ELE 455/555
Computer System Engineering

Section 4 – Parallel Processing

Class 3 – GPUs

Parallel Processing

Graphics Processor Units

- GPU Video

Parallel Processing

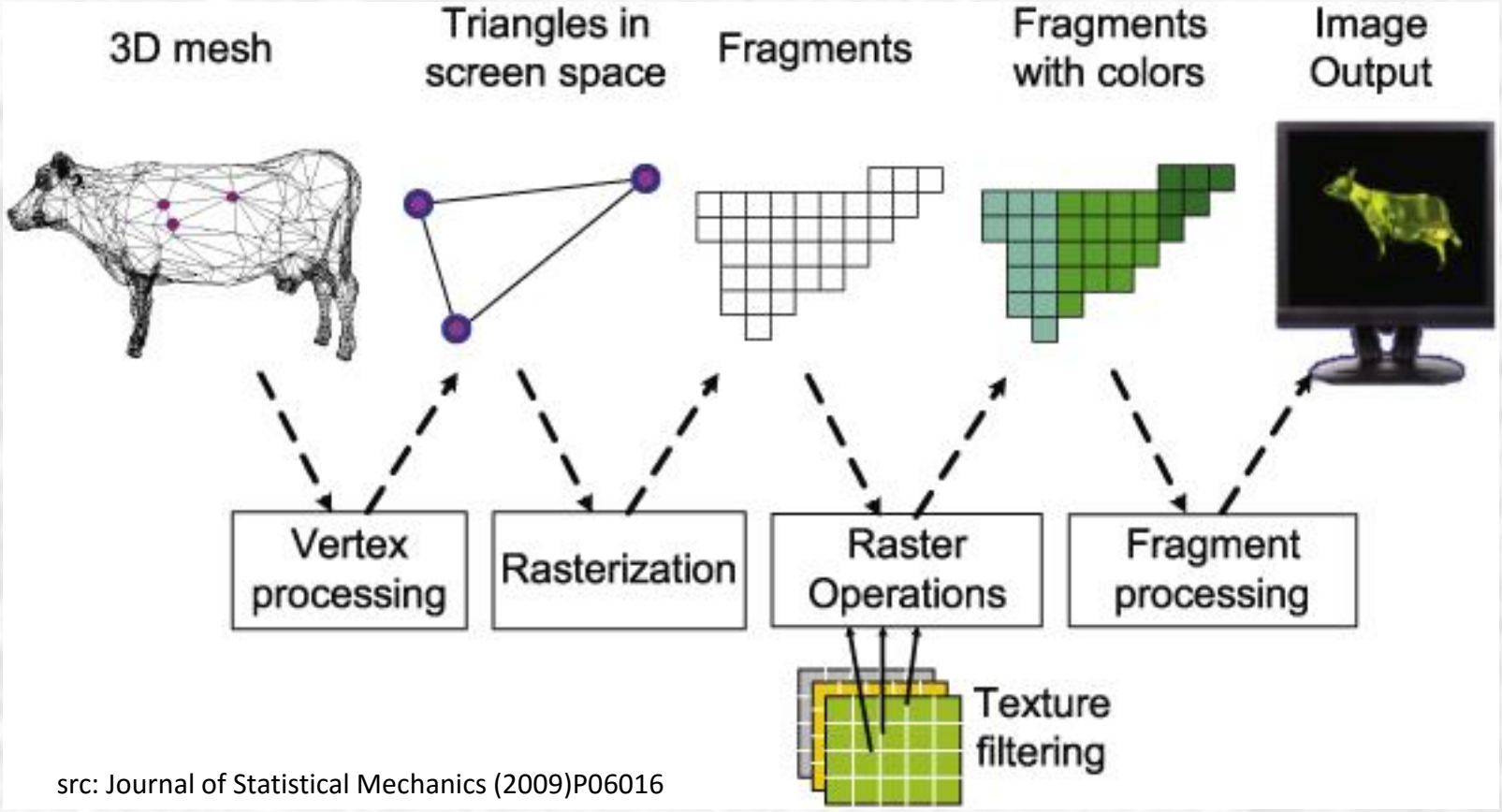
Graphics Processor Units

- Graphics Processing Unit (GPU)
  - Optimized processor for computing 2D and 3D graphics objects
    - 2D/3D graphics
    - Images
    - Video
  - Used in
    - Window based operating systems
    - Graphical user interfaces
    - Video games
  - Highly parallel, highly multithreaded multiprocessor

Parallel Processing

Graphics Processor Units

- Graphics Pipeline



src: Journal of Statistical Mechanics (2009)P06016

Parallel Processing

Graphics Processor Units

- Graphics Pipeline
  - Vertex
    - Location (point) in 3D space on the graphics object
    - Includes: location, color, texture, motion, … information
  - Vertex Shader
    - Operations performed on each vertex
      - Transform 3D location to 2D screen location
      - Includes “Z” processing to emulate depth on the screen

Parallel Processing

Graphics Processor Units

- Graphics Pipeline
  - Geometry
    - Point, line or triangle created from the vertices
    - Includes: location, color, texture, motion, … information
  - Geometry Shader
    - Operations performed on each geometry
      - Creation of the geometry
      - Combine or divide geometries
      - Add or delete geometries based on detail requirements (zoom)

