Lab 2

CMPUT 398

Due date: Next Monday, 23:55.

Objective

The purpose of this lab is to introduce you to the CUDA API by implementing vector addition. You will implement vector addition by writing the GPU kernel code as well as the associated host code.

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

Instructions

Edit the code where the TODOs are specified and perform the following:

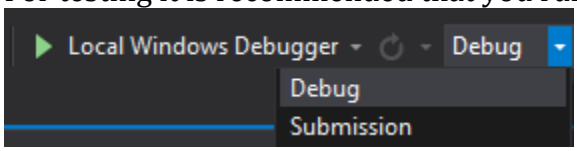
- Allocate device memory
- Copy host memory to device
- Initialize thread block and kernel grid dimensions
- Invoke CUDA kernel
- Copy results from device to host
- Free device memory
- Write the CUDA kernel

Local Setup Instructions

Steps:

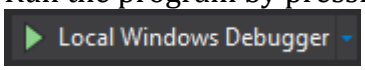
1. Download “Lab2.zip”.
2. Unzip the file.
3. Open the Visual Studios Solution in Visual Studios 2013.
4. Build the project. Note the project has three configurations.
    a. Test
    b. Debug
    c. Submission

For testing it is recommended that you run the “Debug” configuration.



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

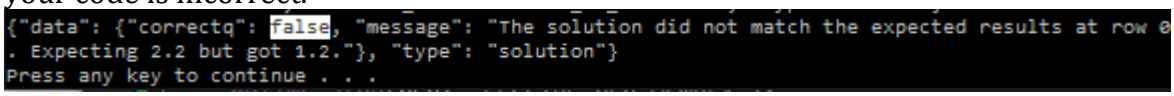
5. Run the program by pressing the following button:



Don’t try to run the program when the “Test” configuration is selected.

Vector Add Testing

1. The “Debug” configuration will show “false” in the last line of the programs output If your code is incorrect.



The outputted vector can be seen in “Dataset\VectorAdd\Test\[0-9]”. For example, “Dataset\VectorAdd\Test\0\myOutput.raw”. The first line is the size of the array. The “Debug” configuration will run the first test “Dataset\VectorAdd\Test\0”.

2. You can also run the program from the Command Prompt (cmd).

VectorAdd -e <expected.raw> -i <intput1.raw>,<input2.raw> \
 -o <output.raw> -t vector

Make sure you are in the directory with the executable before trying to run the command.

3. If you want to run all tests, then you can run the “Test” configuration. To do this simply build the program, with the “Test” configuration selected, and without running the debugger.

Build -> Build Solution

In the Build Output window you should see the following:

```
"Vector Add Testing Test 0..."
COMMAND
Same
```

Note that if the test fails you see “Different or error” instead of “Same”

Alternatively, you can just run the file “Test.bat” provided to you instead.

Using NSIGHT To Analyze Performance Instructions

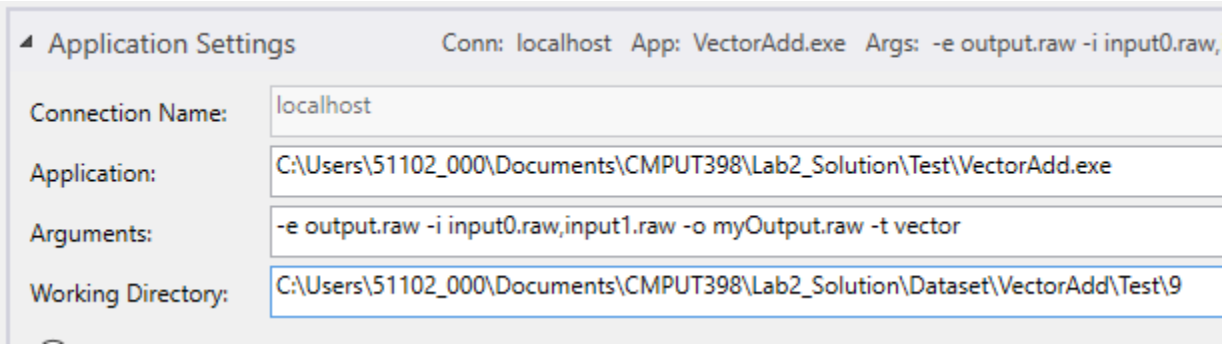
This is complemented by the file “Guide on Debugging, Testing, Submitting and Profiling CUDA Project” on e-class, please read the file if you have not done so.

In Application Setting, enter:

Application: <Path\to\the\project>\Test\VectorAdd.exe

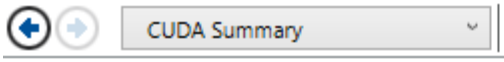
Arguments: -e output.raw -i input0.raw,input1.raw -o myOutput.raw -t vector

Working Directory: <Path\to\the\project>\Dataset\VectorAdd\Test\<Test Number>



Then do the other steps like outlined in the guide file.

Submit an image called “cuda_summary.jpg” contain the screenshot of the CUDA Summary page that you get from running NSIGHT. For example:



CUDA Devices All

	Device ID	Device Name	Contexts	Device %	Host to Device (Bytes)	Device to Host (By
1	[0]	GPU 0 - GeForce GTX 970M	1	0.00 %	32,144	16,072

CUDA Contexts

	Total	No Context	1
CUDA Device ID		-	0
◢ Runtime API Calls Summary | All			
# Calls	17	0	17
# Errors	0	0	0
% Time	54.58	0.00	54.58
◢ Driver API Calls Summary | All			
# Calls	87	87	0
# Errors	0	0	0
% Time	1.11	1.11	0.00
◢ Launches Summary | All			
# Launches	1	0	1
% Device Time	0.00	0.00	0.00
◢ Memory Copies All			
H to D # Copies	2	0	2
H to D # Bytes	32,144	0	32,144
H to D % Time	0.0	0.0	0.0
D to H # Copies	1	0	1
D to H # Bytes	16,072	0	16,072
D to H % Time	0.0	0.0	0.0
D to D # Copies	0	0	0
D to D # Bytes	0	0	0
D to D % Time	0.0	0.0	0.0

Top Device Functions By Total Time Summary | All

	Name	Launches	Device %	Total (µs)	Min (µs)	Avg (µs)	Max (µs)
1	vecAdd	1	0.00	2.432	2.432	2.432	2.432

Questions

Assume that the input vectors to your program has length N. **Answers for the following questions must be based on N**.

1. How many floating operations are being performed in your vector add kernel? EXPLAIN.
2. How many global memory reads are being performed by your vector add kernel? EXPLAIN.
3. How many global memory writes are being performed by your vector add kernel? EXPLAIN.
4. In the vector add project, how many bytes are transferred from the Host to the Device? EXPLAIN.
5. In the vector add project, how many bytes are transferred from the Device to the Host? EXPLAIN.

Submission

After you build your project in 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. Make sure you add your questions as a text file and cuda summary in an image file. This is just like you did for Lab1. Finally zip the folder "Submission" and upload the zipped folder to eclass.

If you don't see this folder than 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

Part	Breakdown (%)
Questions	10 %
Running Code	20 %
Correct Code	50 %
Correct Submission	10 %
CUDA Summary	10%