CUDA: Programming at Light-Speed

By: Steven Beaudoin
Overview

• Evolution of computing
• What is CUDA?
• Details of CUDA operating procedures
• Structure of CUDA programs
• Code examples
• Field areas of interest
• Restrictions and problems with CUDA

http://www.mmorpg.com/gamelist.cfm/loadNews/19317/Nvidia-SLI-Profile-Added
Evolution of Computing

- Since 1980, the primary means of improving computing speed has been increasing clock speed
  - Limits are being reached - transistor size & heat concerns

- In 2005, computers began to ship with multiple cores
  - Again, limited number of cores due to size

- 2006-2007 → idea to use GPU for doing calculations
  - Specialized for mathematical computations
History of the Programmable Pipeline

- Fixed function pipeline (early 1990s)
- Move toward more programmable stages
- General purpose computing on GPUs (2000)
  - GPU used for floating point calculations
  - Had to use OpenGL or DirectX language
  - Disguise computations as graphics problem
CUDA

• November 2006, GeForce 8800 GTX
• First to use Compute Unified Device Architecture

• CUDA is a fully programmable graphics pipeline
  • Supports IEEE floating point precision

• Shortly after, CUDA C released by NVIDIA
  • Extension of C programming language
• All NVIDIA cards released after 8800 have CUDA capability
Hardware Operations

CPU - Host

Main C code

Main C code

Main C code

GPU - Kernel

Parallel Processing

Parallel Processing

Parallel Processing

Parallel Processing

Parallel Processing

Parallel Processing

Parallel Processing

Parallel Processing
Software Operations

• CUDA C works by compiling functions on the GPU
  • Allocate area in GPU memory
__global__ void add( int first, int second, int *sum ){  
  *sum = first + second;
}

int main(){
  int sum;
  int *dev_sum;

  add<<<1,1>>>(first, second, dev_sum);

  cout << "1 + 2 = " << sum;
  return 0;
}
Computationally Intense Problems

Vector A + Vector B = Vector C

\[
\begin{array}{c}
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
[B, B, B, B, B, B, B, B, B, B, B, B,... B] \\
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
\end{array}
\]

For a vector of length N, need N additions
--For loop with N iterations--
__global__ void add( int *first, int *second, int *sum ){
    int i = blockIdx.x;
    sum[i] = first[i] + second[i];
}

#define N 5 //This can be any integer value

int main(){
    //Declare variables
    cudaMemcpy( (void**)&first, N * sizeof(int) ); //Memory Allocation
    add<<<N,1>>>(first,second,sum)
}

int main()
Applications

Medical Imaging:
• TechniScan has created a 3D ultrasound imaging device
• Practical problems – 35 GB of data
• CUDA programming has allowed for images to be created within 20 minutes

Environmental Science:
• Cleaning detergents use particles called surfactants to adhere to dirt and wash away
• Simulations run with interactions between particles by Temple University
• Two GPUs = 128 CPU cores of the Cray XT3 or 1024 CPUs of IBM BlueGene/L

Applications

Fluid Dynamics:
• Air and fluid movement representation is computationally intensive
• Cambridge University used a supercomputer to create simulations
• Small GPU cluster outperformed the supercomputer

http://wn.com/SPH_3D_simulation
Restrictions and Drawbacks

Still has some issues with IEEE double floating point precision
• Single point precision = CPU programming

Requires many transfers of data to and from different memory locations
Conclusions

• The move toward programmable pipelines opened up the possibility of GPU processing

• GPU processing offers advantages over CPU processing
  • Parallel processing of mathematical computations
  • Blocks and threads
• Executes off C code with additional functionality

• Useful in many applications
References

Questions?

http://www.controlinc.com/faq.htm
EE Concepts

Computer Architecture
• Reading/Writing to memory
• Processors
• Clock speeds

Computer Science & Programming
• Programming in C++
• Knowledge of functions
CUDA Speed

- For single operation processes = slower
  - Read/Write

- For computationally intense programs
  - Exponentially faster

- Block and Thread limit:
  - 65,535

Graph shows how much faster CUDA architecture was over CPU processing for matrices. [4]
CUDA Equivalents

AMD
• ATI Stream is approximately equivalent to CUDA
• More optimized for vector operations
• CUDA = scalar operations
• Written in OpenCL