The 8th International Workshop on OpenCL and the SYCL Developer Conference

Live Webinar:
APRIL 28 (16:00 BST)
Register to Join:
www.iwocl.org/iwocl-2020/conference-program/#panel

Thanks to our sponsors
Adoption of parallel programming languages

Papers mentioning parallel programming languages.
Data according to Google Scholar (April 27th 2020)

(c) Simon McIntosh-Smith 2020
Agenda

• **Introduction** by Simon McIntosh-Smith, IWOCL / SYCLcon chair, University of Bristol

• **OpenCL state of the union and announcements** by Neil Trevett, VP at NVIDIA and Khronos President

• **SYCL status and updates** by Michael Wong, SYCL working group chair and VP of software at Codeplay
OpenCL 3.0

Neil Trevett
Khronos President
OpenCL Working Group Chair
NVIDIA VP Developer Ecosystems
ntrevett@nvidia.com | @neilt3d
April 2020

© Khronos Group 2020
This work is licensed under a Creative Commons Attribution 4.0 International License
OpenCL is Widely Deployed and Used

Accelerated Implementations

The industry’s most pervasive, cross-vendor, open standard for low-level heterogeneous parallel programming

Parallel Languages
- OpenACC
- SYCL
- OpenVINO
- Intel
- AMD
- NVIDIA
- ARM
- Apple
- ALTERA
- AMD
- ARM
- Qualcomm
- Samsung
- NVIDIA

Desktop Creative Apps
- Adobe
- otoy
- Dassault Systemes
- Sony
- CyberLink
- Autodesk
- Modo
- CHAOSGROUP
- ptc
- V-Ray
- darktable
- aparapi
- OpenCL
- PyOpenCL
- SYCL
- SYCL-Blas
- ViennaCL
- Blast
- NVIDIA
- Intel
- Xiaomi
- MACE
- PyTorch
- Microsoft
- NVIDIA
- Intel
- Synopsys
- MetaWare EV
- MAE
- TVM
- tvml
- Caffe
- Acuity
- VertiSilicon
- OpenMM
- PLaidML
- FAST
- OpenVX
- Qualcomm
- Knights Landing
- TI DL Library (TIDL)
- Arm Compute Library
- NVIDIA
- Intel
- Synopsys
- MetaWare EV
- MAE
- TVM
- tvml
- Caffe
- Acuity
- VertiSilicon
- OpenMM
- PLaidML
- FAST
- OpenVX
- Qualcomm
- Knights Landing
- TI DL Library (TIDL)
- Arm Compute Library

Machine Learning Libraries and Frameworks
- Intel cDNN
- Xiaomi MACE
- PyTorch
- Microsoft
- NVIDIA
- Intel
- Synopsys
- MetaWare EV
- MAE
- TVM
- tvml
- Caffe
- Acuity
- VertiSilicon
- OpenMM
- PLaidML
- FAST
- OpenVX
- Qualcomm
- Knights Landing
- TI DL Library (TIDL)
- Arm Compute Library

Linear Algebra Libraries
- SYCL-Blas
- CLBlast
- Arm Compute Library

Desktop Creative Apps
- Modo
- CHAOSGROUP
- ptc
- V-Ray
- darktable
- aparapi
- OpenCL
- PyOpenCL
- SYCL
- SYCL-Blas
- ViennaCL
- Blast
- NVIDIA
- Intel
- Xiaomi
- MACE
- PyTorch
- Microsoft
- NVIDIA
- Intel
- Synopsys
- MetaWare EV
- MAE
- TVM
- tvml
- Caffe
- Acuity
- VertiSilicon
- OpenMM
- PLaidML
- FAST
- OpenVX
- Qualcomm
- Knights Landing
- TI DL Library (TIDL)
- Arm Compute Library

Vision and Imaging Libraries
- VisionCpp
- Metashape
- Halide
- FFmpeg
- GNU Octave
- Wolfram Mathematica
- Matlab
- ArrayFire
- Matplotlib

Molecular Modelling Libraries
- Charmm
- Gromacs
- Folding At Home
- ForceBalance
- BioVista
- Schrodinger

Math and Physics Libraries
- TensorFlow
- JAX
- PyTorch
- Microsoft
- NVIDIA
- Intel
- Synopsys
- MetaWare EV
- MAE
- TVM
- tvml
- Caffe
- Acuity
- VertiSilicon
- OpenMM
- PLaidML
- FAST
- OpenVX
- Qualcomm
- Knights Landing
- TI DL Library (TIDL)
- Arm Compute Library

Accelerated Implementations

This work is licensed under a Creative Commons Attribution 4.0 International License

