Applied Parallel Programming on GPU

Alptekin Temizel

atemizel@metu.edu.tr

Outline of the Course

1.Introduction to GPU Programming

2. CUDA Introduction

3. Hands-on Lab

4. Effective Use of Memory and Optimization

5.No Lecture (Republic Day)

6. Effective Use of Memory and Optimization – Lab

7. Project Proposals

8. Control Flow, parallel reduction, optimization

9. CUDA Debugging, Profiler/Parallel Nsight

10. Analytical Modeling of Parallel Systems and Performance Analysis

11. Thrust Library

12. Case studies

13. Multi GPU programming and Advanced Concepts

14. Deployment Tools

Grading

- 5% Lab Assignment
- 30% Assignments - 3 assignments in total
- 50% Project:
  - Phase 1: You will present a project idea.
  - Phase 2: You will design and start implementing, you'll submit a design document.
  - Phase 3: You will finalize implementation, optimize and demonstrate your project. Then present a final report showing the achieved performance gain.
- 15% Quiz (3 quizzes)
- 5% Attendance

Total: 105%

Notes

- I will be uploading the lecture notes shortly after the lecture to ODTUClass.
- You will be told about the quiz time and the content one week before the quiz **at the lecture hours**!
- You will submit your work (assignments and project phases) to ODTUClass.
  - The system closes at 23:55 at the due date --- do not leave it to the last minute.
- Send me an e-mail atemizel@metu.edu.tr to get an appointment if you need help with anything related with the course.

Background Requirement

- Intermediate knowledge of C
  - You should be able to use pointers!
- Intermediate knowledge of parallel computing
  - You should know what a thread is
  - You should know about the synchronization
- Knowledge of algorithms
  - You should know the basic algorithms
  - You should be able to implement algorithms

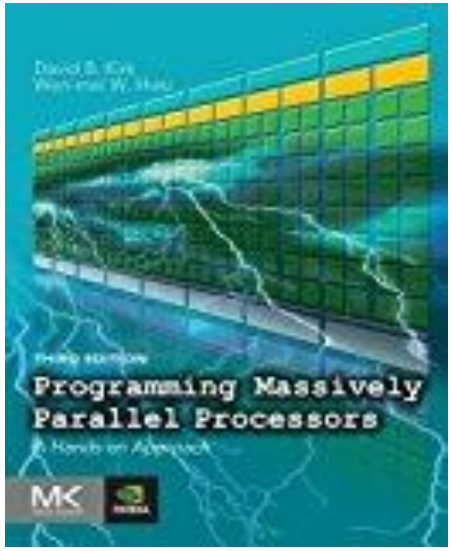
TextBook

- David B. Kirk and Wen-mei W. Hwu “Programming Massively Parallel Processors: A Hands-on Approach”

  Morgan Kaufman, Elsevier

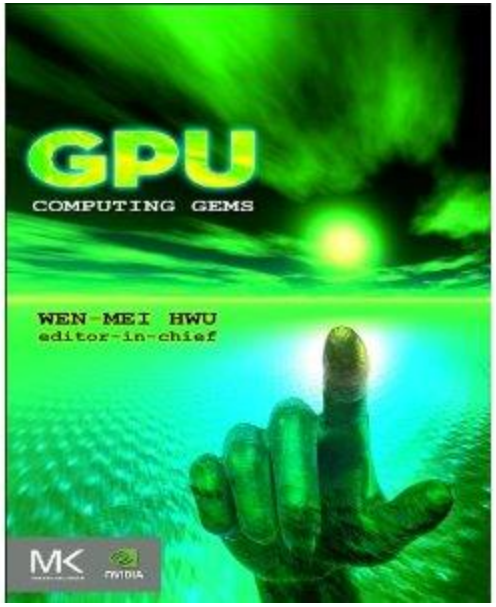
  Publication Date: December 21, 2016

  ISBN-10: 0128119861 | ISBN-13: 978-0128119860 | Edition: 3



Reference Material

- Wen-mei W. Hwu (Editor), “GPU Computing Gems”, ISBN-10: 0123849888, ISBN-13: 978-0123849885.

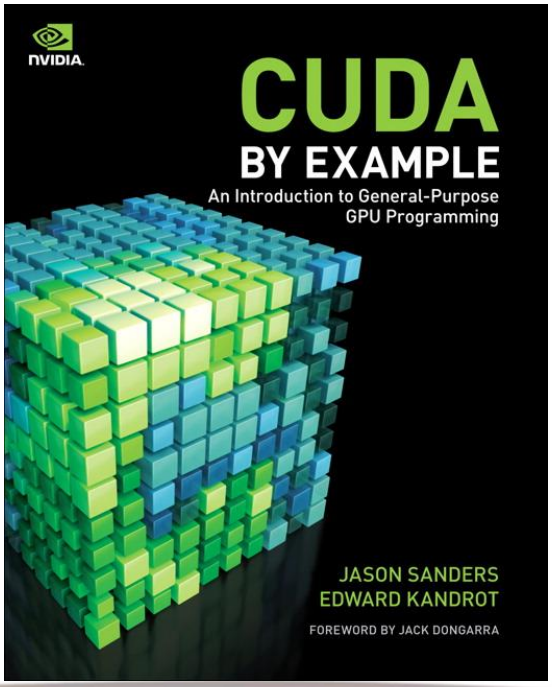


Chapters

1. Scientific Simulation
2. Life Sciences
3. Statistical Modeling
4. Emerging Data-Intensive Applications
5. Electronic Design Automation
6. Ray Tracing and Rendering
7. Computer Vision
8. Video and Image Processing
9. Signal and Audio Processing
10. Medical Imaging

Reference Material

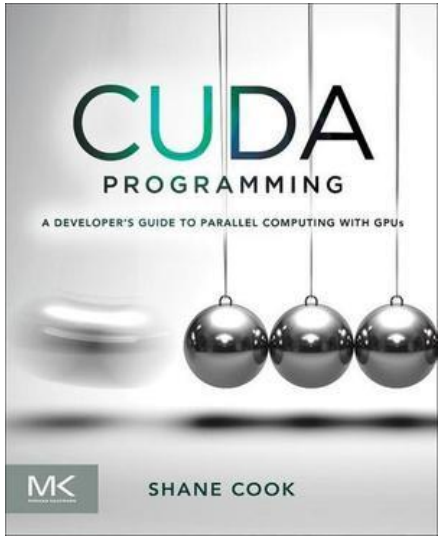
- “CUDA by Example: An Introduction to General-Purpose GPU Programming”, Addison Wesley, ISBN-10: 0131387685, ISBN-13: 978-0131387683.



Reference Materials

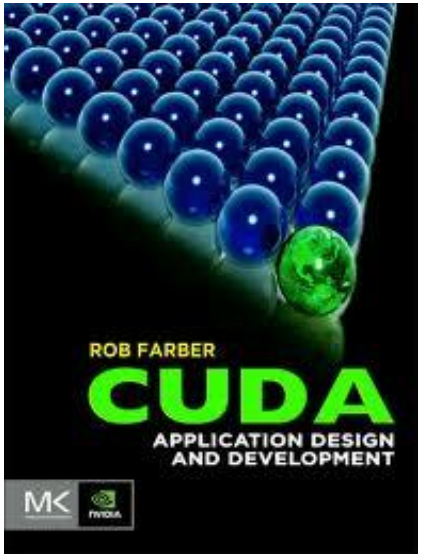
- Shane Cook, “CUDA Programming: A Developer's Guide to Parallel Computing with GPUs”

Publication Date: November 27, 2012 | ISBN-10: 0124159338 | ISBN-13: 978-0124159334



Reference Materials

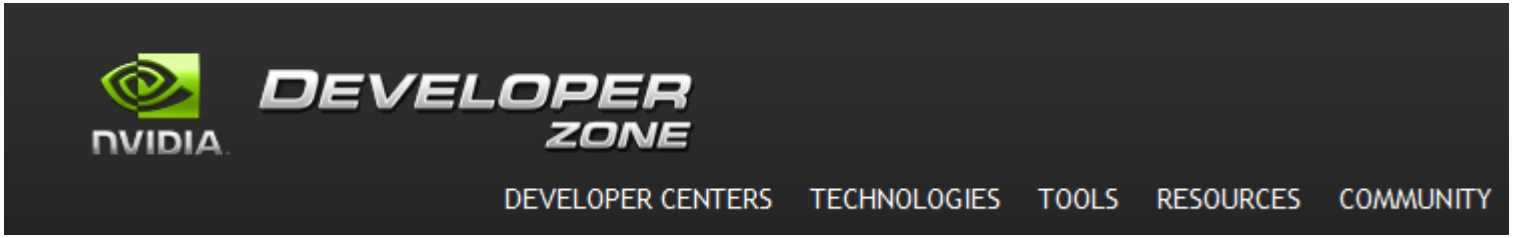
- Rob Farber, “CUDA Application Design and Development”, ISBN-10: 0123884268 | ISBN-13: 978-0123884268.