Parallel Processing

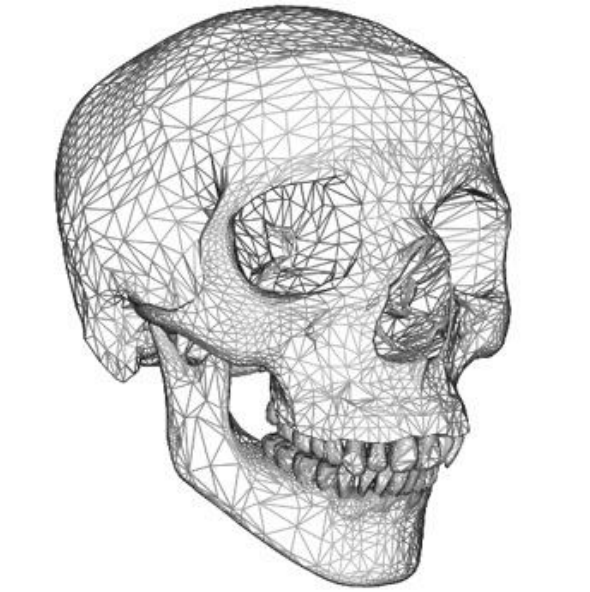
Graphics Processor Units

- Graphics Pipeline
  - Pixel
    - Smallest render unit on the screen
  - Pixel Shader
    - Operations performed on each pixel
      - Color
      - Texture mapping
      - Lighting

Parallel Processing

Graphics Processor Units

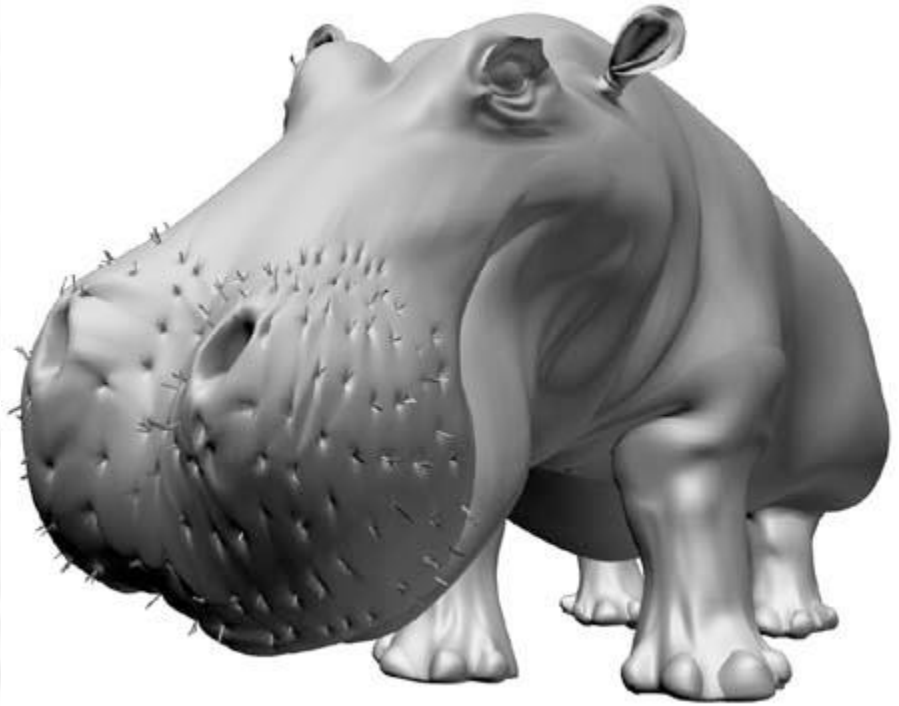
- Graphics Pipeline



Parallel Processing

Graphics Processor Units

- Graphics Pipeline



Parallel Processing

Graphics Processor Units

- History
  - 1990s - Video Graphics Array controller (VGA)
    - Memory controller used to paint to the screen
  - 2000 - Integration allowed most of the processing to happen in the GPU using fixed hardware
    - Triangle setup, rasterization, texture mapping
  - Fixed hardware was replaced with programmable hardware
  - Programmable hardware was consolidated into a multithreaded multiprocessor architecture
  - 2010 – Additional capability added to support general computing operations

Parallel Processing

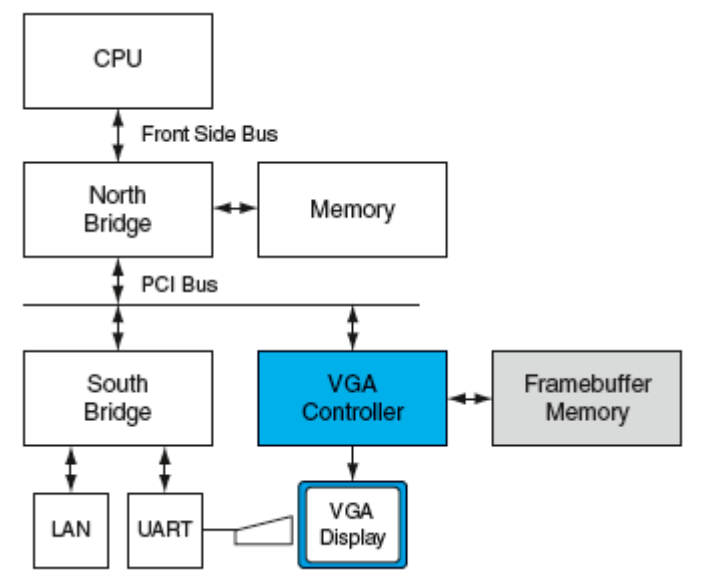
Graphics Processor Units

- Application Programming Interface (API)
  - Allow programmers to write to the API and not be concerned about the underlying hardware
  - Allow the underlying hardware to progress at a rapid pace
  - OpenGL
    - Open standard
    - Broadly available
  - DirectX
    - Microsoft APIs

