History – GPUs

How Did We Get Here?

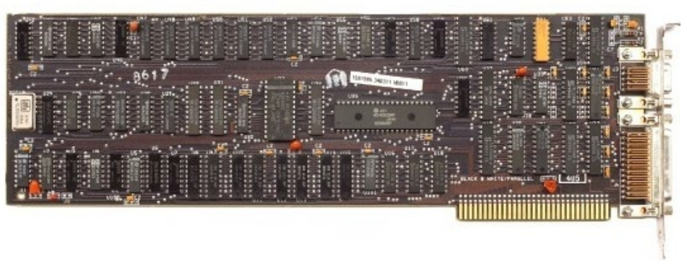
Video cards (GPUs)

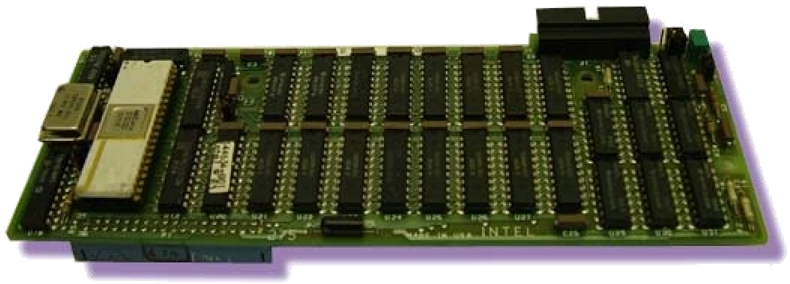
What do these things do for us?

A quick step back

- Early video cards
  - “Video shifters” or “Video Address Generators”
  - Basically a pass-through from processor to display
  - Converted data to a compatible display signal
- Then came dedicated display adapters

Old school graphics cards





- IBM’s Monochrome Display Adapter
  - Released in 1981
  - Displays ASCII characters on a screen
  - 80 columns x 25 rows
- Intel’s iSBX 275 Video Graphics Controller Multimedia Board
  - 8 colors @ 256x256
  - Or 512x512 monochrome!
  - Only $1000 (in 1982)

Plotting pixels

- How did games change pixel colors?
  - One pixel at a time (or one “block” at a time)
  - Big loops that drew the screen

- Resolution was lower (thankfully)
  - 256x256 = 65K pixels
  - 1920x1080 = 2 Million pixels

The 90’s – Lots of colors

- More resolutions and 32K colors!
  - 640x480 @ 72Hz
  - 800x600 @ 60Hz
  - As much as 1MB of video RAM!

- Started to add graphics features
  - Line drawing & color fill
  - Bitwise memory copies (Bit BLT)

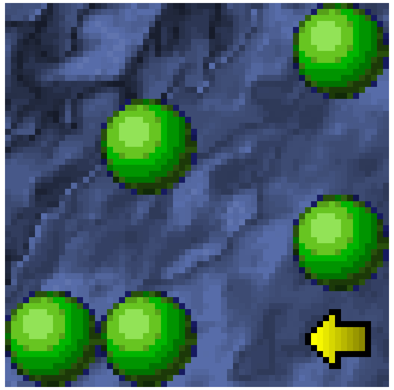
Bit BLT: Bitwise BLock Transfer

- Block Transfer
  - Copies data from one place in memory to another
  - For instance – From CPU RAM to Video RAM
- Copying/overwriting a “block” on the screen isn’t pretty
  - Bitwise operations to the rescue

