

Введение в OpenCL

Николай Пономарев

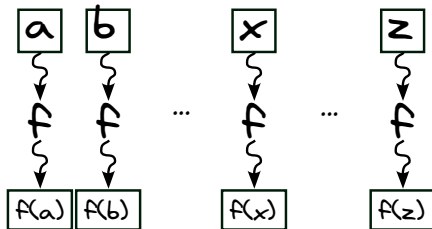
Математико-механический факультет СПбГУ

10 апреля 2025 г.

- OpenCL (Open Computing Language) for **Parallel** Programming of **Heterogeneous** Systems
- Один из стандартов The Khronos Group — консорциума, который занимается разработкой открытых стандартов в областях параллельных вычислений, машинного обучения, компьютерной графики
 - Насчитывает более 180 организаций-участников
 - Поддерживает 21 стандарт, среди них Vulkan, OpenGL, OpenCL, SPIR-V, WebGL
- Можно найти в девайсах от AMD, Arm, Google, Imagination Technologies, Intel, NVIDIA, Qualcomm, ...
- Используется в OpenCV, FFmpeg, ViennaCL, GNU Octave, Matlab, ...

Параллелизм по данным

- Можем применить функцию к различным элементам коллекции независимо¹



- Single Instruction Multiple Data (SIMD) — реализация параллелизма по данным на уровне процессора
 - Инструкция \Rightarrow действия более простые и гранулярные
 - Данные должны быть однородны
- Хорошо подходит для решения задач линейной алгебры и обработки изображений

¹Картинка из

https://github.com/YaccConstructor/articles/tree/master/2025/GPGPU_OpenCL_Intro

Устройства, где работает OpenCL

- Central Processing Unit, CPU
- Graphics Processing Unit, GPU
- Digital Signal Processor, DSP
- Field-Programmable Gate Array, FPGA
- Tensor Processor
- ...

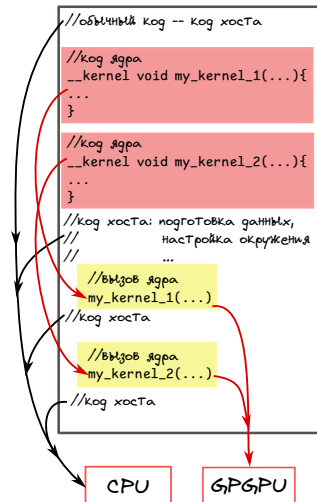
- 1.0** Первый релиз, 2008 г.
- 1.2** Baseline для современного оборудования, 2011 г.
- 2.x** Крупное обновление, но откатили в 3.0, 2013–2017 г.
- 3.0** База из 1.2 и расширения по усмотрению производителя, 2020 г.

Khronos часто показывает в примерах код для 2.x — но устройства почти не поддерживают эти версии!

Обычно в пользовательских устройствах можно найти либо 1.2, либо 3.0

Структура программы

- Программа выполняется на двух устройствах: хосте и ускорителе
- Хост (host) — обычно CPU устройства
 - Отвечает за общение с внешним миром, подготовку данных и запуск кода на девайсе
 - Можно использовать привычный язык программирования
- Ускоритель (девайс, device) — вычислительные модули, поддерживающие OpenCL
 - Исполняет код по просьбе хоста
 - Язык программирования «прибит»
 - Программу обычно называют «ядром» (kernel)



Взаимодействие хоста и ускорителя

- Обнаружение, подключение и конфигурация устройств
- Работа с памятью
 - Выделение/освобождение
 - Контроль доступа и синхронизация между хостом и ускорителем
 - Основной примитив — буфер, но с 2.0 поддерживается и Shared Virtual Memory (SVM)
- Выполнение команд
 - Передача данных, запуск ядер, синхронизация
 - Основной интерфейс взаимодействия — очередь команд
 - Неблокирующая: центральный процессор ставит задачи, но не дожидается их исполнения
 - Нужны дополнительные действия чтобы узнать, когда закончилась определённая задача
 - In-order: команды выполняются строго друг за другом
 - Может быть out-of-order
 - Синхронизация между несколькими очередями возможна, но требует отдельной работы

Языки программирования хоста:

- C
- C++
- Python²
- Rust³
- Julia⁴
- ...

