Introduction to Accelerated/Hybrid Computing with GPGPU Architectures

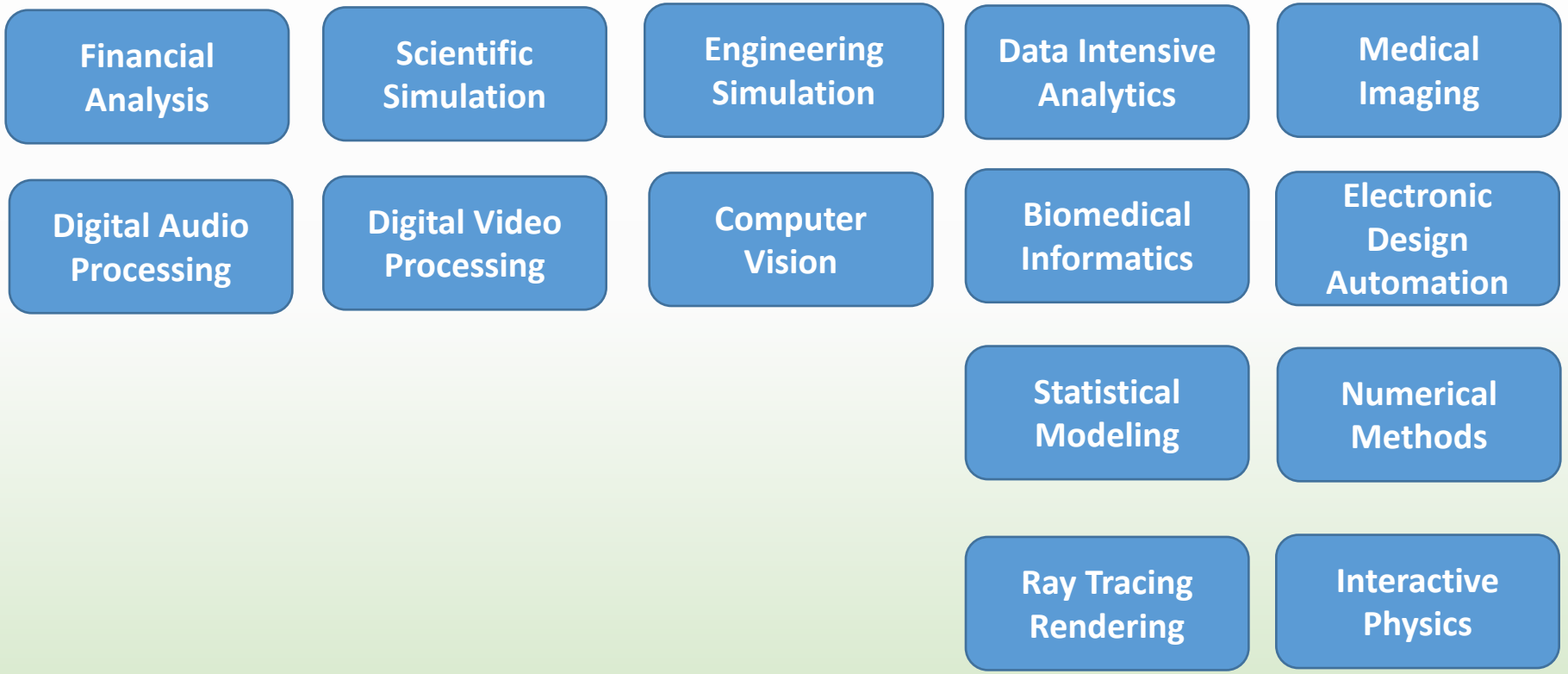
Carlos J. Barrios H., PhD

@carlosjaimebh

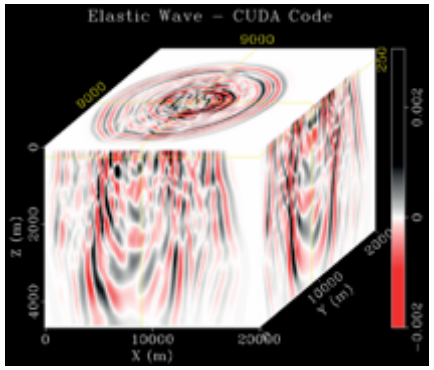
@SC3UIS

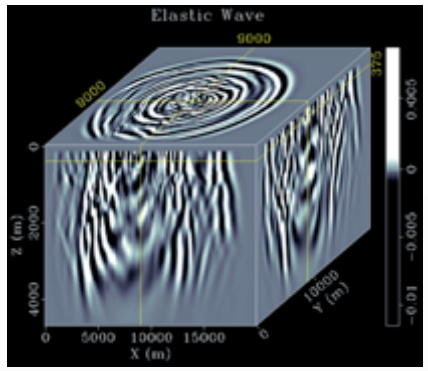
Super Computación y Cálculo Científico UIS

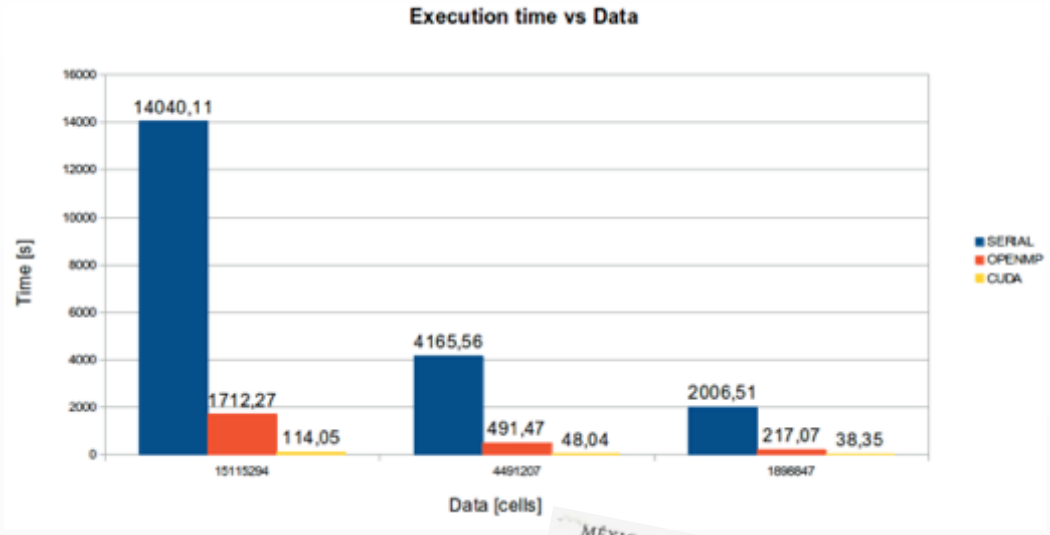
Heterogeneous Parallel Computing in Many Disciplines



Interesting @SC3UIS Experiences







Processing and Visualization for Oil Reservoirs (3D seismic modelling in isotropic and heterogeneous media)

For 10 Millons of bases

0. Original App 3 Months
1. App (3 Weeks)
2. App (2- Days)
3. App (4 Minutes)

Processing Genomic Data for Mexican Flu AHN1 Discovering

About Top500 List -2018 November Ranking

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , **IBM** DOE/SC/Oak Ridge National Laboratory United States	2,397,824	143,500.0	200,794.9	9,783
2	**Sierra** - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , **IBM / NVIDIA / Mellanox** DOE/NNSA/LLNL United States	1,572,480	94,640.0	125,712.0	7,438
3	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , **NRCPC** National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
4	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000 , **NUDT** National Super Computer Center in Guangzhou China	4,981,760	61,444.5	100,678.7	18,482
5	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 , **Cray Inc.** Swiss National Supercomputing Centre (CSCS) Switzerland	387,872	21,230.0	27,154.3	2,384
6	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , **Cray Inc.** DOE/NNSA/LANL/SNL United States	979,072	20,158.7	41,461.2	7,578
7	**AI Bridging Cloud Infrastructure (ABCI)** - PRIMERGY CX2570 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR , **Fujitsu** National Institute of Advanced Industrial Science and Technology (AIST) Japan	391,680	19,880.0	32,576.6	1,649
8	**SuperMUC-NG** - ThinkSystem SD530, Xeon Platinum 8174 24C 3.1GHz, Intel Omni-Path , **Lenovo** Leibniz Rechenzentrum Germany	305,856	19,476.6	26,873.9	
9	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x , **Cray Inc.** DOE/SC/Oak Ridge National Laboratory United States	560,640	17,590.0	27,112.5	8,209
10	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom , **IBM** DOE/NNSA/LLNL United States	1,572,864	17,173.2	20,132.7	7,890

TOP500 The List.

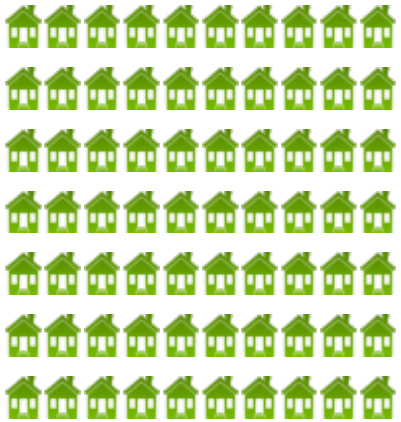
- No. 1 Summit is a GPGPU machine
- 6/10 Powerful Machines are MPP Clusters
- 6/10 are Hybrid Machines with Accelerators
  - 5 NVIDIA GPU Technology
    - 3 Different Generations (Keppler, Pascal and Volta)
  - 1 Intel Xeon Phi Technology
    - 1 Combines GPUs + MICs

Why Computing Perf/Watt Matters?