OpenCL Open Source Ecosystem Momentum

By April 13, Folding@Home hit a new record of 2.4 exaflops, faster than the top 500 traditional supercomputers combined, thanks to almost 1 million new members of the network. Folding@Home uses OpenCL to offload computations onto the GPUs contained in the networked home PCs.
OpenCL Standards Evolution

**Next Steps**
- SYCL 2020
  - Multiple backends
- OpenCL Layered Over Multiple Target APIs for Deployment Flexibility
- Broad LLVM/Clang Language Cooperation
- Increased Pervasive Core Functionality for App Portability
- Increased Optionality for Embedded Flexibility

**Design Influences**
- SYCL 1.2
  - Over OpenCL 1.2 using SPIR 2015
- SYCL 2.2
  - Over OpenCL 2.X 2016
  - Over OpenCL using SPIR-V 2017
- SYCL 1.2.1
  - Over OpenCL using SPIR-V 2017

**Kernel Languages**
- SPIR
  - Using LLVM IR 2014
- SPIR-V
  - Standalone IR 2015

**OpenCL APIs**
- OpenCL C Specification 2008
- OpenCL 1.2
  - Creates Parallel Programming Baseline 2008-2011
- OpenCL 2.X
  - Functionality in Monolithic Specification 2013-2017
- OpenCL 3.0
  - 2.X Functionality with Optionality 2020

**Vulkan**
- Vulkan 1.0
  - 2016
- Vulkan Convergence 2018
- Vulkan 1.2
  - 2020

This work is licensed under a Creative Commons Attribution 4.0 International License
OpenCL 3.0

Increased Deployment Flexibility
All functionality beyond OpenCL 1.2 is optional
Unified API specification slices OpenCL 2.X functionality into coherent, queryable, optionality
OpenCL C 3.0 language specification adds macros for optional language features

Subgroups with SPIR-V 1.3
New (optional) core functionality

Asynchronous DMA extension
Enabling a new class of Embedded Processors

OpenCL C++ not included
Ecosystem has transitioned to open source C++ for OpenCL

Easy for Developers to upgrade to OpenCL 3.0
NO code changes necessary if all used functionally is present
Applications encouraged to query used OpenCL 2.X functionality for future portability

Easy for Implementers to upgrade to OpenCL 3.0
Add queries for OpenCL 2.X functionality - missing or present
Update reported version and add minor entry points for improved app portability
OpenCL Roadmap

**Unified API Specification**
All OpenCL versions documented in one place
Tightly organized queries for all 2.X functionality
OpenCL C 3.0 Language - macros for optional features

**Subgroups and SPIR-V 1.3**
New (optional) core functionality

**Asynchronous DMA extension**
Enabling a new class of Embedded Processors

**Extension Pipeline**
Extended Subgroups
Device UUID Query
Extended Debug Info
External Memory Sharing
Vulkan/OpenCL Interop
Recordable Command buffers?
Machine Learning Primitives?
Indirect Dispatch?
Device Topology?

**Khronos OpenCL SDK**
Headers, Utility Libraries,
Documentation, Samples, ICD Loader

**Open Source Ecosystem**
Tools, Domain Libraries

**SPIR-V 1.4/1.5 ingestion**
Compiler efficiency and expressiveness

**Regular Maintenance Updates**
Clariﬁcations, formatting, bug ﬁxes

**C++ for OpenCL**
Open source C++ kernel language
front-end leveraging Clang and LLVM

**Regular Maintenance Updates**
Clariﬁcations, formatting, bug ﬁxes

**OpenCL 3.0**
April 2020

**New Pervasive Functionality in Core Speciﬁcation**
Integrate proven, widely adopted extensions

**Flexible Proﬁle**
Finer-grain optional functionality for embedded processors

‘Layering’ Proﬁle?
Defined queries and conformance for layered implementations?

**OpenCL Next**
Faster core release cycle

**Time**
## API Layering

<table>
<thead>
<tr>
<th>Layers Over</th>
<th>Vulkan</th>
<th>OpenGL</th>
<th>OpenCL</th>
<th>OpenGL ES</th>
<th>DX12</th>
<th>DX9-11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vulkan</td>
<td>Zink</td>
<td>clspv</td>
<td>GLOVE</td>
<td>vkd3d</td>
<td>DXVK</td>
<td>WineD3D</td>
</tr>
<tr>
<td>OpenGL</td>
<td>gfx-rs</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DX12</td>
<td>gfx-rs</td>
<td>Microsoft ‘GLOn12’</td>
<td>Microsoft ‘CLOn12’</td>
<td>Microsoft D3D11On12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DX9-11</td>
<td>gfx-rs</td>
<td>Angle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metal</td>
<td>MoltenVK</td>
<td>clspv + SPIRV-Cross?</td>
<td>MoltenGL</td>
<td>Angle</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**COLUMNS** Benefit ISVs by making an API available everywhere

Application deployment flexibility by fighting platform fragmentation

Making an API available across multiple platforms even if no native drivers available

**ROWS** Benefit Platforms by adding APIs

Enable content without additional kernel level drivers

Enabled by growing robustness of open source compiler ecosystem
Developers - Please Give Us Feedback!