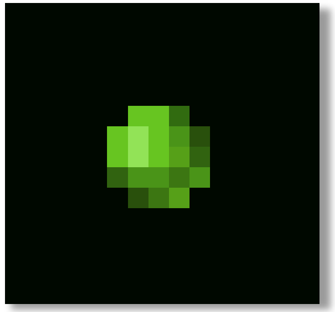
Bit BLT

- Combining data (images) using AND & OR

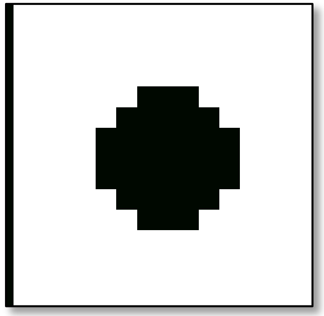
Current Image on Screen



New Sprite



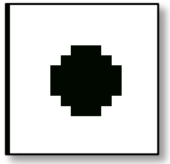
Mask

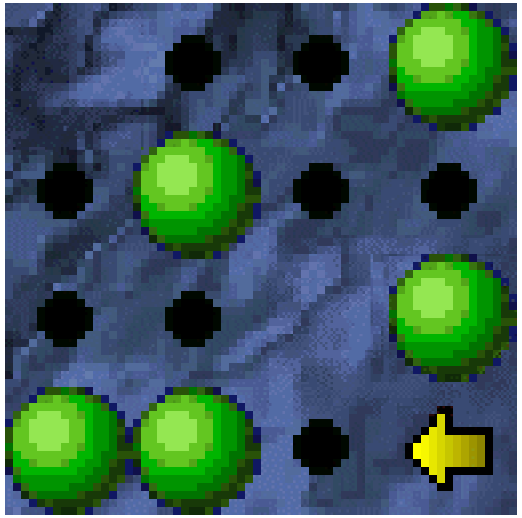


0 = use new color
1 = keep old color

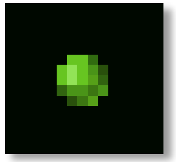
Bit BLT Steps

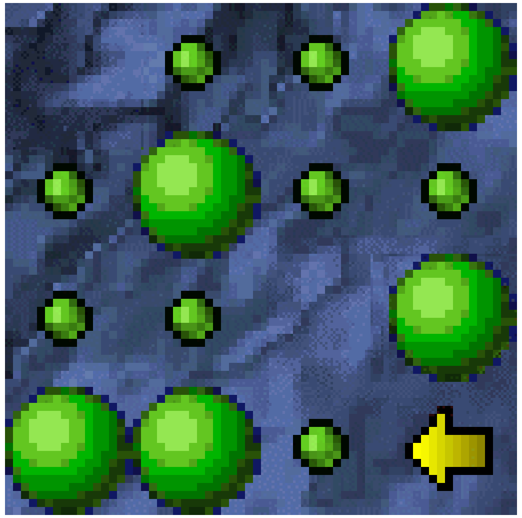
- *Blit* the mask with AND
  - The sprite's intended location becomes black





- *Blit* the sprite with OR
  - Colors remain
  - Black areas do not





Problems with the Bit BLT

- Great at the time – What about today?
  - Transparency, tinting, lighting? Nope
  - Scaling or rotating? Nope
  - Old games used pre-rotated sprites

- The Bit BLT only got us so far

3Dfx – Mid 90's revolution

- 3D-specific graphics cards (Voodoo)
- Revolution: True hardware-accelerated 3D
  - Most other cards obsolete
  - Almost immediately
- Also introduced Scan Line Interleave (SLI)
  - 2 video cards connected together rendering 1 screen

Quake – A year apart

What happened to 3Dfx?

- Eventually outpaced by competitors

- Acquired by Nvidia
  - Mostly for IP
  - Nvidia still uses SLI technology

Hardware accelerated 3D

- Hardware specifically design for:
  - Triangle rasterization
  - Depth-buffering
  - Blending

- If the hardware can do it
  - It's less the CPU needs to do

What can a modern GPU do?

