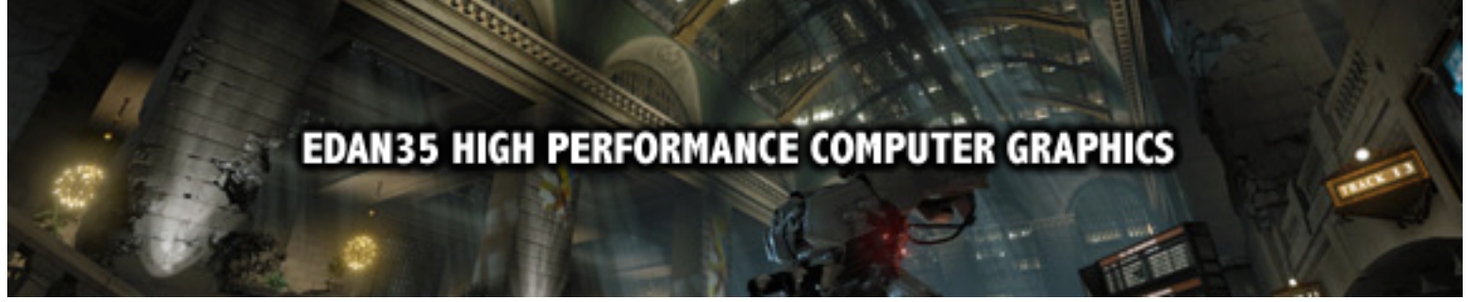


GPU Architecture

Michael Doggett
Department of Computer Science
Lund university

2024

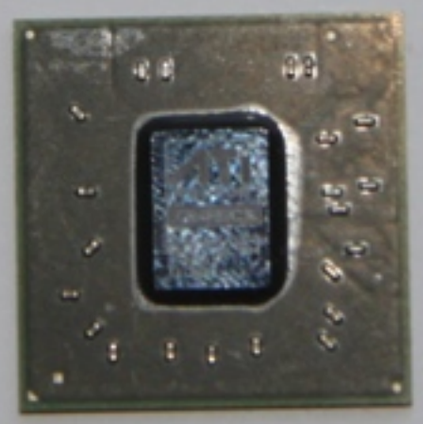
GPUs from my time at ATI/AMD

2001-2009

R200

Xbox360 GPU

R630



R610

R770

Let’s start at the beginning …

Graphics Hardware before GPUs

- 1970s - 1980s
- Very Expensive
  - Real-time Rendering needs a lot of compute power
    - Military, large industrial, flight simulators
      - Evans and Sutherland founded 1968
- Custom built systems, multiple boards
- Custom and off-the-shelf ASICs
  - Today's GPU has similar structure, but all on one chip

Pixel-Planes/Flow

- Research Systems
  - University of North Carolina
    - Pixel-Planes 5 [Fuchs89]
    - PixelFlow [Molnar92]
      - Programmable Shading [Lastra95]
      - Final version built and sold by HP ’97

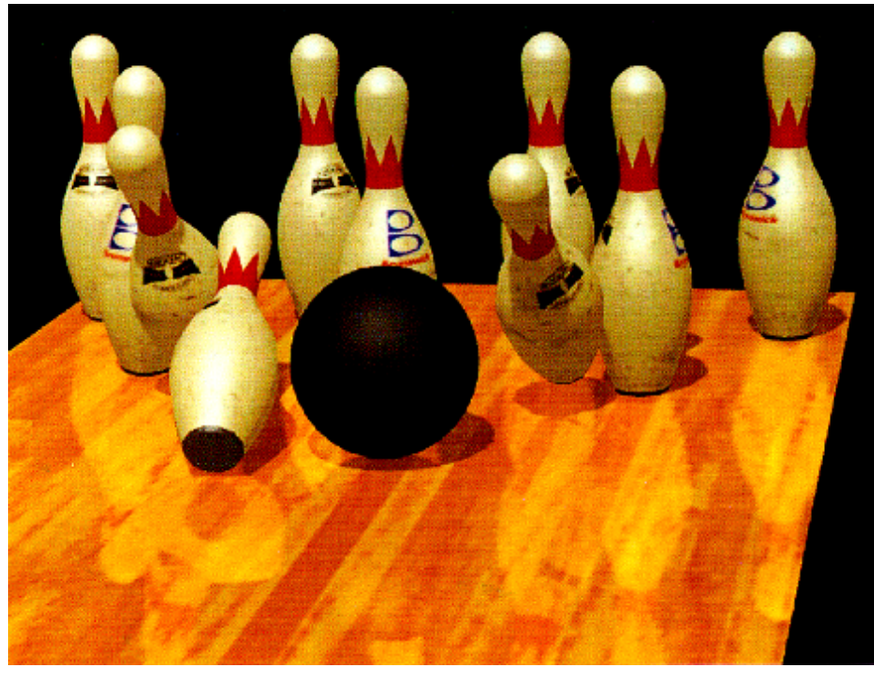
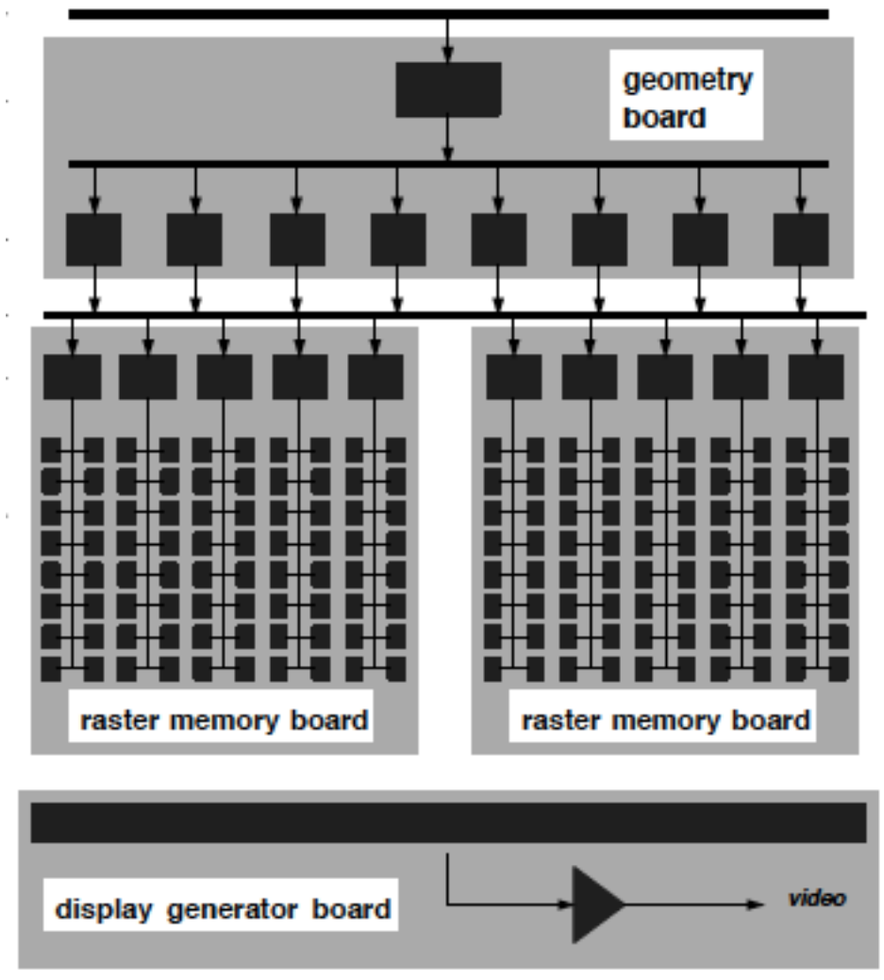
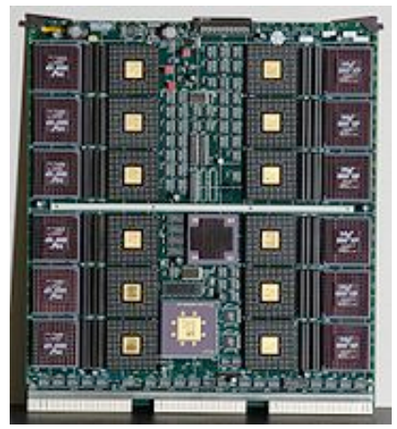


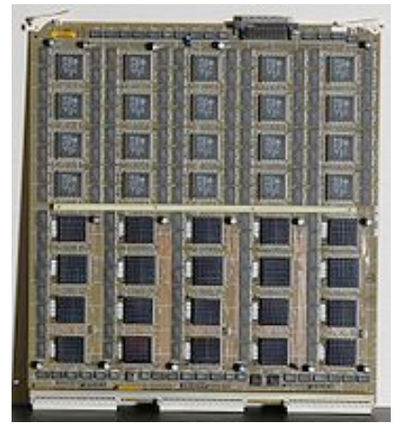
Image courtesy [Lastra95]

Silicon Graphics

- Onyx
  - MIPS R4400 RISC processors
  - Reality Engine ’92[Akeley93]
    - Geometry Engine
      - 50MHz Intel i860XP,
    - Raster Memory, 1-4
  - OpenGL, no shaders



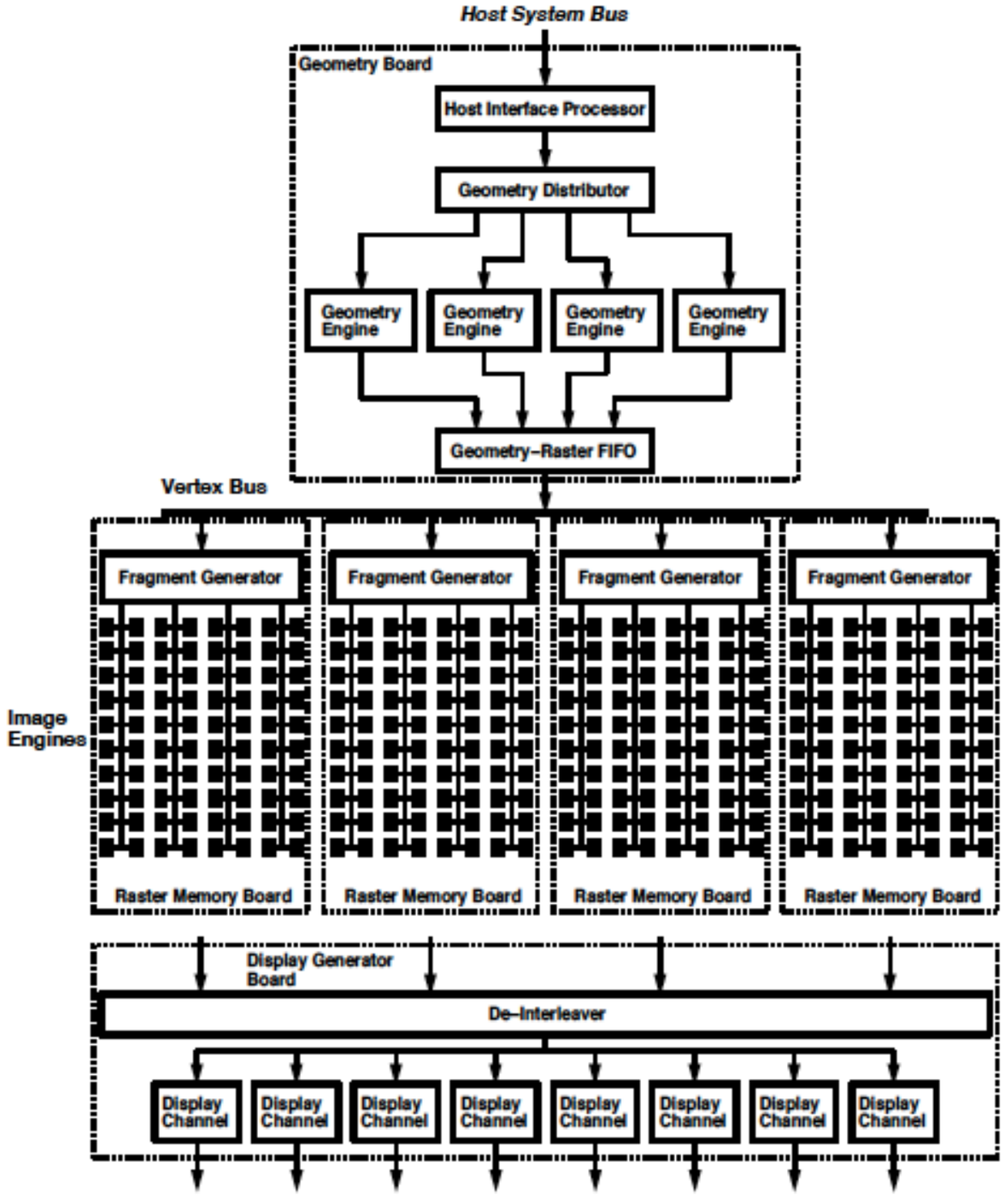




Images courtesy [Akeley93]

Silicon Graphics

- Onyx 10000
  - Infinite Reality [Montrym97]