Parallel Processing

Graphics Processor Units

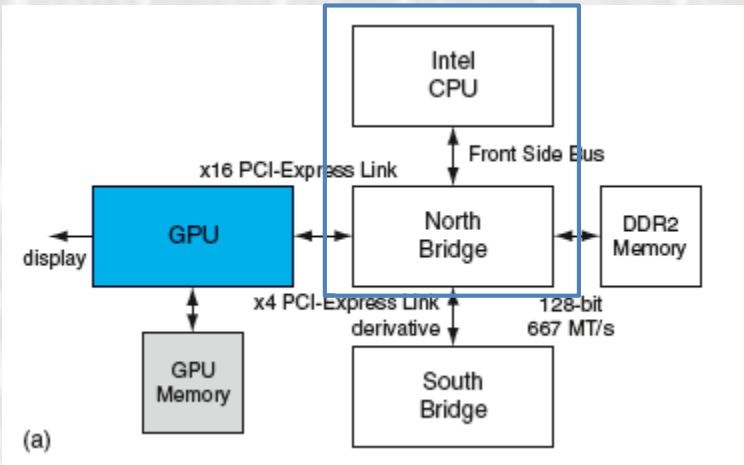
- Heterogeneous system
  - GPUs used as co-processors for the main CPU
  - Early implementation



Parallel Processing

Graphics Processor Units

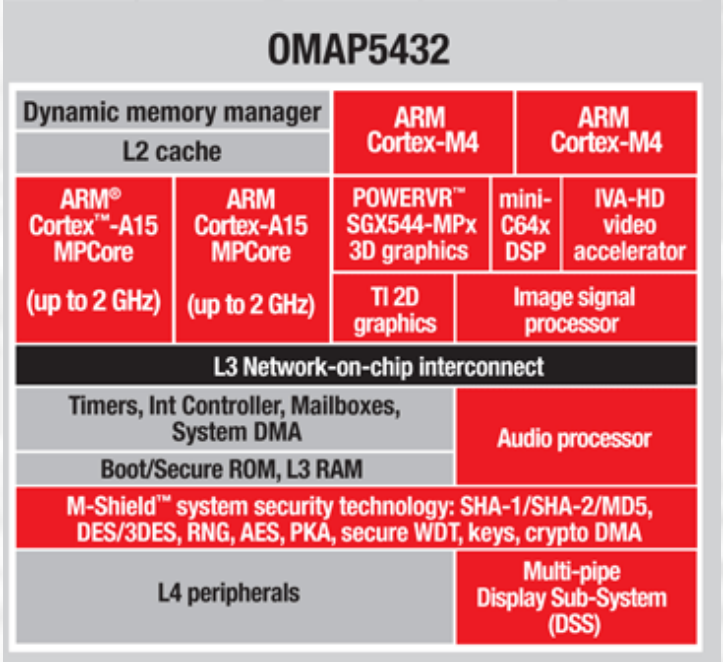
- Heterogeneous system
  - GPUs used as co-processors for the main CPU
  - Current implementation



Parallel Processing

Graphics Processor Units

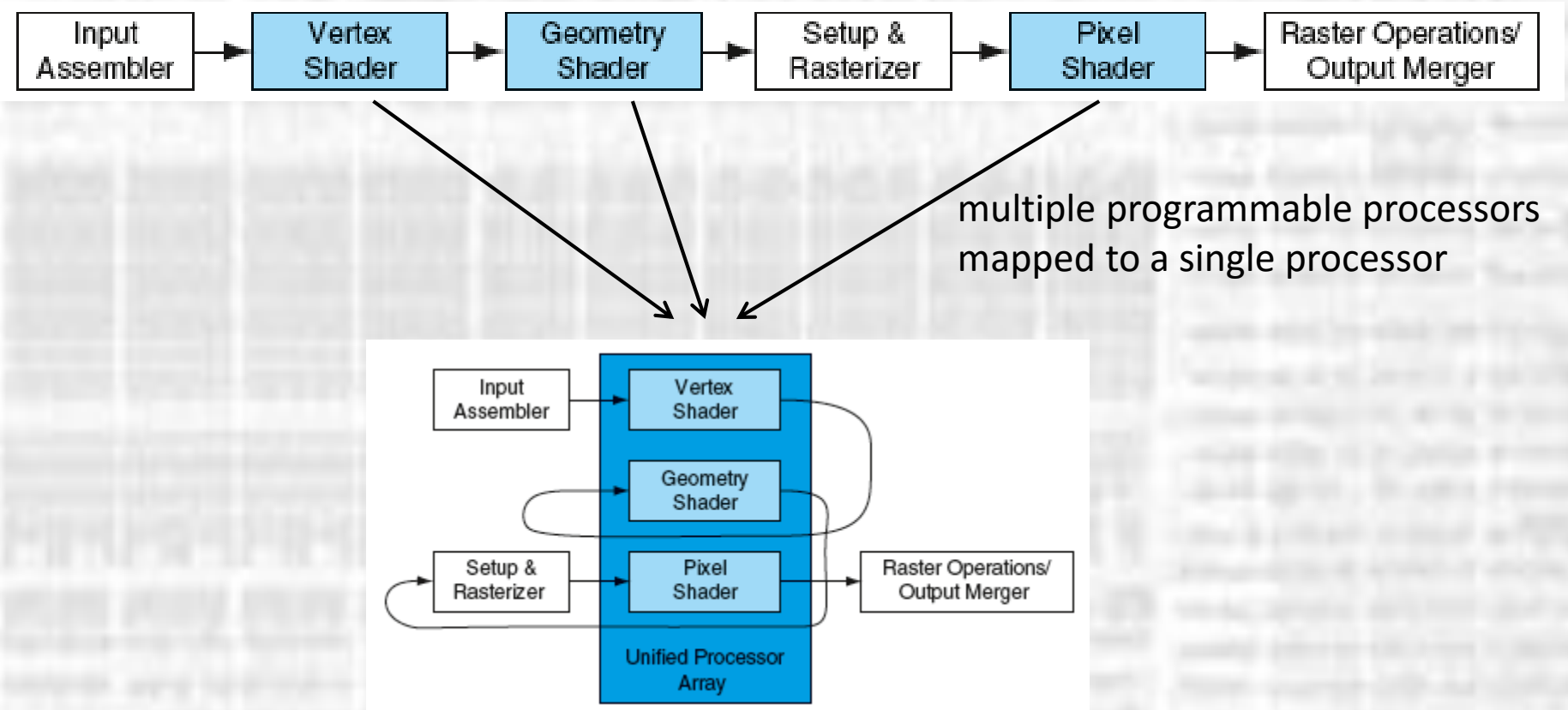
- Heterogeneous system
  - GPUs used as co-processors for the main CPU
  - Current wireless implementation



