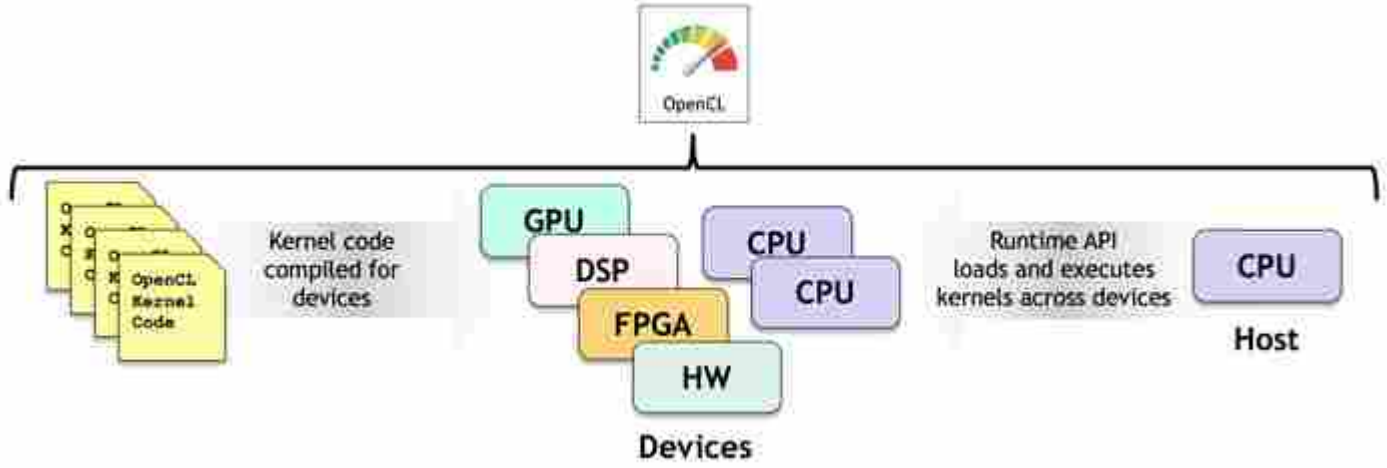
JÜLICH
FORSCHUNGSZENTRUM

OpenCL: History & Future

November 20, 2017

OpenCL – Portable Heterogeneous Computing



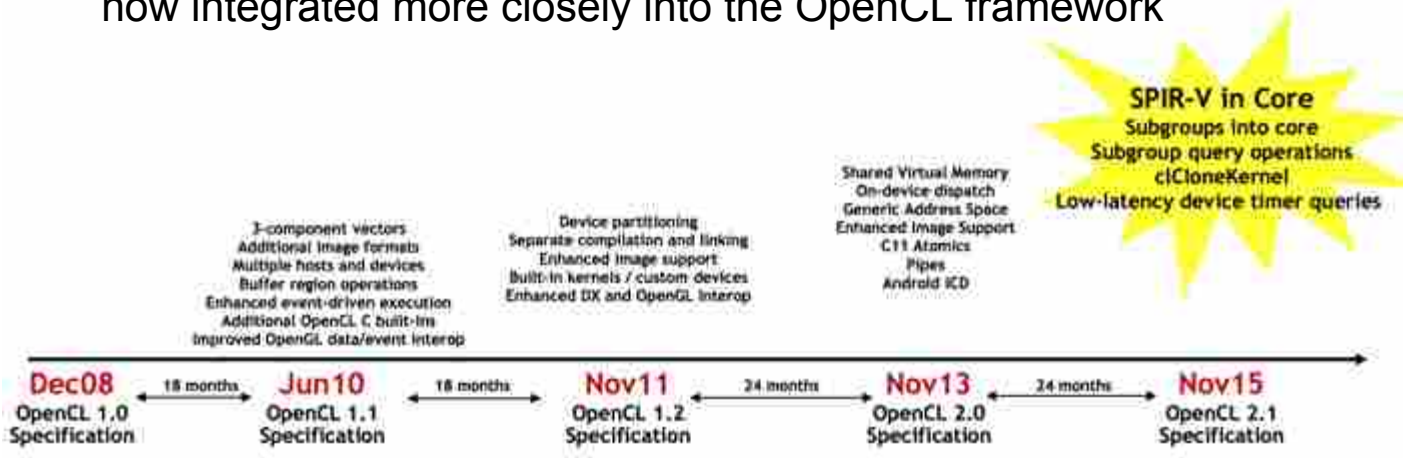
- 2 APIs and 2 kernel languages
  - C Platform Layer API
  - OpenCL C and C++ kernel language to write parallel code
  - C runtime API to build and execute kernels across multiple devices
- one code tree can be executed on CPUs, GPUs, DSPs, FPGAs, and hardware
  - dynamically balance work across available platforms