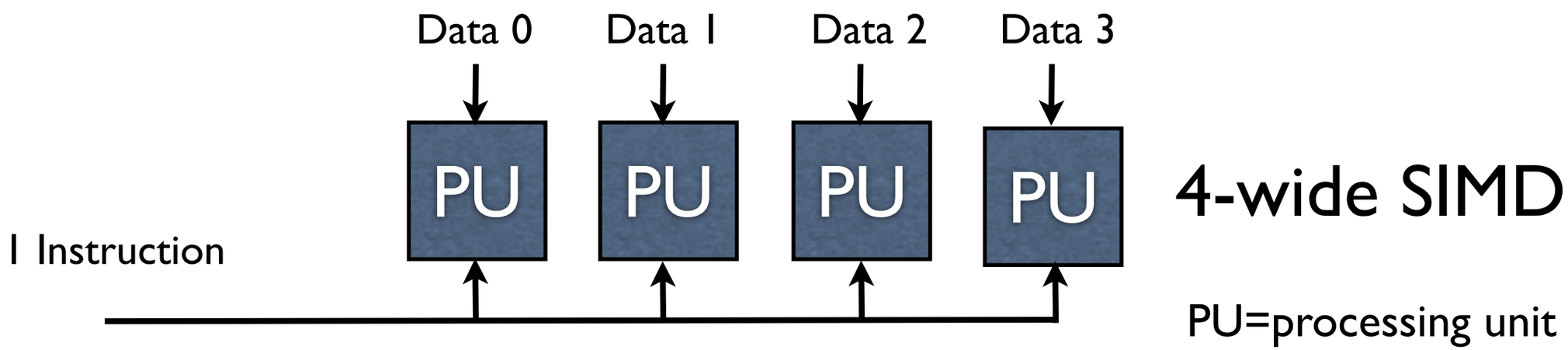


Images courtesy [Montrym97]

GPUs

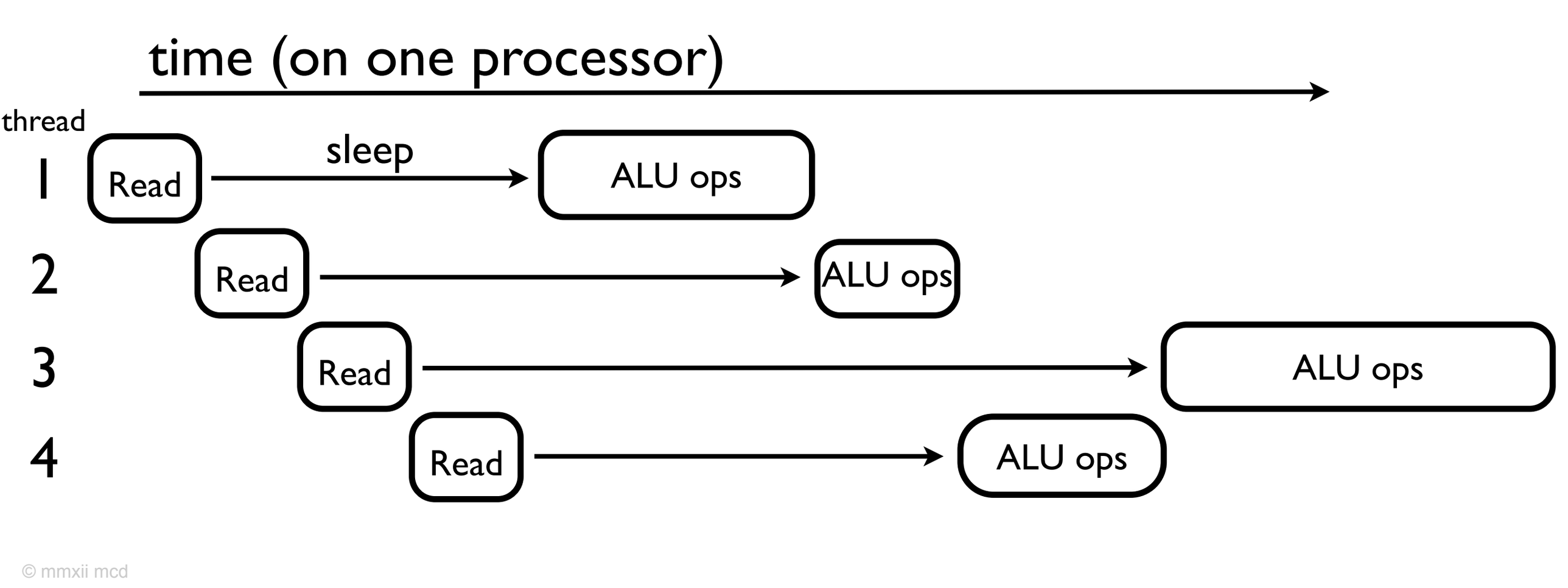
Graphics Processing Unit GPU

- How to turn millions of triangles into pixels in 1/60 of a second?
- **Parallelism** !!
  1. Hardware Pipelining (Graphics pipeline)
  2. Single Instruction Multiple Data (SIMD)
  3. Multiple Instruction Multiple Data (MIMD) multiple cores
- Memory Bandwidth reductions (covered previously)



Fast Thread Switching enables GPU arithmetic intensity

- 1000s of small simple threads with context on chip
- GPUs hide large memory request latency by switching between threads



GPU design parameters

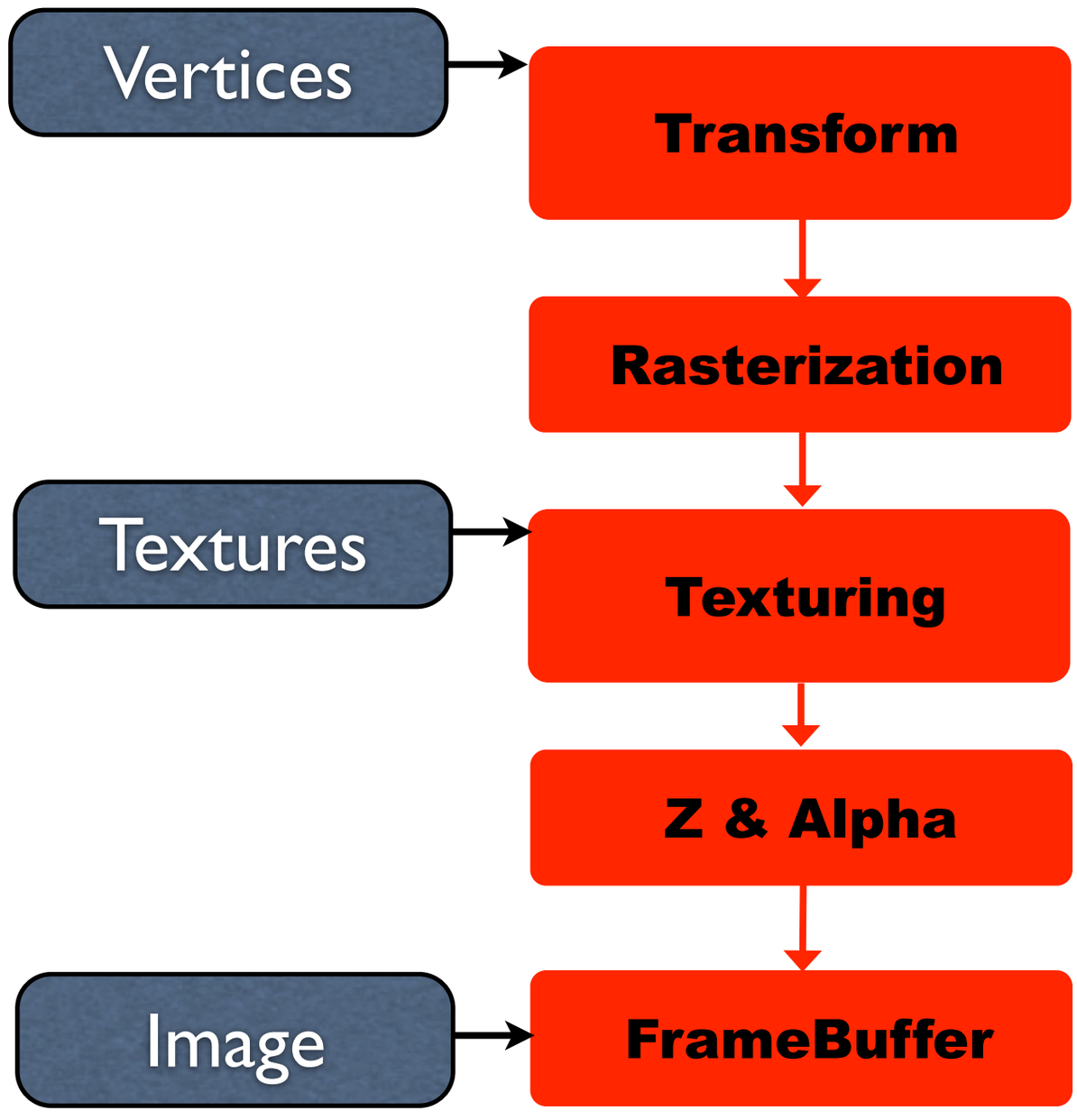
- Competition
  - 2 strong competitors NVIDIA and AMD (ATI)
  - Performance/Dollar
- Moore's law
  - Number of transistors on a chip doubles every two years
- APIs (OpenGL and DirectX) hide details
  - Hide architectural changes
  - Provide backward compatibility (mostly)

Before programmable GPUs

- ATI
  - Founded 1985
  - 2D - Mach series
  - 3D - Rage series ’95-’99
- nVidia
  - Founded 1993
  - Riva, Riva TNT series ’95-’99
- up to DirectX 6

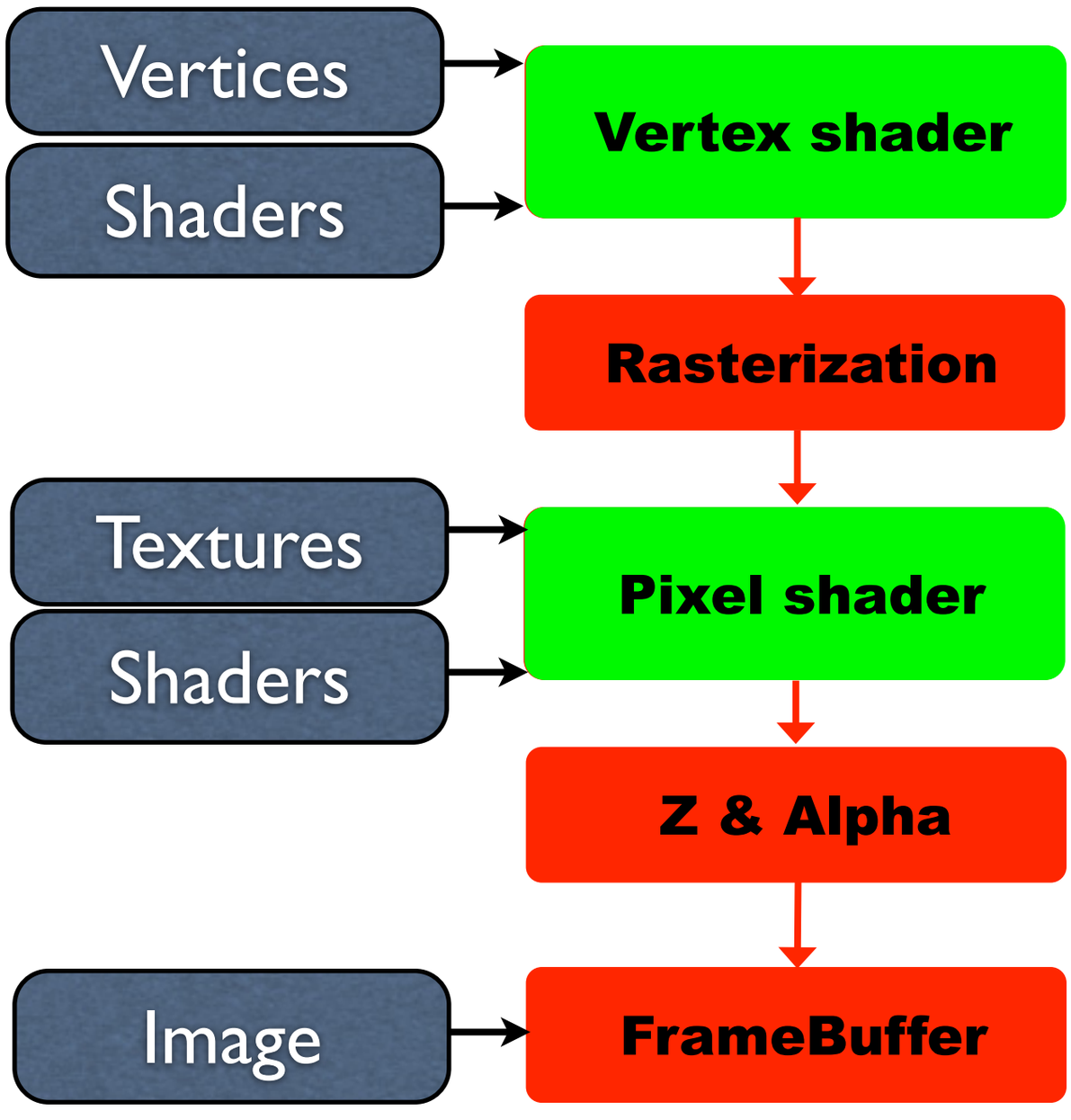
Hardware Transform, Clipping and Lighting

