
The Abdus Salam
International Centre
for Theoretical Physics


# Introduction to OpenCL

**Ivan Girotto – igirotto@ictp.it**

Information & Communication Technology Section (ICTS)

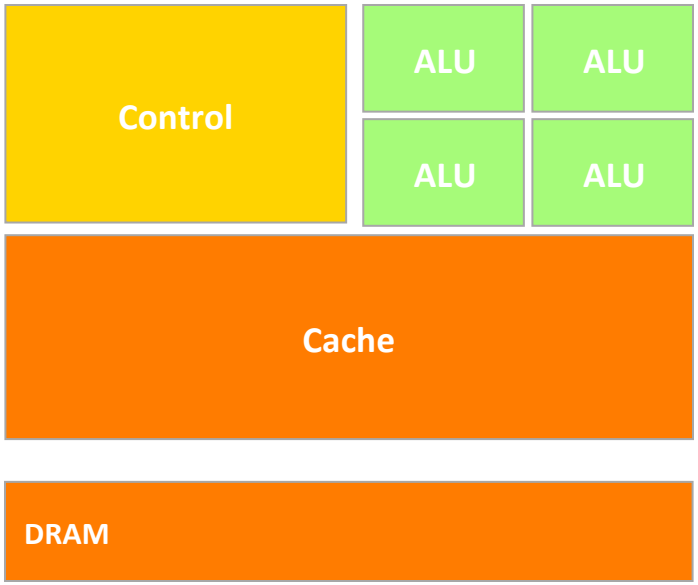
International Centre for Theoretical Physics (ICTP)

# OUTLINE

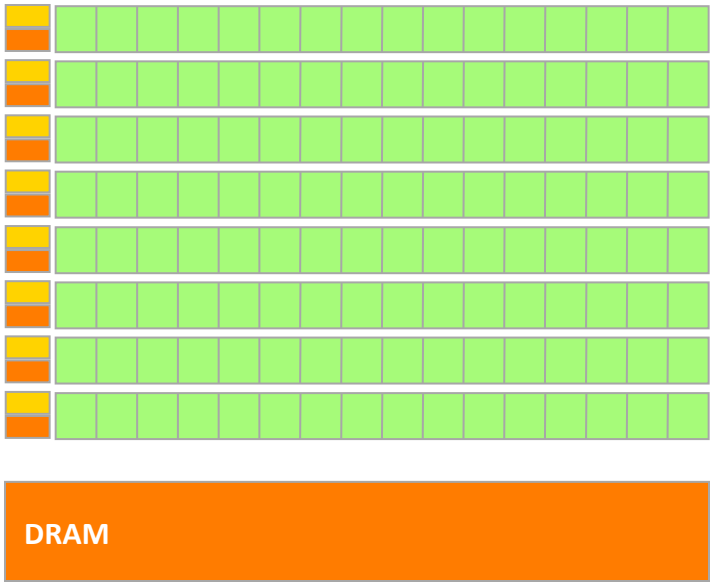
- I little bit of GPU computing
- Introduction to OpenCL
- A practical Example

