A History of Graphics Hardware

What do we want?

Computer-generated imagery (CGI)
in real-time.

computationally extremely demanding

- full HD at 60 Hz:

$$1920 \times 1080 \times 60\ \text{Hz} = 124\ \text{Mpx/s}$$

- And that's just output data!

Problem

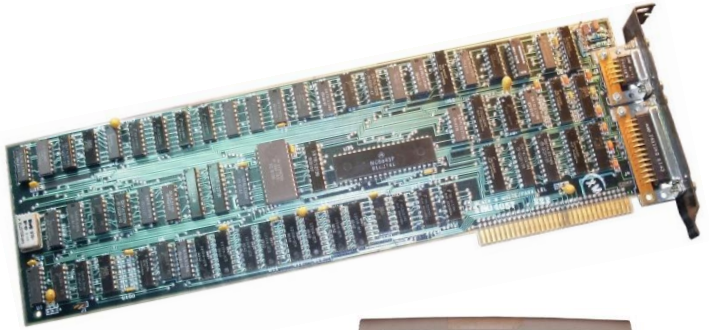
How can we compute millions of Pixels

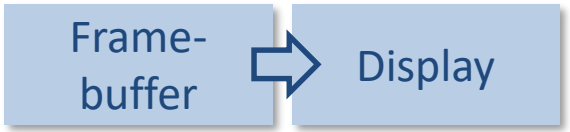
- making up an image of a complex 3D scene
- in real-time?

→ specialized hardware

PC Graphics 1980

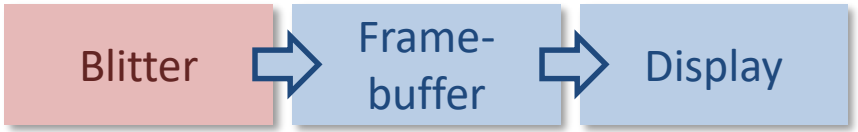
- Video display controller
- IBM Color Graphics Adapter
  Display a color buffer
- Direct drive CRT monitor or TV
- Resolution 320x200
- 16 colors





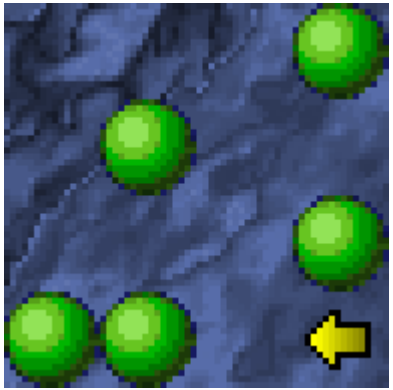
PC Graphics 1987

- Video display processor
- Commodore Amiga
- Blitter co-processor for image bitmap copy
- Resolution: 704x 484
- 4096 colors

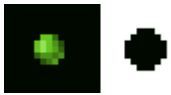


Blitter 1987

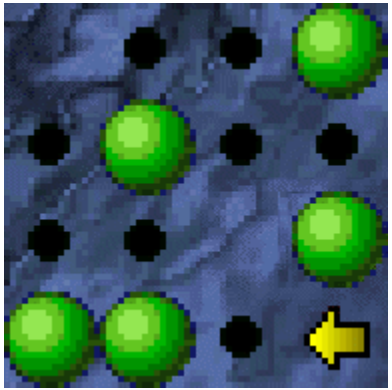
Framebuffer



BMP + mask



AND with mask



OR with BMP

name stems from “bit block transfer”

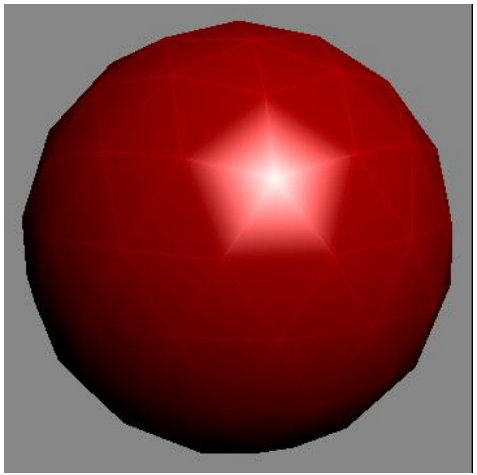
Commander Keen

PC Graphics 1996

- 3dfx Voodoo1
- Triangle rasterization
- 3D only, no 2D support
- Resolution 640x480
- 50 MHz, 45MPixels/s, 4MB RAM
- Process: 500nm
- 1 Texture unit, 1 Raster Unit

