EEC 277 – Computer Graphics Project Proposal

Alex Budilovsky, Winter Quarter, 2017

Overview

For Prof. Owens' Computer Graphics class I am proposing the following project goals to gain a better understanding of the GPU and CPU relationship (specifically for my given Intel 5300 graphics card):

- How many shader cores does my GPU have? How fast can each core shade based on complexity of the shader? Specifically, I will be looking at the instruction set line count and seeing if a larger program affects the throughput of the GPU, and is it linear? Also, can register usage be maxed out on a shader core, or it is limited by arithmetic?
- With no background work, how many GFLOPS can my graphics card shader unit produce?
- Now I load my GPU with various workloads (preferably using PyOpenCL, a python wrapper for OpenCL) with arithmetic heavy processes such as matrix multiply. Using various thread counts I see how this background work affects shader output.
- Can I write a shader in such a way that it reduces the amount of stalls per core? Can I write a really slow shader not limited by arithmetic but rather by stalls?
- Lastly, is it possible to figure out how many ALUS are on each shader core?

Seeing as there is no concrete specification sheet for my graphics card, it would be interesting to figure out these specs through experimentation and compare to a standalone card (specifically core/ALU/register count).

Milestones

Here are the general milestones that I hope to hit.

Week 1 (Week of Feb 27) – have a working shader program (OpenGL), and a working 'workhorse' program (preferably using PyOpenCL). Good would be having it on Linux, better to have it on Windows as well.

Week 2 (Week of Mar 6) – have various shader programs written and working, of various complexities. Look at the machine code of each of the functions, and try to analyze register/arithmetic usage. What are the performance metrics for the various functions, with graphs? Does heavy register usage slow the throughput of each core?

Week 3 (Week of Mar 13) – Find workloads that will completely dampen the performance of the shader, and see if a mathematical solution here available. Can it be explained by X number of threads being blocked because of matrix multiplication, for example?

Week 4 (Week of Mar 20) – Extracurricular. Here I hope to vary other metrics, such as triangle size and lighting, and measure effects of shader throughput. Does texturing affect shading?

Note: Some of this process may seem similar to the homework; however, my understanding is that if I am able to program (control) the shader code, as well as analyze the machine code, I will be able to directly manipulate the instructions that are fetched for each core, and take advantage of the multiple ALUs on each core to optimize the shader.

Ultimate Goal

The ultimate goal of this project is to be able to concretely provide evidence of the specifications of my GPU, using the programmable shader feature as a key component. If the evidence is conclusive and the data leads to a firm conclusion, I will view this project as a success.