Parallel Processing

Graphics Processor Units

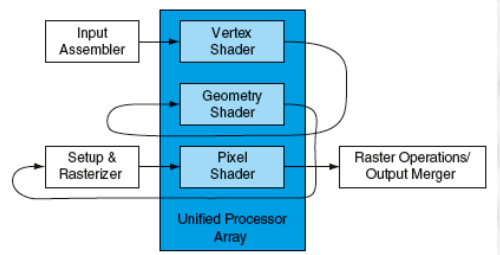
- Unified GPU Architecture



Parallel Processing

Graphics Processor Units

- Unified GPU Architecture

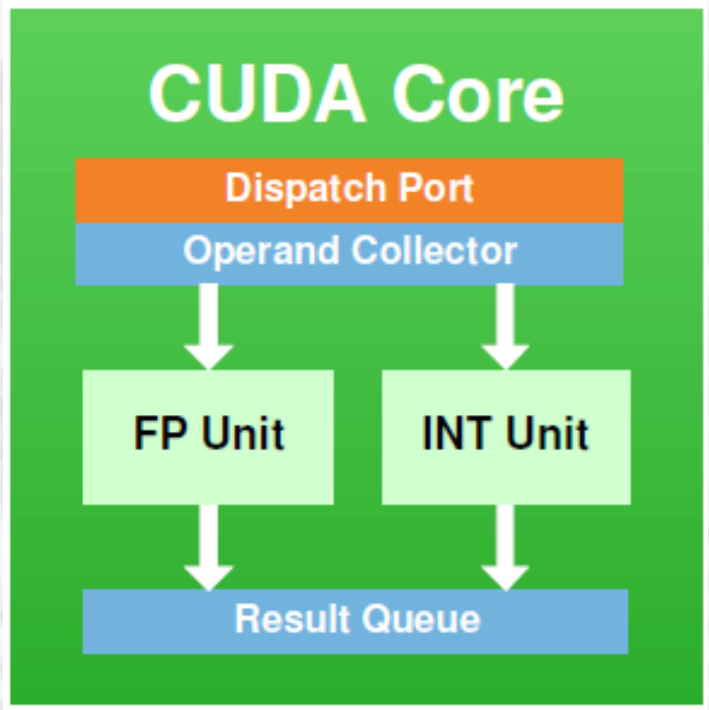


  - Built from a parallel array of unified processors
  - Tightly coupled with fixed function processors
    - rasterization, compression, video decoding, …
  - Focus is on executing large numbers of parallel threads on large numbers of cores
  - Utilize multithreading to hide memory latency instead of using multi-level caches

Parallel Processing

Graphics Processor Units

- Unified GPU Architecture
  - Streaming Processor Core
    - Pipelined
    - Superscalar
    - Highly Multithreaded
      - 96 Concurrent Threads
      - Hardware managed
    - 1024, 32bit registers