Rasterizer → ROP → Frame-buffer → Display

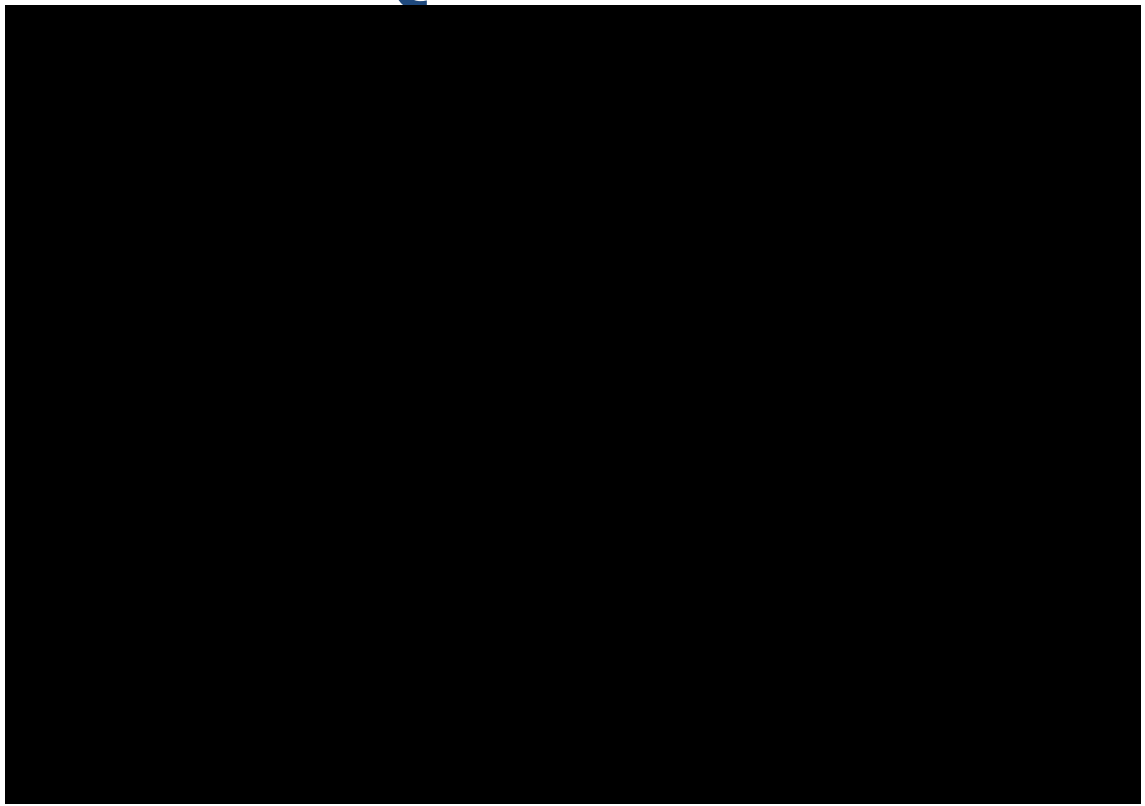
PC Graphics 1996



Gouraud Shading

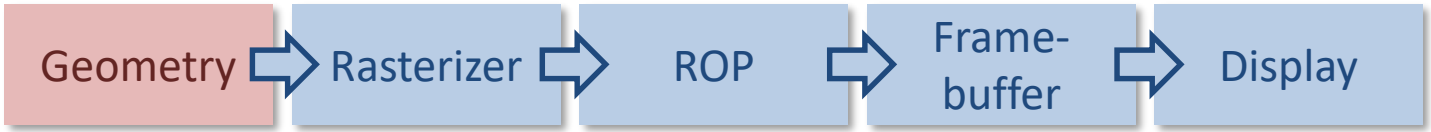
Mip-Mapping

Quake 1



PC Graphics 1999

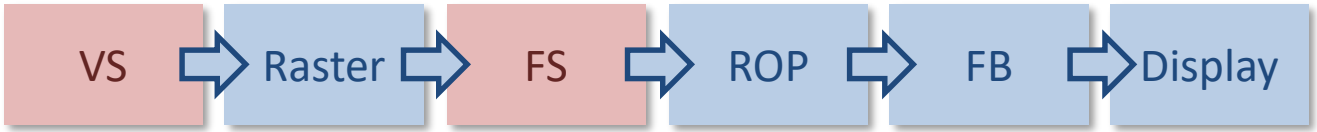
- Nvidia GeForce 256
- Hardware Transform and Lighting
- First ‘GPU’
- Process: 220nm
- 120MHz, 480MPixels/s, 64MB RAM
- 1 geometry processor, 4 pixel processors, 4 texture units, 4 ROP units



Quake III Arena

PC Graphics 2001

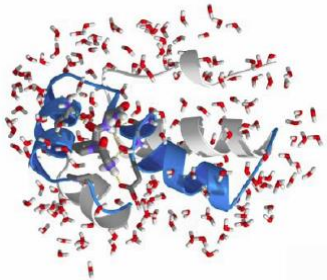
- Nvidia GeForce 3
- Programmable Vertex and Pixel Shader