Useful Links

gpucomputing.net

Connect • Communicate • Collaborate



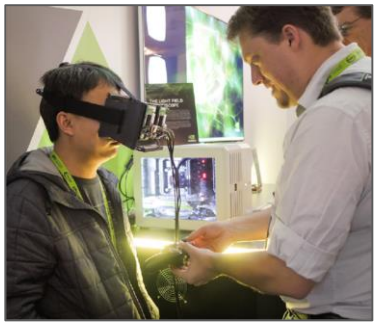
http://developer.nvidia.com/page/home.html

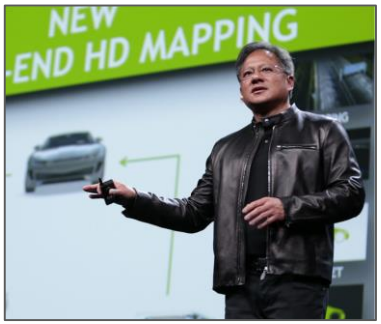
KHRONOS GROUP

CONNECTING SOFTWARE TO SILICON

OpenCL Developer Zone,
http://www.khronos.org/opencl/

GPU TECHNOLOGY CONFERENCE





Silicon Valley, March 26-29
Munich, October 10-11

Beijing,
Israel,
Washington DC
Tokyo

Facilities (in VRCV Lab and student lab)

2 x Quadro 2000

1x TESLA K20

1x TESLA K40

1x TITANX

2x RTX 2070

3x Jetson TK1 dev kits

1x Jetson TX1 dev kit

GPU Related Activities

- Professor Partnership Award – July 2009
- GPU Teaching Center – December 2010
- GPU Research Center – February 2012
- Deep Learning Institute (DLI) Certification – August 2017

Graphics processing unit (GPU) (1)

Graphics Processing Unit is a single-chip processor designed to process mathematically intensive tasks.

Graphics processing unit (GPU) (2)

- GPU's are originally designed for graphics rendering and gaming.
- Found on PCs, game consoles, embedded systems, mobile phones, cars.
- Big players: AMD, Intel and NVIDIA

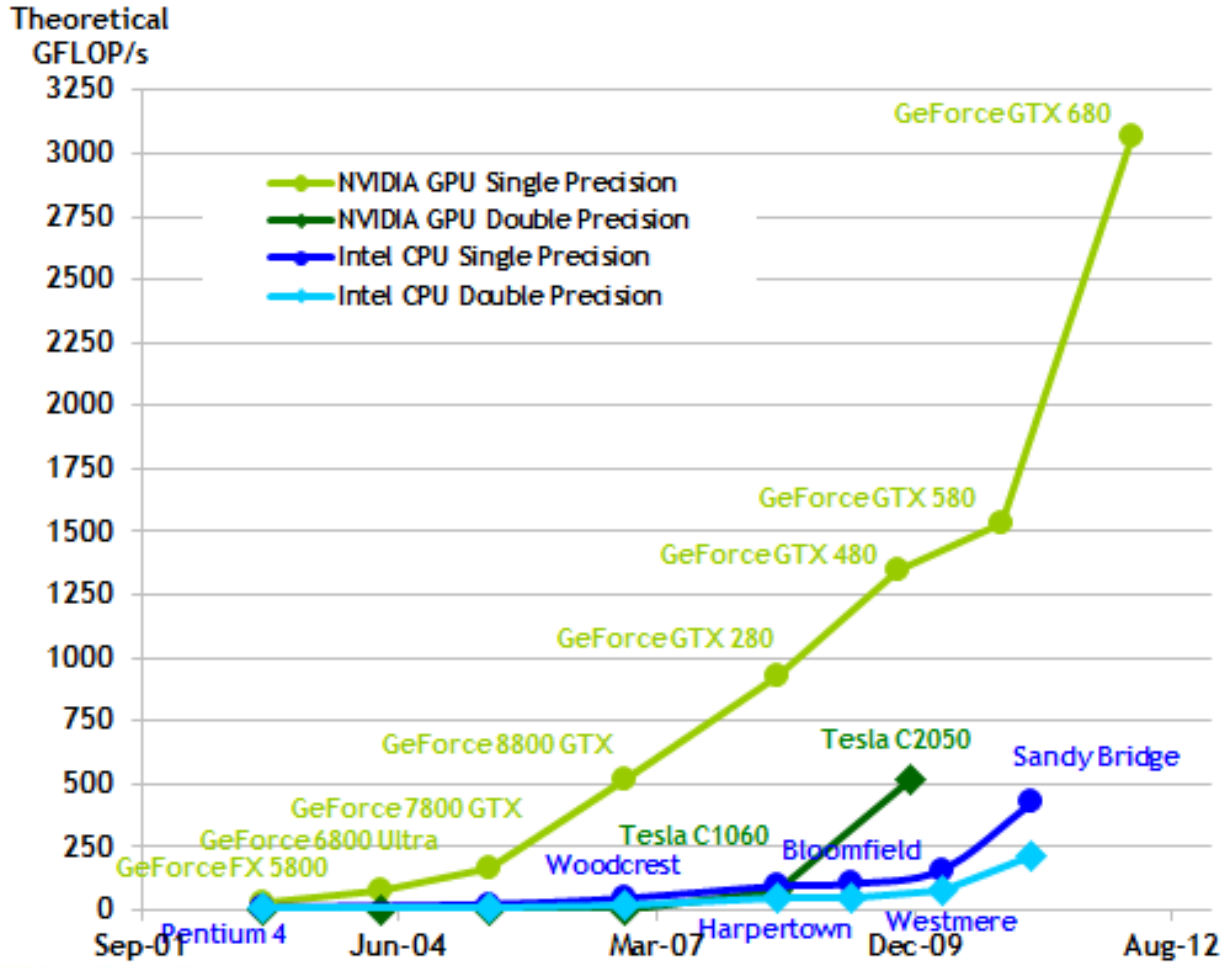
Parallel Computing with GPUs?

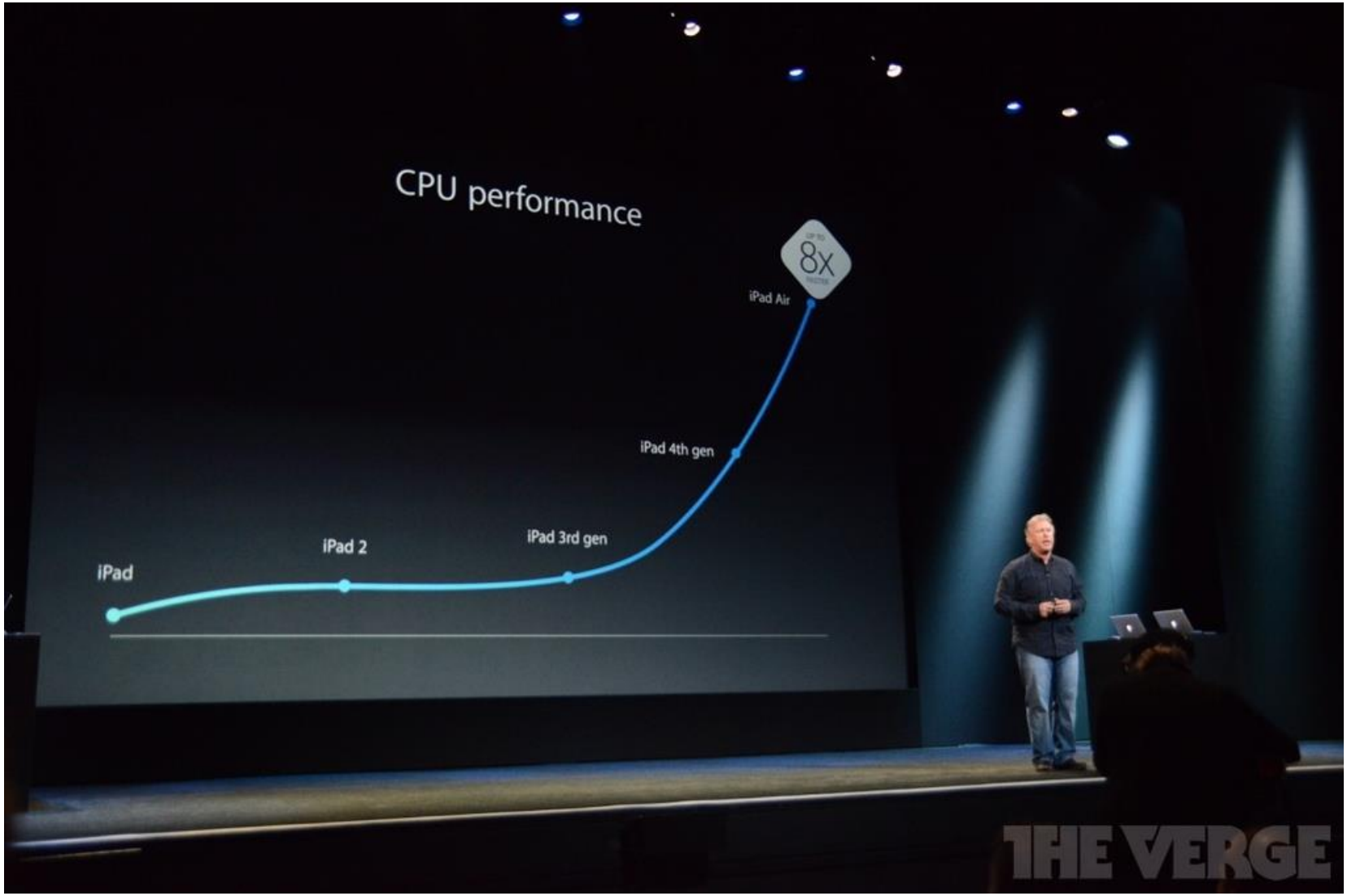
GPUs are fast…
GTX 285 has 240 cores, 1 TFLOPS

