



# Energy Efficiency Analysis of The SoC Based Processors for The Scientific Applications

Swapnil Gaikwad

August 23, 2013

MSc in High Performance Computing

The University of Edinburgh

Year of Presentation: 2013



## Abstract

Processors used in the smartphones and tablets are getting more powerful with every new generation. Due to limited amount of battery storage, these processors are designed for the energy efficiency as the primary focus. Power consumption of the HPC systems is one the biggest challenges for them. There is a lot of experiments and research going on to use these embedded SoC based processors to address the power challenge for the HPC systems. However, these low powered processors provide energy efficiency at the cost of performance which is the critical factor for the HPC applications. In this project the analysis was done to understand the role of such embedded systems in the future HPC systems.

For the analysis the ARM architecture based processors and commonly used Intel Xeon processors were used. ARM based processors were from the Pandaboard and Boston Viridis system. Three clusters containing these three different types of processors were built. There were different types of benchmarks selected to test the performance of these processors and clusters. Then the tests were conducted to check their energy efficiency for the application specific benchmarks. Finally, the real-world scientific applications were tested for their energy efficiency on these systems.

Interestingly it was found that the Intel Xeon processors are much more powerful as well as energy efficiency than the SoC based processors. This was due to the different requirements of the HPC applications than that of the smartphone and tablet market. During this dissertation an analysis was done on these performance critical aspects of the processors. There were some interesting benefits the SoC based processors and their potential to power future HPC systems were found. They are also discussed here.

ISC'13 Student Cluster Challenge was part of this dissertation. It involved building the cluster and running real-world scientific codes on it. This dissertation also contains the information about the competition and the work done as part of it.

## **Acknowledgements**

I wish to sincerely thank my supervisor Ms. Xu Guo, for her guidance and the honest feedback that she gave me throughout the project. I would like to thank David Power and Micheal Holiday from the Boston Ltd. who helped us with the hardare and technical support throughout the competition and dissertation.

I would especially thank my ISC' 13 cluster challenge teammates Don Browne, Nikolaos Koutsikos and Paolo Martino for their invaluable technical help throughout the competition and the dissertaion period.

I would like to thank my parents for their unconditional love and support throughout my MSc course.

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Report Organization . . . . .	3
<b>2</b>	<b>Project Background</b>	<b>5</b>
2.1	Student Cluster Challenge . . . . .	5
2.2	Previous Work . . . . .	6
2.3	Embedded System-on-Chip and Energy efficiency . . . . .	7
2.3.1	Case Study of Blue Gene/Q Chip . . . . .	8
2.4	Current HPC systems and Energy Efficiency . . . . .	10
2.5	Current HPC Projects towards Energy Efficiency . . . . .	13
2.5.1	Project Mont-Blanc . . . . .	13
2.6	Summary . . . . .	16
<b>3</b>	<b>Project Hardware</b>	<b>17</b>
3.1	Hardware Selection . . . . .	17
3.2	Hardware details . . . . .	17
3.2.1	Pandaboard ES . . . . .	17
3.2.2	Boston Viridis . . . . .	20
3.2.3	Intel Xeon . . . . .	22
3.3	Table of Comparison . . . . .	23
3.4	Summary . . . . .	24
<b>4</b>	<b>Project Setup</b>	<b>25</b>
4.1	Hardware Setup . . . . .	25
4.1.1	Hardware Setup Overview . . . . .	25
4.1.2	Pandaboard Cluster . . . . .	25
4.1.3	Viridis Cluster . . . . .	26
4.1.4	Intel Cluster . . . . .	28
4.2	Software Setup . . . . .	29
4.2.1	Software Setup Overview . . . . .	29
4.2.2	Overview . . . . .	29
4.2.3	Viridis Cluster . . . . .	33
4.2.4	Intel Cluster . . . . .	34
4.3	Summary . . . . .	34
<b>5</b>	<b>Experimental Methodology</b>	<b>35</b>
5.1	Methodology . . . . .	35
5.2	Power Measurement . . . . .	36

