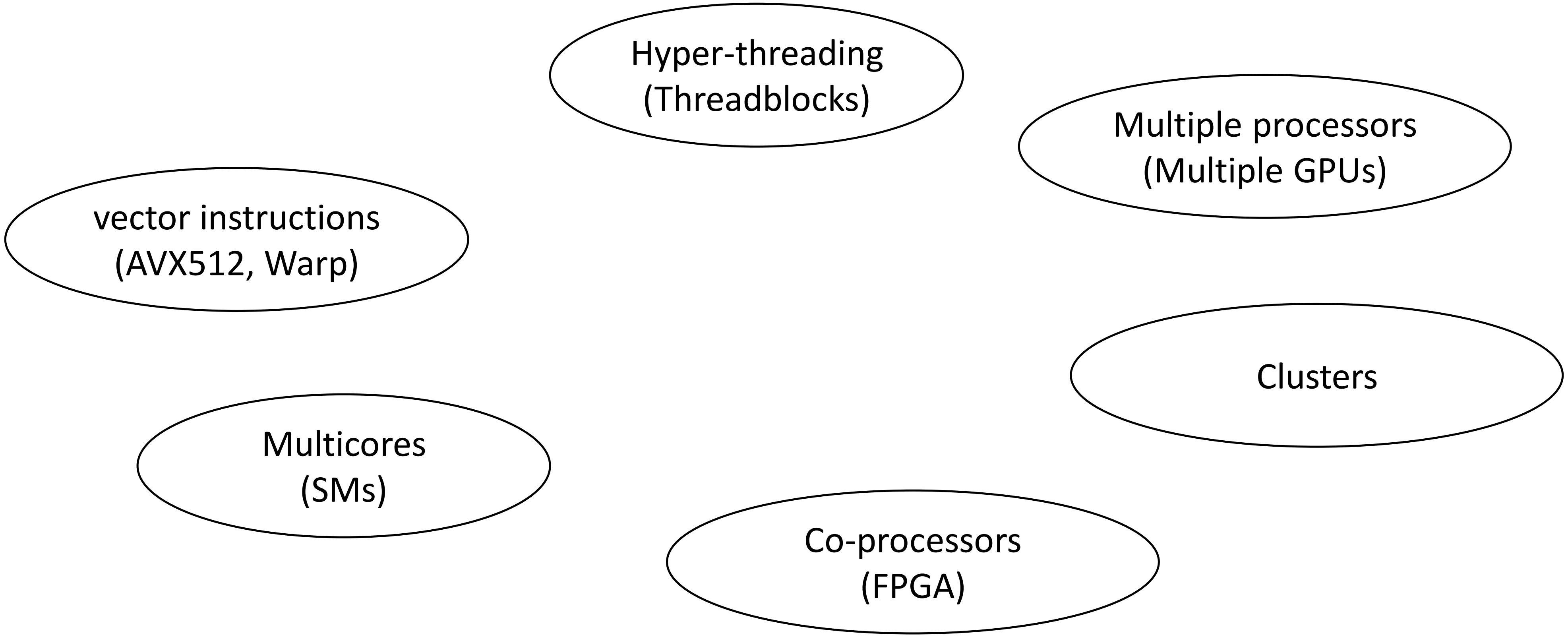
Introduction

ECE 285 GPU Programming

Cheolhong An

Parallel computing is everywhere even in the your phone



Hyper-threading (Threadblocks)

Multiple processors (Multiple GPUs)

vector instructions (AVX512, Warp)

Clusters

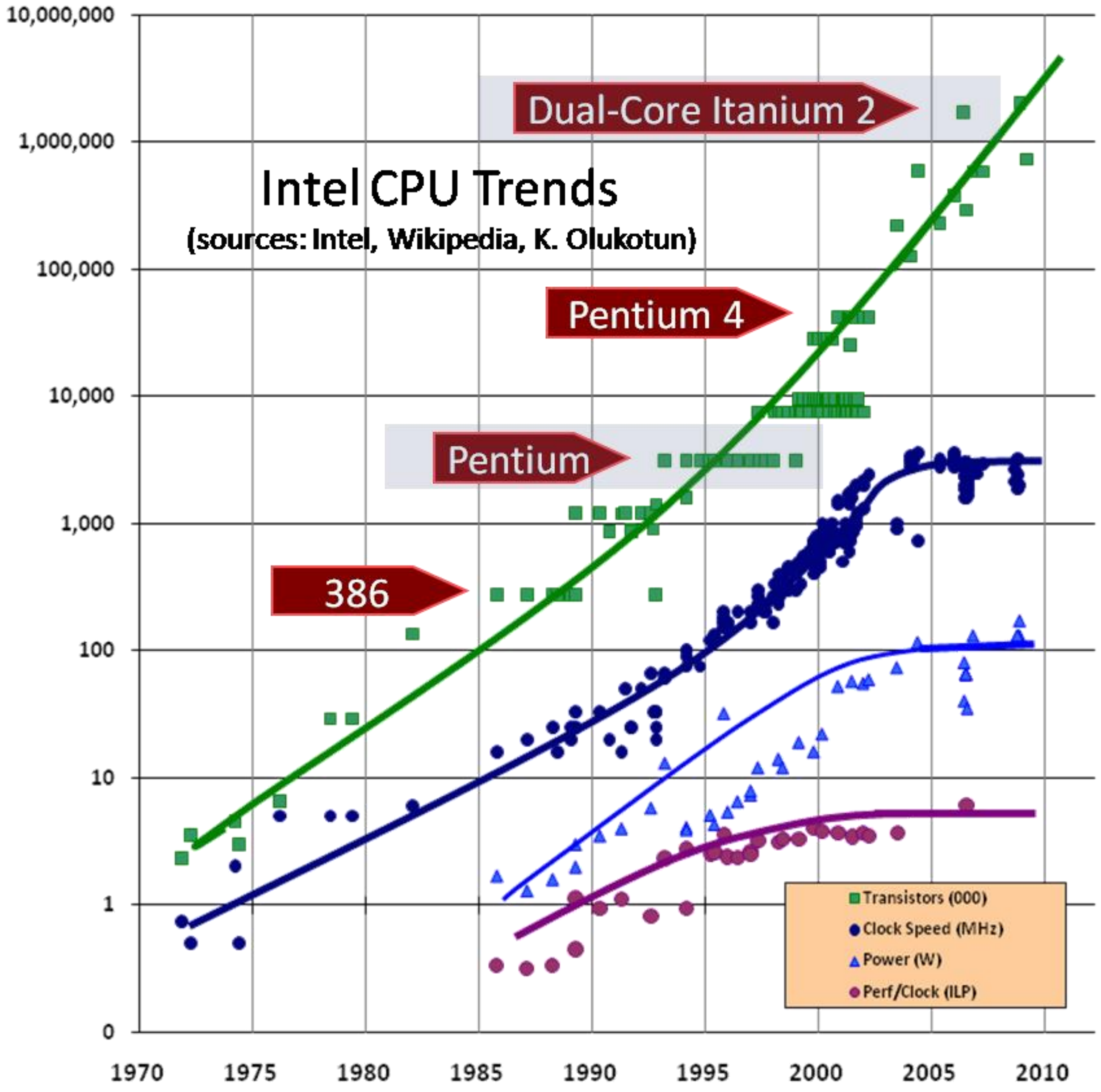
Multicores (SMs)

Co-processors (FPGA)

