On the developments of NVIDIA GPUs and CUDA

Dr. Tuan-Tu Tran
trant@uni-mainz.de

GPU: Graphics Processing Unit

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

- **Defination**: “a GPU is a single chip processor with integrated transform, lighting, triangle setup/clipping, and rendering engines that is capable of processing a minimum of 10 million polygons per second”
- 31.08.1999: The world's first GPU, NVIDIA Geforce 256

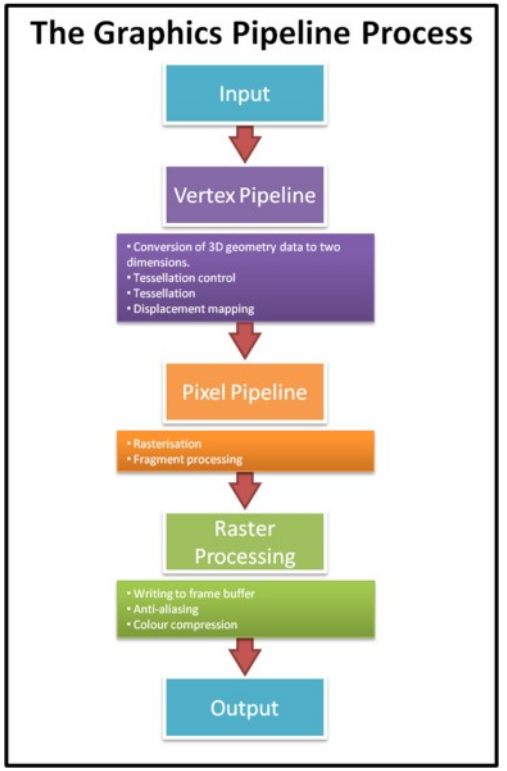
(http://www.nvidia.com/page/geforce256.html)

(*Image credit: Anandtech)*

What does a GPU do ?

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

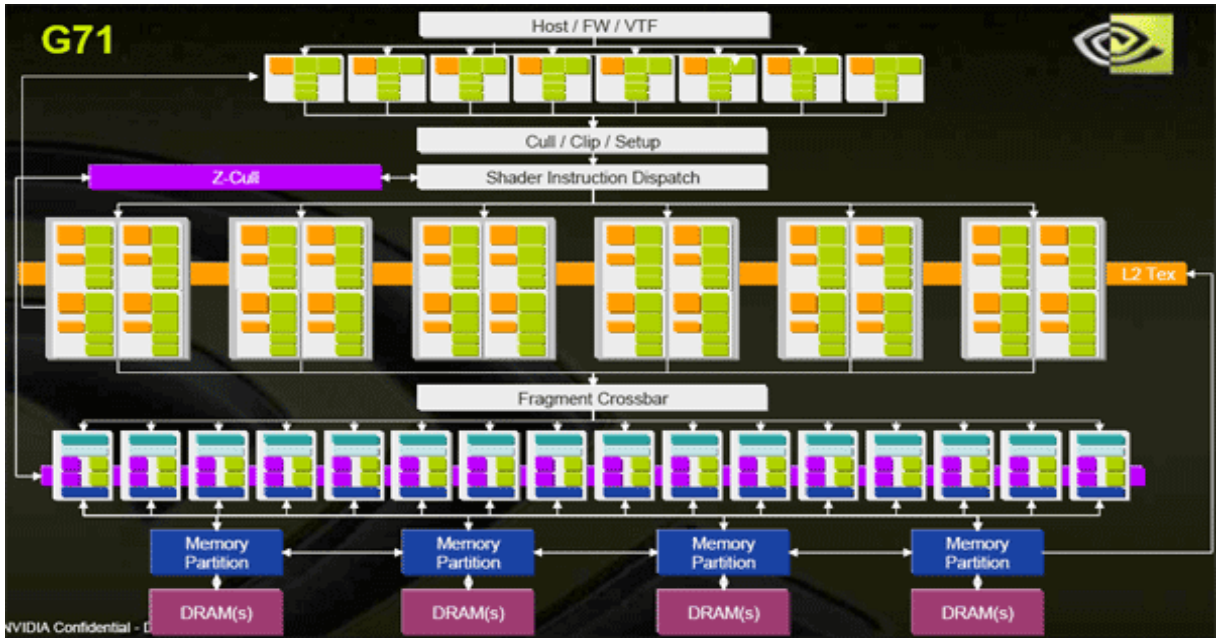
- Graphics Pipeline:
- “Shader” = “Graphics Function”:
  - Vertex Shader: for vertex processing
  - Pixel Shader: for pixel fragment processing