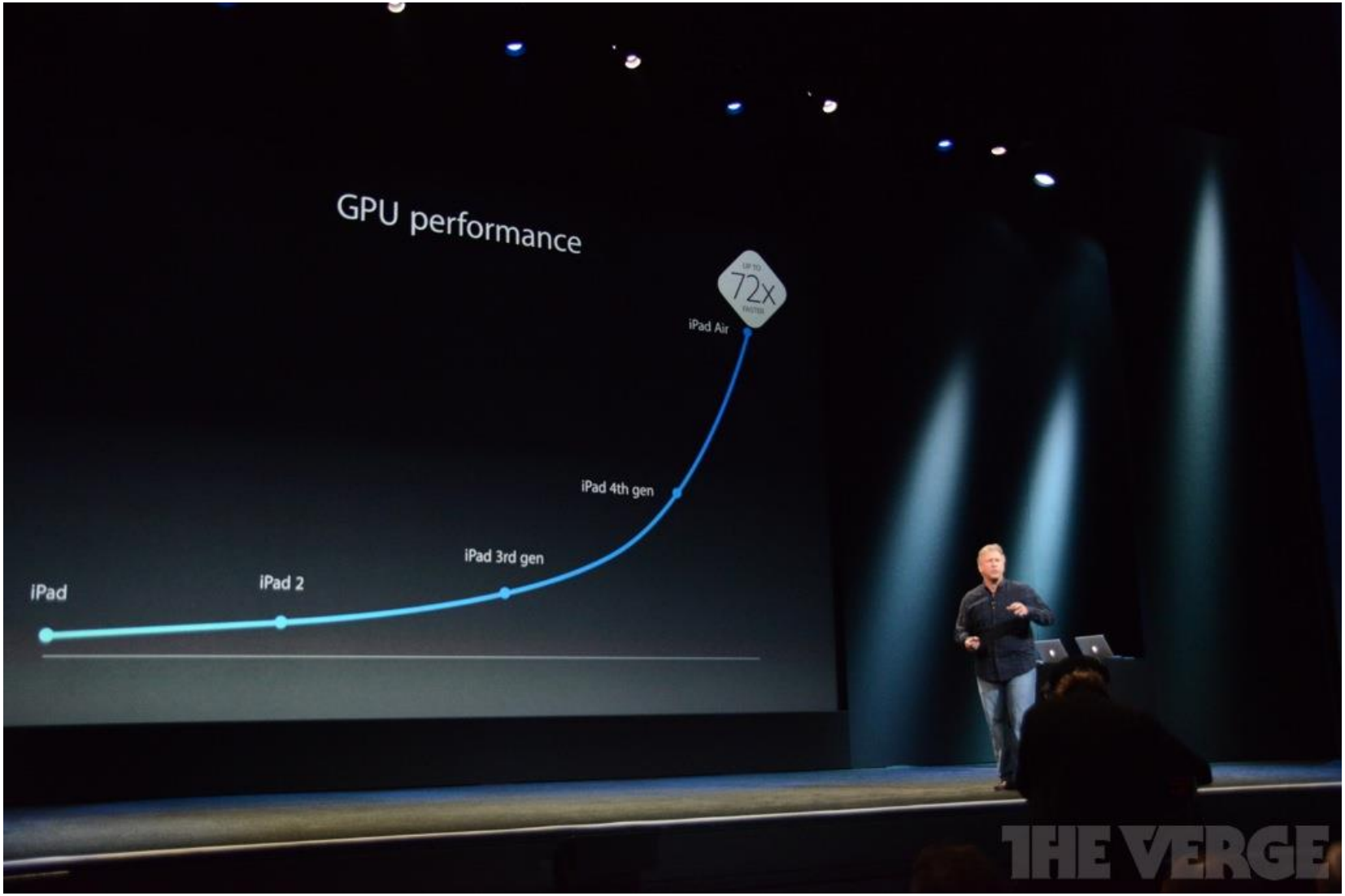
GTX 480: 1345 GFLOPS 250W, March 2010
GTX 590: 2488 GFLOPS 244W, March 2011
GTX 680: 3090 GFLOPS 195W, March 2012
GTX 780Ti: 5046 GFLOPS 250W, November 2013 (649$)
GTX 980: 4612 GFLOPS , 165W, September 2014 (549$) (Later: 5632 GLOPS, 250W)
GTX 980 notebook: 4612 GFLOPS, 145 W, September 2015
GTX 1080: 9 TFLOPS, 180W, May 2016 (599$)
GTX 1080TI: 11.3 TFLOPS, 250W March 2017 (699$)
GTX 2080TI: 13.4 TFLOPS, 250W, Sept 2018 (999$)

Note: Intel Core i7-8700K 6-core CPU has a performance of 218 GFLOPS @95W

GPU vs. CPU Performance




CPU performance
8X
iPad Air
iPad 4th gen
iPad 3rd gen
iPad 2
iPad
THE VERGE


GPU performance
UP TO
72X
iPad Air
iPad 4th gen
iPad 3rd gen
iPad 2
iPad
THE VERGE

Parallel Computing with GPUs?

- The power and flexibility of GPUs makes them an attractive platform for general-purpose computation
- Example applications range from in-game physics simulation to conventional computational science
- Goal: make the inexpensive power of the GPU available to developers as a sort of computational co-processor

"Massively Parallel Computing"

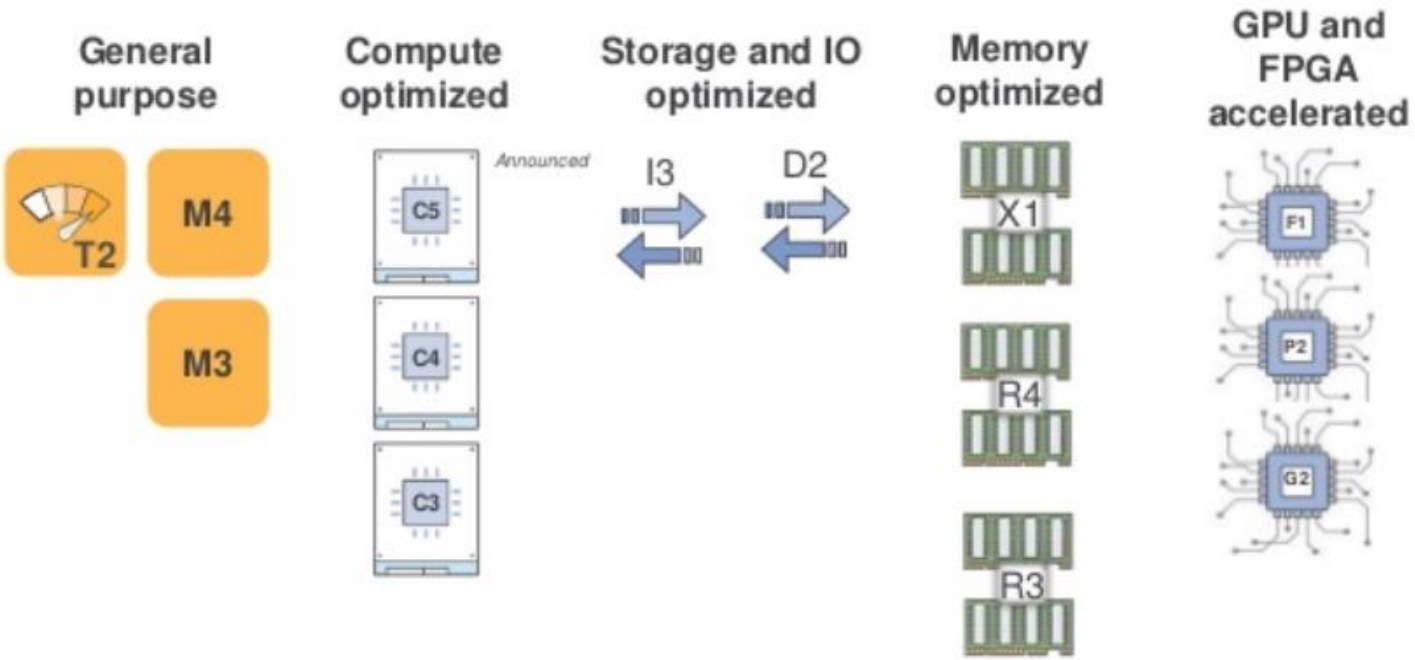
Courtesy David Luebke, University of Virginia

GPU Data Centers / Clouds

Supermicro 1U – 4GPU Server

Amazon GPU Instances

EC2 Compute Instance Types



P2 Instance Details

Name	GPUs	vCPUs	RAM (GiB)	Network Bandwidth	Price/Hour*	RI Price / Hour**
p2.xlarge	1	4	61	High	$0.900	$0.425
p2.8xlarge	8	32	488	10 Gbps	$7.200	$3.400
p2.16xlarge	16	64	732	20 Gbps	$14.400	$6.800

Tegra Based Products

Tegra 4: Nvidia Shield

Tegra 3 Processor To Power Audi's Next-Gen Infotainment And Digital Instrument Cluster

