

# **OpenCL**

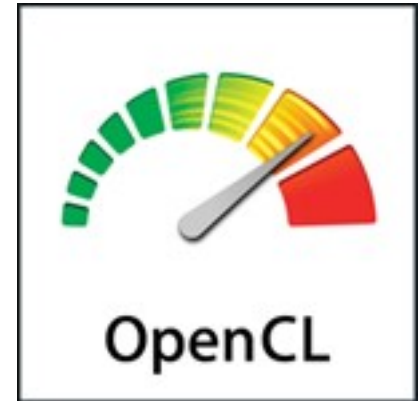
**CSCI 4239/5239**

**Advanced Computer Graphics**

**Spring 2021**

# What is OpenCL

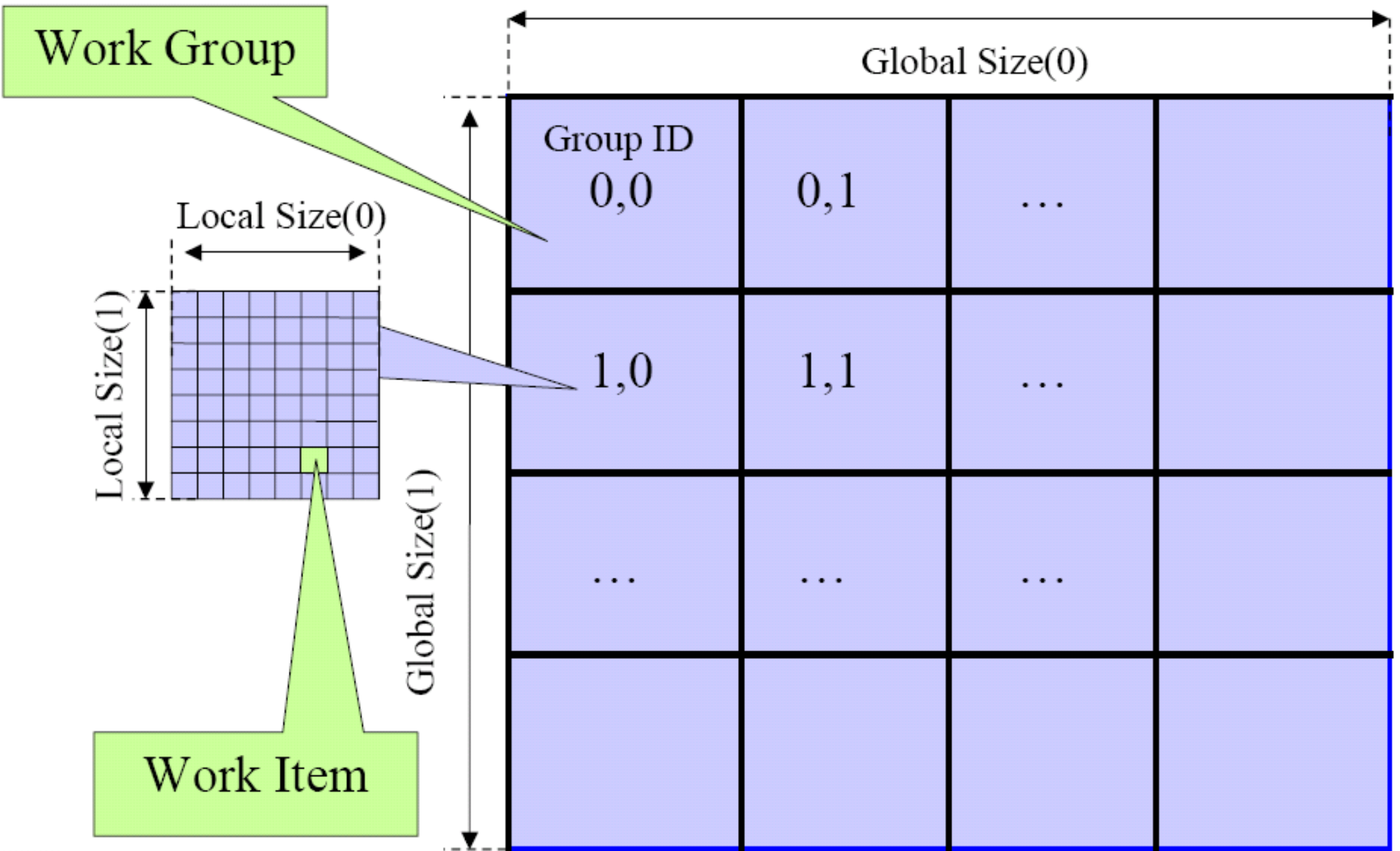
- **Open Computing Language**
  - Initially developed by Apple
  - Now managed by Khronos
- **Parallels CUDA**
  - Not just NVIDIA hardware
  - Supports CPU, GPU and Accelerators
  - Conceptually the same, API and syntax different
    - Using OpenCL a bit more tedious than CUDA