(*Image credit: pcgamingwiki)*

Discrete Architecture

JOHANNES GUTENBERG UNIVERSITÄT MAINZ



(NVIDIA GT 7900 diagram, image credit: NVIDIA)

Vertex Shaders

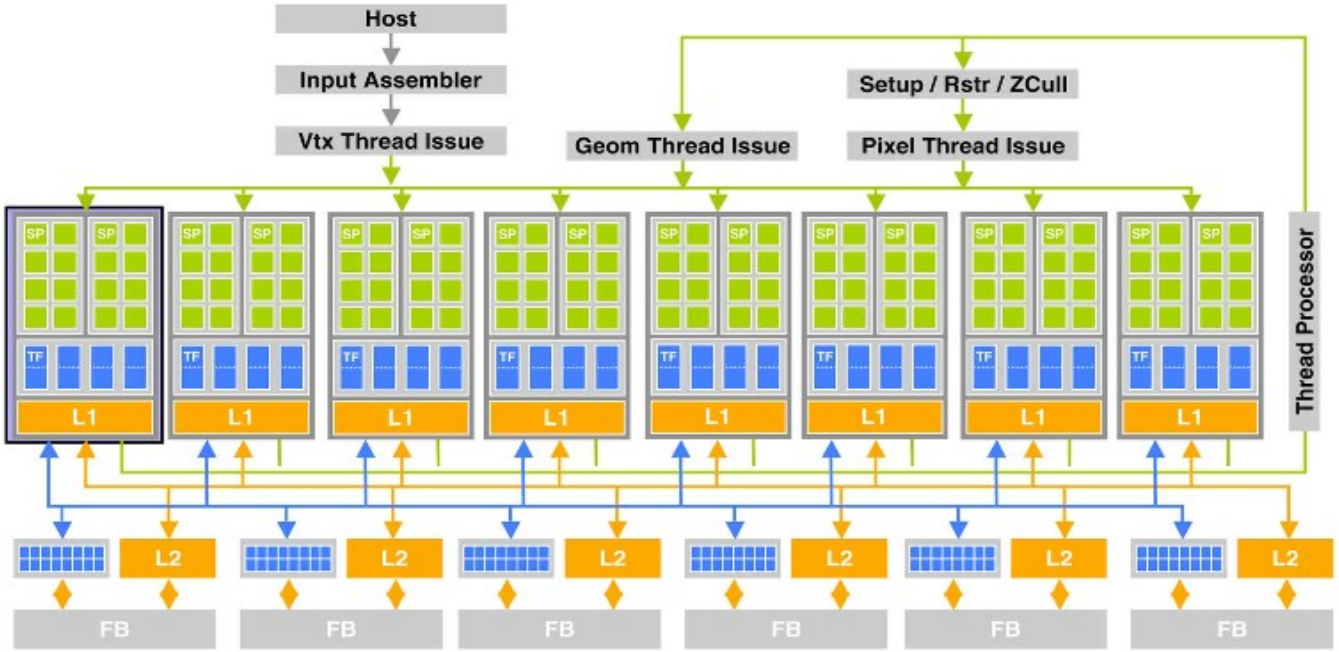
Pixel Shaders

Raster Operations

- 1st Generation: Configuable Shaders
- 2nd Generation: Programmable Shaders

Unified Architecture

JOHANNES GUTENBERG UNIVERSITÄT MAINZ



(NVIDIA GT 8800 diagram, image credit: NVIDIA)

November 2006

- Hardware: NVIDIA 8800 GTX
- Parallel Computing Platform and Programming Model: Compute Unified Device Architecture (**CUDA**)

GPGPU

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

- Unified Architecture => General Purpose Computation on GPU (GPGPU)
  - GPU for not only graphics processing
  - Application from any domain have been mapped to GPU
  - Plenty contributions from NVIDIA

More details: http://gpgpu.org/

NVIDIA GPU and CUDA road map

- Compute Capability: 1.x, 2.x, 3.x
- Architecture: Tesla, Fermi, Kepler, Maxwell
- CUDA 1.0 … CUDA 5.0, CUDA 5.5, CUDA 6.0