# Parallel Processing on GPUs

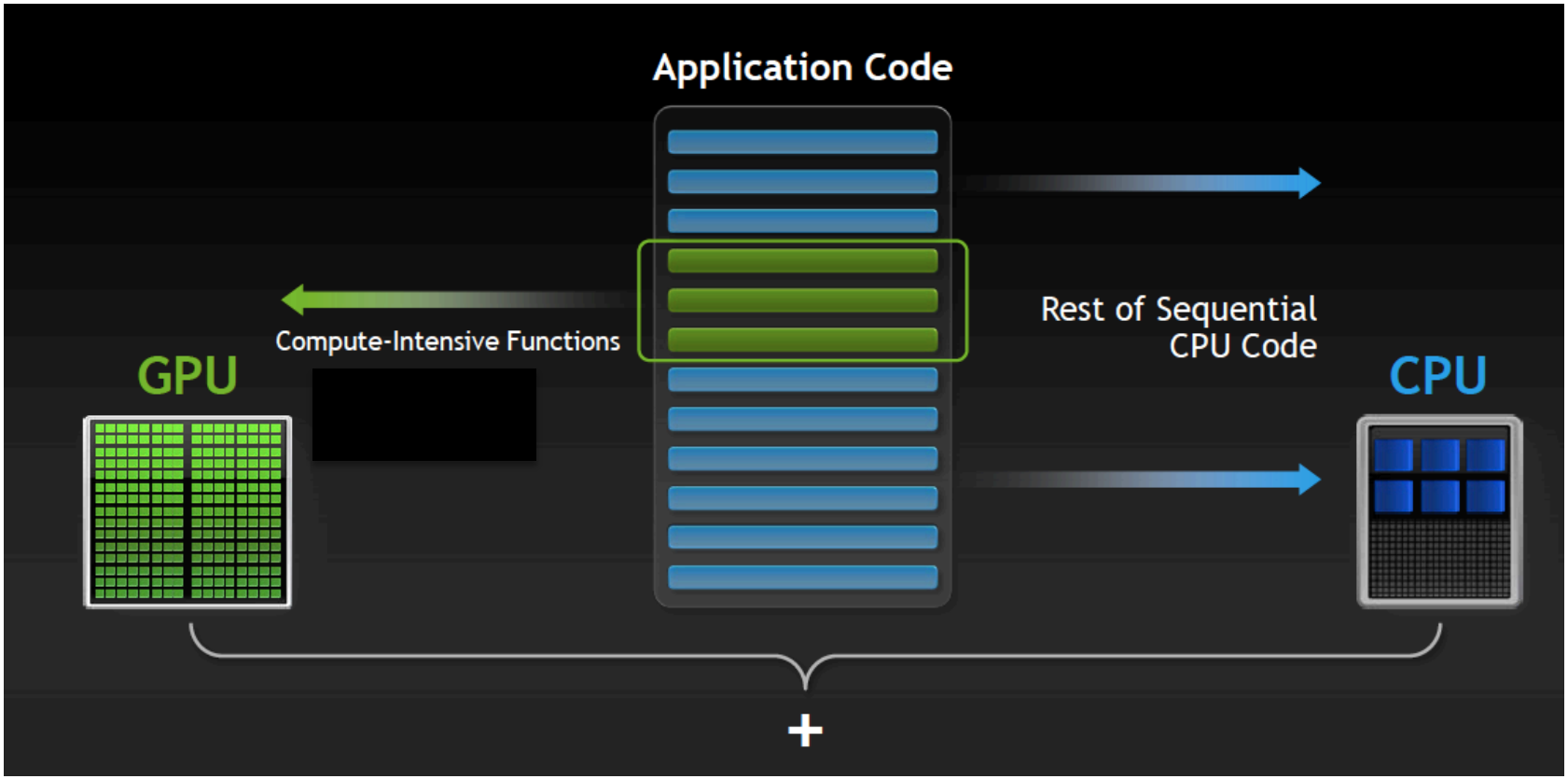
## CPU

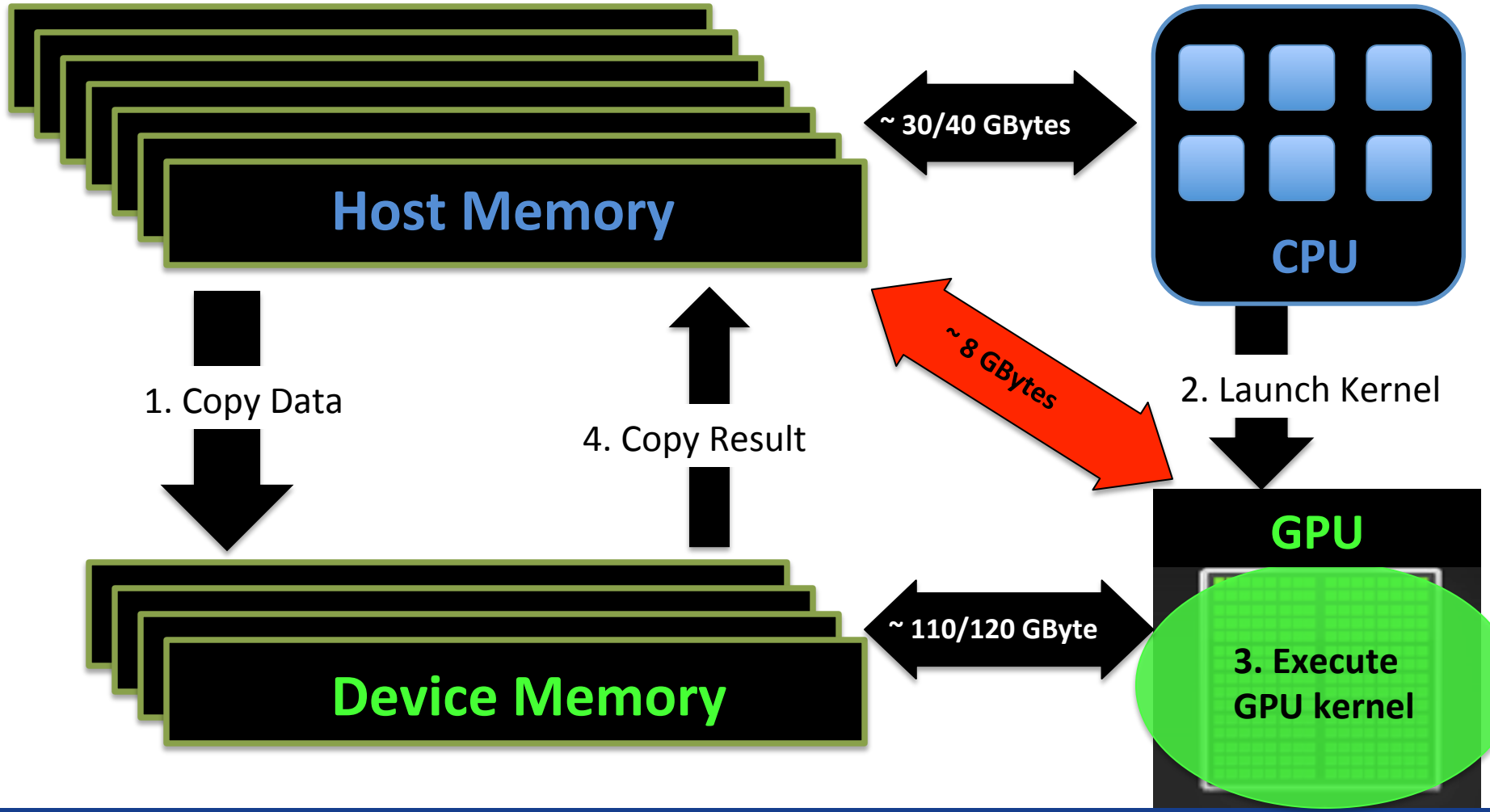



## GPU



# The GPU Idea


Application Code
Compute-Intensive Functions
GPU
Rest of Sequential CPU Code
CPU
+


Host Memory
~ 30/40 GBytes
CPU
~ 8 GBytes
1. Copy Data
4. Copy Result
2. Launch Kernel
GPU
Device Memory
~ 110/120 GByte
3. Execute GPU kernel