2.3 PFlops

7.0 Megawatts

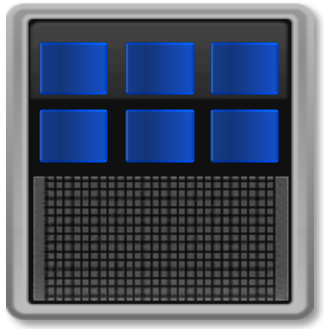
7000 homes



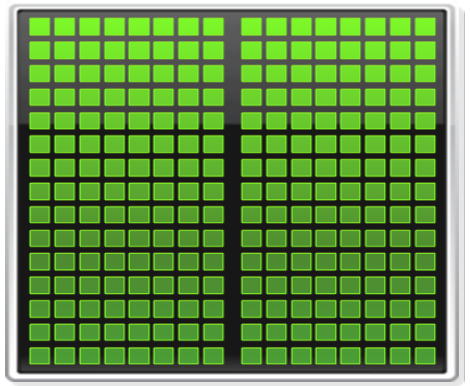
7.0 Megawatts

Traditional CPUs are not economically feasible

CPU
Optimized for Serial Tasks



GPU Accelerator
Optimized for Many Parallel Tasks

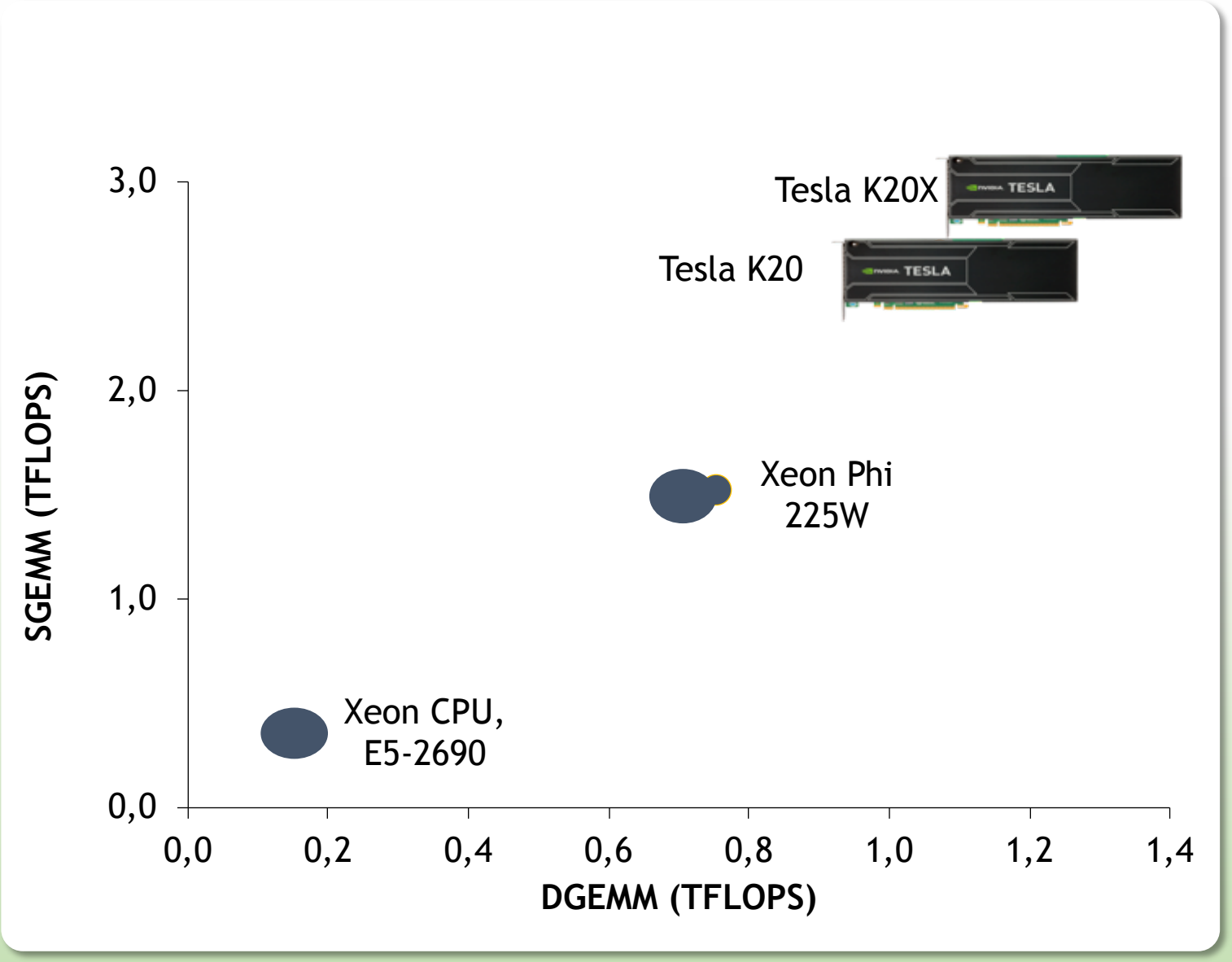


10x performance/socket

> 5x energy efficiency

Era of GPU-accelerated computing is here

World's Fastest, Most Energy Efficient Accelerator



Tesla K20X vs Xeon CPU

8x Faster SGEMM

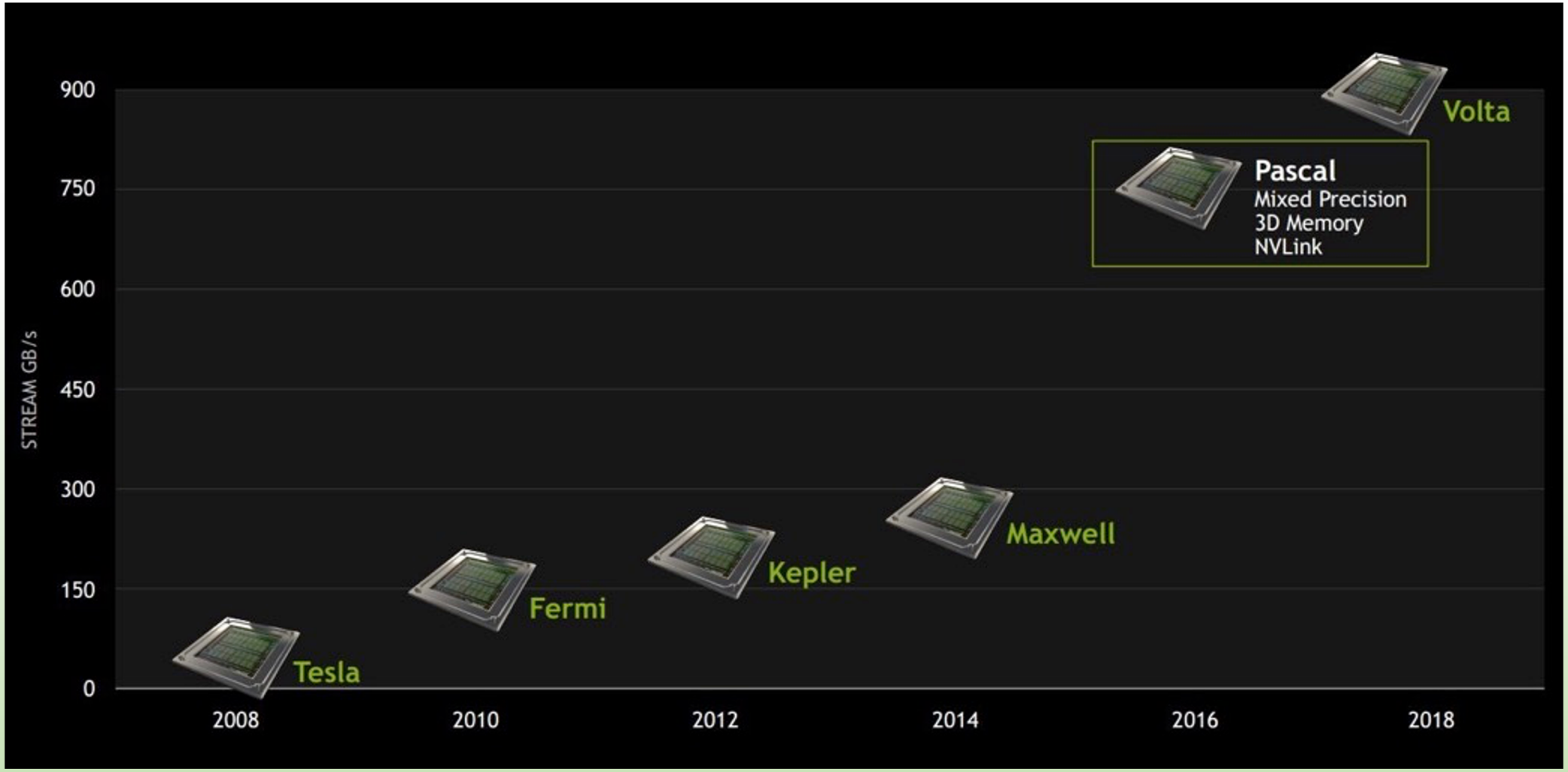
6x Faster DGEMM

Tesla K20X vs Xeon Phi

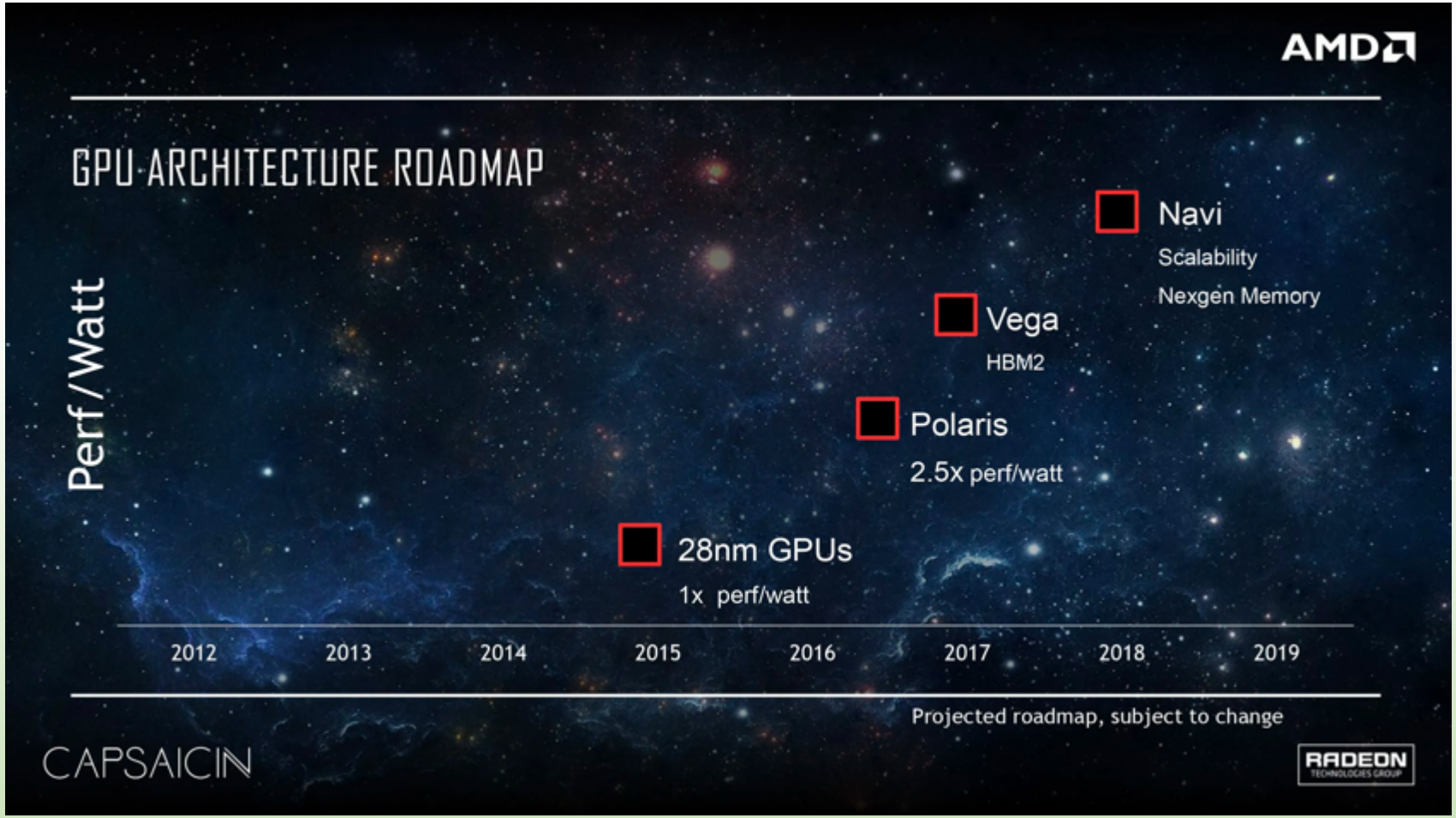
90% Faster SGEMM

60% Faster DGEMM

10 Years NVIDIA GPUs Development

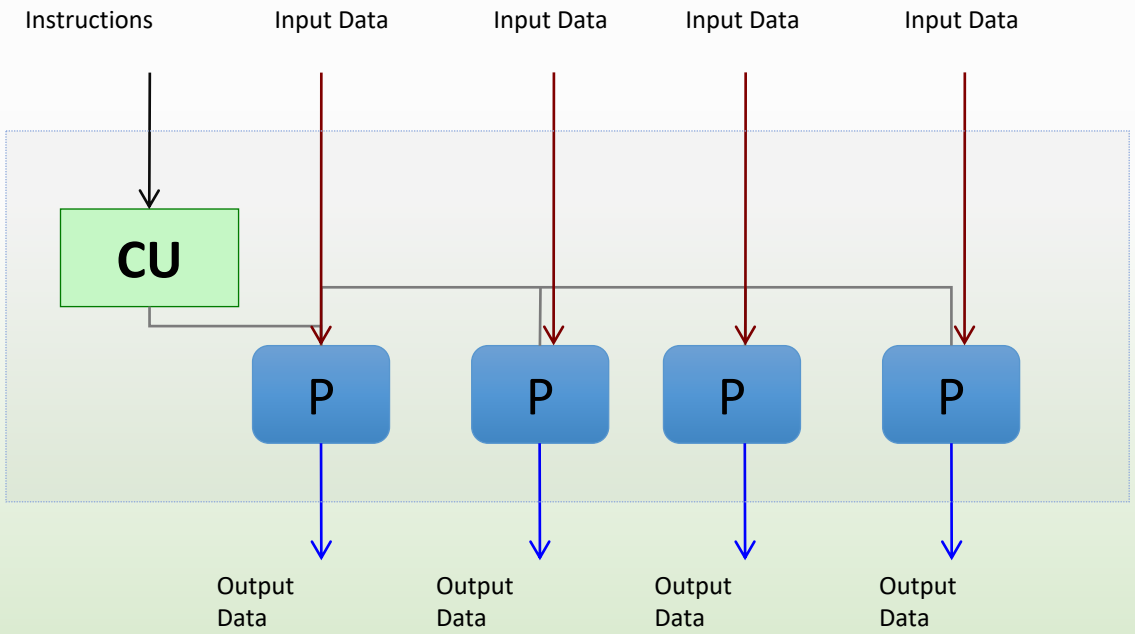


… and the 10 Years from AMD®



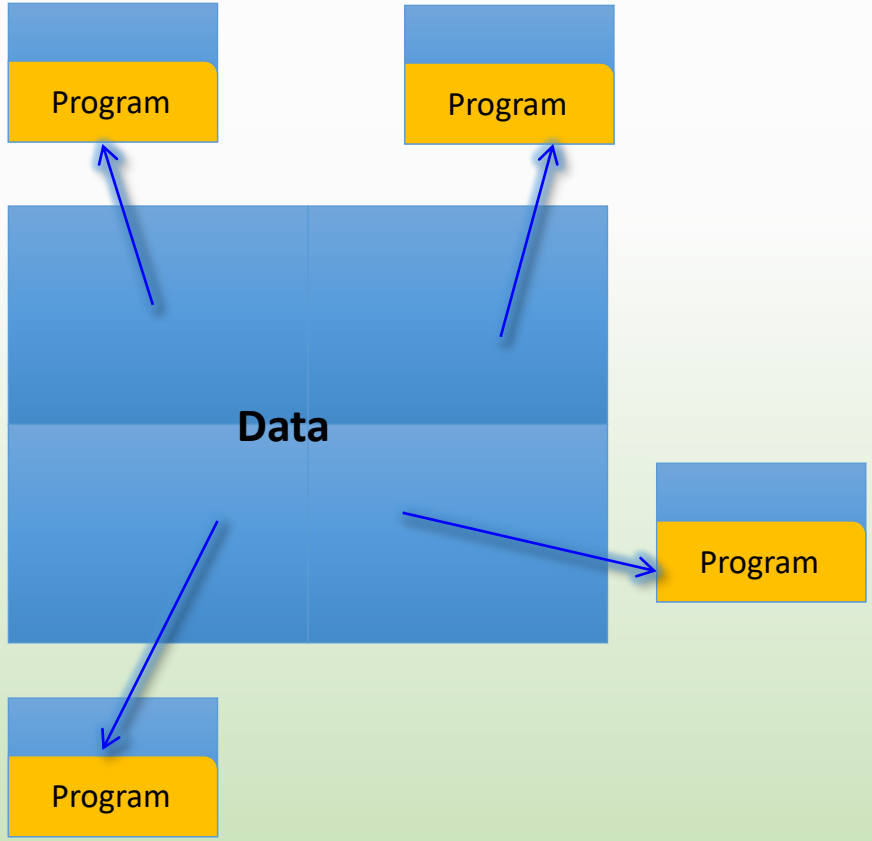
(Remember) Architectural Systems Facts (From Flynn's Taxonomy)

SPMD: Parallel Processing Units execute the same program on multiple parts of data



SIMD

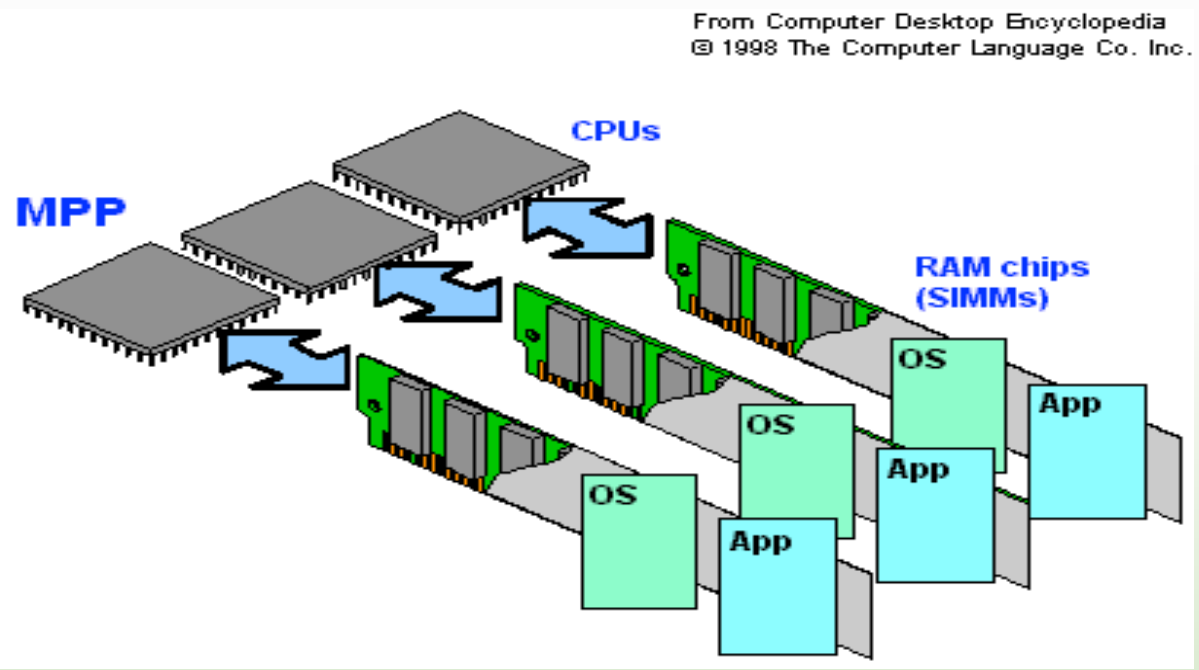
SIMD: All processors units are executing the same instructions in any instant.



