How Well Can Low-Power Embedded GPUs Handle Large Non-Graphical Tasks?

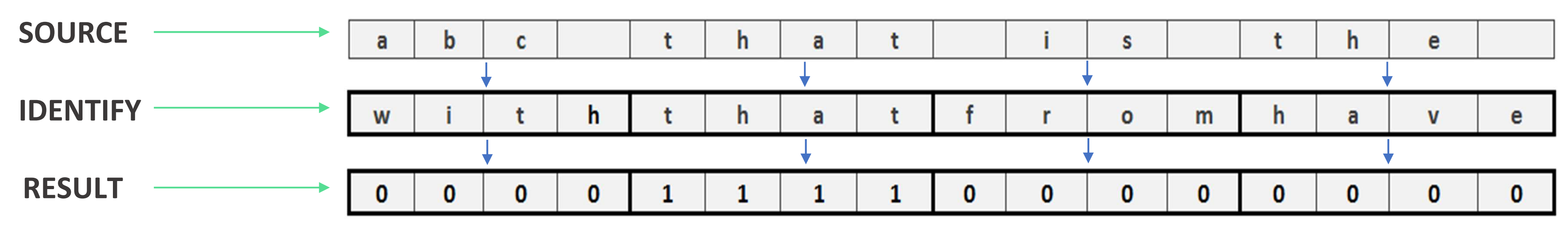
A study into low power embedded GPU and CPU run time and power consumption processing non graphical loads through OpenCL framework

INTRODUCTION

In recent years the shift of the role of the GPU (graphics processing unit), traditionally a fixed function special purpose graphics processor, to a more general purpose programmable unit has meant greater productivity from heterogeneous computing[1]. Heterogeneous computing involves the use of multi-cores, CPUs, GPUs and DSPs (Digital Signal Processors) to synergistically accelerate large computations[2]. This is vital in modern applications including video/audio processing[2]. The use of the OpenCL programming model is what allows programming between these compute units[3]. The emergence of the OpenCL framework means that low power embedded systems can now also implement GPGPU processing (General Purpose computing on Graphics Processing Units)[1-3].

METHOD

Using the map reduce algorithm for string searching



Map reduce is a mathematical algorithm used to divide a task into small parallel friendly parts that can be assigned to each CPU or GPU core[2]. The string searching algorithm works by first declaring a vector that contains the words that are being searched for, this is shown by the identify vector in the diagram. The source text is then read from and each 16 character length vector (including spaces and punctuation) is compared with the identify vector. If the letters match then it scores a 1 otherwise it scores a 0. The 1s and 0s are then collated in a result vector.

HARDWARE

The experiment was conducted using the Odroid XU4 connected with the SmartPower2 to measure the voltage, current and power consumption.
The Odroid was an important part of this experiment containing the low power GPU and CPU units needed:

- CPU: Arm Cortex – A7 (8 cores)
- GPU: 2 Mali-T628s, one with 4 cores and one with 2 cores

Monitoring CPU activity showed all 8 cores to be operational when running OpenCL codes.

SOFTWARE

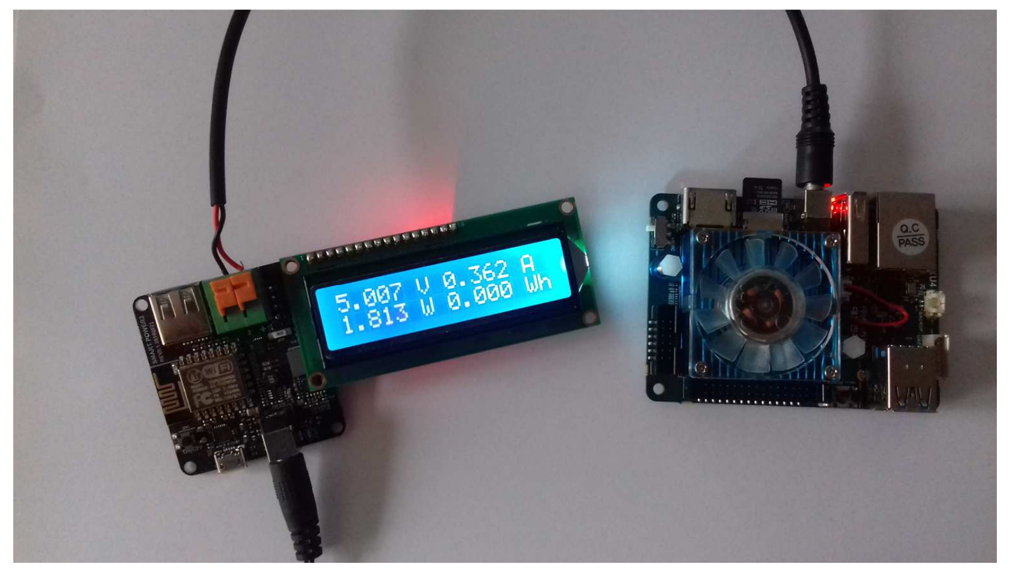
The latest Ubuntu image (15.04) was installed onto the Odroid to provide an operating system. The OpenCL framework allows the same code to be run on both the CPU and the GPU for run times to be measured. The run times used are an average of 3 runs of the application for each word count.

