GPGPU: Terminology and Examples

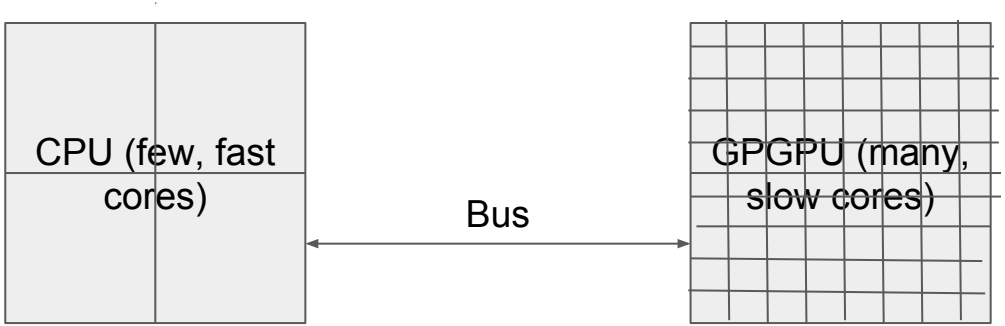
Aaron Weeden
Shodor Education Foundation, Inc.
2015

Review

It is recommended to first review the slides on OpenMP and MPI. They cover some key terms that will be used in these slides.

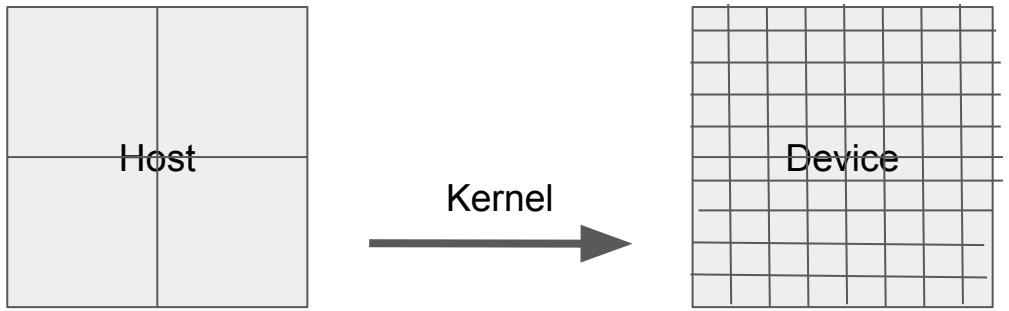
Key term: **GPGPU**

- General Purpose Graphics Processing Unit.
- Graphics card used for number crunching.
- “Massively parallel” - thousands of cores in a single device.
- Each core is not very powerful compared to a CPU core.
- Connects to the CPU using a high-speed bus.
- Has its own RAM.



Key terms: **Kernel**, **Host**, **Device**

- **Kernel**: Function executed in parallel by the cores of a GPGPU.
- **Host**: CPU that sends kernels to a GPGPU to execute.
- **Device**: GPGPU that executes the kernels.



Key terms: **Thread**, **SM**, **Warp**, **Block**, **Grid**

- **Thread**: runs instructions on a core of the GPGPU.
- **Streaming Multiprocessor (SM)**: group of cores.
- **Warp**: group of (usually 32) threads that execute in parallel on a single SM (see analogy: http://en.wikipedia.org/wiki/Warp_%28weaving%29).
- **Block**: collection of warps (1D, 2D, or 3D); all threads in a block share memory. Each block executes on a single SM. Multiple blocks may execute on the same SM.
- **Grid**: collection of blocks (1D or 2D). Blocks do not share memory, but they can all access the global GPGPU memory. All blocks in a grid have the same size and shape (i.e. how many threads per block in the x, y, and/or z dimensions).

GPGPU Example: **Forest Fire Model**

- Same basic model as serial version, with a few differences (shown below in blue).
- Data
  - Trees
  - NewTrees
- Tasks
  - Create copies of Trees and NewTrees on the device.
  - **InitData**: Launch a kernel on the device to light the center tree on fire.
  - For each time step:
    - **ContinueBurning**: Launch a kernel on the device to check for trees already burning that haven't burnt out, and burn those trees another step.
    - **BurnNew**: Launch a kernel on the device to check for trees next to burning neighbors, and catch those trees on fire with some probability.
    - **AdvanceTime**: Launch a kernel on the device to copy NewTrees into Trees.

GPGPU Example: **Forest Fire Model**

- Data needs to be created on the device at the beginning for **Trees** and **NewTrees**.
- Data needs to be copied from the device to the host:
  - If a visualization is being generated, **NewTrees** needs to be copied at each time step.
  - At the end of the simulation, the number of burning trees needs to be copied.
- Data does not need to be copied from the host to the device.

CUDA

- API for GPGPU parallelism.
- Not directive based -- uses function calls.
- Examples of basic functionality:
  - Allocate device memory (**`cudaMalloc`**).
  - Copy memory from host to device (**`cudaMemcpy`**).
  - Execute kernels on a device.
  - Copy memory from device to host (**`cudaMemcpy`**).
  - Deallocate device memory (**`cudaFree`**).
- Example kernel syntax:

```
functionName<<<BlocksPerGrid, ThreadsPerBlock>>>(args);
```

OpenACC

- API for GPGPU parallelism.
- Directive based -- similar syntax to OpenMP.
- Syntax example: execute iterations of a loop in parallel on a GPGPU:

```
#pragma acc parallel loop
for (i = 0; i < N; i++) {
}
```

Blue Waters key terms: **XE** and **XK** nodes

- **XE node**: 32 CPU cores, no GPGPU.
- **XK node**: 16 CPU cores, 1 GPGPU (NVIDIA “Kepler”).

Key Term: **Weak Scaling**

- Like strong scaling, increase the number of processes or threads and observe the effect on the run time.
- Unlike strong scaling, also increase the size of the problem along with the number of processes or threads. The amount of work per process/thread stays constant.
- Example for CUDA (blue) and OpenACC (red): number of threads in the x-axis, run time in the y-axis, 1 tree per thread, problem size is square root of number of threads: