中山大學
SUN YAT-SEN UNIVERSITY

国家超级计算广州中心
NATIONAL SUPERCOMPUTER CENTER IN GUANGZHOU

GPU简介

任课教师：吴迪、杜云飞

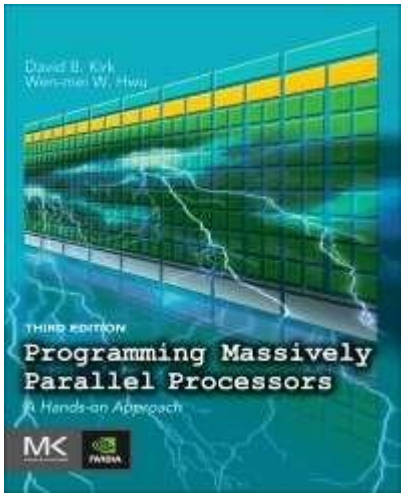
Roadmap

- Why GPUs
- GPU Architecture
- GPU-CPU Interaction
- GPU programming model
- When GPUs excel? When not?
- Solving real-life problems using GPUs

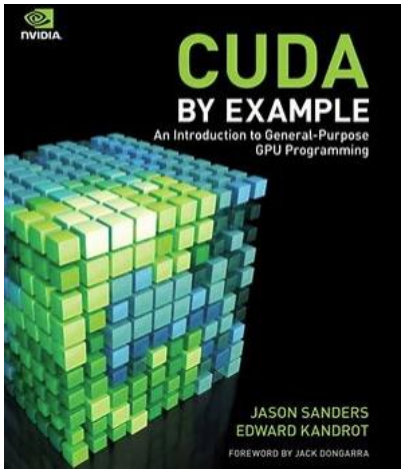
– With the best performance we can get!

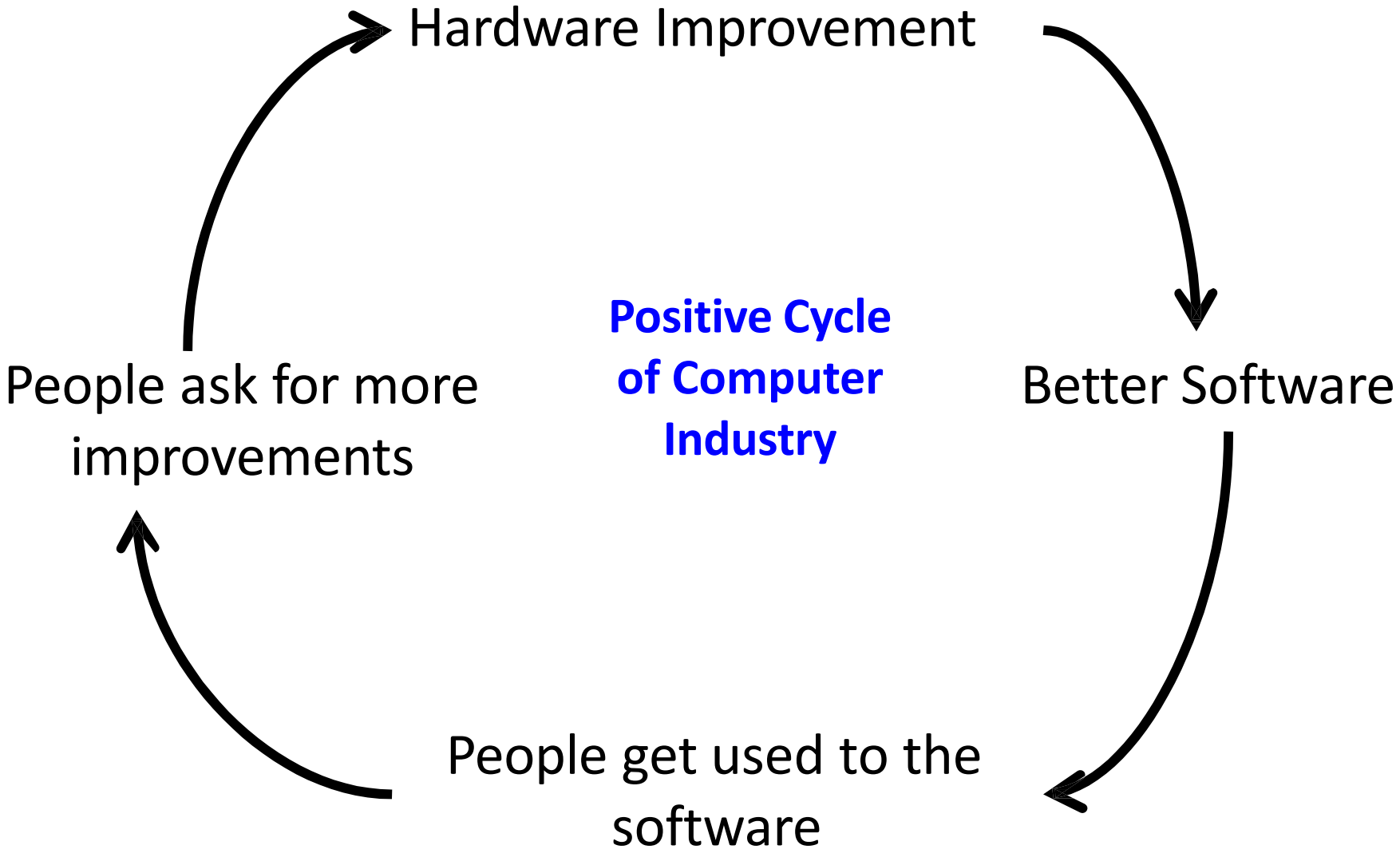
References

- Programming Massively Parallel Processors: A Hands-on Approach, By David B. Kirk, Wen-mei W. Hwu 3rd Edition



- CUDA by Example: An introduction to General-Purpose GPU Programming, by Jason Sanders, Edward Kandrot





Software cost dominates hardware cost.

Important Questions

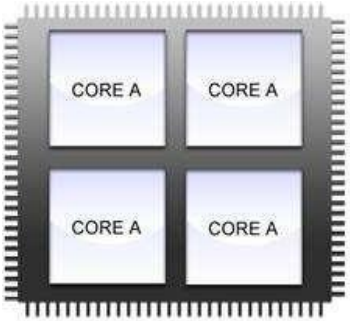
- **How to control software cost?**
  - By reducing redesigning of the software.
- **And how to do that?**
  - By making the application **scalable**
    - More cores
    - More threads per core
    - More memory
    - Faster interconnect
    - Basically: scalability in the face of hardware growth.
  - By making the application **portable**
    - Across different instruction sets (x86, ARM, …)
    - From multicore to GPU to FPGA to ….
    - Shared vs distributed memory
    - …

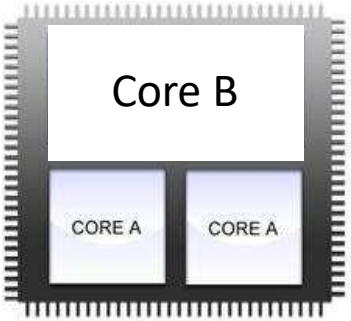
The Status-Quo

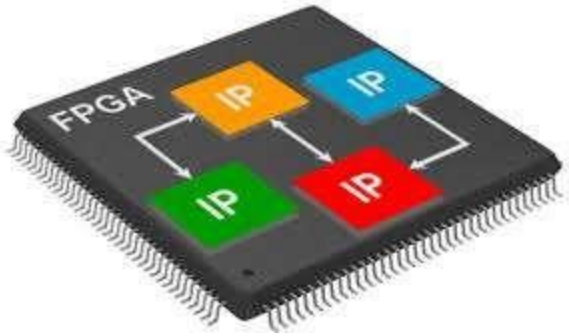
- We moved from single core to multicore
  - for technological reasons
- Free lunch is over for software folks
  - The software will not become faster with every new generation of processors
- Not enough experience in parallel programming
  - Parallel programs of old days were restricted to some elite applications -> very few programmers
  - Now we need parallel programs for many different applications

Not only parallel programming

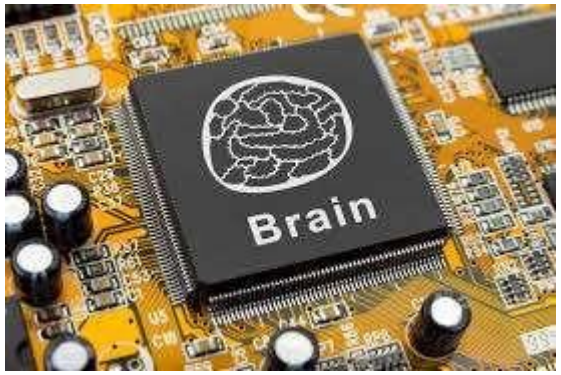
But Heterogeneous parallel programming!



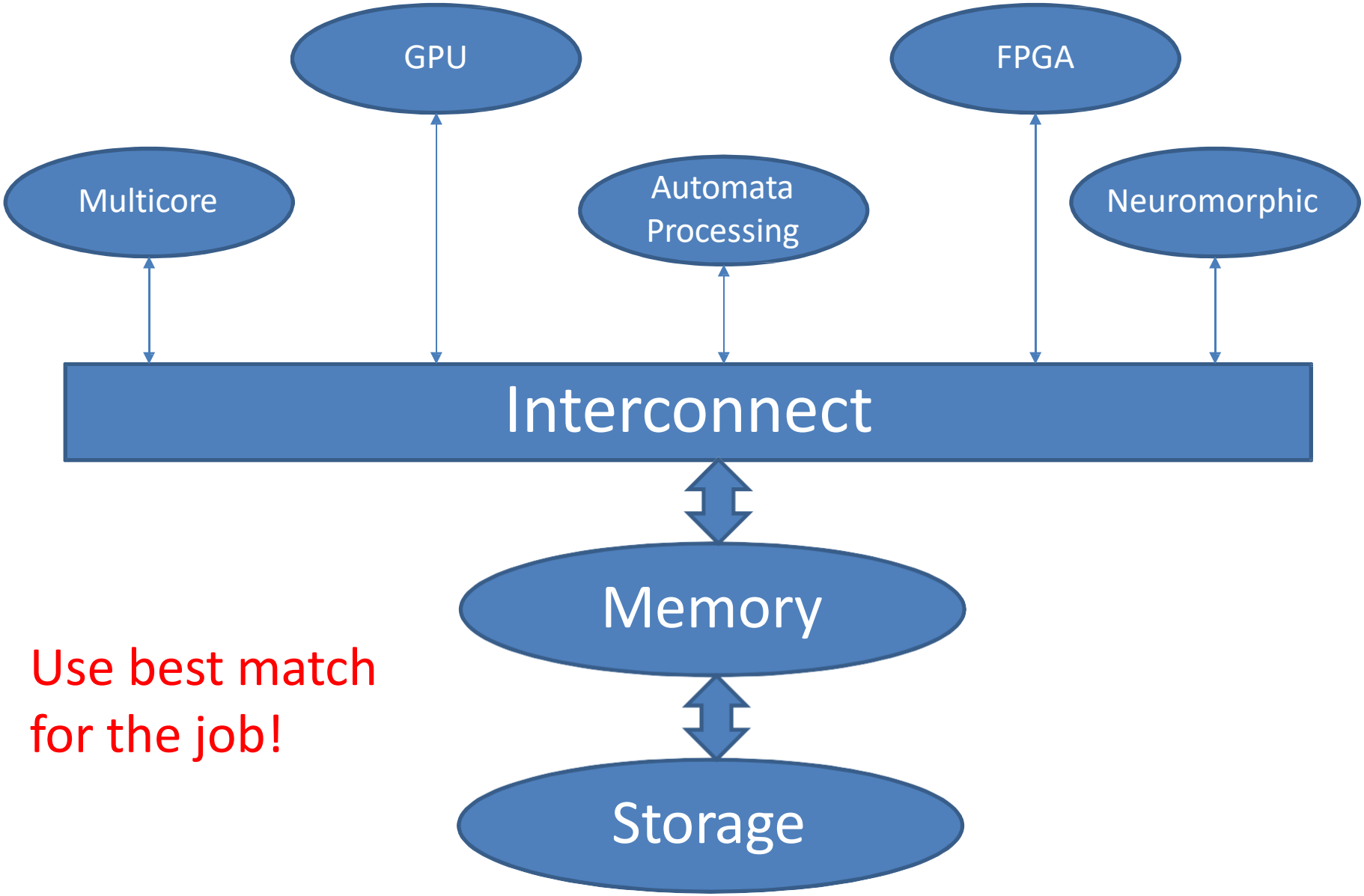




Heterogeneity Everywhere



Neuromorphic Chips



Use best match for the job!

Software Perspective

Two types of developers

Performance Group

(C/C++, CUDA, OpenCL, ….)

Productivity Group

(Python, Scala, …)

Two Main Goals

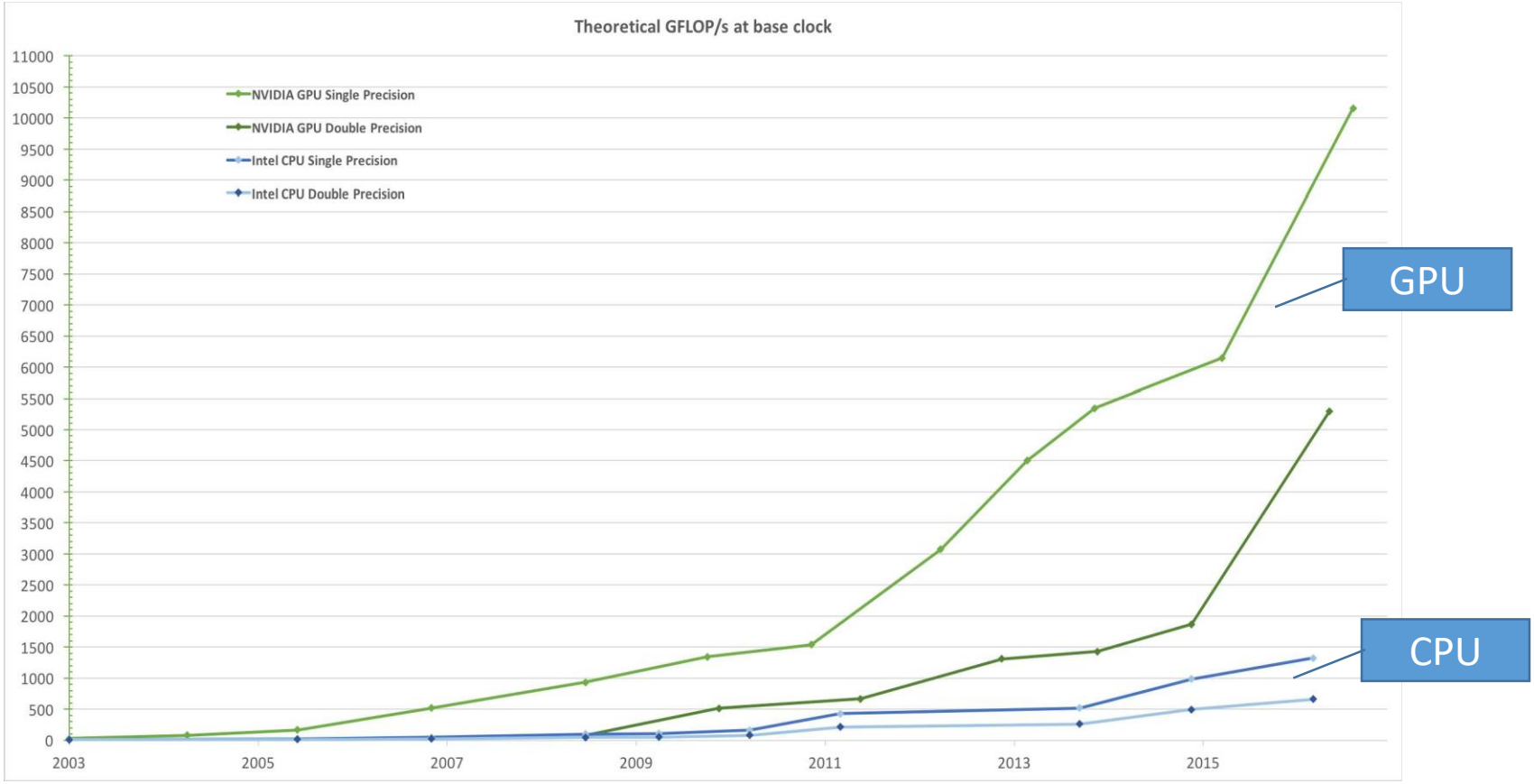
- Maintain execution speed of old sequential programs
- Increase throughput of parallel programs

Two Main Goals

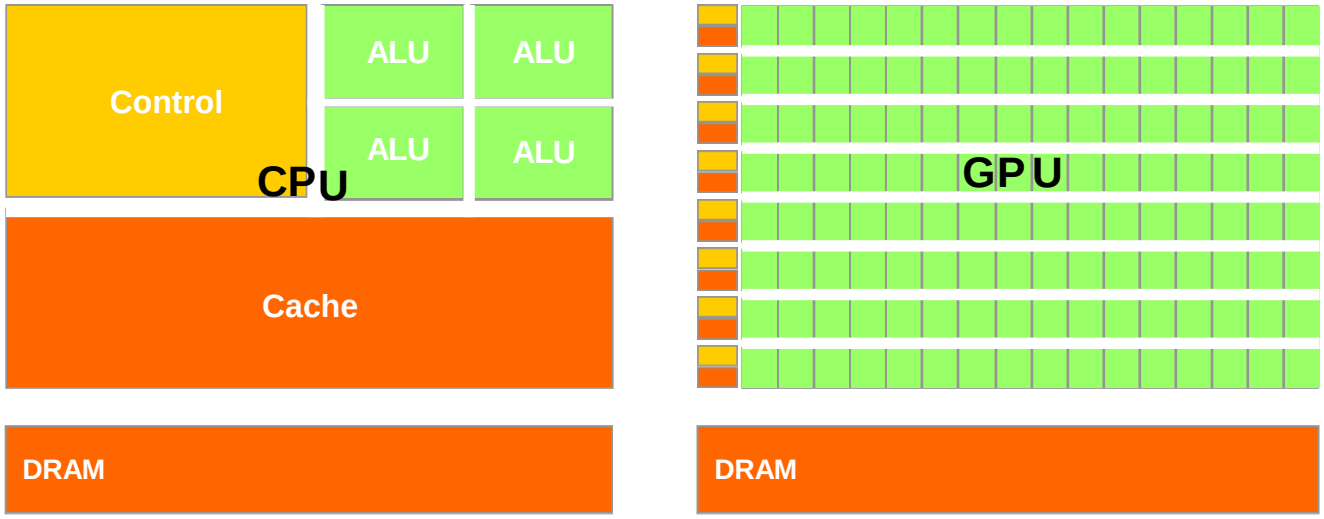
- Maintain execution speed of old sequential programs

