What is Sycl

André Cerqueira

FCUP

2024

REPÚBLICA PORTUGUESA | CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

fct Fundação para a Ciência e a Tecnologia

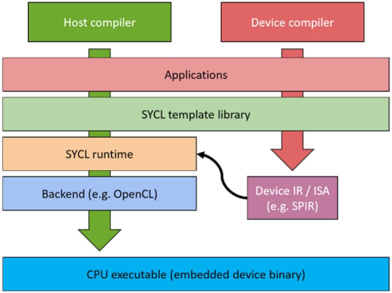
Este trabalho foi financiado por: Projeto 10110190 – EUROCC2, com apoio financeiro da FCT/MCTES através de fundos nacionais (PIDDAC).

What is Sycl

SYCL is a single source, high-level, standard C++ programming model, that can target a range of heterogeneous platforms

What is Sycl

SYCL is a **single source**, high-level, standard C++ programming model, that can target a range of heterogeneous platforms



- SYCL allows the programmer to write both host and device code in the same C++ source file.
- This requires two compilation passes; one for the host and one for the device code

What is Sycl

SYCL is a single source, **high-level**, standard C++ programming model, that can target a range of heterogeneous platforms

- Platform/device selection
- Dependency management and scheduling
- Buffer creation and data movement

What is Sycl

SYCL is a single source, high-level, **standard C++** programming model, that can target a range of heterogeneous platforms

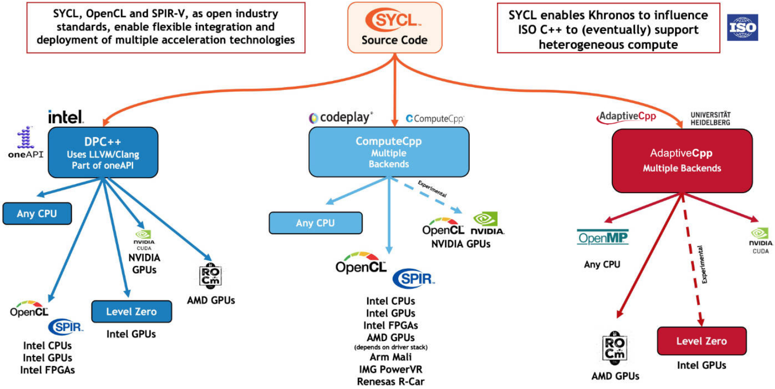
- Supports standard C++ features like:
  - Templates
  - classes
  - operator overloading
  - lambdas
- It's basically standard C++ code

What is Sycl

SYCL is a single source, high-level, standard C++ programming model, that **can target a range of heterogeneous platforms**

- Sycl can target any device supported by its backend
- Current implementations support backends such as:
  - OpenCL
  - CUDA (Nvidia)
  - HIP (AMD)
  - OpenMp
  - and others !

Sycl Implementations



SYCL, OpenCL and SPIR-V, as open industry standards, enable flexible integration and deployment of multiple acceleration technologies

SYCL Source Code

SYCL enables Khronos to influence ISO C++ to (eventually) support heterogeneous compute

Hello World

```
#include <iostream>
#include <sycl/sycl.hpp>
using namespace sycl;

const std::string secret{"Ifmmp-!xpsme\"\012J(n!tpssz-!Ebwf/!"
                         "J(n!bgsbje!J!dbo(u!ep!uibu/!.!IBM\01"};

const auto sz = secret.size();

int main() {
  queue q;

  char *result = malloc_shared<char>(sz, q);
  std::memcpy(result, secret.data(), sz);

  q.parallel_for(sz, [=](auto &i) { result[i] -= 1; }).wait();

  std::cout << result << "\n";
  free(result, q);
  return 0;
}
```

Let's Explore The Code

This Hello World introduces us to a number of fundamental concepts in SYCL:

Let's Explore The Code

- **Host and Device Code** are in the same source code
- Thanks to the implementation of unified shared memory, we are able to employ a pointer-based method for managing memory, which seamlessly operates on both the host and devices. (We will cover this topic again later).
- A **queue** is the system we use to coordinate tasks in the devices.
- **actions** are submitted to queues that then runs in the specified device. In the *Hello World* the action is **parallel_for**
- Inside actions we execute **Kernels**
- Actions are performed in an **asynchronous manner**. The host adds tasks to a queue and continues with its other responsibilities. In case we require the results of an action, we must patiently **wait** for its completion.