- Process: 150nm
- 200 MHz, 800MPixels/s, 64MB RAM
- 1 geometry processor, 4 pixel processors, 8 texture units, 4 ROP units



Chameleon Shader Demo

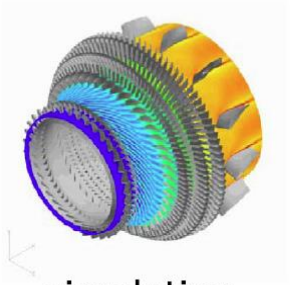
PC Graphics 2001

- “Abuse” pixel shaders for parallel programming
- “Shader Meta Programming”, Brook for GPUs

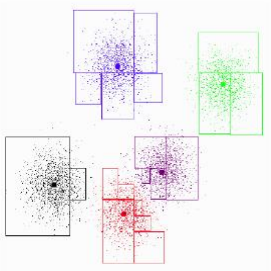


bioinfomatics

rendering



simulation



statistics

PC Graphics 2006

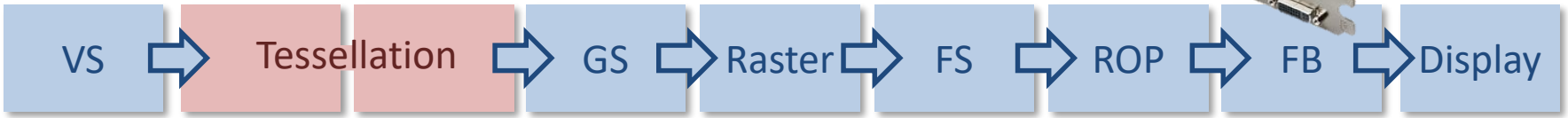
- Nvidia GeForce 8
- Unified Shader Model
- CUDA 1.0
- Process: 90nm
- Shader clock 1500 MHz
- 576 GFLOPs