Compute Capability

- Benefit from hardware developments
- Number of registers, cores … => threadsPerBlock, blockPerGrids, etc
- Memory size: => cache, shared memory, …
- Advanced architecture => special functions, …

More details:

http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#compute-capabilities

An example of special function: Atomic function

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Requirement: Parallel update value of a element => sum reduction

- From 1.1: atomic function (atomicAdd, …)
- Hardware support for 2.x:

> While G80 had atomic instructions, by allowing these atomic values to be placed in the shared L2 cache in Fermi, atomic instructions can be 5X to 20X faster. Once again, Fermi joins its mainstream brethren in providing a fast version of an important primitive. Like caches and the unified address space, fast atomic instructions increase the types of programs that can run well on GPUs.

http://www.nvidia.com/content/PDF/fermi_white_papers/D.Patterson_Top10InnovationsInNVIDIAFermi.pdf

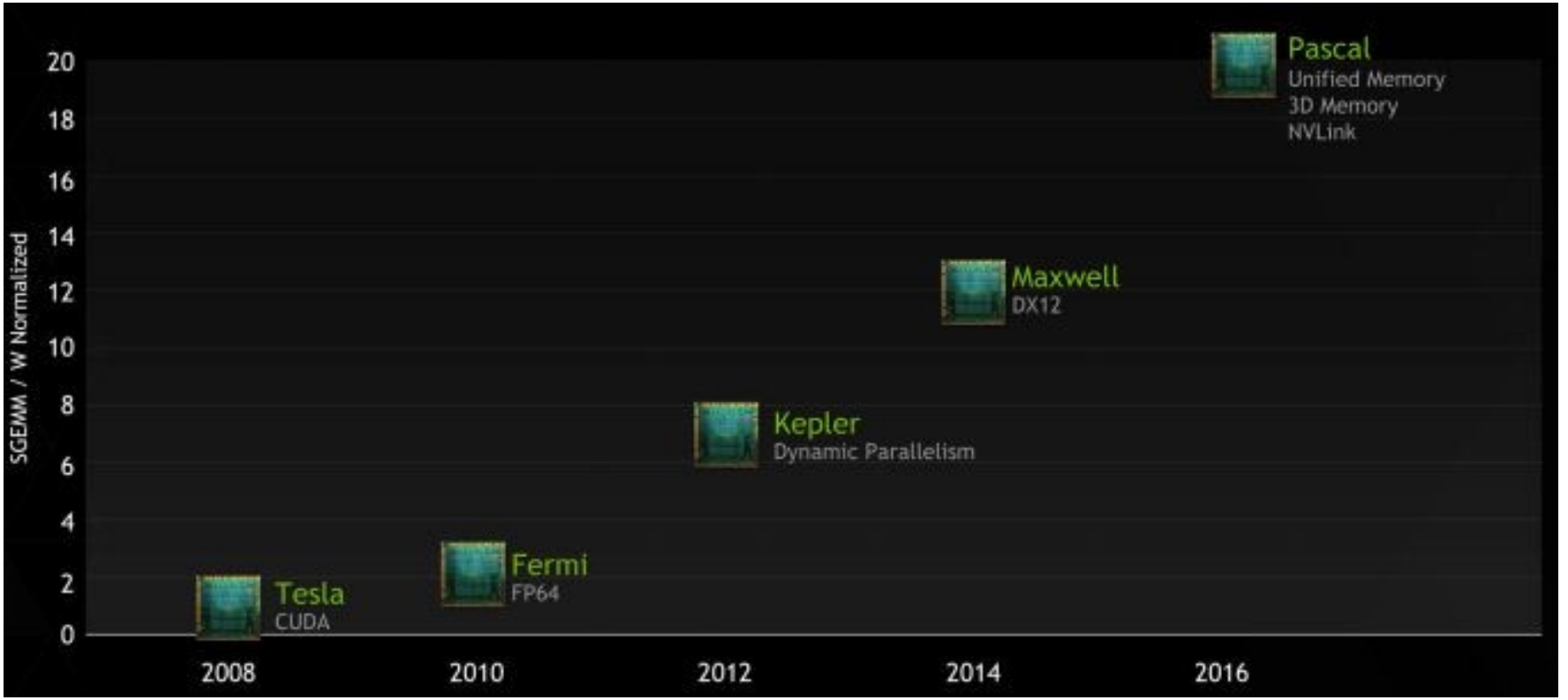
- Hardware support for 3.x:

> Throughput of global memory atomic operations on Kepler GK110 is substantially improved compared to the Fermi generation. Atomic operation throughput to a common global memory address is improved by 9x to one operation per clock. Atomic operation throughput to independent global addresses is also significantly accelerated, and logic to handle address conflicts has been made more efficient. Atomic operations can often be processed at rates similar to global load operations. This speed increase makes atomics fast enough to use frequently within kernel inner loops, eliminating the separate reduction passes that were previously required by some algorithms to consolidate results. Kepler GK110 also expands the native support for 64-bit atomic operations in global memory. In addition to atomicAdd, atomicCAS, and atomicExch (which were also supported by Fermi and Kepler GK104), GK110 supports the following:

http://www.nvidia.de/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf

NVIDIA GPU Road Map

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

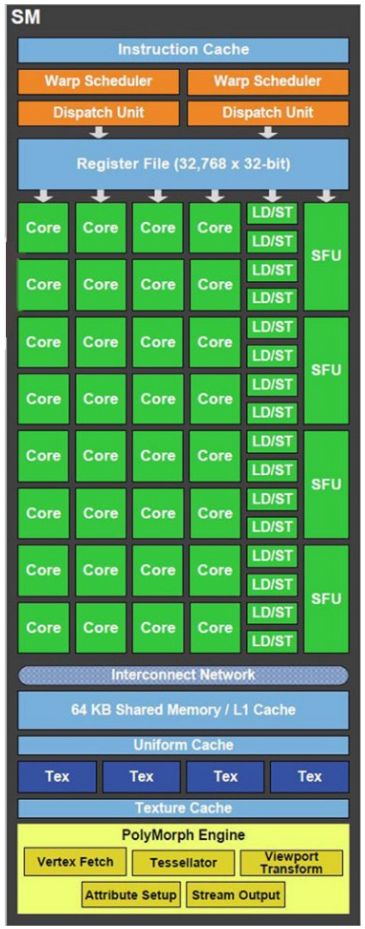


http://www.anandtech.com/show/7900/nvidia-updates-gpu-roadmap-unveils-pascal-architecture-for-2016