Tegra VCM: Digital Dashboard In New Tesla Motors Electric Sedan

Tegra K1 Based Products

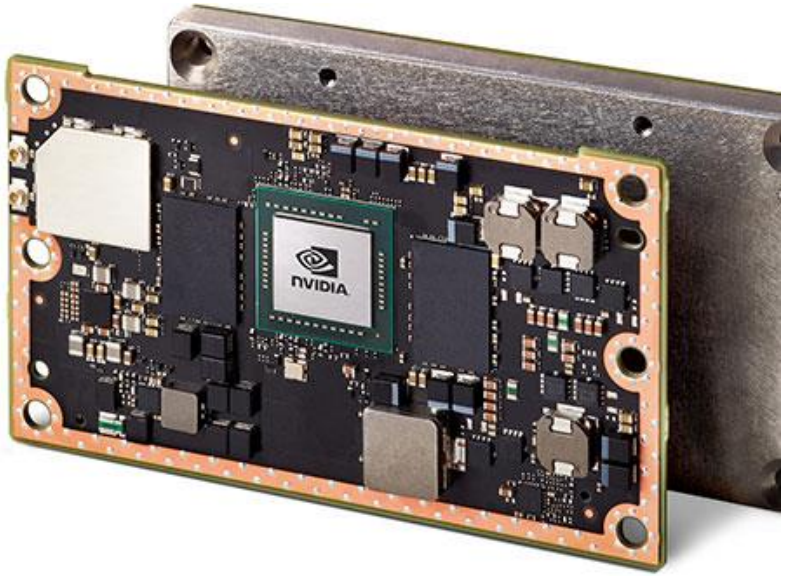
Tegra K1: Nvidia Shield Tablet

Tegra K1: Acer Chromebook

Embedded Devices: NVIDIA Jetson TK1

- Built around Tegra K1 System-on-a-Chip (SoC)
  - NVIDIA 4-Plus-1 quad-core ARM Cortex A-15 CPU
  - NVIDIA Kepler GPU with 192 CUDA cores (~365 GLOPS)
- Runs Linux for Tegra

Embedded Devices: NVIDIA Jetson TX1



	Jetson TX2	Jetson TX1
GPU	NVIDIA Pascal™, 256 CUDA cores	NVIDIA Maxwell ™, 256 CUDA cores
CPU	HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2	Quad ARM® A57/2 MB L2
Video	4K x 2K 60 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (12-Bit Support)	4K x 2K 30 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (10-Bit Support)
Memory	8 GB 128 bit LPDDR4 59.7 GB/s	4 GB 64 bit LPDDR4 25.6 GB/s
Display	2x DSI, 2x DP 1.2 / HDMI 2.0 / eDP 1.4	2x DSI, 1x eDP 1.4 / DP 1.2 / HDMI
CSI	Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.2 (2.5 Gbps/Lane)	Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.1 (1.5 Gbps/Lane)
PCIE	Gen 2 | 1x4 + 1x1 OR 2x1 + 1x2	Gen 2 | 1x4 + 1x1
Data Storage	32 GB eMMC, SDIO, SATA	16 GB eMMC, SDIO, SATA
Other	CAN, UART, SPI, I2C, I2S, GPIOs	UART, SPI, I2C, I2S, GPIOs
USB	USB 3.0 + USB 2.0	
Connectivity	1 Gigabit Ethernet, 802.11ac WLAN, Bluetooth	
Mechanical	50 mm x 87 mm (400-Pin Compatible Board-to-Board Connector)	

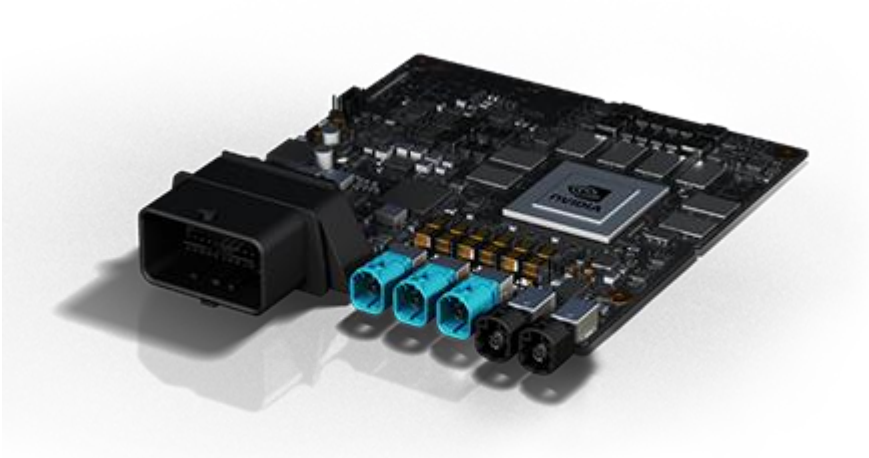
Embedded Devices: Jetson Xavier

Jetson Xavier	
GPU	512-core Volta GPU with Tensor Cores
DL Accelerator	(2x) NVDLA Engines
CPU	8-core ARMv8.2 64-bit CPU, 8MB L2 + 4MB L3
Memory	16GB 256-bit LPDDR4x | 137 GB/s
Storage	32GB eMMC 5.1
Vision Accelerator	7-way VLIW Processor
Video Encode	(2x) 4Kp60 | HEVC
Video Decode	(2x) 4Kp60 | 12-bit support
Mechanical	100mm x 87mm with 16mm Z-height (699-pin board-to-board connector)

I/O	
Display	(3x) eDP/DP/HDMI at 4Kp60 | HDMI 2.0, DP HBR3
Camera Inputs	16 lanes CSI-2, 40 Gbps in D-PHY V1.2 or 109 Gbps in CPHY v1.1 8 lanes SLVS-EC Up to 16 simultaneous cameras
PCIe	5x 16GT/s gen4 controllers | 1x8, 1x4, 1x2, 2x1 • (3x) Root Port + Endpoint • (2x) Root Port
USB	(3x) USB 3.1 (10GT/s) (4x) USB 2.0 Ports
Ethernet	(1x) Gigabit Ethernet-AVB over RGMII
Other I/Os	UFS, I2S, I2C, SPI, CAN, GPIO, UART, SD

Autonomous Driving Solutions

- Drive PX Systems
  - Sensor Fusion
  - Artificial Intelligence and Deep Learning
  - NVIDIA DriveWorks SDK
  - End-to-end HD Mapping



Autonomous Driving Solutions

DRIVE PX Xavier

DRIVE PX Xavier will deliver 20 TOPS of performance, while consuming only 20 watts of power. Packed with 7 billion transistors, a single Xavier AI processor will be able to replace today's DRIVE PX configured with dual mobile SoCs and dual discrete GPUs — at a fraction of the power consumption. Available Q4 of 2017. Learn More >

DRIVE PX FOR AUTOCHAUFFEUR

DRIVE PX configuration with two SoCs and two discrete GPUs is available today for point-to-point travel.

DRIVE PX FOR AUTOCRUISE

Small form factor DRIVE PX for AutoCruise is designed to handle functions including highway automated driving, as well as HD mapping.

GPU Supercomputers

Rank	Site	Computer/Year Vendor	Cores	R_{max}	R_{peak}	Power
1	National Supercomputing Center in Tianjin China	Tianhe-1A - NUDT TH MPP, X5670 2.93Ghz 6C, NVIDIA GPU, FT-1000 8C / 2010 NUDT	186368	2566.00	4701.00	4040.00
2	DOE/SC/Oak Ridge National Laboratory United States	Jaguar - Cray XT5-HE Opteron 6-core 2.6 GHz / 2009 Cray Inc.	224162	1759.00	2331.00	6950.60
3	National Supercomputing Centre in Shenzhen (NSCS) China	Nebulae - Dawning TC3600 Blade, Intel X5650, NVidia Tesla C2050 GPU / 2010 Dawning	120640	1271.00	2984.30	2580.00
4	GSIC Center, Tokyo Institute of Technology Japan	TSUBAME 2.0 - HP ProLiant SL390s G7 Xeon 6C X5670, Nvidia GPU, Linux/Windows / 2010 NEC/HP	73278	1192.00	2287.63	1398.61
5	DOE/SC/LBNL/NERSC United States	Hopper - Cray XE6 12-core 2.1 GHz / 2010 Cray Inc.	153408	1054.00	1288.63	2910.00

GPUs are difficult to use…

- GPUs designed for and driven by video games
  - Programming model is unusual & tied to computer graphics
  - Programming environment is tightly constrained
- Underlying architectures are:
  - Inherently parallel
  - Rapidly evolving (even in basic feature set!)
  - Largely secret
- Can't simply "port" code written for the CPU!

