GPU

Graphics Processing Unit

- **A graphics processing unit** (GPU), is similar CPU
- Designed specifically for performing the complex mathematical and geometric calculations that are necessary for graphics rendering.

Graphics Processing Unit

- A graphics processing unit (GPU) is a computer chip that performs rapid mathematical calculations, primarily for the purpose of rendering images.
- occasionally called **visual processing unit** (**VPU**)
- GPU is able to render images more quickly than a CPU because of its parallel processing architecture
- Nvidia introduced the first GPU, the GeForce 256, in 1999
- Others include AMD, Intel and ARM.
- In 2012, Nvidia released a virtualized GPU, which offloads graphics processing from the server CPU in a virtual desktop infrastructure.

Graphics Processing Unit

- GPUs are used in
  - Embedded Systems
  - Mobile phones
  - Personal computers
  - Workstations
  - Game consoles

GPU Vs CPU

- A GPU is tailored for highly parallel operation while a CPU executes programs serially
- For this reason, GPUs have many parallel execution units and higher transistor counts, while CPUs have few execution units and higher clock speeds
- A GPU is for the most part deterministic in its operation
- GPUs have much deeper pipelines (several thousand stages vs 10-20 for CPUs)
- GPUs have significantly faster and more advanced memory interfaces as they need to shift around a lot more data than CPUs

What are GPU’s Growth?

- Entertainment Industry has driven the economy of these chips?

– Males age 15-35 buy $10B in video games / year

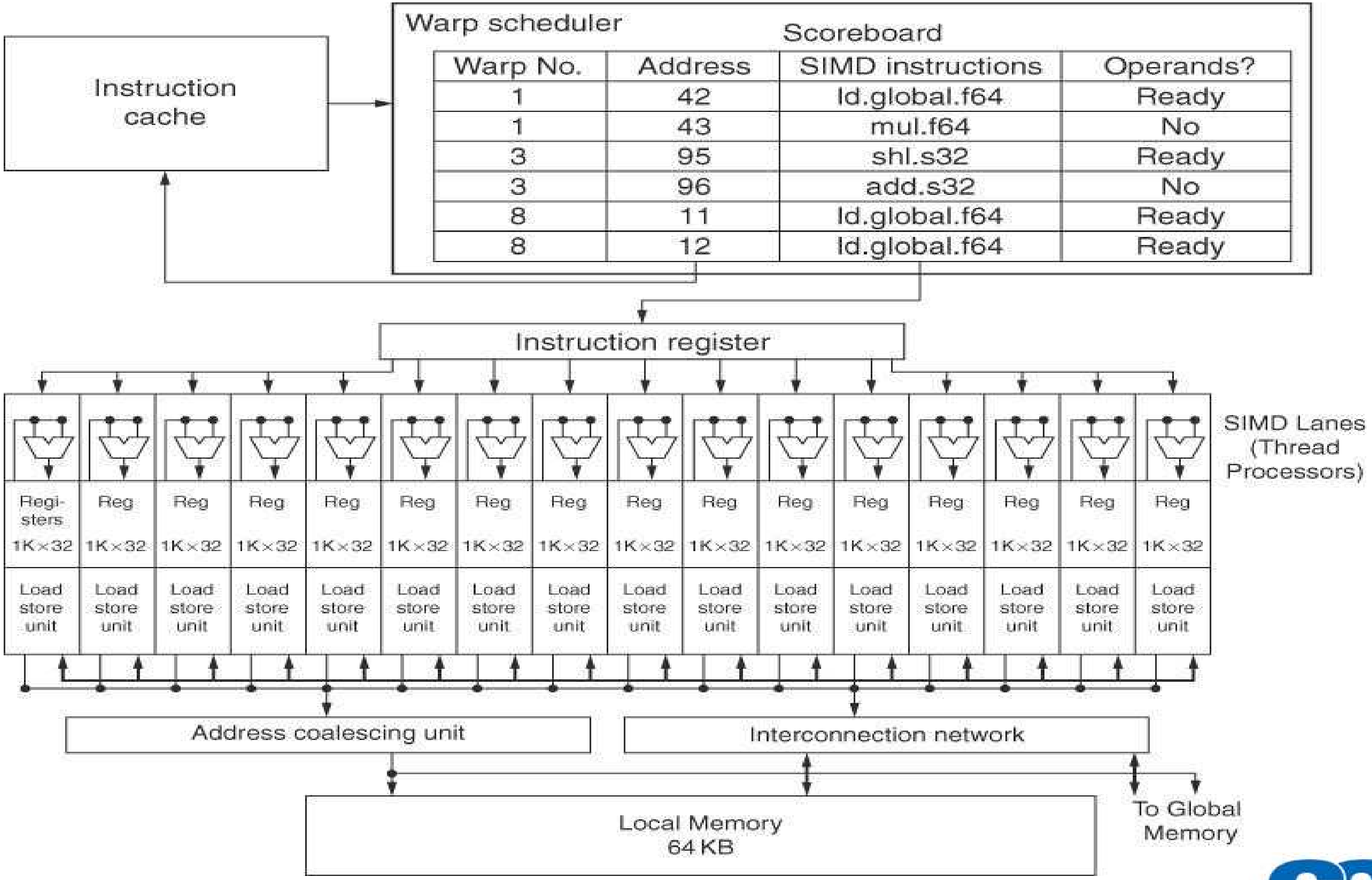
- Moore’s Law ++
- Simplified design (stream processing)
- Single-chip designs