<b>6</b>	<b>Synthetic Benchmarks</b>	<b>37</b>
6.1	CoreMark . . . . .	37
6.2	HPCC Benchmark Suite . . . . .	38
6.2.1	DGEMM . . . . .	39
6.2.2	STREAM . . . . .	39
6.2.3	RandomAccess . . . . .	40
6.2.4	PTRANS . . . . .	41
6.2.5	Pingpong . . . . .	42
6.3	HPL . . . . .	44
6.3.1	Pandaboard Cluster . . . . .	44
6.3.2	Viridis Cluster . . . . .	46
6.3.3	Intel Cluster . . . . .	48
6.3.4	Linpack Energy Efficiency . . . . .	50
6.4	Summary . . . . .	51
<b>7</b>	<b>Application Specific Benchmarks</b>	<b>53</b>
7.1	NAS Parallel Benchmark Suite (NPB) . . . . .	53
7.1.1	Fourier Transform (FT) . . . . .	53
7.1.2	Conjugate Gradient (CG) . . . . .	55
7.1.3	MultiGrid (MG) . . . . .	56
7.2	Summary . . . . .	57
<b>8</b>	<b>Scientific Applications</b>	<b>59</b>
8.1	MILC . . . . .	59
8.1.1	MILC Overview . . . . .	59
8.1.2	MILC Scaling . . . . .	61
8.1.3	MILC Energy Efficiency . . . . .	62
8.2	AMG . . . . .	64
8.3	Summary . . . . .	65
<b>9</b>	<b>ISC'13 Student Cluster Challenge Work</b>	<b>67</b>
9.1	Overview . . . . .	67
9.2	Training Trip . . . . .	67
9.3	Cluster and software Configuration . . . . .	68
9.4	MILC on GPUs . . . . .	68
9.5	MILC Benchmarking . . . . .	69
9.6	HPCC Hack . . . . .	70
9.7	Summary . . . . .	70
<b>10</b>	<b>Project Conclusion</b>	<b>71</b>
10.1	Conclusion . . . . .	71
10.2	Future Work . . . . .	73
10.3	Project Evaluation . . . . .	74
	<b>Appendices</b>	<b>77</b>
<b>A</b>	<b>Scripts</b>	<b>79</b>
<b>B</b>	<b>Linpack Sample output</b>	<b>81</b>
<b>C</b>	<b>Scaling Results for AMG</b>	<b>83</b>







# List of Figures

2.1	Blue Gene/Q Compute Chip Architecture(Image reproduced from [23]) . . . . .	8
2.2	Packaging of Blue Gene/Q Compute Card(Image reproduced from [22]) . . . . .	9
2.3	Processor Generation in the Top500 supercomputers June 2013 (Image reproduced from [24]) . . . . .	10
2.4	Kepler GK110 Die Photo (Image reproduced from [29]) . . . . .	11
2.5	Kepler Streaming Multiprocessor in the Kepler architecture based GPU (Image reproduced from [29]) . . . . .	12
2.6	Road map of the ARM based prototypes (Image reproduced from [33]) . . . . .	13
2.7	Building Blocks of Tibidabo Cluster (Image reproduced from [34]) . . . . .	14
2.8	Building Blocks of the Pedraforca Cluster (Image reproduced from [36]) . . . . .	14
2.9	Expected performance growth of ARM Mali GPU (Image reproduced from [35])	15
2.10	Building blocks of next Mont-Blanc Prototype (Image reproduced from [36]) .	15
3.1	Pandaboard ES 4460 and its Components(Image reproduced from [42]) . . . . .	18
3.2	Pandaboard ES 4460 Block Diagram(Image reproduced from [42]) . . . . .	19
3.3	Viridis Chassis and Viridis Energy Card (Image reproduced from [43]) . . . . .	20
3.4	Block Diagram of a Viridis Node containing Calxeda EnergyCore(Image reproduced from [43]) . . . . .	20
3.5	Compute Blade with Dual Socket Intel Xeon E5-2600 series Processors (Image reproduced from [48]) . . . . .	22
3.6	Compute Blade used at ISC'13 SCC (Image reproduced from [48]) . . . . .	23
4.1	Block Diagram of the Pandaboard Cluster . . . . .	26
4.2	Block Diagram of the Viridis Cluster . . . . .	26
4.3	Block Diagram of the Intel Cluster . . . . .	28
5.1	Set of selected benchmarks for the experiment . . . . .	35
6.1	Performance of the cores in different clusters for the CoreMark benchmark . .	38
6.2	Performance Panda, Viridis and Intel Clusters on DGEMM benchmark . . . . .	39
6.3	Performance Panda, Viridis and Intel cores on STREAM benchmark . . . . .	40
6.4	Performance Panda, Viridis and Intel Clusters on RandomAccess benchmark .	41
6.5	Results of the PTRANS benchmark on all the clusters . . . . .	41
6.6	Latency for the Small and Large Size message on the clusters . . . . .	42
6.7	Network bandwidth for the Small and Large Size message on the clusters . . .	43
6.8	Linpack scaling on the Pandaboard Cluster . . . . .	46
6.9	Linpack scaling on the Viridis Cluster . . . . .	47
6.10	Linpack scaling on the Intel Cluster . . . . .	49
6.11	Energy efficiency comparison of Panda, Viridis and Intel Cluster . . . . .	50