Nalu

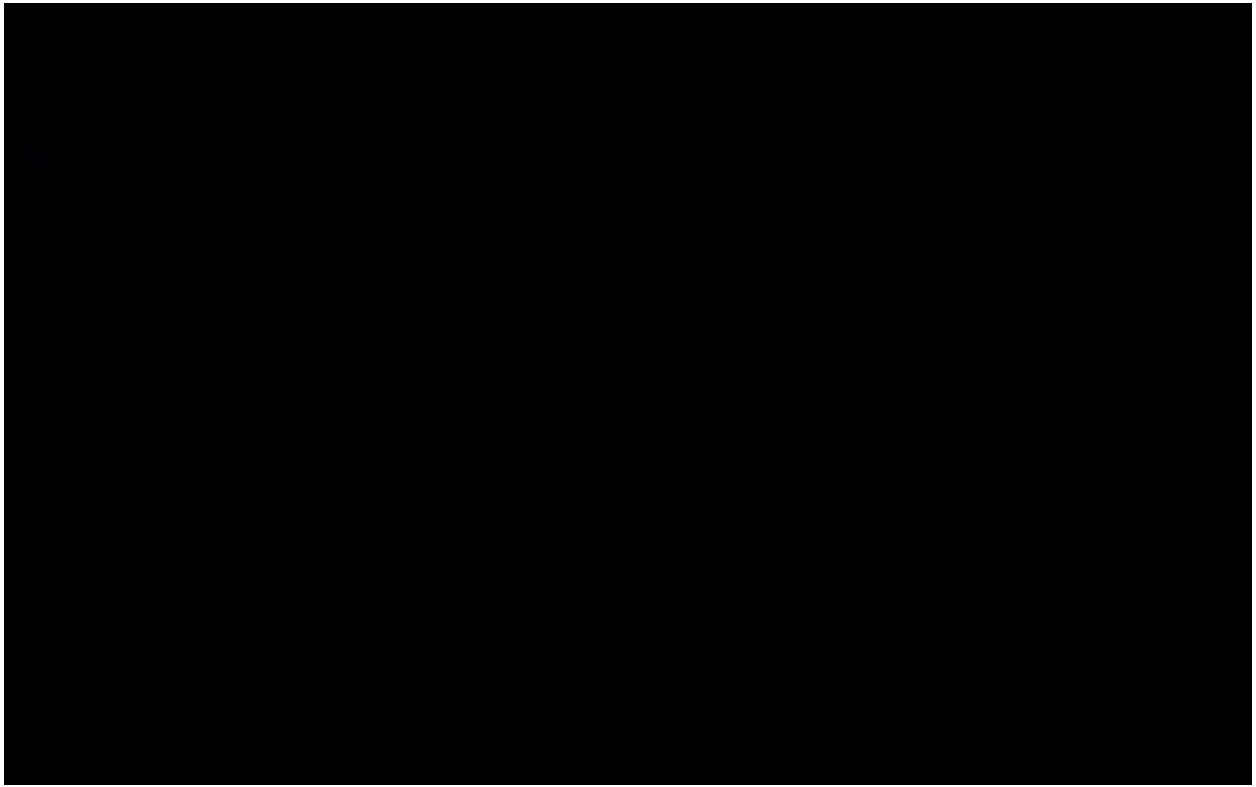

Global
Shading
Navigation
Animation
Draw
Emotions
About
NVIDIA.