# OpenCL

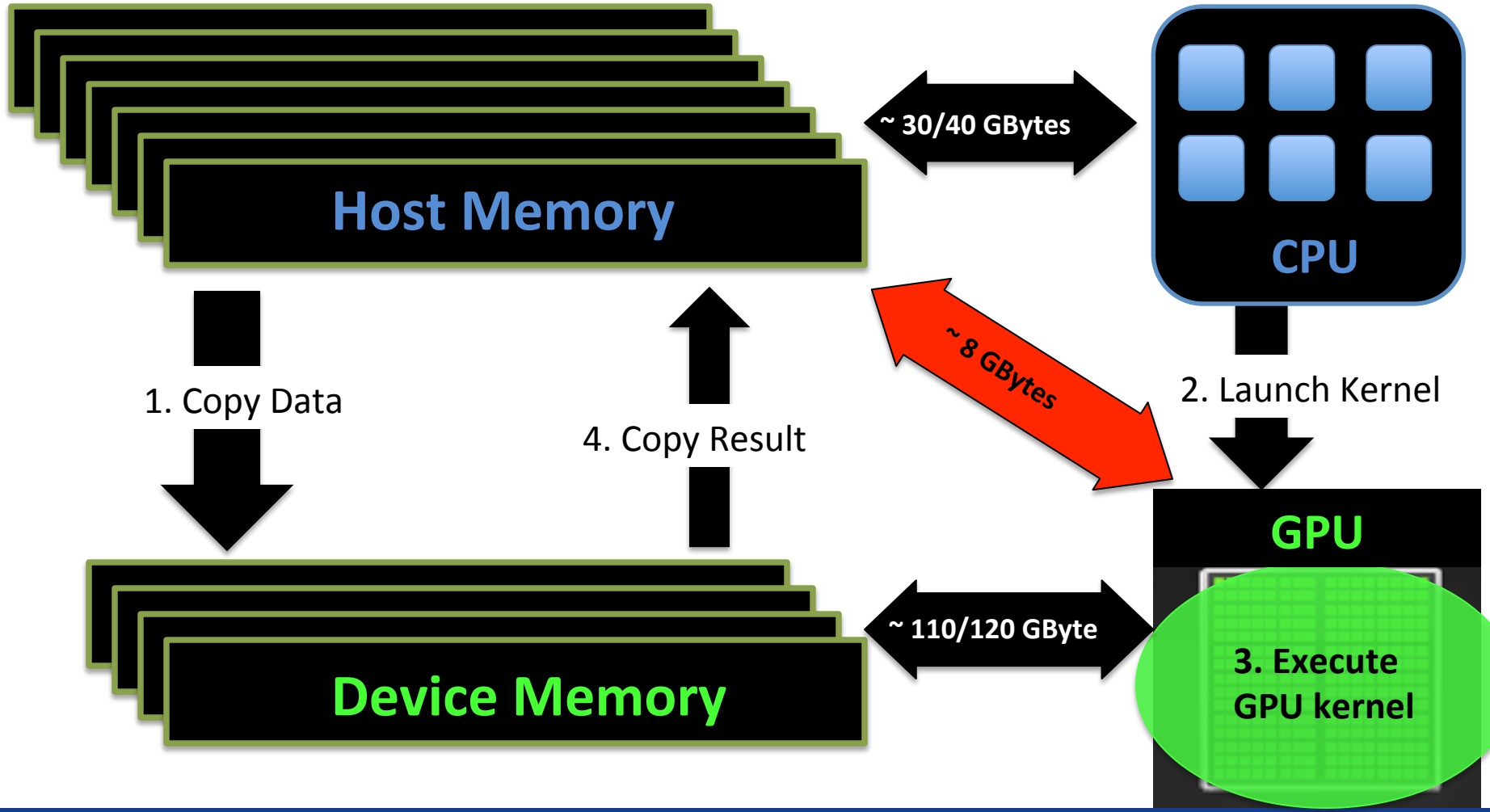
- *O*pen *C*ompute *L*anguage
- Open, royalty-free standard for cross-platform,
- For heterogeneous parallel-computing systems
- Cross-platform. Implementations for
  - ATI GPUs
  - NVIDIA GPUs
  - x86 CPUs

The Abdus Salam International Centre for Theoretical Physics


Source from http://www.khronos.org/

# The OpenCL Specification

- Platform Model
  - One Processor coordinates the execution (host), one or more processors execute OpenCL code (devices)
- Execution Model
  - Defines how the OpenCL environment is configured (host) and how kernels are executed (devices)
- Memory Model
  - Define an abstraction of the memory model
- Programming Model
  - Defines how the concurrency model is mapped to physical HW


Host Memory
~ 30/40 GBytes
CPU
~ 8 GBytes
1. Copy Data
4. Copy Result
2. Launch Kernel
GPU
Device Memory
~ 110/120 GByte
3. Execute GPU kernel