Processor

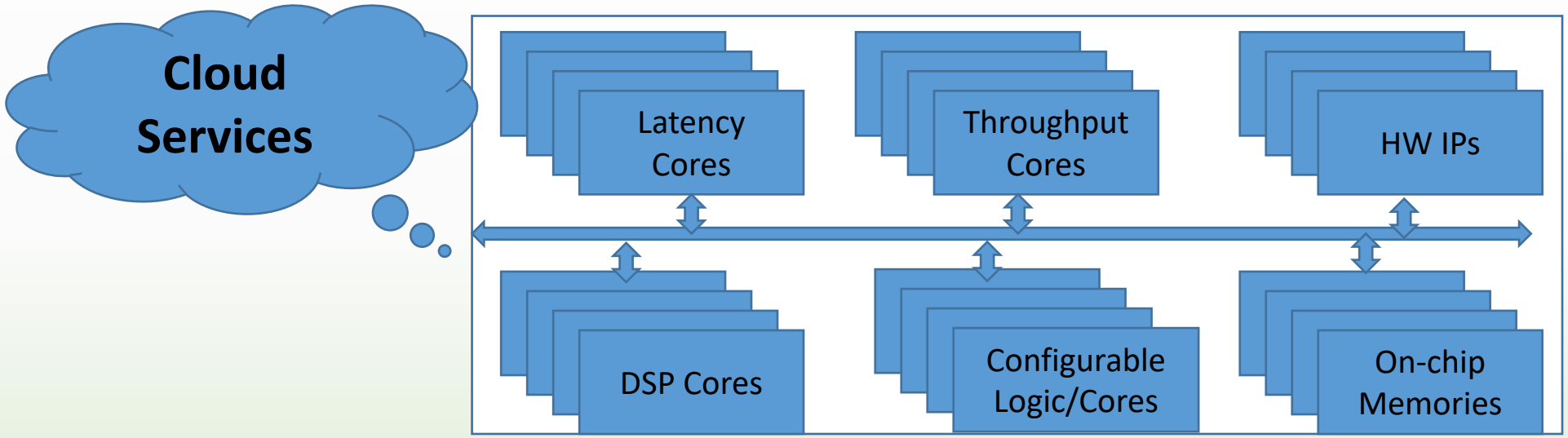
(Remember)Massive Parallel Processing (MPP)

- Computer system with many independent arithmetic units or entire microprocessors, that run in parallel
- MPPA is a MIMD (Multiple Instruction streams, Multiple Data) architecture, with distributed memory accessed locally, not shared globally

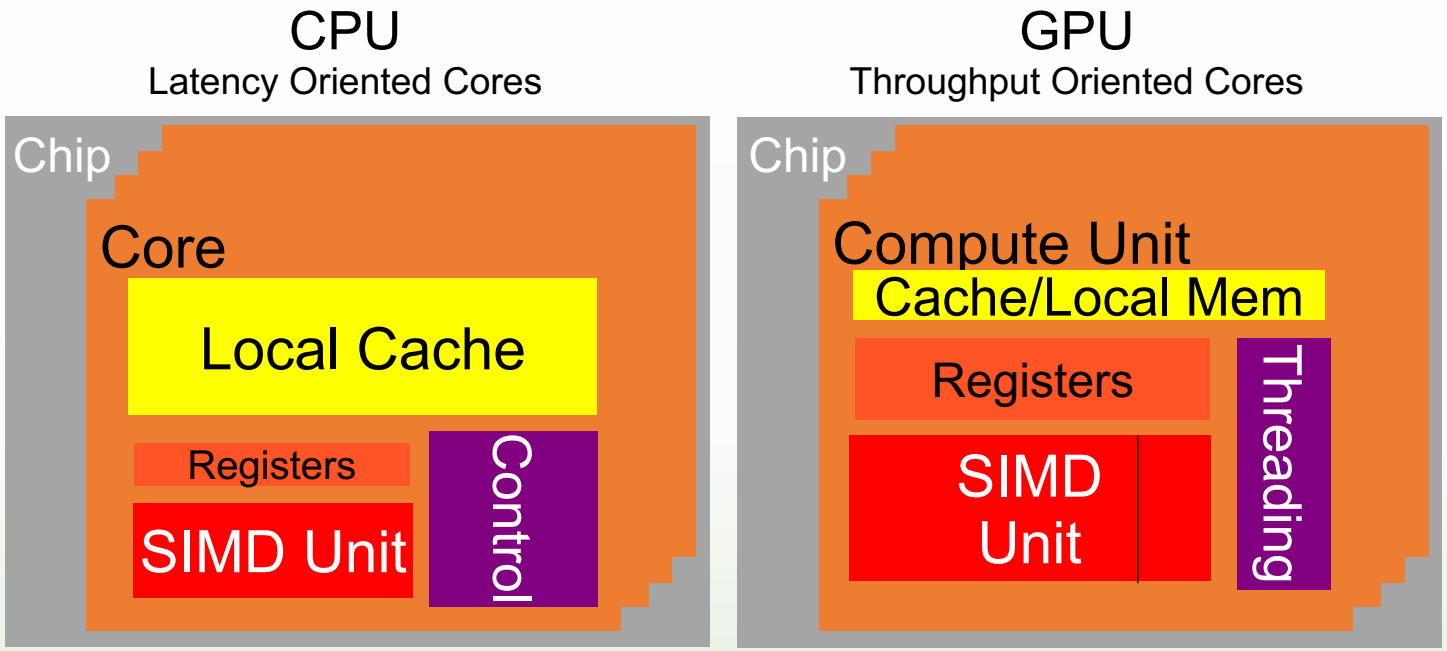


Heterogeneous Parallel Computing

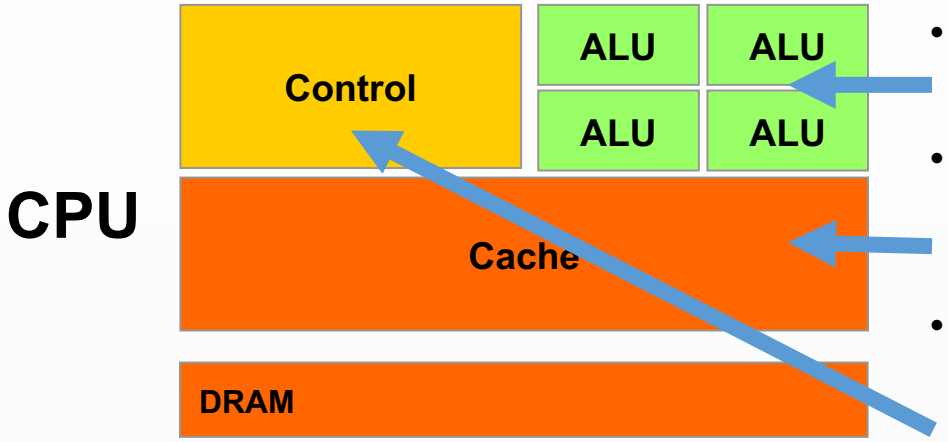
– Use the best match for the job (heterogeneity in mobile SOC)



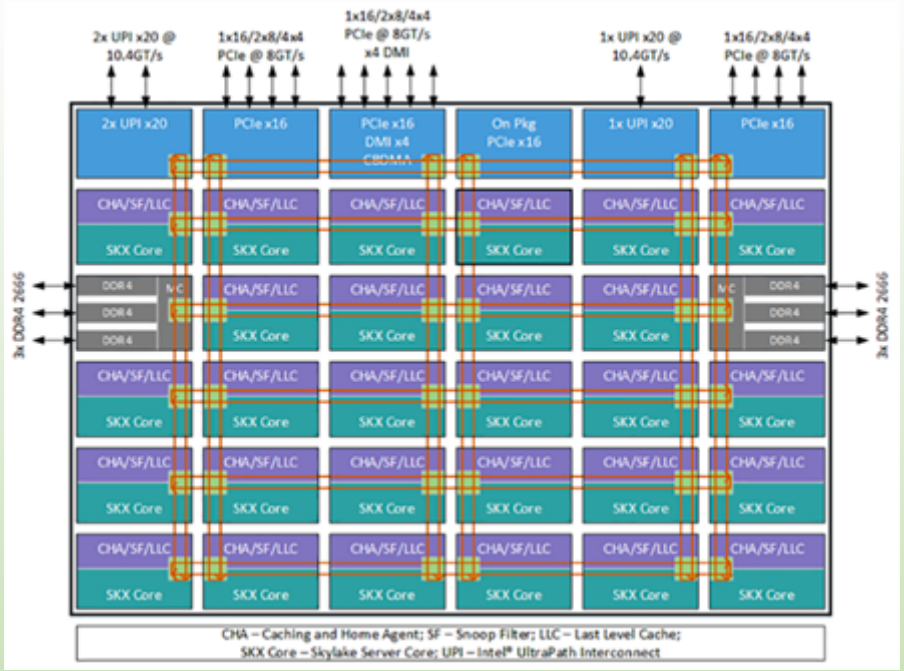
CPU and GPU are designed very differently



CPUs: Latency Oriented Design

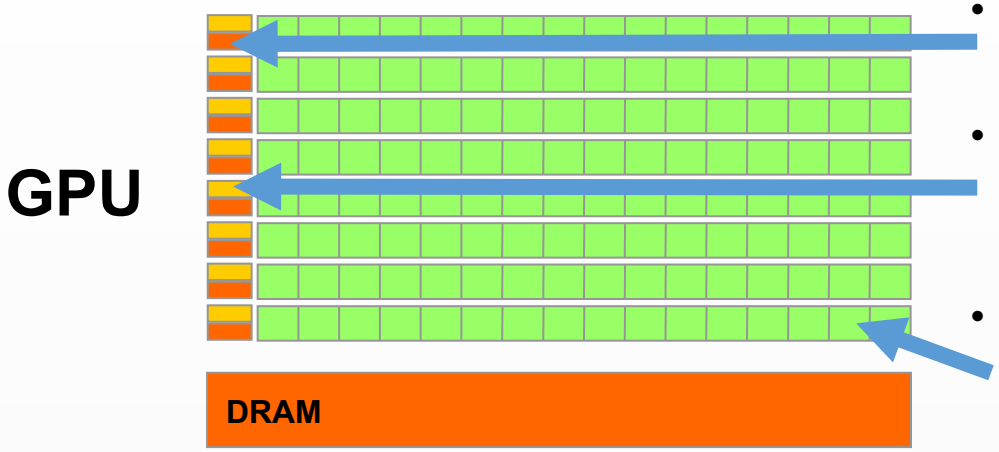


- Powerful ALU
  - Reduced operation latency
- Large caches
  - Convert long latency memory accesses to short latency cache accesses
- Sophisticated control
  - Branch prediction for reduced branch latency
  - Data forwarding for reduced data latency