PC Graphics 2009

- ATI Radeon HD 5000
- Tessellation
- Process: 40nm
- 850 MHz
- 2720 GFLOP

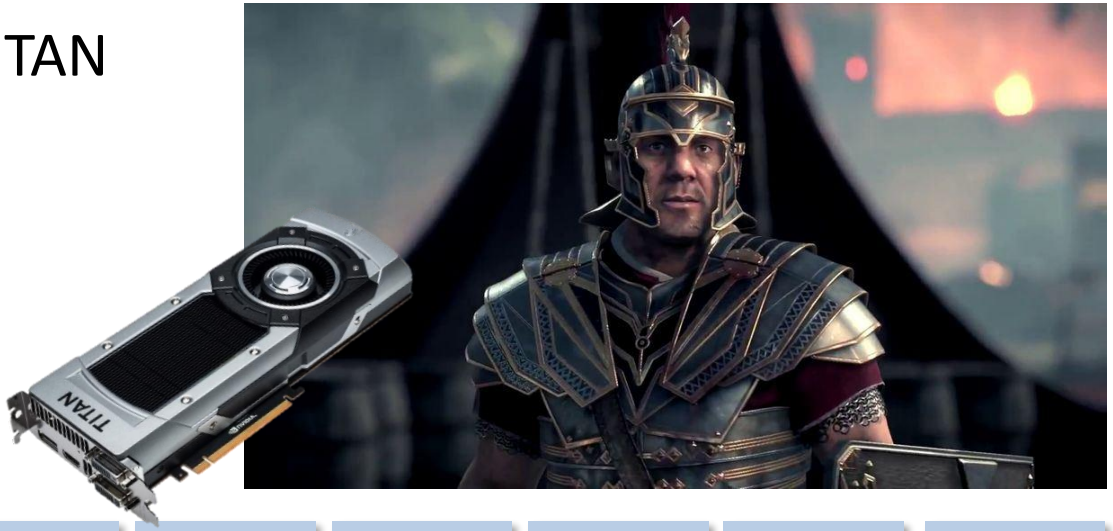


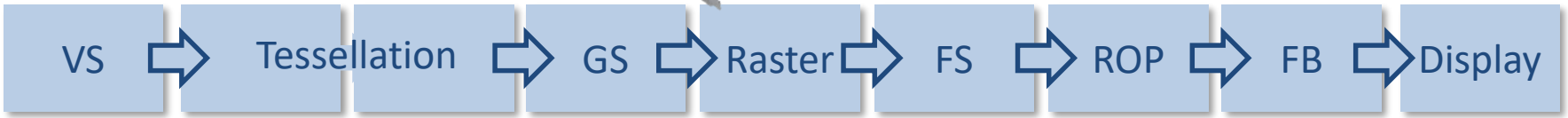
Unigine Heaven: Tessellation



PC Graphics 2013

- Nvidia Geforce GTX TITAN
- Process: 28nm
- 890 MHz
- 5100 GFLOP @ 250W

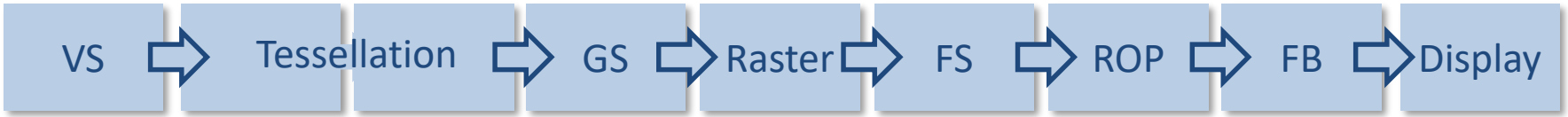




Digital Ira

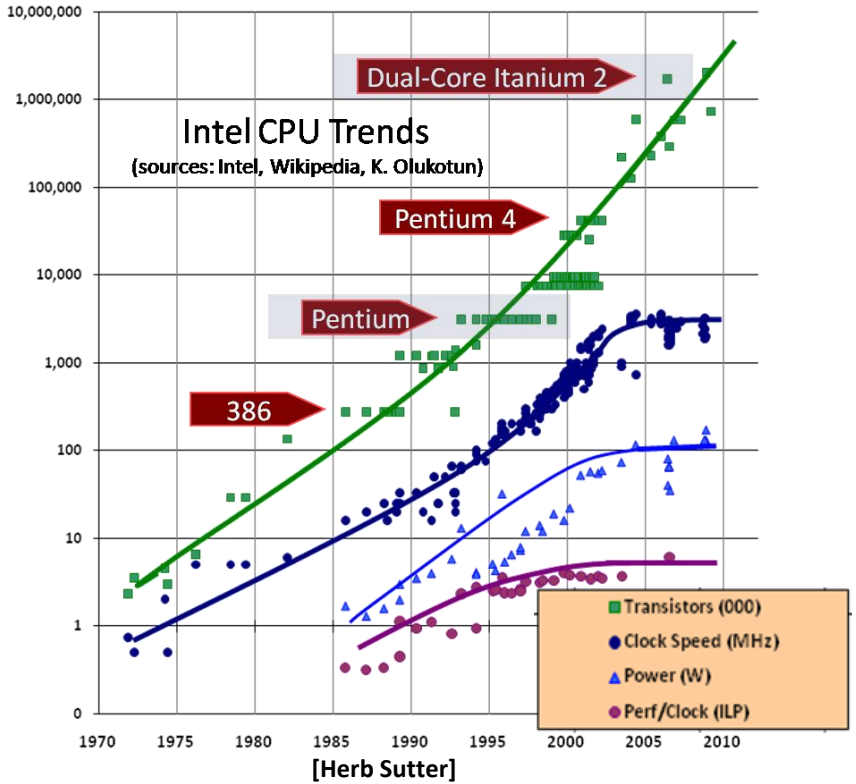
PC Graphics 2015

- Nvidia Geforce GTX 980
- Process: 28nm
- 890 MHz
- 4900 GFLOP @ 165W



NVIDIA Turf

Computing 1970-2005



[Herb Sutter]

- free performance lunch is over
- in 2005 processors hit the power wall
- clock rates cannot be increased significantly
- only way out: parallelism

Power Wall

the good old days

- limited by number of transistors
- use as much power as you want

today

- limited by energy consumption
- transistors basically “free” (can put more on a chip than can afford to turn on)

We might have many specialized cores on one chip, but only use the ones best suited for the current problem.

