Offload</td>
<td>296</td>
<td>3.8</td>
<td>159.7</td>
</tr>
<tr>
<td>N-body</td>
<td>SYCL</td>
<td>86</td>
<td>2.7</td>
<td>233.3</td>
</tr>
<tr>
<td></td>
<td>VE-Offload</td>
<td>166</td>
<td>5</td>
<td>240.2</td>
</tr>
<tr>
<td></td>
<td>Origin</td>
<td>66</td>
<td>3.3</td>
<td>173.7</td>
</tr>
<tr>
<td>BFS</td>
<td>SYCL</td>
<td>133</td>
<td>4.5</td>
<td>248</td>
</tr>
<tr>
<td></td>
<td>VE-Offload</td>
<td>225</td>
<td>7.4</td>
<td>302</td>
</tr>
<tr>
<td></td>
<td>Origin</td>
<td>116</td>
<td>4.5</td>
<td>196.2</td>
</tr>
</tbody>
</table>
Final words

• SYCL can be a part of a standard programming model for all HPC including Europe/Asia/NA
  • HPC is now used in Embedded and Automotive
• SYCL is home grown EU, UK company led its development since 2012, now open standard with multiple company contributions, lots of European/Asia projects
  • Celerity from the University of Innsbruck and Salerno, CINECA Bologna, neoSYCL
• Moves with ISO C++, updates every 1.5-3 years
• Part of oneAPI
• Adapts to HPC hardware changes, moving towards safety critical
• Adapted by ECP for first Exascale computer in Aurora, now also in the Perlmutter, and we hope in European and Asia HPC
Enabling Industry Engagement

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

- SYCL FAQ
  - 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

Public contributions to Specification, Conformance Tests and software
  https://github.com/KhronosGroup/SYCL-CTS
  https://github.com/KhronosGroup/SYCL-Docs
  https://github.com/KhronosGroup/SYCL-Shared
  https://github.com/KhronosGroup/SYCL-Registry
  https://github.com/KhronosGroup/SyclParallelSTL

Kronos SYCL Forums, Slack Channels, Stackoverflow, reddit, and SYCL.tech
  Open to all!
  https://community.khronos.org/www.khr.io/slack
  https://app.slack.com/client/TDMDFS87M/CE9UX4CHG
  https://community.khronos.org/c/sycl/
  https://stackoverflow.com/questions/tagged/sycl
  https://www.reddit.com/r/sycl
  https://github.com/codeplaysoftware/syclacademy
  https://sycl.tech/

Invited Experts
  https://www.khronos.org/advisors/

Kronos members
  https://www.khronos.org/members/
  https://www.khronos.org/registry/SYCL/