CSE 520 Computer Architecture -- Spring 2019

Programming Assignment 4

In this assignment, you will execute matrix multiplication on different devices (CPU or GPU) using an OpenCL program. The goal of this assignment is to compare the performance of single thread execution with multi-thread execution as well as to experience OpenCL programming framework.

Requirement

You have to measure the execution time on following configurations:

- Matrix size: 512x512, 1024x1024, 2048x2048
- Reference C/C++ matrix multiplication.
- OpenCL naive matrix multiplication on CPU
- OpenCL naive matrix multiplication on GPU
- OpenCL tiled matrix multiplication on CPU (tile size = 8 and 16)
- OpenCL tiled matrix multiplication on GPU(tile size = 8 and 16)

All these configurations must be put together and executed under a **single** main.cpp. To do so, you need to duplicate some variables and specific OpenCL structure for different devices (say CPU and GPU). For example, two OpenCL command queues and contexts must be created for CPU and GPU separately.

Sample programs, including an OpenCL host program and a kernel file, are provided as **main.cpp** and **matrix_mul.cl** respectively. The main.cpp has a reference implementation of matrix multiplication and an OpenCL framework to execute kernel function on the GPU target. The default matrix size is 1024x1024.

In the kernel file, there are two kernel functions of matrix multiplication, i.e., a naive version “matrix_mul”, and a tiled version “matrix_mul_tile”. Please refer to the course material for the details of these implementations. You don’t have to modify kernel file in this assignment. Only main.cpp is required to change to perform the following steps:

1. initialize the matrices with random numbers between 20 to -20.
2. ddjust the matrix size by modifying the SIZE macro definition.
3. run the sequential version of matrix multiplication and measure the execution time
4. run the “matrix_mul” kernel on CPU and GPU and measure the execution times
5. run the “matrix_mul_tile” kernel on CPU and GPU and measure the execution times

The following table is provided for you to record the execution time. In the report, you only have to include this table without answering any questions. In addition, your report should include:

1. The models of CPU and GPU of the machine on which you collect the measurement data. Note that Intel’s GPU is integrated with CPU cores in processor chip. You may find a list of Intel’s GPUs in https://en.wikipedia.org/wiki/List_of_Intel_graphics_processing_units.
2. A console (terminal) screenshot of running main.cpp and its output message.

Environment

All measured execution time must be reported from PC systems with Linux OS. You may use the PCs in BYENG 2nd floor computer lab 217 which have OpenCL driver installed. You may also install necessary OpenCL SDK/driver in your own Linux systems. The installation steps are included at the end of the assignment document. If you have any issue of installing the SDK/Driver, you may contact the TA or use the machines in the lab.

Matrix size	Reference C implementation (single thread)	OpenCL on CPU			OpenCL on GPU		
		Normal kernel	Tiled kernel (tile_size=8)	Tiled kernel (tile_size=16)	Normal kernel	Tiled kernel (tile_size=8)	Tiled kernel (tile_size=16)
512x512							
1024x1024							
2048x2048							

Due Date

The assignment is due by April 24 at 11:59pm.

What to Turn in for Grading

1. Create a working directory, named "cse520-assgn04-LastName_FirstInitial", for the assignment to include
   - A pdf report to include a list of CPU/GPU models, a screenshot, and the table of measured execution times. Don't forget to add your name and ASU id in the report.
   - the modified main.cpp and matrix_mul.cl.
   - A README file to explain how to compile and run your program.
2. Compress the directory into a zip archive file named cse420-assgn04-LastName_FirstInitial.zip. Note that any object code or temporary files should not be included in the submission. Submit the zip archive to the course Canvas by the due date and time.
3. There will be 20 points penalty per day if the submission is late. Note that submissions are time stamped by Canvas. If you have multiple submissions, only the newest one will be graded. If needed, you can send an email to the instructor and TA to drop a submission.
4. The assignment must be done individually. No collaboration is allowed, except the open discussion in the forum on Canvas.
5. ASU Academic Integrity Policy (http://provost.asu.edu/academicintegrity), and FSE Honor Code (http://engineering.asu.edu/integrity) are strictly enforced and followed.

Important Notes:

- Using fake data to draw table or graph and make comparisons will be treated as unethical practice and a violation of academic integrity will be reported.
- Only the submissions in Canvas will be graded. Any modifications to your submission or replacement of files will be handled as a new submission and may be subject to late submission penalty.

Installing Intel SDK OpenCL/System Studio for your PC - Ubuntu

Step1: Go to https://registrationcenter.intel.com/en/forms/?productid=2345

Step2: Check all the options and click Accept.

Step3: Enter your credentials and submit the form. In the following page create an account with Intel.

Step4: Select Linux, Linux and Android for Ubuntu package.

Step5: Add OpenCL Tools and click continue. In the following page click Download.

Step6: Unzip the downloaded file and untar the system_studio_2019_update_3* file.

Step7: cd system_studio_update_3*

Step8: sudo ./install to get the installation gui.

Step9: You can follow the on-screen instructions to complete the installation process.

Step10: When the installation completes click finish. To check if the installation is right you can cd /opt/intel/system_studio_2019/ to see opencl, opencl-sdk and opencl_compiler_and_libraries_18.1.0.013 present.

Step11: You can now write your OpenCL program with #include<CL/cl.h> header.