History....

Stream Computing on Graphics Hardware

Ian Buck

GPGPU in 2004
recent trends

GFLOPS

(observed peak)

- NVIDIA NV30, 35, 40
- ATI R300, 360, 420
- Pentium 4

July 01  Jan 02  July 02  Jan 03  July 03  Jan 04
translating transistors into performance

- 1.8x increase of transistors
- 20% decrease in clock rate
- 6.6x GFLOP speedup
Early GPGPU (2002)

• Ray Tracing on Programmable Graphics Hardware
  Purcell et al.

• PDEs in Graphics Hardware
  Strzodka, Rumpf

• Fast Matrix Multiplies using Graphics Hardware
  Larsen, McAllister

• Using Modern Graphics Architectures for
  General-Purpose Computing: A Framework and Analysis.
  Thompson et al.
Programming model challenge

- Demonstrate GPU performance
- PHD computer graphics to do this
- Financial companies hiring game programmers

“GPU as a processor”
C with streams

- streams
  - collection of records requiring similar computation
    - particle positions, voxels, FEM cell, ...
      ```c
      Ray r<200>;
      float3 velocityfield<100,100,100>;
      ```
  - similar to arrays, but...
    - index operations disallowed: `position[i]`
    - read/write stream operators:
      ```c
      streamRead (positions, p_ptr);
      streamWrite (velocityfield, v_ptr);
      ```
Building GPU Computing Ecosystem

- Convince the world to program an entirely new kind of processor
- Tradeoffs between functional vs. performance requirements
- Deliver HPC feature parity
- Seed larger ecosystem with foundational components
CUDA: C on the GPU

- A simple, explicit programming language solution
- Extend only where necessary
  ```
  __global__ void KernelFunc(...);
  __shared__ int SharedVar;
  KernelFunc<<< 500, 128 >>>(...);
  ```
- Explicit GPU memory allocation
  - `cudaMalloc()`, `cudaFree()`
- Memory copy from host to device, etc.
  - `cudaMemcpy()`, `cudaMemcpy2D()`, ...
CUDA: Threading in Data Parallel

- Threading in a data parallel world
  - Operations drive execution, not data
- Users simply given thread id
  - They decide what thread access which data element
  - One thread = single data element or block or variable or nothing….
  - No need for accessors, views, or built-ins
- Flexibility
  - Not requiring the data layout to force the algorithm
  - Blocking computation for the memory hierarchy (shared)
  - Think about the algorithm, not the data
Divergence in Parallel Computing

- Removing divergence pain from parallel programming

SIMD Pain
- User required to SIMD-ify
- User suffers when computation goes divergent

GPUs: Decouple execution width from programming model
- Threads can diverge freely
- Inefficiency only when granularity exceeds native machine width
- Hardware managed
- Managing divergence becomes performance optimization
- Scalable
CUDA By the Numbers:

- 415,000,000 CUDA-Capable GPUs
- 1,500,000 Toolkit Downloads
- 120,000 Active Developers
- 580 Universities Teaching CUDA
3 Ways to Accelerate Applications

- Libraries: “Drop-in” Acceleration
- OpenACC Directives: Easily Accelerate Applications
- Programming Languages: Maximum Flexibility

Applications
OpenACC Directives

Simple Compiler hints

Compiler Parallelizes code

Works on many-core GPUs & multicore CPUs

Now Shipping CAPS, Cray, PGI

OpenACC 2.0 Prelim Spec

www.openacc.org

Your original Fortran or C code

Program myscience
... serial code ...
!$acc kernels
do k = 1,n1
  do i = 1,n2
    ... parallel code ...
  enddo
endo
do
do
!$acc end kernels
... End Program myscience

CPU

GPU
S3D Accelerated with OpenACC

Total Lines of Code Changed

CUDA: 3%
OpenACC: 0.4%

Total Performance

CUDA = OpenACC
GPU Libraries: Plug In & Play

FFT's up to 10x Faster than MKL
1D used in audio processing and as a foundation for 2D and 3D FFTs

cuBLAS Level 3 Performance
Up to ~800GFLOPS and ~17x speedup over MKL

cuRAND Performance
cuRAND 64-bit Scrambled Sobol' 8x faster than MKL 32-bit plain Sobol'

cuSPARSE is up to 6x Faster than MKL
Sparse Matrix x Dense Vector

Thrust
Parallel Algorithms

QUDA
Lattice QCD
Rapid Parallel C++ Development

- Resembles C++ STL
- Open source
- High-level interface
  - Enhances developer productivity
  - Enables performance portability between GPUs and multicore CPUs
- Flexible
  - CUDA, OpenMP, and TBB backends
  - Extensible and customizable
  - Integrates with existing software

```c++
// generate 32M random numbers on host
thrust::host_vector<int> h_vec(32 << 20);
thrust::generate(h_vec.begin(), h_vec.end(), rand);

// transfer data to device (GPU)
thrust::device_vector<int> d_vec = h_vec;

// sort data on device
thrust::sort(d_vec.begin(), d_vec.end());

// transfer data back to host
thrust::copy(d_vec.begin(), d_vec.end(), h_vec.begin());
```

The CUDA Platform is a foundation that supports a diverse parallel computing ecosystem.
Open Source CUDA LLVM Compiler

Developers want to build front-ends for Java, Python, R, DSLs

Target other processors like ARM, FPGA, GPUs, x86
Diversity of Programming Languages

http://www.ohloh.net
Domain-specific Languages

MATLAB

R Statistical Computing Language

Liszt
A DSL for solving mesh-based PDEs
Building blocks for Exascale

**GPU Direct**

- CPU
- GPU
- InfiniBand

**UMA**

- System Memory
- GPU0 Memory
- GPU1 Memory

**Dynamic Parallelism**

**Atomic Ops**

- Atomic operations for thread-to-thread communication

```c
atom(.space).op.type d, [a], b;
atom(.space).op.type d, [a], b, c;
.space = {.global, .shared};
.op = {.and, .or, .xor, // .b32 only
   .cas, .exch, // .b32, .b64
   .add, // .u32, .s32, .f32, .u64
   .inc, .dec, // .u32 only
   .min, .max}; // .u32, .s32, .f32
.type = {.b32, .b64,
   .u32, .u64,
   .s32,
   .f32};
```
Programming model evolution

Today
Eliminating Barriers to Porting
Optimizing locality and computation
Task, Thread & Data Parallelism
Hybrid operating system Enablement
Parallel Compiler Foundation Enablement
Ubiquitous parallel programming
Domain Specific Languages
Power Aware Programming
Education & Research

Domain Science

Languages and Compilers

Heterogeneous Architectures

Computer Science

CUDA on ARM
Try out CUDA 5

CUDA 5 available today!
- [www.nvidia.com/getcuda](http://www.nvidia.com/getcuda)
- docs.nvidia.com
- developer.nvidia.com

Become a registered developer

File Bugs

Try out early versions of CUDA

Provide feedback to NVIDIA via CUDA Forums and
- [CUDA_RegDev@nvidia.com](mailto:CUDA_RegDev@nvidia.com)