# OpenCL to CUDA Data Parallelism Model Mapping

OpenCL Parallelism Concept	CUDA Equivalent
kernel	kernel
host program	host program
NDRange (index space)	grid
work item	thread
work group	block

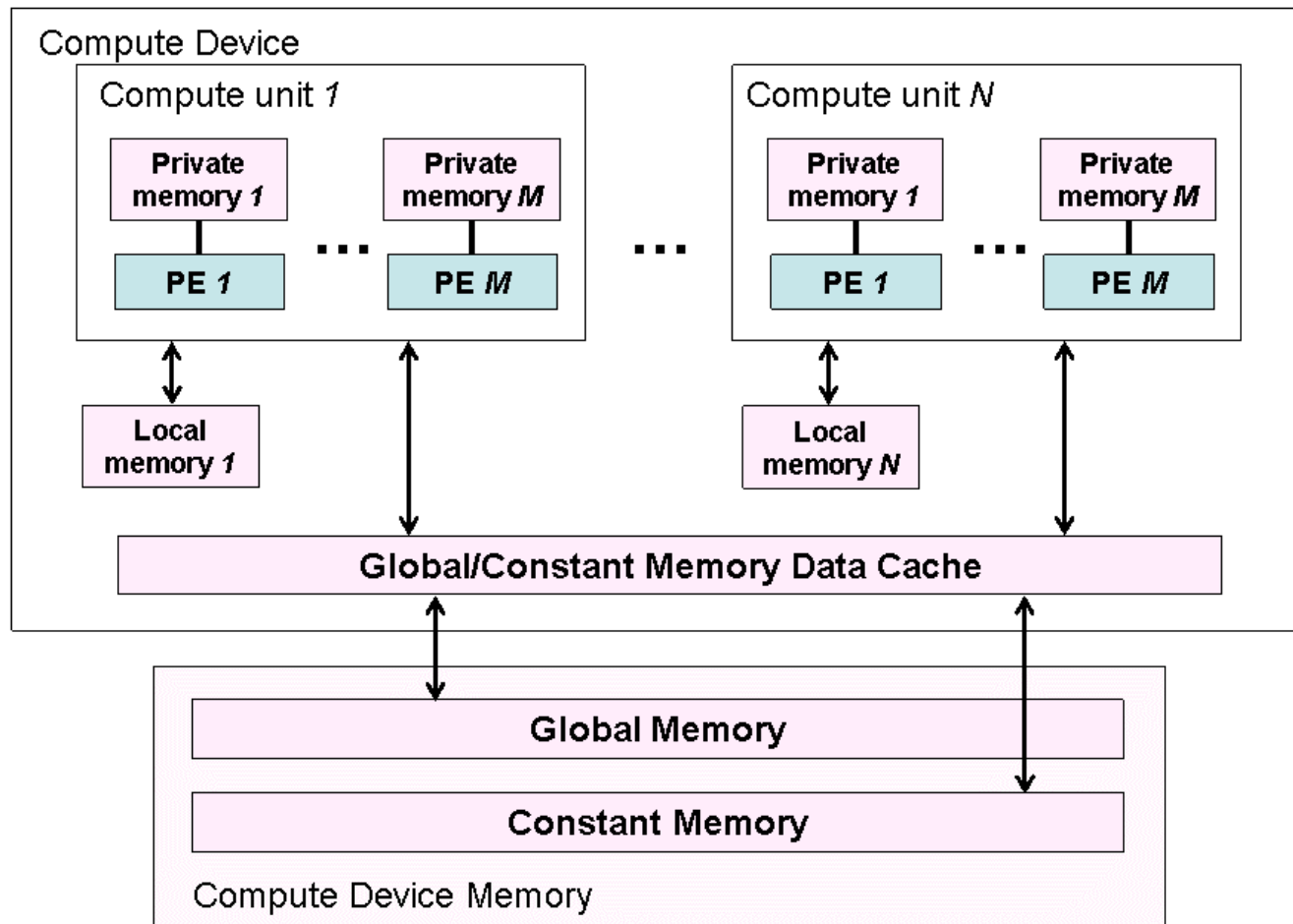
# Overview of OpenCL Execution Model



# Mapping of OpenCL Dimensions and Indices to CUDA

OpenCL API Call	Explanation	CUDA Equivalent
<code>get_global_id(0);</code>	global index of the work item in the x dimension	$\text{blockIdx.x} \times \text{blockDim.x} + \text{threadIdx.x}$
<code>get_local_id(0)</code>	local index of the work item within the work group in the x dimension	$\text{blockIdx.x}$
<code>get_global_size(0);</code>	size of NDRange in the x dimension	$\text{gridDim.x} \times \text{blockDim.x}$
<code>get_local_size(0);</code>	Size of each work group in the x dimension	$\text{blockDim.x}$

