Data parallelism

General Purpose GPU Programming

Supercomputer hidden in the GPU

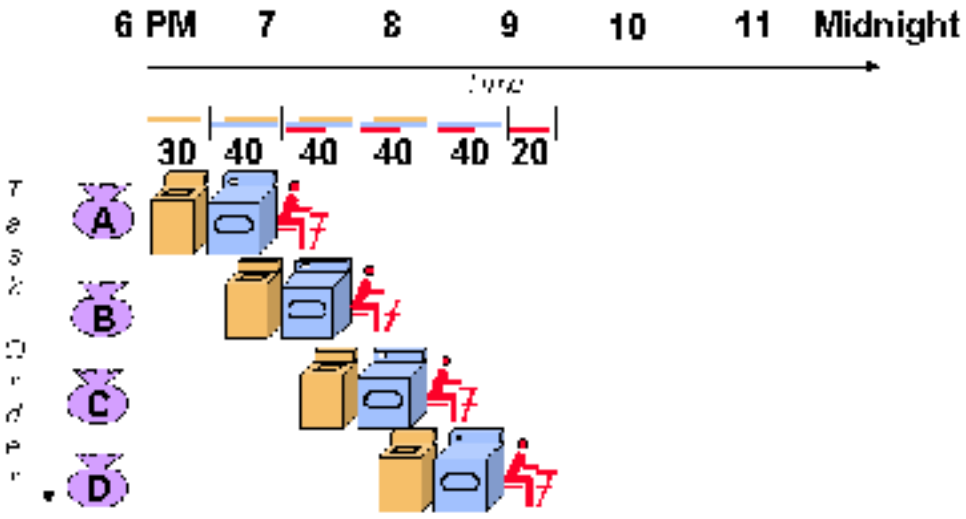
- Computer graphics is all about manipulating huge amounts of data…
- …but the actual operations on that data are relatively simple vector or matrix operations

⇩

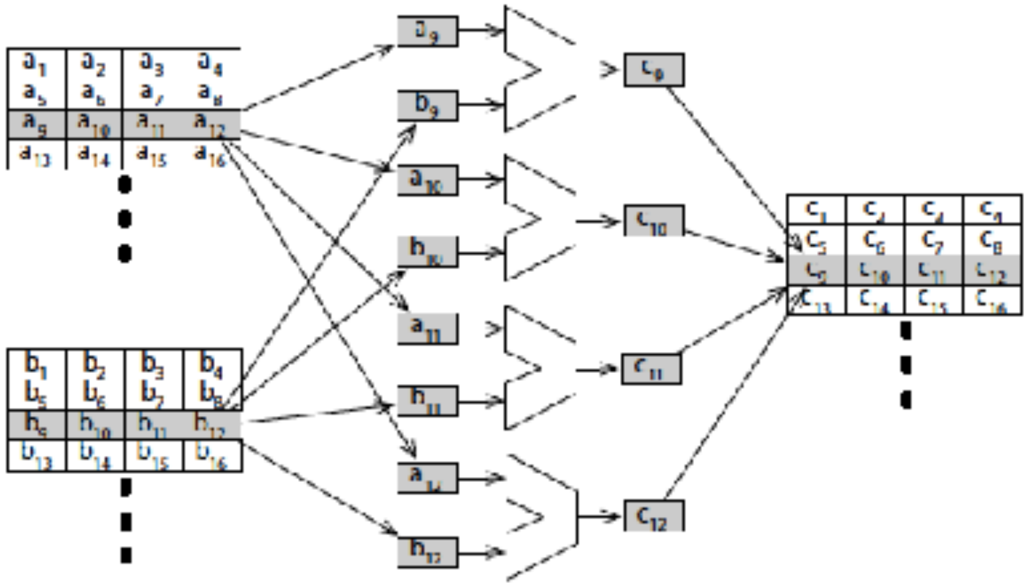
Data Parallelization

2 ways of Data parallelism

- Pipelining



- Multiple ALUs (with wide memory bus)