- NV10 ’99
  - Nvidia GeForce 256
  - The first GPU
- R100 ’00
  - ATI Radeon 7500
- Transformation requires **32bit float** 4x4 matrix multiplication
- Texturing for 4 component (RGBA) 8bit pixels
  - Low precision math
- DirectX 7



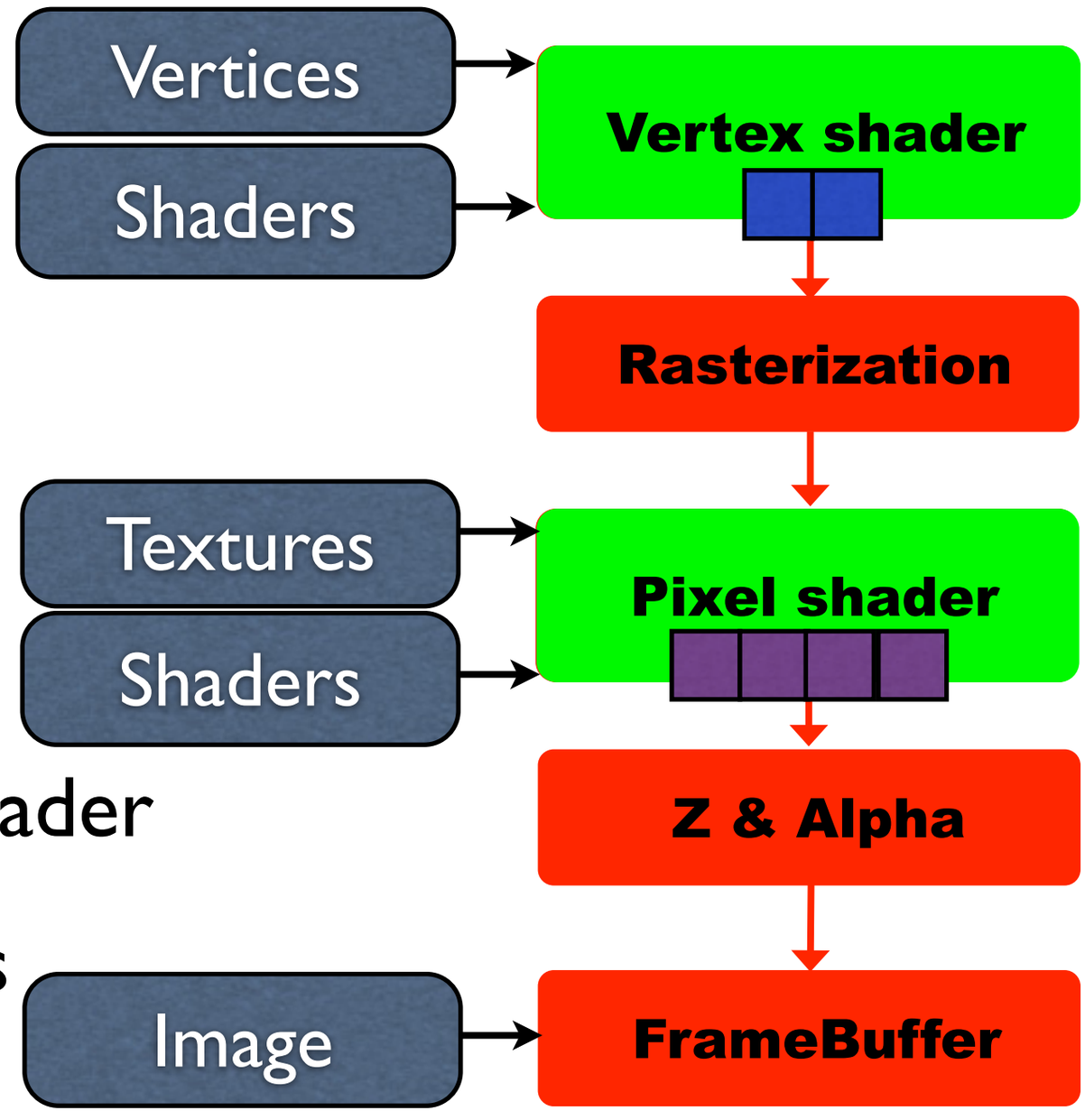
Programmable GPUs
1st Generation

- Shaders run programs that do what fixed function hardware did
- GPU programs (shaders) have simpler requirements than CPU programs



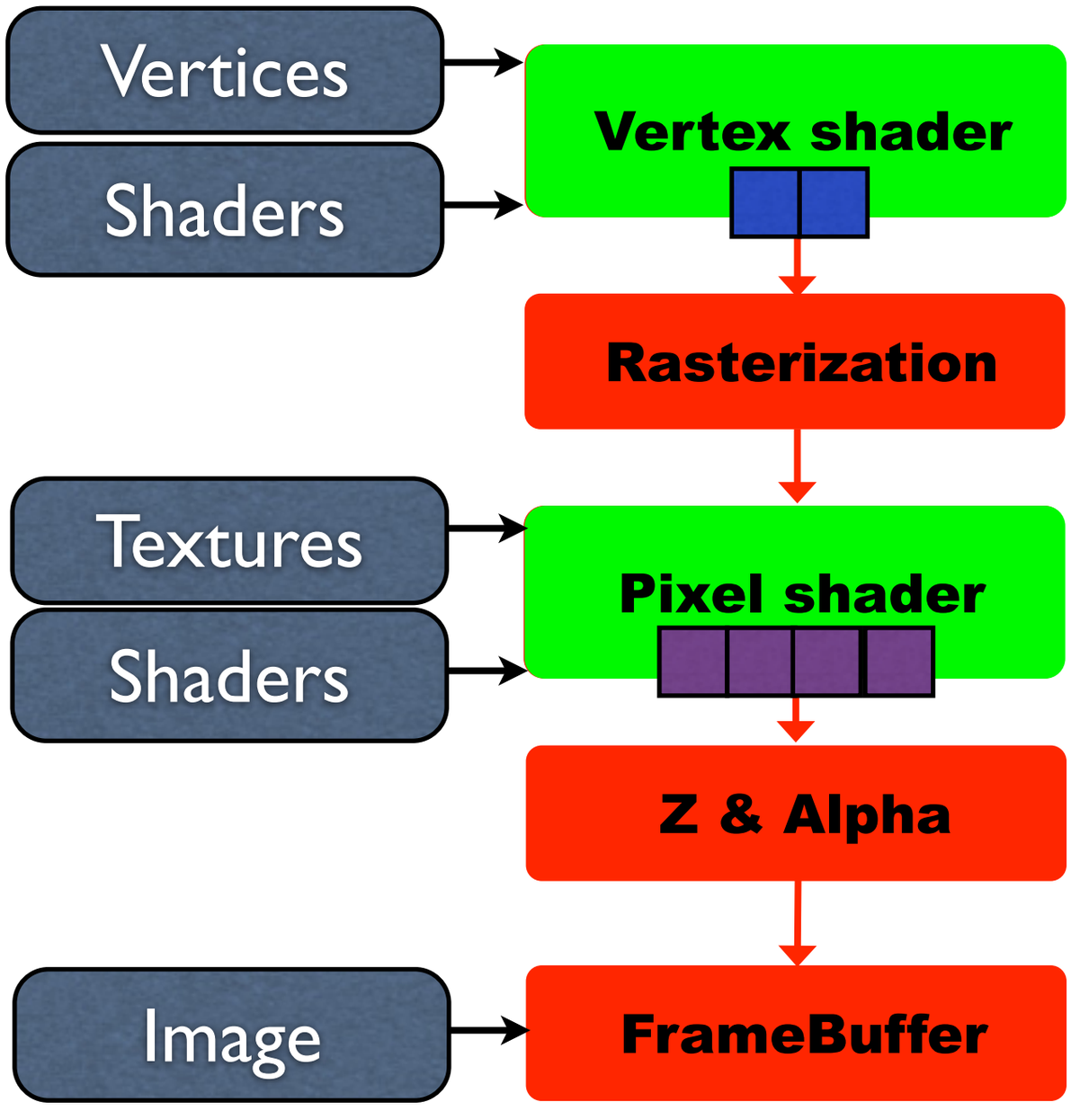
Programmable GPUs
1st Generation

- NV20 '01
  - Nvidia GeForce 3 [Lindholm01]
- R200 '01
  - ATI Radeon 8500
  - 2-wide Vertex shader
  - 4-wide SIMD Pixel shader
    - Fixed point ~16bits



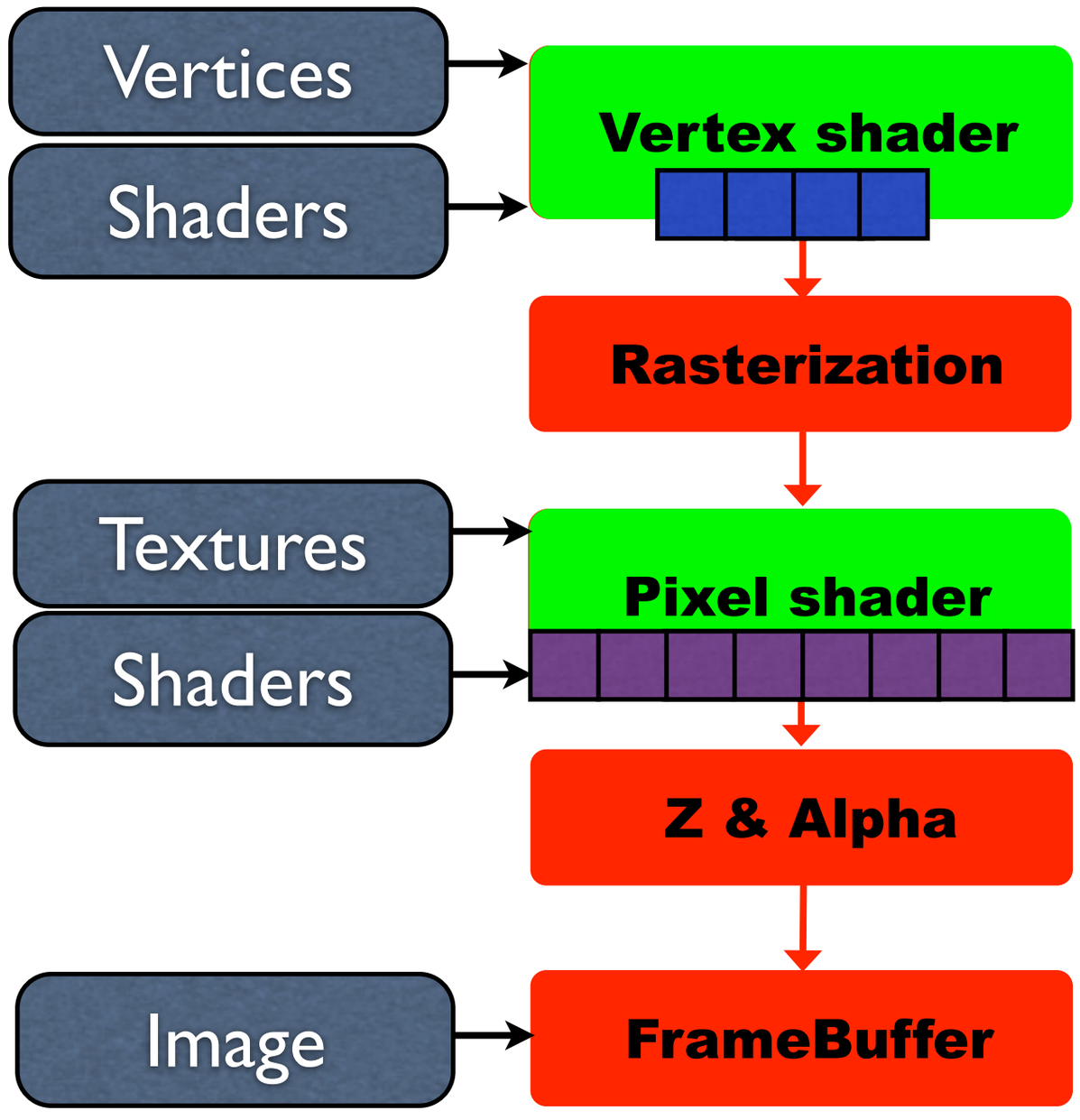
Programmable GPUs
1st Generation

- DirectX8 Pixel Shaders
- Multiple versions
  - 1.1, 1.3, 1.4
- 13-22 instructions
- assembler language