ILP Wall

the good old days

- Huge performance gains from superscalar design

today

- Diminishing returns on spending more hardware on instruction-level parallelism (ILP)

We need to provide parallelism at a higher level to get more computations done.

Memory Wall

the good old days

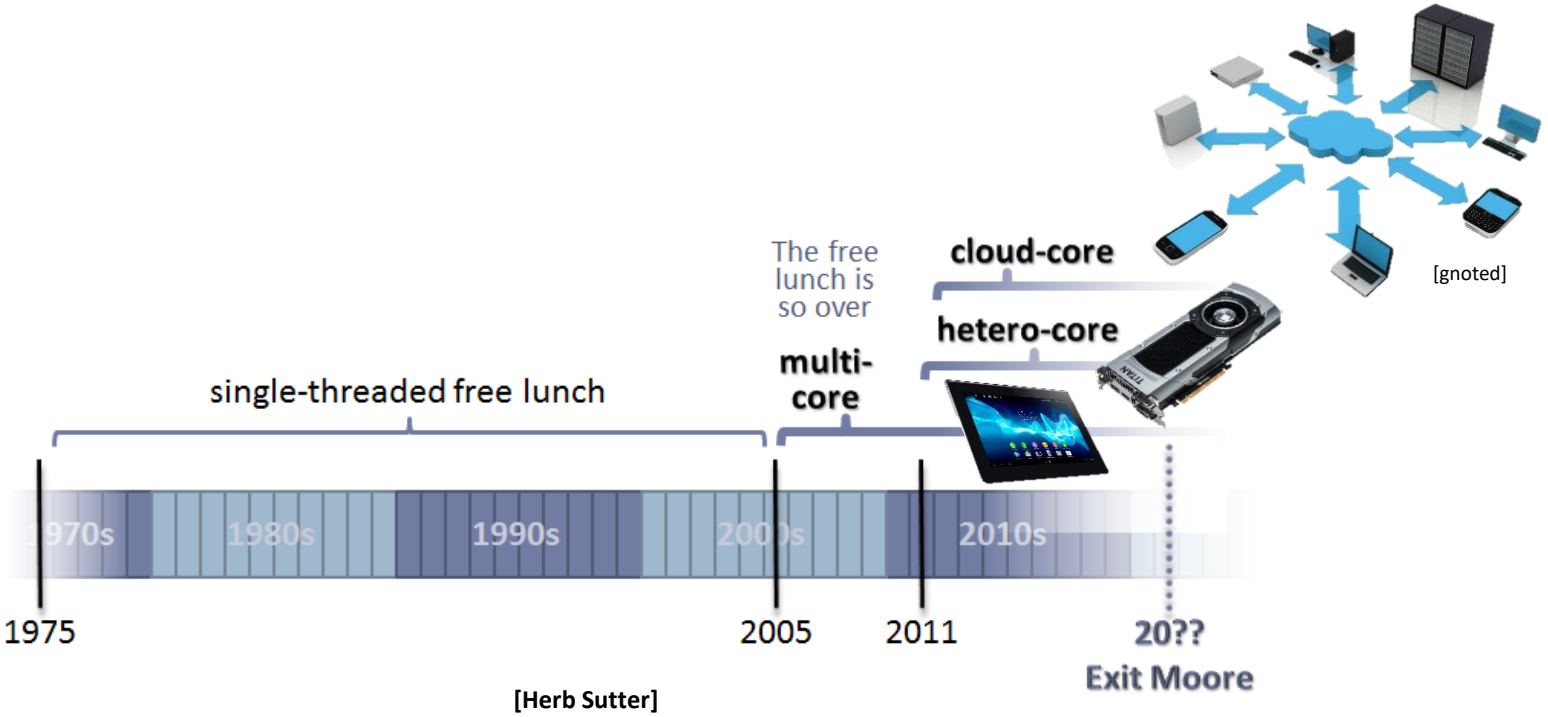
- Instructions are costly
- Memory access is fast (precompute and load whenever possible)