Courtesy David Luebke, University of Virginia

GPUs were difficult to use…

- GPUs used to be designed for and driven by video games
  - Programming model and programming environments are now more general and programmer friendly
- Underlying architectures are:
  - Inherently parallel (but easier to program)
  - Rapidly evolving (but now have standards!)
  - Largely secret (we can code independent of the architecture)
- "porting" the code written for the CPU much simpler with CPU like programming

The Ox vs. Chicken Analogy (1)

If you were plowing a field, which would you rather use: Two strong oxen or 1024 chickens?*

Seymour Cray (*1925-1996)*

Founder of Cray Research

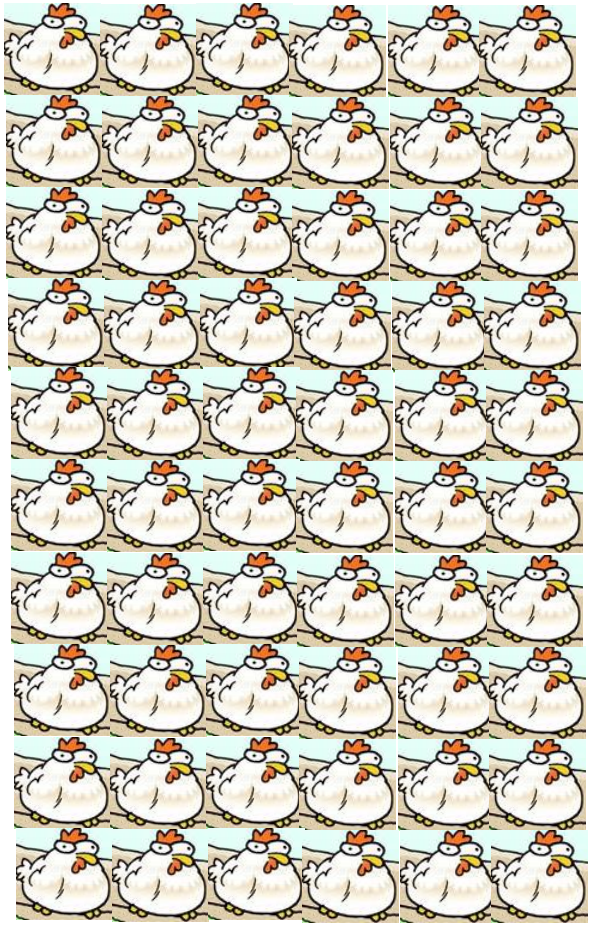
plowing: tarla sürmek*

The Ox vs. Chicken Analogy (2)

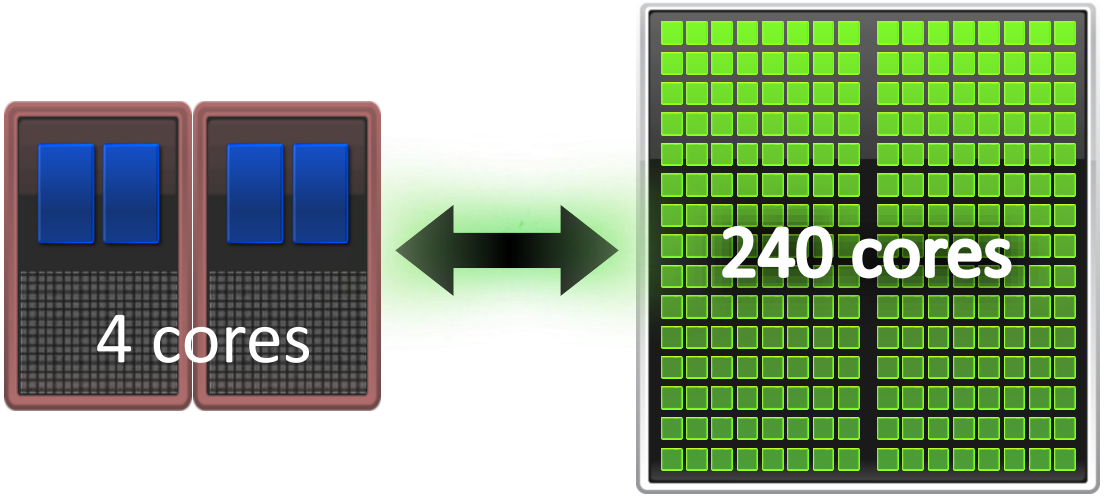
Single core Ox

vs.

60-core chicken



The Ox vs. Chicken Analogy (3)



CPU + GPU Co-Processing

Heterogeneous Computing

The Ox vs. Chicken Analogy (4)

If you were plowing a field, which would you rather use?

The Ox vs. Chicken Analogy (5)

- The CPU is good at performing sequential calculations, I/O, and program flow.

- The GPU is perfectly suited for performing massive parallel calculations.

Massively Parallel Computing

- Intel i9-9900X CPU
  - 10 cores
  - can execute 20 threads.
- GTX 1080TI GPU
  - 3584cores
  - can execute 57344 threads

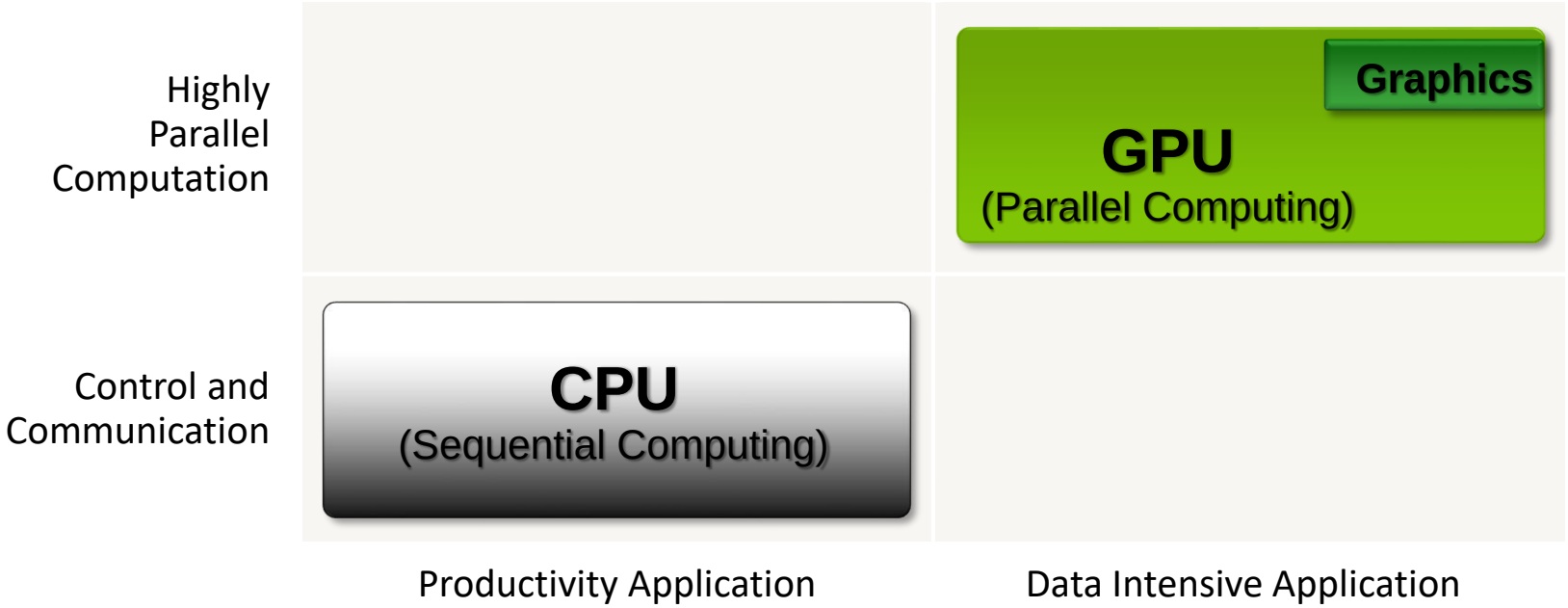
Winning Applications Use Both CPU and GPU (1)

- Use CPUs for sequential parts where **latency** matters
  - CPUs can be several times faster than GPUs for sequential code
  - CPUs are designed to reduce latency of operations within the same thread
    - Uses low latency on chip caches and low latency arithmetic units.
    - Sophisticated design consume chip area and power that could be otherwise used to provide more arithmetic units and memory access channels.

Winning Applications Use Both CPU and GPU (2)

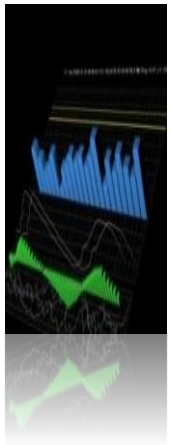
- GPUs for parallel parts where throughput wins
  - GPUs can be several times faster than CPUs for parallel code
  - GPUs are designed throughput oriented
    - Optimized for execution throughput of massive number of threads.
    - Chip area is maximized and power budget is dedicated to floating point calculations.
    - GPUs have higher memory bandwidth – CPUs need to satisfy the requirements of the legacy operating systems, applications and I/O devices that make memory bandwidth more difficult to increase.

Heteregenous Computing



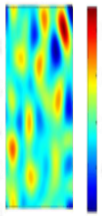
GPU Computing Domains


Oil & Gas


Finance


Medical


Biophysics


Numerics


Audio


Video


Imaging

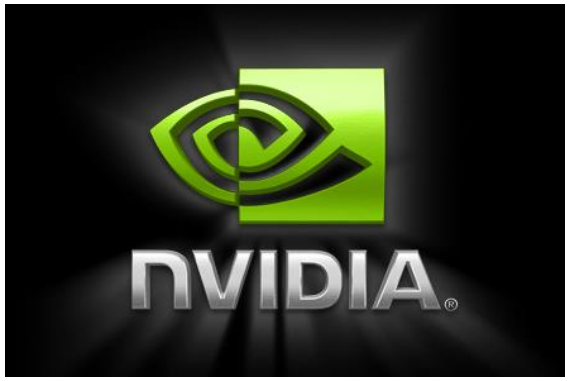
GPU Programming

- Programs used to be implemented through graphics API : inflexible, lots of limitations, needed a lot of effort to program
  - OpenGL and Direct3D techniques were needed to program: a computation must be expressed as a function that paints a pixel in some way to execute.