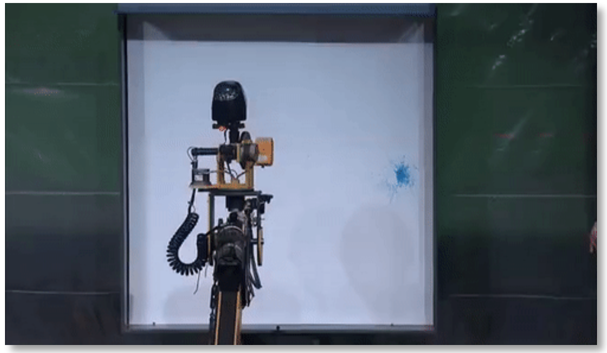
- **Rasterization** – Turning triangles into pixels
- **Texture sampling** – Interpolating image colors
- **Hardware blending** – Transparency
- **Depth buffering** – Basic surface occlusion
- **Shader execution** – Run our code in parallel

GPU parallelism

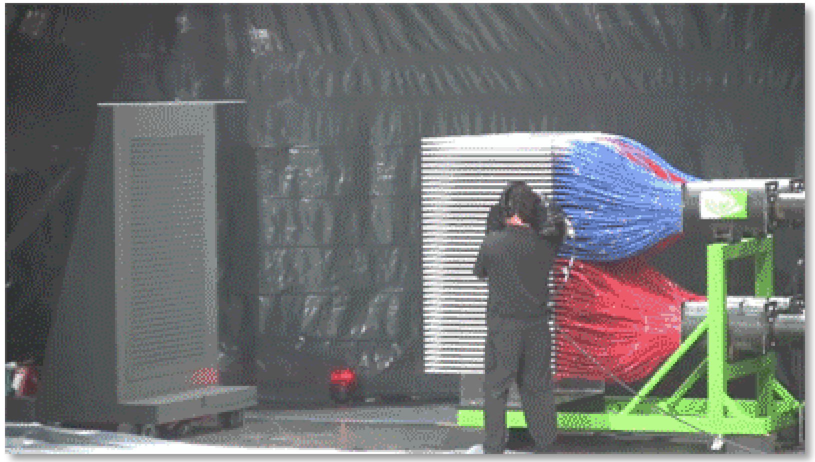
- GPUs
  - Process large amounts of data
  - In parallel
- Vertices – Independent of one another
- Pixels – Independent of one another

Mythbusters CPU/GPU robots

- Built two robots that paint pictures



Like a CPU



Like a GPU

- Full video: http://www.youtube.com/watch?v=ZrJeYFxpUyQ

Some stats

- Number of cores:
  - Intel Core i7 4960X : 6
  - GeForce GTX TITAN: 2688
- Floating Point Operations per Second (FLOPS)
  - Intel Core i7 4960X: 156.5 GigaFLOPS
  - GeForce GTX TITAN: 1.3 TeraFLOPS
- GeForce GTX TITAN pixel fillrate:
  - 187.5 billion pixels per second

Are GPUs just better?