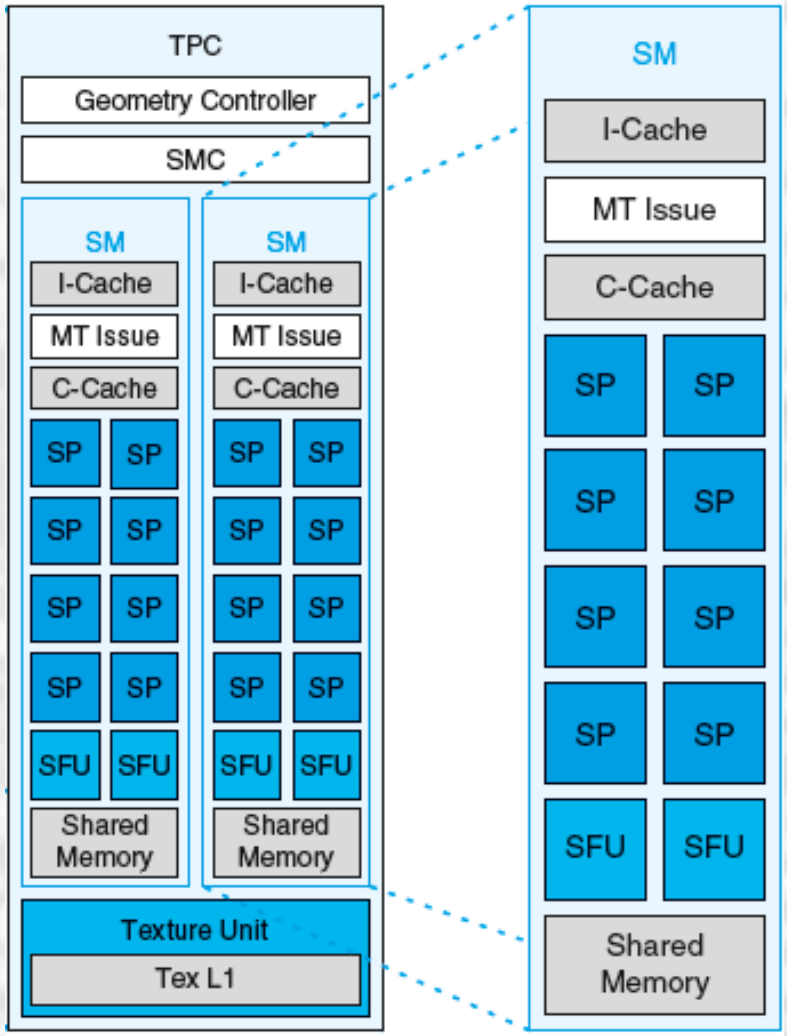


Nvidia Tesla

Parallel Processing

Graphics Processor Units

- Unified GPU Architecture
  - Streaming Multiprocessor
    - 8 Streaming Processor Cores (SP)
    - 2 Special Function Units (SFU)
      - Transcendentals (sin, cos, log, exp, …)
    - Instruction Cache
    - Constant Cache
    - Multithreaded Issue unit
    - Shared memory
  - Texture Processor Cluster
    - 2 Streaming Multiprocessors
    - Controller
    - Texture Unit

