OpenCL API

GPU programming

OpenCL API


UNIVERSITAS BUDAPESTINENSIS DE ROLANDO EÖTVÖS NOMINATA
1635


OpenCL

OpenCL basics (#1)

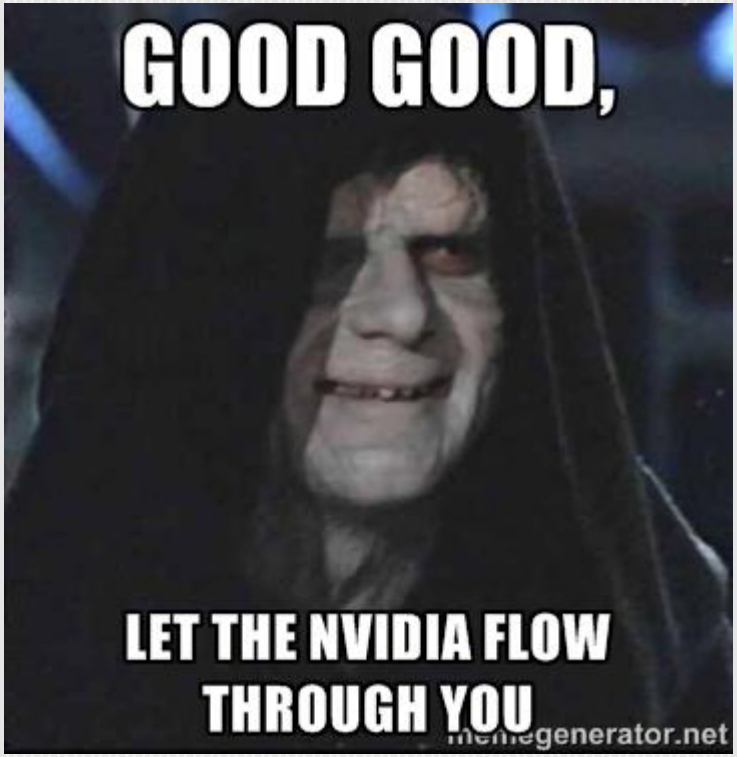
- Data- and task-parallel models
- Az OpenCL open standard
  - Khronos Group
- OpenCL-C programming language
  - subset of ISO C99 standard
- Numeric operations: IEEE754
- Heterogeneous computing platofrm
  - GPU, CPU, Cell processor, DSP, Intel Xenon Phi, Altera FPGA, ...

OpenCL basics (#1.1)

- **(CPU / GPU)**
  - AMD (OpenCL >v2.0)
  - ARM (OpenCL >v2.0)
  - Intel (OpenCL >v2.0)
  - NVIDIA (OpenCL v1.2)
  - …
  - (Android)

We'll only use OpenCL v1.2 ☹.

OpenCL basics (#1.1)



We'll only use OpenCL v1.2 ☹.

OpenCL basics (#2)

- Parts of OpenCL
  - Platform model
    - The connection of the Host and Device
  - Program model
    - Options for Data- and Task-parallelism
  - Execution scheme
  - Memory model

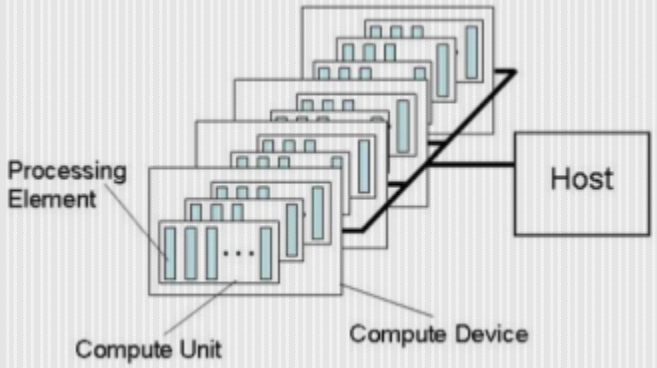
OpenCL basics (#2)