Языки программирования ускорителя:

- **The OpenCL C Kernel Language**
- C++ for OpenCL

²PyOpenCL: <https://document.tician.de/pyopencl/>

³ocl: <https://github.com/cogciprocate/ocl>

⁴OpenCL.jl: <https://github.com/JuliaGPU/OpenCL.jl>

The OpenCL C Kernel Language

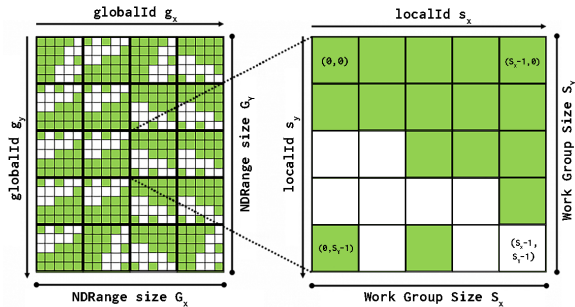
- Диалект C99: специальные типы данных (векторы, изображения, очереди команд), модификаторы памяти (глобальная, локальная, константная, приватная)
- Online компиляция, но можно и offline
- Собственный stdlib
- Расширения
 - Предусмотренные стандартом: `cl_khr_fp64`, `cl_khr_int64_extended_atomics`, ...
 - Расширения от вендоров: `cl_nv_pragma_unroll`, `cl_amd_fp64`, ...

Пример 1

См. директорию `cpp_vadd_example`

Параллельность

- Виртуальная решётка (**NDRange**) «поток» (**work-items**)
 - одно-, двух-, трёхмерная
 - Группировка узлов в рабочие группы (**work-group**) фиксированного размера
 - Возможность получить глобальные и локальные координаты «потока»
- Параметры (размеры) решётки связаны с «размерами» обрабатываемых данных

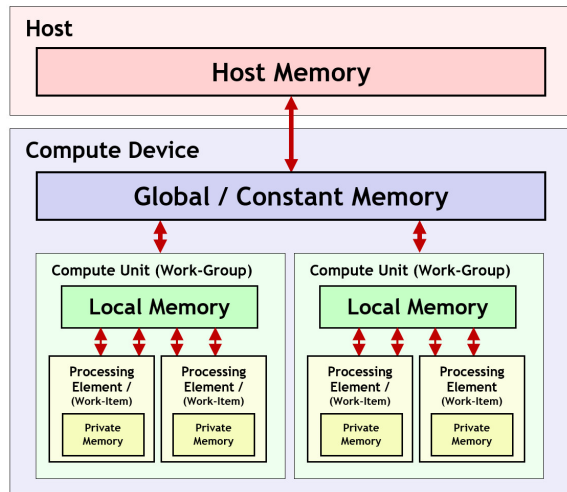


Модель памяти

Память в OpenCL выстраивается в иерархию:

- Память хоста — не доступна потокам напрямую
- Глобальная/константная память — доступна в любой части ускорителя
- Локальная — доступна группе «потоков»
- Приватная — доступна только «потоку»

Управление памятью ручное. Перемещение и синхронизация осуществляются явно, по команде программиста.



Пример 2

См. `myGEMM/src/kernels.cl`, ядро `myGEMM2`

И пояснения из <https://cnugteren.github.io/tutorial/pages/page4.html>

- OpenCL поддерживает изображения на уровне типов
- Для работы с ними имеются соответствующие API
- Информация о формате и размере изображения передаётся при загрузке в память
- Внутри ядра необходимо использовать сэмплеры — объекты, которые описывают, как извлекаются элементы изображения
 - какая используется координатная система (целочисленные или нормализованные координаты)
 - как обрабатывается ситуация выхода координат за пределы изображения (обнуление или установка «цвета» ближайшего элемента)
 - производится ли интерполяция при извлечении значений, расположенных «между» элементами изображения

Пример 3

См. директорию `cpp_gaussian_filter_example`

- Интеграция OpenCL и OpenGL
- Offline компиляция ядер
- Асинхронность
- Использование нескольких устройств

- Презентация Григорьева С. В.
 - https://github.com/YaccConstructor/articles/blob/master/2025/GPGPU_OpenCL_Intro/SemyonGrigorev_YWS_GPU_OpenCL.pdf
- Khronos OpenCL
 - <https://www.khronos.org/opencl/>
- Курс по разработке под GPGPU от EuroCC National Competence Center Sweden
 - <https://enccs.github.io/gpu-programming/>
- Cornell University, Understanding GPU Architecture
 - <https://cvw.cac.cornell.edu/gpu-architecture>
- Антонюк В. А. OpenCL Открытый язык для параллельных программ, учебное пособие, 2017.
 - <https://istina.msu.ru/publications/book/76785028/>
- Tutorial: OpenCL SGEMM tuning for Kepler
 - <https://cnugteren.github.io/tutorial/pages/page1.html>
- OpenCL Programming Guide 1.2 Examples
 - <https://github.com/bgaster/opencl-book-samples>

- ① OpenCL: общие сведения, поддерживаемый вид параллелизма, поддерживаемые устройства, языки программирования
- ② OpenCL: структура программы, взаимодействие хоста и ускорителя, рабочие группы и модель памяти