GPU architecture

- GPUs use pipelining, multiple ALUs and other techniques

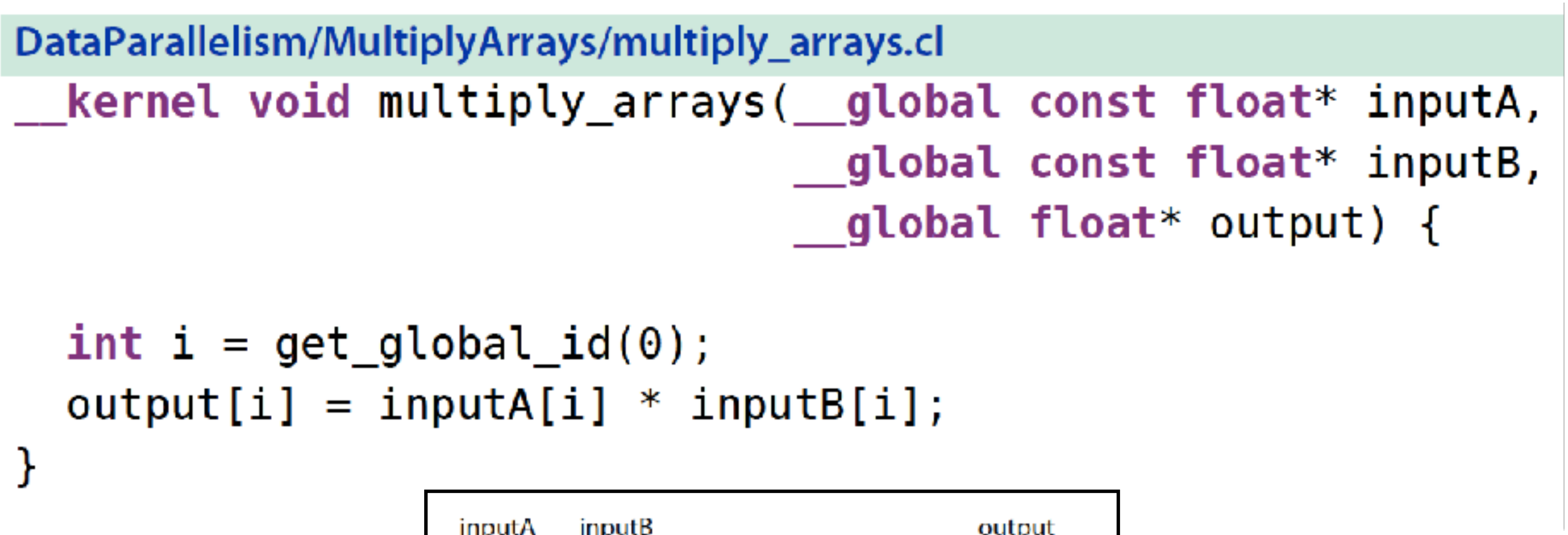
- Different architecture for every GPU

- **OpenCL** targets multiple architectures by defining a C-like language that allows us to express a parallel algorithm abstractly

OpenCL programming

- The task of the **programmer** is to divide the problem into the smallest work-items he can.
  - **Kernel**: specifies what each work-item has to do
- The **OpenCL compiler** and runtime then worry about how best schedule those work-items on the available hardware so that that hardware is utilized as efficiently as possible