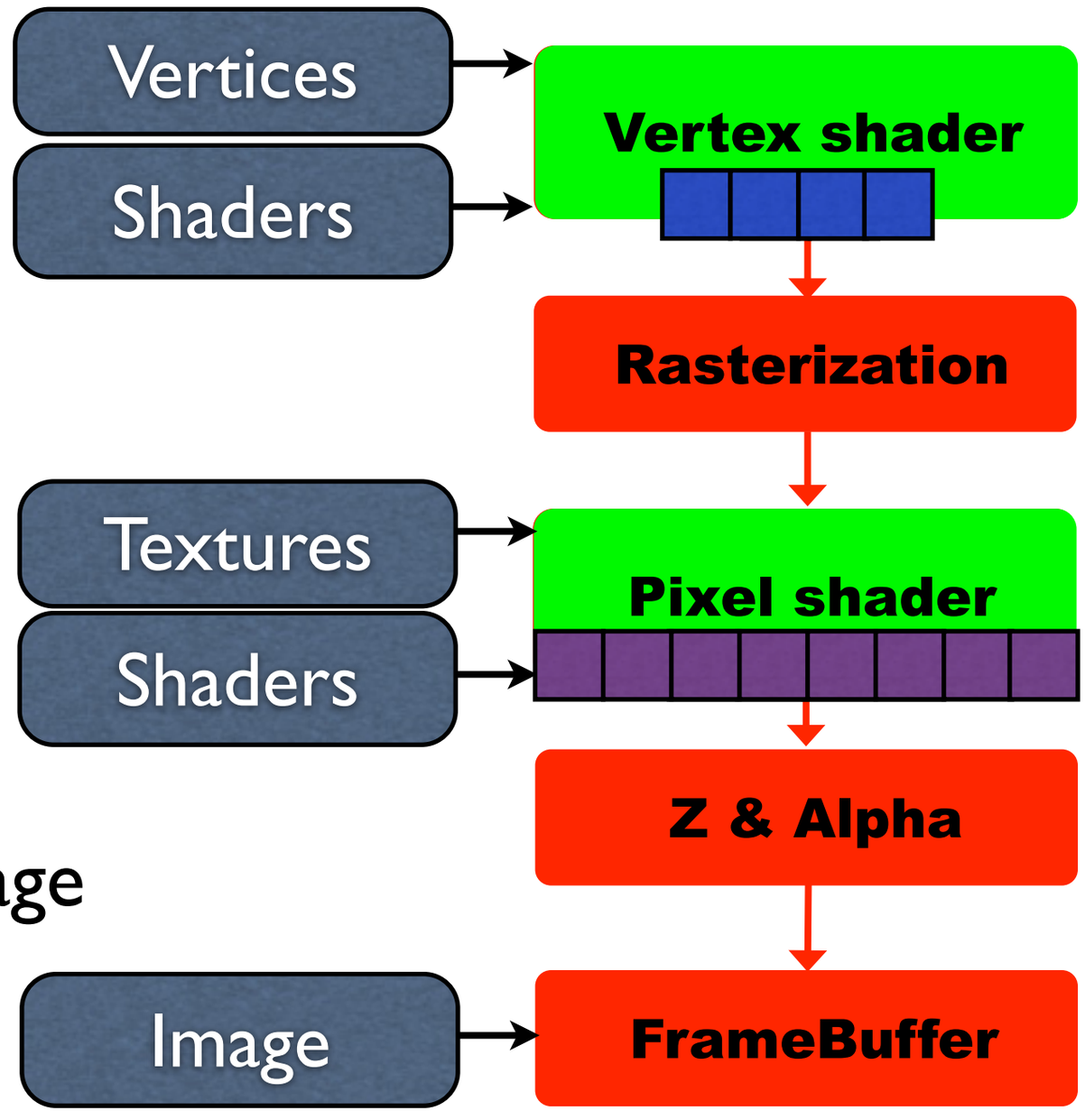
Programmable GPUs
2nd Generation

- R300 '02
  - ATI Radeon 9700
  - **256bit memory bus**
  - 4-wide Vertex shader
  - 8-wide SIMD Pixel shader
    - **24bit float**



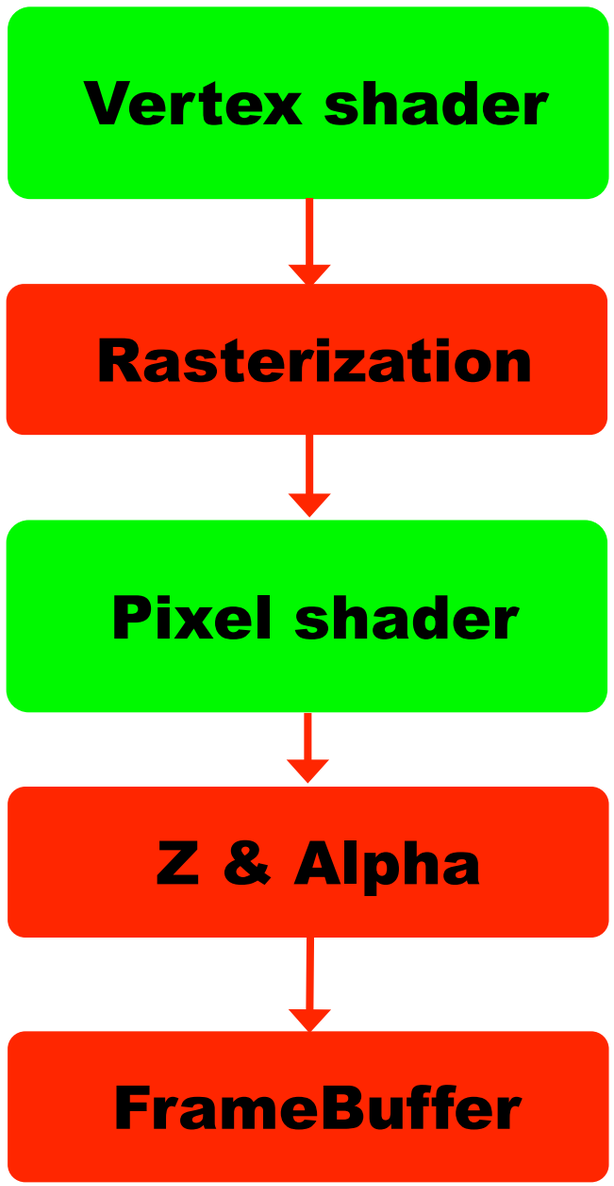
Programmable GPUs
2nd Generation

- NV30 ’03
  - Nvidia GeForce FX 5800
    - **16** and 32 bit float
  - **Cg (C for graphics)**
- NV40 ’04 GeForce 6800 [Montrym05]
- DirectX 9
  - High Level Shading Language (HLSL)