- GPGPU: General Purpose computation on the GPU

CUDA

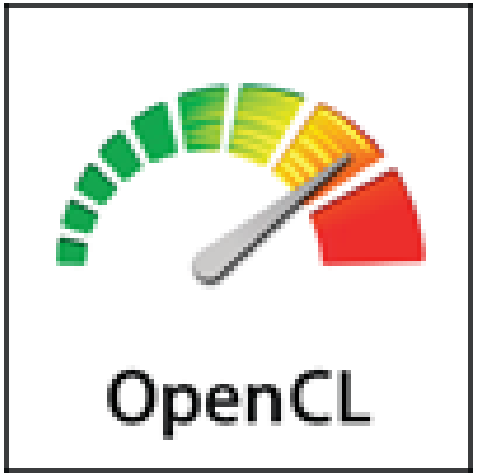
- **C**ompute **U**nified **D**evice **A**rchitecture
- Released in 2007
- Developed by NVIDIA
- Essentially C language with some extensions



OpenCL (1)

- Open, royalty-free standard for cross-platform, parallel programming.
- Initially developed by Apple Inc., in collaboration with technical teams at AMD, IBM, Intel, and NVIDIA.

- OpenCL 1.0 : released in May 2009
- OpenCL 1.1: June 2010
- OpenCL 1.2 : November 2011
- OpenCL 2.0 : October 2013



OpenCL (2)

- OpenCL provides a heteregenous environment. The code can run on different platforms, i.e. CPU and GPU.

In my experience:

- More difficult to program.
- Inferior performance compared to CUDA.
- ~~Currently your only choice~~ if you want your code to run on different platforms of heteregenous environments.

Stream Framework

- GPU programming solution by AMD/ATI
- Based on *Brook*, initially developed by Stanford Uni.
- ATI Stream now supports OpenCL.

DirectCompute

- Developed by Microsoft, Windows 7 has native support.
- Part of DirectX 11 API (supported by DirectX 10).
- Works independent of GPU hardware.

OpenACC

- Provides a set of compiler directives, library routines and environment variables to write data-parallel code to run on accelerator devices.

- Initially developed by the Portland Group (PGI), Cray Inc. and NVIDIA with support from CAPS enterpise.

GPU Producers

- **NVIDIA** is currently leading with easy to use development tools and high performance.
- **AMD/ATI** is providing competing performance but with less advanced development tools.

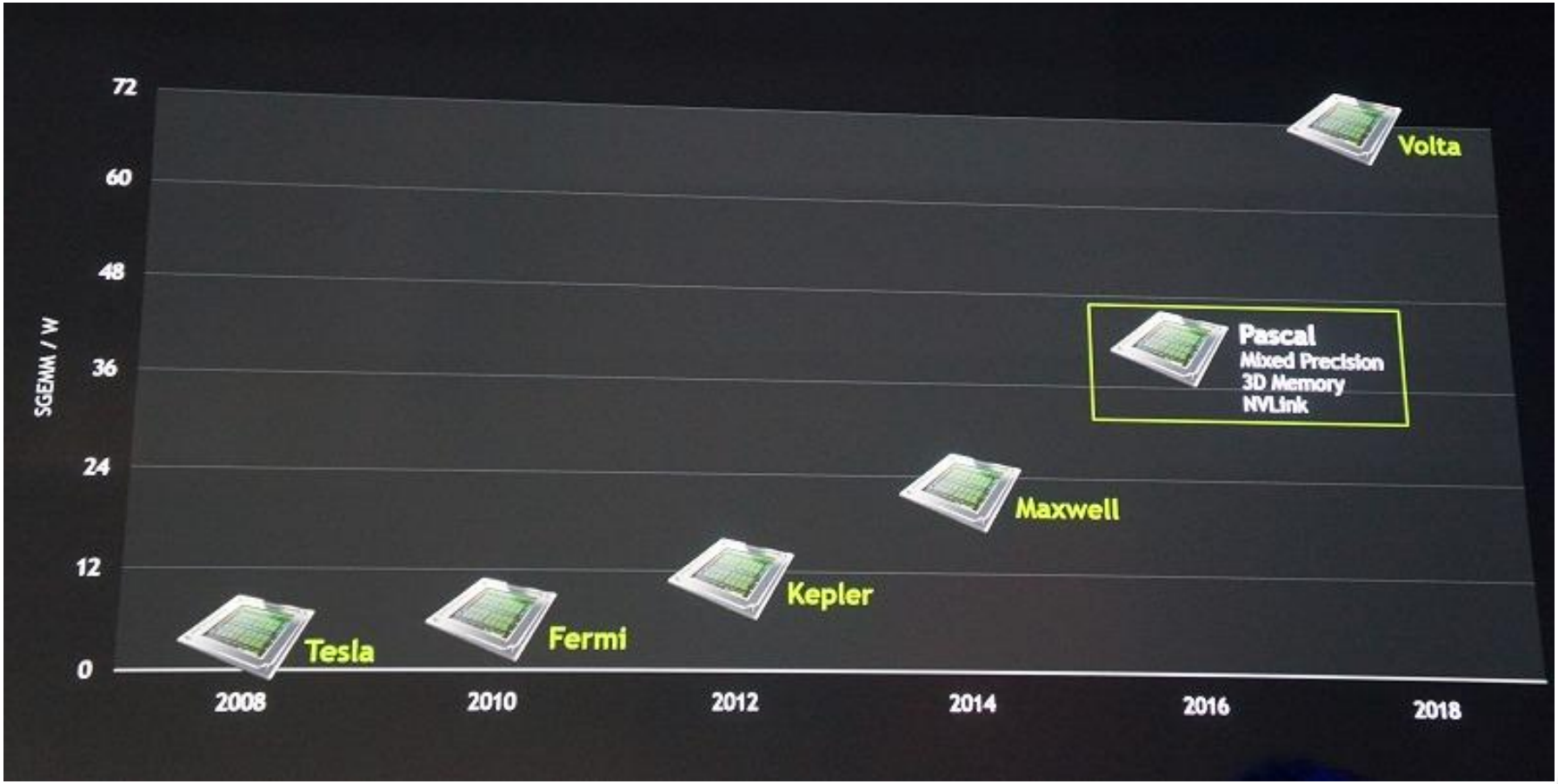
Intel Xeon Phi Coprocessor

- Intel is investing in Many Core CPU architectures.

Intel® Core™ i9-9980XE Extreme Edition Processor

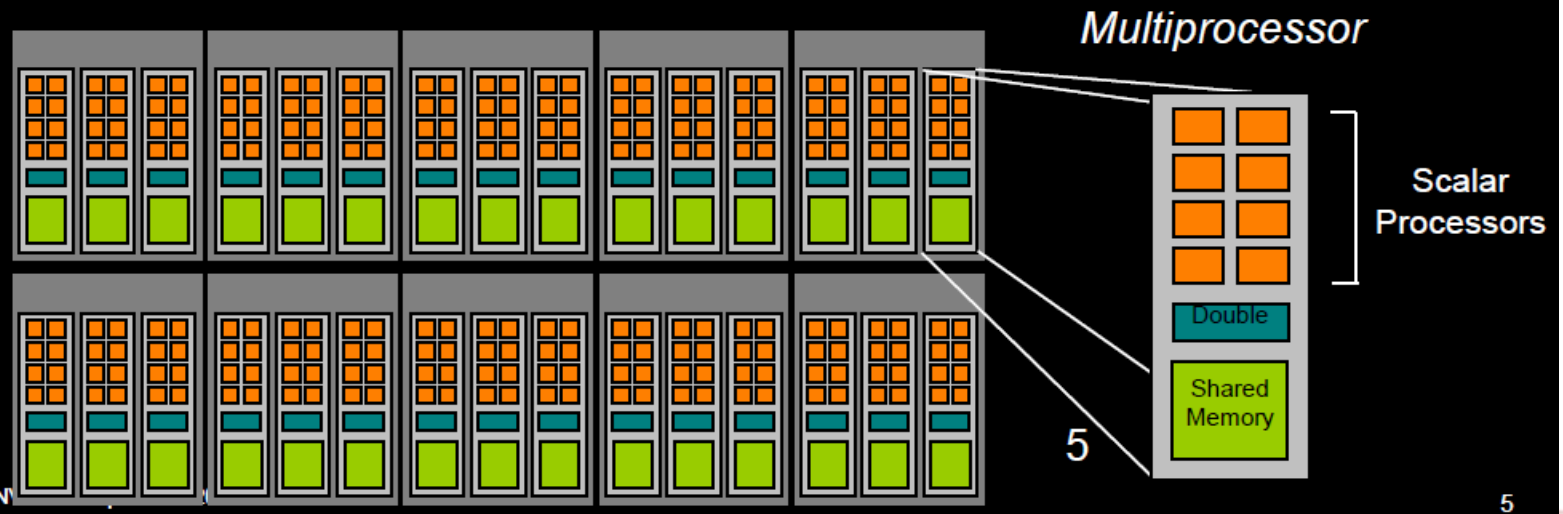
- 24.75 MB SmartCache Cache
- 18 Cores
- 36 Threads
- 4.40 GHz Max Turbo Frequency
- XE - Extreme performance and mega-tasking, unlocked
- 9th Generation