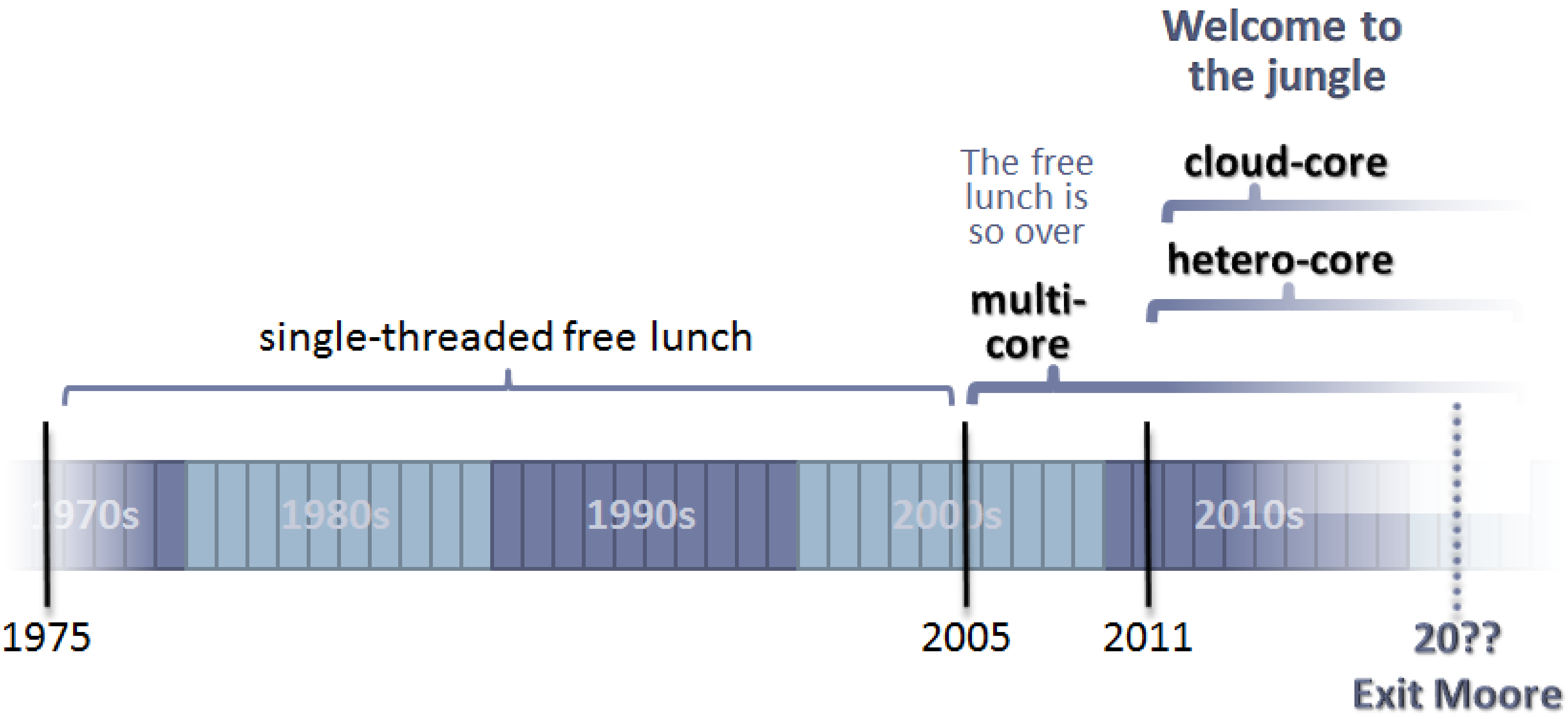
Free lunch is over

- CPU clock speed is bounded at 5Ghz
  1) Power wall
  2) Instruction-level parallelism
  3) Memory wall

- Performance gain with faster clock speed is over
- Performance improvement can be achieved by parallelism

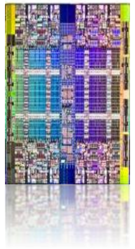
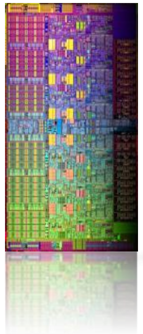


http://www.gotw.ca/publications/concurrency-ddj.htm



https://herbsutter.com/welcome-to-the-jungle/

Xeon: SIMD processors

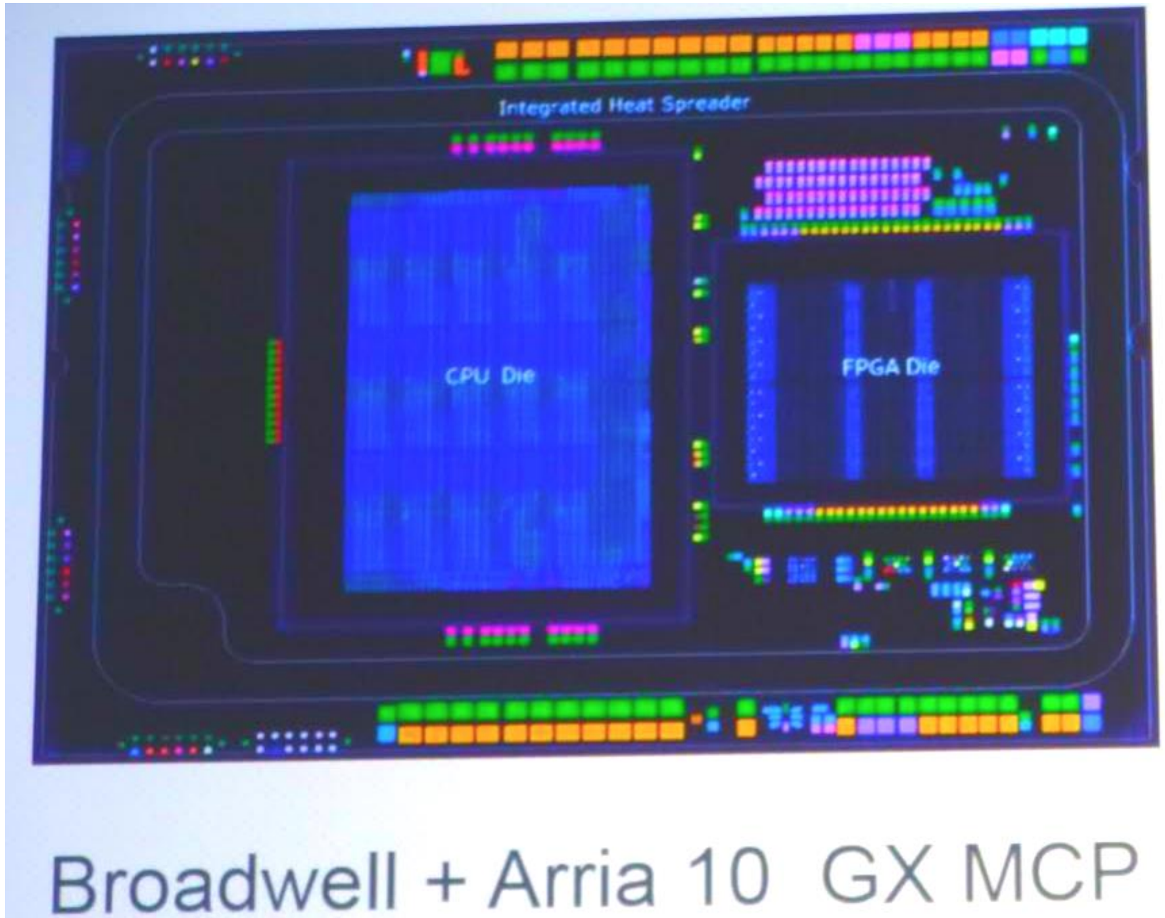



Images do not reflect actual die sizes

	Intel® Xeon® processor 64-bit	Intel® Xeon® processor 5100 series	Intel® Xeon® processor 5500 series	Intel® Xeon® processor 5600 series	Intel® Xeon® processor code-named Sandy Bridge	Intel® Xeon® processor code-named Ivy Bridge	Intel® Xeon® processor code-named Haswell	Intel® Xeon Phi™ coprocessor code-named Knights Corner
Core(s)	1	2	4	6	8			57-61
Threads	2	2	8	12	16			228-244
SIMD Width	128	128	128	128	256	256	256	512
	SSE2	**SSSE3**	**SSE4.2**	**SSE4.2**	**AVX**	**AVX**	**AVX2** **FMA3**	**IMCI**