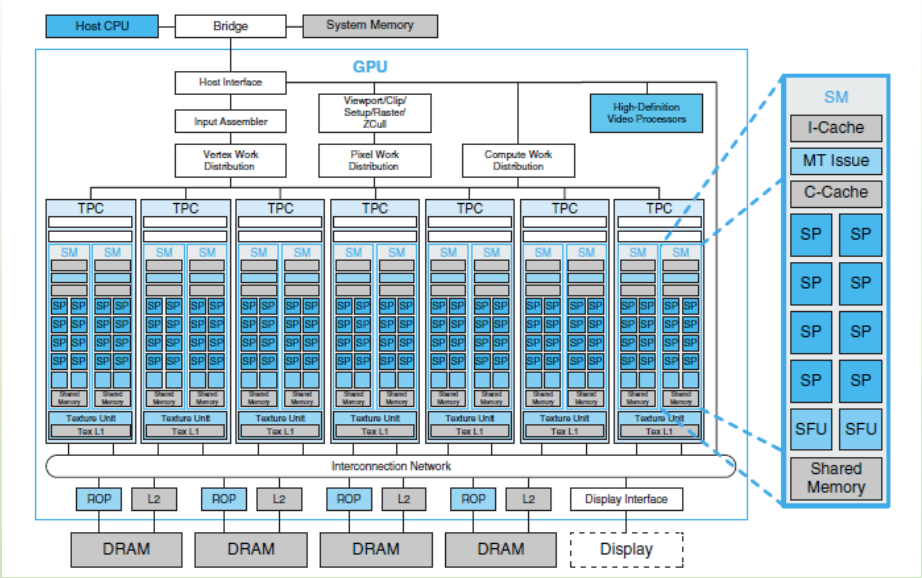


Intel Skylake Architecture

GPUs: Throughput Oriented Design



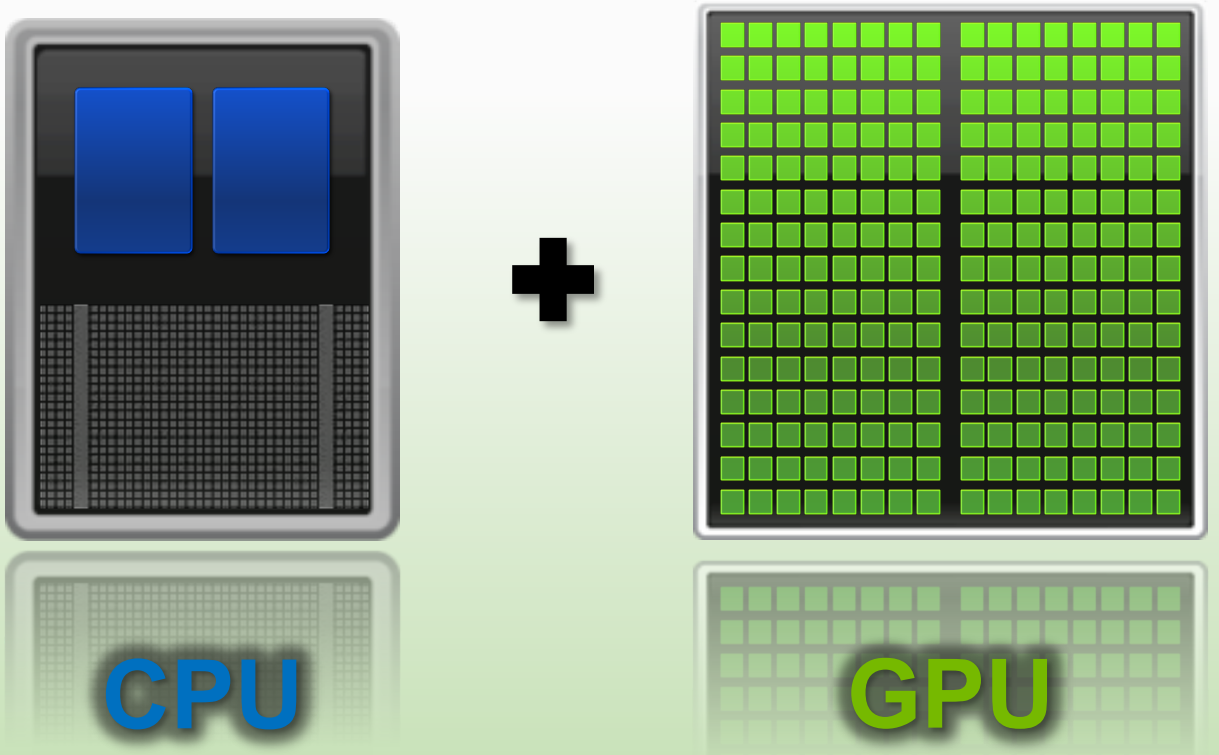
- Small caches
  - To boost memory throughput
- Simple control
  - No branch prediction
  - No data forwarding
- Energy efficient ALUs
  - Many, long latency but heavily pipelined for high throughput
- Require massive number of threads to tolerate latencies
  - Threading logic
  - Thread state



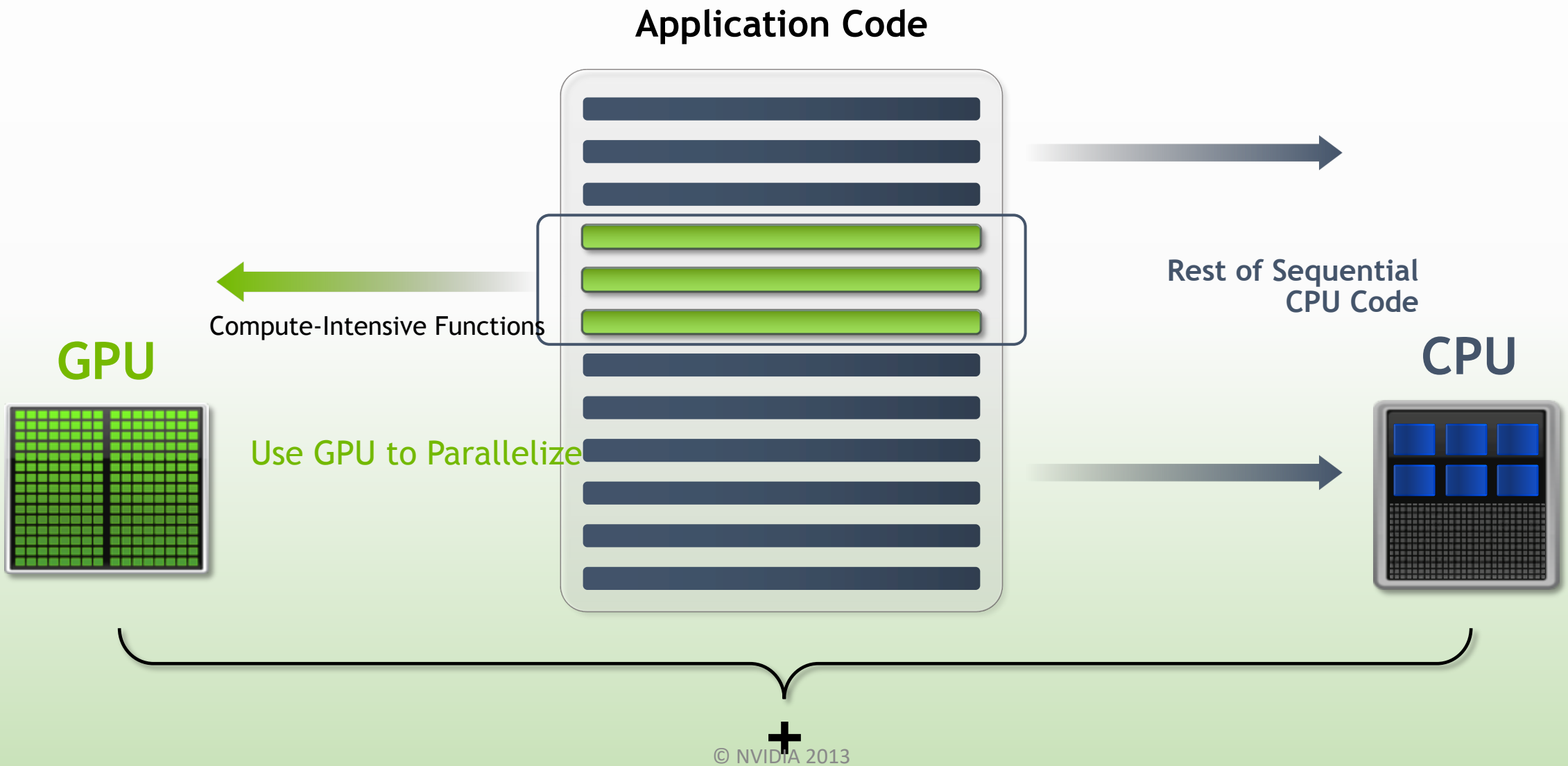
NVIDIA TESLA Architecture

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



Small Changes, Big Speed-up



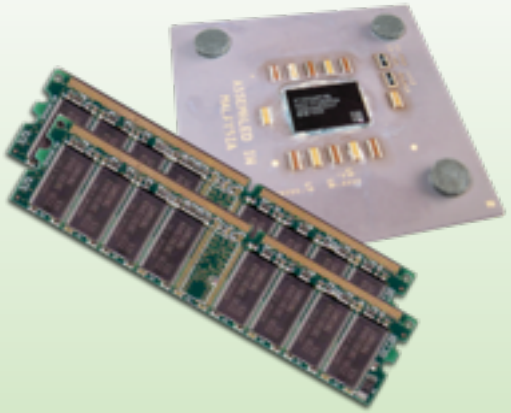
© NVIDIA 2013

NVIDIA TESLA™ Graphics and Computing Architecture Features

- TESLA™ shader processors are fully programmable
  - Large instructions memory
  - Cache Instructions
  - Logic Sequence Instructions
- TESLA™ to non-graphics programs:
  - Hierarchical Parallel Threads
  - Barrier Synchronization
  - Atomic Operators (Manage Highly Parallel Computing Work)

Heterogeneous Computing

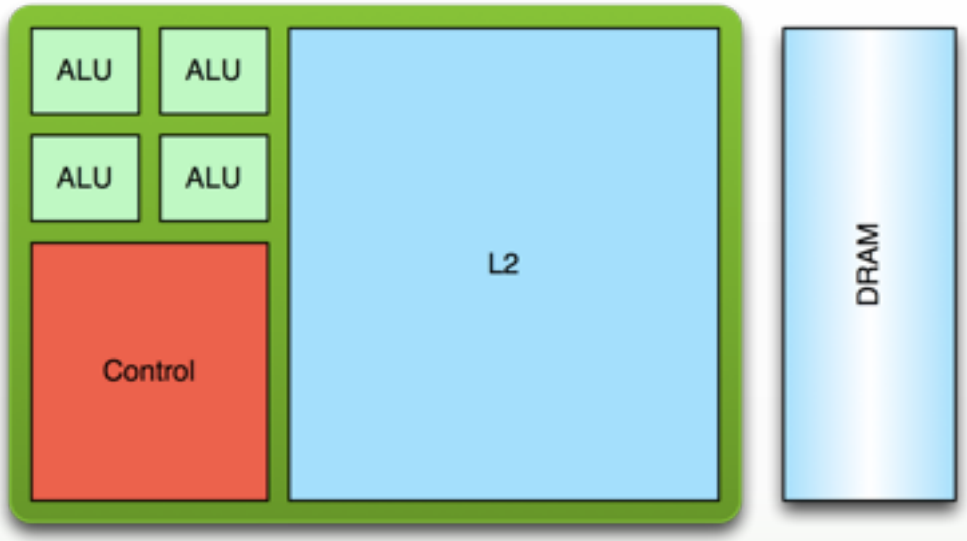
- Terminology:
  - ***Host*** **The CPU and its memory (host memory)**
  - ***Device*** **The GPU and its memory (device memory)**



Host

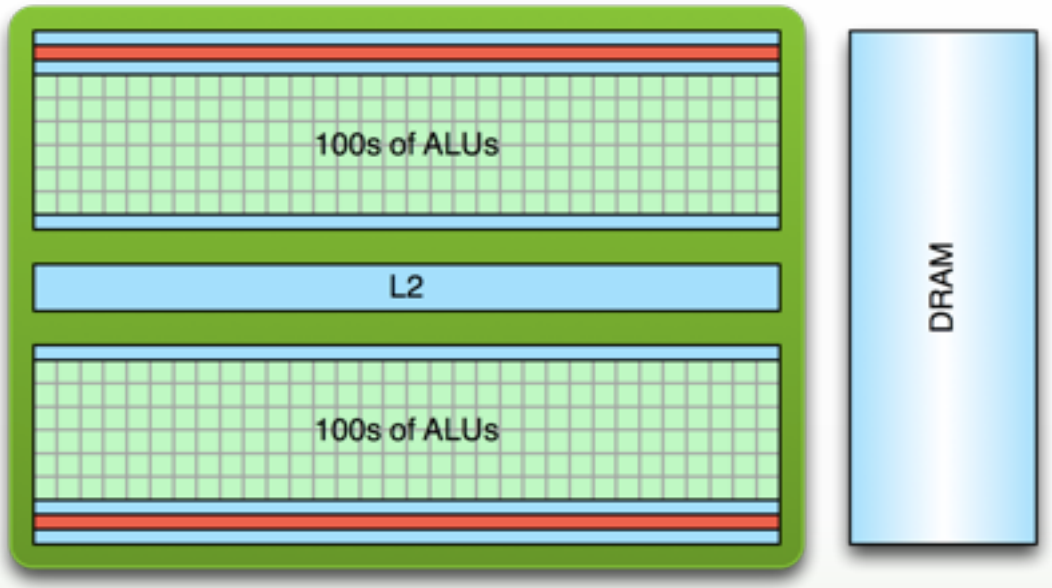
Device

Low Latency or High Throughput?