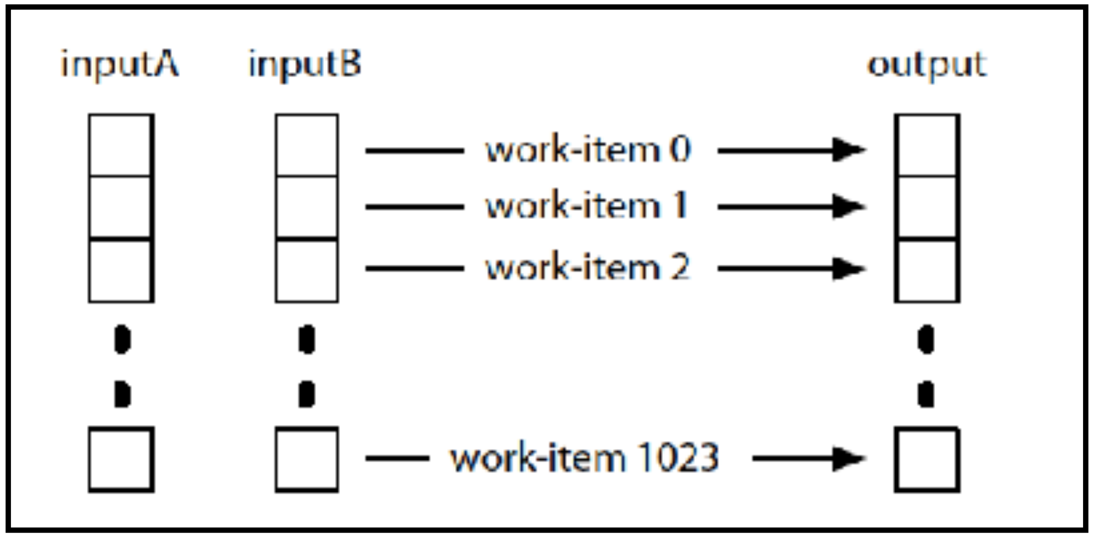
Example: array multiplication



DataParallelism/MultiplyArrays/multiply_arrays.cl

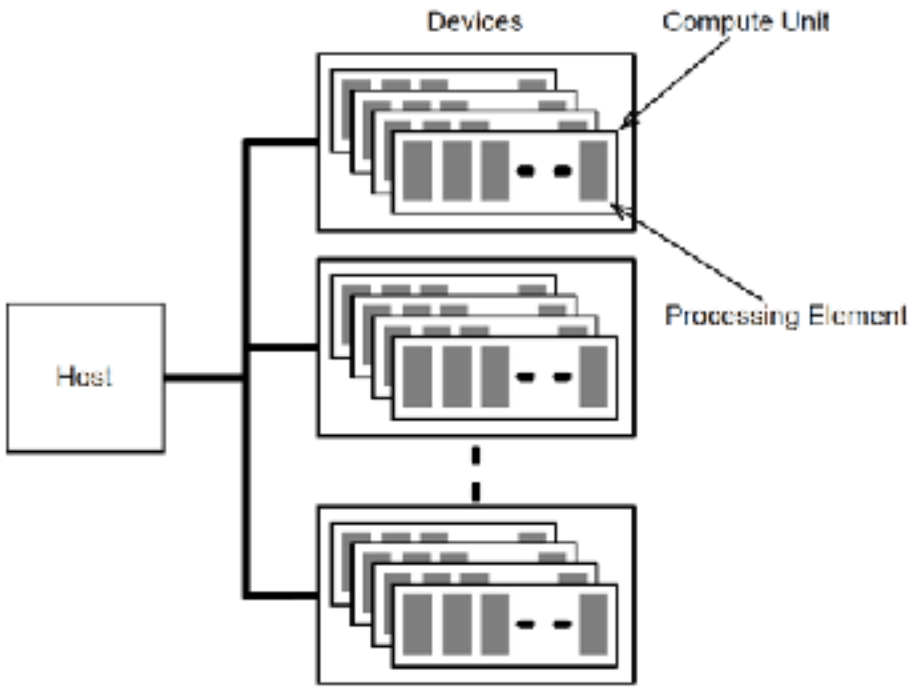
```
__kernel void multiply_arrays(__global const float* inputA,
                              __global const float* inputB,
                              __global float* output) {

  int i = get_global_id(0);
  output[i] = inputA[i] * inputB[i];
}
```