7.1	Energy Efficiency for the NAS 3D-FFT Benchmark . . . . .	54
7.2	Energy Efficiency for the NAS CG Benchmark . . . . .	55
7.3	Energy Efficiency for the NAS MG Benchmark . . . . .	56
8.1	Execution time for the different optimization for the su3_rmd application . . . . .	60
8.2	Scaling of MILC on Pandaboard Cluster . . . . .	61
8.3	Scaling of MILC on Viridis Cluster . . . . .	62
8.4	Scaling of MILC on Intel Cluster . . . . .	63
8.5	Results on Performance/Watt matric for the MILC application . . . . .	63
8.6	Results on performance/Watt matric for the AMG application . . . . .	65
9.1	Scaling with and without GPU version of the MILC . . . . .	68
9.2	Intel profiler screenshot showing the functions and loops taking the maximum time . . . . .	69
10.1	Dissertation work distribution . . . . .	74

# List of Tables

3.1	Details of hardware specifications of the three types of processors . . . . .	23
4.1	Network Configuration of the Intel Cluster . . . . .	28
4.2	Network Configuration of the Pandaboard Cluster . . . . .	30
6.1	Performance optimization results for a single Pandaboard Node . . . . .	44
6.2	Linpack scaling for the Pandaboard Cluster . . . . .	45
6.3	Performance optimization results for a single Viridis Node . . . . .	47
6.4	Linpack scaling for the Viridis Cluster . . . . .	48
6.5	Performance optimization results for a single Intel Node . . . . .	48
6.6	Linpack scaling for the Intel Cluster . . . . .	49
7.1	Energy Efficiency for the NAS 3D-FFT Benchmark . . . . .	54
7.2	Energy Efficiency for the NAS CG Benchmark . . . . .	55
7.3	Energy Efficiency for the NAS MG Benchmark . . . . .	56
8.1	Performance of MILC scaling in MFLOP/sec for different clusters . . . . .	62
8.2	Linpack scaling the Intel Cluster . . . . .	64



# Chapter 1

## Introduction

In today's research methodology, modeling and simulation has become an integral part to the conventional approach of theory and experimentation. Use of the scientific computing has made it possible to precisely measure, analyze, understand and predict complex events which are not yet occurred. It is now possible to accurately simulate the behavior of a system depending on hundreds of its components. From simulation of a complete nuclear reactor where temperature exceeds 100 million degrees centigrade to the prediction of the climate, a century down the road is already been accomplished [1]. Such simulations, research require enormous amount of the computation power, and supercomputers of the age are consistently providing it from past few decades.

