General Purpose GPU Programming

A Short Introduction

Joseph Kehoe[1]

[1]Department of Computing and Networking
Institute of Technology Carlow

CDD101, 2017

Table of Contents

Table of Contents

Definition

GPU: Graphics Processing Unit

- A Integrated Circuit dedicated to graphics processing
- Uses an SIMD model (as this best suits Graphics programming)
  - Many "simple" cores
  - Cores contain many ALUs
- SIMT model now employed in GPUs

Table of Contents

Architecture

- Different manufacturers have different architectures but in general terms:
- Each GPU has multiple Cores (simpler than CPU cores)
- Each Core has a number of SIMD units
- Each core is an SIMD procressor

Table of Contents

Memory Model

- GPUs tend to be seperate from CPU
- There is a cost to getting data from CPU to GPU
  - GPU and CPU have seperate memory
  - must offload data to GPU before computation and then move result back to CPU at end
  - up to 10m instructions per offload
- Allowing GPU and CPU to share the same memory would change dynamics of GPU usage

Memory Model

Four types of memory in GPU

Private Memory Available only to single work item/SIMD unit

Local Memory Local to workgroup/core

Global Memory Shared between entire GPU

Constant Memory Read only memory

Much effort is put into fitting data to these memory sizes

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster

Table of Contents

API's

From low level to high:

- OpenCL, CUDA
- ArrayFire, Thrust, C++ AMP
- OpenACC, OpenMP

APIs are in flux and changing (improving) year to year

5 Step Process

- Create a context within which the **kernel** will run with a command queue
- Compile the kernel
- Create buffers for inpuit and output data
- Enqueue a command that executes the kernel once for each work item
- Retrieve the results

There can be a lot of boiler plate code but the kernel is the heart of the computation

Table of Contents

Saxpy

```
//kernel
__global__
void saxpy(int n, float a, float *x, float *y)
{
  int i = blockIdx.x*blockDim.x + threadIdx.x;
  if (i < n) y[i] = a*x[i] + y[i];
}

int main(void)
{
int N = 1<<20;
float *x, *y, *d_x, *d_y;
x = (float*)malloc(N*sizeof(float));
y = (float*)malloc(N*sizeof(float));
 cudaMalloc(&d_x, N*sizeof(float));
 cudaMalloc(&d_y, N*sizeof(float));
```

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster

Saxpy

```
//later that day...
 cudaMemcpy(d_x, x, N*sizeof(float),
                cudaMemcpyHostToDevice);
 cudaMemcpy(d_y, y, N*sizeof(float),
                cudaMemcpyHostToDevice);
 // Perform SAXPY on 1M elements
 saxpy<<<(N+255)/256, 256>>>(N, 2.0f, d_x, d_y);
 cudaMemcpy(y, d_y, N*sizeof(float),
                cudaMemcpyDeviceToHost);
 cudaFree(d_x);
 cudaFree(d_y);
 free(x);
 free(y);
}
```

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster

Matrix Multiplication Serial

```
//serial
int a[M][P], b[P][N], c[M][N];
for(int i=0;i<M;++i){
  for(int k=0;k<N;++k){
    int sum=0;
    for(int f=0;f<P;++f){
      sum+=a[i][f]*b[f][k];
    }
    c[i][k]=sum;
  }
}
```

Matrix Multiplication Kernel- OpenCL

```
__kernel void matrixmult(uint widthA,
                         __global const int* inA,
                         __global const int* inB,
                         __global int *out){
    int i=get_global_id(0);
    int j=get_global_id(1);
    int outWidth=get_global_size(0);
    int outHeight=get_global_size(1);
    int sum=0;
    for(int k=0;k<widthA;++k){
      sum+=a[i*widthA+k]*b[k*outWidth+j];
    }
    out[i*outWidth+i]=sum;
}
```

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster

Reduction Kernel- OpenCL

```
__kernel void minum(__global const int* inA,
                    __global int sum,
                    __local int* tmp){
                int i=get_global_id(0);
                int n=get_global_size(0);
                tmp[i]=in[i];
                barrier(CLK_LOCAL_MEM_FENCE);
                for(int k=n/2;k>0;k=k/2){
                   if(i<k){
                     tmp[i] +=tmp[i+k];
                   }
                   barrier(CLK_LOCAL_MEM_FENCE);
                }
                if (i==0) sum=tmp[0];
}
```

Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster