

Independent Project Proposal FINAL - Sam Kelly 2017070

by Samuel Jackson Kelly

Submission date: 21-Oct-2022 01:17PM (UTC+0100)

Submission ID: 187195984

File name: 41605_Samuel_Jackson_Kelly_Independent_Project_Proposal_FINAL_-
_Sam_Kelly_2017070_443262_1513385453.docx (184.16K)

Word count: 2945

Character count: 14999

Exploration of the CUDA Framework around Implementation Boundaries

Individual Project Proposal

Samuel Kelly (2017070)

Dr, Tim Bashford

Table of Contents

1. Introduction..... 1

2. Background Research 2

3. Aims & Objectives..... 5

4. Project Design..... 6

5. Resources & Planning..... 8

References..... 10

Table of Figures

Figure 1 Rough models of a CPU and GPU's Architecture..... 3

Figure 2. Gantt Chart showing early stages of project..... 9

Figure 3. Brief plan with aim breakdown and some rough deadlines for work..... 9

1. Introduction

In recent years graphics processing units (GPUs) have improved in performance massively from generation to generation. Recently Nvidia announced the GeForce RTX 40 series and the performance benchmarks for these cards display an increase in performance of anywhere from 50% to 70% over the previous generation [1]. With this in mind we can take a deeper look at the computational power of these new cards and explain some of the processes behind what makes them so vital to modern computer systems. With limitations of Central Processing Units (CPUs) becoming more of a bottleneck offloading some computations to the GPU has never been more vital to improve efficiency. This project proposal will be discussing a few of the key technologies that make GPUs so powerful in today's computers and will go through the aims and objectives, as well as the proposed research methods to test performance benchmarks of GPU processing.

2. Background Research

This literature review will focus on what Compute Unified Device Architecture (CUDA) is, as well as some background on the technology used leading up to the creating of CUDA namely the introduction of General Purpose computation on Graphics Processing Units (GPGPU), what it has done for technology in terms of computational power, and some limitations of the framework.

At their first conception GPUs were used to decrease the time it would take to render 2D and 3D graphics. With gaming starting to get more popular in the 80s it became more and more apparent that a system would be needed in order to process these games images at a reasonable speed [2]. In the early 2000s CPUs were starting to reach a point in development where generational improvements were only yielding very small improvements compared to previous years. Around 2000 to 2006 CPU performance had stayed in a relatively similar state with processors struggling to get past 3.6 Gigahertz (GHz) clock speeds [3]. The combination of CPU development reaching a difficult point in its life and the success of the GPU in aiding games and other forms of 3D graphics with rendering times and increases in quality due to the GPU's ability to process texture maps and vertex shaders far quicker than the CPU, researchers began to look at ways in which they could utilise the GPU for tasks that were not necessarily graphics related [4]. GPGPU is essentially a way for software to use the GPU in order to assist the CPU with some processing. To write a GPU enabled application

the developer will need a way to program their application with the correct algorithms for utilising the GPU. A plethora of Application Programming Interfaces (APIs) are available to developers in order to make their program GPU enabled [5]. This is where the creation and invention of CUDA by Nvidia becomes immeasurably useful for software developers and engineers.

The CUDA framework is an architecture that allows software to use GPUs for general purpose processing and not exclusively parallel computations. This allows the CPU to offload some of the processing tasks to the GPU speeding up processing times [6]. When talking about CUDA it is important to note the architectural differences between the CPU and the GPU that make CUDA such an efficient way of processing. Looking at figure 1 the major differences between the CPU and GPU can be seen. The GPU is comprised of hundreds of cores allowing it to carry out many parallel process much faster than the CPU is capable of [7].

