State of the Union
OpenCL Working Group
Neil Trevett
Khronos President
OpenCL Working Group Chair
NVIDIA VP Developer Ecosystems
ntrevett@nvidia.com | @neilt3d
Primary Developments Since IWOCL 2020

**OpenCL 3.0 Finalized and Released**
Including subgroups and embedded processor extensions

**Multiple OpenCL 3.0 Implementations Shipping**
With many more Adopters in the pipeline

**Conformance Testing Improvements**
>250 commits to the OpenCL test suite since IWOCL 2020

**Regular Releases**
Including OpenCL 3.0.7 here at IWOCL with new extensions!

**OpenCL Guide Released and SDK Enhanced!**
Tutorial on SDK Layers here at IWOCL

**C++ for OpenCL gaining momentum**
Interaction with LLVM community deepening

**Increased activity in layered implementations**
Microsoft’s OpenCLon12 in addition to Google’s clspv

**Roadmap Discussions Underway**
Building Advisory Panel Interactions
OpenCL Open-Source Project Momentum

# OpenCL-based GitHub Repos

Tripling in the last four years
OpenCL 3.0

Increased Ecosystem Flexibility
All functionality beyond OpenCL 1.2 queryable
Macros for optional OpenCL C language features
Widely adopted extensions to be integrated into core

OpenCL C++ for OpenCL
Open-source C++ for OpenCL front end compiler combines
OpenCL C and C++17 replacing OpenCL C++ language spec

Unified Specification
All versions of OpenCL in one specification for easier
maintenance, evolution and accessibility
Source on Khronos GitHub for community feedback,
functionality requests and bug fixes

New Functionality
Subgroups with SPIR-V 1.3 in core (optional)
Asynchronous DMA extension for embedded processors

Easy OpenCL 3.0 migration for applications
OpenCL 1.2 applications - no change
OpenCL 2.X applications - no code changes if all used
functionality present
Queries recommended for future portability

https://www.khronos.org/registry/OpenCL/

Released September 2020
OpenCL is Widely Deployed and Used

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

Desktop Creative Apps

Parallel Languages

Machine Learning Libraries and Frameworks

Molecular Modelling Libraries

Linear Algebra and FFT Libraries

Math and Physics Libraries

Machine Learning Compilers

Vision, Imaging and Video Libraries

Accelerated Implementations

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

OpenCL 3.0 Adopters

- ARM
- Codeplay
- Imagination
- Kalray
- Microsoft
- QNX
- Qualcomm
- VeriSilicon
- NVIDIA
- Intel

OpenCL 3.0 Adopters Already Shipping Conformant Implementations

Product Conformance Status
https://www.khronos.org/conformance/adopters/conformant-products/opencl
C++ for OpenCL

- Open-Source Compiler Front-end
  - Replaces the OpenCL C++ kernel language spec
  - Official releases published in OpenCL-Docs repo

- Enables full OpenCL C and most C++17 capabilities
  - OpenCL C code is valid and fully compatible
  - Enables gradual transition to C++ for existing apps
  - Language documentation

- Supported in Clang since release 9.0
  - Generates SPIR-V 1.0 plus SPIR-V 1.2 where necessary
  - Full details are provided in OpenCL-Guide

- Online compilation via cl_ext_cxx_for_opencl

OpenCL Offline Compiler Flow

Language Definitions

OpenCL C

C++ for OpenCL

Clang

LLVM

SPIR-V LLVM IR Translator (and other optional SPIR-V tools)

SPIR-V

OpenCL

Most of C++17:
- inheritance,
- templates,
- type deduction,
...

Check it out in Compiler Explorer
Asynchronous DMA Extensions

OpenCL embraces a new class of Embedded Processors

Many DSP-like devices have Direct Memory Access hardware

Transfer data between global and local memories via DMA transactions
Transactions run asynchronously in parallel to device compute enabling wait for transactions to complete
Multiple transactions can be queued to run concurrently or in order via fences

OpenCL abstracts DMA capabilities via extended asynchronous workgroup copy built-ins

(New!) 2- and 3-dimensional async workgroup copy extensions support complex memory transfers
(New!) async workgroup fence built-in controls execution order of dependent transactions
New extensions complement the existing 1-dimensional async workgroup copy built-ins

Async 3D-3D Copy Transaction

Async Fence controls order of dependent transactions

All transactions prior to async_fence must complete before any new transaction starts, without a synchronous wait

The first of significant upcoming advances in OpenCL to enhance support for embedded processors
OpenCL 3.0.7 Release at IWOCL

Second Maintenance release since OpenCL 3.0 in September 2020
Clarifications, formatting, bug fixes
Adds optional extensions

`cl_khr_spirv_extended_debug_info`
Enables SPIR-V modules to use the OpenCL.DebugInfo.100 extended instruction set

`cl_khr_pci_bus_info`
Query PCI domain, bus, device, and function information for an OpenCL device

`cl_khr_extended_bit_ops`
Adds OpenCL C built-in functions to insert, extract, and reverse bits in a bitfield

`cl_khr_suggested_local_work_size`
Adds a query for a suggested local work group size for a kernel running on an OpenCL device

`cl_khr_spirv_linkonce_odr`
Enables LinkOnceODR SPIR-V link type to separately compile and link C++ programs

Specification available on the OpenCL Registry
OpenCL SDK - In Development

- Bringing together all the components needed to develop OpenCL applications
  - OpenCL Headers (include/api)
  - OpenCL C++ bindings (include/cpp)
  - OpenCL Utility Libraries (include/utils)
  - Build system and CI
- Other resources useful to OpenCL developers
  - OpenCL Guide
  - Code samples (samples/)
  - Documentation (docs/)
- Loader and Layers
  - Initial layers implemented
  - SDK and Layers Tutorial here at IWOCL
- Watch GitHub Repo for updates
  - Community contributions welcome!