NVIDIA GPU Roadmap

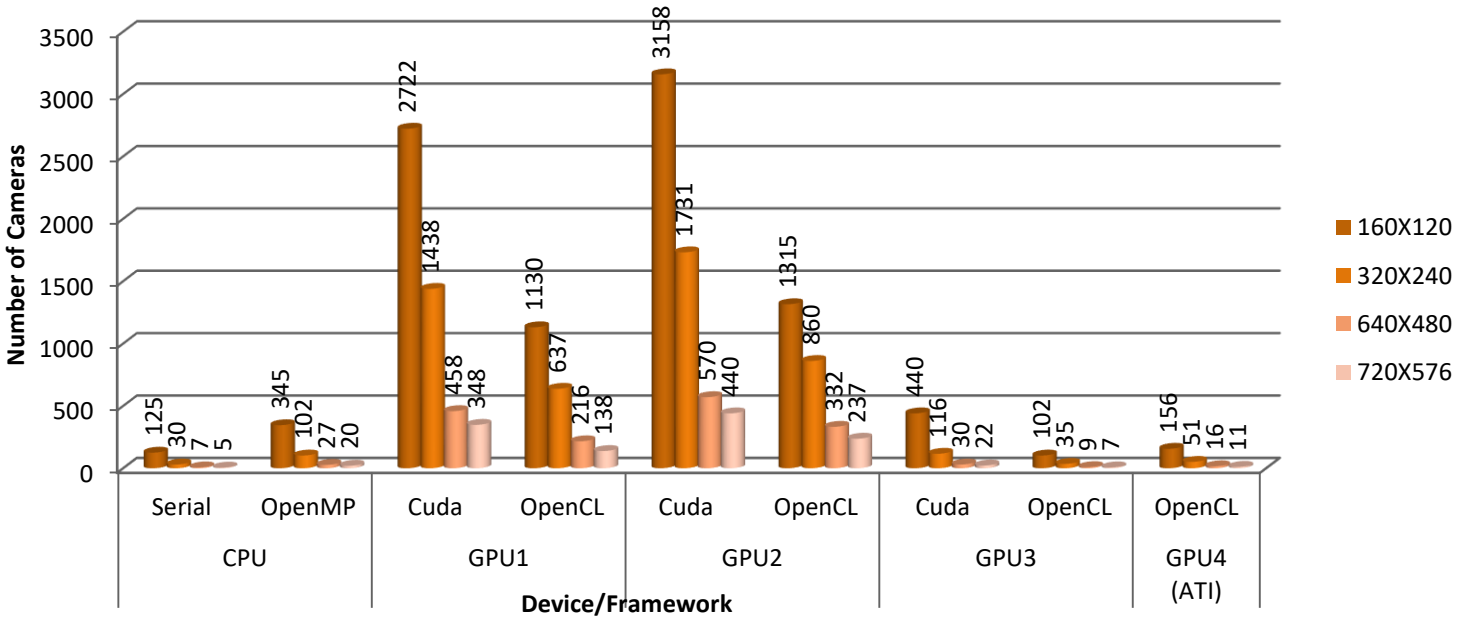


GT200 Architecture

- Device contains 30 Streaming Multiprocessors (SMs)
- Each SM contains
  - 8 scalar processors
  - 1 double precision unit
  - 2 special function units
  - shared memory (16 K)
  - registers (16,384 32-bit=64 K)

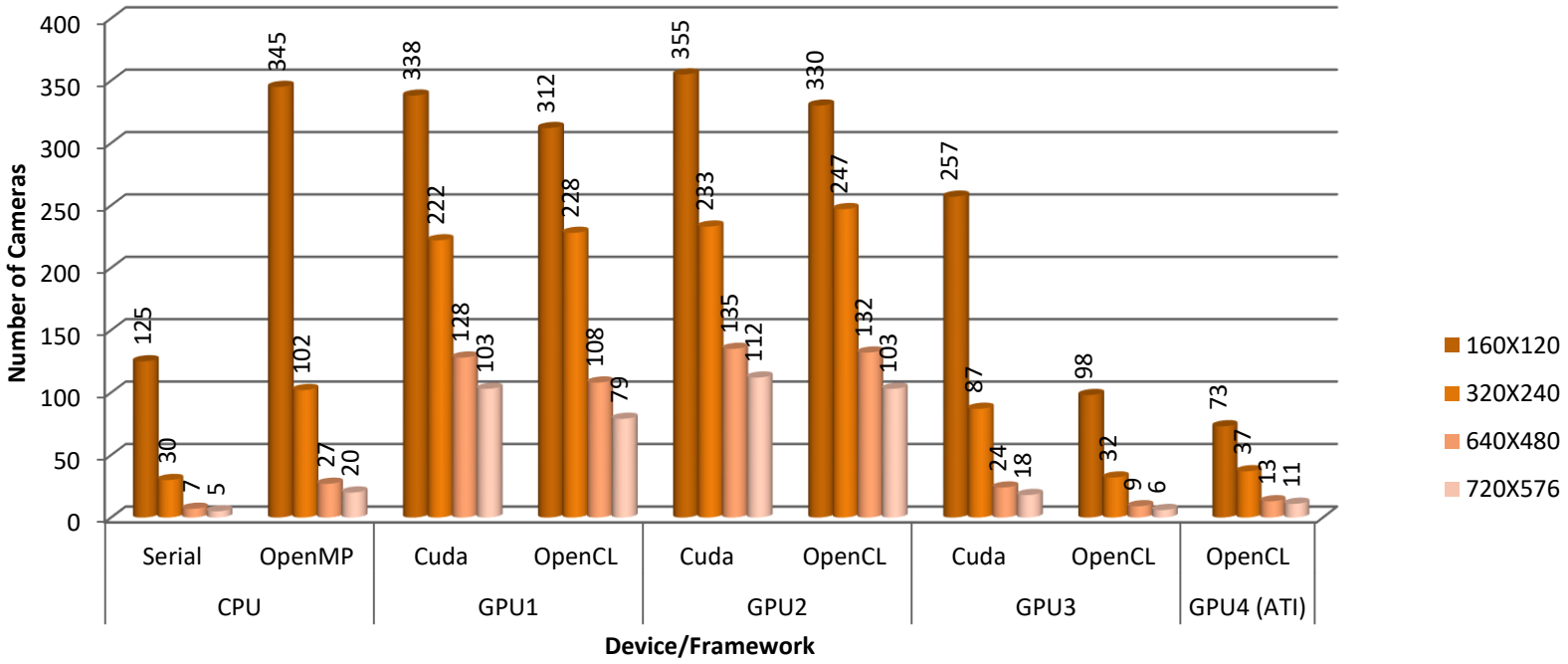


GPU Computing Bottlenecks (1)



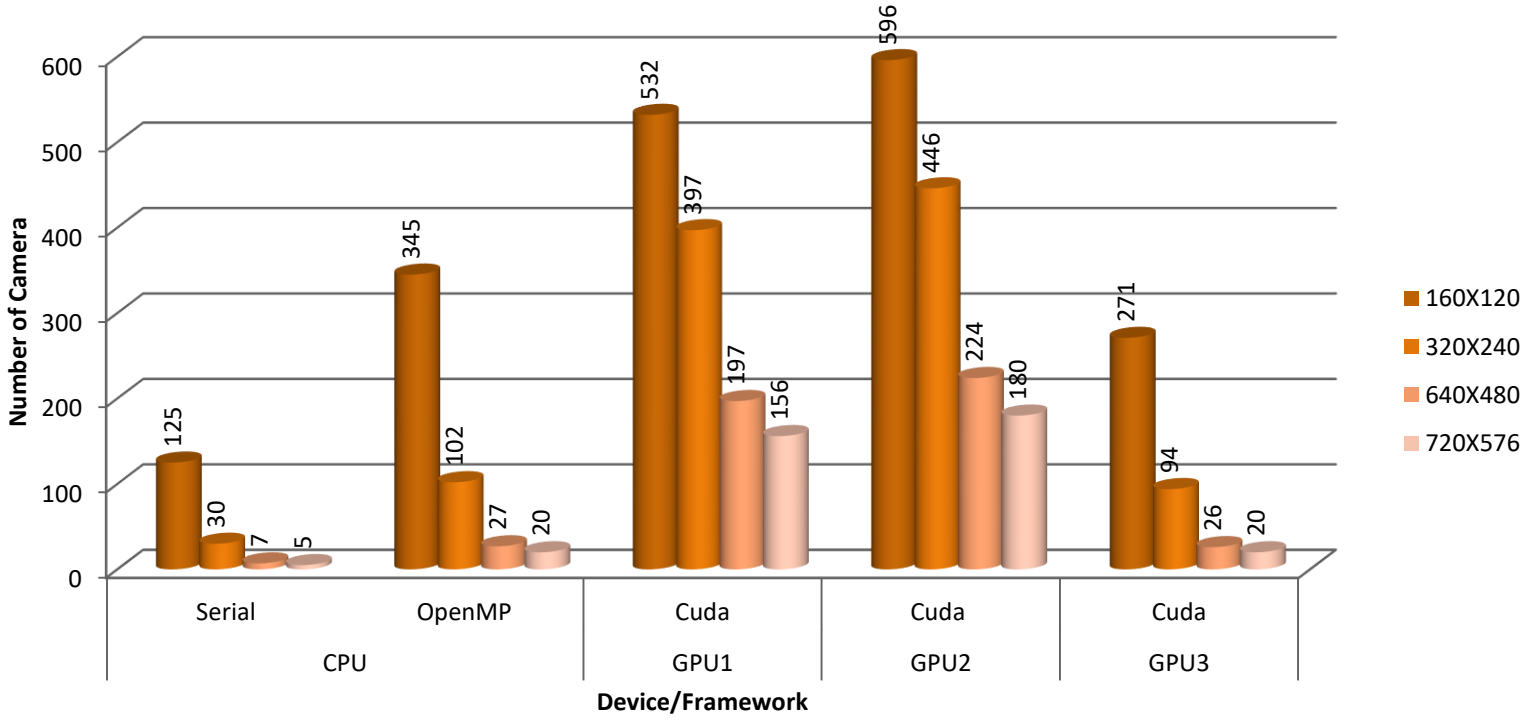
Maximum number of cameras that can be supported for real-time background subtraction using CPU and GPUs (**I/O not included**).