- Increase throughput of parallel programs

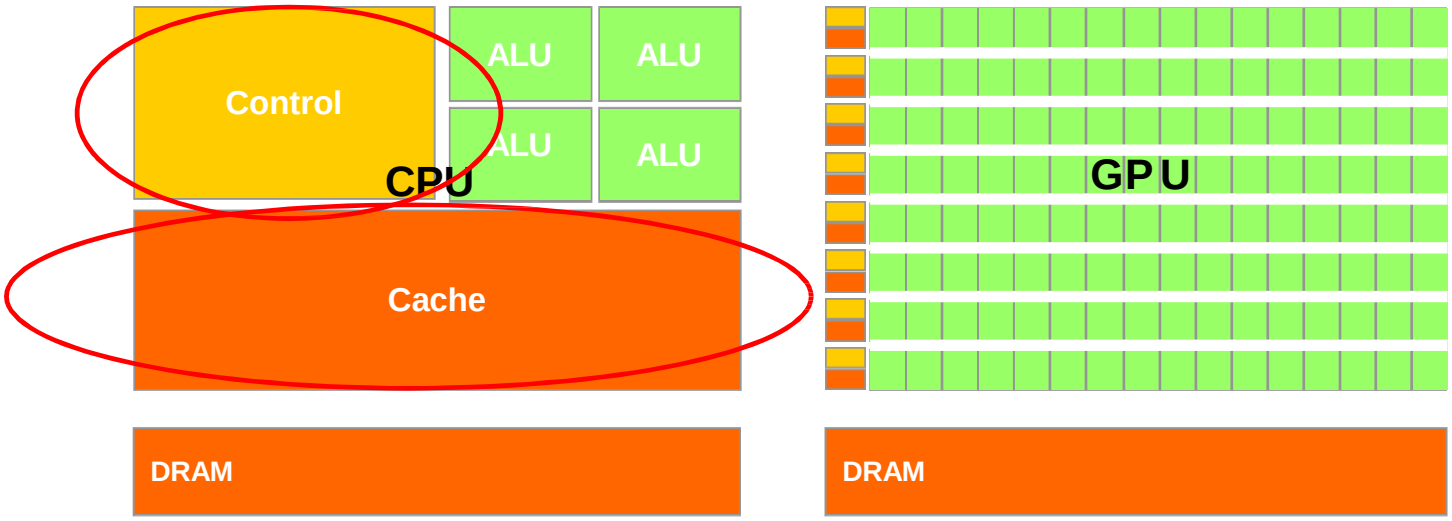
Performance

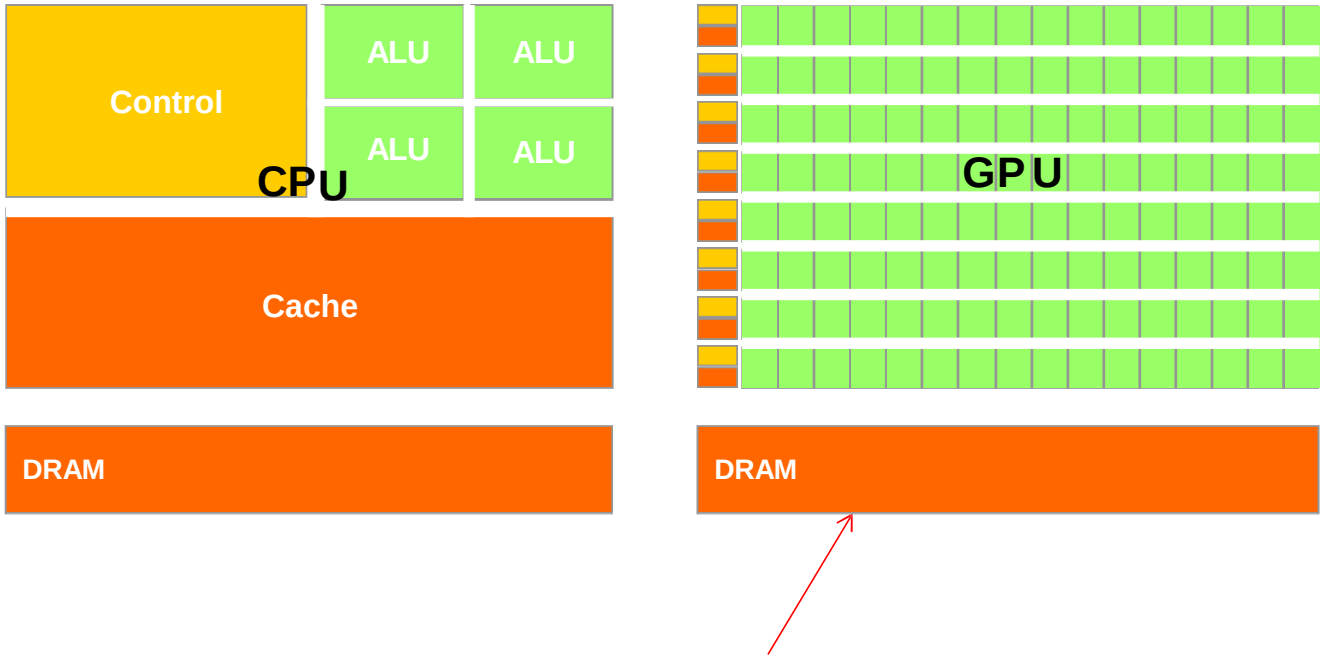


Source: NVIDIA CUDA C Programming Guide


Control
ALU
ALU
ALU
ALU
CPU
Cache
DRAM
GPU
DRAM

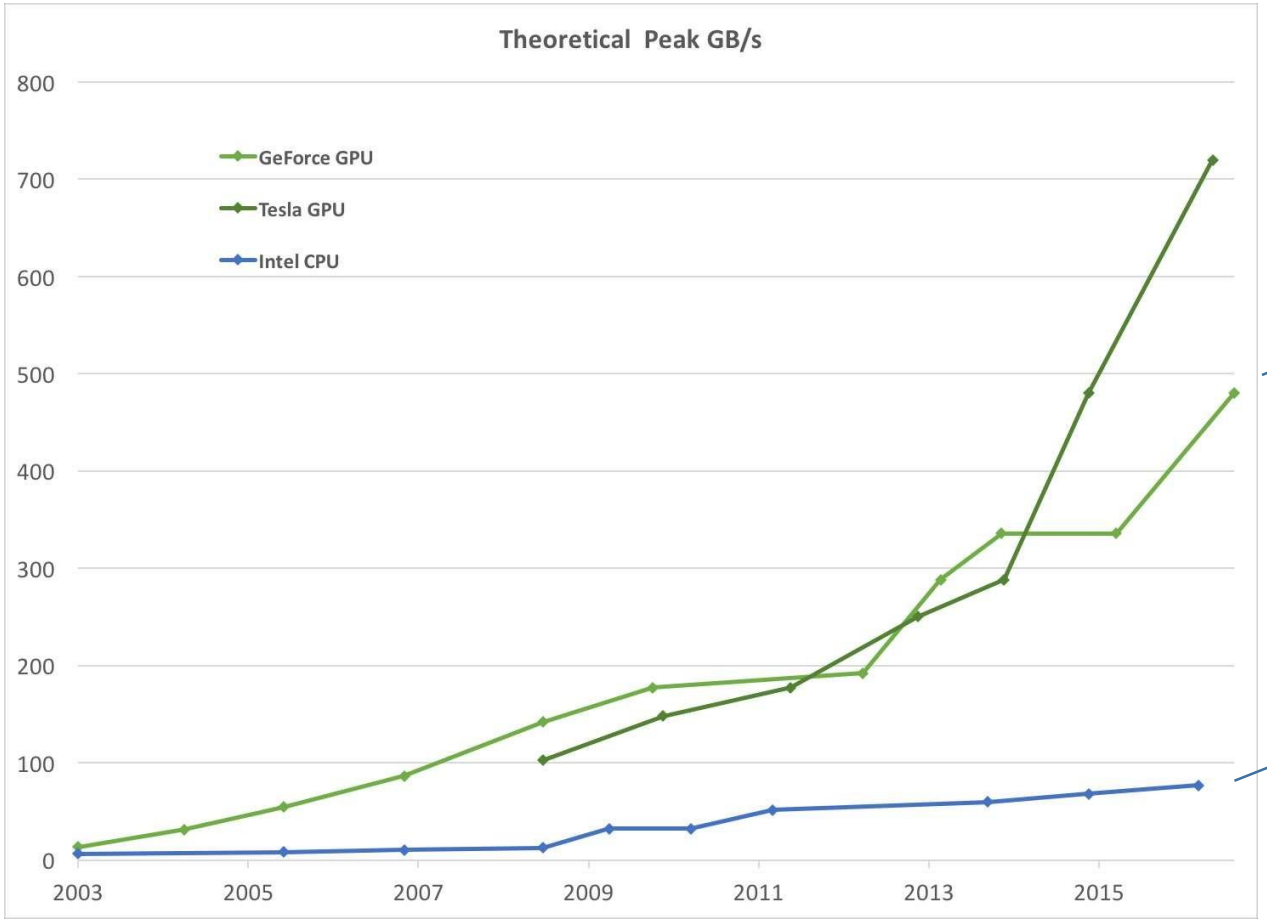
CPU is optimized for sequential code performance





Almost 10x the bandwidth of multicore (relaxed memory model)

Memory Bandwidth

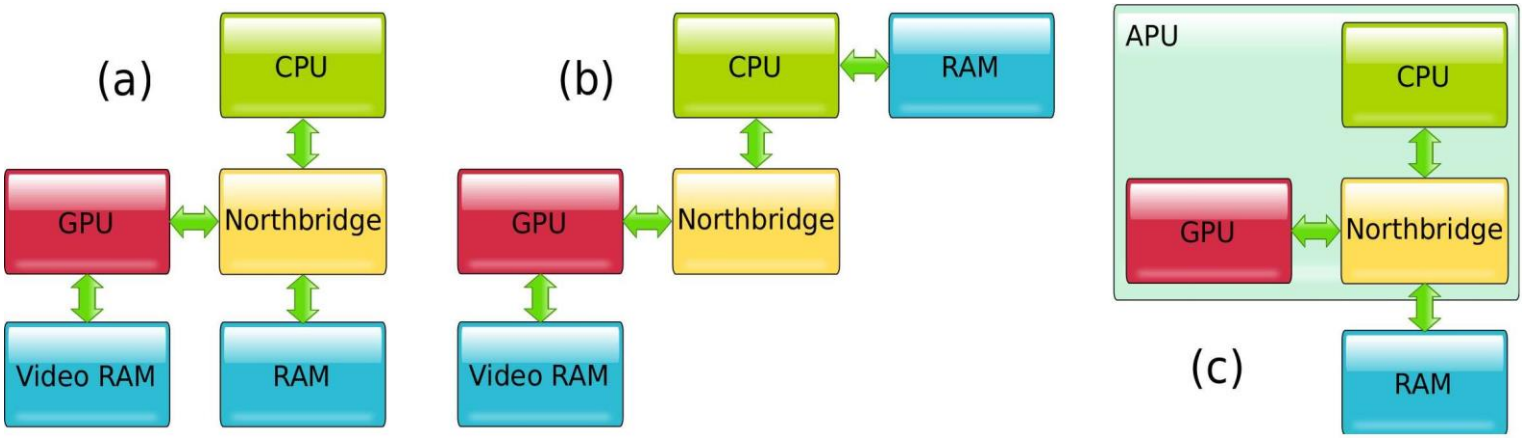


Source: NVIDIA CUDA C Programming Guide

How to Choose A Processor for Your Application?