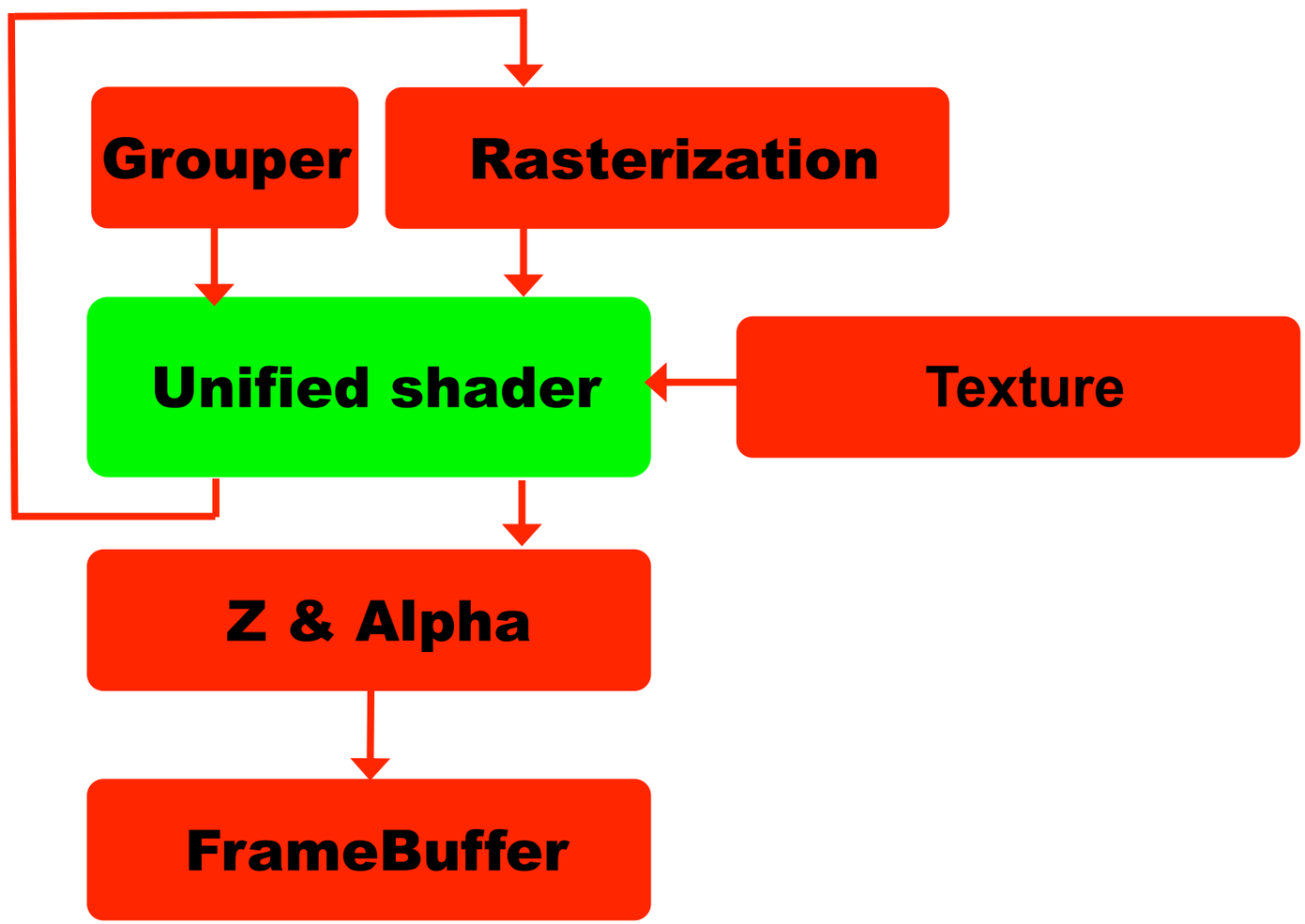


Unified Shaders

Unified Shader

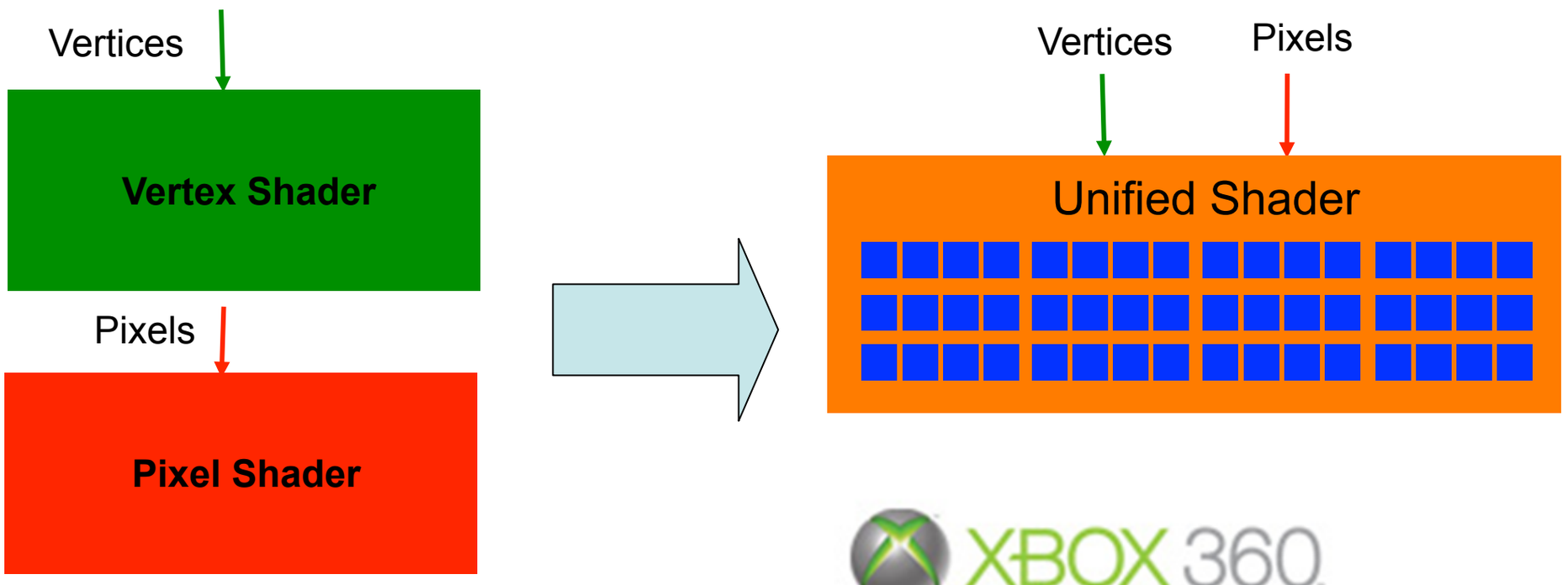


Unified Shader



Unified Shader

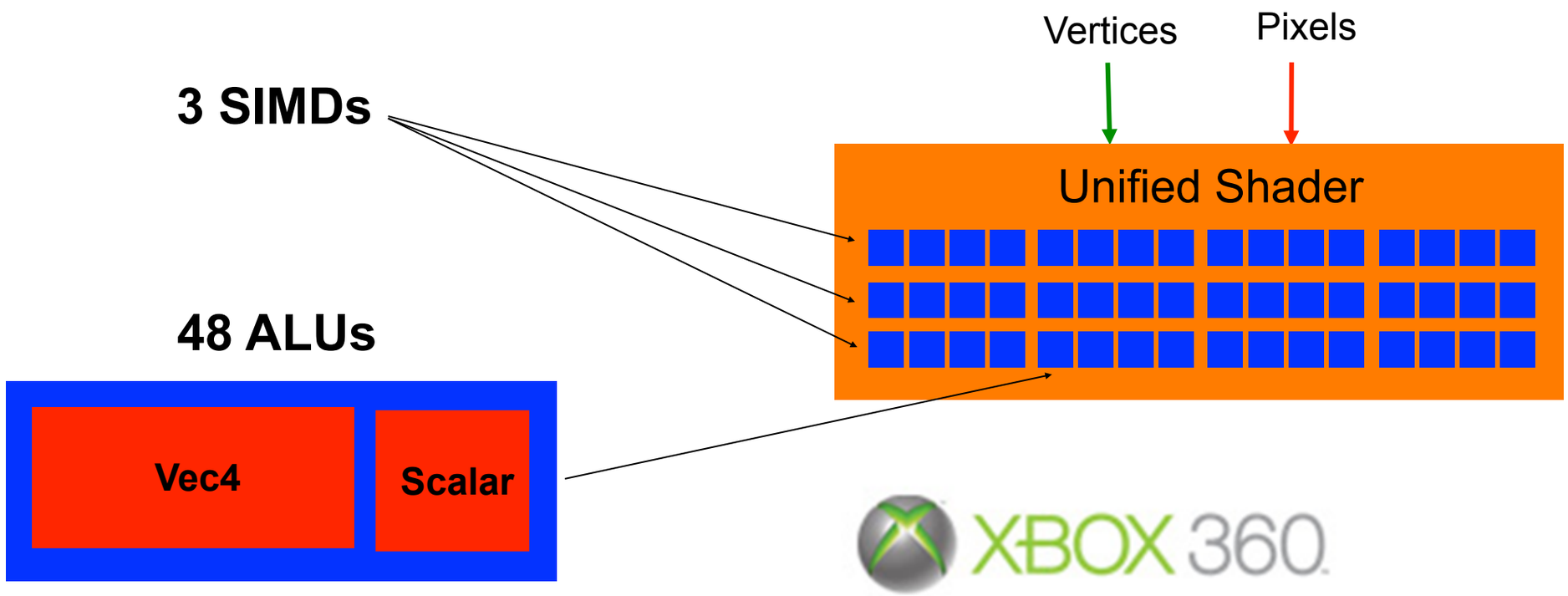
- A revolutionary step in Graphics Hardware
- One hardware design that performs both Vertex and Pixel shaders
- Vertex processing power



XBOX 360

Unified Shader

- GPU based vertex and pixel load balancing
  - Better vertex and pixel resource usage
- Union of features
  - E.g. Control flow, indexable constant, …
  - All 32bit floating point
- DX9 Shader Model 3.0+
  - Shader write at address instruction



XBOX 360

XBox360 GPU architecture - ’05

GPU

Primitive Setup
Clipper
Rasterizer
Hierarchical Z/S

Index Stream
Generator
Tessellator

Unified Shader

Texture/Vertex Fetch

Output
Buffer

Memory Export

Vertex Pipeline

Pixel Pipeline

Display Pixels

UNIFIED MEMORY

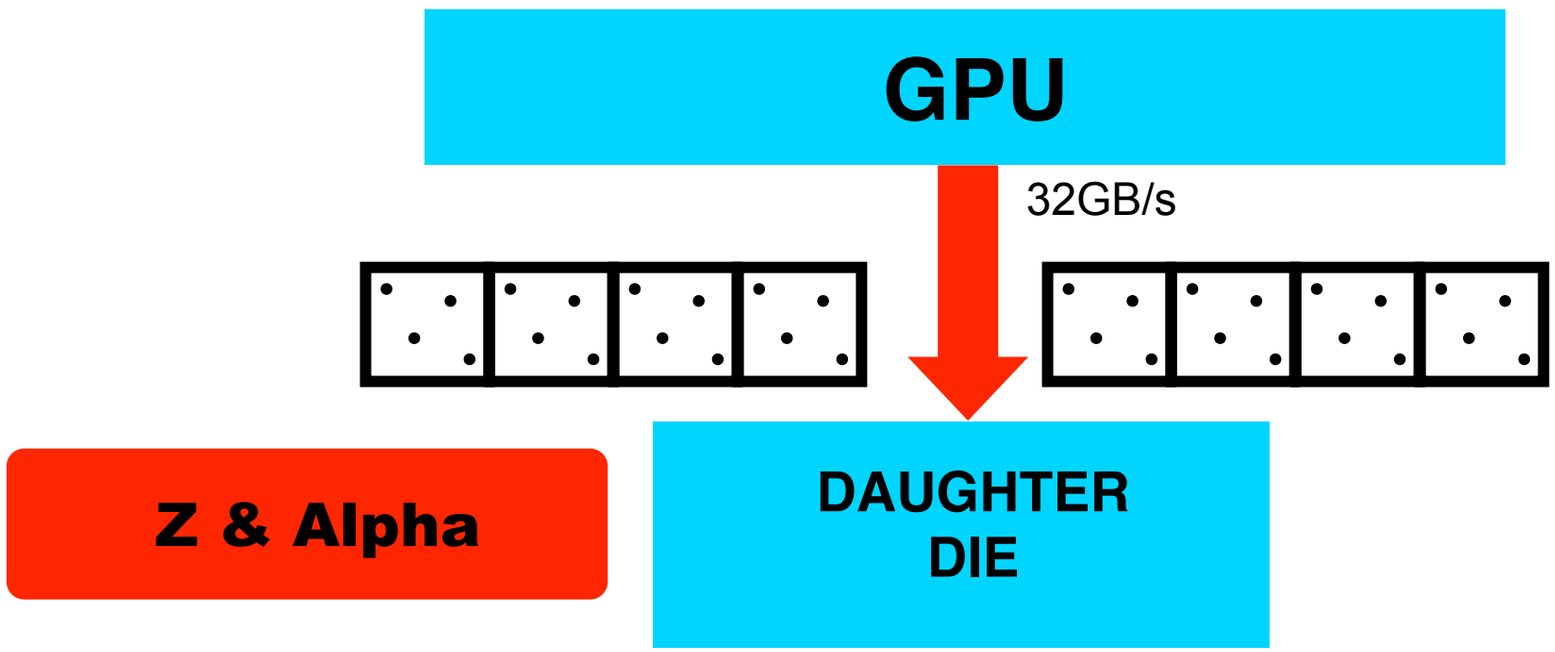
DAUGHTER DIE

SIMD is 16 vector math units wide
- **Shared instr. decode and control**
- **Makes GPUs simpler than CPUs**

XBOX 360

Rendering performance

- 2 Dies (1 GPU, 1 Logic Embedded DRAM)
- GPU to Daughter Die high speed interface
  - 8 pixels/clk
    - 32BPP color, 4 samples/pixel Z - Lossless compression
  - 16 pixels/clk – Double Z (No color)
    - 4 samples/pixel Z - Lossless compression

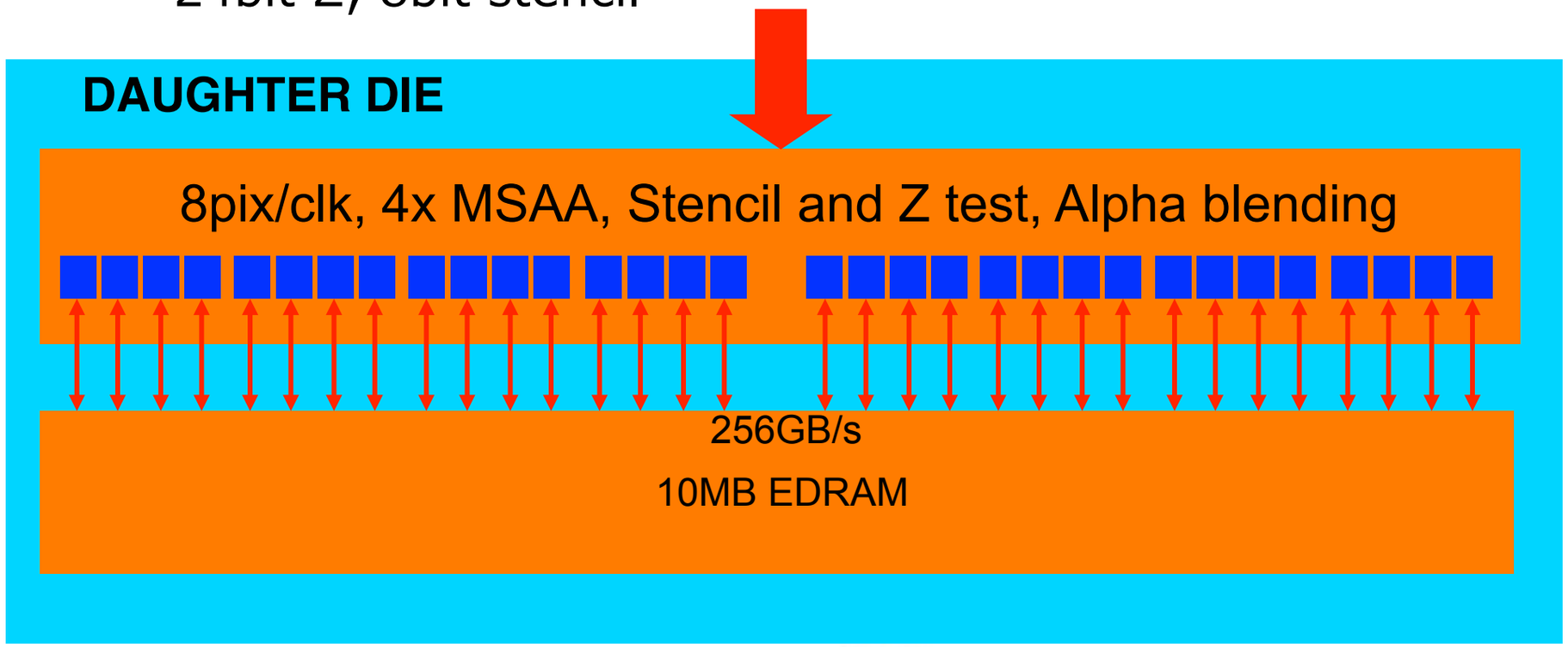


XBOX 360

Rendering performance

Z & Alpha

- Z and Alpha logic to EDRAM interface
  - 256GB/s
  - Color and Z - 32 samples
    - 32bit color, 24bit Z, 8bit stencil
  - Double Z - 64 samples
    - 24bit Z, 8bit stencil

