GPU (Graphics Processing Unit):
A processor designed to execute a class of programs that includes 3D graphics, scientific computation, and machine learning using a large number of threads.

GPU Goals

Goal: Maximize operation throughput of a chip.

Approach:

We start with a target area (or number of gates) and power budget.

Chip will consist of multiple cores.

Find a core design that maximizes...
... FLOPS per unit area ...
... or FLOPS per unit power.

Then fill chip.

Do this with a *target workload* in mind.

A Brief History

GPUs originally designed *only* for 3D graphics [1].

Large economies of scale made them cheap.

Resourceful scientific users disguised their work as 3D graphics [2].

GPU makers started supporting scientific and other non-graphical work.

GPUs evolved into a second kind of processor, with 3D graphics just one application.

Current Status

Always used in systems needing 3D graphics, from cell phones to workstations.

Typically used for ML *training* workloads.

Often used for scientific computation, but acceptance has been gradual due to difficulty of porting code.

GPU Product Examples

- NVIDIA RTX 5090 — High-end GPU for home use.
- NVIDIA Blackwell B100 — High-end GPU for non-graphical computing...
  ... including half-precision FP support for deep learning applications.
- AMD Radeon R9 — High-end GPU for home use.

CPU v. GPU Comparison

The Devices

These are roughly contemporary.

Devices used in ECE Student Workstation Laboratory

CPU: Intel(R) Xeon(R) Silver 4316 CPU @ 2.30GHz

GPU: NVIDIA GeForce RTX 4090 @ 2.52 GHz

Note: RTX 4090 in some ways superior to data-center class H100.

Floating-Point Bandwidth Comparison

Comparison Units

Bandwidth:
Peak rate, in units of operations per time.

Operation: One fused FP multiply/add operation.

Precision: IEEE 754 Binary Single Precision (32 bit).

Time Unit: Cycles. (In this comparison about equivalent.)

FP Bandwidths

Xeon — 20 cores, two 16-SP-lane vector (SIMD) functional units / core.

$$20 \times 2 \times 16 = 640\,\text{FLOP/cyc}$$

RTX 4090 — 128 symmetric multiprocessors (SMs), 128 FP32 units per SM.

$$128 \times 128 = 16384\,\text{FLOP/cyc}$$

GPU has about 25× FP bandwidth.

Off-Chip Data Bandwidth Comparison

Comparison Units

Rate data can be moved off chip.

Data Bandwidths

Xeon – Based on measurement.

38 GB/s.

RTX 4090.

Data retrieved using API: 1008 GB/s.

Measured: 971 GB/s.

GPU has about 25× data bandwidth.

Comparison of FP and Data bandwidths.

Computational Intensity: FP BW / Data BW.

GPU and CPU both at about 65.

Probably not a coincidence that these are similar.

Comparison of Threads of Control.

Minimum Number

Xeon — 20.

4090 — $128 \times 128 = 16384$.

Wow, over 800× more.

Maximum Number

Xeon — $20 \times 2 = 40$ (Hyperthreading on).

4090 — $128 \times 48 \times 32 = 196608$.

Wow, almost 5000× more.

Making use of these threads is a primary challenge of GPU programming.

Comparison of Cache, Scratchpad

```
Cache Xeon
 L1d:  20 *    48 kiB =   960 kiB
 L2:   20 *  1280 kiB = 25600 kiB
 L3:       30720 kiB = 30720 kiB
 BW:   38 GB/s (measured)

 CTX: 32 AVX 512 = 2048 B
      16 GPR  8B =  182 B
      40 * 2130 B =  85 kB

RTX 4090
 L1:  128 *   100 kiB = 12800 kiB
 L2:       73728 kiB = 73728 kiB
 BW: 1000 GB/s (API)  About 26x CPU
 CTX: 128 64 K 4 B    = 32768 kiB
```

SIMT Characteristics

Core (SM) holds many threads (thread contexts).