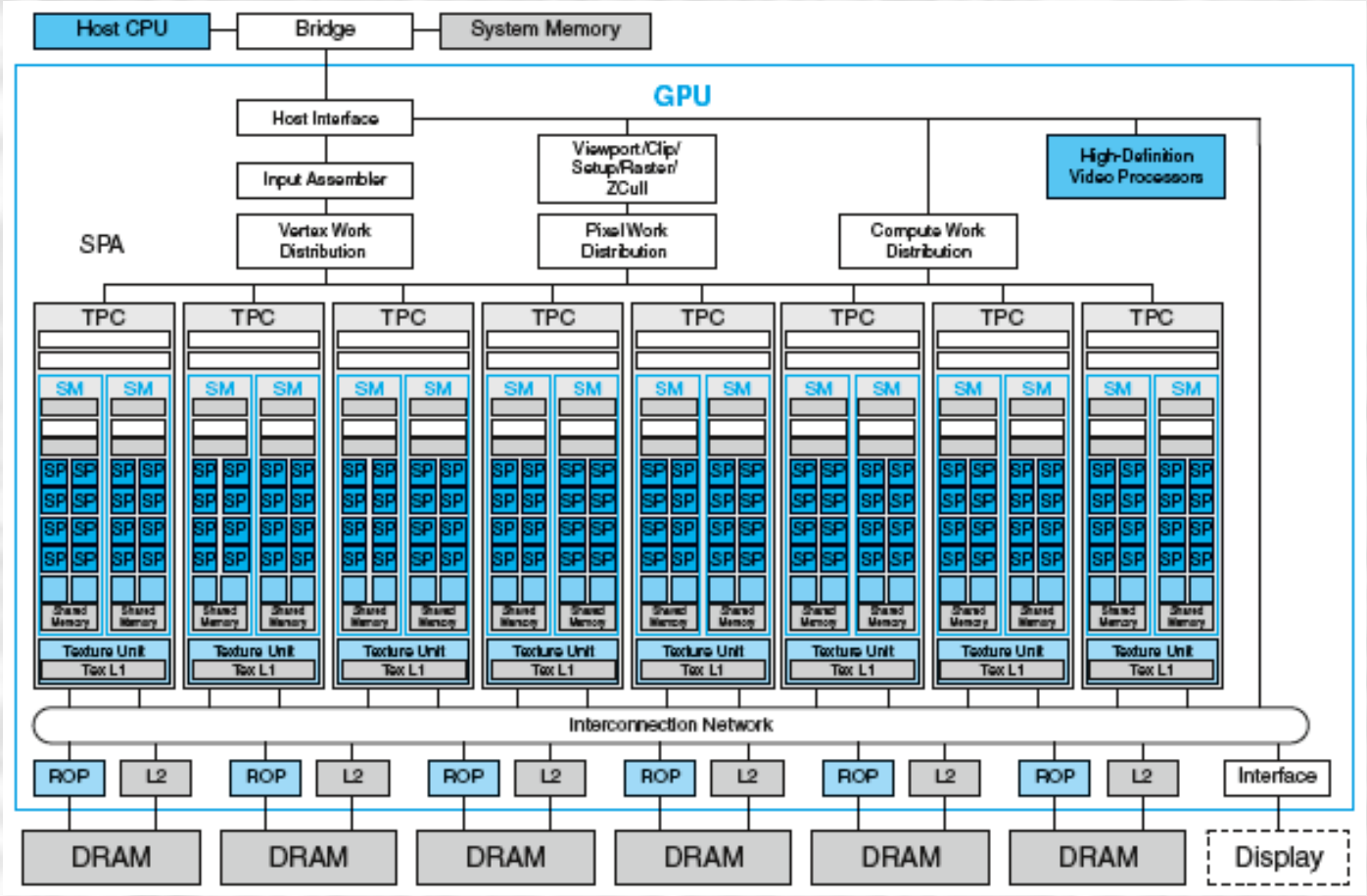


Nvidia Tesla

Parallel Processing

Graphics Processor Units

- Unified GPU Architecture



Nvidia Tesla

Parallel Processing

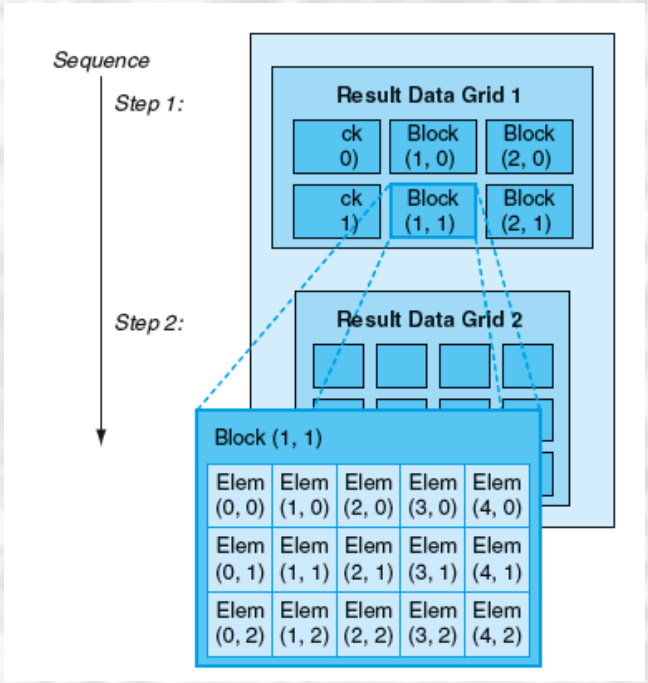
Graphics Processor Units

- Programming – CUDA
  - C like code is written in serial fashion
  - Code calls parallel Kernals
    - Kernals are parallelizable functions, blocks, programs
  - Kernels execute across parallel processors as a set of threads
  - Threads are organized into Thread Blocks
    - Sets of concurrent threads that can work together
      - Through synchronization or shared private memory
  - Independent thread blocks are grouped together as a Grid
    - Can be executed in parallel

Parallel Processing

Graphics Processor Units

- Programming – CUDA
  - 3 Abstractions
    - Thread Groups
    - Shared Memories
    - Barrier Synchronization
  - Kernals
    - Functions or entire programs whose operations can be done in parallel
    - Specifies # of Blocks and # of threads/block in a grid
  - Blocks ← blockIDx
  - Threads ← threadIDz



Parallel Processing

Graphics Processor Units

- Programming – CUDA

Computing y = ax + y with a serial loop:

```
void saxpy_serial(int n, float alpha, float *x, float *y)
{
     for(int i = 0; i<n; ++i)
          y[i] = alpha*x[i] + y[i];
}
// Invoke serial SAXPY kernel
saxpy_serial(n, 2.0, x, y);
```

Parallel Processing

Graphics Processor Units

- Programming – CUDA

Indicates the following is a kernel

Computing y = ax + y in parallel using CUDA:

```
__global__
void saxpy_parallel(int n, float alpha, float *x, float *y)
{
     int i = blockIdx.x*blockDim.x + threadIdx.x;

     if( i<n ) y[i] = alpha*x[i] + y[i];
}

// Invoke parallel SAXPY kernel (256 threads per block)
int nblocks = (n + 255) / 256;
saxpy_parallel<<<nblocks, 256>>>(n, 2.0, x, y);
```

Code segment that can be run in parallel

Invoke the kernel

- All parallelization is handled by the processor
  - Thread management is handled by hardware

Parallel Processing

Graphics Processor Units

- Programming – CUDA
  - Synchronization
    - A synchronization barrier can be created - _syncthreads_
    - No thread can pass the barrier until all threads reach the barrier
    - Applies to threads with-in a block
  - Thread blocks cannot be directly synchronized
    - Blocks must be able to operate independently
    - Can synchronize by using atomic memory processes
    - Can synchronize grids

