OpenCL

Contents of Lecture 12

- GPUs
- OpenCL
- About the exam

GPUs

- The term GPU was coined by Nvidia in 1999 when their GeForce 256 was launched
- It was a one chip processor for transformations, lighting, and triangle setup and clipping
- Performance was 10 million polygons per second
- Their competitor ATI launched Radeon 9700 in 2002.
- These early GPUs where not programmable but had APIs used by the host computer
- Modern GPUs are programmable e.g. in OpenCL — based on C

Differences between CPUs and GPUs

- CPUs have higher clock frequency compared with GPUs
- GPUs have many many more cores (e.g. hundreds or thousands)
- GPUs are good at SIMD processing
- GPUs cannot run OSes — no interrupt mechanism
- CPUs are better at control (such as if-then-else)
- CPU instruction sets are published while those of GPUs' are not
- GPU instructions can change between chip generations
- Intel makes integrated GPUs — on the same chip as the CPUs
- In Intel Core M, the GPU use about 60 % of the chip area
- CPUs and GPUs can complement each other

OpenCL

- OpenCL originates from Apple and is an Apple trademark
- Can e.g. be used with a C/C++ program on a host computer
- The purpose of OpenCL is to
  - Let software exploit all parallel hardware on a machine: CPUs, GPUs, DSPs etc
  - Provide a language that vendors agree on
  - Virtual memory works in compute devices such as GPUs — with OpenCl 2.0
  - Make software portable! (impossible with this hardware support?)
- How can this be achieved?

Portability despite different ISAs

- Due to different instruction set architectures, OpenCL programs are compiled at runtime!
- This costs some execution time (obviously) but achieves portability
- So each vendor provides an OpenCL-compiler and a runtime library.
- OpenCL is developed through an industry consortia called the Khronos Group

OpenCL

Based on C99

- No pointers to functions
- No bit-fields
- No variable-length arrays
- No structs
- New keyword `__kernel`
- A kernel is a function which should be executed by some compute engine

OpenCL Hardware Model

- Host — CPU
- Compute device — e.g. a GPU
- A compute unit — part of a compute device
- A processing element — part of a compute unit

An OpenCL Application

- Host code + device code
- Host tells device to transfer data to/from memory
- Host tells device to execute code
- An application consists of serial (only host) and parallel parts (run at devices)

Work items

- An N-dimensional domain is defined
- A ***kernel*** is executed on each point in the domain
- A kernel is a short function

```
__kernel void mul(
        __global const float* a,
        __global const float* b,
        __global float* c)
{
        int     id;

        id = get_global_id(0);
        c[id] = a[id] * b[id];
}
```

Execution Model

- A work-item is executed by a kernel function
- A ***context*** is the environment where a work-item is executed
  - device
  - device memory
  - command queue
- A ***command queue*** is used by the host to submit work to a device

Memory

- Private memory — per work item
- Local memory — shared within a work group
- Global/Constant memory — visible to all work groups
- Host memory — system RAM
- Programmer is responsible for transferring data

OpenCL language features

- Scalar types from C
- `image2d_t`, `image3d_t`, `sampler_t`
- Various vector types
- Memory fence — as in C11
- Memory barrier — all wait here

Kernel compilation

- A `cl_program` contains source code of kernels and the most recent successful build
- `clBuildProgram` performs the compilation
- Kernel source can be simply a string
- After setting up arguments, the code is sent to a device queue for execution

About the exam

- See reading advice on course page — I will update it with page numbers for Amazon printing
- There can be questions related to the labs and lecture notes (obviously)
- I will not about details such as language syntax
- There will be no questions where you will write code
- The questions are about "principles" and some "facts"
- What is meant by release and acquire belongs to "principles"
- I will not ask you to do dependence analysis but may ask about how a certain loop might be executed in parallel...
- Max 60 p, grade = $\lfloor score/10 \rfloor$
- There will be one simple question about history (2p).
- The group winning the competition will get coffee mugs at the exam.