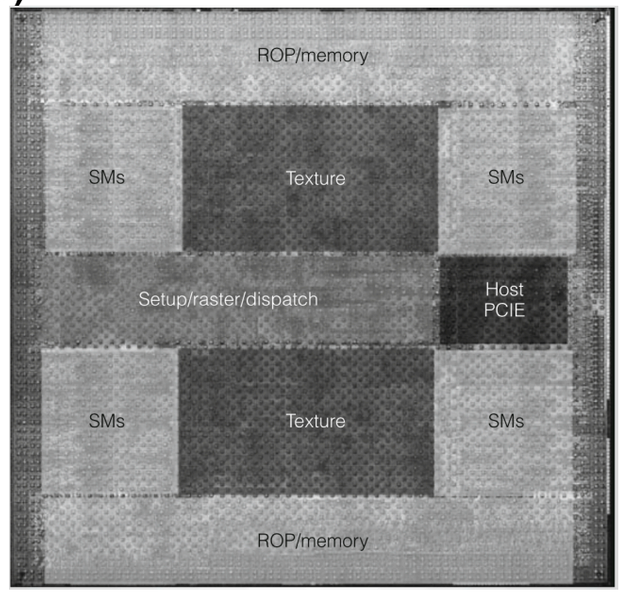


XBOX 360

2013 Xbox360 GTAV

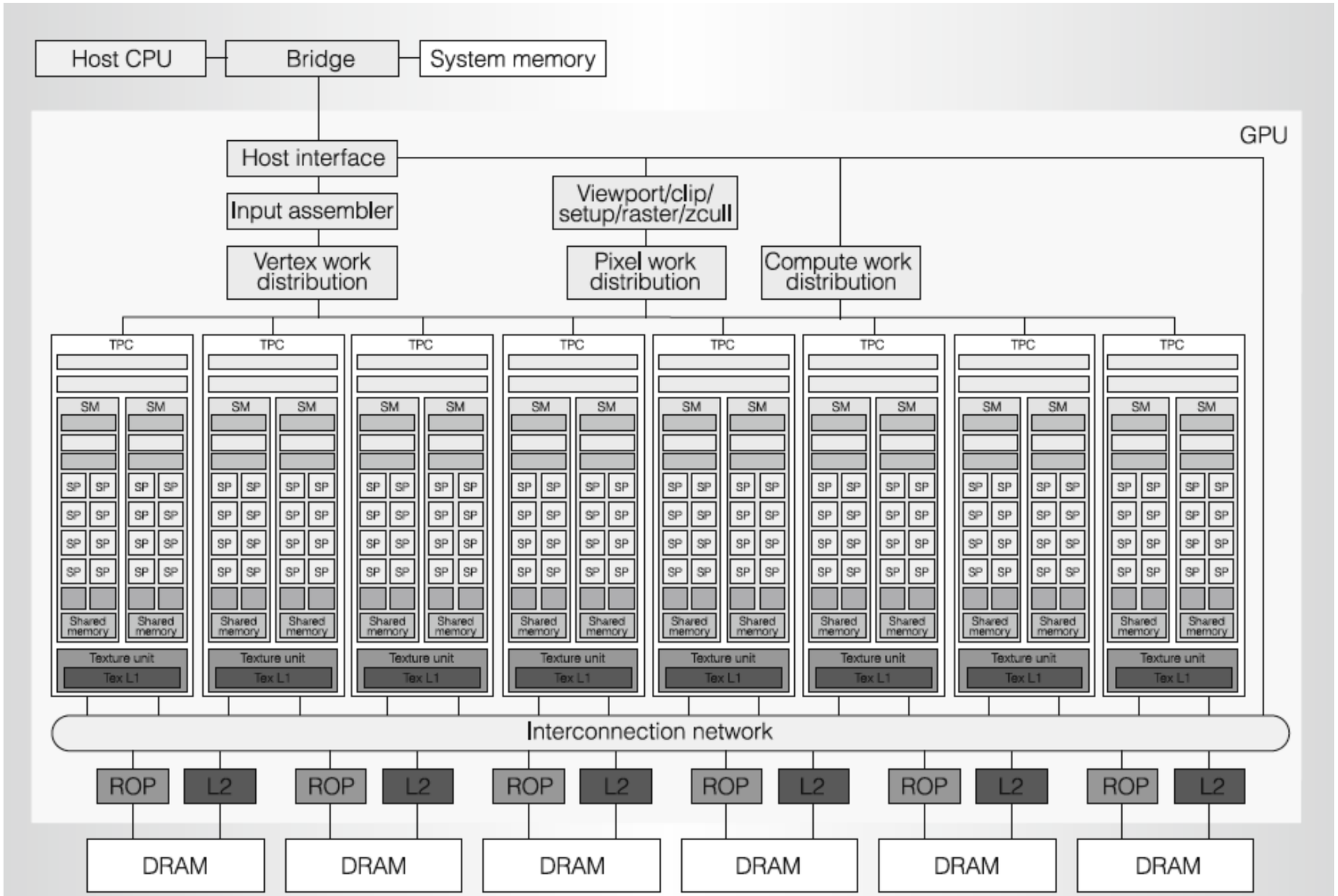
Unified Shaders - PC

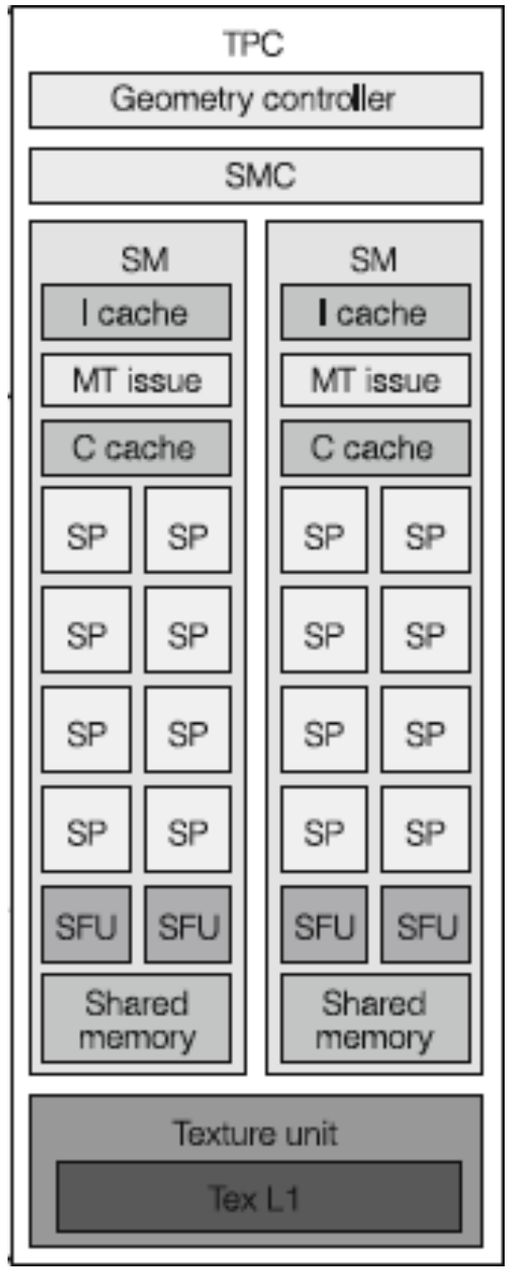
- G80 ’06 (Tesla [Lindholm08])
  - nVidia GeForce 8800
  - 8 Texture Processor Cluster (TPC)
    - 2 Streaming Multiprocessor (MP)
      - 8 Streaming processors (SP), Multiply Add (MAD)
  - 128 SPs @ 1350 MHz
    - 2022 RTX 4090 has 128x SPs (roughly double every 2 years - Moore’s law)
  - Level 2 cache placed close to the individual memories (DRAMs)
- DirectX 10
- **Shader Shared Memory**
  - **CUDA - new compute focused API**



GeForce 8800 Ultra die layout

NVIDIA G80

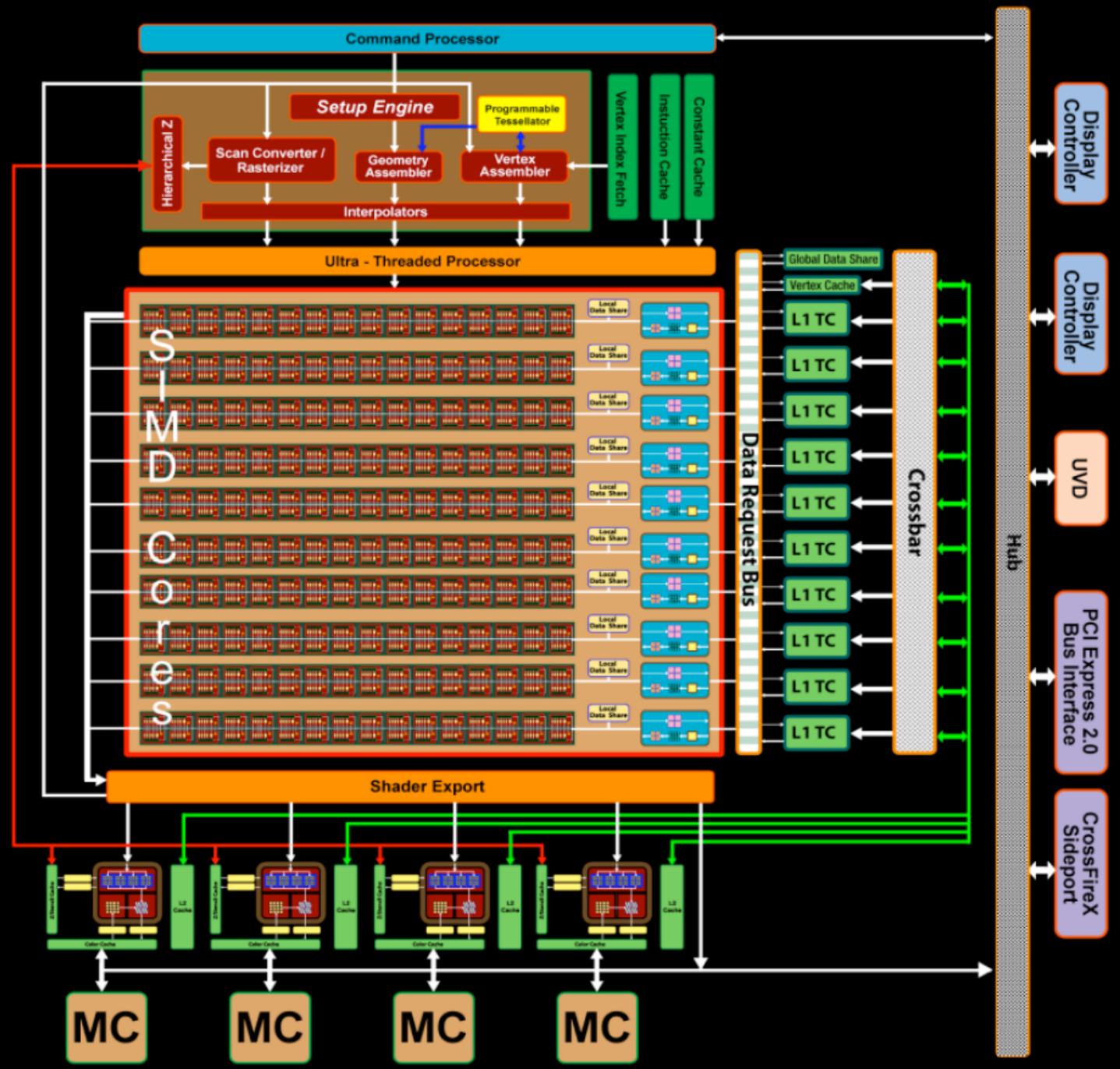




Images courtesy [Lindholm08]

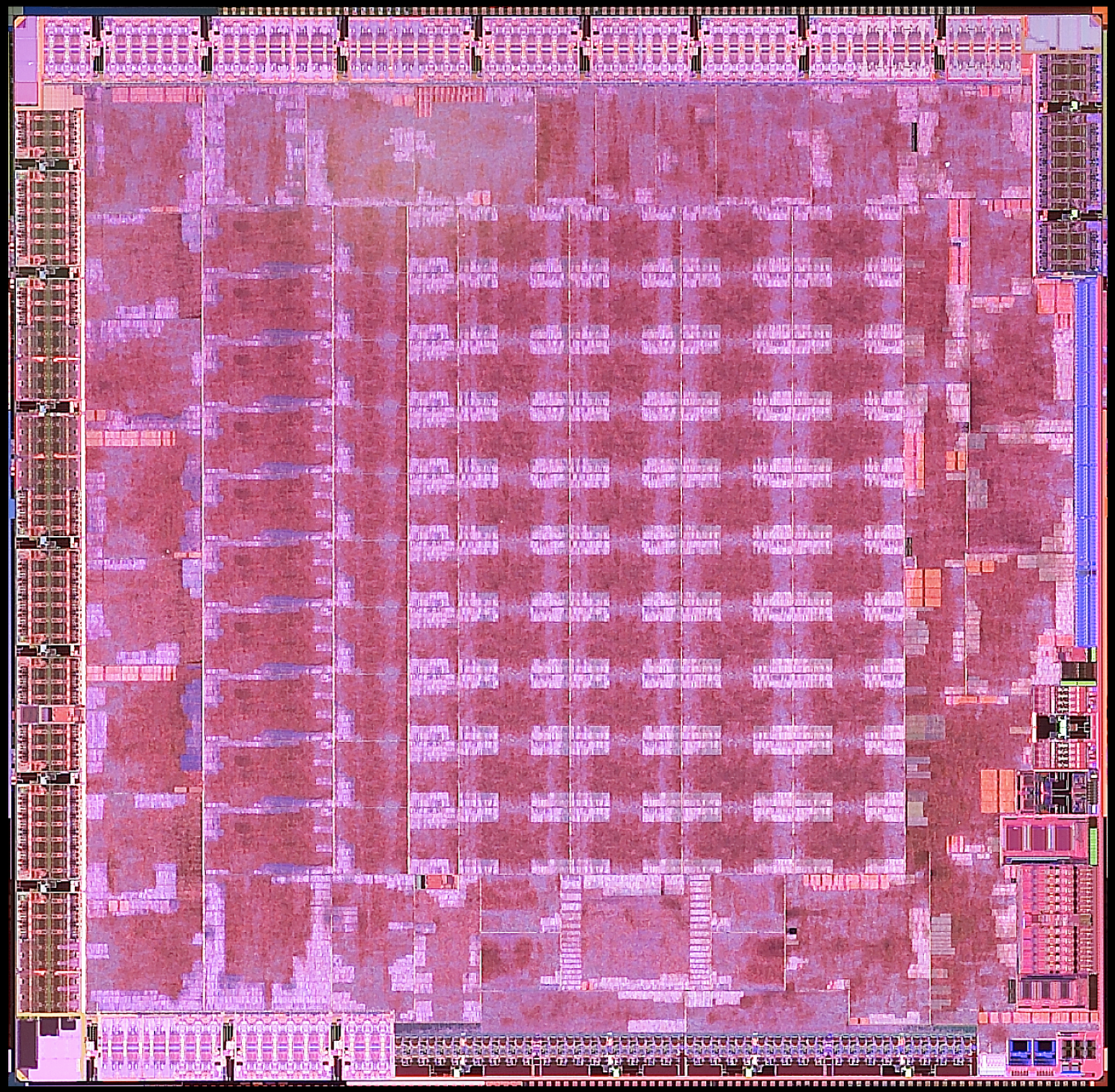
R770 ’08
ATI Radeon 4870

- 10 SIMDs
  - –SIMD = 16 x 5 ALU
- New memory architecture
  - –L2 cache partitioned at MCs
  - –Crossbar to L1s
- **1.2 TeraFLOPS**