- Parts of OpenCL
  - Platform model
    - The connection of the Host and Device
  - Program model
    - Options for Data- and Task-parallelism
  - Execution scheme
  - Memory model

OpenCL basics (#3)

Platform model

- Host device (**Host**)
- OpenCL device (Compute Device, „**Device**”)
- Compute Unit („CU”)
  - E.g. NVidia cards' multiprocessor
- Processing Element („PE”)
  - E.g. Graphics card’s Stream processor
  - E.g. CPU core



OpenCL basics (#4)

- Parts of OpenCL
  - Platform model
    - The connection of the Host and Device
  - Program model
    - Options for Data- and Task-parallelism
  - Execution scheme
  - Memory model

OpenCL basics (#4)

Program model

- Options for Data- and Task-parallelism
- Data-parallel model
  - Joining data and task
  - Executing a set of instructions for multiple data
  - Automatic distribution/execution of tasks
- Task-parallel model
  - Parallel execution of multiple independent tasks

OpenCL basics (#2)

- Parts of OpenCL
  - Platform model
    - The connection of the Host and Device
  - Program model
    - Options for Data- and Task-parallelism
  - Execution scheme
  - Memory model

OpenCL basics (#5)

Execution scheme

- **Host**
  - Managing the context
  - Managing execution
- **Kernel** program
  - Managing CU-s
  - Managing/executing the same task in a Work Group

OpenCL basics (#5)

Execution scheme

- **Kernel** program example (***OpenCL C***)

```
__kernel void dataParallel(
        __global float* A, // Input data 1
        __global float* B, // Input data 2
        __global float* C) // Output data
{
        // Work-item identifier, element of the ND-Range
        int base = 4*get_global_id(0);
        C[base+0] = A[base+0] + B[base+0];
        C[base+1] = A[base+1] - B[base+1];
        C[base+2] = A[base+2] * B[base+2];
        C[base+3] = A[base+3] / B[base+3];
}
```

OpenCL basics (#5.1)

Execution scheme

- Kernel program
  - Work-Items
    - Global identifier (global ID)
    - Same program for each work group
    - Execution may vary from unit to unit
  - Work Groups
  - NDRange

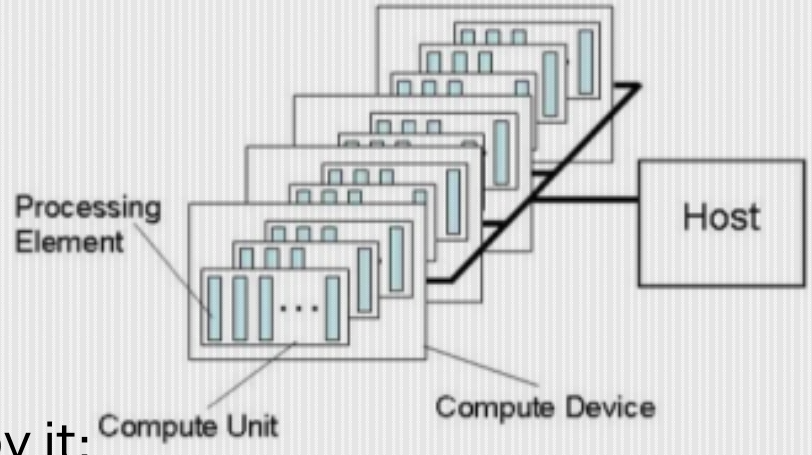
OpenCL basics (#5.2)

Execution scheme

- Kernel program
  - Work-Items
  - **Work Groups**
    - Work Group identifier (work-group ID)
    - Local identifier (local ID)
  - NDRange

OpenCL basics (#5.3)
Execution scheme

- Kernel program
  - Work Groups
  - Work-Items
  - **NDRange**
    - N-dimensional grid
      N = 1, 2, 3
    - The following are specified by it:
      - Global space for indexing
      - Work Group size
    - Indentifier / indexing in N dimensions:
      - Global ID [e.g. get_global_id(1)]
      - Local ID [e.g. get_local_id(0)]



