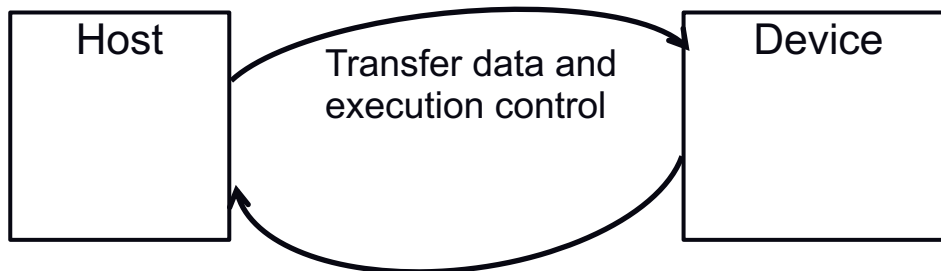# Aurora high-level Overview

- Hardware
  - Intel GPUs and CPUs
- Software (Intel One API umbrella):
  - Intel compilers (C,C++,Fortran)
  - Programming models: DPC++, OpenMP
  - Libraries: MKL, MKL-DNN,
  - Tools: VTune, Advisor
  - Python!

Three Pillars

Simulation	Data	Learning
HPC Languages	Productivity Languages	Productivity Languages
Directives	Big Data Stack	DL Frameworks
Parallel Runtimes	Statistical Libraries	Statistical Libraries
Solver Libraries	Databases	Linear Algebra Libraries

Compilers, Performance Tools, Debuggers

Math Libraries, C++ Standard Library, libc

I/O, Messaging

Containers, Visualization

Scheduler

Linux Kernel, POSIX

OpenMP 5

Host ⟷ Transfer data and execution control ⟷ Device

Distributes iterations to the threads, where each thread uses SIMD parallelism

Creates teams of threads in the target device
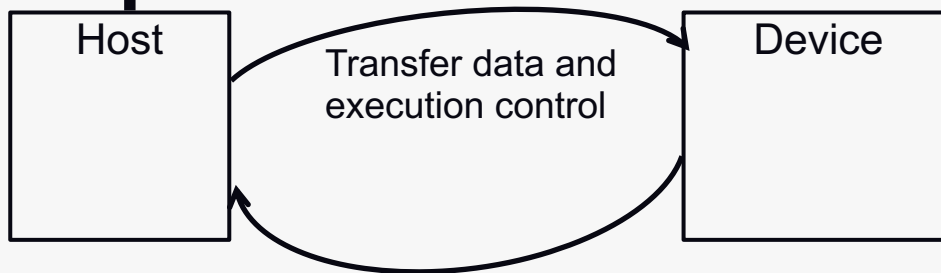
Controlling data transfer

```
extern void init(float*, float*, int);
extern void output(float*, int);
void vec_mult(float*p, float*v1, float*v2, int N)
{
  int i;
  init(v1, v2, N);
  #pragma omp target teams distribute parallel for simd \
                    map(to: v1[0:N], v2[0:N]) map(from: p[0:N])
  for (i=0; i<N; i++)
  {
      p[i] = v1[i]*v2[i];
  }
  output(p, N);
}
```

DPC++ Example

Host

Transfer data and
execution control

Device

Get a device

SYCL buffer
using host
pointer

Data
accessor

Kernel

Buffer out of
scope

```cpp
 default_selector selector; // Selectors determine which device to dispatch to.
{
  queue myQueue(selector); // Create queue to submit work to, based on selector

  // Wrap data in a sycl::buffer
buffer<cl_int, 1> bufferA(A, 1024);

  myQueue.submit([&](handler& cgh) {

    //Create an accessor for the sycl buffer.
    auto writeResult = bufferA.get_access<access::mode::discard_write>(cgh);

    // Kernel
    cgh.parallel_for<class hello_world>(range<1>{1024}, [=](id<1> idx) {
        writeResult[idx] = idx[0];
      }); // End of the kernel function
  }); // End of the queue commands
} // End of scope, wait for the queued work to stop.
```

VTune and Advisor

- Vtune Profiler
  - Widely used performance analysis tool
  - Currently supports analysis on Intel integrated GPUs
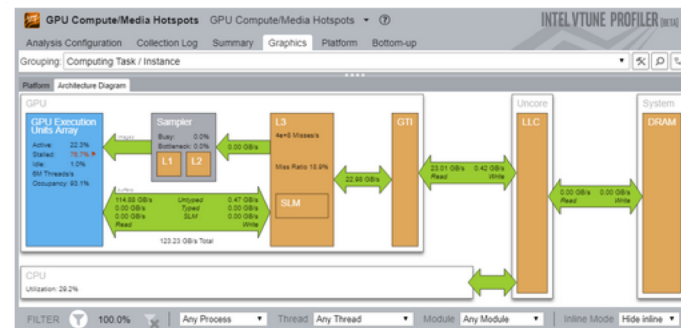 - Will support future Intel GPUs
- Advisor GPU Roofline features
  - Characterize offloaded kernels on GPUs
  - Visualize possible further optimizations
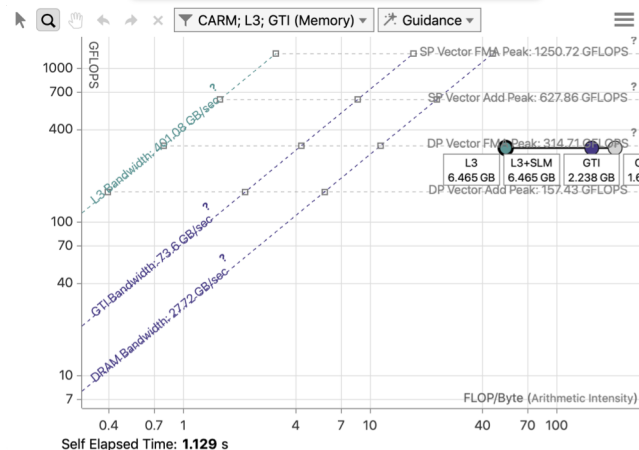 - Verify via manually counted performance data w/ micro-kernels
- Offload Advisor
  - Identify beneficial regions to offload
  - Support multiple programming models
  - C/C++/Fortran
  - DPC++, SYCL, OpenCL, OpenMP

Vtune example:

GPU Roofline on Gen9:

Data science

- Intel Distribution for Python
- Numba
- Deep learning
  - oneAPI Deep Neural Network Library (oneDNN)
  - Powers Tensorflow, PyTorch
- Classical Machine Learning Algorithms
  - oneAPI Data Analytics Library (oneDAL)
  - Easy to use one line daal4py Python interfaces
  - Powers Scikit-Learn

Data & Learning Examples

Projects preparing for Aurora

- Predicting & mitigating disruptions in fusion (for a clean energy source)
- Predicting drug responses
- Discovering singlet fission materials (for efficient solar cells)
- Mapping neurons and connections from brain data
- Determining possible interactions between particles (to understand dark matter)

More at: https://www.alcf.anl.gov/science/projects