Today's supercomputer has achieved a petascale milestone, which is a capability to perform order of  $10^{15}$  floating point operations per second. However, there is a whole set of problems which require much more computation power than the petascale systems. Problems like the complete aircraft simulation, finer resolution climate models and modeling complete weapon system are a few of the names [2]. Such problems require the exascale systems which are thousand times faster than the existing petascale systems. Building an exascale system is extremely challenging task due to many technology limitation. DARPA commission published a report which described the challenges to achieve exascale system with current technology. It also addressed the key factors for these limitations [3]. Along with the need of scaling applications to such system, a challenge to limit the power consumption of the system is considered as one of the biggest challenge for an exascale system. As per the report, with the current technology, it will require a nuclear power plant that will supply GigaWatts of power to satisfy the power requirement of the system. To make an operating cost of the exascale system reasonable, after the study, DARPA came up with 20MW as the power limit for the exascale system. It is also considered as the power wall. Although it is not the strict limit, achieving about 50 GFlops/Watt is about 16 times more energy efficiency than the current top system in the gree500 list [4]. It can also be seen that the design of future supercomputers will be driven by the power consumption of its components.

To achieve the required energy efficiency for the current and next generation of the HPC systems, many alternatives are being explored. Use of accelerators, DSPs has become common in the list of top 500 systems [5]. However, HPC market is still very small and majority of the investments in the chip manufacturing is driven by the desktop systems. In the last decade, there was a huge revolution in the embedded systems market due to smartphones, tablets and other embedded devices. Market of such System on-Chip (SoC) devices is growing by leaps and bounds every year. Due to limited amount of power supply, embedded systems are designed to provide a performance with extreme energy efficiency. Use of such energy efficient SoCs in the field HPC as well as server market is now widely being considered. IBM Blue Gene series

is a good example which showed the potential and suitability of the embedded systems for the HPC systems. It is based on the PowerPC 440 architecture used in Blue Gene/L, which was originally designed for the embedded market [6]. Latest system Blue Gene/Q is well known for its energy efficiency. 19 out of top 25 systems in latest Green500 list are Blue Gene/Q.

On the other hand, ARM architecture based processors are leading the embedded devices market of the smartphones and tablet processors. ARM based processors are famous for their energy efficiency. Earlier ARM architecture based processors were 32 bit, but the latest architecture ARMv8 is now support 64 bit processors [7]. ARMv8 architecture based chips are not yet in the market. AMD will be releasing 64 bit ARM architecture based Opteron processors early next year for the microserver market [8]. It can be seen that the SoCs and embedded device market will influence the architecture of future server processors. These energy efficient solutions like ARM based processors are also being considered to address the power challenge for the HPC systems. ARM based processors has a major benefit over Blue Gene that they are cheaper than the Blue Gene processors. Reason behind it is that the investments and research efforts required for designing new processor architecture is supported by the growing market of mobile and smartphones. Use of ARM processors in HPC system is now beyond the theory and idea. Mont-Blanc project from Barcelona Supercomputing Centre (BSC) is aiming to build a system providing exascale performance using ARM based processors and accelerators, and with 15 to 30 times better energy efficient than the current systems [9]. Latest prototype cluster of this system is being tested from the month of July this year [10].

In this dissertation project, we will try to investigate the role of low powered SoC based processors in the future HPC systems. For this purpose, ARM architecture based processors were selected to represent the class low powered SoCs due to their popularity and prevalence in the embedded devices market. They also represent the low powered SoC from other vendors like Atom processor family from Intel. There were two types of ARM chips selected, one from the Pandaboards which are popular in the community of the developers and programming enthusiasts. Such single board SoCs are much cheaper than the desktop or laptop computers. Hence they are widely used to carry out different experiments with the ARM based processors. Other type was selected from the Boston Viridis project which was a server class solution, built using the ARM based processors [11]. To represent the class of processors from the conventional HPC systems, processor was selected from the Intel Xeon family which also powers the current top system [12]. Comparison was made on the basis of the performance/Watt using the benchmarks that are designed to test the performance of processors for the scientific applications. Results of these benchmarks were used to understand and predict the performance and efficiency of the system for the real scientific applications. These predictions were also validated from the performance of the applications MILC (MIMD Lattice Computation) and AMG (Algebraic Mutigrid).