- Performance
- Very large installation base
- Practical form-factor and easy accessibility
- Support for IEEE floating point standard

Integrated GPU vs Discrete GPU



- (a) and (b) represent discrete GPU solutions, with a CPU- integrated memory controller in (b).
- Diagram (c) corresponds to integrated CPU-GPU solutions, as the AMD's Accelerated Processing Unit (APU) chips.

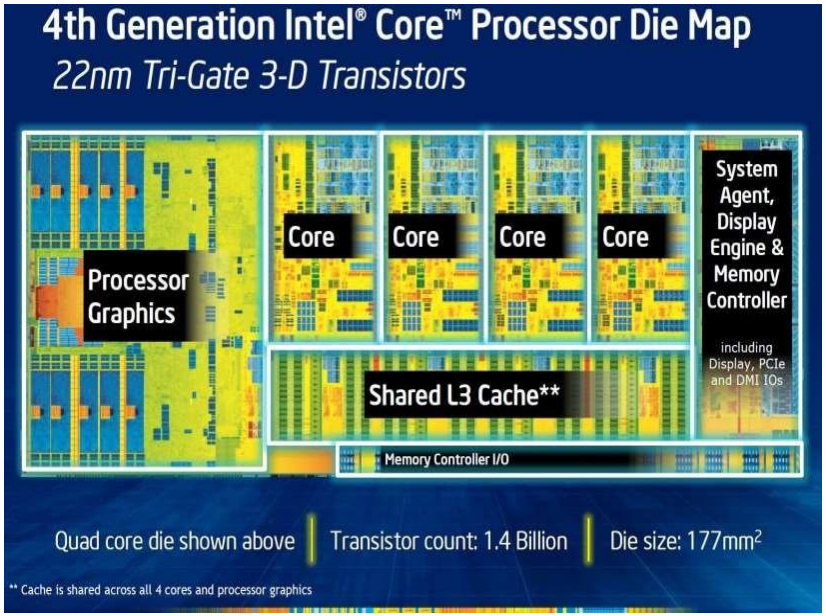
source: *Multicore and GPU Programming: An Integrated Approach by* G. Barlas, 2014