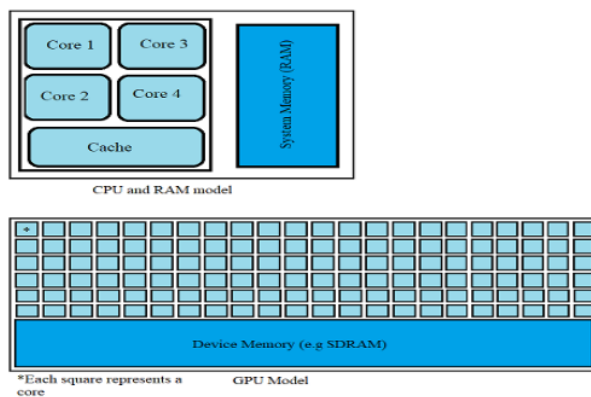


Figure 1 Rough models of a CPU and GPU's Architecture

Nvidia are the creators of CUDA and have been a major part in the reason process such as CUDA and similar ones are popular today. In order to make CUDA accessible and easy to implement to projects Nvidia designed it using C/C++ languages and even provide a Software Development Kit (SDK) to aid with the process. The SDK is an incredibly useful tool for developers and is a huge bonus when looking to implement CUDA into a project [8]. When taking advantage of CUDA multithreaded applications and highly parallel algorithms yield the highest benefit in processing times [9]. Coding in C/C++ gives the program an edge when it comes to CUDA performance and allows for breaking down a program into thousands of threads. If a program can not be broken down into a large number of threads

then it may be more beneficial to not use CUDA. This is something that can be explored and tested in more detail during the project. The CUDA API is an incredible invention and was a major breakthrough in computing, it allows the everyday developer to create GPU enabled software and improve computational efficiency without requiring extensive background knowledge on the inner workings of GPUs and coding. However, there are some shortcomings for CUDA, the major one being CUDA can only run on Nvidia graphics cards as it is of course created by Nvidia [10]. This means that a program designed only with CUDA would not work with GPUs from other manufacturers such as AMD or Intel.

In order to account for the variety of computers and systems the Open Computing Language (OpenCL) API was developed by Apple Inc and the Khronos Group. OpenCL provides a way for developers to code GPU enabled software that will work on a variety of systems regardless of the manufacturer of the GPU in the system. A group of large companies have also submitted conformance tests with OpenCL to ensure that the software create with OpenCL will work on the respective systems [11]. OpenCL generally provides a better standard if a developer wishes to have their app or program run on a plethora of systems, as CUDA is only available to work with Nvidia GPUs and on the major operating systems Windows, Linux, and MacOS. Because of the CUDA API and framework being owned by Nvidia they provide good support in order to make sure that the app or program a developer might be creating runs well on their devices [12]. Because of this using CUDA generally makes for a smoother and more efficient application. However, of course this is heavily reliant on the skills of the developer, if a developer is proficient with the OpenCL API and has made numerous apps or programs with it before then they may well create a better program than say a novice who chooses to use CUDA because of the support available to them.

In summary CUDA has been and will be an incredibly useful tool for developers for the foreseeable future. Nvidia GPUs continue to grow in performance at an incredible rate and so do their manufacturing competitors namely AMD who were very close to having a GPU many would consider to be better than the 30 series from the previous generation of cards. OpenCL will always have limitations in how efficient you can optimise a program for such a wide variety of systems and has a long way to go in order to catch up to CUDA which has been such a vital addition in the world of modern technology.

3. Aims & Objectives

There will be four main aims with this project in order to help track progress and make sure the project stays on track.