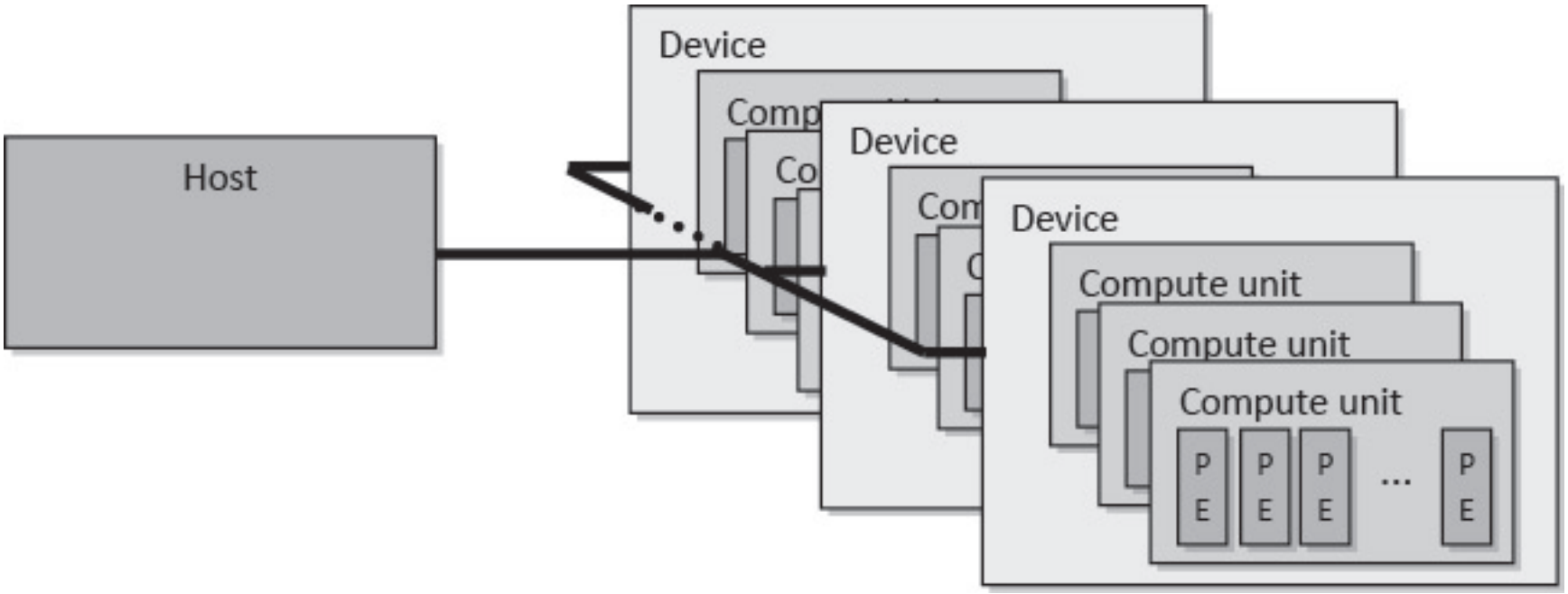
# Kernels and the Execution Model

- The unit of concurrent execution is a work-item
- each work-item executes the kernel
- The programmer specifies the number of work-items that should be created as an n-dimensional range
- Work-items are distributed among equally sized work-groups. Work-items within a work-groups have a special relation (shared memory, synchronization)

# Host-Device Interaction /1

- In the platform model there is a single host that coordinates execution on one or more devices.
- Platforms can be thought of as vendor-specific implementations of the OpenCL API
- It is also a runtime driver-like

# Host-Device Interaction /2



# THE OPENCL ENVIRONMENT

# The Execution Environment /1

- Context
  - handles the host-device interaction, manages the memory objects on the devices and tracks program and kernels created for each device
- Command Queues
  - Any API that specifies host-device interaction will always begin with clEnqueue, requiring to specify a queue

# The Execution Environment /2

- Events
  - Dependences (for asynch execution) & Profiling
- Memory Objects
  - It encapsulates data to be transferred on a device
  - It is valid for a single context
- Barriers (Flush & Finish)