Software challenge: Develop scalable software

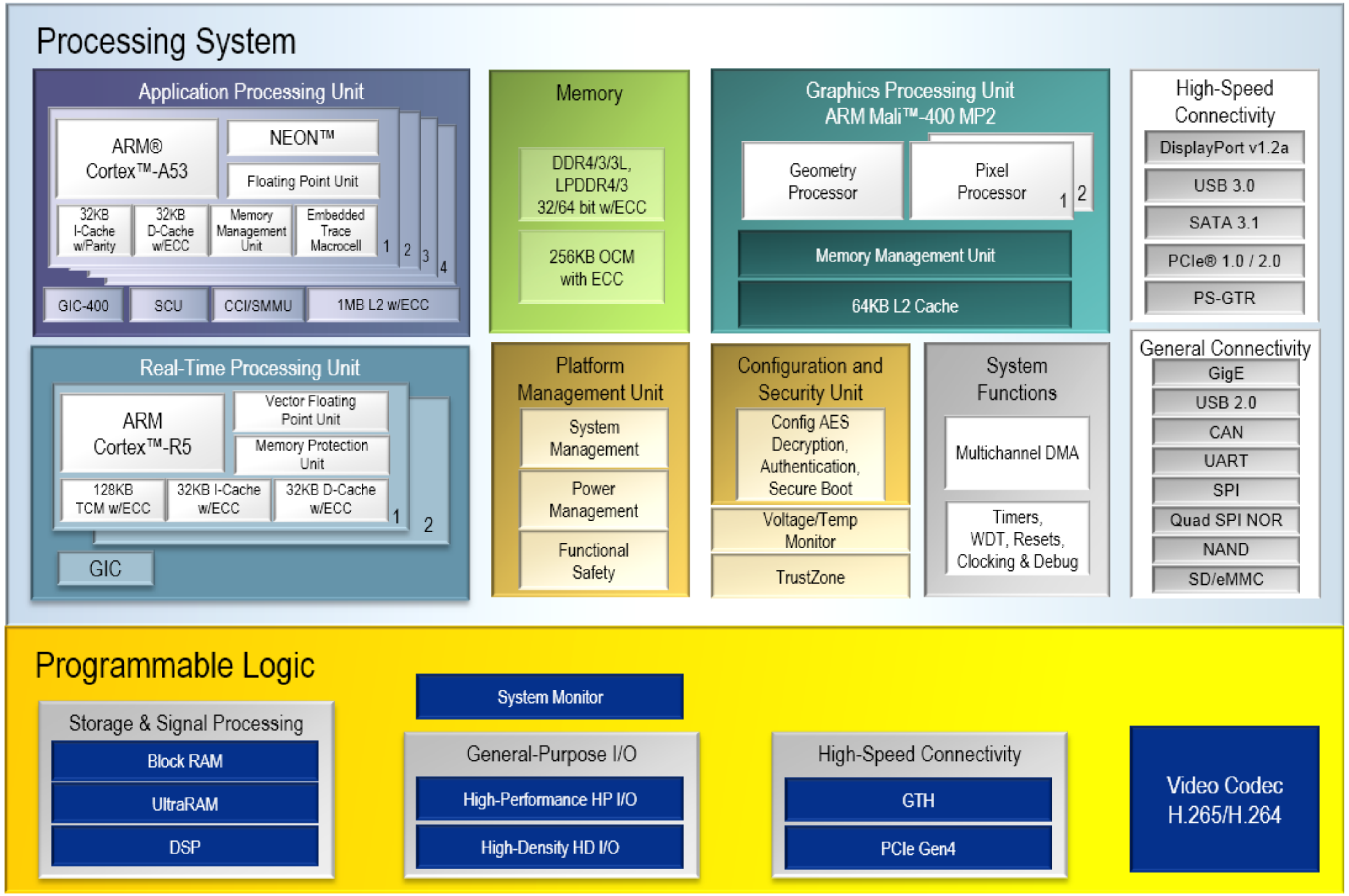
Intel: Heterogeneous processor



Broadwell + Arria 10 GX MCP

https://www.nextplatform.com/2016/03/14/intel-marrying-fpga-beefy-broadwell-open-compute-future/