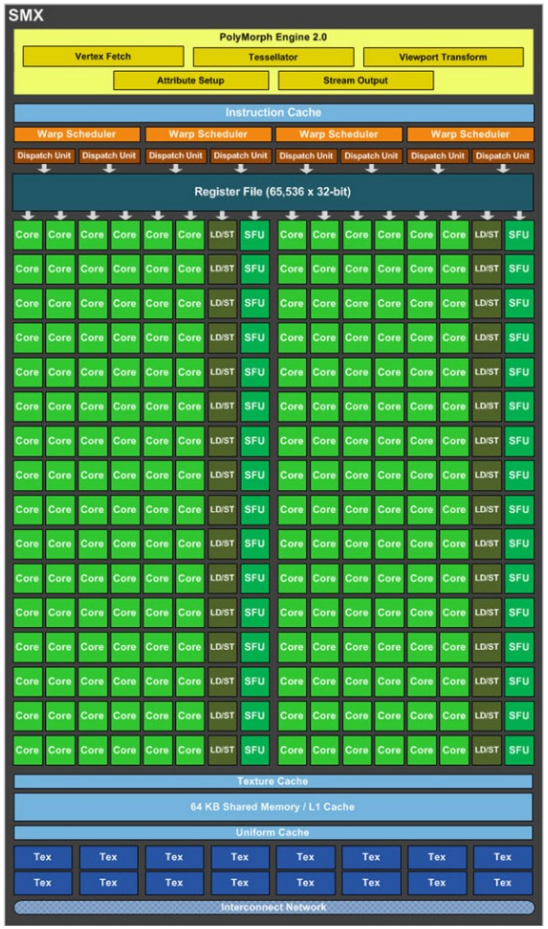
SM/SMX/SMM

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

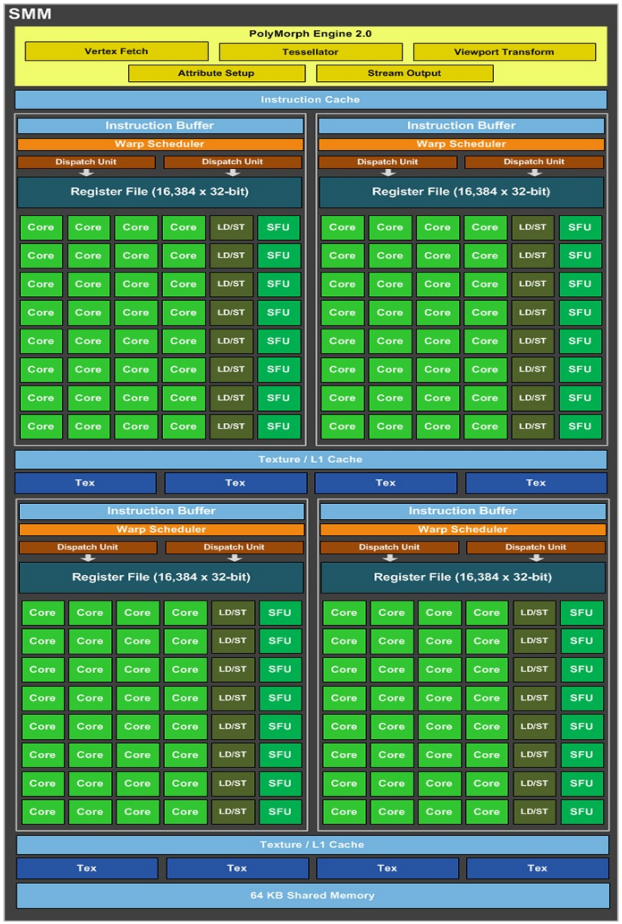
Fermi



Kepler



Maxwell

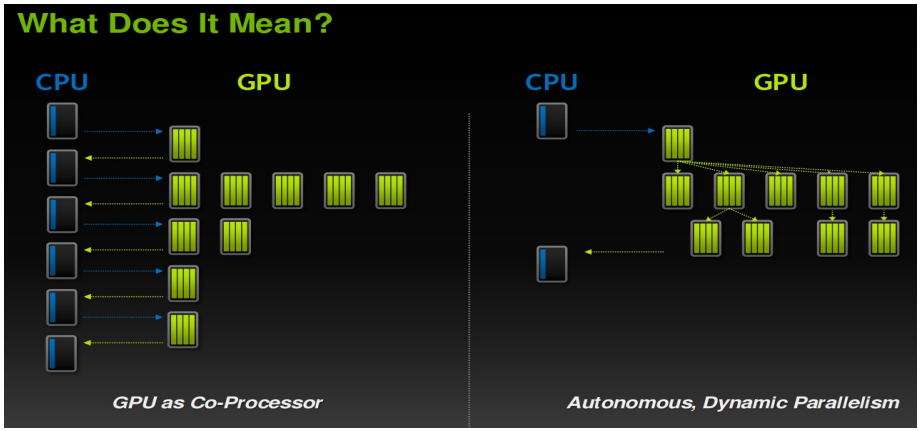


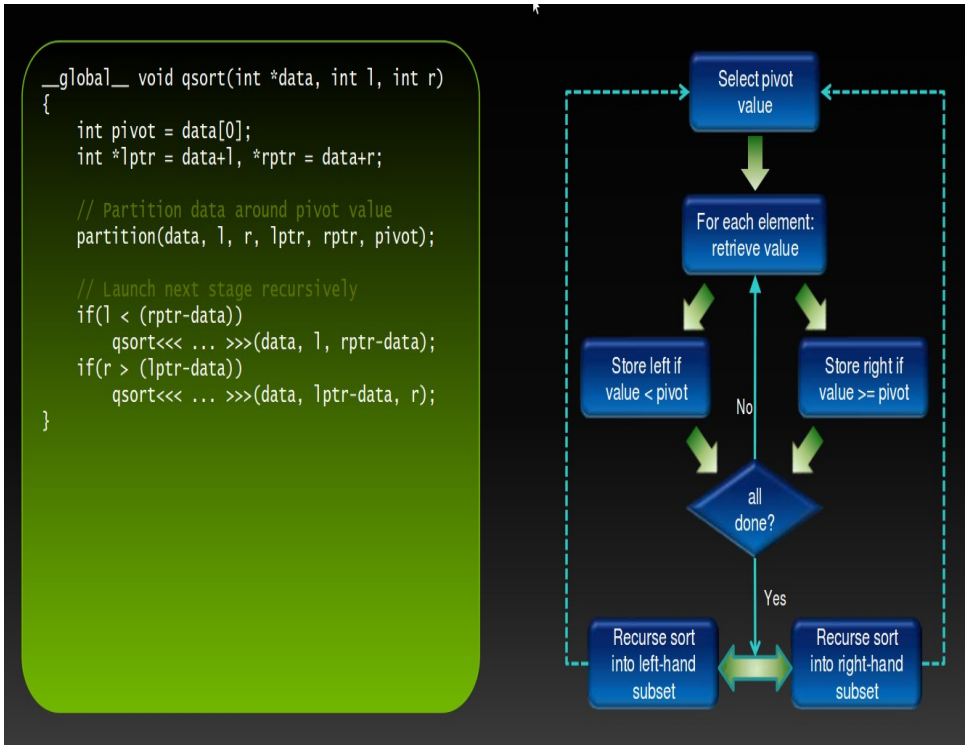
(Images credit: NVIDIA)