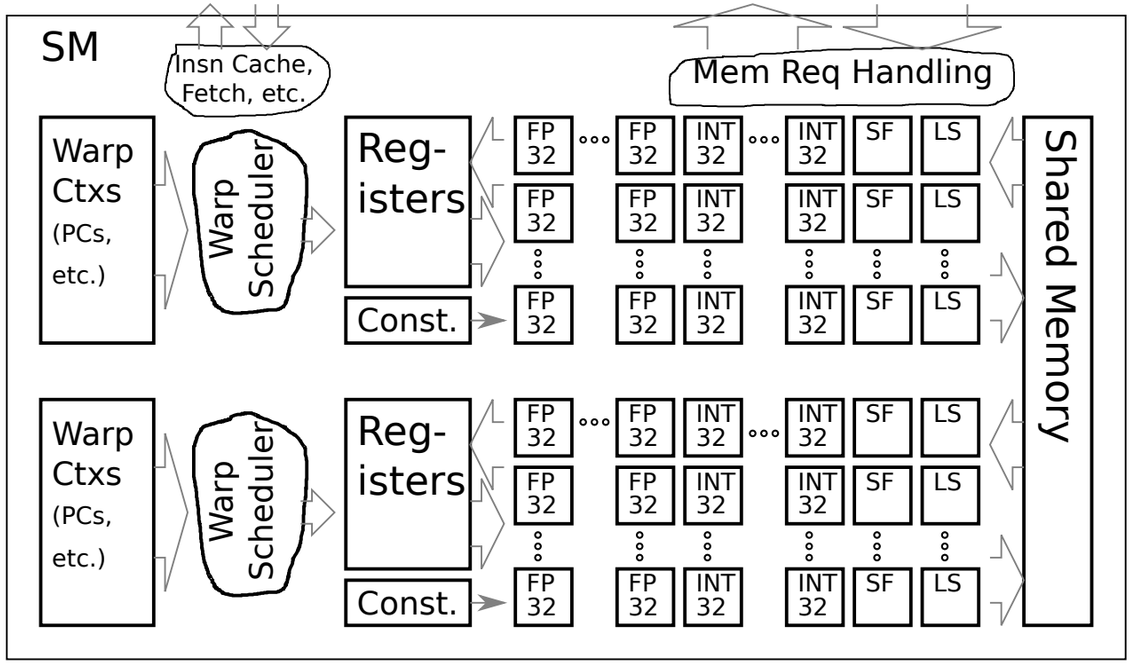
Threads are organized into groups called *warps*.

As in an ordinary multithreaded system . . .
. . . each thread has its own PC.

Instruction fetch is performed for an entire warp . . .
. . . using the PC of one of the threads in the warp.

This works well when . . .
. . . all of the threads in a warp have the same PC.

If threads have different PC values process must be repeated.



Typical SM, with 2 Warp Schedulers.

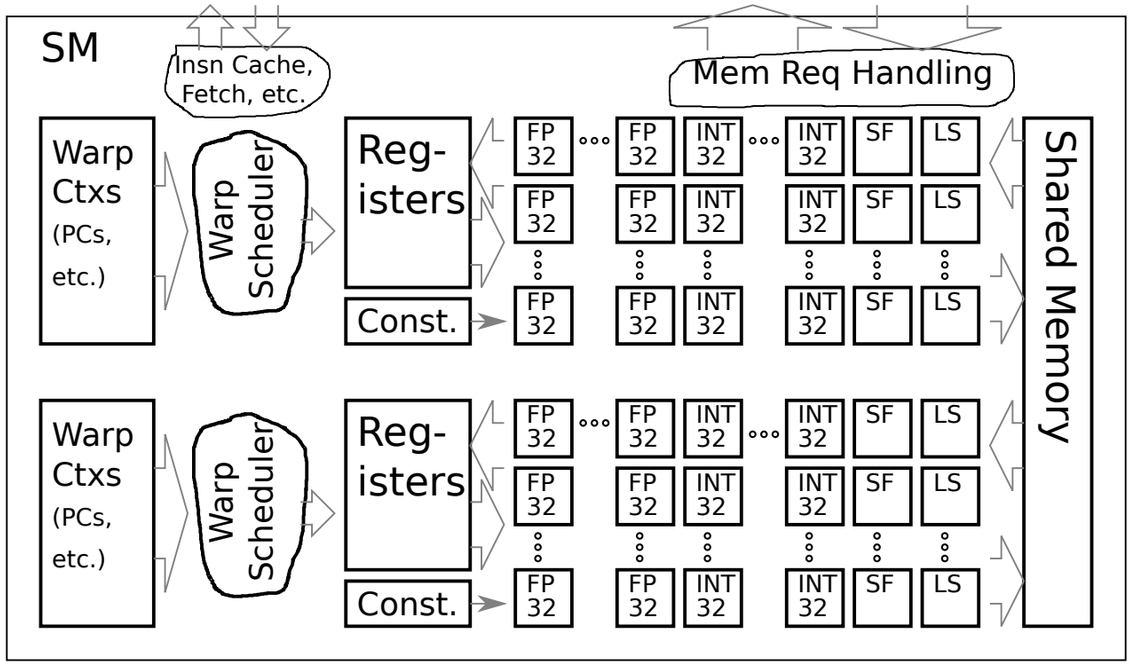
SIMT Thread Dispatch

Dispatch:

Sending a thread (in this case) to an execution unit.

In other core organizations dispatch was one cycle . . .

. . . but for SIMT can be multiple cycles.



Typical SM, with 2 Warp Schedulers.

References:

[1] Dally, W. J., Keckler, S. W., and Kirk, D. B. Evolution of the graphics processing unit (GPU). *IEEE Micro 41*, 06 (Nov 2021), 42–51. http://dx.doi.org/10.1109/MM.2021.3113475.

[2] Thompson, C. J., Hahn, S., and Oskin, M. Using modern graphics architectures for general-purpose computing: a framework and analysis. In *Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture* (Washington, DC, USA, 2002), MICRO 35, IEEE Computer Society Press, p. 306317.