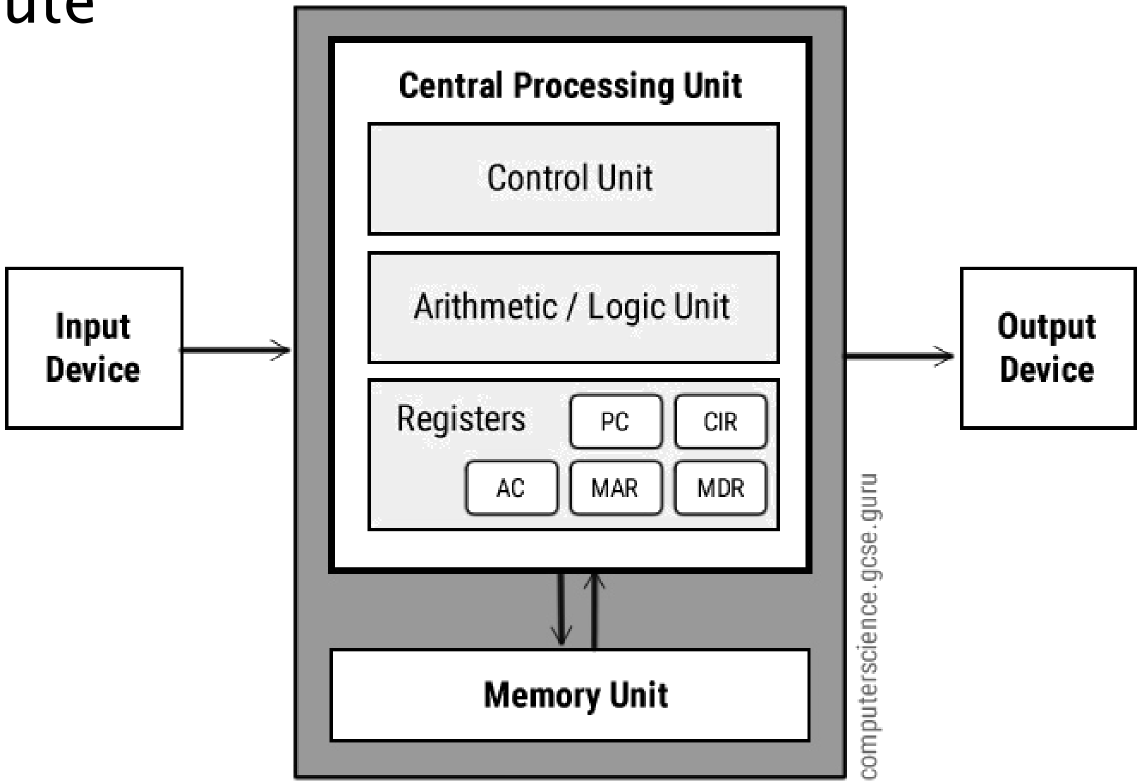
- Those numbers seem impressive

- But the story isn't that simple (of course)

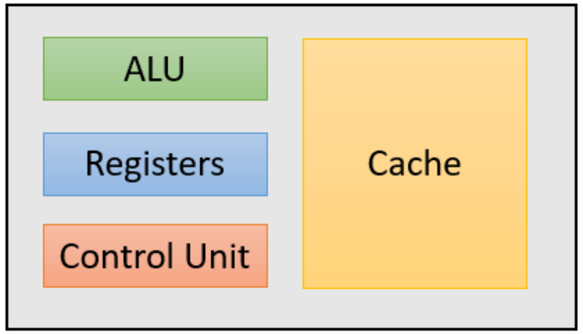
- GPU cores not exactly the same as CPU cores
  - Divided into groups of cores called *stream multiprocessors (SM)*
  - Each SM runs SIMD