Along with the objective of analyzing the energy efficiency of the SoC based processors, this project had a few more aspects which influenced the goals and deliverables of it. This project was aimed to study and understand the novel architecture of a Boston Viridis project. Viridis server was world's first hyperscale server based on the ARM architecture based processors [11]. As part of this project, the cluster from the Pandaboards was also built. All the tests, benchmarks and applications that ran for the Viridis and Intel cluster, were also ran on the Pandaboard cluster. Comparison between the Pandaboards and the Viridis systems was done to understand the areas that impact the performance of the ARM based processors. Comparison with the Intel Xeons was done to understand the performance requirements of the existing HPC systems. Comparisons of these processors also covered the understanding of the architectures, performance, energy efficiency and their interdependency. As there was no cluster management suite available for the Pandaboards, building a cluster from Pandaboards had an

additional benefits. It provided a great learning experience of building a cluster along with the cluster software stack from scratch.

Another aspect which majorly shaped the dissertation was the Student Cluster Challenge (SCC). It was organized by the HPC Advisory Council (HPCAC) at the International Supercomputing Conference 2013 (ISC'13). Objectives and deliverables of this project were tailored in order to accommodate maximum work done for the SCC into the dissertation work.

Work done in a group by a SCC team is clearly mentioned, work presented in this dissertation was carried out individually by a presenter.

## 1.1 Report Organization

**Chapter 2** – This chapter describes a cluster challenge background for this project. It also describes the literature reviewed for this project. It includes the information about the current industry initiatives towards the energy efficiency of system. It covers the hardware and architectural changes along with some examples from the industry.

**Chapter 3** – This chapter provides the detailed information about the hardware used for this project.

**Chapter 4** – This chapter provides the details of the architecture of all the three clusters which were built and used for the project. It is followed by the description of the software stack used and its importance in the cluster setup.

**Chapter 5** – This chapter contains the information about the methodology used to measure the performance of the clusters.

**Chapter 6** – This chapter contains the results and analysis of the performance of the clusters on the synthetic benchmarks.

**Chapter 7** – In this chapter the results of the application specific benchmarks are discussed.

**Chapter 8** – It contains the results and analysis of the performance of selected real-world scientific applications.

**Chapter 9** – This chapter describes the work done as part of the ISC'13 Student Cluster Challenge.

**Chapter 10** – This chapter contains the conclusion derived from the obtained results. It also describes the possible future work that can be done to extend the work done during this project. It is followed by the evaluation of the project and effort distribution during the dissertation period.





## Chapter 2

# Project Background

This chapter provides the information about the background of this project. Since ISC'13 Student Cluster Challenge was one of the major parts of the dissertation, a brief overview of the competition and work done for it is given here. It is followed by the summary of the work done in the previous dissertations in the domain of energy efficiency of the low power hardware. Then there is a discussion about the SoC chips along with reasons behind their low power consumption. This chapter also involves a work done in the field of energy efficiency of the processors and a project in the same area.

### 2.1 Student Cluster Challenge

At ISC'13 the Student Cluster Challenge was an event organized during 16th-20th June 2013, in Leipzig, Germany. Eight teams from the different countries around the world participated in this competition. Challenge was to build a cluster in the power limit of 3kWatts [13]. HPCC benchmark suite was used to compare the performance of the cluster. Three applications were given before the competition, GROMACS, WRF and MILC which were expected to run on the cluster. Two applications AMG and CP2K were given on 2<sup>nd</sup> and 3<sup>rd</sup> day of the competition. Performance of these applications was measured using the provided benchmarks.

Preparation work done for the competition included selection of the hardware, cluster building, configuring a software stack for the cluster and a lot of administrative work of the cluster management. These tasks were carried out by our team of four students and with the help of vendor. Boston Ltd. and Viglen were the vendors. Task of porting the MILC application and optimizing it for the cluster was carried out by the author. It involved experimenting with the different versions of MPI, libraries in order to obtain better performance. The competition and the preparation work was part of the MSc project work. It was extended after the competition for the MSc project.

## 2.2 Previous Work

There were three dissertations from the previous year identified, in which there was a work done in the field of energy efficient hardware. Current dissertation covers many topics mentioned in the future work of these dissertations. Quick overview of the work done in the previous dissertations is given below.