Parallel Processing

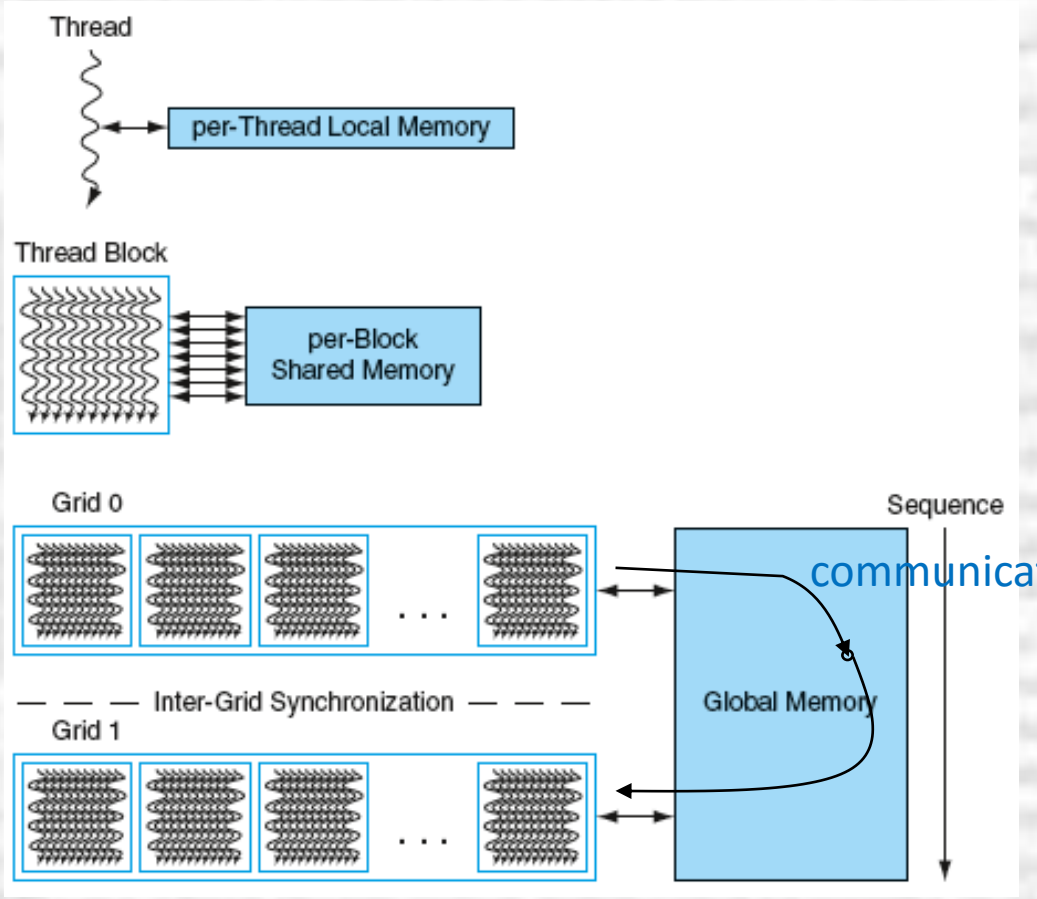
Graphics Processor Units

- GPU Memory Considerations
  - Each thread has its own context
    - PC, registers
  - Each thread has its own private local memory
    - For anything that does not fit in its registers – incl. stack
  - Each thread block has a shared memory
    - Visible to all threads in the block
    - Exists as long as the block exists
    - On chip ram
  - All threads have access to global memory
    - Grids pass data via global memory
    - DRAM

Parallel Processing

Graphics Processor Units

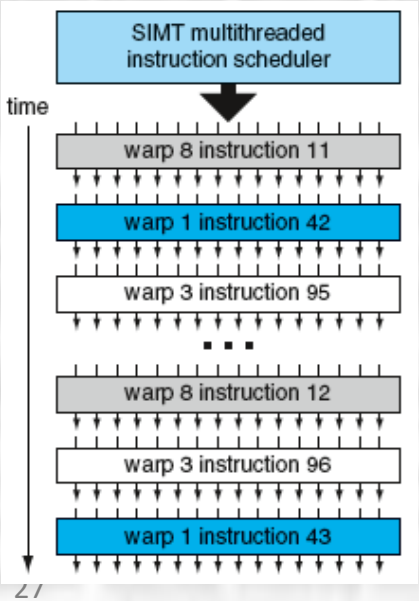
- GPU Memory Considerations



Parallel Processing

Graphics Processor Units

- Using the GPU as a SIMT Multi-processor
  - Architecture already supports multiple kernel (programs) running via multiple threads
  - Define a Warp to be a set of threads running the same instruction
  - Tesla
    - 32 threads/wrap
    - Hardware managed