Tradeoff: Low energy vs higher performance

Integrated CPU+GPU processors

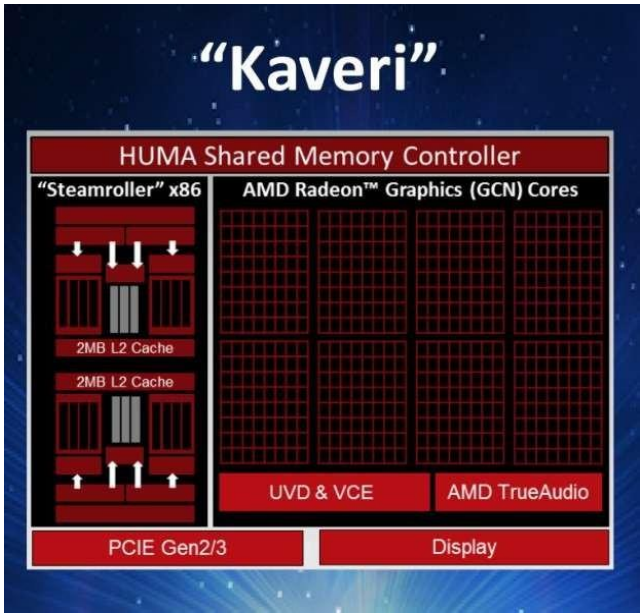
- More than 90% of processors shipping today include a GPU on die
- Low energy use is a key design goal

Intel 4th Generation Core Processor: "Haswell"



4-core GT2 Desktop: **35 W package**
2-core GT2 Ultrabook: **11.5 W package**

AMD Kaveri APU



http://www.geeks3d.com/20140114/amd-kaveri-a10-7850k-a10-7700k-and-a8-7600-apus-announced/

Desktop: **45-95 W package**
Mobile, embedded: **15 W package**

source: Performance and Programmability Trade-offs in the OpenCL 2.0 SVM and Memory Model
by Brian T. Lewis, Intel Labs

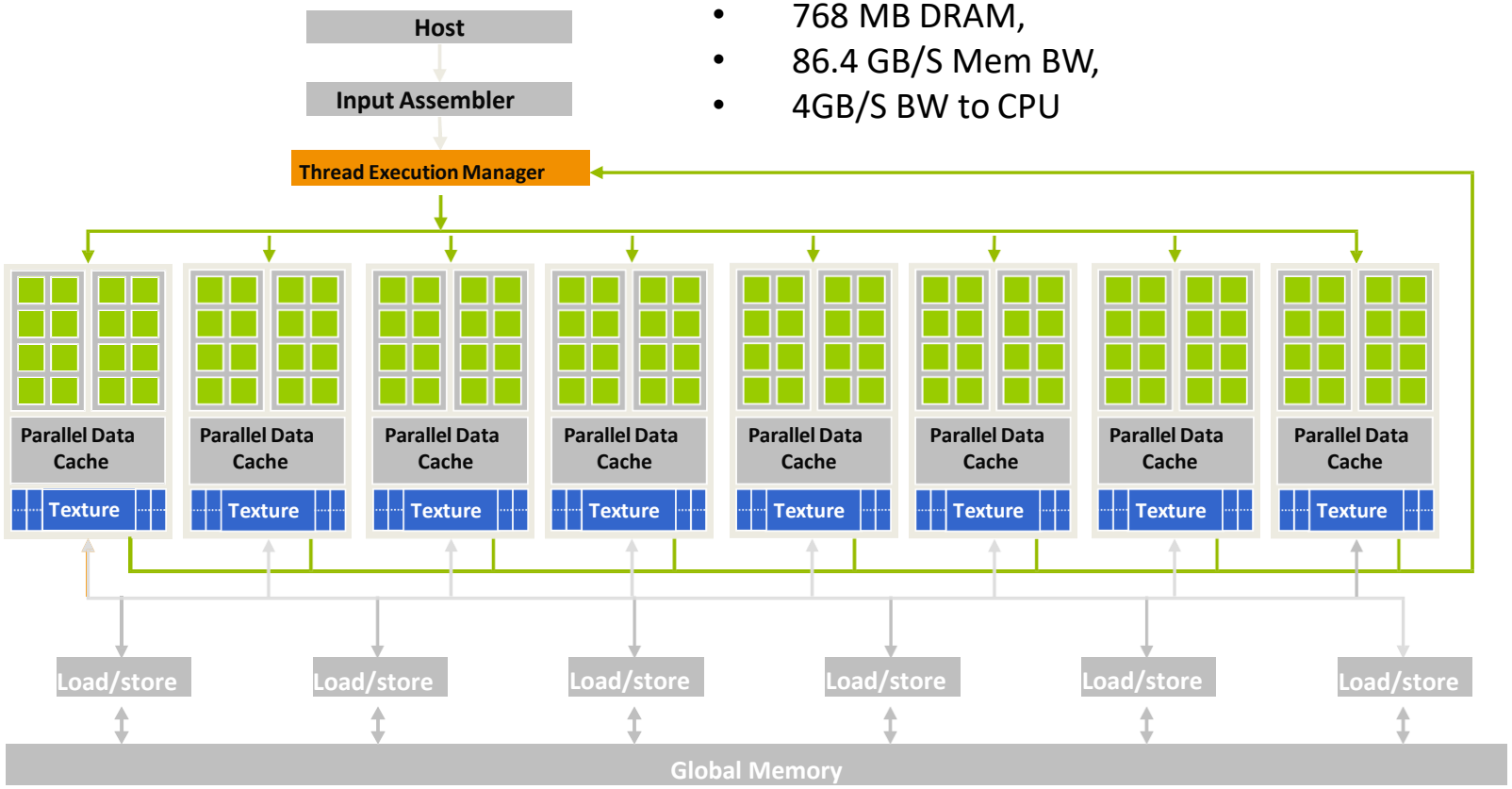
Is Any Application Suitable for GPU?