First one, "*Novel energy efficient compute architectures on the path to Exascale*" [14]. In this dissertation, the Tibidabo cluster built at Barcelona Supercomputing Centre as part of the Mont-Blanc project was used [15]. Tibidabo cluster was built from NVIDIA Tegra 2 SoC processors which were based on the ARMv7 architecture and low power NVIDIA GeForce GPU. Energy efficiency of Tibidabo cluster was tested using Linpack benchmark. It showed that the Tibidabo cluster achieved the energy efficiency equivalent to the 226<sup>th</sup> system in the Green500 list published in June 2012 [14]. Performance and scaling tests were conducted for a cluster using the benchmarks and real scientific codes. Results from these tests were compared against the performance of HECToR. It was interesting to see that the combination of low power CPU and GPU in the Tibidabo cluster showed better scaling and speedup results than the HECToR. For these performance and scaling tests, scientific codes from Quantum Chromodynamics (QCD), Molecular Dynamics (MD) and finance domain were selected. Energy efficiency aspect for these scientific codes was considered as extension to analysis.

In the "*Low-Power High Performance Computing*" dissertation, energy efficiency of the Intel Atom and Xeon processor families (versions not mentioned) was compared based on the performance per watt metric [16]. Xeon processors from the ECDF machine at the University of Edinburgh were used in this project. Dissertation compared the impact of different versions of GCC and Intel compilers on the performance and energy efficiency of the processors. Analysis was also made to understand the impact of the low latency interconnect on the performance and power consumption of the system. For this purpose a comparison was made between Gigabit Ethernet and Infiniband interconnects. It was seen that the use of Infiniband interconnect showed better energy efficiency compared to the Gigabit Ethernet on the CoreMark benchmark. Energy efficiency analysis for the other synthetic benchmarks and scientific applications was mentioned as a future work for this project.

Last one is "*Building and benchmarking a low power ARM cluster*" [17]. In this dissertation, a cluster was built using a single board SoCs from the RaspberryPi and Pandaboard. There was an interesting analysis made on the role of operating systems (OS) on the performance of these boards. Results showed that the specially optimized OS Raspbian gave about six time better performance than the generic Debian armel OS for the Linpack benchmark. Benchmarking results also analyzed the impact of CPU frequency scaling on the performance and energy efficiency of a board as well as the cluster. Comparison between the successive arm architectures ARMv6 and ARMv7 showed the latest version ARMv7 is more energy efficient than the older ARMv6. There was a comparison made between ARM processors with the Intel Atom and Xeon family processors. Results for the processors from Atom and Xeon families (versions not mentioned) were referred from the earlier dissertation [18]. This comparison showed that the ARM processors performed better than the Atom and Xeons on the performance per watt metric. Future work of this dissertation mentioned to run a scientific application on such low powered cluster to analyze the energy efficiency of such hardware for the HPC applications.

## **2.3 Embedded System-on-Chip and Energy efficiency**

In this section we will discuss about what System-on-Chip (SoC) is and why they are energy efficient.

System-on-Chip is a single chip which contains all the necessary components that are required for a computer [19]. They typically have more powerful processors than microcontrollers and they are also capable of running a desktop OS. They are mainly used in the electronic devices like smartphones, digital televisions and cameras. These processors are having excellent power efficiency because they are majorly used in the battery operated devices where a limited amount of power is available. Due to these restrictions on the power consumption of the processors, energy efficiency is primary focus of the chip designers. Chip designing requires a large amount of investment. Profitable and fast growing market of the smartphone and tablets is backing up the funds for the efforts towards energy efficient chip designs.

