Lab 9

CMPUT 398

Objective

Implement a program that runs Radix sort on the GPU. The numbers given for sorting are integers from 0 to 65534

This lab will be submitted as one zipped file through eclass. Details for submission are at the end of the lab.

Instructions

There are instructions on how to implement Radix sort at the following link:

http://http.developer.nvidia.com/GPUGems3/gpugems3_ch39.html

If you want to read a more detailed version of Radix sort check out the following link:

http://mgarland.org/files/papers/nvr-2008-001.pdf

If you are unsure what Radix sort is then watch the following videos:

Radix Sort Algorithm
https://www.youtube.com/watch?v=2109VzXVTus

Radix Sort Part 1 - Intro to Parallel Programming
https://www.youtube.com/watch?v=dPwAA7j-8o4

Radix Sort Part 2 - Intro to Parallel Programming
https://www.youtube.com/watch?v=K-tNYzw8pm0

Radix Sort Part 3 - Intro to Parallel Programming
https://www.youtube.com/watch?v=iS0S7F2U4-o

Instructions - Kernels

You will most likely have to implement at least three kernels; however, you can implement the Radix sort any way you want. The only restrictions are that it needs to be Radix sort written in CUDA and the calculations on the arrays must be performed on the GPU. Therefore, you shouldn't be running the following kernels on the CPU:

1) Calculate if a given bit is 0 or 1

Example given the following array return an array of 0 and 1. Everywhere the first bit is zero, least significant bit, return 1.

000	010	111	101	100

1	1	0	0	1

2) Exclusive scan. This should be the same scan you implemented in the previous lab.

3) Scatter. This is emplaned in the GPU Gems 3 chapter 39.

Important: after launching a kernel make sure you call:

```
wbCheck(cudaDeviceSynchronize());
```

Or

```
cudaDeviceSynchronize();
```

Instructions – Pseudo Code

1. FUNCTION Radix_Sort(output, input, len)
2. Allocate memory if needed
3. LOOP through bits 0 to 15
4. Check bits
5. Exclusive scan the checked bits
6. Scatter the input array into the output
7. END LOOP
8. Free Allocated memory

It is probably a good idea if you scatter the input into the output. Then on every iteration the input becomes the output and the output becomes the input.

Iteration 1:
Input -> Check bits -> Scan
Input -> Scatter -> Output

Iteration 2:
Output -> Check bits -> Scan
Output -> Scatter -> Input

Iteration 3:
Input -> Check bits -> Scan
Input -> Scatter -> Output

Just make sure on the last iteration the sorted array is in the output array.

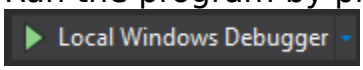
Local Setup Instructions

Steps:

1. Download “Lab9.zip”.
2. Unzip the file.
3. Open the Visual Studios Solution in Visual Studios 2013.
4. Build the project. Note the project has two configurations.
    a. Debug
    b. Submission

But make sure you have the “Submission” configuration selected when you finally submit.

5. Run the program by pressing the following button:



Make sure the “Debug” configuration is selected. Running the program in Visual Studios will run one of the tests located in “Dataset /Test”.

Testing

To run all tests located in “Dataset/ Test”, first build the project with the “Submission” configuration selected. Make sure you see the “Submission” folder and the folder contains the executables.

To run the tests, click on “Testing_Script.bat”. This will take a couple of seconds to run and the terminal should close when finished. The output is saved in “Marks.js”, but to view the calculated grade open “Grade.html” in a browser. If you make changes and rerun the tests, then make sure you reload “Grade.html”. You can double check with the timestamp at the top of the page.

Submission

After you build your project with the “Submission” configuration selected (see above) you will see a new folder called “Submission” in the project directory. If you open the folder you will see your executable. We only want your optimized Reduce kernel and don’t need the Simple Reduce kernel.

Make sure you add your questions as a text file. Finally zip the folder “Submission” and upload the zipped folder to eclass.

If you don’t see this folder or are missing the executable, then one of two things happened. First your build could have failed and you should check for any errors in your build log in Visual Studios. Also you might have had the “Debug” configuration selected (see above). Make sure the “Submission” configuration is selected.

Mark Breakdown

It is important that you **do not** add or remove any print statements or wb functions. We use them for part of your grading so if you make changes the marking script could fail.

Part	Breakdown (%)
Radix Sort	100 %