- ***Heck no!***
- You will get the best performance from GPU if your application is:
  - Computation intensive
  - Many independent computations
  - Many similar computations

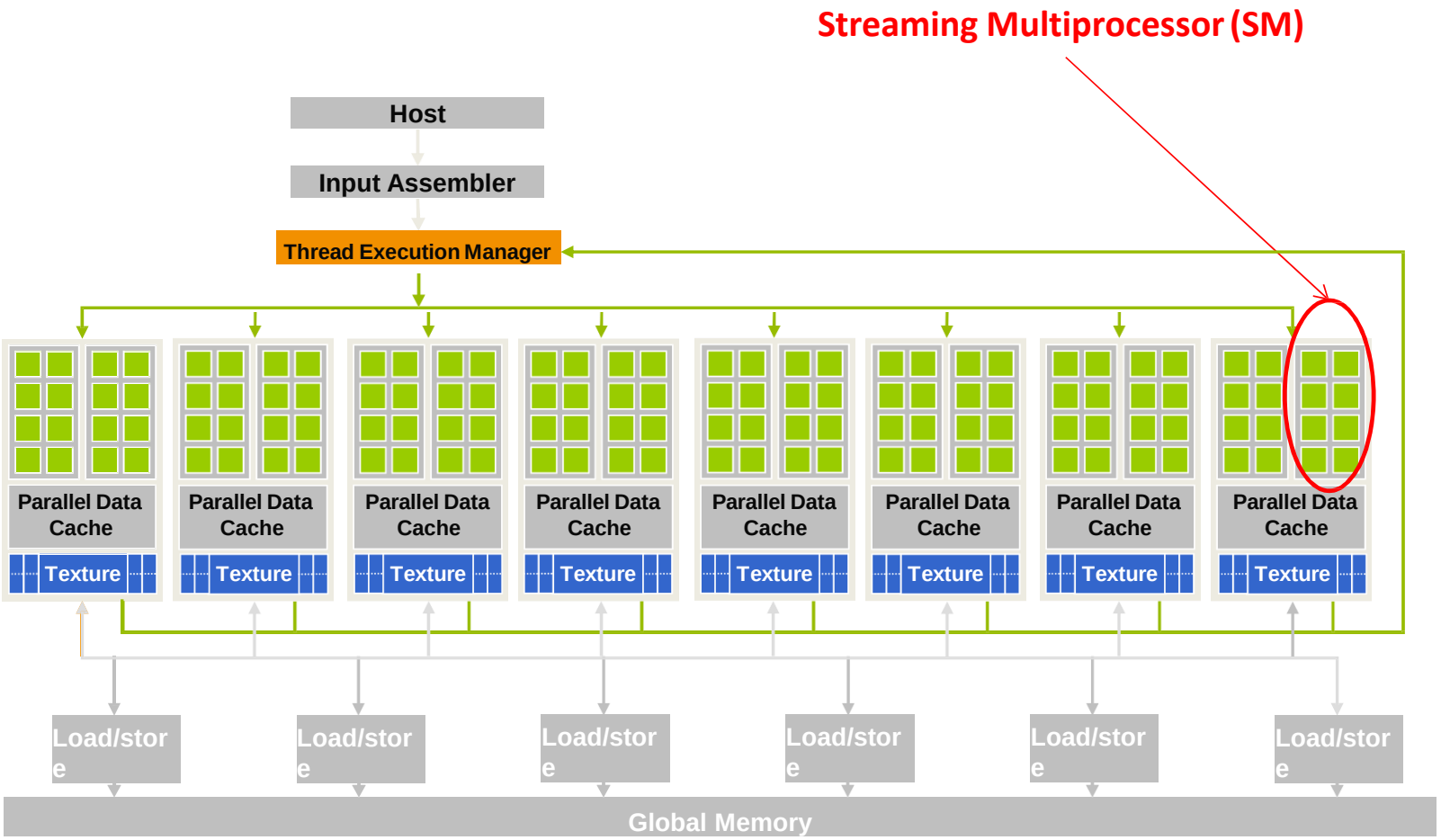
A Glimpse at a GPGPU:

GeForce 8800 (2007)

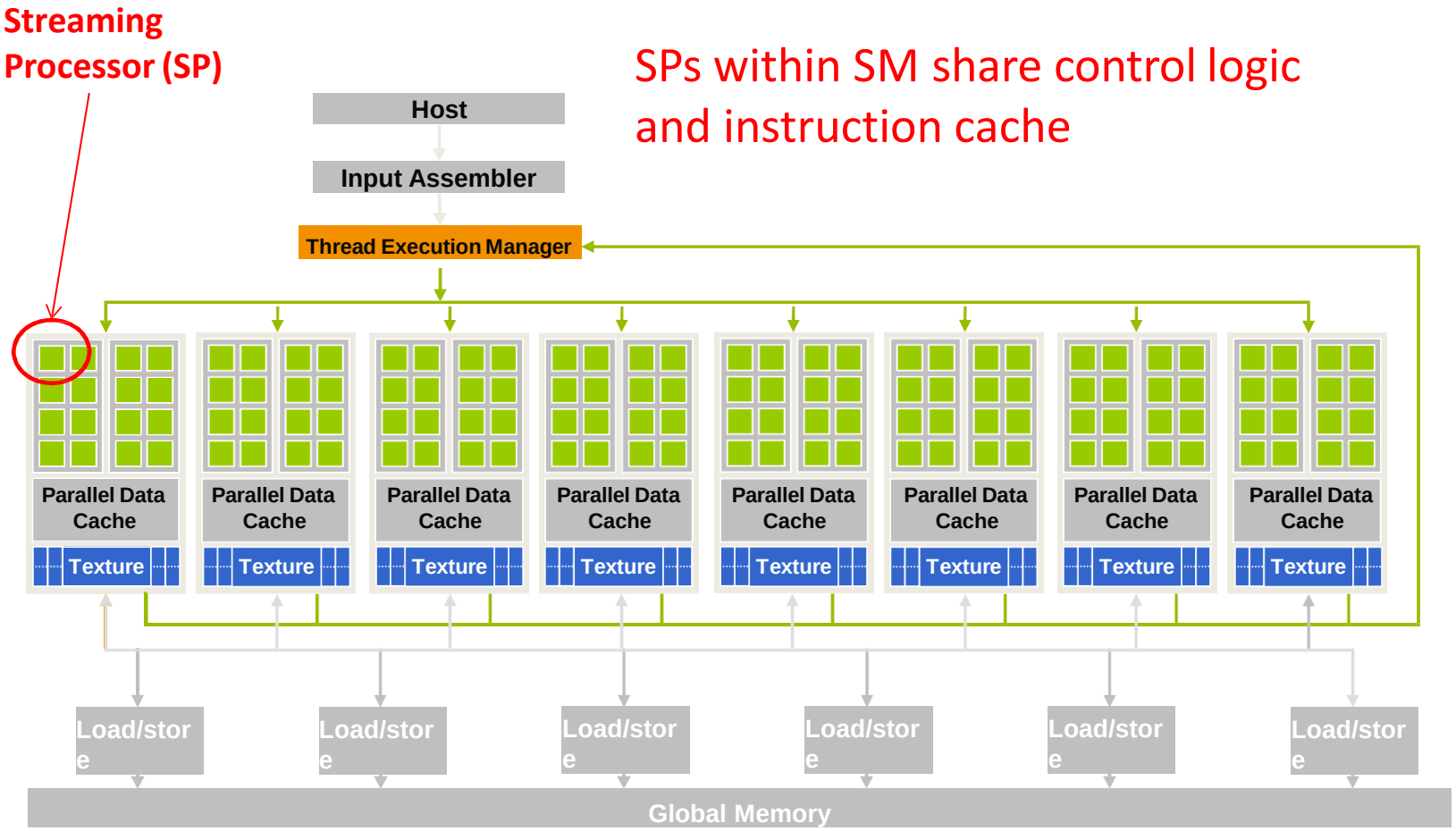
- 16 highly threaded SM's,
- >128 FPU's, 367 GFLOPS,
- 768 MB DRAM,
- 86.4 GB/S Mem BW,
- 4GB/S BW to CPU