CPU example

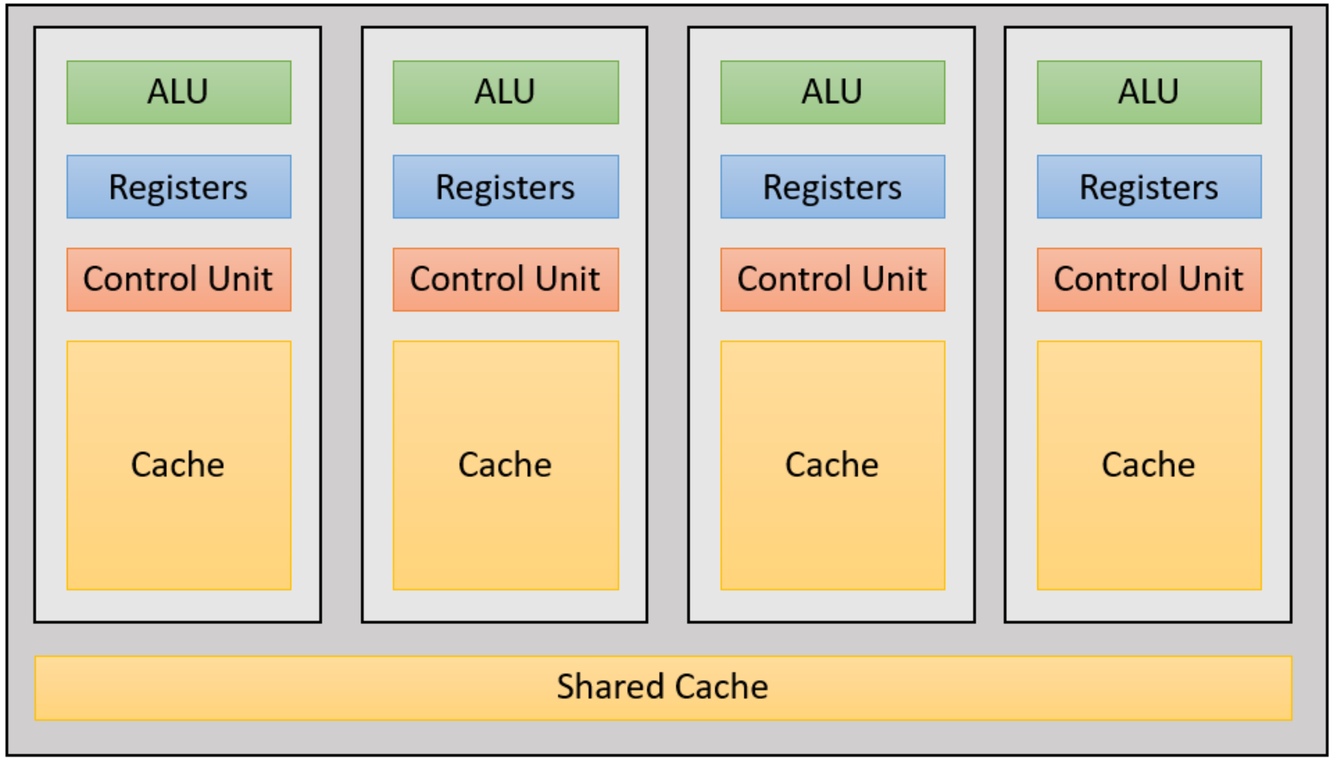
- CPUs all have several common features:
  - **Arithmetic Logic Unit** (ALU)
  - **Registers** to hold data for ALU operations
  - **Control Unit** to fetch, decode & execute actual instructions
- All of these together make up a single "core"