OpenCL basics (#5.4)

Execution scheme

- Context (**Context**)
  - Device (Device)
  - Kernels (OpenCL functions)
  - Program objects (Program)
    - Source
    - Executable binary
  - Memory objects
    - Memory used by the host and the device

OpenCL basics (#5.5)

Execution scheme

- Command queues (**command-queue**)
  - Managed by the host
  - Handles memory operations
  - Handles execution of the kernels
  - Synchronization
  - In-order / Out-of-order modes for execution

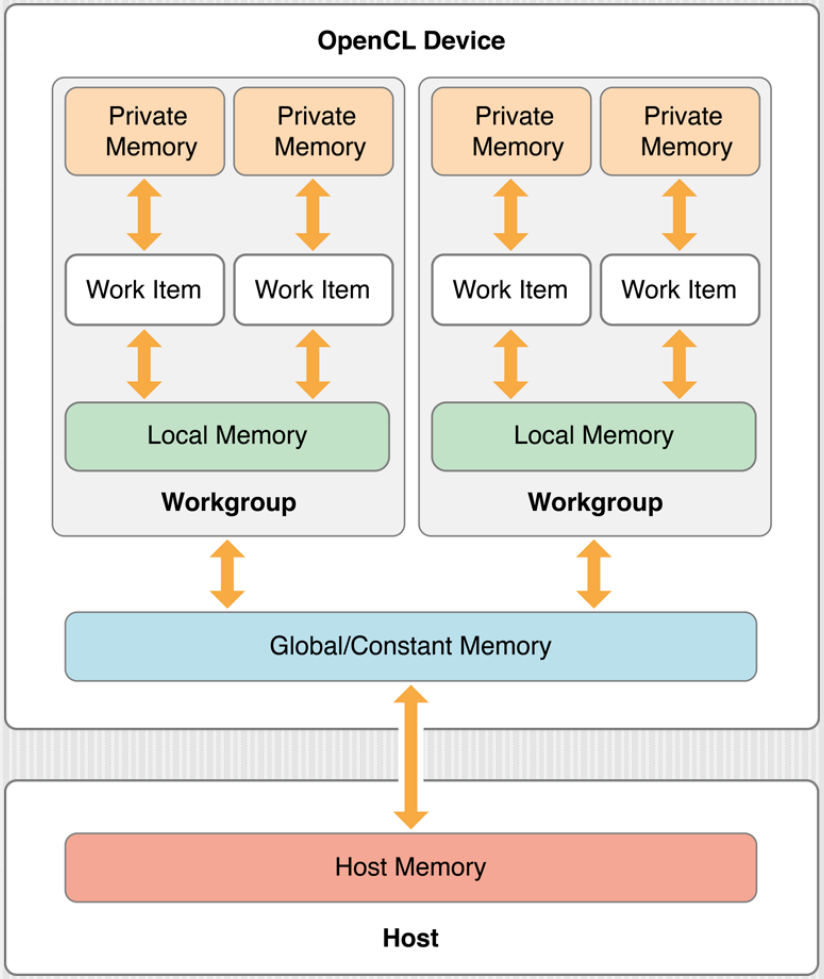
OpenCL basics (#6)

- Parts of OpenCL
  - Platform model
    - The connection of the Host and Device
  - Program model
    - Options for Data- and Task-parallelism
  - Execution scheme
  - Memory model

OpenCL basics (#6)

Memory model

- For memory regions on the device
  - Global
  - Constant
  - Local
  - Private



OpenCL basics (#6.1)

Memory model

- Global memory
  - Any workitem can read/write it
  - == Any PE can access it
  - The Host…
    - Allocates,
    - Executes copy operations,
    - Deallocates.