- Is the set of optional features sliced too finely, or too coarsely?
  - Are they easy to understand?

- Which optional features do you expect to use in your application or library?
  - Usage data drives which optional features should be made mandatory in future

- What new features do you most need?
  - We will use extensions to prove new functionality before adding to core specification
  - What extensions would you like to see in the second half of 2020?

OpenCL Working Group has maximized information in Khronos public GitHub to accelerate finalization

Provisional OpenCL 3.0 Specification sources released on GitHub
https://www.khronos.org/registry/OpenCL/

OpenCL 3.0 Conformance Tests WIP sources released on GitHub
https://github.com/KhronosGroup/OpenCL-CTS

Spec feedback and pull requests welcome on GitHub
https://github.com/KhronosGroup/OpenCL-Docs/issues

Tests feedback and pull requests welcome on GitHub
https://github.com/KhronosGroup/OpenCL-CTS/issues

Vendor OpenCL 3.0 Implementations in flight

Urgency to Finalize and Ship
Finalized OpenCL 3.0 Specifications
Completed Conformance Tests
Multiple Shipping Conformant Implementations
SYCL WG
State of the Union 2020

Michael Wong
SYCL WG Chair
Codeplay VP of R&D
ISO/IEC JTC1/SC22/WG21 Director & VP
ISO C++ Directions Group Chair

michael@codeplay.com  |  wongmichael.com/about
<table>
<thead>
<tr>
<th>Year</th>
<th>SYCL Version</th>
<th>C++ Standard</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>SYCL 1.2</td>
<td>C++11</td>
<td>Single source programming</td>
</tr>
<tr>
<td></td>
<td>OpenCL 1.2</td>
<td></td>
<td>OpenCL C Kernel Language</td>
</tr>
<tr>
<td></td>
<td>SPIR-V in Core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td>SYCL 1.2.1</td>
<td>C++11</td>
<td>Single source programming</td>
</tr>
<tr>
<td></td>
<td>OpenCL 2.1</td>
<td></td>
<td>OpenCL 2.1</td>
</tr>
<tr>
<td></td>
<td>SPIR-V in Core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>SYCL 1.2.1</td>
<td>C++11</td>
<td>Single source programming</td>
</tr>
<tr>
<td></td>
<td>OpenCL 2.2</td>
<td></td>
<td>OpenCL 2.2</td>
</tr>
<tr>
<td></td>
<td>SPIR-V in Core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2020</td>
<td>SYCL 2020</td>
<td>C++17</td>
<td>Single source programming</td>
</tr>
<tr>
<td></td>
<td>OpenCL 3.0</td>
<td></td>
<td>OpenCL 3.0</td>
</tr>
<tr>
<td></td>
<td>SPIR-V in Core</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2021</td>
<td>SYCL 2021</td>
<td>C++20</td>
<td>Single source programming</td>
</tr>
<tr>
<td></td>
<td>OpenCL 3.0</td>
<td></td>
<td>OpenCL 3.0</td>
</tr>
<tr>
<td></td>
<td>SPIR-V in Core</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SYCL community is vibrant

SYCL F2F meetings attendance

- 2X growth
- SYCL-1.2.1

Annual Meetings:
- 2016/04 Frankfurt
- 2016/10 Seoul
- 2017/04 Vancouver
- 2017/09 Amsterdam
- 2018/01 Chicago
- 2018/04 Taipei
- 2018/09 Montreal
- 2019/01 Budapest
- 2019/04 San Diego
- 2019/09 Singapore
- 2020/02 New Orleans
- 2020/02 Barcelona
**SYCL Evolution**

**SYCL 2020 Potential Features**
- Generalization (a.k.a the Backend Model) presented by Gordon Brown
- Unified Shared Memory (USM) presented by James Brodman
- Improvement to Program class Modules presented by Gordon Brown
- Host Task with Interop presented by Gordon Brown
- In order queues, presented by James Brodman