ARM processor architectures are one of the most popular SoCs that are powering about 95% of smart phones [20]. Design of these chips is based on the RISC architecture and hence requires a significantly less amount of the transistors compared to the processors in traditional systems [21]. This makes them cheaper, low powered and simpler in terms of a design. Simpler design of these chips also makes it possible to build multi-core processors at higher energy efficiency and at lesser cost. Very tight integration of the components requires less wiring which ultimately makes chips more energy efficient by reducing the power wastage and heat dissipation [21]. However, these processors have down side as well. Lower operating frequency makes the processors slower. For HPC applications, the time of execution is very important aspect. These SoCs are commonly used in the smartphone and tablet processors which run the less floating point intensive applications. This makes these SoCs less efficient for the floating point operations and makes them more energy efficient. However, HPC applications require high floating point performance. This requires the efficient floating point units (FPU) which ultimately consumes more power [21]. Compared to the commonly used x86 based processors, SoC are less flexible in terms of changing or upgrading a particular component of the system like CPU, GPU and RAMs. Hardware upgrades for the supercomputers is common activity which generally involves upgrading CPUs and memories. These issues raise questions about the suitability of such SoC based devices for the HPC systems. In spite of all the issues, principles behind the SoC designs have potential to provide a better performance with greater energy efficiency. This can be sensed from the case study of Blue Gene/Q chip in the next section.

### 2.3.1 Case Study of Blue Gene/Q Chip

Blue Gene is a project of IBM aimed to build the powerful supercomputers with low power consumption. Blue Gene/L, Blue Gene/P and Blue Gene/Q are the three generations of Blue Gene series who consistently dominated the Green500 list by their excellent power efficiency from past several years [6]. Blue Gene family of supercomputers has won National Medal of Technology and Innovation in 2008 [22].