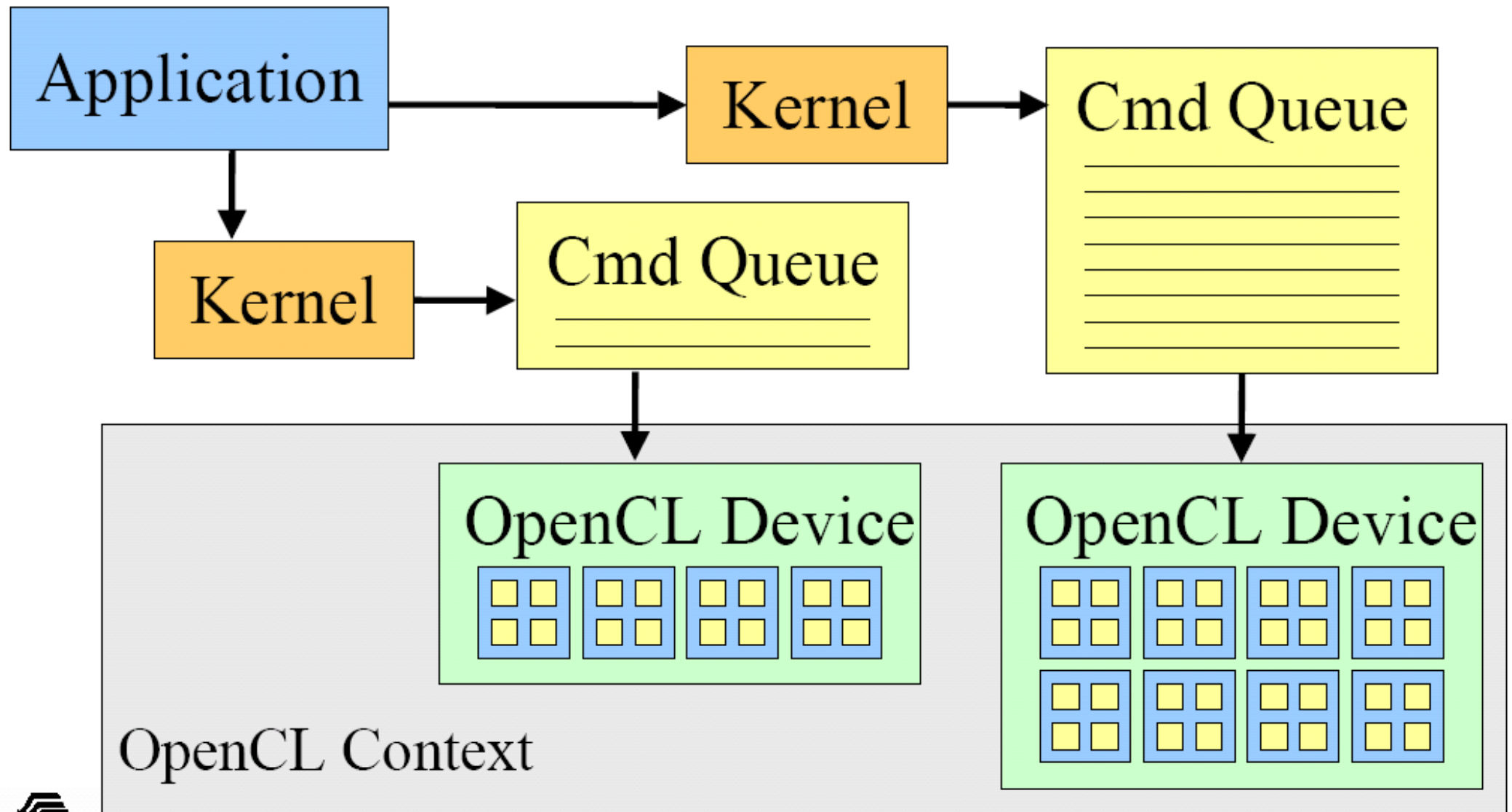
# Conceptual OpenCL Device Architecture



# Mapping OpenCL Memory Types to CUDA

OpenCL Memory Types	CUDA Equivalent
global memory	global memory
constant memory	constant memory
local memory	shared memory
private memory	Local memory

# OpenCL Context for Device Management





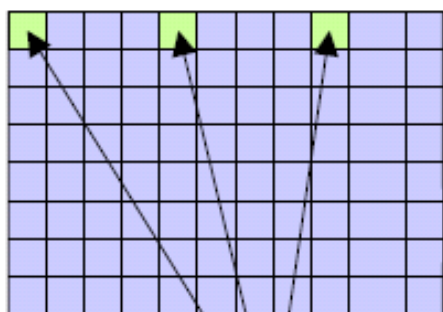
# OpenCL Version of DCS Kernel 3

(unrolled, coalesced)

Grid of thread blocks:

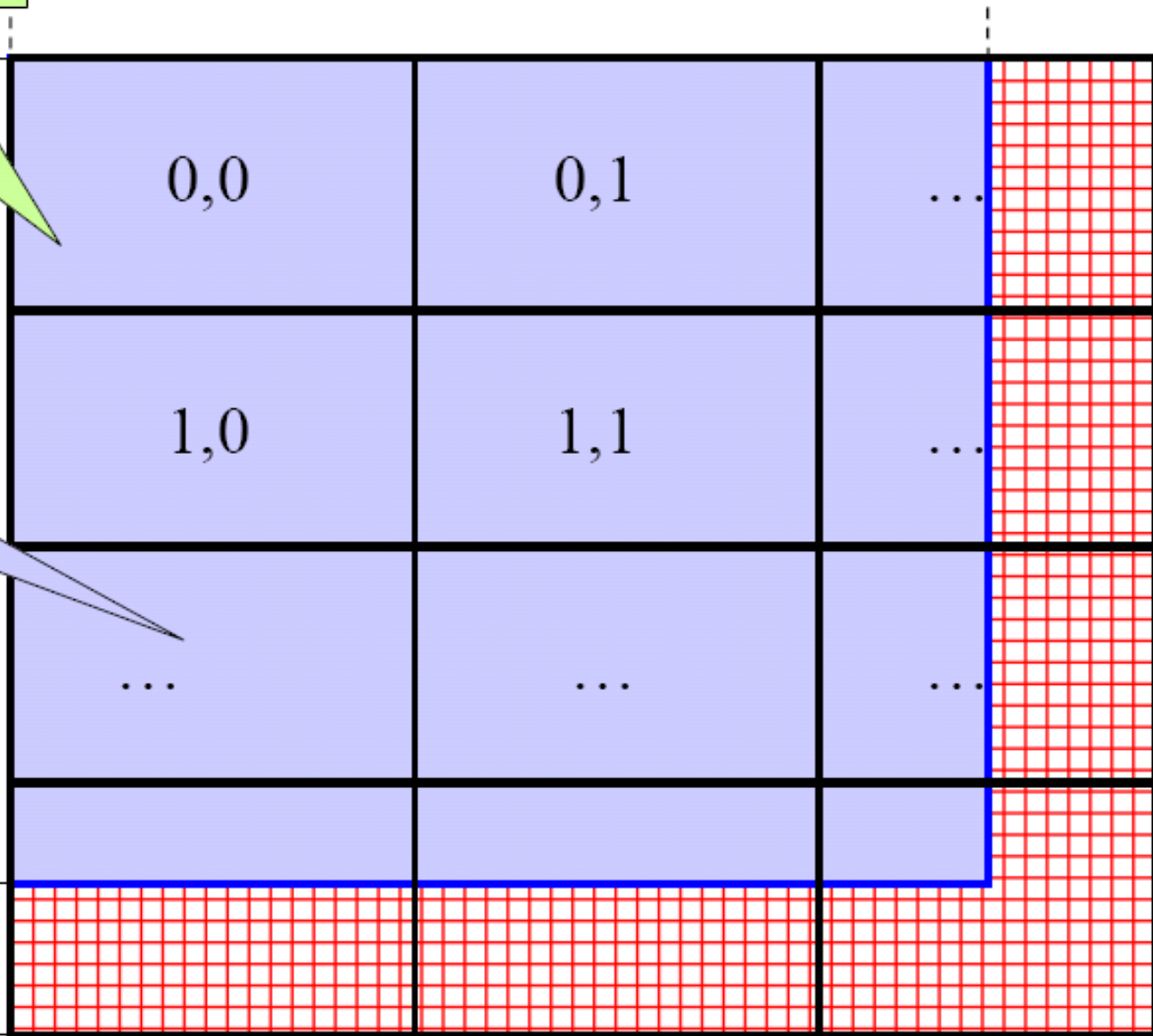
Unrolling increases computational tile size