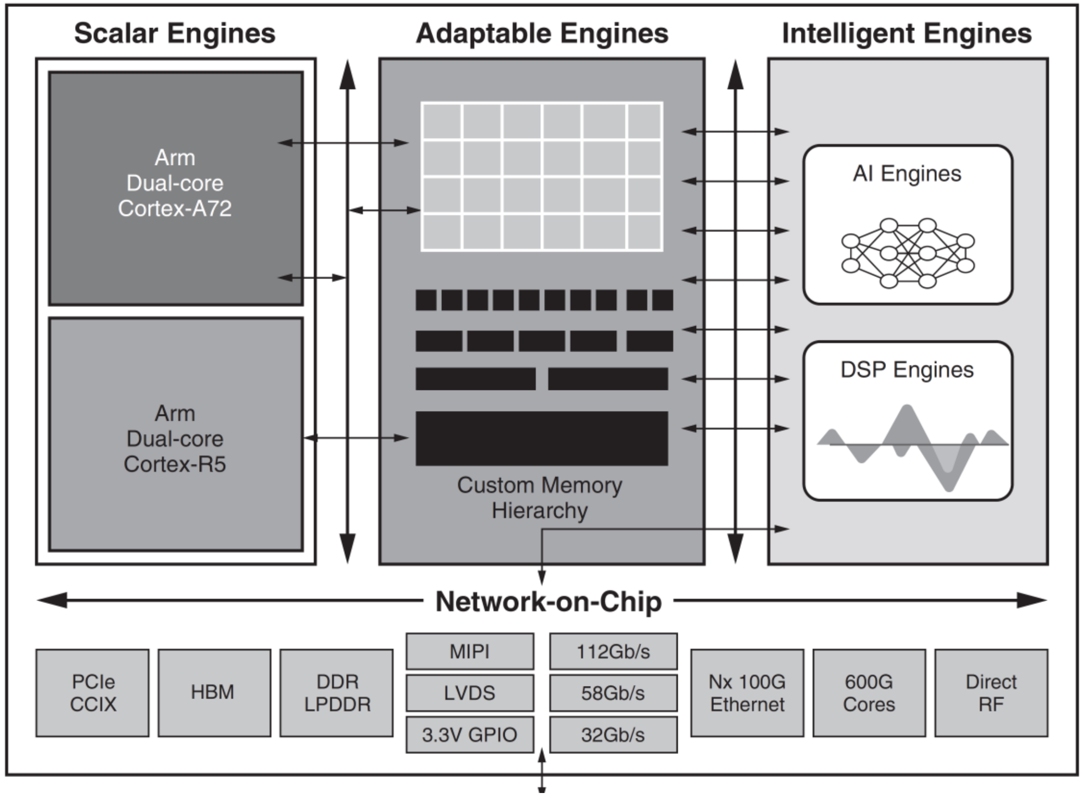
Xilinx: Multicore processors + FPGA

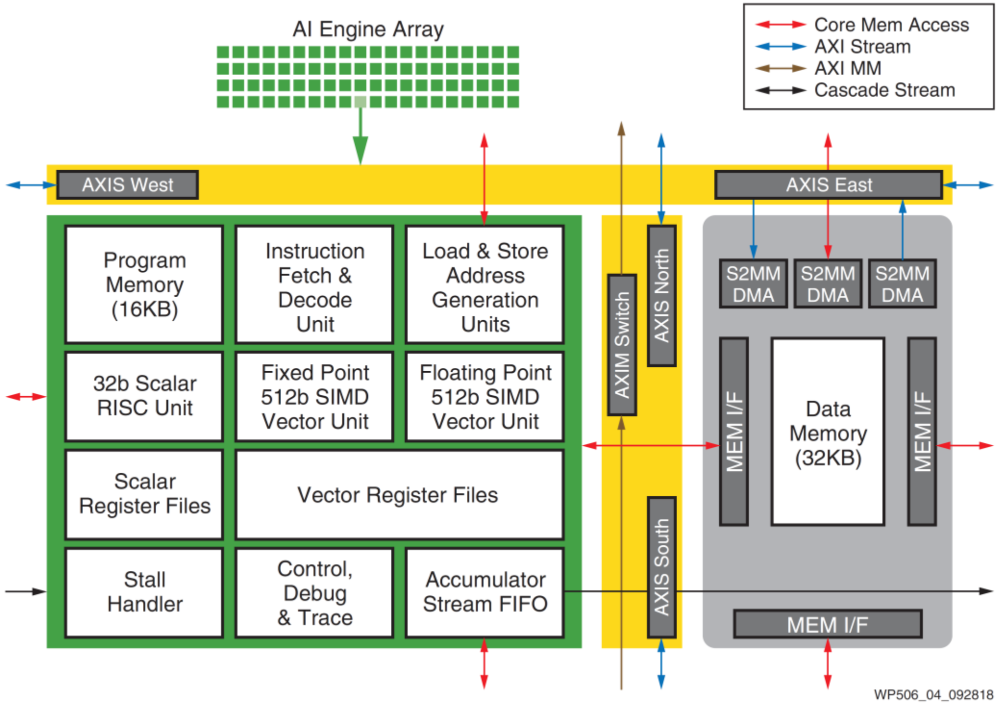


https://www.xilinx.com/content/dam/xilinx/imgs/products/zynq/zynq-ev-block.PNG

Xilinx: Adaptive Compute Acceleration Platform(AI FPGA)

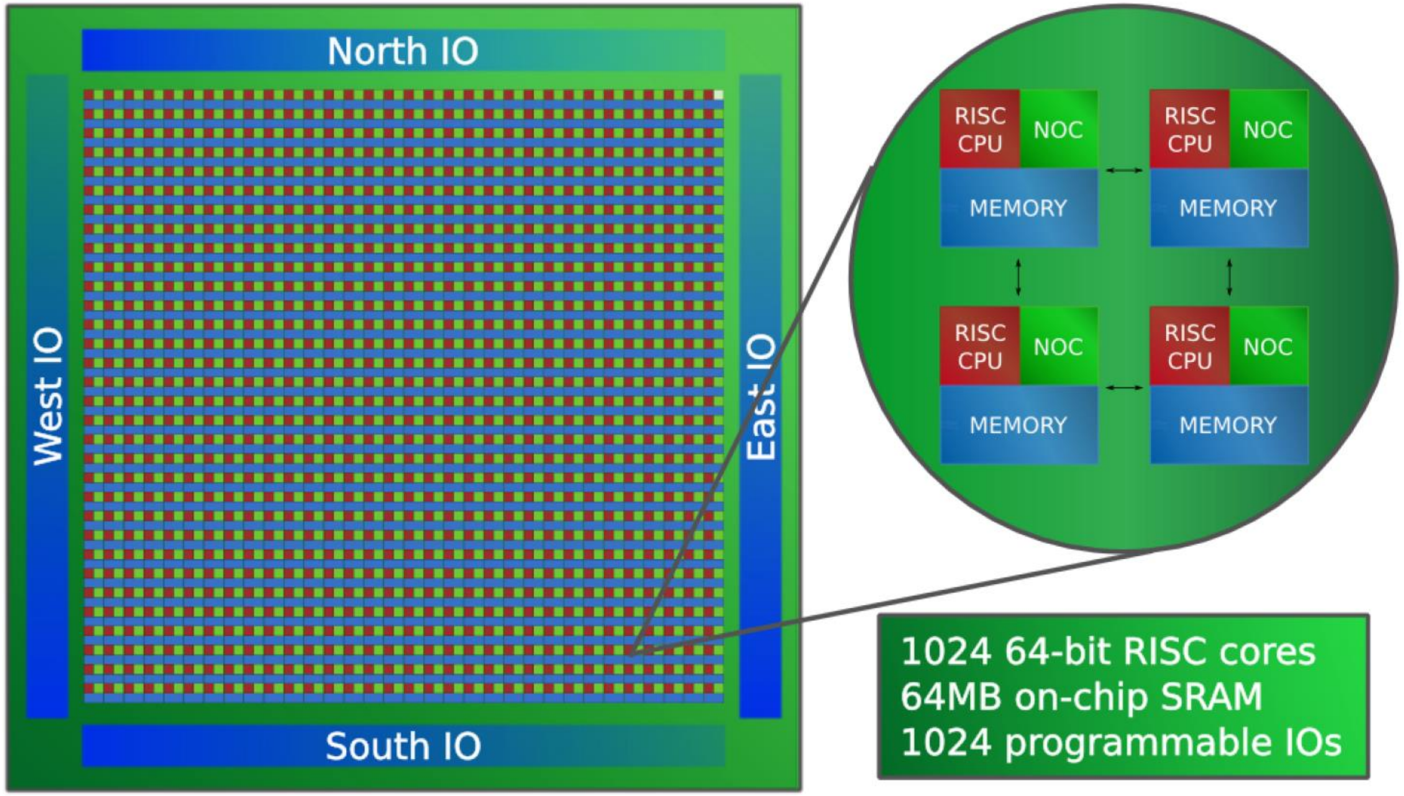
- Programable GPU (SIMD)
- Virtex FPGA (high performance) -> Versal FPGA (AI processor)





Epiphany-V: A 1024 processor 64-bit RISC System-On-Chip

1024 64-bit RISC processors



1024 64-bit RISC cores
64MB on-chip SRAM
1024 programmable IOs

Andreas Olofsson, Epiphany-V: A 1024 processor 64-bit RISC System-On-Chip, CoPR, 2016