1. Research and write the literature review – start finalizing review by 26/10/2022
 - a. Develop a background understanding of CUDA and related technologies
 - b. Review papers and developments made in the field of study
 - c. Write a brief summary of background info for project proposal – 21/10/2022
 - d. Write a more in-depth full literature review 26/10/2022
2. Create the program to test the algorithms
 - a. Develop a piece of software that is able to test various GPU enabled algorithms
 - b. Add functionality for the software to output a set of results that can be used for analysis
3. Run tests with the program and collect the results as data
 - a. Test the results of efficiency and run time of various algorithms
 - b. Tabulate results from tests
4. Report findings and discuss the results collected
 - a. Create data visualisations of results
 - b. Analyse the runtime and efficiency of each algorithm tested.
 - c. Report findings and write up final analysis and conclusion

4. Project Design

4.1 Outline of Project Process

A relatively straightforward process will be followed in order to make sure the development of the project comes along smoothly. The first few weeks will be spent reviewing relevant literature and gathering a good understanding of the background in the related subject. Technologies that have led to the development of CUDA as well as some competitors or alternatives to CUDA will be looked at and research papers, journals, websites, and books where possible will be analysed and included in the final literature review with relevant reference.

Weekly meetings with the supervisor will provide for a way to track progress and make sure the deadlines are being hit. A few of the major deadlines in the project would be the initial project proposal being due 21st of October 2022. The majority of the full literature review is planned to be finished by the 30th of October 2022. Such a tight deadline has been chosen as the principal researcher and supervisor believe this date best suits the current schedule of the researcher. Having a beta version or early version of the program which will be used to test the algorithms is expected to be complete around 5th of January 2023. There are other major deadlines such as the collection of all data chosen, and review of the data collected throughout the project. However, these have not yet been set in stone at this point in the project.

The hypothesis for this project is in what ways does the CUDA framework effect software efficiency and runtime? This was chosen so that the algorithm efficiency can be measured easily and the data can be reported and presented in a medium that is neat and easy to digest for the reader.

4.2 Data Collection

Data collection will involve adding some functionality to the software created for the project. At the end of running the software a breakdown of runtime and other elements required to analyse the effectiveness of the program will be displayed. These results will then be recorded and added to a document or database with all other results collected. Collected data will only be collected from the reports generated by the program. No human test subjects will be used, and no other programmers will be asked to develop a similar system of their own to

compare results with. This could be a limitation of the project. However, the scope of this project is not big enough to warrant recruiting the help of other programmers who may want financial compensation in order to devote their efforts to creating useful data for this project.

4.3 Data Evaluation

Evaluation of the data will include the creation of a number of graphs and tables. Line graphs to compare execution times side by side will prove most useful in analysis of the results and allow for easy summary and comments. Each algorithm will be briefly talked about in the final report and then an overall conclusion will be drawn. Using the data collected some reasonable predictions and assessments can be drawn about the benefit of GPU enabled applications and software for computing.

4.4 Risks & Limitations

There are minimal risks to this project in most aspects. One of the biggest risks the investigator may encounter is one of personal harm, eye strain, muscle soreness, bad posture, unhealthy eating schedules are the biggest risks. In order to mitigate all of these consistent reminds will be set at health intervals to make sure breaks of at least 5 minutes are taken to stretch, drink fluids, and give the investigators eyes a break. Another risk would be the possibility of the computer or short circuiting possibly because of a faulty part or even because of a power surge due to stormy weather. To minimize these risks extra care will be taken regarding not pushing the computer to any sort of limit in terms of power draw or consumption. The best way to mitigate the risk of external power surges is to use a surge protected wall socket or a surge protected extension cord. There are little to no environmental risks created from this project. However, to be on the safe side fire hazards from the computer have been acknowledged and to best combat this fire extinguishers will be near to each location of the computer and proper fire protocols will be studied and followed for places such as the UWTSD campus.

Limitations on this project are mainly imposed by the investigators lack of expertise in the area. Although the investigator is competent with programming in the languages related to CUDA they do not have the experience of a person who may have been developing CUDA enabled applications for 3+ years as an occupation. The second limitation would be

related to the first in that the level of program and depth of the algorithm written in order to test will be relatively low compared to an application or even a game that is developed professionally to be GPU enabled.

4.5 Ethical Issues

This research project is very ethically low risk. There should be no cause for concern on any ethical fronts as the only participants are the chief investigator Sam Kelly and the computers involved in testing. No test subjects will be used to collect any data of any form.