Dynamic Parallelism

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

(From CUDA 5.0 and 3.x)



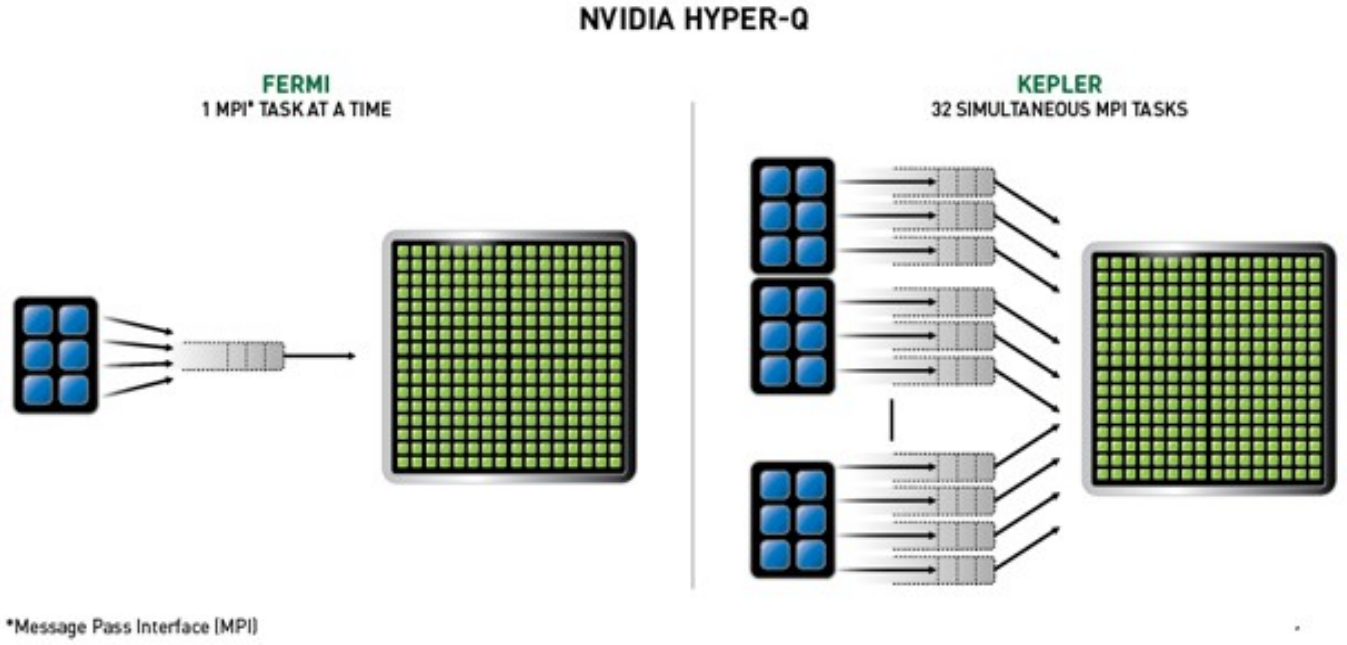


http://on-demand.gputechconf.com/gtc/2012/presentations/S0338-GTC2012-CUDA-Programming-Model.pdf

Hyper Q

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

(From 3.x)



http://docs.nvidia.com/cuda/samples/6_Advanced/simpleHyperQ/doc/HyperQ.pdf

Private Data Exchange

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

(From CUDA 5.0 and 3.x)