CPU

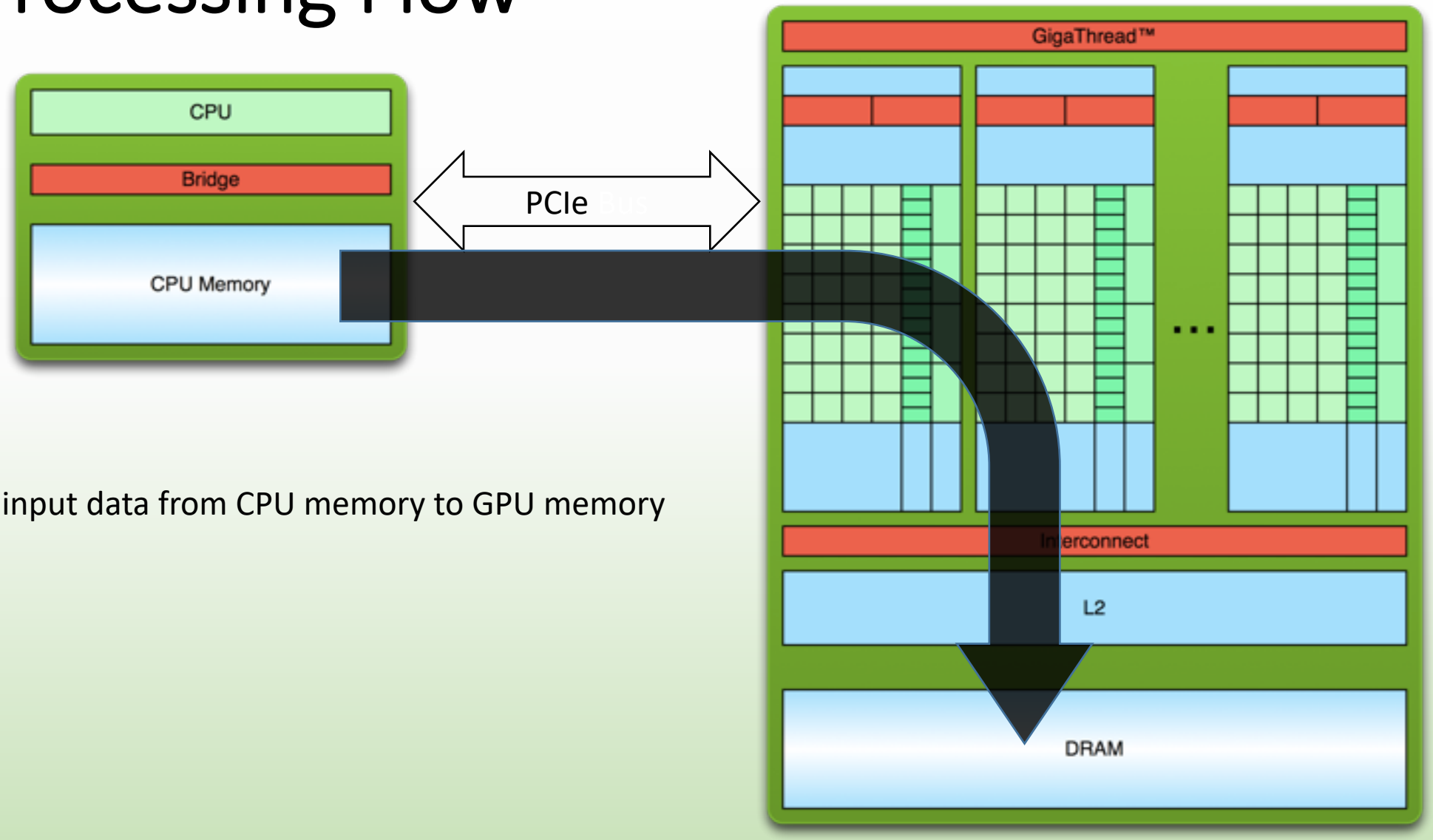
- Optimized for low-latency access to cached data sets
- Control logic for out-of-order and speculative execution



GPU

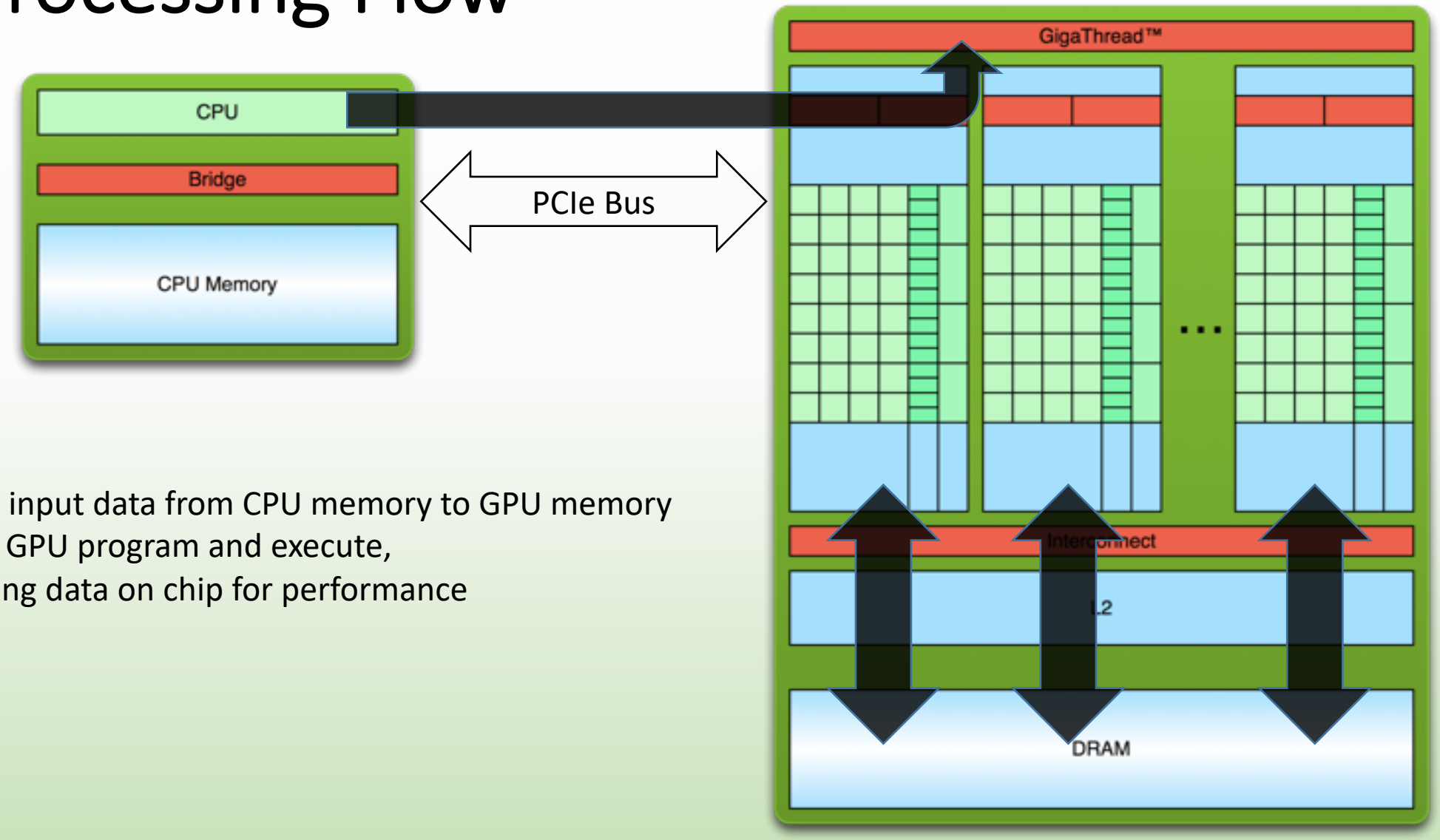
- Optimized for data-parallel, throughput computation
- Architecture tolerant of memory latency
- More transistors dedicated to computation

Processing Flow



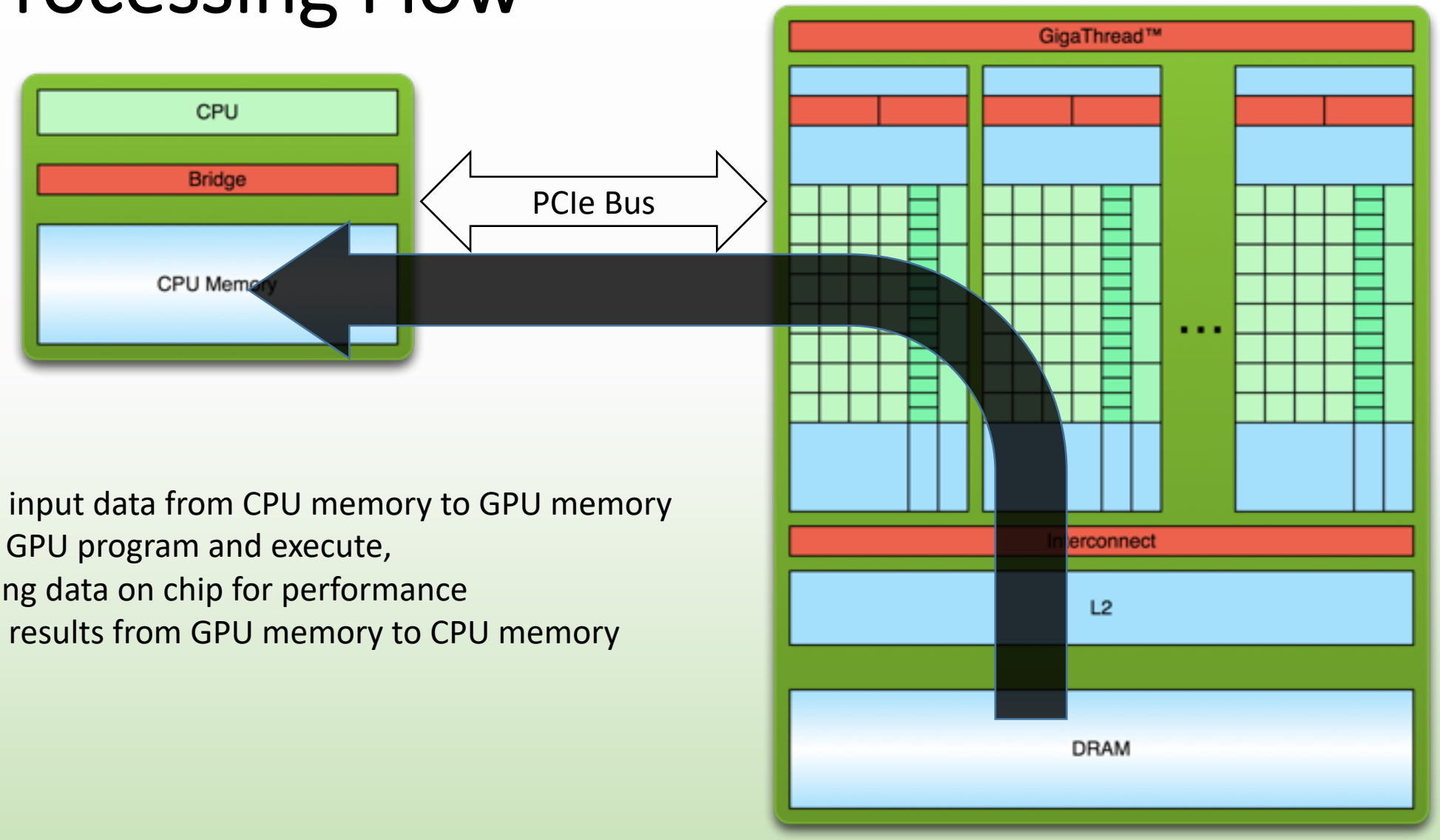
1. Copy input data from CPU memory to GPU memory

Processing Flow



1. Copy input data from CPU memory to GPU memory
2. Load GPU program and execute,
   caching data on chip for performance

Processing Flow

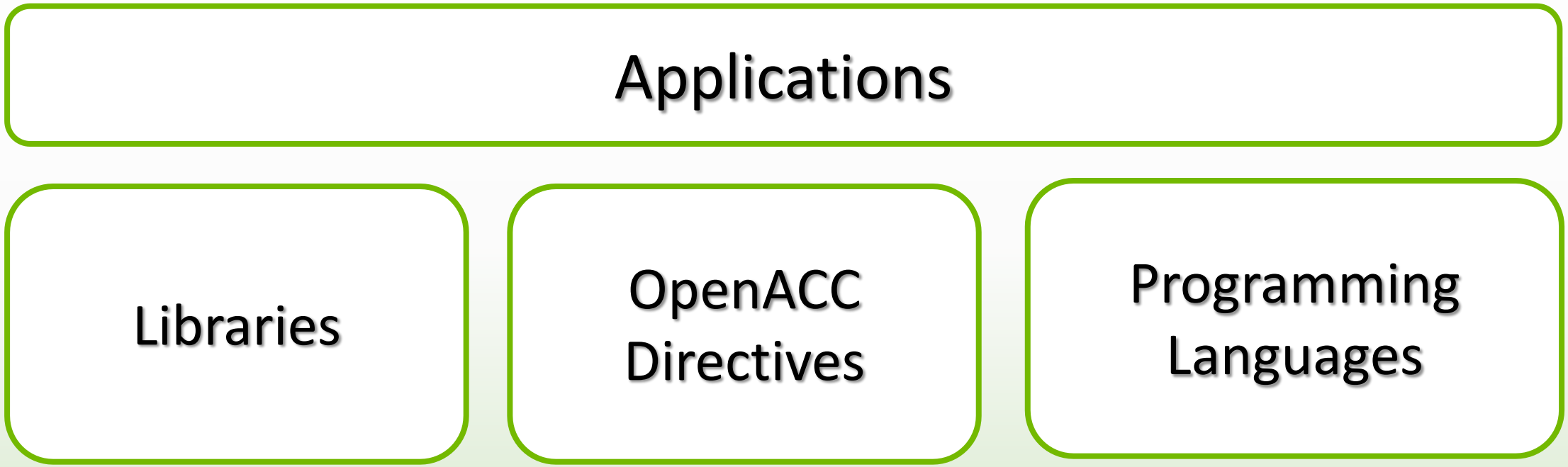


1. Copy input data from CPU memory to GPU memory
2. Load GPU program and execute,
   caching data on chip for performance
3. Copy results from GPU memory to CPU memory

Options to Develop GPUs Codes

- **CUDA®**
- OpenCL
- **OpenACC**
- OMPSs

3 Ways to Accelerate Applications



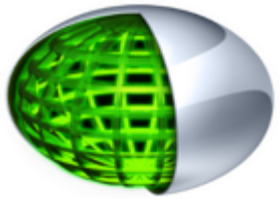
CUDA Parallel Computing Platform

www.nvidia.com/getcuda

Programming Approaches

- **Libraries** — "Drop-in" Acceleration
- **OpenACC Directives** — Easily Accelerate Apps
- **Programming Languages** — Maximum Flexibility

Development Environment



Nsight IDE
Linux, Mac and Windows
GPU Debugging and Profiling

CUDA-GDB debugger
NVIDIA Visual Profiler

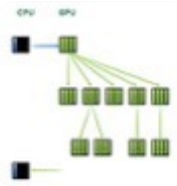
Open Compiler Tool Chain

Enables compiling new languages to CUDA platform, and CUDA languages to other architectures