today

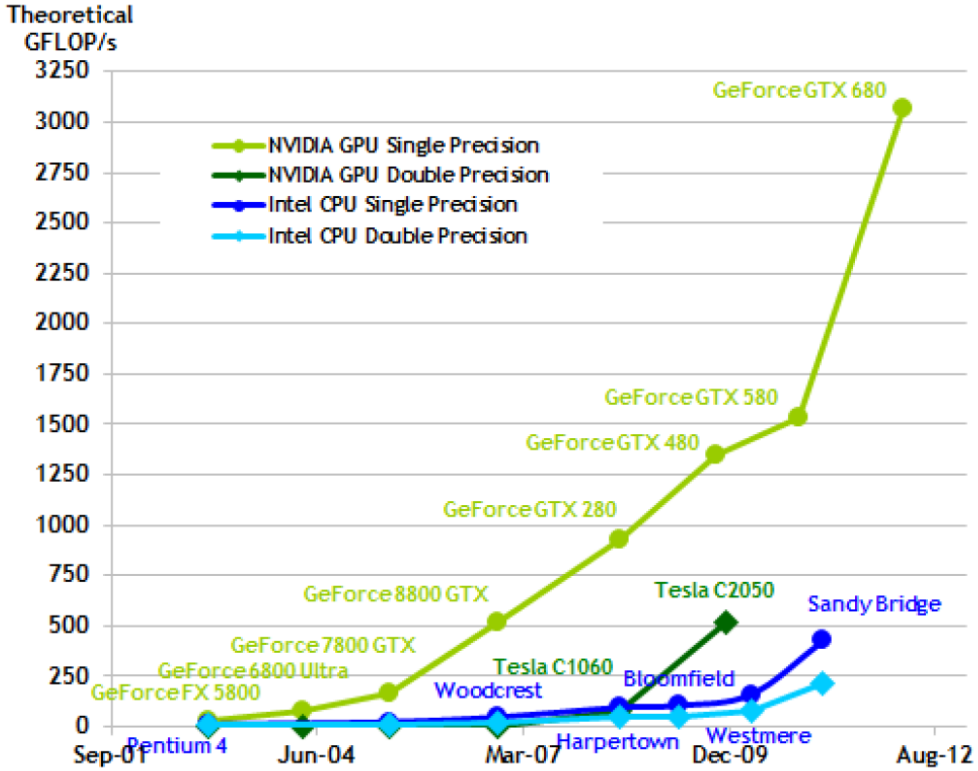
- Memory is slow (>200 cycles to go to DRAM)
- Instructions are fast

We need to avoid memory transaction and rather compute things on demand.

