Argonne NATIONAL LABORATORY

CUDA -> SYCL

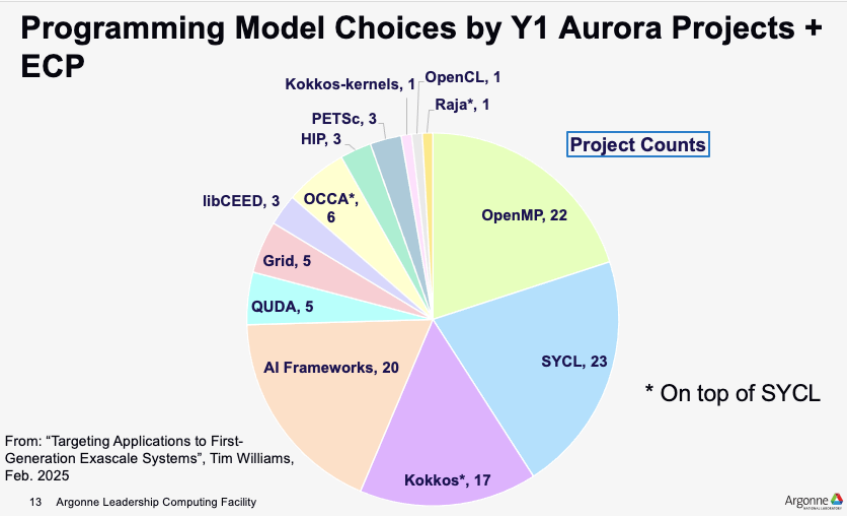
Thomas Applencourt, Abhishek Bagusetty

April 29, 2025

Table of Contents

Overview of SYCL

Is this thing really used?



Code Example

```
1
2  #include <sycl/sycl.hpp>
3
4  int main(int argc, char **argv) {
5    sycl::queue Q;
6    std::cout << "Running on "
7              << Q.get_device().get_info<sycl::info::device::name>() << "\n";
8
9    // Allocate Device Memory
10   int *A = sycl::malloc_device<int>(global_range, Q);
11   // Submit blocking kernel who use the memory
12   Q.parallel_for(global_range, [=](auto id) { A[id] = id; });
13   Q.wait();
14   // Allocate Host Memory
15   std::vector<int> A_host(global_range);
16   // Copy the device memory to the host
17   Q.copy(A, A_host.data(), global_range);
18   Q.wait();
19   sycl::free(A, Q);
20 }
```

In a nutshell

- At worse, just see SYCL as a ”nice”[1] C++ wrapper around CUDA API [2]
- Kernel code, will look similar[3]
- Run on Intel, NVIDIA, and AMD machine[4]

[1] As much as C++ syntax can be nice []();
[2] At least no state machine, far cleaner API than CUDA Runtime
[3] Abhi and Steve are far more knowledgeable than me
[4] In the order of love received by the backend

Difference / Similitude

- Stream == In Order Queue (default sycl queue out-of-order)
- Need to pass the ID to kernel
- Thread == Work-Item, Warp == Work-group
- Same synchronization primitive for kernel side (group barrier, shuffle, ...)
- Faster Indexing follow C++ (the reverse from CUDA – blame C++ / Nevin)
- Dependency via Event (DAG)
- Interopt with CUDA (getting native object, or mix and match)

The SYCL Working Group listened

Official Khronos (KHR) Extension

- Free Function Query (no need to pass the ID)
- Queue Empty query
- Graph Extension to lower latency
- printf (help wanted)

SYCLomatic and cO