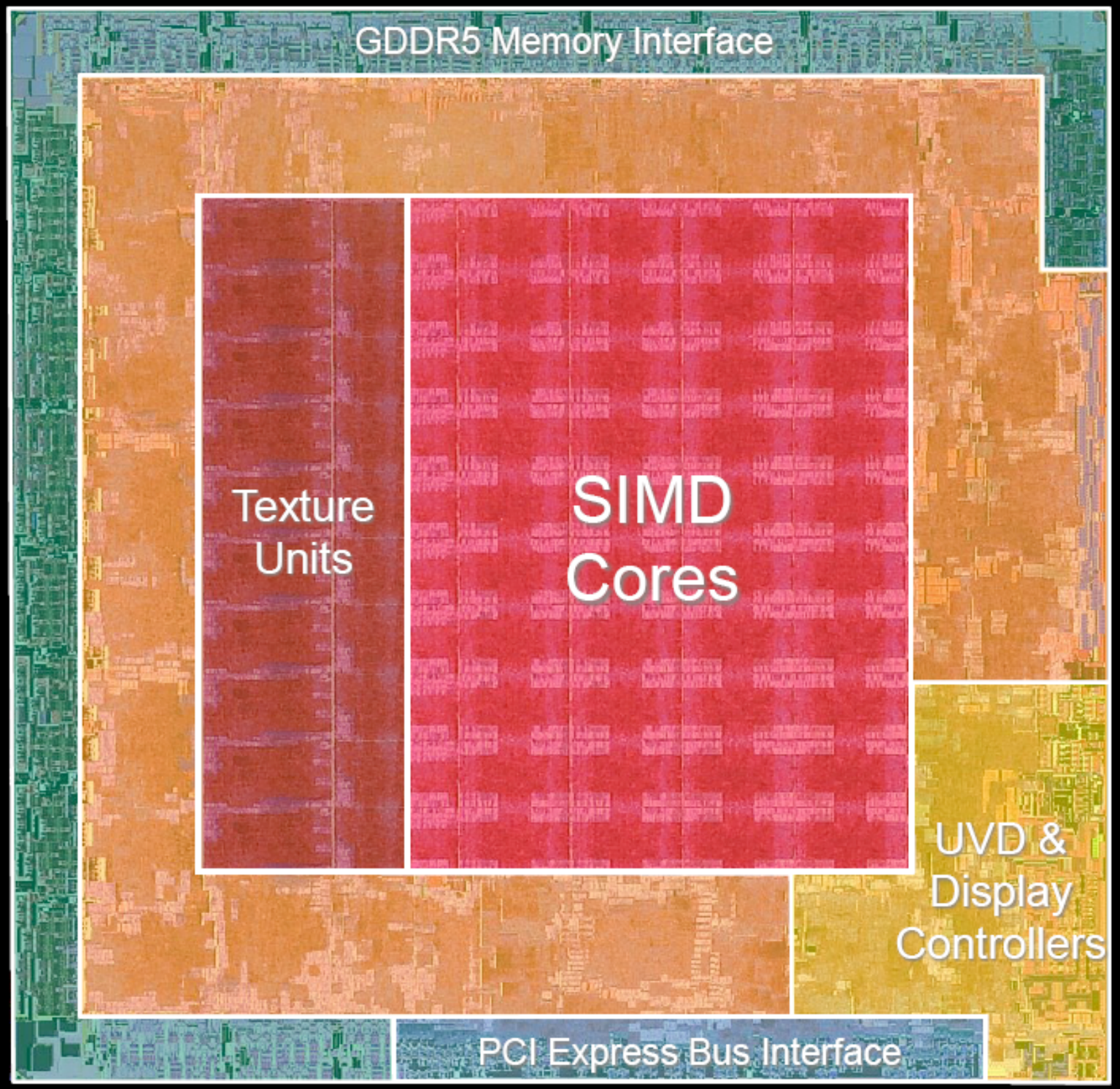
R770 Die

- 260mm2 ->small
- 956 MTransistors
  - ~1 Billion Transistors

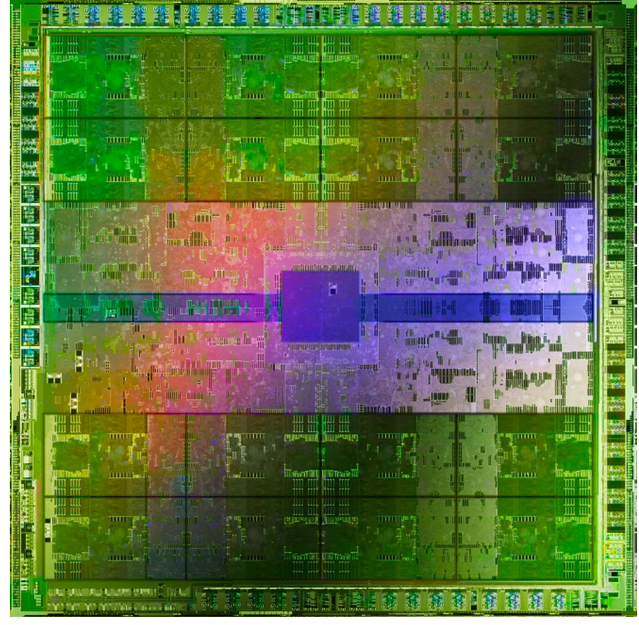


R770 Die usage

- Red
  - 10 SIMDs
- Orange
  - 64 z/stencil
  - 40 texture



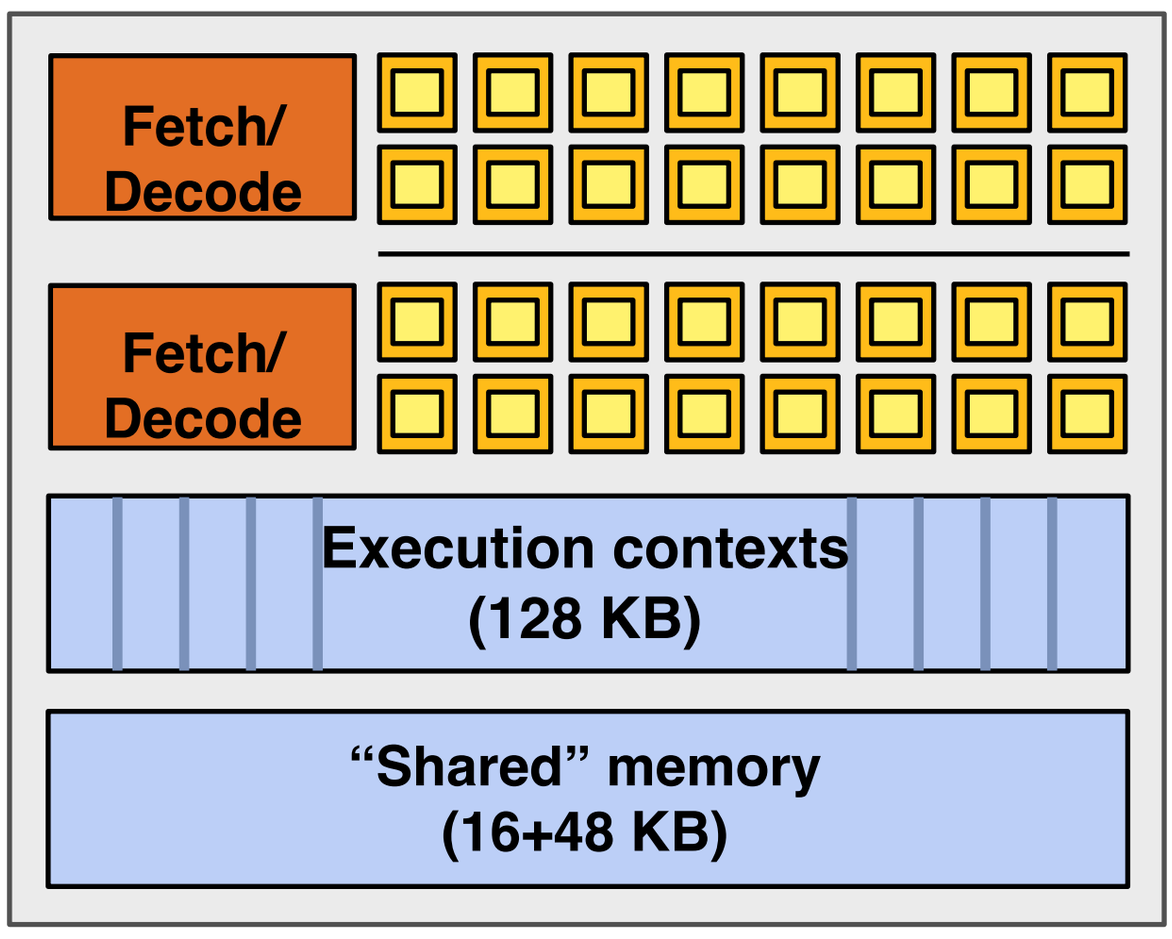
Multi-Graphics core nVidia GF100 (Fermi)



- GeForce GTX 480
  - Released March, 2010
- **4 x Triangle rate, 4 rasterization engines**
- 4 GPC x 3-4 SM x 32 scalars = 480 scalars @ 1401MHz
- Memory Hierarchy
  - **New L1-L2 cache for shader read/write**
  - L1 cache (per SM) is 16 or 48 KB
  - 768KB L2 cache is coherent and shared with texture and other parts of the graphics pipeline

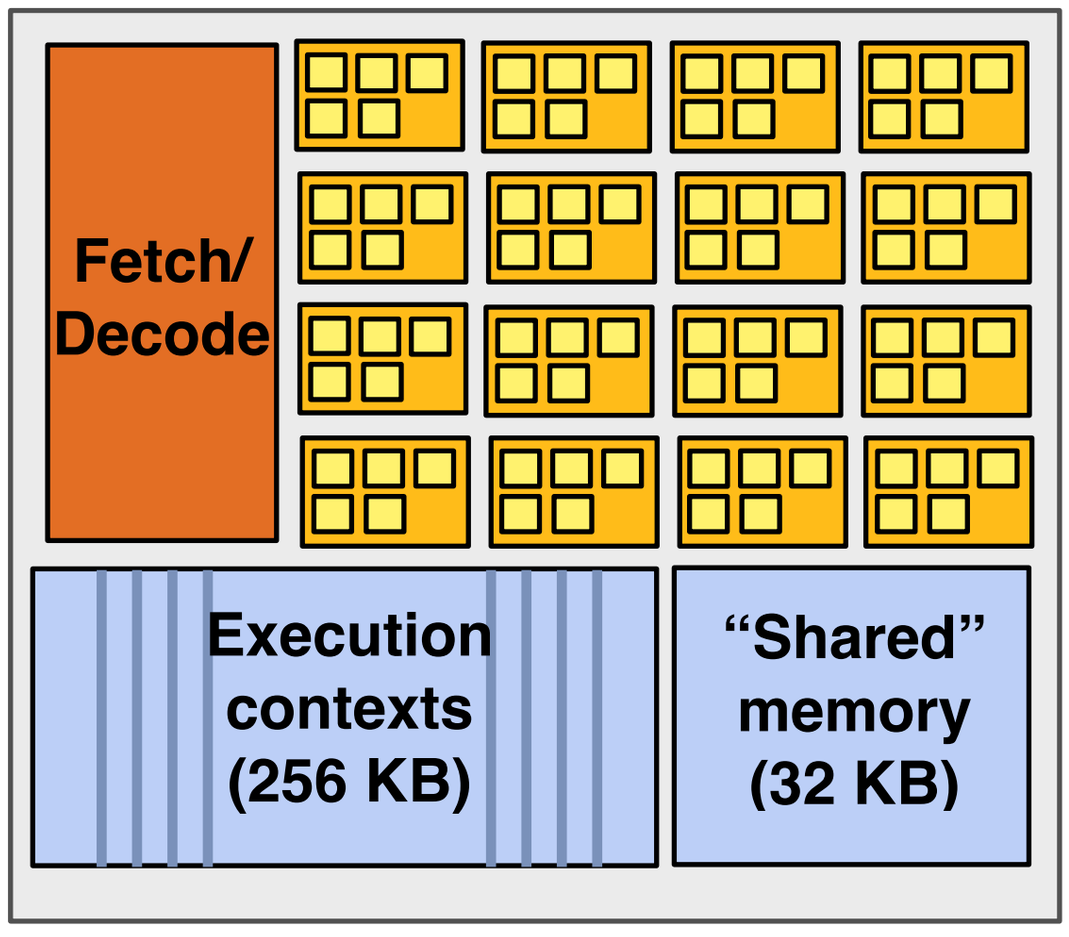
Comparing GPU shaders

NVIDIA GeForce GTX 480 "SM"



15 Streaming Multiprocessors

ATI Radeon HD 5870 "SIMD-engine"

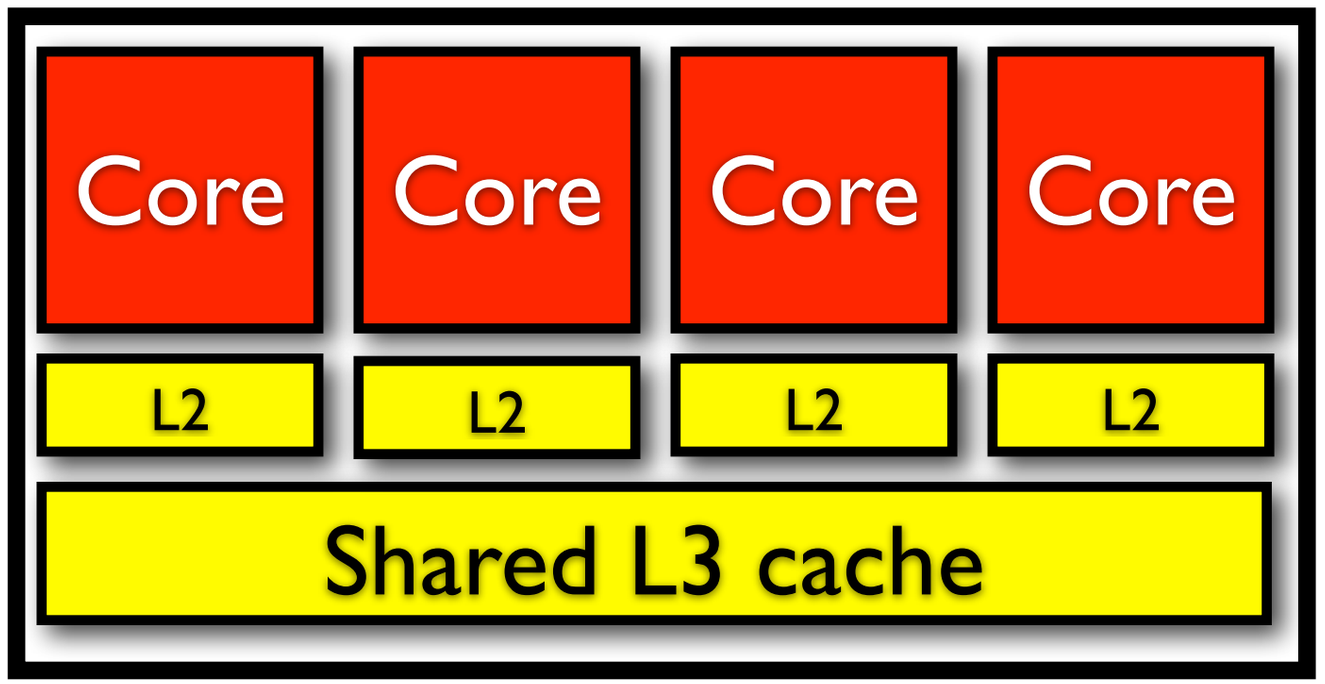


20 SIMD-engines

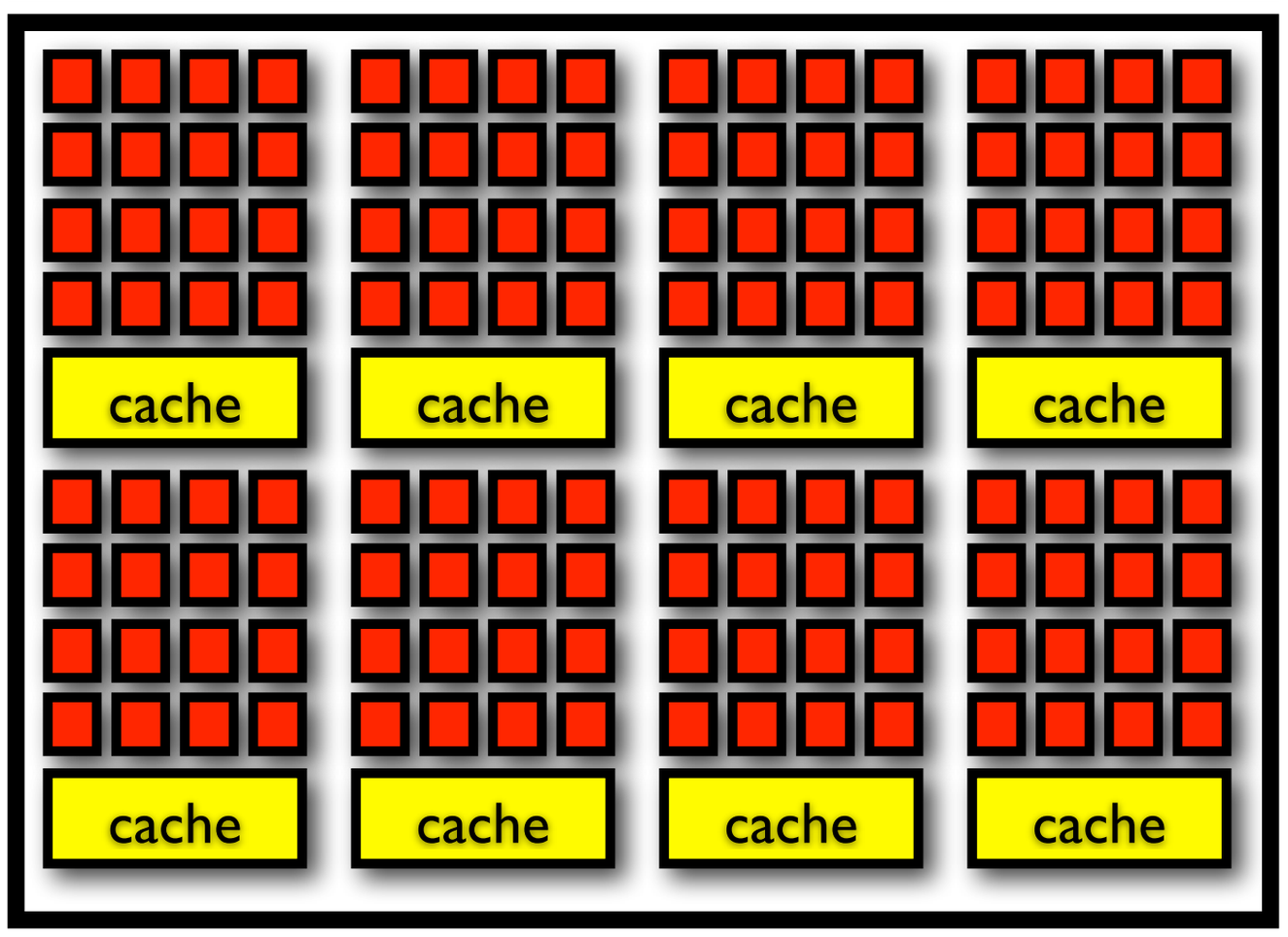
Images courtesy Kayvon Fatahalian

Comparing CPUs and GPUs

CPU



GPU

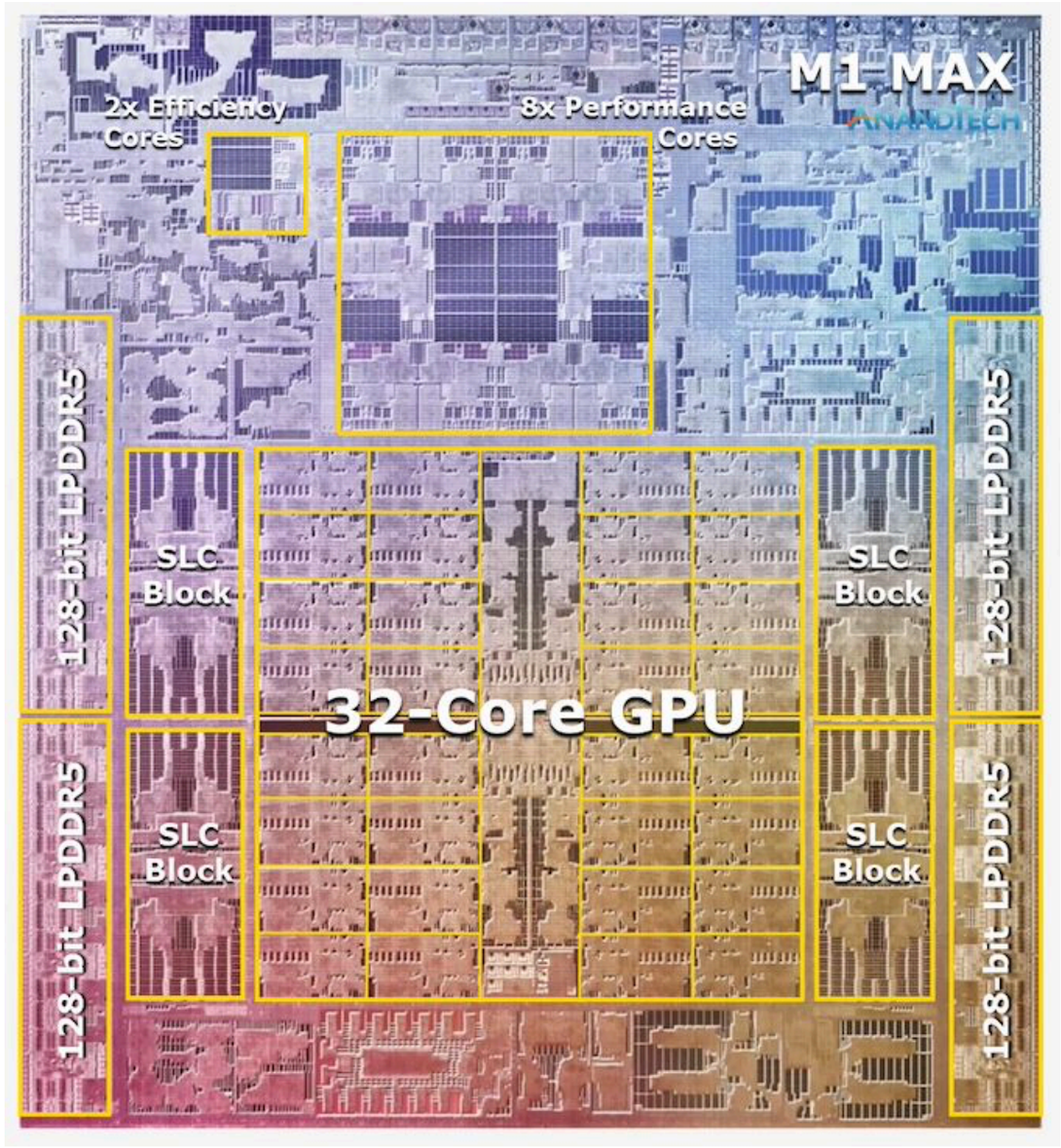


NVIDIA Ada RTX 4090

- 16,384 SPs, 82 TeraFLOPs, released October 2022
- 576 Tensor cores

GPUs in 2022

- NVIDIA RTX 4090, 82 TFLOPS
- AMD RX 7900 XTX, 46 TFLOPS
- Intel
  - Arc A770 17 TFLOPS
- Apple M1/M2
  - Large low-power GPU with a Tiled based architecture
  - ~10 TFLOPS
  - Similar to PS5 and XBSX



Summary

- GPU shaders grew from low precision simple programs
  - GPU ALU is simpler than CPUs
    - SIMD allows less hardware for many instructions
    - GPUs hide memory latency with 1000s of threads
    - GPUs are massively parallel devices and can be used for General Purpose compute (GPGPU)

References

- Henry Fuchs et. al. "A Heterogeneous Multiprocessor Graphics System Using Processor-Enhanced Memories," Proceedings of SIGGRAPH '89
- Steven Molnar et. al. "PixelFlow: high-speed rendering using image composition", Proceedings of SIGGRAPH 92
- Kurt Akeley, "RealityEngine Graphics". Proceedings of SIGGRAPH '93
- John Montrym et. al. "InfiniteReality: a real-time graphics system", Proceedings of SIGGRAPH '97
- Joel McCormack, "Neon: A (Big) (Fast) Single-Chip 3D Workstation Graphics Accelerator" WRL Research Report 98/1 (Revised 99)
- Lindholm et. al., "A user-programmable vertex engine", Proceedings of SIGGRAPH '01
- Montrym et. al. "The GeForce 6800", IEEE Micro, March '05
- Lindholm et. al. "Nvidia tesla: A unified graphics and computing architecture", IEEE Micro, March-April '08
- Larry Seiler et. al. "Larrabee: A Many-Core x86 Architecture for Visual Computing", Proceedings of SIGGRAPH'08
- Jeremy Sugerman et. al. "GRAMPS: A Programming Model for Graphics Pipelines", ACM Transactions on Graphics January, '09
- Timo Aila et. al. "Understanding the Efficiency of Ray Traversal on GPUs. " High-Performance Graphics 2009.