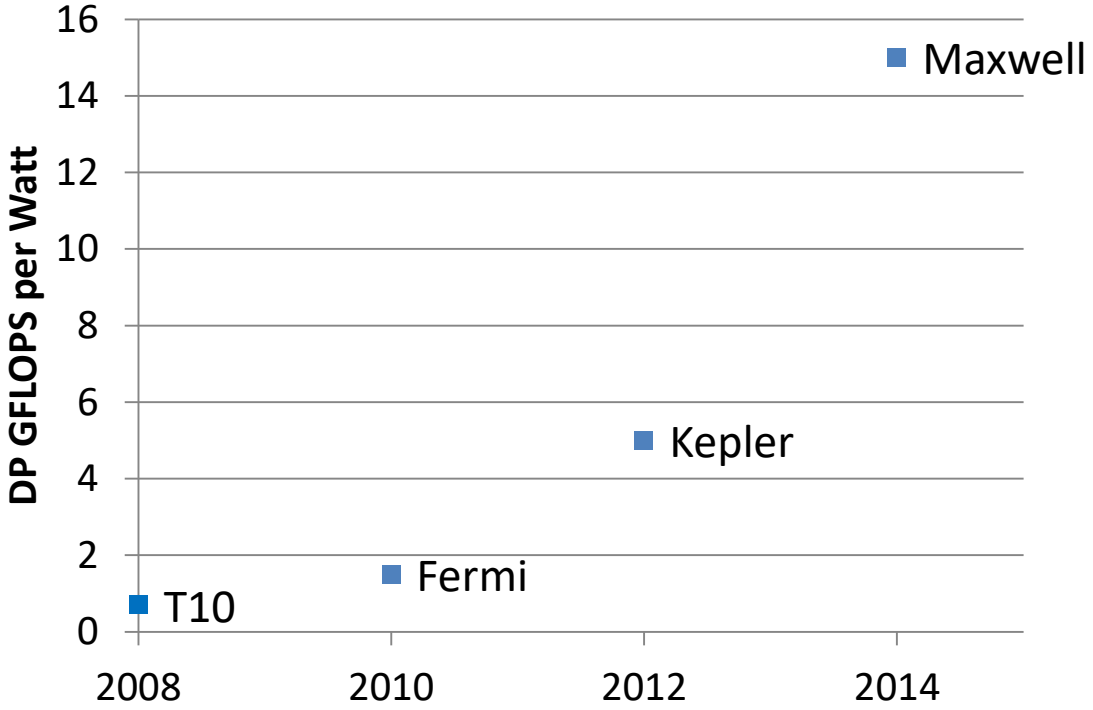
Parallel Computing



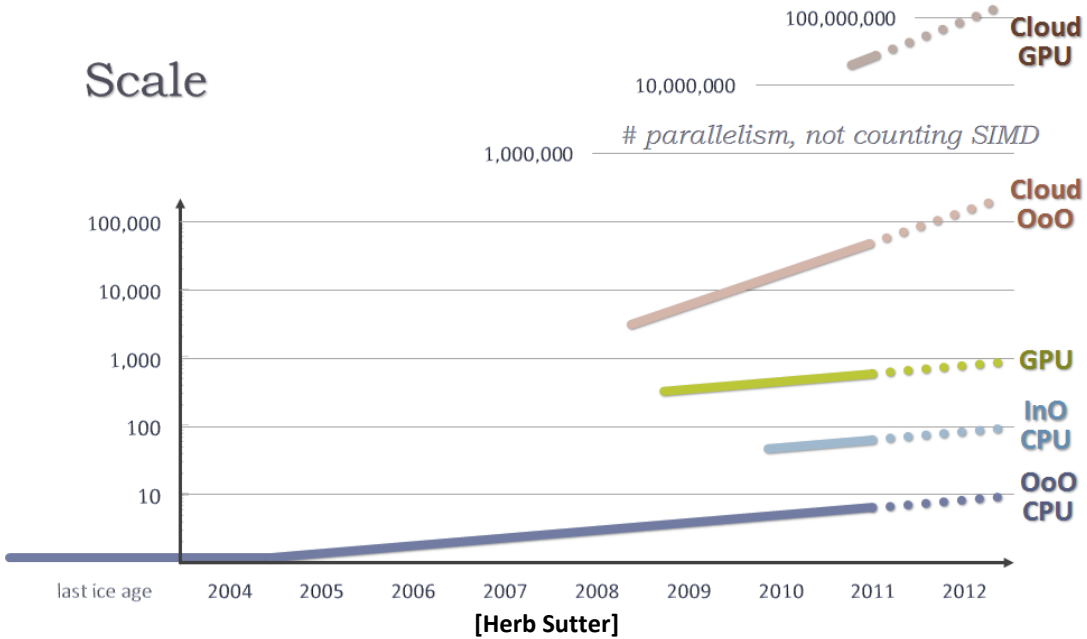
GPU performance



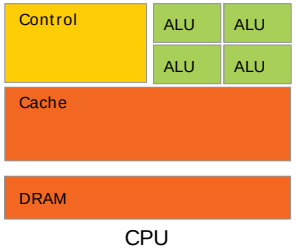
Performance per Watt

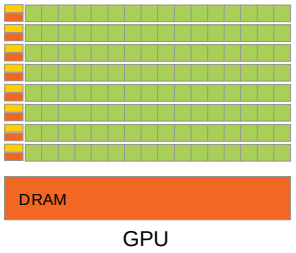


Computing Trends



[Herb Sutter]





Questions

Paradigms

- GPU programming paradigms generalized
  - Latency Hiding vs Latency Avoidance
  - Fine-grained Parallelism vs Heavyweight Threads
  - Constrained Based Synchronization vs Global Barriers
  - Adaptive Locality Control vs Static Data Distribution
  - Move Work to the Data vs Move Data to the Work
  - Message Driven Computation vs Message Passing