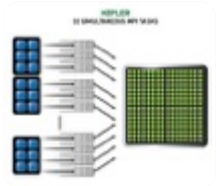
Hardware Capabilities

SMX

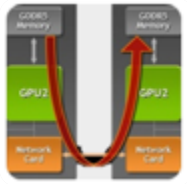
Dynamic Parallelism



HyperQ

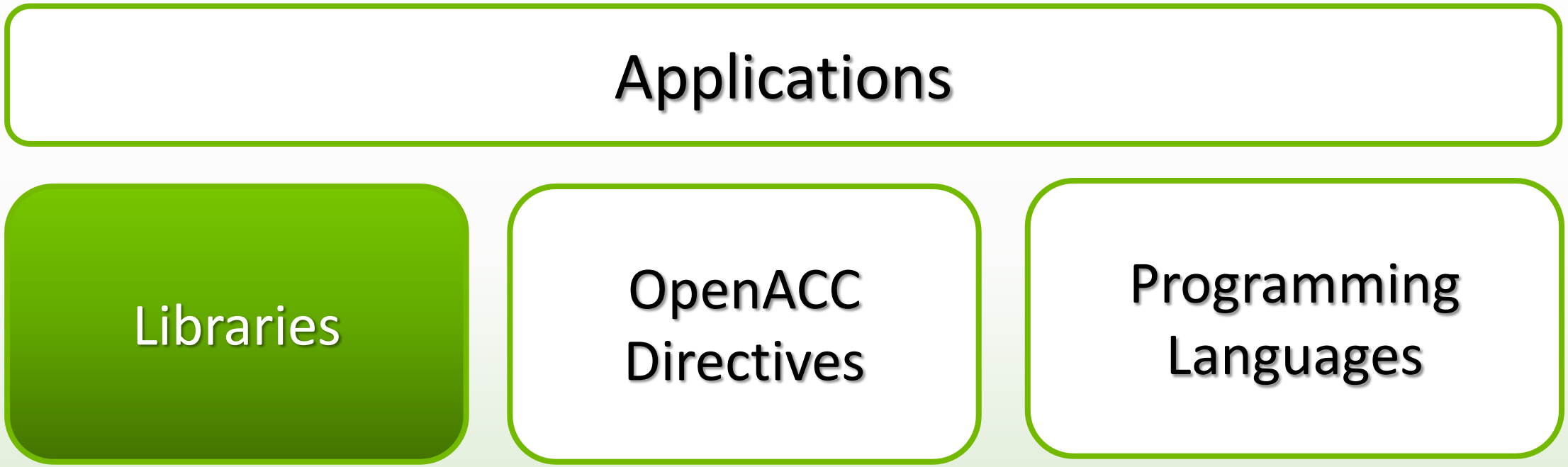


GPUDirect



© NVIDIA 2013

3 Ways to Accelerate Applications



© NVIDIA 2013

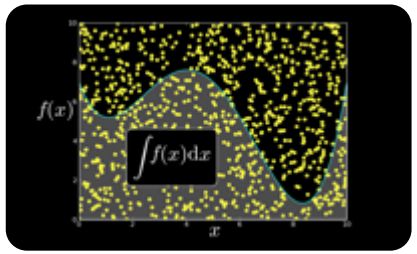
Libraries: Easy, High-Quality Acceleration

- **Ease of use:** Using libraries enables GPU acceleration without in-depth knowledge of GPU programming
- **"Drop-in":** Many GPU-accelerated libraries follow standard APIs, thus enabling acceleration with minimal code changes
- **Quality:** Libraries offer high-quality implementations of functions encountered in a broad range of applications
- **Performance:** NVIDIA libraries are tuned by experts

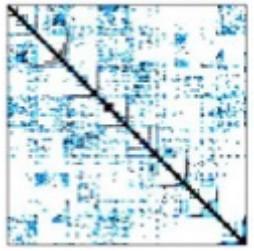
© NVIDIA 2013

Some GPU-accelerated Libraries

NVIDIA cuBLAS



NVIDIA cuRAND



NVIDIA cuSPARSE

NVIDIA NPP

GPU VSIPL

Vector Signal Image Processing

CULA tools

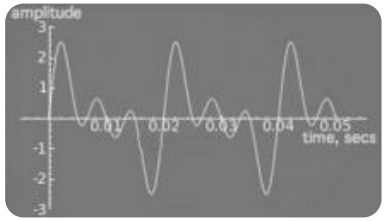
GPU Accelerated Linear Algebra

MAGMA

ICL UT

Matrix Algebra on GPU and Multicore

open source initiative



NVIDIA cuFFT

ROGUE WAVE SOFTWARE

IMSL Library

ArrayFire Matrix Computations

CUSP

Sparse Linear Algebra

open source initiative

Thrust

C++ STL Features for CUDA

open source initiative

© NVIDIA 2013

3 Steps to CUDA-accelerated application

- **Step 1:** Substitute library calls with equivalent CUDA library calls

```
saxpy ( … )          cublasSaxpy ( … )
```

- **Step 2:** Manage data locality

  - with CUDA: `cudaMalloc(), cudaMemcpy(), etc.`
  - with CUBLAS: `cublasAlloc(), cublasSetVector(), etc.`

- **Step 3:** Rebuild and link the CUDA-accelerated library

```
nvcc myobj.o –l cublas
```

© NVIDIA 2013

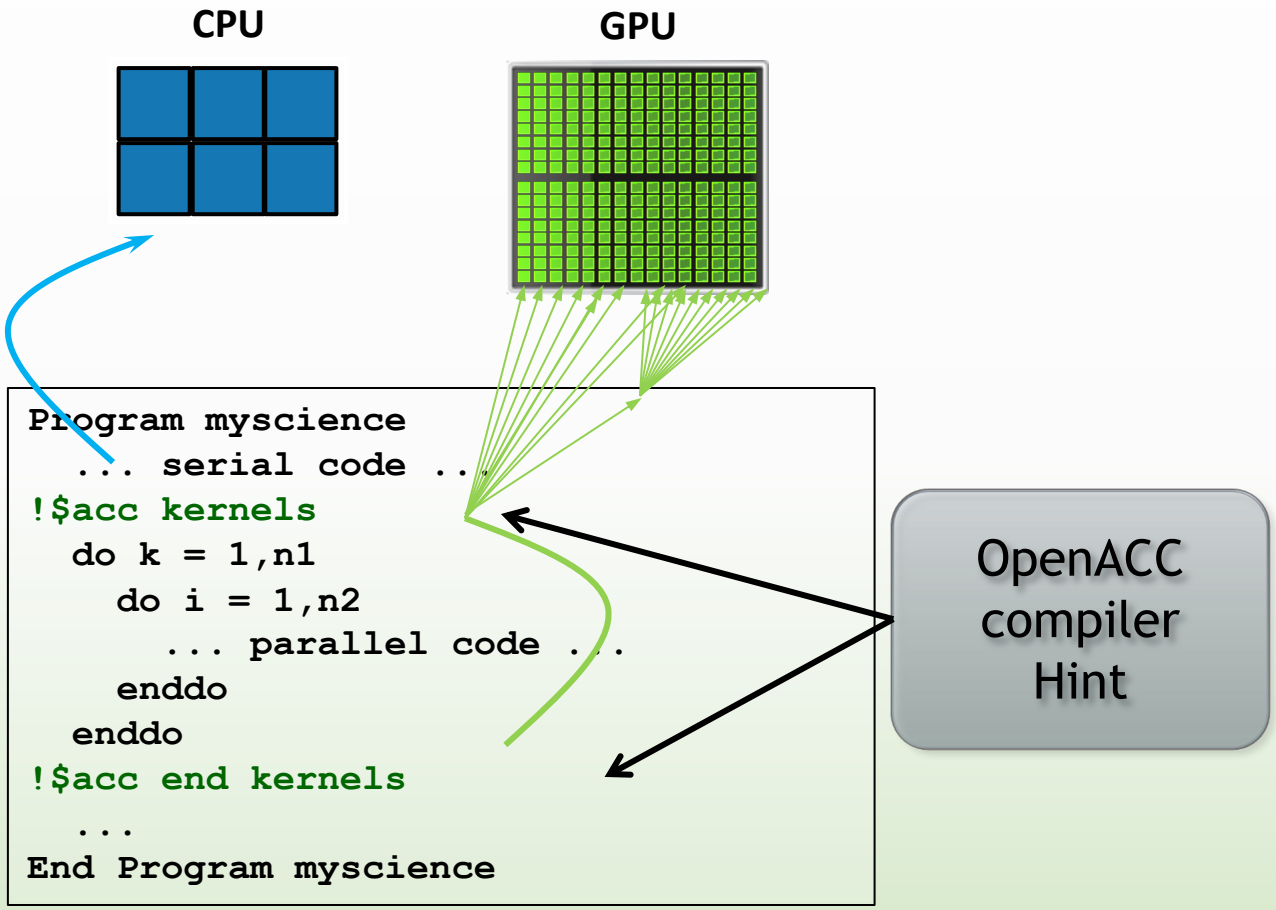
Explore the CUDA (Libraries) Ecosystem

- CUDA Tools and Ecosystem described in detail on NVIDIA Developer Zone: developer.nvidia.com/cuda-tools-ecosystem
- Remember your nVIDIA developer account

3 Ways to Accelerate Applications

© NVIDIA 2013

OpenACC Directives



Simple Compiler hints

Compiler Parallelizes code

Works on many-core GPUs & multicore CPUs

© NVIDIA 2013

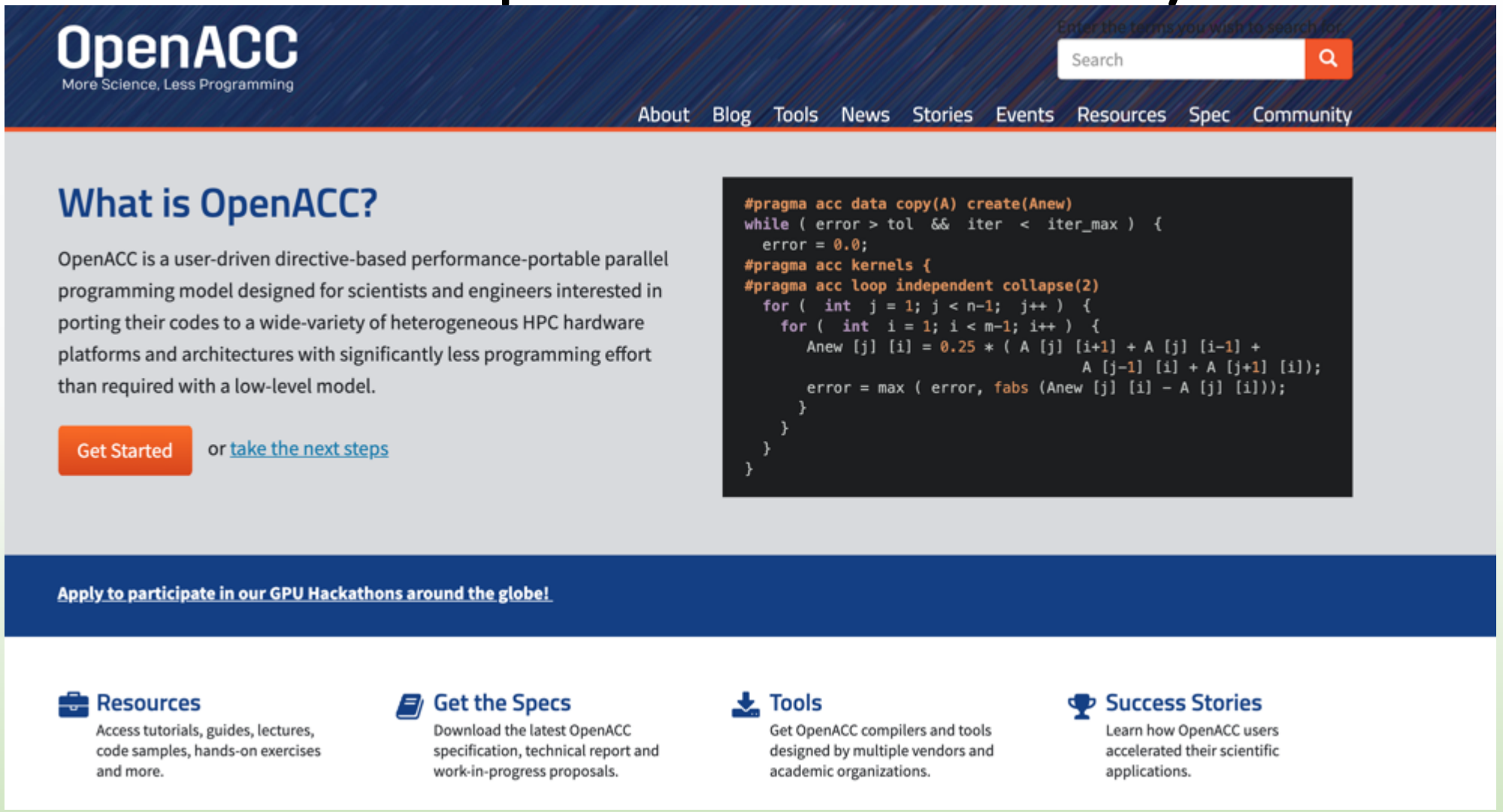
OpenACC

The Standard for GPU Directives

- **Easy:** Directives are the easy path to accelerate compute intensive applications
- **Open:** OpenACC is an open GPU directives standard, making GPU programming straightforward and portable across parallel and multi-core processors
- **Powerful:** GPU Directives allow complete access to the massive parallel power of a GPU

© NVIDIA 2013

Join the OpenACC community!