Single vs. multi-core CPUs



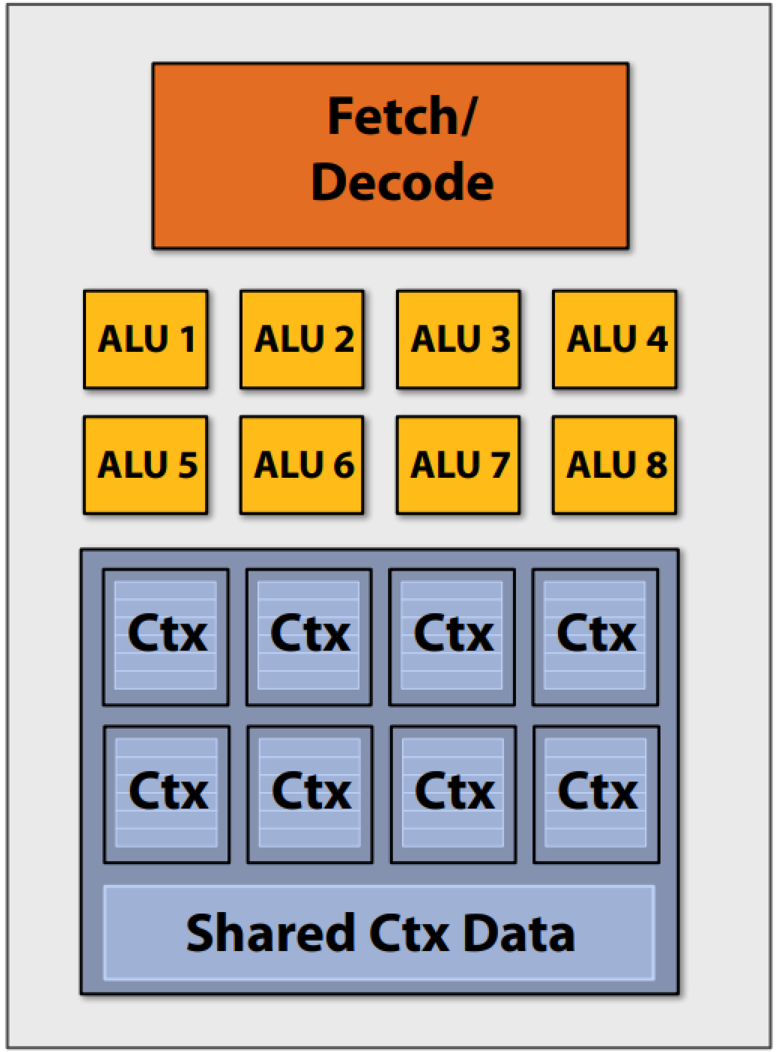
Single-Core CPU



Quad-Core CPU

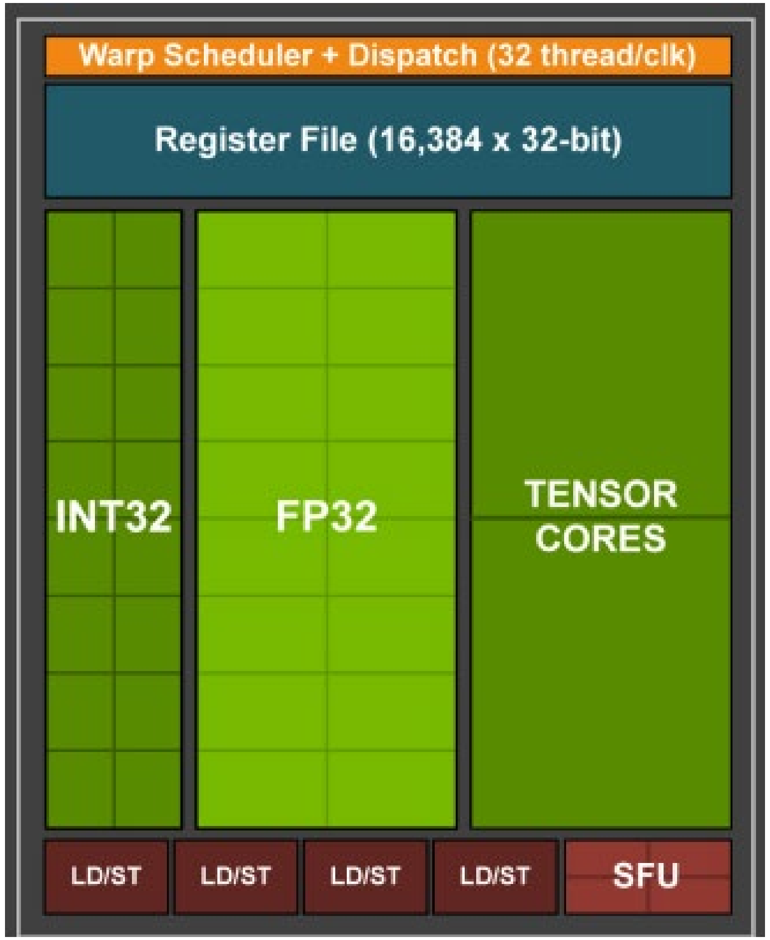
GPU cores

- GPU cores are grouped differently
- Multiple ALUs exist together
  - But with a single Control Unit
  - All execute the same instruction!
- They each have their own execution context (registers)



Actual GPU architecture from NVidia

- This is a *Warp*
- Each green block is a "core" (ALU)
  - 16 for integer math
  - 16 for floating point math
  - Two for advanced uses
- Register File contains data for ALUs
- Warp Scheduler is the Control Unit