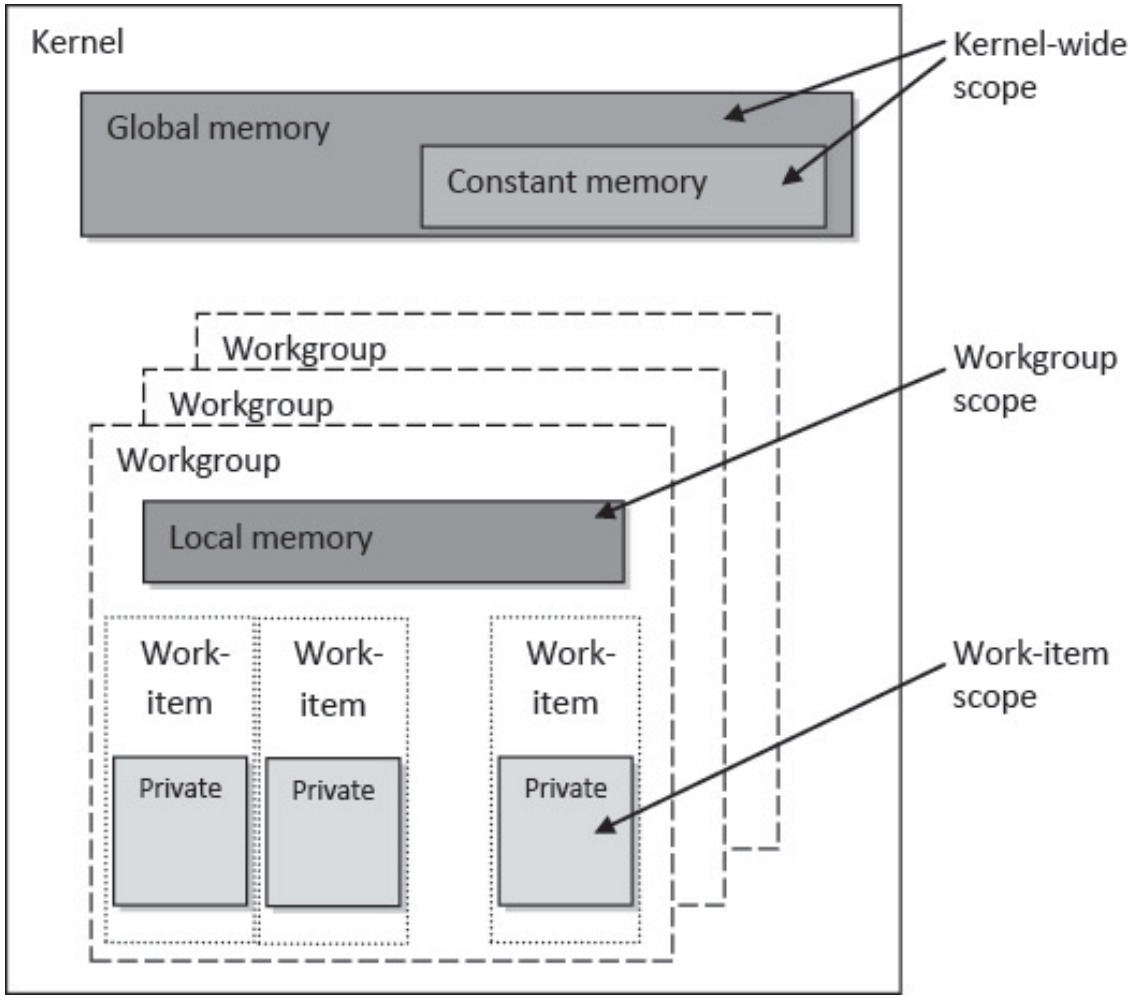
# The Execution Environment /3

- Creating an OpenCL Program Object
  - Collection of one or more kernels (dynamic lib like)
  - The program is stored in a C string
  - The source is turned into a program object
  - The program object is completed
- The OpenCL Kernel
  - Extract the kernel from the program, set the parameters, run the kernel
  - The clEnqueueNDRRangeKernel() call is asynchronous and it returns immediately once the kernel is queued

# The Memory Model /1

- Global Memory
  - visible to all compute units on the device
  - hosts data transferred from host to device
- Constant Memory
  - read-data only, specifically for simultaneous access
- Local Memory
  - shared within a work group (NVIDIA GPU shared_mem)
- Private Memory
  - individual to work-item (registers)

# Memory Model /2



# Basic Program structure

- Get platform & devices info
- Create contexts
- Load and compile the program
- Create queue
- Set data on the device
- Load and run kernels
- Store results

# Example Overview: Vector Add

- OpenMP:
  - https://computing.llnl.gov/tutorials/openMP/
- OpenCL:
  - http://www.heterogeneouscompute.org/wordpress/wp-content/uploads/2011/06/Chapter2.txt

# References

- http://www.khronos.org/
- http://www.heterogeneouscompute.org/

KHRONOS GROUP
CONNECTING SOFTWARE TO SILICON