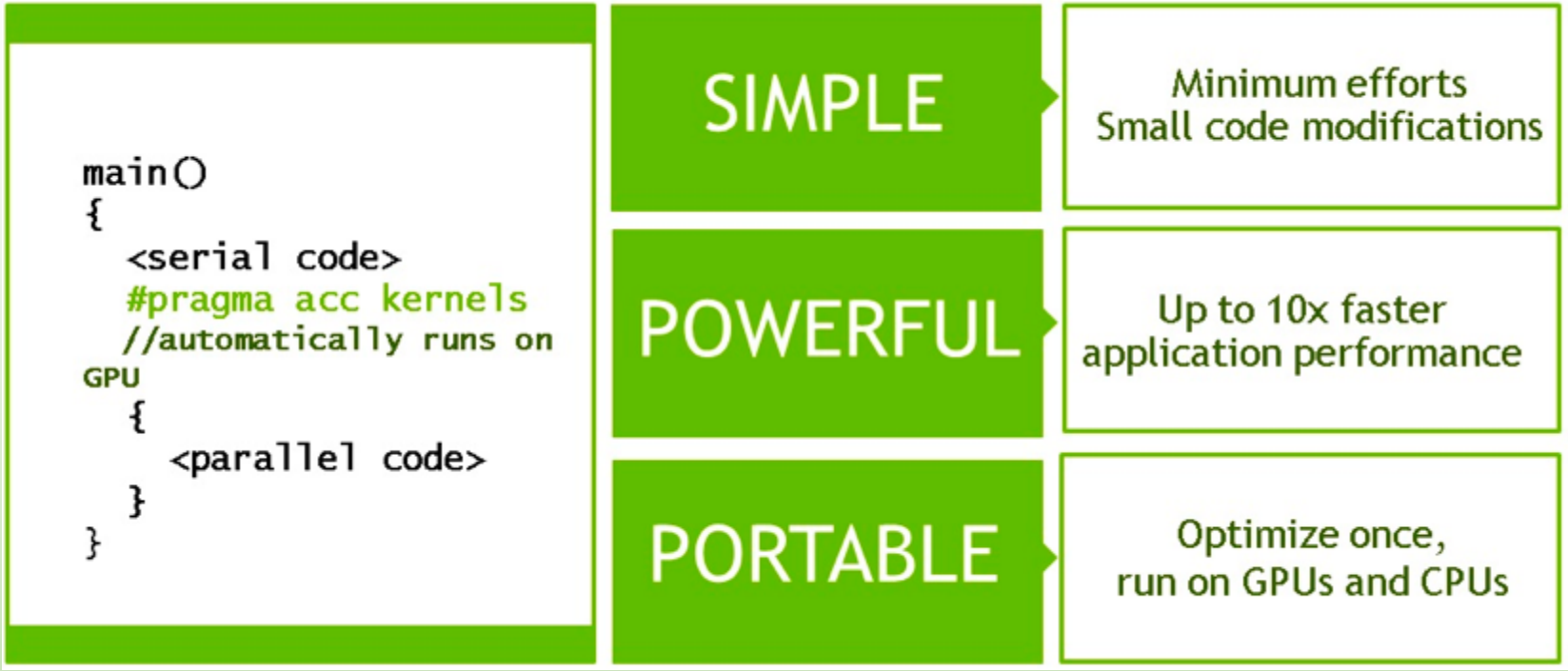


https://developer.nvidia.com/openacc
https://www.openacc.org/

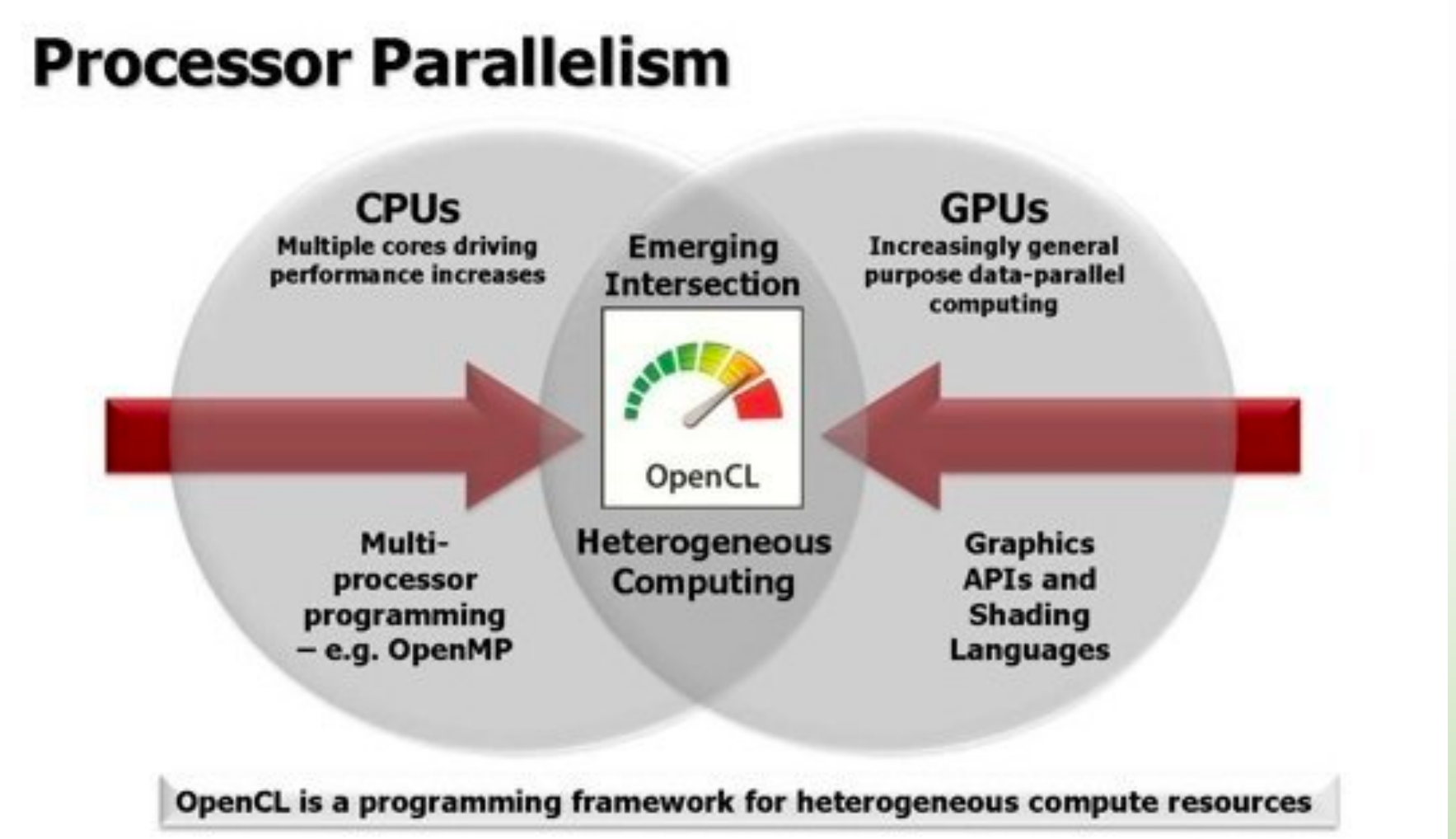
OpenACC

More Science, Less Programming

The main goal of OpenACC is to be simple



What About OpenCL?



Some Characteristics

Programming and Runtime Framework for Application Acceleration

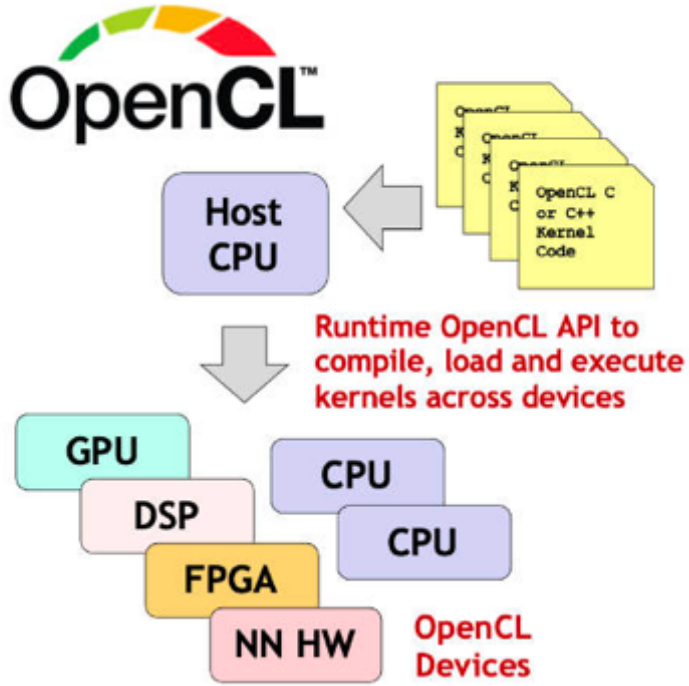
Offload compute-intensive kernels
onto parallel heterogeneous processors
CPUs, GPUs, DSPs, FPGAs, Tensor Processors
OpenCL C or C++ kernel languages

OpenCL APIs to Enable Kernel Execution

Platform Layer API
Query, select and initialize compute devices
Runtime API
Build and execute kernels programs
on multiple devices

Explicit Application Control

Which programs execute on what device
Where data is stored in memories in the system
When programs are run, and what operations are
dependent on earlier operations

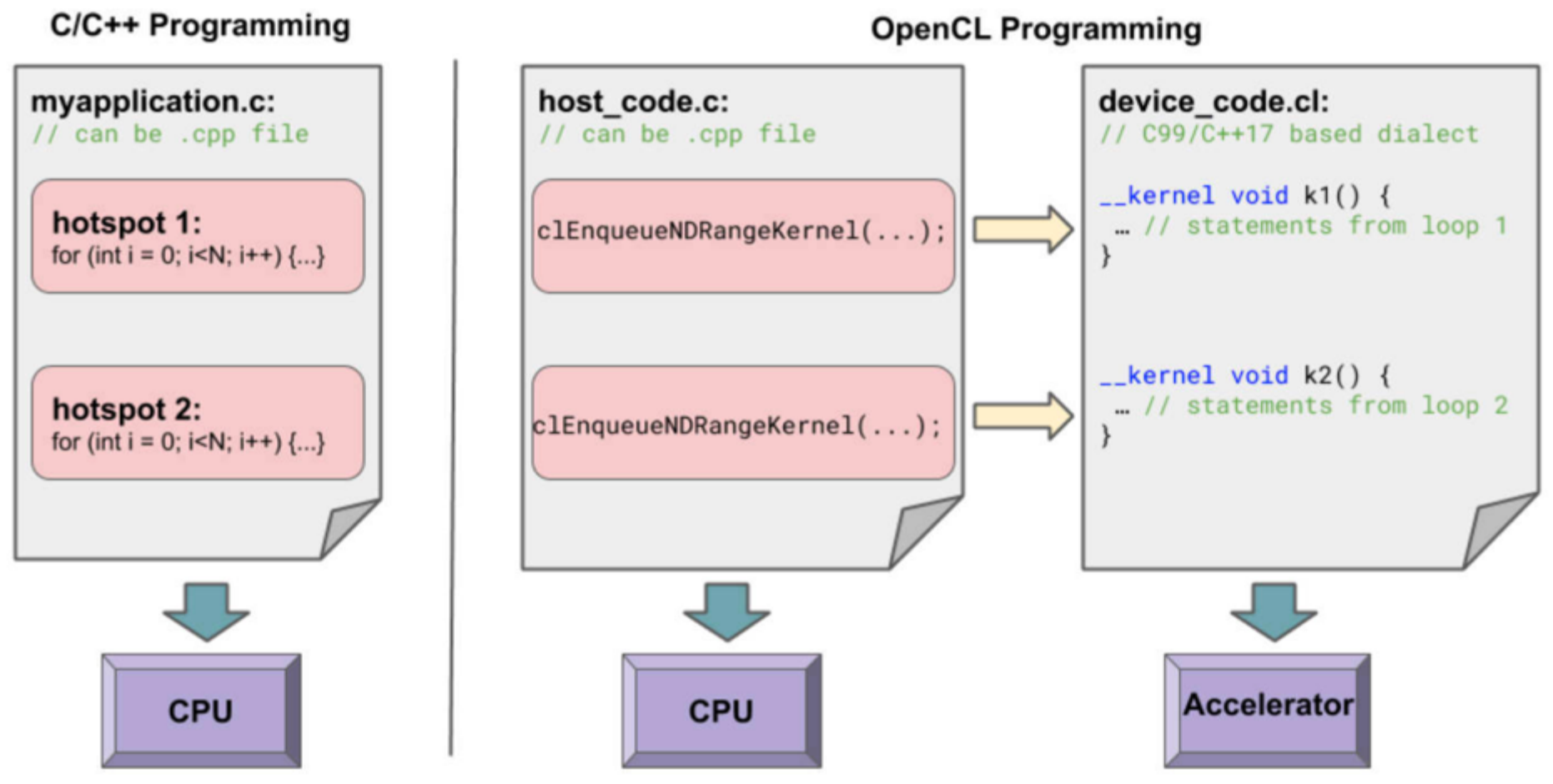


Complements GPU-only APIs

Simpler programming model
Relatively lightweight run-time
More language flexibility, e.g. pointers
Rigorously defined numeric precision

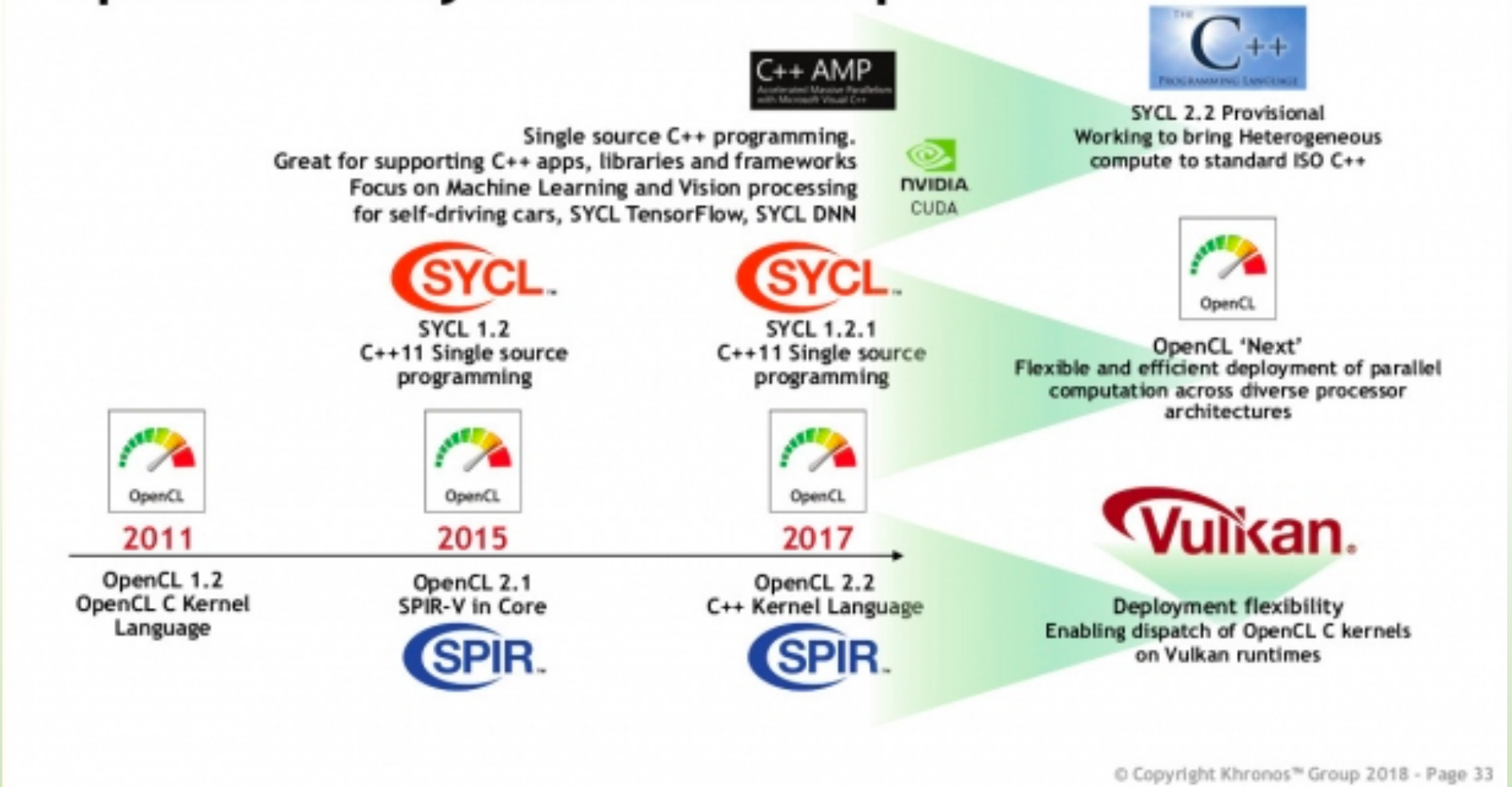
More information at: https://www.khronos.org/opencl/