**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

**Integration of successful Extensions plus new Core functionality**

**SYCL 2020 Roadmap (WIP, MAY CHANGE)**

- **2017**
  - SYCL 1.2.1
- **2017**
  - Improving Software Ecosystem
    - Tool, libraries, GitHub
- **2018**
  - Expanding Implementation
    - DPC++
    - ComputeCpp
    - triSYCL
    - hipSYCL
- **2019**
  - Regular Maintenance Updates
    - Spec clarifications, formatting and bug fixes
    - [https://www.khronos.org/registry/SYCL/](https://www.khronos.org/registry/SYCL/)
- **Target 2020**
  - Provisional Q3 then Final Q4
  - Selected Extension Pipeline aiming for SYCL 2020 Provisional Q3
    - Reduction
    - Subgroups
    - Accessor simplification
    - Atomic rework
    - Extension mechanism
    - Address spaces
    - Vector rework
    - Specialization Constants

**Converge SYCL with ISO C++ and continue to support OpenCL to deploy on more devices**
- CPU
- GPU
- FPGA
- AI processors
- Custom Processors

Repeat The Cycle every 1.5-3 years

This work is licensed under a Creative Commons Attribution 4.0 International License
SYCL Ecosystem, Research and Benchmarks

Active Working Group Members

Implementations

Research

Benchmarks

Linear Algebra Libraries

Machine Learning Libraries and Parallel Acceleration Frameworks

oneAPI

DATA PARALLEL C++

Celerity

Exascale Computing Project

SYCL-BLAS

SYCL-BLAS

SYCL-ML

SYCL-DNN

Eigen

TensorFlow

oneMKL

SYCL Parallel STL

RSBench

SYCL-ML

SYCL-DNN

Machine Learning Libraries and Parallel Acceleration Frameworks

Active Working Group Members

Implementations

Research

Benchmarks

Linear Algebra Libraries

© The Khronos Group Inc. 2020 - Page 17

This work is licensed under a Creative Commons Attribution 4.0 International License
SYCL 2020 Provisional is coming

- In a few months, SYCL 2020 provisional will be released
- We need your feedback asap
  - [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://sycl.tech](https://sycl.tech)
- What features are you looking for in SYCL 2020?
- What feature would you like to aim for in future SYCL?
- How do you join SYCL?
Engaging with the Khronos SYCL Ecosystem

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

SYCL Working Groups

SYCL Advisory Panels

Khrnos SYCL Forums, Slack Channels, stackoverflow, reddit, and SYCL.tech

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.

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://https://sycl.tech/

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

Spec fixes and suggestions made under the Khronos IP Framework. Open source contributions under repo’s CLA - typically Apache 2.0

Invited Advisors under the Khronos NDA and IP Framework can comment and contribute to requirements and draft specifications

Any member or non-member can propose a new SYCL feature or fix

Khronos members under Khronos NDA and IP Framework participate and vote in working group meetings. Starts at $3.5K/yr.
Thank You!

- Khronos SYCL is creating cutting-edge royalty-free open standard
  - For C++ Heterogeneous compute, vision, inferencing acceleration
- Information on Khronos SYCL Standards: [https://www.khronos.org/sycl](https://www.khronos.org/sycl)
- Any entity/individual is welcome to join Khronos SYCL: [https://www.khronos.org/members](https://www.khronos.org/members)
- Join the SYCLCon Tutorial Monday and Wednesday Live panel: Wednesday Apr 29 15:00-18:00 GMT
  - Have your questions answered live by a group of SYCL experts
- Michael Wong: michael@codeplay.com | wongmichael.com/about

**Benefits of Khronos membership**

- Gather industry requirements for future open standards
- Gain early insights into industry trends and directions
- Influence the design and direction of key open standards that will drive your business
- Draft Specifications Confidential to Khronos members
- Publicly Release Specifications and Conformance Tests
- Network with domain experts from diverse companies in your industry
- State-of-the-art IP Framework protects your Intellectual Property
- Accelerate your time-to-market with early access to specification drafts
- Enhance your company reputation as an industry leader through Khronos participation
Panel Chair and Announcements by:

Simon McIntosh-Smith  
University of Bristol  
Conference Chair

Neil Trevett  
OpenCL Working Group Chair  
Khronos President, VP NVIDIA

Michael Wong  
SYCL Working Group Chair  
VP of R&D, Codeplay

Alastair Murray  
Codeplay  
Principal SW Eng. Compilers

Ben Ashbaugh  
Intel  
Principal Engineer

Dennis Adams  
Sony Creative Software  
Director of Technology

Eric Berdahl  
Adobe  
Senior Engineering Manager

Hal Finkel  
Argonne National Laboratory  
Lead for Compiler Technology

Jeremy Kemp  
Imagination  
Snr. Software Design Engineer

Kévin Petit  
Arm  
Principal Software Architect

Martin Schreiber  
Technical University of Munich  
Researcher, IWOCL Local Chair

Ronan Keryell  
Xilinx  
Principal Software Engineer

Live Webinar:  
APRIL 28 (16:00 BST)  
Register to Join:  
www.iwocl.org/iwocl-2020/conference-program/#panel