Exchange private variable between threads in the same warps

```
float __shfl_[down/up](
     float var,           // Variable to read from source thread
     unsigned int delta,  // # of threads you want to shuffle down/up
     int width=warpSize   // Division of warp into segments of size width
);
```

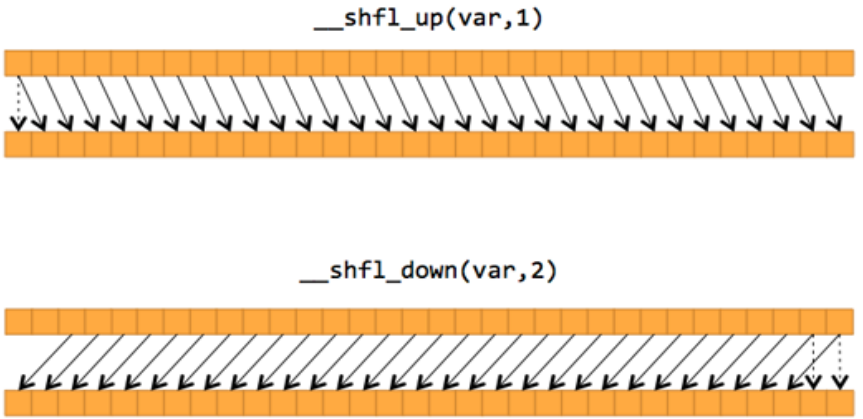


Figure 3 – Shuffle up and down instructions illustrated

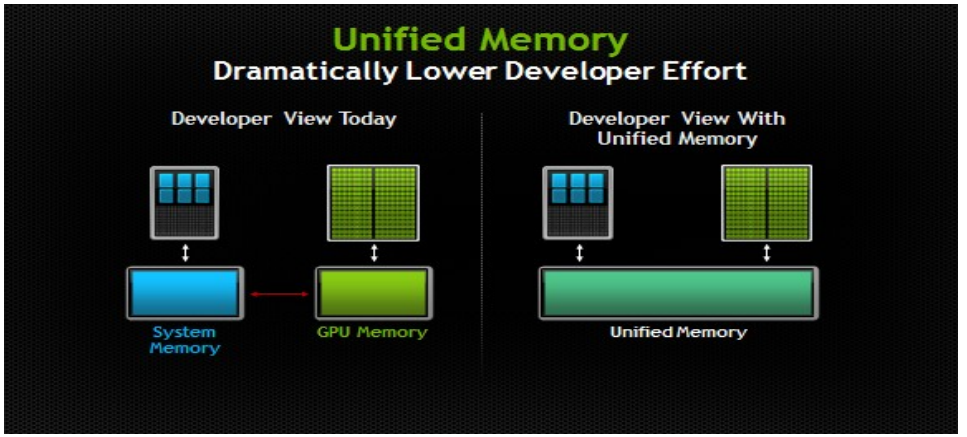
http://acceleware.com/blog/keplers-shuffle-instruction

Unified memory between GPU and host

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

(From CUDA 6.0 and 3.x)

http://devblogs.nvidia.com/parallelforall/unified-memory-in-cuda-6/



CPU Code

```
void sortfile(FILE *fp, int N) {
  char *data;
  data = (char *)malloc(N);

  fread(data, 1, N, fp);

  qsort(data, N, 1, compare);


  use_data(data);

  free(data);
}
```

CUDA 6 Code with Unified Memory

```
void sortfile(FILE *fp, int N) {
  char *data;
  cudaMallocManaged(&data, N);

  fread(data, 1, N, fp);

  qsort<<<...>>>(data,N,1,compare);
  cudaDeviceSynchronize();

  use_data(data);

  cudaFree(data);
}
```

… 2016

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

- The Pascal Architecture
  - Stack memory: GPU size, memory size, speed, …
  - NVLink: data exchange (CPU – GPU or GPU – GPU)
- More details:

https://devblogs.nvidia.com/parallelforall/nvlink-pascal-stacked-memory-feeding-appetite-big-data/

Conclusions

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

- NVIDIA GPUs and CUDA:
  - still being developed: rapidly and innovatively
- Master the fundamentals
  - Parallel programming:
    - Data structure
    - Algorithms
    - Programming language
  - Processor architecture
    - Basic feature of GPU
    - Advanced feature
- Keep updating

JOHANNES GUTENBERG
UNIVERSITÄT MAINZ

Question ?