OpenCL platform model

- Each device has one or more **compute units**, each of which provides some **processing elements**



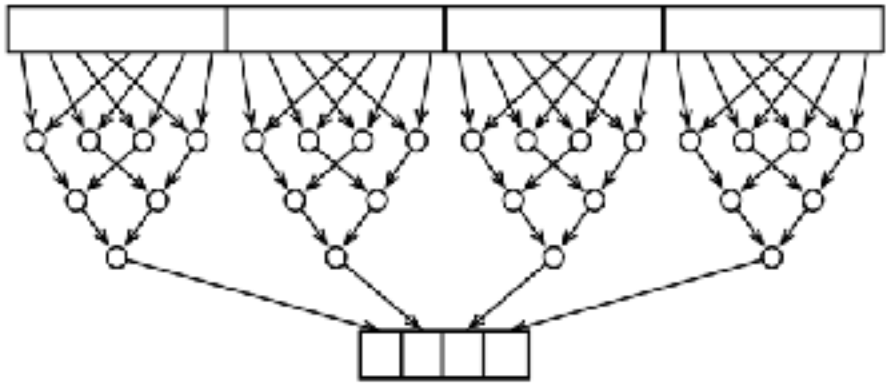
- **Work-items** execute on processing elements. A collection of work-items executing on a single compute unit is a **work-group**

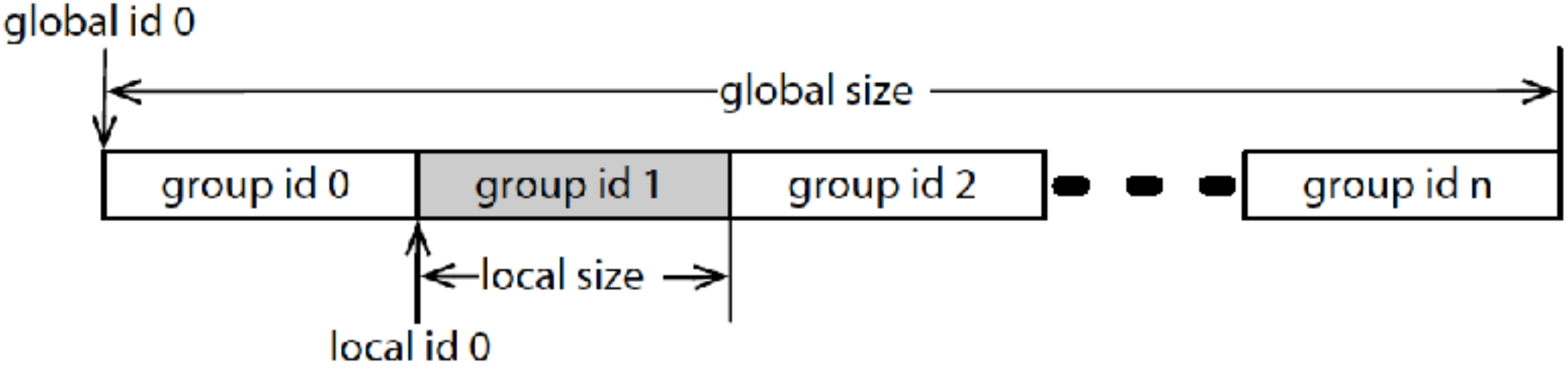
Memory model

- **Global memory**: Memory available to all work-items executing on a device
- **Local memory**: Memory local to a work-group
  - communication between work-items executing in a work-group (e.g. barrier)

How big is a work-group

- Size of work-groups is variable
  - Solution: Break the problem into sub-problems





Conclusions

- Data parallelism is ideal whenever you’re faced with a problem where **large amounts of numerical data** needs to be processed
- The runtime helps to work with different architectures
- The programmer's task is to model the problem in order to make it parallelizable