OpenCL 1.x

- 1.0 : released with Apple Mac OS X Snow Leopard on August 28, 2009
  - support announced by many companies, e.g. AMD, NVIDIA, IBM
- 1.1 : released June 2010, major new features:
  - new data types, e.g. 3-component vectors
  - handling commands from multiple host threads
  - processing buffers across multiple devices
  - subbuffers
  - operations on buffer regions
  - enhanced use of events to drive command execution
  - new built-in C functions, e.g. integer clamp, shuffle, and asynchronous strided copies
  - improved OpenGL interoperability

OpenCL 1.x

- 1.2 : released November 2011, major new features:
  - device partitioning: the ability to partition a device into sub-devices; work assignments can be allocated to individual compute units
  - separate compilation and linking of objects; functionality to compile OpenCL into external libraries
  - enhanced image support: 1.2 adds support for 1D images and 1D/2D image arrays
  - built-in kernels: custom devices that contain specific unique functionality are now integrated more closely into the OpenCL framework



OpenCL 2.x

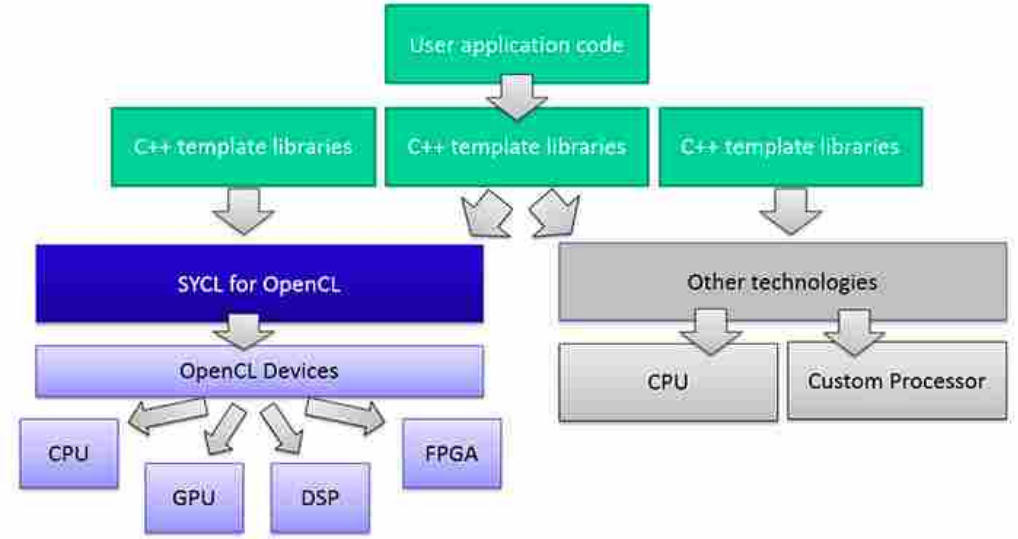
- 2.0 : released November 2014, major new features:
  - device-side enqueue (dynamic parallelism):
    kernels can add new work to device-side queues;
    clCreateCommandQueueWithProperties (host), enqueue_kernel (device)
  - SVM - shared virtual memory:
    host and OpenCL devices can share the same virtual address range;
    use shared pointers instead of copying buffers across devices;
    coarse-grain/fine-grain SVM; clSVMAlloc, clSetKernelArgSVMPointer
  - passing data between kernels using pipes:
    clCreatePipe (host), read_pipe, write_pipe (device)
  - new built-ins on workgroup/subgroup level (i.e. warps, wavefronts):
    e.g. work_group_all/any, _broadcast, _reduce
  - “generic” address space: a pointer to it can reference data in the private, local, or global address spaces
  - C11 atomics

OpenCL 2.x

- 2.1 : released November 2015
  - subgroups supported in OpenCL core (no longer as extension)
  - additional subgroup query operations
  - evolving to include a C++ kernel language:
    new OpenCL C++ kernel language based on C++ 14
  - SYCL: single source programming model
  - industry support for OpenCL 2.1:
    SPIR: Standard Portable Intermediate Representation
    SPIR-V: true cross-API, fully defined by Khronos
- 2.2 : released May 2017
  - OpenCL C++ kernel language static subset of C++ 14
  - new Khronos SPIR-V™ 1.2 intermediate language which fully supports the OpenCL C++ kernel language
  - pipe storage is new device-side type in OpenCL 2.2 that is useful for FPGA implementations
  - for the first time, released the full source of the OpenCL 2.2 specifications and conformance tests

SYCL

- C++ Single-source Heterogeneous Programming for OpenCL
- SYCL single-source programming enables host and kernel code to be contained in the same source file using the same templates for both, with full OpenCL acceleration
- seamless integration with OpenCL programs, C/C++ libraries and frameworks such as OpenMP
- includes templates and lambda functions for higher-level application software that can be cleanly coded for optimized acceleration

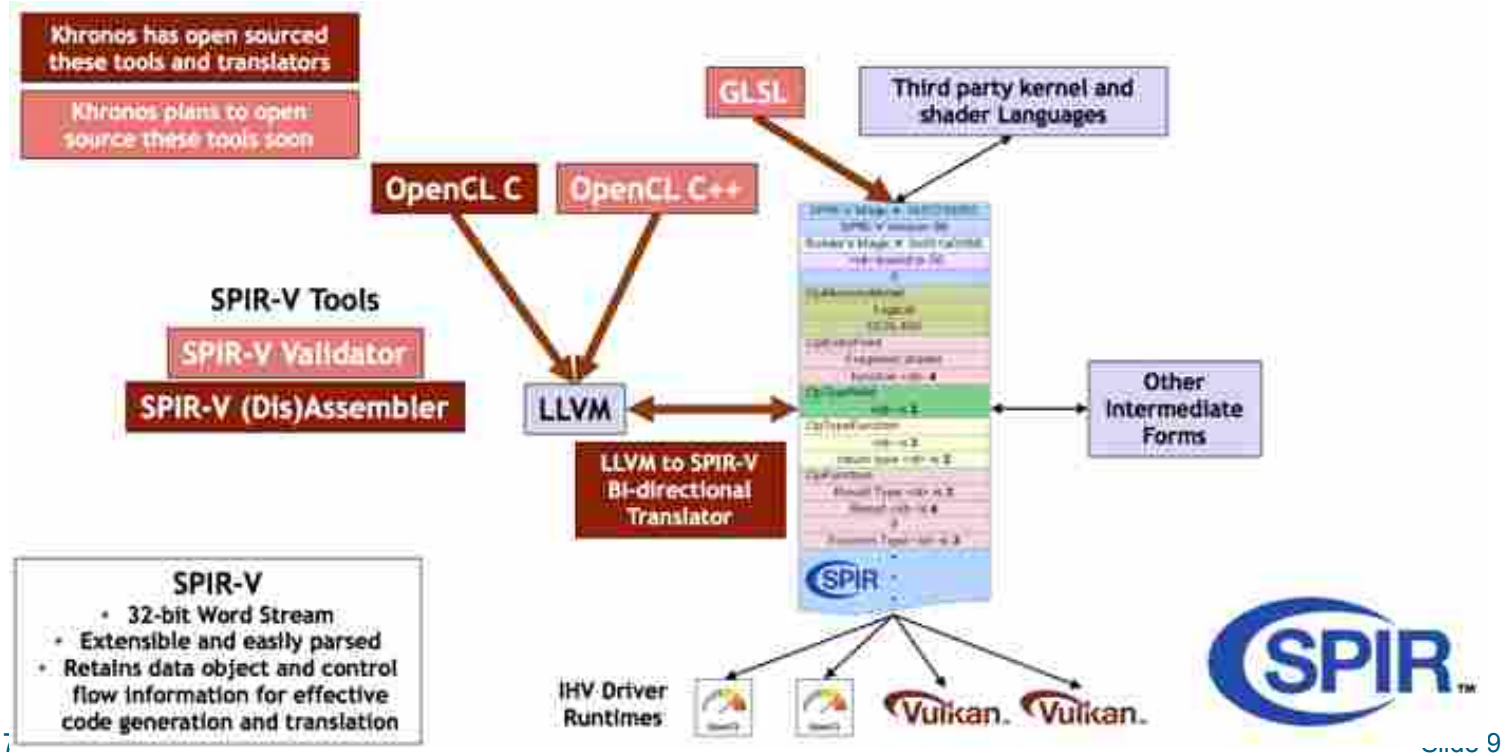


SYCL – Example Code

```
#include <CL/sycl.hpp>
int main ()
{
....
    // Device buffers
    buffer<float, 1 > buf_a(array_a, range<1>(count));
    buffer<float, 1 > buf_b(array_b, range<1>(count));
    buffer<float, 1 > buf_c(array_c, range<1>(count));
    buffer<float, 1 > buf_r(array_r, range<1>(count));
    queue myQueue;
    myQueue.submit([&](handler& cgh)
    {
        // Data accessors
        auto a = buf_a.get_access<access::read>(cgh);
        auto b = buf_b.get_access<access::read>(cgh);
        auto c = buf_c.get_access<access::read>(cgh);
        auto r = buf_r.get_access<access::write>(cgh);
       // Kernel