Power Systems

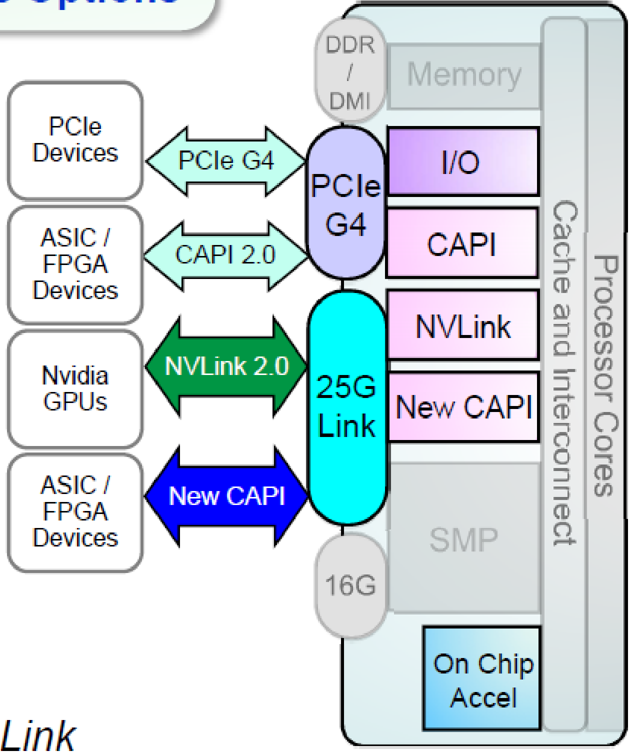
POWER9 – Premier Acceleration Platform

IBM

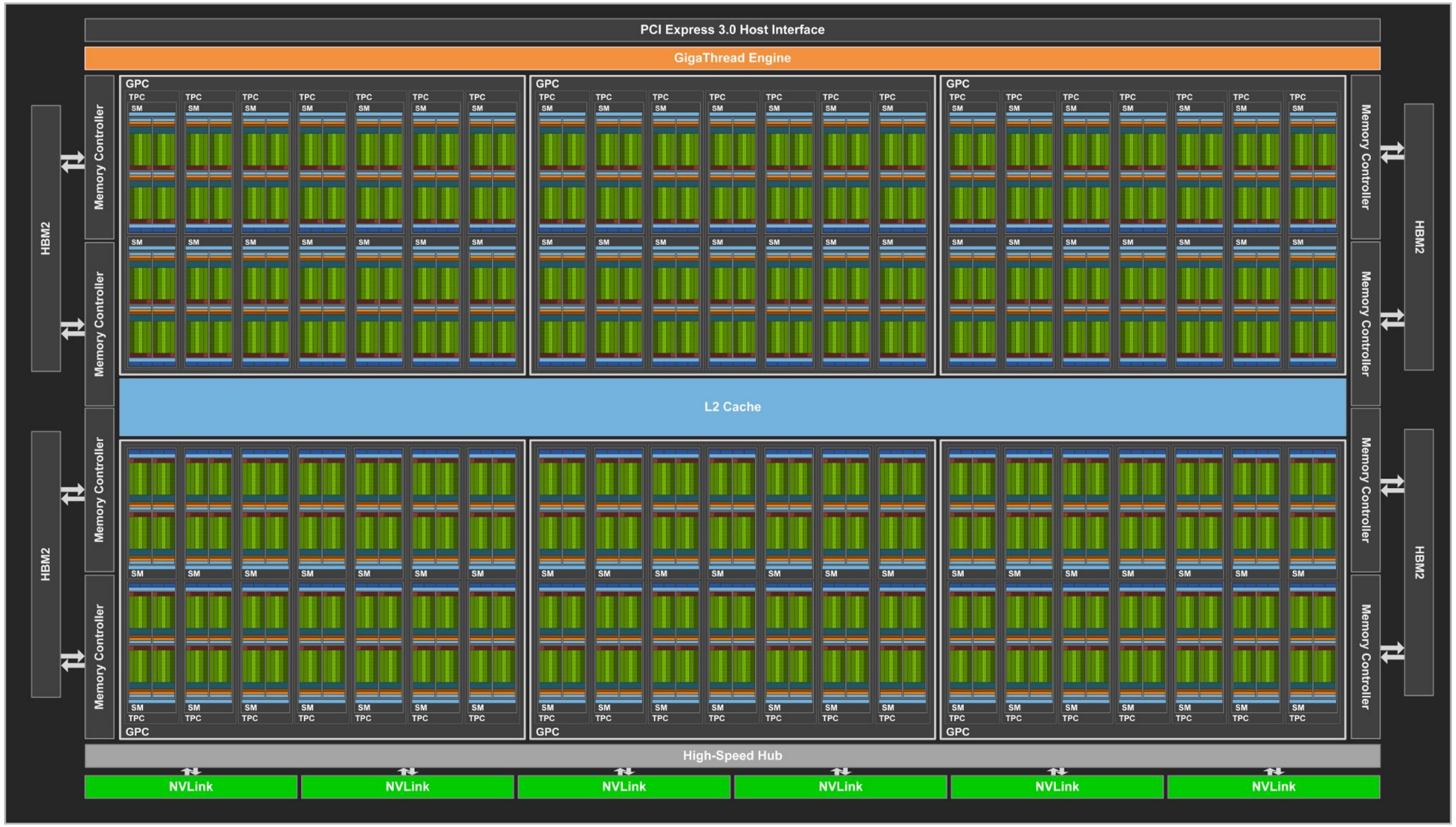
- Extreme Processor / Accelerator Bandwidth and Reduced Latency
- Coherent Memory and Virtual Addressing Capability for all Accelerators
- OpenPOWER Community Enablement – Robust Accelerated Compute Options

- **State of the Art I/O and Acceleration Attachment Signaling**
  - **PCIe Gen 4** x 48 lanes – 192 GB/s duplex bandwidth
  - **25G Link** x 48 lanes – 300 GB/s duplex bandwidth
- **Robust Accelerated Compute Options with OPEN standards**
  - **On-Chip Acceleration** – Gzip x1, 842 Compression x2, AES/SHA x2
  - **CAPI 2.0** – 4x bandwidth of POWER8 using *PCIe Gen 4*
  - **NVLink 2.0** – Next generation of GPU/CPU bandwidth and integration
  - **New CAPI** – High bandwidth, low latency and open interface using *25G Link*

