CE/CZ 3001:
Advanced Computer Architecture

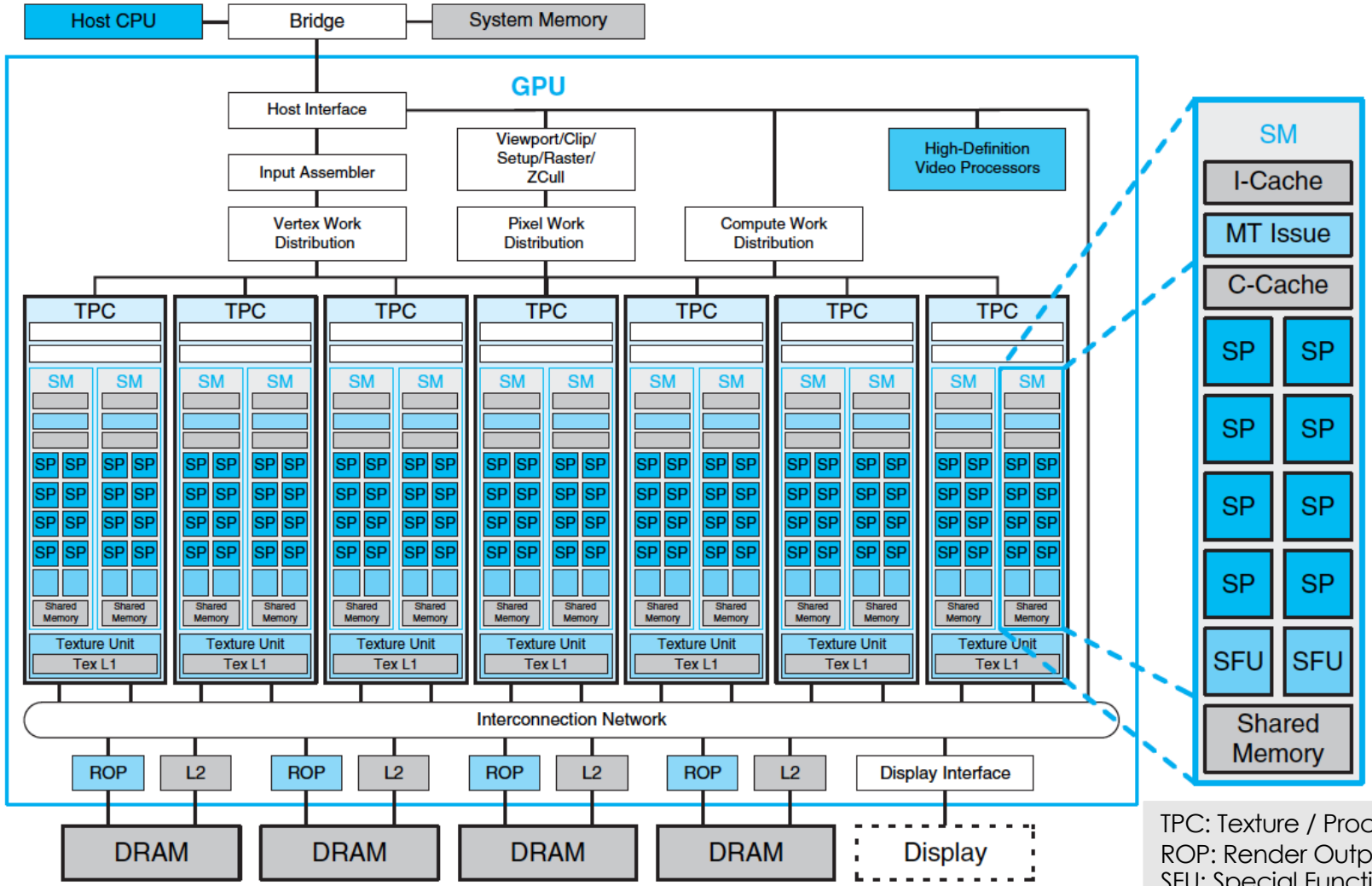
Module 6: GPU Architecture and CUDA Programming

- GPU Internal Operation

Asst Prof Liu Weichen
School of Computer Science and Engineering
Nanyang Technological University, Singapore

Summary of Part 3

- Block allocation to SM
- Warp and warp scheduler
- SIMT architecture
- Hardware resource allocation
- Hardware resource usage



TPC: Texture / Processor Cluster
ROP: Render Output Unit
SFU: Special Function Unit

Excise 1

For vector addition, assume that the vector length is 8,000, each thread calculates four output elements, and the thread block size is 1,024 threads. The programmer configures the kernel launch to have a minimal number of blocks to cover all output elements. How many threads will be in the grid?

(A) 1,024

(B) 8,196

(C) 2,048

(D) 8,000

Each block covers 4096 elements.
The minimal number of blocks to cover 8000 is 2.

So the number of threads: 2*1024 = 2048.

Excise 2

Your kernel runs on a device where each Streaming Multiprocessor has 32K registers and 64KB of shared memory. The kernel code uses 30 registers (local variables) and 5KB of shared memory. The kernel is launched with 1,920,000 threads in total organized into a square grid with dimension 50 on each side. What is the maximum number of simultaneous blocks that will run on a single SM?

(A) 1
(B) 2
(C) 3
(D) 4

Each block has 1920000/50/50 = 768 threads,
so will use 768*30 = 23,040 registers.
Each SM has 32K registers.
There can only be 1 block at a time.

Excise 3

You would like to run a kernel on a GPU device. There are a total of 1,000,000 threads needed to launch the kernel. The kernel uses 10 registers per thread and 10KB shared memory per block. The kernel is launched as a cubic grid of cubic blocks. The grid dimension is 10 in X, Y and Z direction. What is the maximum number of simultaneous blocks that will run on a single SM?

Maximum number of threads per block	1024
Maximum number of resident blocks per multiprocessor	32
Maximum number of resident warps per multiprocessor	64
Maximum number of resident threads per multiprocessor	2048
Number of 32-bit registers per multiprocessor	64K
Maximum amount of shared memory per multiprocessor	64KB
Maximum amount of shared memory per thread block	48KB

(A) 1
(B) 2
(C) 3
(D) 6

Excise 3

Total: 1,000,000 threads

Resources: 10 registers per thread
10KB shared memory per block

blocks: 1,000

threads per block: 1,000,000/1,000 = 1,000 (# warps per block: 1,000/32 = 32)

What is the maximum number of simultaneous blocks that will run on a single SM?

Maximum number of threads per block	1024
Maximum number of resident blocks per multiprocessor	32
Maximum number of resident warps per multiprocessor	64
Maximum number of resident threads per multiprocessor	2048
Number of 32-bit registers per multiprocessor	64K
Maximum amount of shared memory per multiprocessor	64KB
Maximum amount of shared memory per thread block	48KB

(A) 1
(B) 2
(C) 3
(D) 6

Excise 4

Your friend has profiled a kernel and discovered that the performance is limited by the latency of the divide operator. It turns out variable d in his kernel code (below) is 1 in half of the time, so your friend proposed the following optimization:

```
if (1 == d)
    r = d;
else
    r = 1 / d;
```

Unfortunately, the performance did not improve. In one sentence, explain why.

Answer: The code is executed by all threads and some of the threads will still succeed so now the code has thread divergence.

Summary

- Basic GPU architectures
- Kernels, threads and thread blocks
- Sync and memory management between host and device
- Data sharing and sync between threads

- Block allocation to SM
- Warp and warp scheduler
- SIMT architecture
- Hardware resource allocation
- Hardware resource usage