       cgh.parallel_for<class three_way_add>(count, [=](id<> i)
           {
               r[i] = a[i] + b[i] + c[i];
            })
        );
    });
....
}
```

SPIR - Standard Portable Intermediate Representation

- portable encoding of device programs; enable 3rd party code generation targeting OpenCL platforms without going through OpenCL
- SPIR 1.2 is an encoding of OpenCL C device programs in LLVM IR
- SPIR-V - true cross-API standard that is fully defined by Khronos with native support for shader and kernel features



AMD Boltzmann Initiative

- strategic investments in heterogeneous system architecture (HSA)
- suite of tools designed to ease development of high-performance, energy efficient heterogeneous computing systems
  - new compiler for Heterogeneous Computing (HCC)
  - Linux driver and runtime focused on the needs of HPC cluster-class computing
  - HIP-ifying CUDA applications

    translating CUDA source to run on AMD GPUs
- availability
  - an early access program for the "Boltzmann Initiative" tools is planned for Q1 2016

IWOCL 2017 Conference (Toronto, CA)

Advanced Hands-On OpenCL Tutorial

- Simon McIntosh-Smith and James Price, University of Bristol
- Based on the 2-day OpenSource course at: https://handsonopencl.github.io

Workshops and Talks

- Heterogeneous Computing Using Modern C++ with OpenCL Devices
- An Open Ecosystem for Software Programmers to Compute on FPGAs
- Harnessing the Power of FPGAs with the Intel FPGA SDK for OpenCL
- Towards an Asynchronous Data Flow Model for SYCL 2.2
- KART – A Runtime Compilation Library for Improving HPC Application Performance

IWOCL 2018

- 14-16 May, 2018, Oxford, UK
- http://www.iwocl.org/

References

- Khronos OpenCL
  - https://www.khronos.org/opencl
- IWOCL 2016 – International Workshop on OpenCL
  - 19-21 April 2016, Vienna
  - www.iwocl.org
- Portable Performance with OpenCL on Intel Xeon Phi
  - http://www.techenablement.com/portable-performance-opencl-intel-xeon-phi