POWER9 PowerAccel

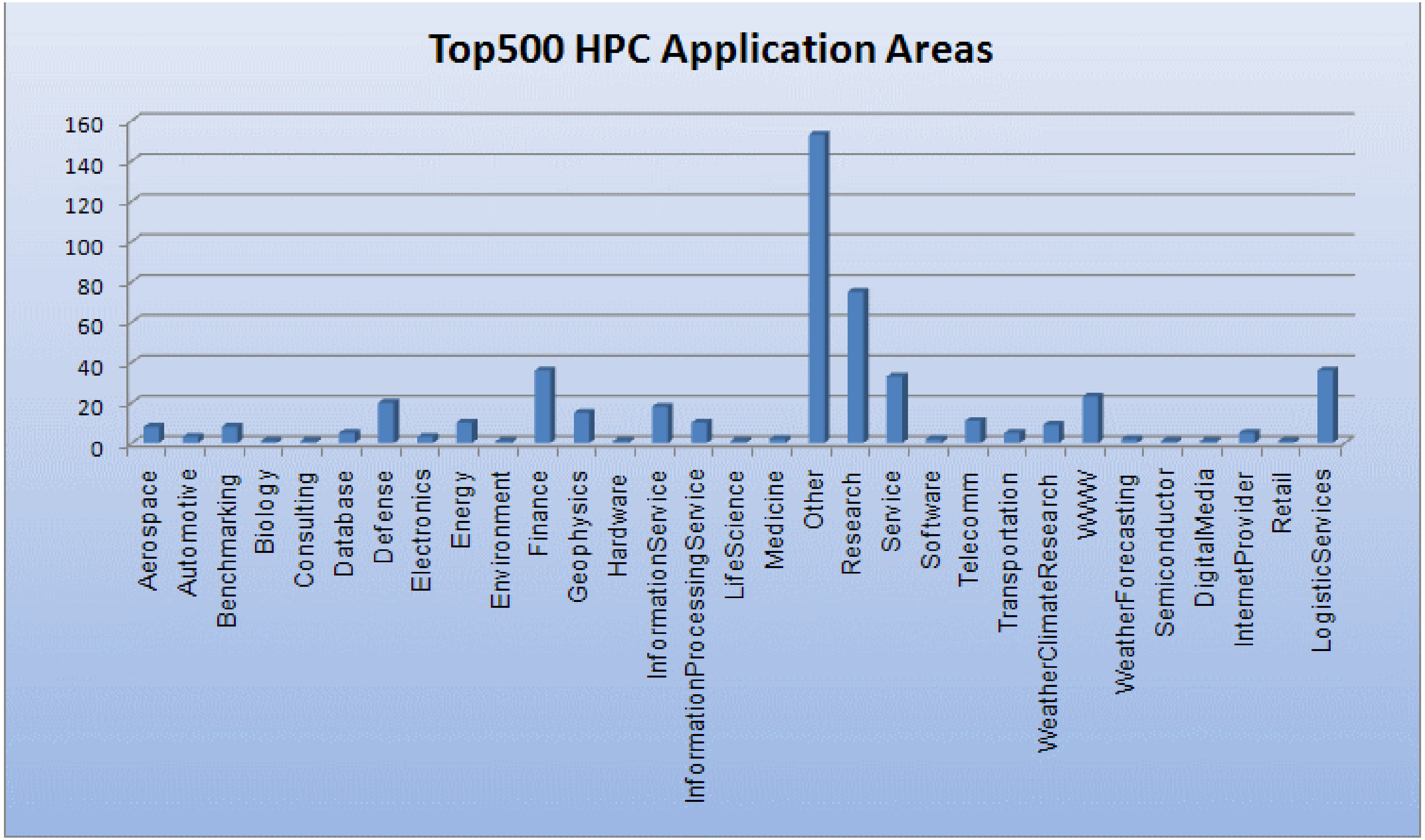


Nvidia GPU



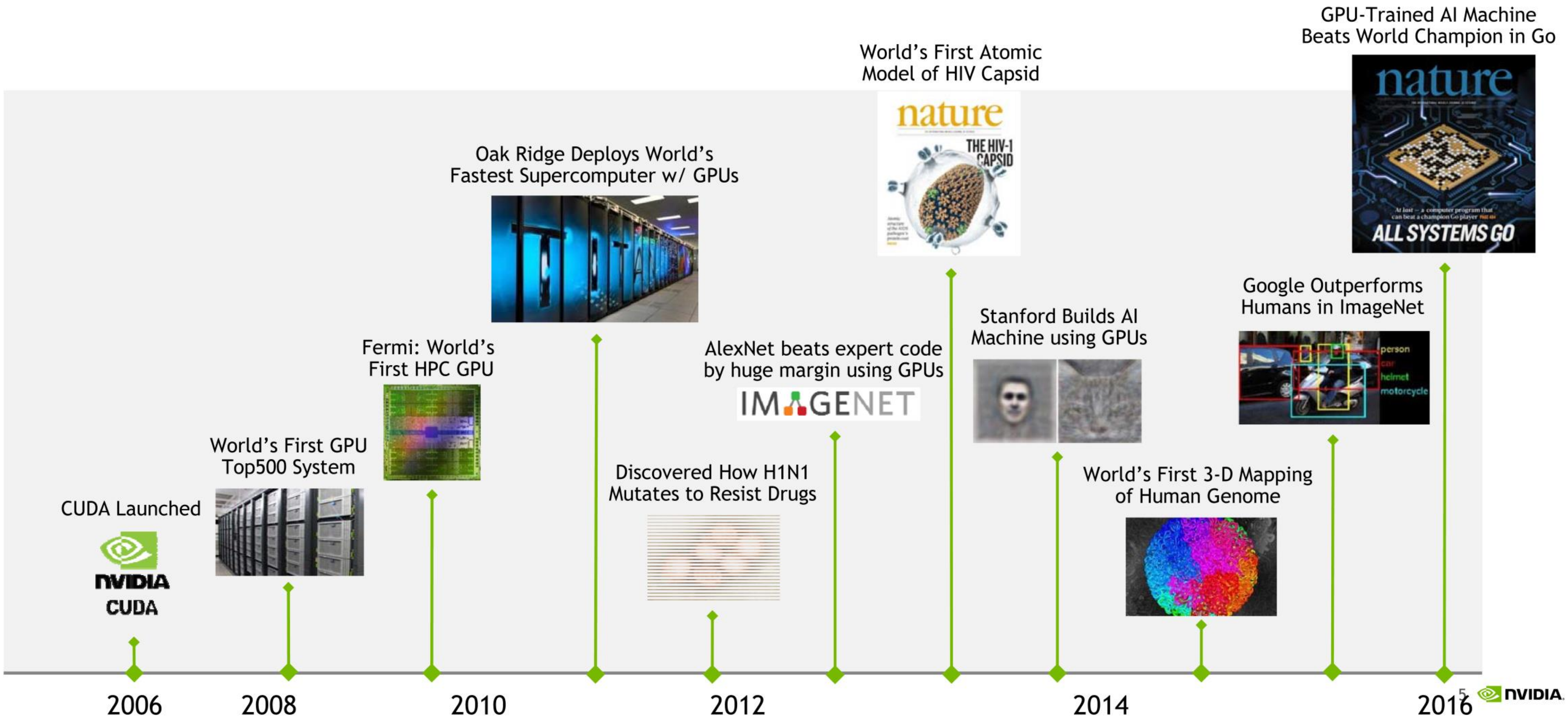
84 SM cores

High Performance Computing Applications



https://www.top500.org/

TEN YEARS OF GPU COMPUTING



CUDA Launched

World's First GPU Top500 System

Fermi: World's First HPC GPU

Oak Ridge Deploys World's Fastest Supercomputer w/ GPUs

Discovered How H1N1 Mutates to Resist Drugs

AlexNet beats expert code by huge margin using GPUs

World's First Atomic Model of HIV Capsid

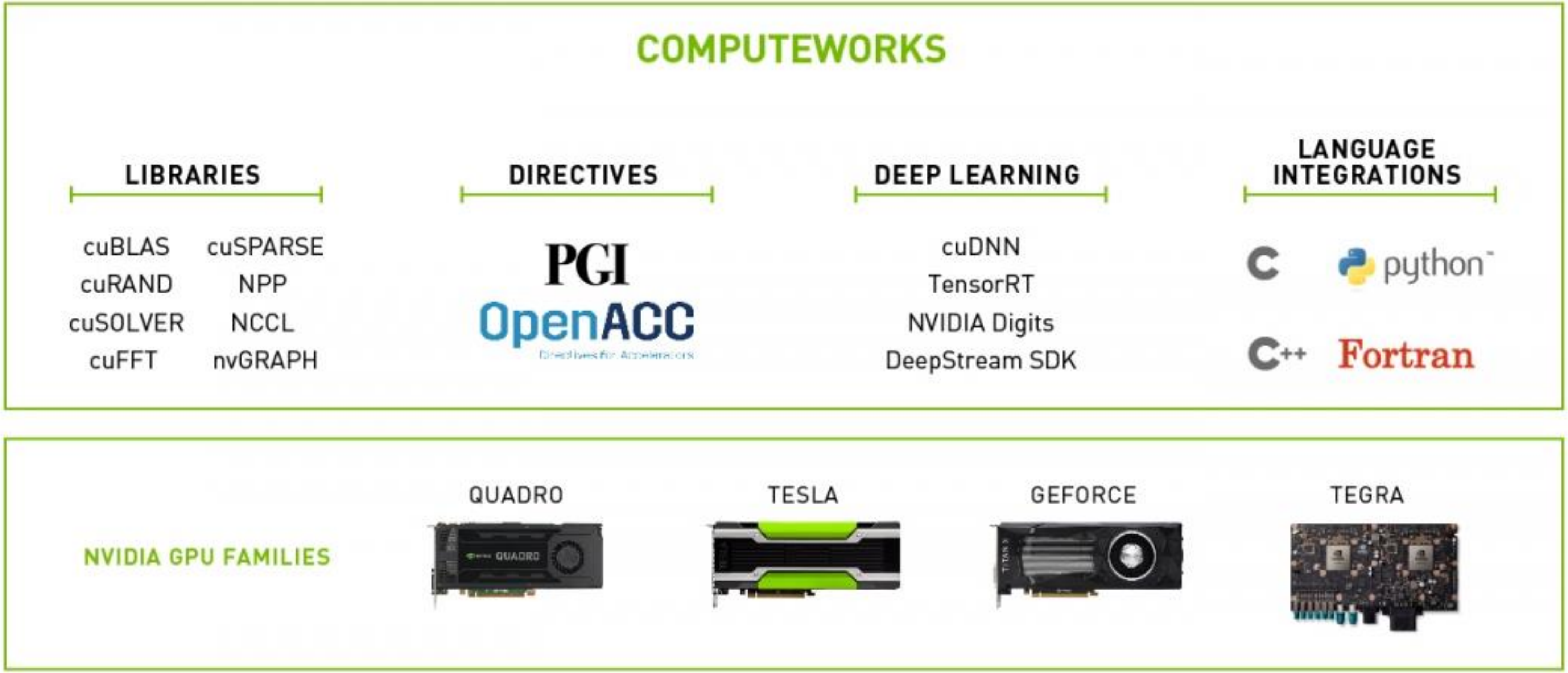
Stanford Builds AI Machine using GPUs

World's First 3-D Mapping of Human Genome

Google Outperforms Humans in ImageNet

GPU-Trained AI Machine Beats World Champion in Go

H. Luo, NVIDIA DEEP LEARNING PLATFORM Oct 2016



COMPUTEWORKS

LIBRARIES

cuBLAS cuSPARSE
cuRAND NPP
cuSOLVER NCCL
cuFFT nvGRAPH

DIRECTIVES

PGI
OpenACC
Directives for Accelerators

DEEP LEARNING

cuDNN
TensorRT
NVIDIA Digits
DeepStream SDK

LANGUAGE INTEGRATIONS

C python
C++ Fortran

NVIDIA GPU FAMILIES

QUADRO TESLA GEFORCE TEGRA

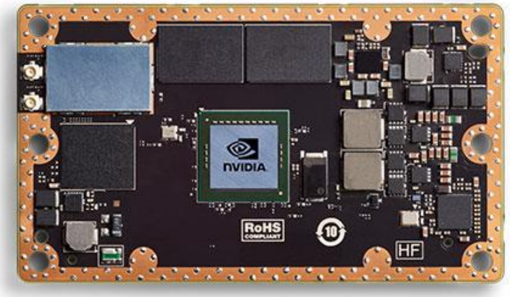
https://developer.nvidia.com/

Nvidia GPU types

Tesla : Computational GPU

GeForce : Gaming GPU

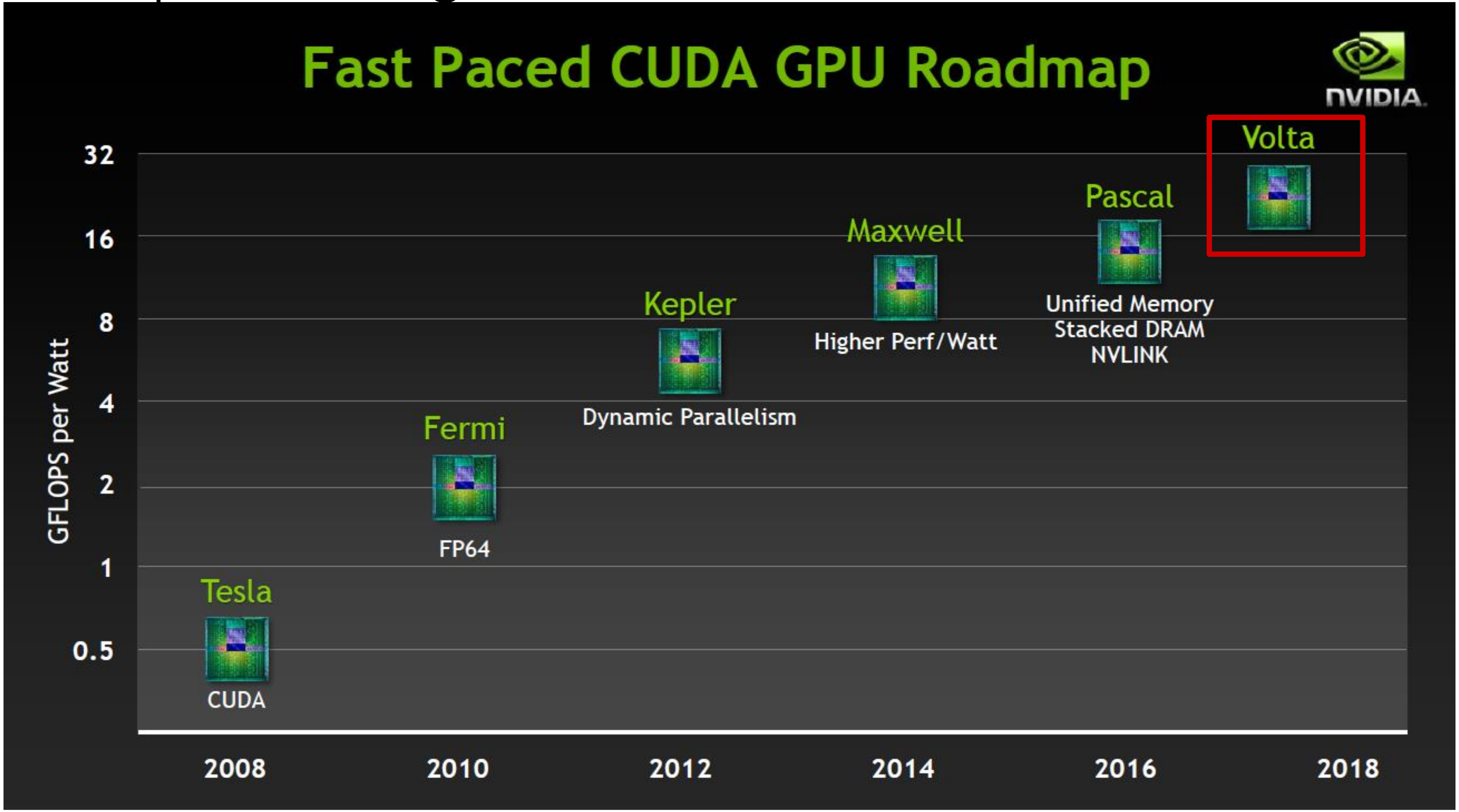
Embedded GPU



Jetson