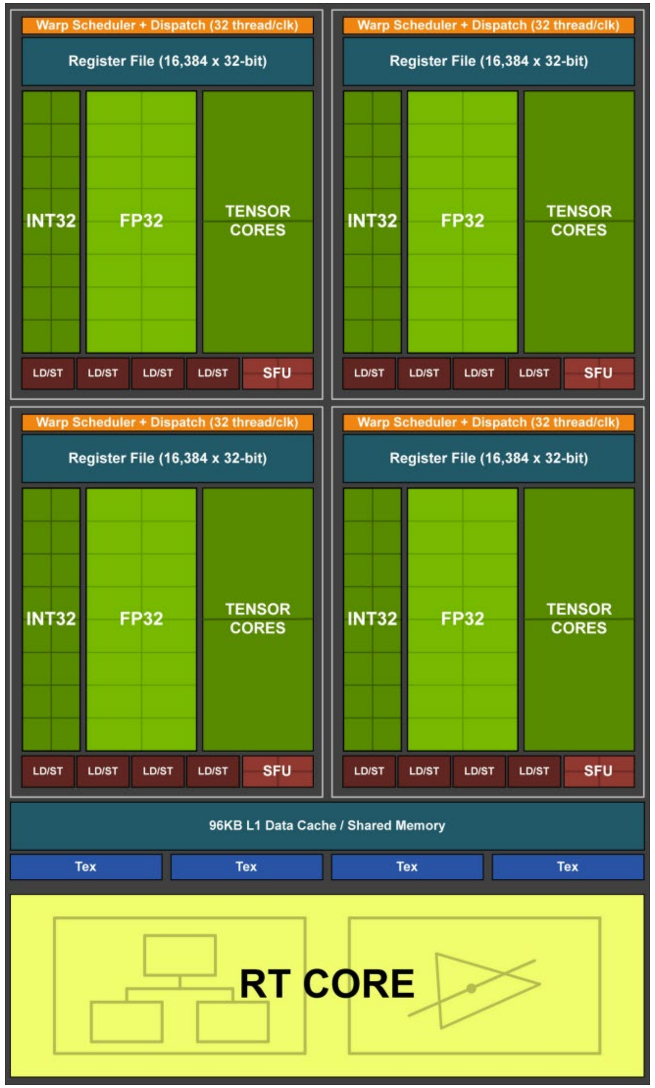


Multiple Warps make up one SM

- Streaming Multiprocessor (SM)

- Combines several warps
  - Also contains shared memory
  - Texture units
  - Raytracing core (in modern GPUs)

- That's a lot of cores!
  - Well…



The whole chip

- Far more cores than a CPU!
- However, most cores must be performing the same operation at any given time
- Since they share control units

SIMD paradigm

- Single Instruction, Multiple Data
  - Multiple cores execute same instructions
  - Each acting on different data
- Maps extremely well to shaders
  - Same code is run for each pixel/vertex
  - Breaks down with branching though!

The rabbit hole goes deep

- This is not explicitly a hardware course
  - You don’t need to know this stuff for class!
- However…
  - Understanding how a GPU works can help us write better engines
  - If we know its limitations, we can work around them
- Some articles if you do want to dig deeper:
  - Understanding the Parallelism of GPUs
  - How a GPU Works slides from CMU

GPU vs. CPU takeaway

- GPUs better at:
  - Parallel number crunching
  - Running same code for multiple pieces of data

- CPUs better at:
  - General tasks
  - Branching (conditional statements)
  - Doing exactly one thing at a time

Proper GPU usage

- GPUs can process lots of “stuff” in parallel
  - Vertices, pixels, compute threads, etc.
  - If you give it lots of “stuff” at once
- Optimized for specific use cases
  - Process lots of “stuff” at once? Fast!
  - Process each “thing” separately? Slow!

More graphics history

- If you like this stuff, here's more!

- Fantastic look at the history of the GPU:
  - http://www.techspot.com/article/650-history-of-the-gpu/

- "How 'Old School' Graphics Worked"
  - Part 1: https://youtu.be/Tfh0ytz8S0k
  - Part 2: https://youtu.be/_rsycfDliZU