5. Resources & Planning

5.1 The testing environment related to such a research project as this one is relatively simple and straightforward. No major specialist equipment will be needed and no specialist software will be needed

5.1.1 Hardware Resources

The main hardware resource needed for this project would be a somewhat modern computer with an Nvidia GPU. The investigator personally has access to a computer with a GeForce RTX 2060 which will allow for running CUDA based applications. Additionally there are in university resources at UWTSD Swansea campus with similar or better hardware available.

5.1.2 Software Resources

Similarly to the hardware resources for this project the software requirements are also very simple. Visual Studio Code is the main piece of software that will be used to code the software used in testing. The Nvidia CUDA Toolkit and CUDA Zone will also be used as there is a large library of resources available to aid in the development process of CUDA enabled software. Time permitting OpenCL may also be used to test and compare to the efficiency of CUDA based software.

5.1.3 Other Resources

In order to reach the university campus for testing on different computers a car will be needed to commute to and from the campus. However, should something happen to the car being used it is also possible to get a train or alternative mode of transport to the campus itself to carry out testing.

5.2 Planning

To aid with planning this project a Gantt chart will be used and update alongside the project to make sure the time used for the project is being managed carefully and effectively. The Gantt chart also provides a way for the investigator to see if any steps of the project are falling behind and need to be addressed or have more resources dedicated to them.

The weekly meetings with the supervisor will also provide as a way to make sure the project stays on track as well as the interim viva to discuss the progress that has been made on the project so far at the point of interview.

To-Do	Week 1 26/09/22	Week 2 03/10/22	Week 3 10/10/22	Week 4 17/10/22	Week 5 24/10/22	Week 6 31/10/22	Week 7 07/11/22	Week 8 14/11/22	Week 9 21/11/22
Read Background Research									
Write Background lit review									
Write complete lit review									
Begin developing software									