Work Groups:  
64-256 work items



Work items compute up to 8 potentials, skipping by memory coalescing width

Padding waste



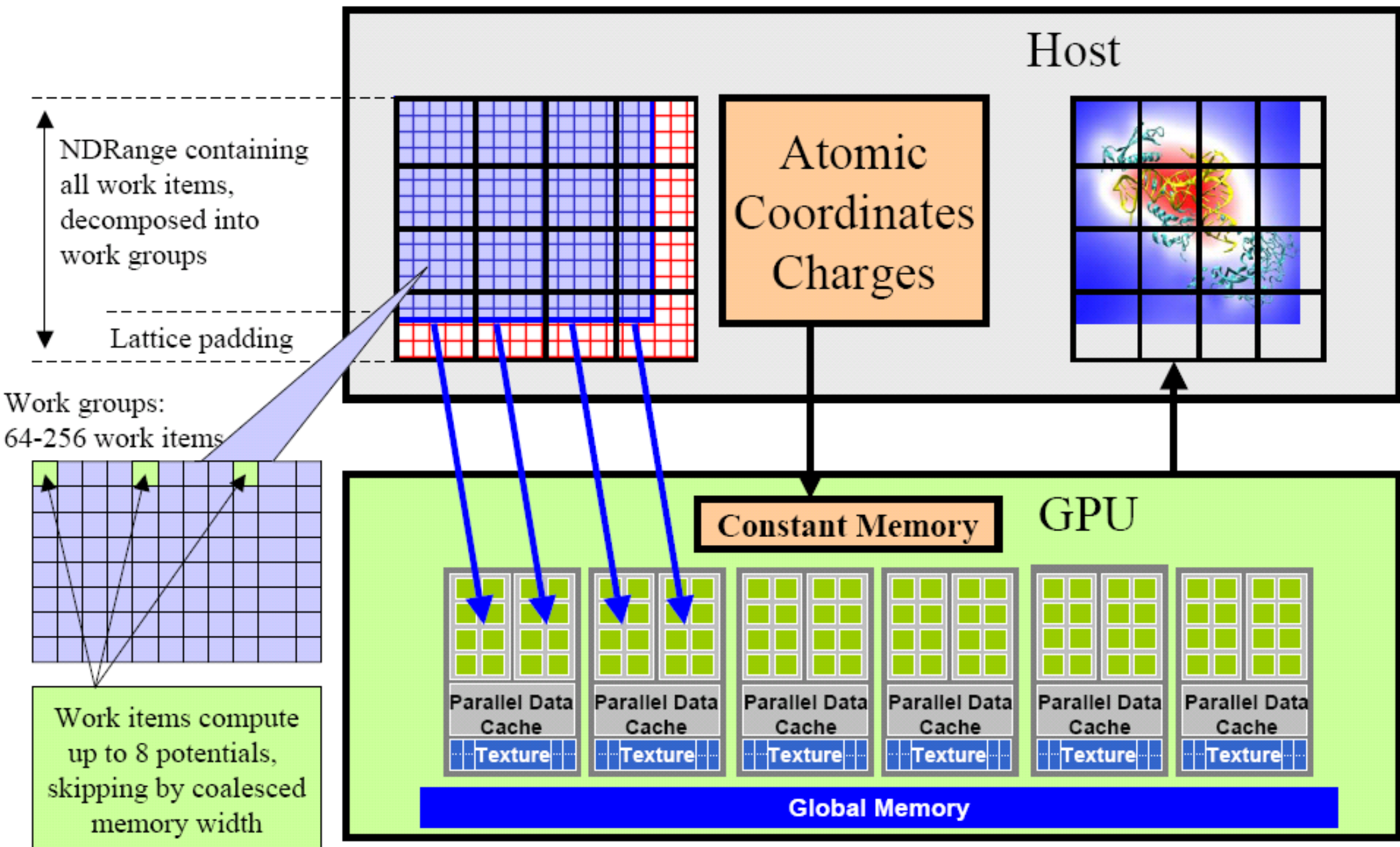


Figure 11.10 Mapping DCS NDRange to OpenCL Device

# Compiling OpenCL Programs

- kernel is compiled at run time
  - Compiler supplied by OpenCL+device driver
    - Easier to support variety of devices
    - Similar to shaders in OpenGL
  - No need for specialized compiler (nvcc)
- Download SDK from Apple/NVIDIA/AMD/...
  - Supports Linux/OSX/Windows
  - All CUDA capable NVIDIA Hardware
  - Recent AMD/ATI hardware
  - Others (e.g. S3 Chrome)

# Ex 21: OpenCL Matrix Multiply

- Ex 20 (CUDA Matrix Multiply) ported to OpenCL
- InitGPU: initialization
- AxBd: Multiply AB
  - Copy matrices from host to device
  - Run kernel
  - Copy result back to host
- AxB: kernel
  - Calculate one element (row • column)

# Homework 10: GPU Computing

- Make sure that you check that the answer you get is correct
- Just doing meaningless computations on the answer is **not** acceptable
- To see a speed gain the problem must be big enough