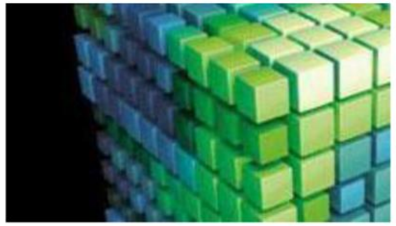
Xaiver

GPU processor generations



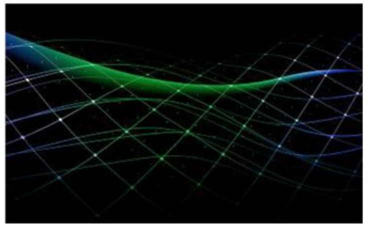
CUDA Libraries

cuBLAS
cuBLAS-XT
NVBLAS



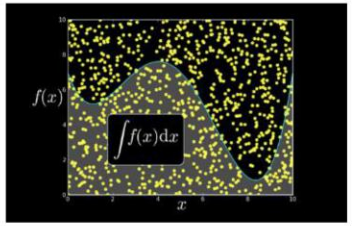
cuFFT
cuFFT-XT

cuSPARSE
cuSOLVER
AMGX



cuDNN

cuRAND

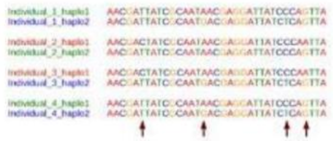


THRUST

NPP

NVENC

NVBIO



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

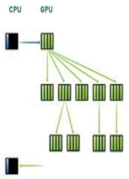
Open Compiler Tool Chain

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

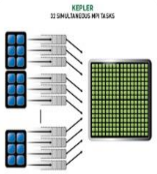
Hardware Capabilities

SMX

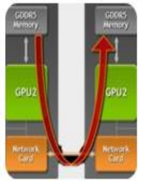
Dynamic Parallelism



HyperQ



GPUDirect



What is CUDA?

- CUDA Architecture
  Expose GPU parallelism for general-purpose computing
  Retain performance

- CUDA C/C++
  Based on industry-standard C/C++
  Small set of extensions to enable heterogeneous programming
  (CPU code + GPU code)
  Straightforward APIs to manage devices, memory and graphics etc.

- This course introduces CUDA C/C++

Mark Harris, **Introduction to CUDA C,** NVIDIA Corporation

Efficient CUDA programming

- Picking or developing good algorithms
  Domain Knowledge
  Parallelize the algorithms for the specific applications
- Basic principles
  Block partitions, memory access pattern, cache-aware coding
- Architecture specific optimization
  shuffle operation, cache control
- Instruction-level optimization
  (Pseudo) Assembly (ex. ptx)

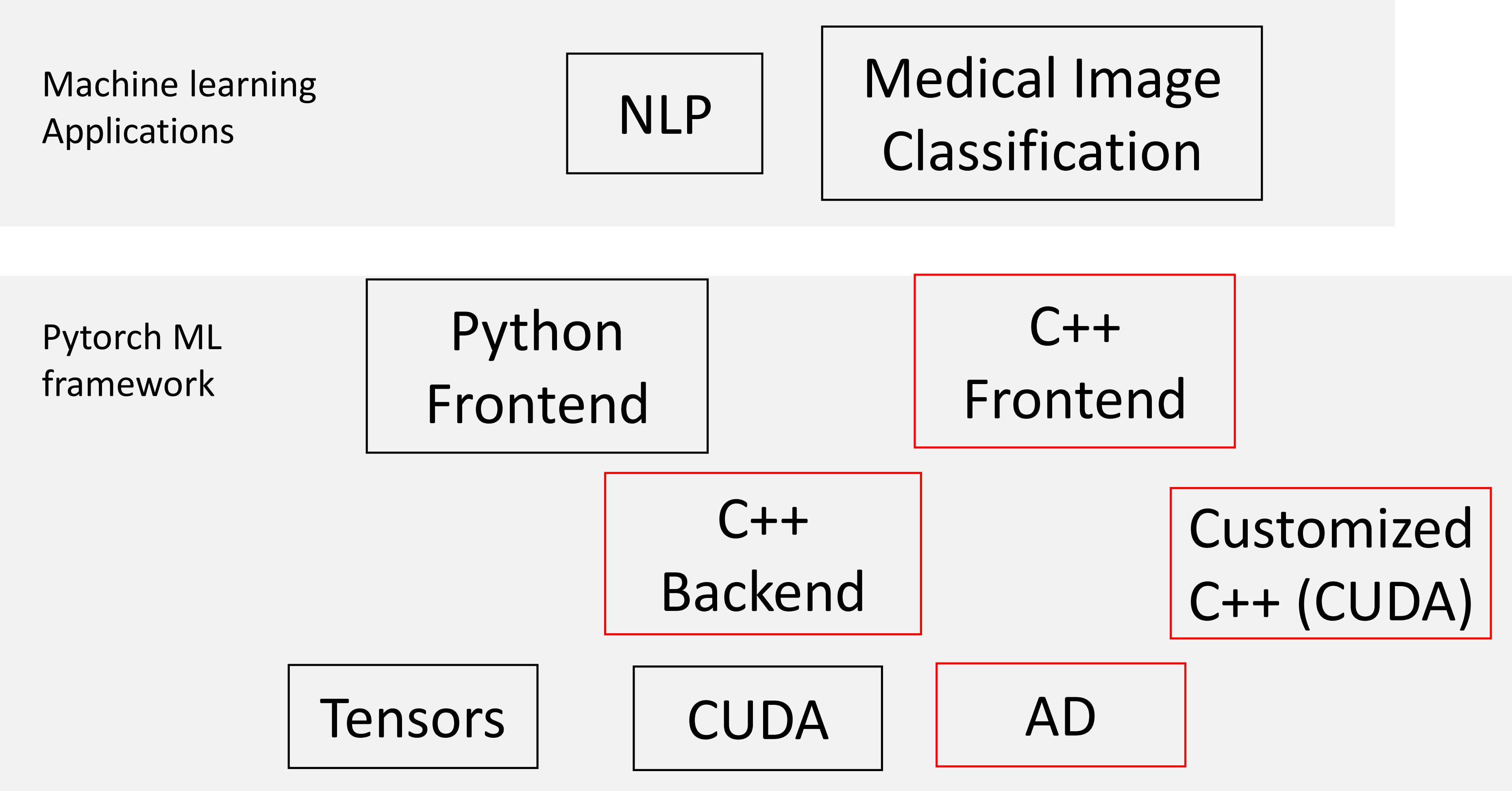
Impart to
performance

↓

What makes parallel programming hard?

- "Serial illusion" or "Serial traps"
  - Every HW is naturally parallel, but has been programmed serially
- Task execution in parallel programs is not deterministic
- Automatic parallelization from a serial code is very limited
  - The complier weakness to parallel computing (only AVX instructions, ILP)
  - OS can schedule multiple threads to different programs
- Still, Human can program
- But, There are some programming languages to help parallel programming at the high-level
  - OpenACC, OpenMP
  - (eg. #pragma omp parallel private(nthreads, tid))

Machine learning framework with CUDA



Machine learning Applications

NLP

Medical Image Classification

Pytorch ML framework

Python Frontend

C++ Frontend

C++ Backend

Customized C++ (CUDA)

Tensors

CUDA

AD

AD: Automatic differentiation

Think Parallel

Shift in thinking from serialized algorithms to parallel algorithms

References

- Ian Foster, Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering, 1995, Addison-Wesley Longman Publishing Co., Inc