AIMS AND OBJECTIVES

The main aim of the internship was trying to learn the OpenCL framework to appreciate how this method of coding can enable programming across heterogeneous units such as multi-cores and GPUs. To evaluate this learning an experiment was devised to identify the productivity of low power GPUs against CPUs when processing non graphical loads. While many research papers have looked into the GPGPU based problems with the goal of maximising speed-up on high end GPUs the goal of this experiment was to give a low power CPU and GPU a large processing task and monitor the power consumption and run times.

OBJECTIVE

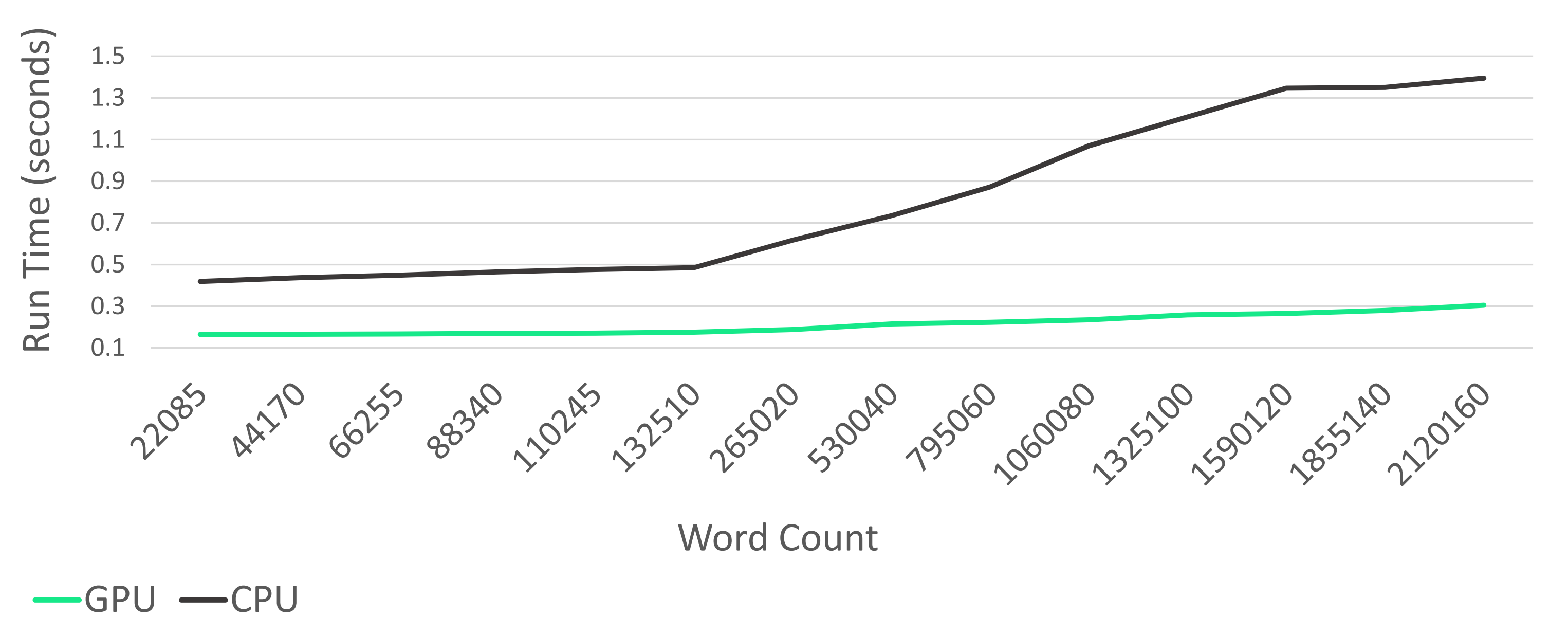
Learn to understand and write OpenCL programs such that an algorithm based test can be devised to demonstrate these learnings where the performance of a low power embedded system's GPU and CPU can be compared taking into consideration the run times and the power consumption.



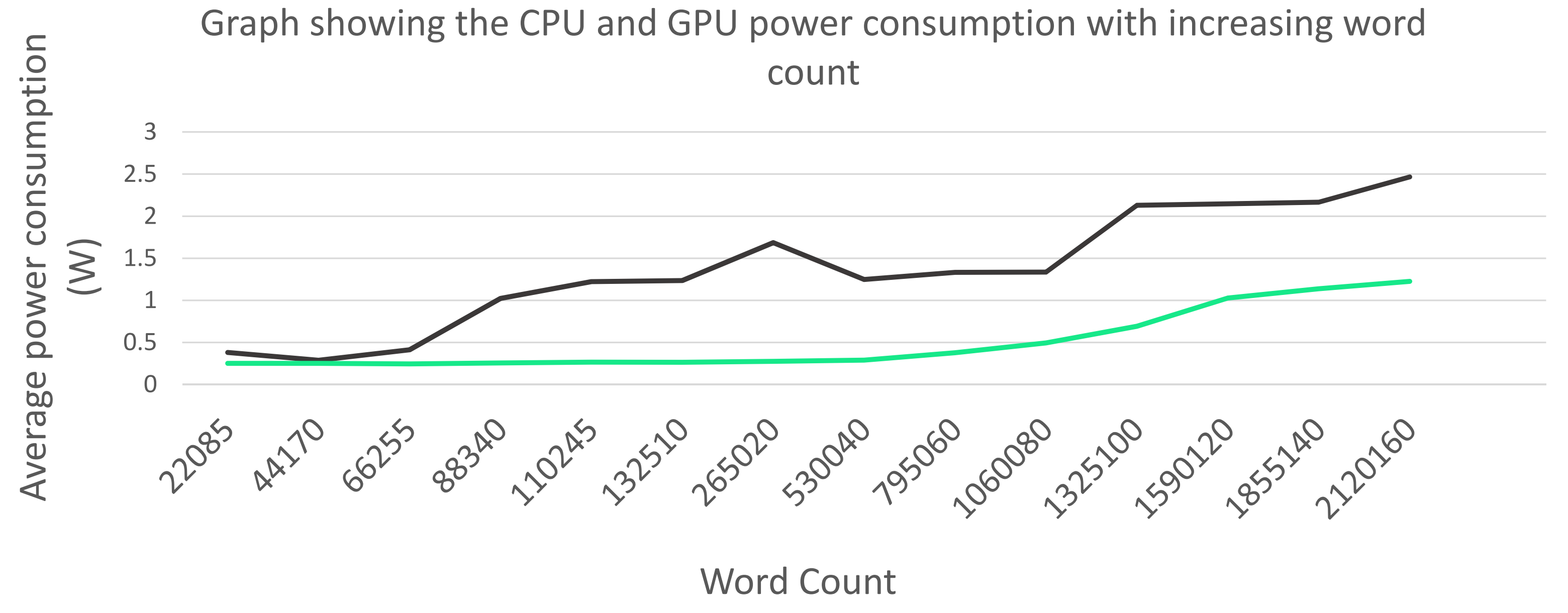
Shown to the left is the Odroid XU4 and SmartPower2 meter used to get the readings for the power consumption

RESULTS AND DISCUSSION

Graph showing the CPU and GPU run times with increasing word count



Graph showing the CPU and GPU power consumption with increasing word count



Both results indicate that the GPU is superior to the CPU in terms of providing faster run time and smaller power consumption for every word count. While the CPU run time can be seen to increase quite quickly with increasing word count the GPU run time increases slowly. The same is true for the power consumption graph. On average the GPU run time was calculated to be 56.4% faster and consume 59.3% less power.

After the maximum word count shown on the graph the Odroid reached its maximum computational ability and the system crashed with word counts above 2500000.

While the GPU does appear to process the data quicker and consume less power for this application intelligent trade-offs should be made when applying this finding to other non graphic work loads. Other practical aspects need to be considered such as the fact the GPU has more restricted features compared to the CPUs (e.g. limited or no user defined memory, small instruction set and limited number of registers).

In short if the restrictions of the GPU do not pose a problem for the data to be processed then a larger proportion of the processing task can be given to the GPU.

FUTURE WORK

Currently it is only possible for either the CPU or the GPU to be used exclusively when writing OpenCL code on the Odroid. This is down to issues in the OpenCL installation process. It is hoped that once both the CPU and GPU are simultaneously operational further work can be done exploring the CPU vs GPU performance.
By splitting the work load between the CPU and GPU and once again examining the run times this study can be repeated to find the optimum non graphical work load distribution between CPU and GPU for string searching algorithm.
Extending this idea further, once the installation issue is resolved it could be possible to chain several Odroid XU4s together. This would allow for big data processing and the run times can then be measured against high end CPUs and GPUs.

ACKNOWLEDGEMENTS

I would like to acknowledge my supervisor for this project, Dr Rishad Shafik, for allowing me to work with him and his colleagues in the micro systems group at Newcastle University. I would also like to acknowledge postgraduate student Sidharth Maheshwari, who acted as a secondary point of contact throughout the research project. I would finally like to thank Newcastle University, for awarding me this scholarship to explore my interest in this particular research area and in postgraduate study.

References

[1] A. Maghazeh, U. Bordoloi, P. Eles and Z. Peng, "General Purpose Computing on Low-Power Embedded GPUs: Has It Come of Age?", *Liu.diva-portal.org*, 2018. [Online]. Available: http://liu.diva-portal.org/smash/get/diva2:610784/FULLTEXT01.pdf. [Accessed: 01- Jul- 2018].

[2] M. Scarpino, *OpenCL in action*. Shelter Island, N.Y: Manning, 2013.

[3] S. Mittal and J. Vetter, "A Survey of Methods for Analyzing and Improving GPU Energy Efficiency", *Academia.edu*, 2018. [Online]. Available: https://www.academia.edu/6644474/A_Survey_of_Methods_For_Analyzing_and_Improving_GPU_Energy_Efficiency. [Accessed: 12- Jul- 2018].

Sheikh Tousif Rahman
Student Number: 16008908
Contact: s.rahman@newcastle.ac.uk