A Glimpse at a GPU

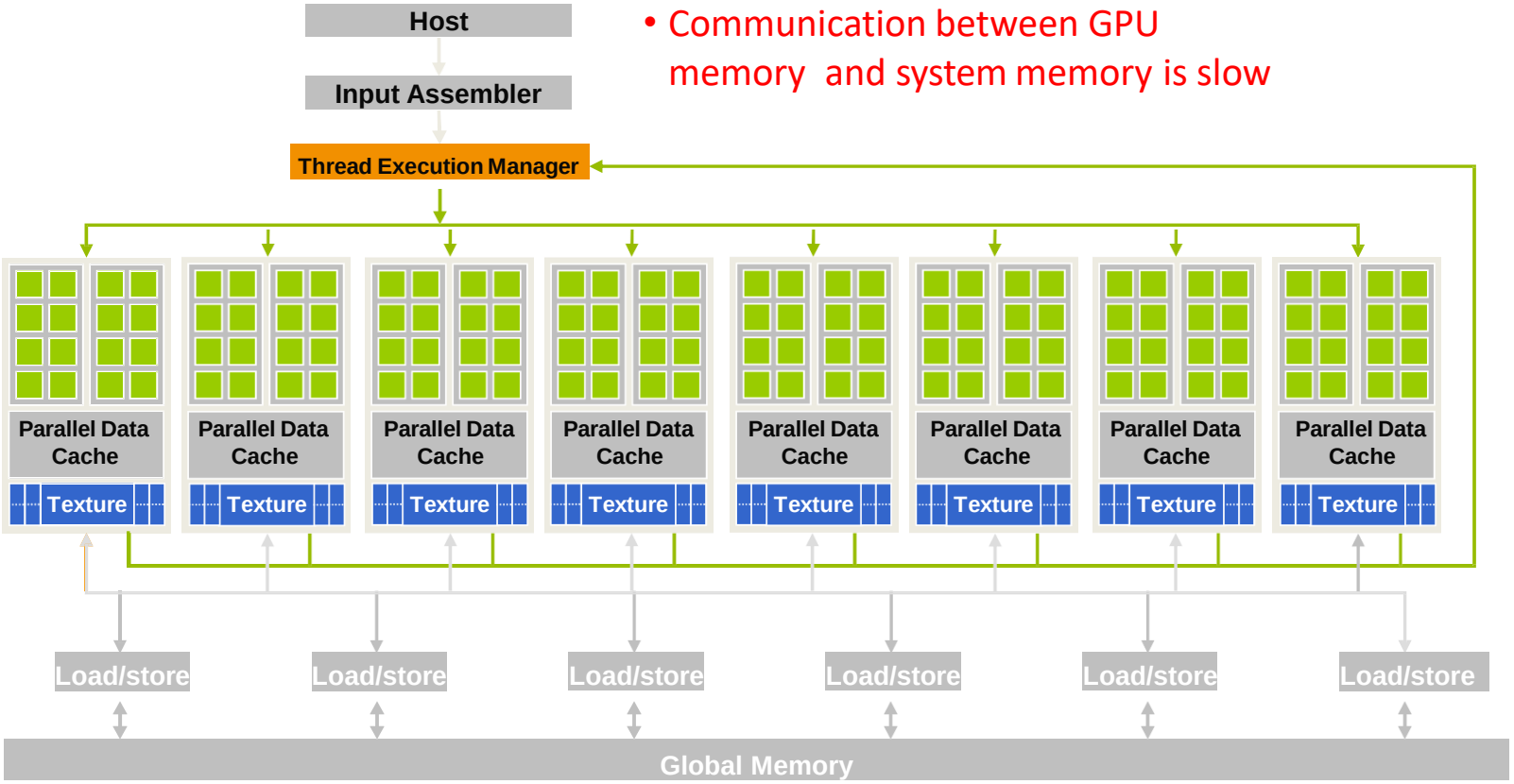


A Glimpse at a Modern GPU



A Glimpse at a Modern GPU

- Much higher bandwidth than typical system memory
- A bit slower than typical system memory
- Communication between GPU memory and system memory is slow



Winning Applications Use Both CPU and GPU

- CPUs for sequential parts where latency matters
  - CPUs can be 10X+ faster than GPUs for sequential code
- GPUs for parallel parts where throughput wins
  - GPUs can be 10X+ faster than CPUs for parallel code

Things to Keep in Mind

- Try to increase the portion of your program that can be parallelized
- Figure out how to get around limited bandwidth of system memory
- When an application is suitable for parallel execution, a good implementation on GPU can achieve more than 100x speedup over sequential implementation.