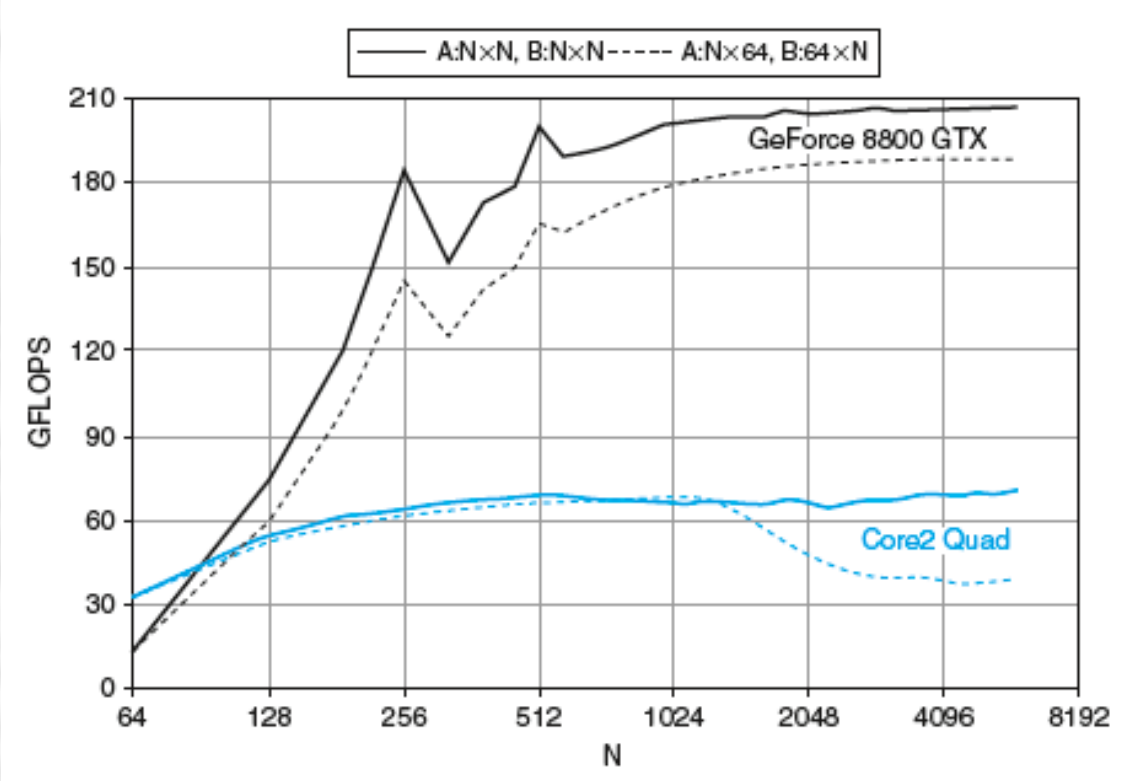


Parallel Processing

Graphics Processor Units

- Performance
  - Matrix multiplication

  - GPU: 1.35GHz
  - CPU: 2.4GHz

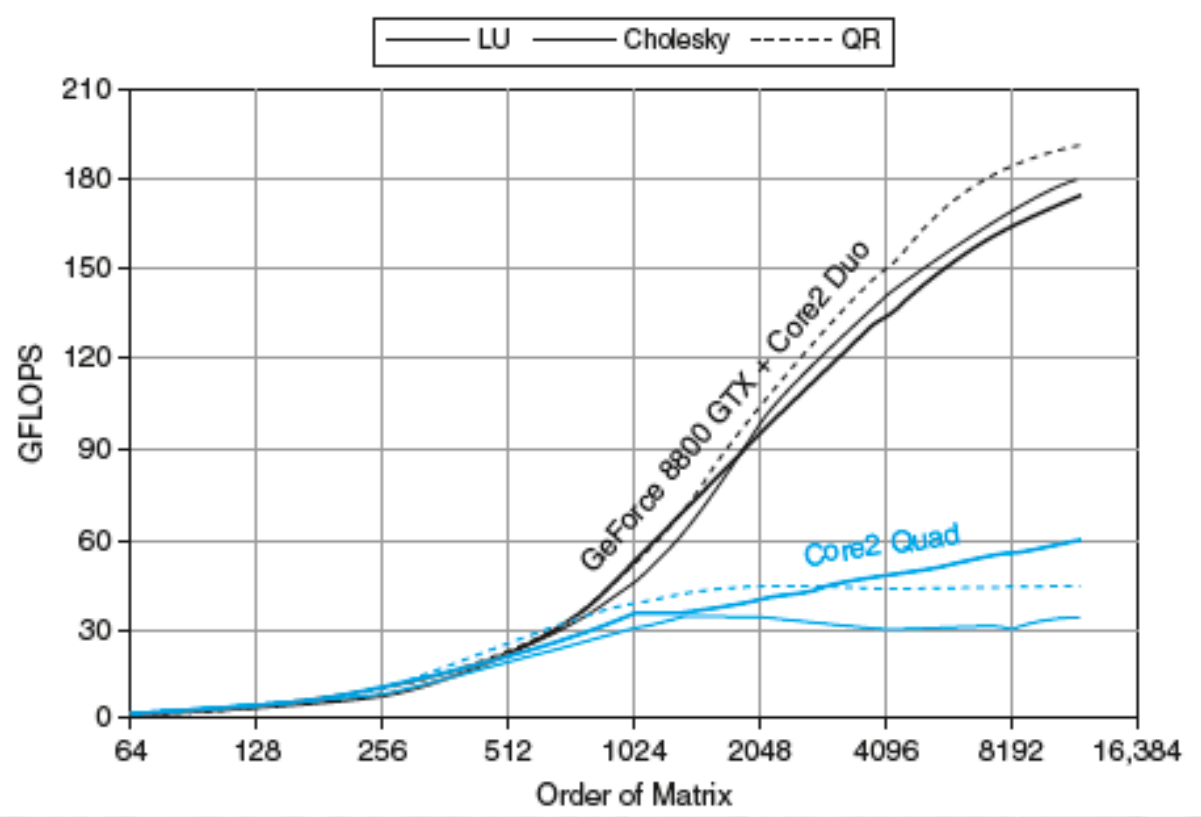


Parallel Processing

Graphics Processor Units

- Performance
  - Matrix factorization

  - GPU: 1.35GHz
  - CPU: 2.4GHz



Parallel Processing

Graphics Processor Units

- Nvidia Kepler
  - 192 cores

Parallel Processing

Graphics Processor Units

- Huang Video