GPU

- Very Efficient For
  - Fast Parallel Floating Point Processing
  - Single Instruction Multiple Data Operations
  - High Computation per Memory Access

- Not Efficient For
  - Double Precision
  - Logical Operations on Integer Data
  - Branching-Intensive Operations
  - Random Access, Memory-Intensive Operations

NVIDIA GPU –MTSIMD



Instruction cache

Warp scheduler

Scoreboard

Warp No.	Address	SIMD instructions	Operands?
1	42	ld.global.f64	Ready
1	43	mul.f64	No
3	95	shl.s32	Ready
3	96	add.s32	No
8	11	ld.global.f64	Ready
8	12	ld.global.f64	Ready

Instruction register

SIMD Lanes (Thread Processors)

Regi-sters	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg	Reg
1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32	1K×32
Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit	Load store unit

Address coalescing unit

Interconnection network

Local Memory 64 KB

To Global Memory

NVIDIA GPU- MTSIMD

- GPU is a multiprocessor composed of MTSIMD processors.
- It is similar to vector processor but with many parallel FU’s that are deeply pipelined.
- MTSIMD is a processor that executes code in the form of thread blocks.
- GPU H/W contains a collection of MTSIMD Processors execute a Grid of Thread Blocks.

NVIDIA GPU- MTSIMD

- GPU H/W has two levels of H/W schedulers

1.Thread Block Scheduler:

- – Thread block scheduler is similar to control unit in Vector processor
- – det the no of thread blocks for a loop and allocates them to diff MTSIMD processors.
- – ensures that thread blocks are assigned to the processors whose local memories have the corresponding data.

NVIDIA GPU- FERMI MTSIMD

2. SIMD Thread Scheduler:

- SIMD Thread scheduler has scoreboard logic
- It keeps track of 48 threads of SIMD instructions
- It tells that which thread of SIMD instructions are ready to run
- It sends those instructions to dispatch unit to be run on MTSIMD processor
- within a SIMD Processor, which schedules when threads of

SIMD instructions should run

NVIDIA GPU- MTSIMD

- It has many parallel functional units
- SIMD Processors with separate PCs and are programmed using threads.
- Each MTSIMD Processor is assigned 512 elements of the vectors to work on
- SIMD processors have 32,768 registers
- Like vector processor these registers are logically divided across SIMD lanes.

NVIDIA GPU- MTSIMD

- Each SIMD Thread has 64 vector registers of 32 elements with 32 bit each.
- FERMI has 16 physical lanes each contain 2048 registers
- Thread Blocks would contain 512/32 = 16 SIMD threads.
- Each thread of SIMD instructions in this example compute 32 of the elements of the computation.

NVIDIA GPU- MTSIMD

- GPU applications have so many threads of SIMD instructions that multithreading can
  - hide the latency to DRAM
  - increase utilization of multithreaded SIMD Processors

NVIDA GPU ISA

- PTX(Parallel Thread Execution) provides a stable instruction set for GPUs
- H/W instruction set is hidden from the programmer
- PTX instructions describe the operations on a single CUDA thread
- PTX uses virtual registers
- Translation to machine code is performed in software

NVIDA GPU ISA

- Format of a PTX instruction is

opcode.type d, a, b, c;

– where d is the destination operand; a, b, and c are source operands

- Source operands are 32-bit or 64-bit registers or a constant value. Destinations are registers, except for store instructions.

NVIDA GPU ISA

- the operation type is one of the following:

Type	.type Specifier			
• Untyped bits 8, 16, 32, and 64 bits	. b8,	b16,	. b32,	b64
• Unsigned integer 8, 16, 32, and 64 bits	.U8,	. U16,	U32,	u64
• Signed integer 8, 16, 32, and 64 bits	. S8,	. S16,	. S32,	S64
• Floating Point 16, 32, and 64 bits	.J16,	J32,	J64	

Conditional Branching

- Like vector architectures, GPU branch hardware uses internal masks
- Also uses
  - Branch synchronization stack
  - Entries consist of masks for each SIMD lane
  - i.e. which threads commit their results
- Per-thread-lane 1-bit predicate register, specified by programmer