More Information at
https://github.com/KhronosGroup/OpenCL-SDK
## 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>clvk</td>
<td>GLOVE</td>
<td>Angle</td>
<td>DXVK</td>
</tr>
<tr>
<td>OpenGL</td>
<td>gfx-rs</td>
<td>Ashes</td>
<td></td>
<td>Angle</td>
<td>vkd3d-Proton</td>
<td>WineD3D</td>
</tr>
<tr>
<td>DX12</td>
<td>gfx-rs</td>
<td></td>
<td>Microsoft ‘GLOn12’</td>
<td>Angle</td>
<td>Microsoft D3D11On12</td>
<td></td>
</tr>
<tr>
<td>DX9-11</td>
<td>gfx-rs</td>
<td>Ashes</td>
<td></td>
<td>Angle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metal</td>
<td>MoltenVK</td>
<td>gfx-rs</td>
<td></td>
<td>MoltenGL</td>
<td>Angle</td>
<td></td>
</tr>
</tbody>
</table>

**Enabled by growing robustness of open-source compiler ecosystem**

**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

© The Khronos® Group Inc. 2021 - Page 11
SPIR-V enables a rich ecosystem of languages and compilers to target low-level APIs such as Vulkan and OpenCL, including deployment flexibility: e.g., running OpenCL kernels on Vulkan.
Layered OpenCL Implementations

**clspv + clvk**
- clspv - Google’s open-source OpenCL kernel to Vulkan SPIR-V compiler
  - Tracks top-of-tree LLVM and Clang - not a fork
- Clvk - prototype open-source OpenCL to Vulkan run-time API translator
  - Used by shipping apps and engines on Android e.g., Adobe Premiere Rush video editor - 200K lines of OpenCL C kernel code

**OpenCLOn12**
- Microsoft and COLLABORA
  - GPU-accelerated OpenCL on any DX12 PC and Cloud instance (x86 or Arm)
  - Leverages Clang/LLVM AND MESA
- OpenGLOn12 - OpenGL 3.3 over DX12 is already conformant
OpenCL Roadmap

Extensions in Ratification
Expected Public Release 2Q 2021

**cl_khr_integer_dot_product**
Adds support for SPIR-V instructions and OpenCL C built-in functions to compute the dot product of vectors of integers

External Sharing Extensions (Provisional)

**cl_khr_external_memory**
Create OpenCL memory objects from OS-specific memory handles (similar to VK_KHR_external_memory)

**cl_khr_semaphore**
Semaphore synchronization object that can be signaled and reset multiple times and signaled from outside OpenCL

The **cl_khr_external_semaphore** and
**cl_khr_external_semaphore_sync_fd**
Create OpenCL semaphore objects from OS-specific semaphore handles

**cl_khr_vk_sharing extension**
Associate an OpenCL context with a Vulkan physical device

The External Sharing Extensions are Provisional to enable developer feedback before finalization

OpenCL imports memory & semaphore handles created by Vulkan

Vulkan/OpenCL Interop

Use semaphores to synchronize memory ownership & access

External Sharing Extensions
Generic extensions to import external memory and semaphores exported by other APIs
API-specific interop extensions e.g., Vulkan
More flexible than previous interop APIs using implicit resources

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

© The Khronos® Group Inc. 2021 - Page 14
Longer Term Roadmap Discussions

- Command Buffer Recording and Replay
- Unified Shared Memory
- Floating-point Atomics
- Global Barriers
- YUV Multi-planar Images
- Generalized Image from Buffer
- Indirect Dispatch
- Collective Programming
- Expect and Assume Optimization Hints
- Required Subgroup Size
- Machine Learning Operations
- Extended Async Copies
- 2D and 3D Prefetch Built In Functions

New functionality is proven as extensions before being added to core

Developer Feedback Welcome!
What is your highest priority?
What is missing?
Requirements and use cases
See ‘Extensions Feedback’ issue on GitHub
https://github.com/KhronosGroup/OpenCL-Docs/issues/604
OpenCL Advisory Panel

Specification drafts and invitations to provide requirements and detailed feedback

Working Group makes decisions on standards evolution

Shared Email list and Repository

Hosted by Khronos. Under Khronos NDA

Requirements and detailed feedback on specification drafts

Advisory Panel

Panel Members
Invited industry experts.
$0 Cost.
Covered by NDA and IP Framework

Khranos Members
Any company can join.
Membership Fee.
Covered by NDA and IP Framework

Working groups can share draft specifications and accept detailed design contributions as Panel Members are covered by IP Framework

Chaired by Máté Ferenc Nagy-Egri at StreamHPC
OpenCL Advisory Panel meeting here at IWOCL
Regular meetings to give feedback on roadmap and draft specifications

Please reach out to opencl-chair@lists.khronos.org if you wish to apply
Developers - Please Give Us Feedback!

- How is your transition to using OpenCL 3.0?
  - Are you encountering any issues?

- 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?
  - What roadmap extensions would you prioritize, and are there any gaps?
  - https://github.com/KhronosGroup/OpenCL-Docs/issues/604

- Consider applying to join the OpenCL Advisory Panel!
  - Email opencl-chair@lists.khronos.org

More OpenCL information!
https://www.khronos.org/opencl/

Feedback Welcome!
https://github.com/KhronosGroup/OpenCL-Docs