Figure 2. Gantt Chart showing early stages of project

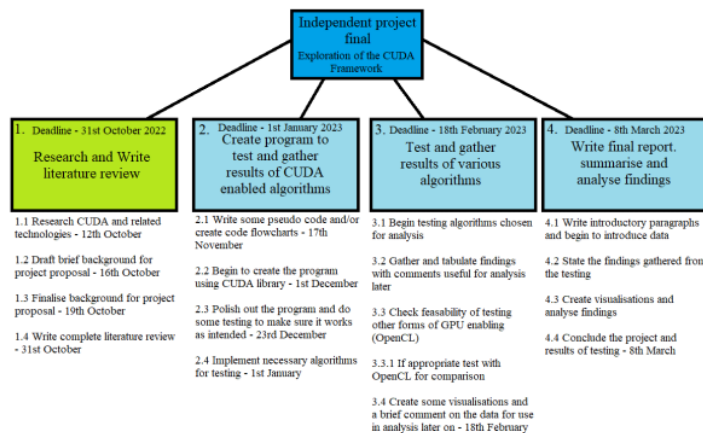


Figure 3. Brief plan with aim breakdown and some rough deadlines for work

References

- [1] JayzTwoCents, *We need to have a chat about these 4090 Benchmarks...*, Claremont, CA: JayzTwoCents, 2022.
- [2] B. House, "A Brief History of GPUs," Medium, 19 July 2019. [Online]. Available: <https://medium.com/neuralmagic/a-brief-history-of-gpus-27122d8fd45>. [Accessed 13 October 2022].
- [3] M. Martin, *History of Processor Performance*, Philadelphia, PA: University of Pennsylvania, 2012.
- [4] R. Vuduc and J. Choi, "A Brief History and Introduction to GPGPU," in *Modern Accelerator Technologies for Geographic Information Science*, New York, NY, Springer, 2013, pp. 9-23.
- [5] Heavy.ai, "https://www.heavy.ai/technical-glossary/gpgpu," Heavy.ai, 2022. [Online]. Available: <https://www.heavy.ai/technical-glossary/gpgpu>. [Accessed 13 October 2022].
- [6] F. Abi-Chahla, "Nvidia's CUDA: The End of the CPU?," Tom's Hardware, 18 June 2008. [Online]. Available: <https://www.tomshardware.com/reviews/nvidia-cuda-gpu,1954.html>. [Accessed 8 October 2022].
- [7] P. Kaur and Nishi, "A Survey on CUDA," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp. 2210-2014, 2014.
- [8] P. Gupta, "CUDA Refresher: Getting started with CUDA," 6 May 2020. [Online]. Available: <https://developer.nvidia.com/blog/cuda-refresher-getting-started-with-cuda/>. [Accessed 2022 October 9].
- [9] J. Ghorpade, J. Parande, M. Kulkarni and A. Bawaskar, "GPGPU PROCESSING IN CUDA ARCHITECTURE," *An International Journal*, vol. 3, no. 1, pp. 102-115, 2012.
- [10] K. Shan, "Performance & Limitations of CUDA on GPU for Computation of Fourier Transform," in *National Conference on Information Scieces*, Princeton, NJ, 2010.
- [11] Khronos Group, "API Adapters," Khronos Group, [Online]. Available: <https://www.khronos.org/conformance/adapters/conformant-companies#opencl>. [Accessed 14 October 2022].
- [12] D. Exterman, "CUDA vs OpenCL: Which to Use for GPU Programming," Incredibuild, 7 June 2021. [Online]. Available: <https://www.incredibuild.com/blog/cuda-vs-opencl-which-to-use-for-gpu-programming#:~:text=CUDA%20is%20able%20to%20run,system%20and%20most%20hardware%20varieties..> [Accessed 14 October 2022].

Independent Project Proposal FINAL - Sam Kelly 2017070

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

64/100

Intro / A&O (400 words)		
Project title; Identification of a specific problem to be investigated; Well-defined aims and SMART objectives; Explaining why this project is worth doing; Discussion of what the project needs to do in order to be a success; Description any deliverables that will be produced at the end of the project A reasonable introduction, with a valid if somewhat broad hypothesis. Aims and objectives would benefit from being more precise.	20	11
Background to the Project / Literature Review (400 words)		
Description of how the work to be done by in the project relates to the work identified in the literature review; Discussion of any ethical issues raised by the project Identifying what has already been done within the field; Topic familiarity; Brief literature review using appropriate sources – 6-8 journal papers and book chapters rather than just websites; Selection of relevant models & concepts	20	12

A reasonable introduction to part of the work. This focuses on the nature of GPUs and introduces the idea underpinning CUDA but would ideally go a little further into the benchmarking portion of the work.		
Project Design (500 words)		
Clear statement of which development methods are to be used and why they are appropriate; Clear statement of which tools are to be used and why they are appropriate; Clear description of how data collected will be analysed A good methodology proposed – the writing, however, is a bit verbose in this section. Greater tersity would actually aid readability and clarity.	30	21
Resources / Planning (200 words)		
List of any deliverables; Clear diagram showing both timing and resource use; Completed ethics form; Limited and relevant appendices Not too much is required for this project, so the resources are light. Your Gantt chart (note spelling), however, should include the plan for the remainder of the project.	15	9

Document Presentation		
Correct spelling, grammar and punctuation; Flow of narrative/fluency; Coherence; Word processed ; Correct use of referencing with bibliography; Strict word limit The proposal is well-written, there are some minor SPG issues. It is reasonably well presented. Try to use vector graphics if you can, some of them are a tad pixelated.	15	11
TOTAL	100	64

PAGE 1
PAGE 2
PAGE 3
PAGE 4
PAGE 5
PAGE 6
PAGE 7
PAGE 8
PAGE 9