OpenCL basics (#6.2)

Memory model

- Constant memory
  - ~ Global memory, but read-only
  - Can be statically defined in a Kernel
  - Some devices might have designated harware for constant memory.

OpenCL basics (#6.3)
Memory model

- Local memory
  - The Host cannot access it
  - Shared memory of a WG
    - Any WI can read/write it in the same WG
- Private memory
  - The Host cannot access it
  - Only a work-item can access it

OpenCL basics (#6.4)

Memory model

- Consistency
  (Where do memory operations interfere?)
  - At Work-Item level?
    - Between WI-s not consistent,
    - Consistent in a WI.
  - At Work-Group level?
    - Local and Global memory is consistent in a WG
    - Global memory is not consistent between WG-s

OpenCL basics (#7) Synchronization

- Work Group Synchronization
  - Synchronizing WI-s
  - Barrier
    - Blocking operation
  - Between WG-s: no synchronization!!!!
- CommandQueue synchronization
  - Command Queue **Barrier**
    - the execution of pre-barrier commands is guaranteed
    - No sync between CQ
  - Waiting for an **Event**
    - Any CQ operation can generate an Event
    - …

OpenCL basics (#8)

OpenCL C programming language

OpenCL basics (#8)

OpenCL C

- C99 language modified
- Scalar types
- Vector types (n ∈ {2,4,8,16})
  - (u)charn
  - (u)shortn
  - (u)intn
  - (u)longn
  - floatn
- Accessubg vector components (e.g. float4 f)
  - Swizzle operators (f.xyzw, f.x, f.xy, f.xxyy, stb.)
  - Numerical indexing (f[i])
  - Halving (f.odd, f.even, f.lo, f.hi)

OpenCL basics (#8.1)

OpenCL C

- Implicit conversion
  - Limited use; between scalar types
- Explicit conversion (few examples)
  - `float4 f = (float4)1.0;`
  - `uchar4 u;`
    `int4 c = convert_int4(u);`
  - `float f = 1.0f;`
    `uint u = as_uint(f); // 0x3f800000 is the result`

OpenCL basics (#8.2)

OpenCL C

- Attributes on memory objects
  - __global, __local, __constant, __private
    - Example:
      __global float4 color;
- Attributes on functions
  - __kernel
    An OpenCL C function is denoted as a Kernel.
  - __attribute__
    Attributes for the compiler…

OpenCL basics (#8.3)

OpenCL C

- Built-in functions related to the exec. scheme
  - get_work_dim()
  - size_t get_{global/local}_{id/size}(uint dimIdx);
    - E.g. size_t id = get_global_id(1);
  - size_t get_num_groups(uint dimIdx);
  - size_t get_group_id(uint dimIdx);

OpenCL basics (#8.4)

OpenCL C

- Synchronization instructions
  - barrier(flag);
    - CLK_LOCAL_MEM_FENCE : consistency on local memory
    - CLK_GLOBAL_MEM_FENCE : … on global memory
  - mem_fence(flag);
  - write_mem_fence(flag);
  - read_mem_fence(flag);

OpenCL basics (#8.5)

OpenCL C

- Additional built-in functions
  - General, regular functions
  - Geometric functions
  - Comparison functions on floatn types
    - (isequal, isinfinite, etc.)
  - Regarding memory:
    - Asynchronous memory read
    - Prefetch (load stuff to cache from global memory)

A few software using OpenCL

- VexCL
  - C++ template library for computing vector expressions, using OpenCL/CUDA
  - https://github.com/ddemidov/vexcl
- HadoopCL
  - http://pubs.cs.rice.edu/node/336
  - MapReduce on heterogeneous systems, with Hadoop - OpenCL integration
- Apple OS X Snow Leopard
  - http://en.wikipedia.org/wiki/Mac_OS_X_Snow_Leopard#OpenCL
- OpenCV
  - An open source computer vision and machine learning software library. Some of its functions are OpenCL-enabled.
  - https://opencv.org/
- …