GPU Computing Bottlenecks (2)



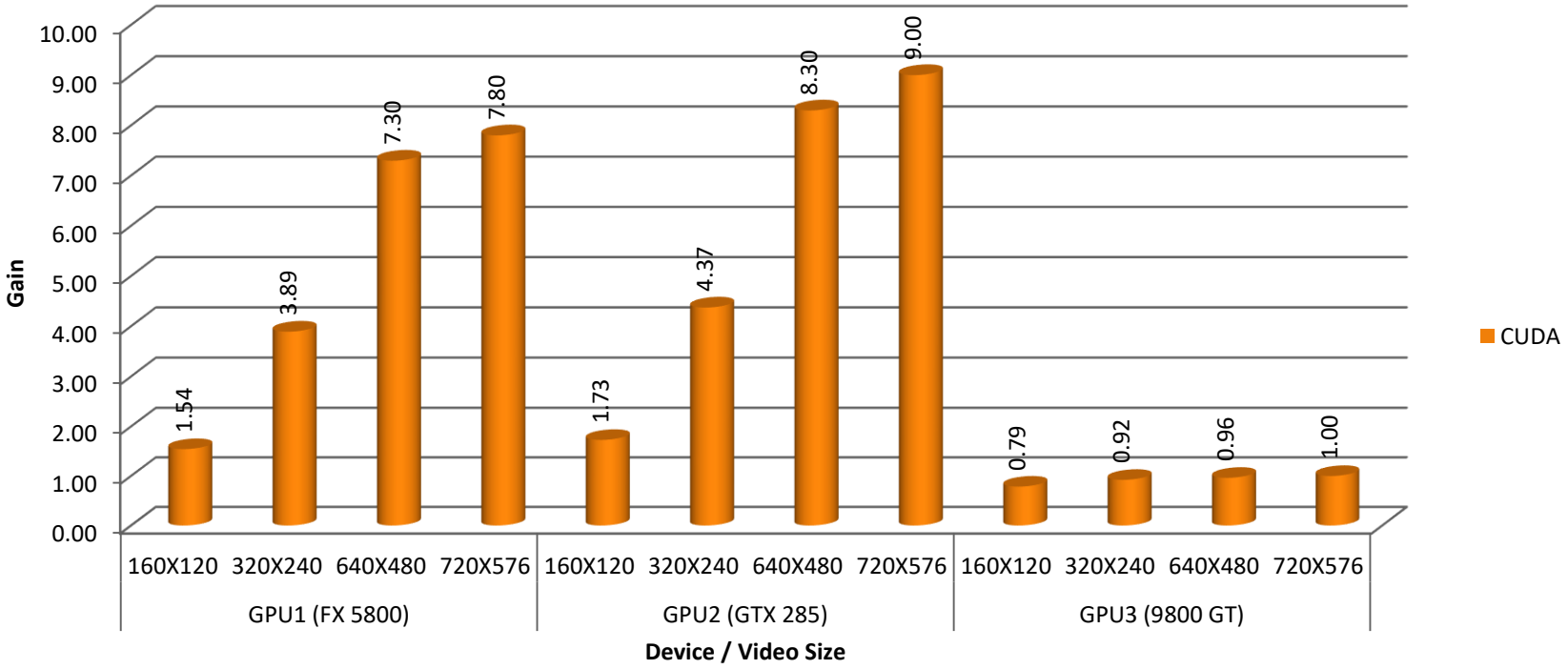
Maximum number of cameras that can be supported for real-time background subtraction using CPU and GPUs (**I/O included**).

GPU Computing Bottlenecks (3)



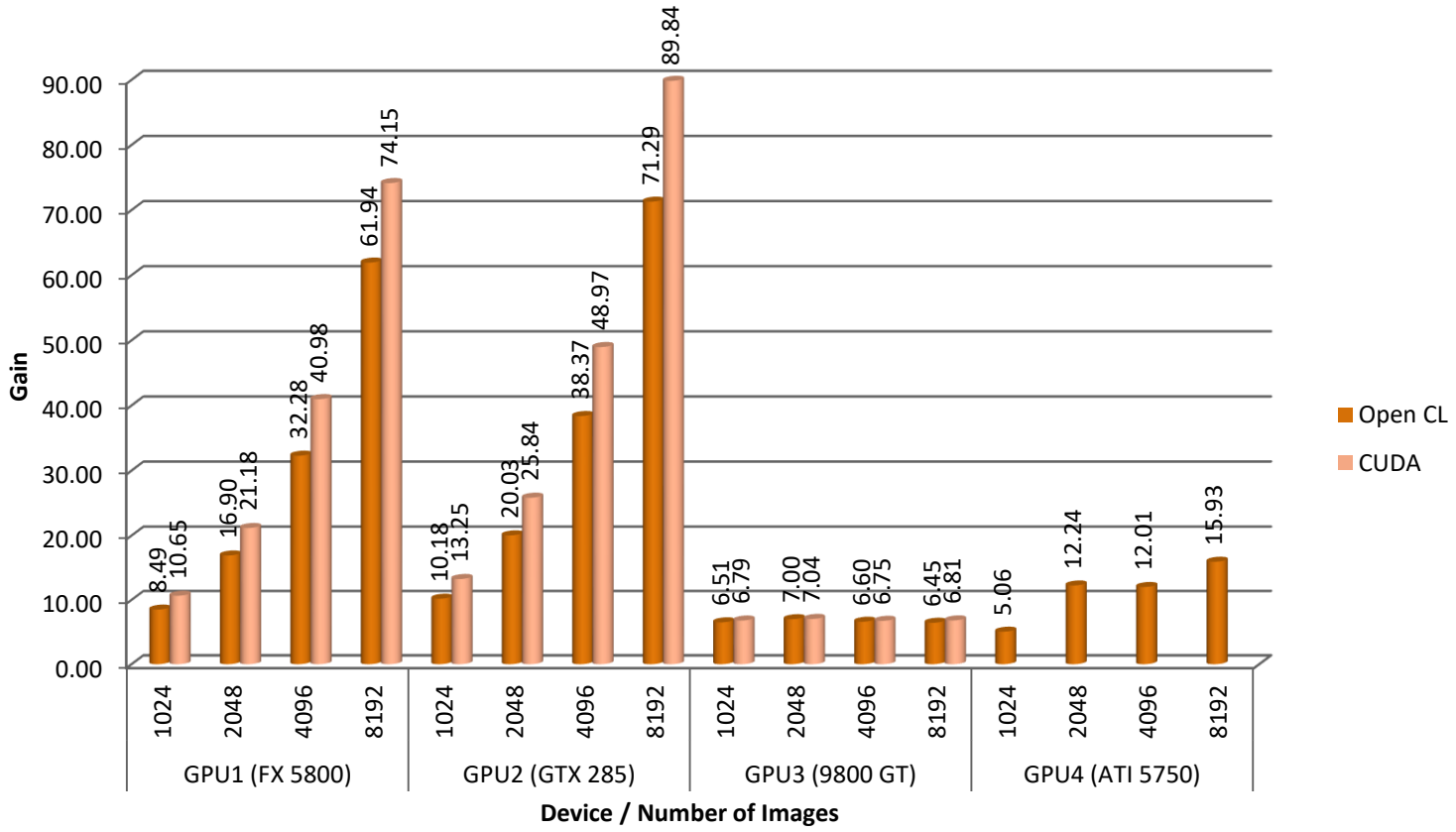
Maximum number of cameras that can be supported for real-time background subtraction using CPU and GPUs (**Asynchronous I/O**).

GPU Computing Bottlenecks (4)



Performance gain ratio of various GPUs vs. OpenMP CPU (Intel i7 920) implementation (Asynchronous implementation).

GPU Computing Bottlenecks (4)



Performance gain for Pearson cross correlation implementation over OpenMP implementation on the CPU for 160x120 image sizes.

Summary

- GPUs have the potential of increasing the computation speed drastically.
- The algorithm has to be data parallel.
- Memory operations might be a bottleneck.
- Development, optimization and debugging on GPU not trivial.
- Different GPUs have to be tested, most expensive one might not be the best for the problem.

Not all the problems are suitable for GPU,
one should decide on using GPU depending on the problem in hand