OpenCL Programming Paradigm



More information at: https://www.khronos.org/opencl/

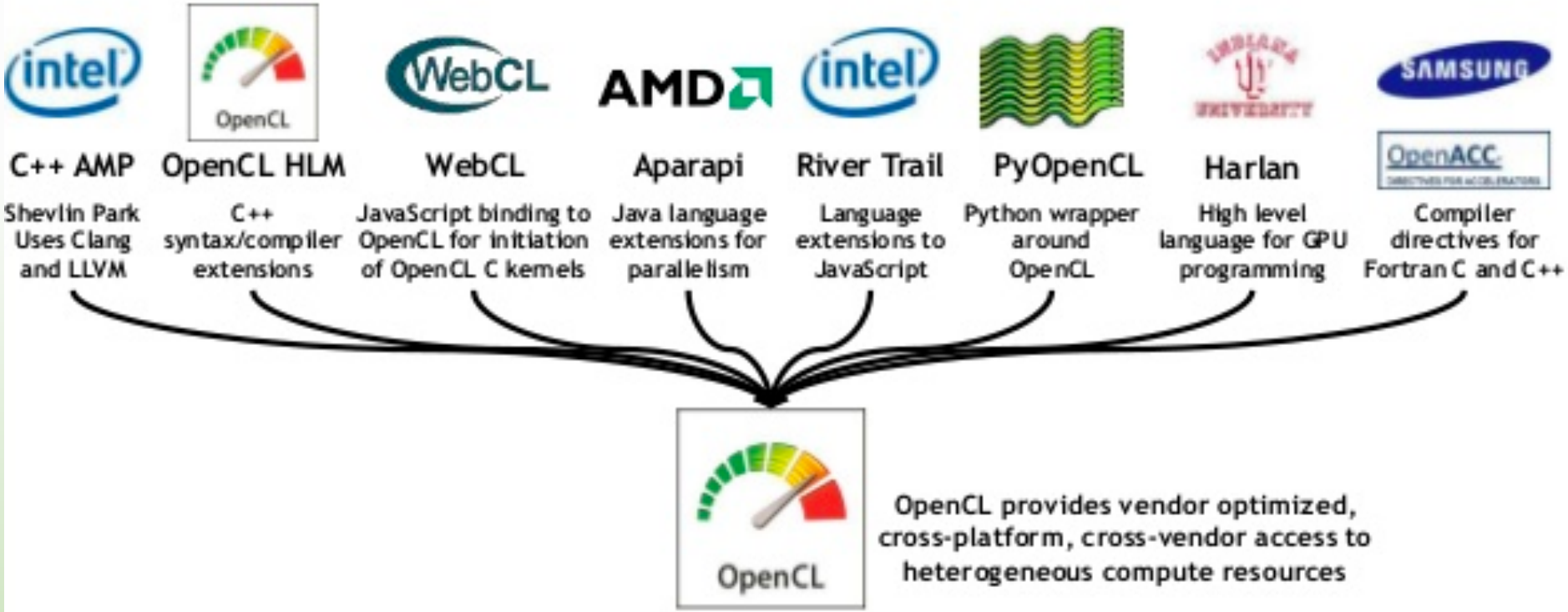
OpenCL Ecosystem Roadmap



More information at: https://www.khronos.org/opencl/

OpenCL as Parallel Compute Foundation

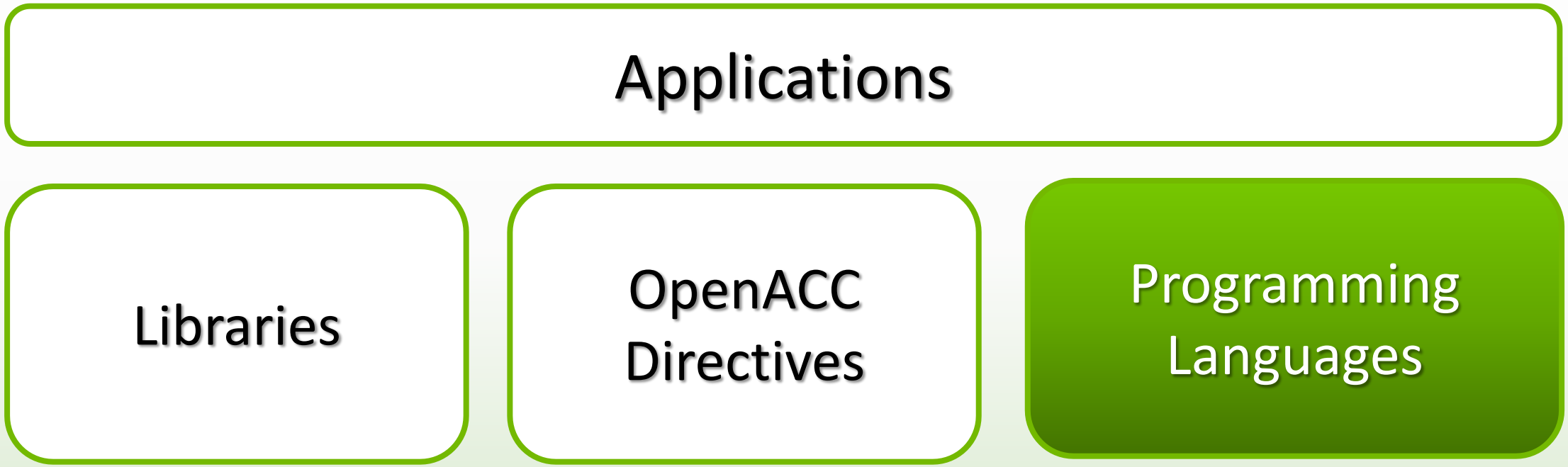
- 100+ tool chains and languages leveraging OpenCL
  - Heterogeneous solutions emerging for the most popular programming languages



© Copyright Khronos Group 2013 - Page 16

More information at: https://www.khronos.org/opencl/

3 Ways to Accelerate Applications



© NVIDIA 2013

GPU Programming Languages

Numerical analytics	MATLAB, Mathematica, LabVIEW
Fortran	OpenACC, CUDA Fortran
C	OpenACC, CUDA C
C++	Thrust, CUDA C++
Python	PyCUDA, Copperhead
F#	Alea.cuBase

© NVIDIA 2013

Rapid Parallel C++ Development

- Resembles C++ STL
- High-level interface
  - Enhances developer productivity
  - Enables performance portability between GPUs and multicore CPUs
- Flexible
  - CUDA, OpenMP, and TBB backends
  - Extensible and customizable
  - Integrates with existing software
- Open source

```
// generate 32M random numbers on host
thrust::host_vector<int> h_vec(32 << 20);
thrust::generate(h_vec.begin(),
                 h_vec.end(),
                 rand);

// transfer data to device (GPU)
thrust::device_vector<int> d_vec = h_vec;

// sort data on device
thrust::sort(d_vec.begin(), d_vec.end());

// transfer data back to host
thrust::copy(d_vec.begin(),
             d_vec.end(),
             h_vec.begin());
```

http://developer.nvidia.com/thrust

Learn More

These languages are supported on all CUDA-capable GPUs.

You might already have a CUDA-capable GPU in your laptop or desktop PC!

CUDA C/C++
http://developer.nvidia.com/cuda-toolkit

Thrust C++ Template Library
http://developer.nvidia.com/thrust

CUDA Fortran
http://developer.nvidia.com/cuda-toolkit

PyCUDA (Python)
http://mathema.tician.de/software/pycuda

GPU.ORG Different Ressources
http://gpgpu.org

MATLAB
http://www.mathworks.com/discovery/matlab-gpu.html

Mathematica
http://www.wolfram.com/mathematica/new-in-8/cuda-and-opencl-support/ or
http://www.wolfram.com/gridmathematica/

Getting Started

NVIDIA DEEP LEARNING INSTITUTE
www.nvidia.com/dli

- Download CUDA Toolkit & SDK: https://developer.nvidia.com/cuda-downloads
- Nsight IDE (Eclipse or Visual Studio): http://www.nvidia.com/object/nsight.html

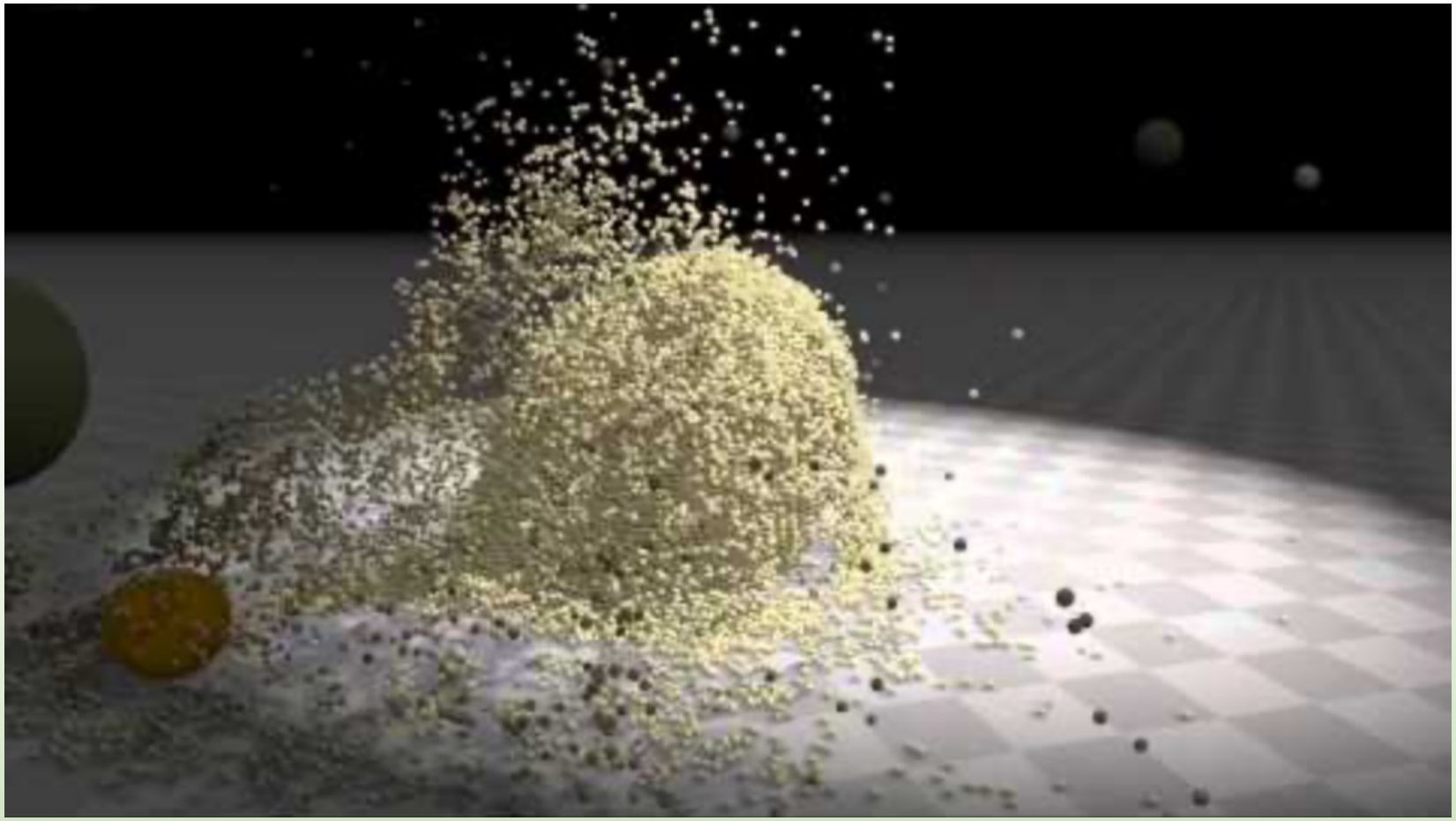
- General GPU Computing Community: http://gpgpu.org/

- Programming Guide/Best Practices:
  - docs.nvidia.com

- Questions:
  - NVIDIA Developer forums: devtalk.nvidia.com
  - Search or ask on: www.stackoverflow.com/tags/cuda

- Developer Community: https://developer.nvidia.com/ (Join Now)
- OpenACC Community: https://www.openacc.org/

Thank you and Enjoy!
@carlosjaimebh



NVIDIA DEEP LEARNING INSTITUTE

www.nvidia.com/dli