**Figure 2.1** shows the architecture of Blue Gene/Q Compute (BGC) Chip. It is based on the PowerPC A2 processor core.

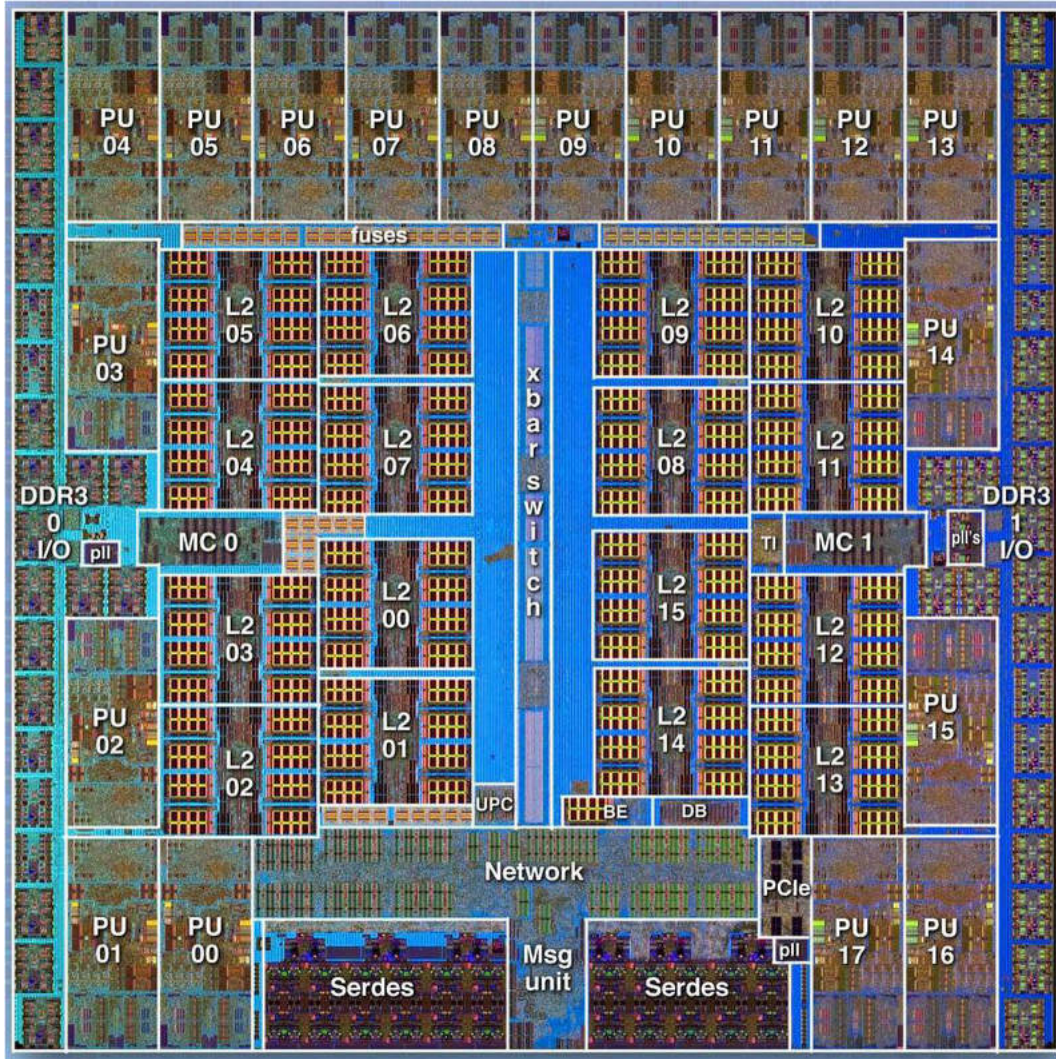


Figure 2.1: Blue Gene/Q Compute Chip Architecture(Image reproduced from [23])

Each A2 core is four-way simultaneously multithreaded (SMT). There are 18 cores on chip while user is exposed to only 16 out of them. One core is dedicated run an OS while other one is redundant. This redundant core is activated if manufacturing defect is found with any other core during testing. Activation of core is done by making changes to the on-card EEPROM, which contains information about active cores that is used by the OS while booting. This A2 core runs at reduced voltage of 0.8 V nominal at 1.6 GHz. This low voltage and operating frequency reduces both the active and leakage power which ultimately contributes towards the energy efficiency of the core [23]. This SoC design has an exception of memory which is not

on the chip. It is soldered on the compute card to leverage similar benefits.

Below **Figure 2.2** BGCPackaging shows the packaging of the Blue Gene/Q compute card [22].

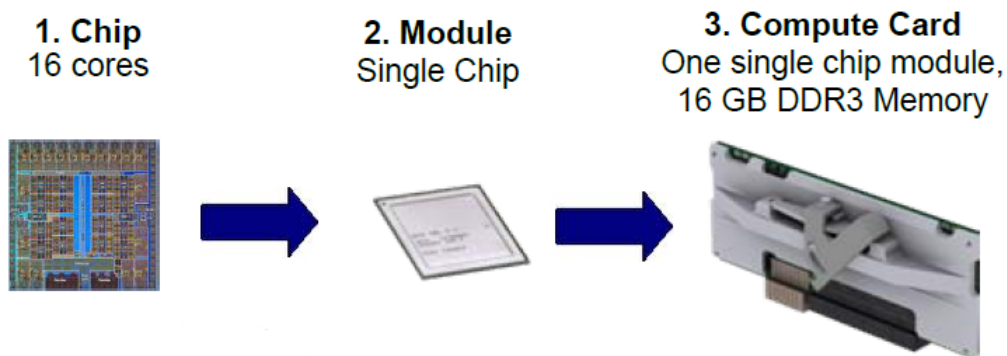


Figure 2.2: Packaging of Blue Gene/Q Compute Card(Image reproduced from [22])

Compute card shown in the above figure is having 16 GB of DDR3 memory, which operates at 1.35 V. Such packaging of memory and card has benefit of short interconnects. This makes the compute cards more energy efficient. This arrangement has additional benefit that the data nets of the compute card are left unterminated which improves the reliability of the compute card [22].

Along with the A2 cores, BGC chip is having additional hardware of L1 prefetch unit, WakeUp unit and Quad Floating Point Processing Unit. They are replicated for all 18 cores. These units are one of the major contributors for the BGC chip's higher performance by maintaining energy efficiency [23].

First is L1 prefetch unit (L1P), it tries to hide the latencies of accessing L2 cache by prefetching. The new list prefetching anticipates the memory accesses from the arbitrary sequence and reduces the cache misses. This is optimized for the QCD applications. WakeUp unit is present for each core on a BGC chip. WakeUp unit saves the clock cycles wasted by a tread while polling and improve the overall application performance [23]. Finally, quad floating point processing unit. It is based on the Quad Processing eXtension (QPX) added to the Power ISA. It makes each processing core capable of performing maximum of eight double precision floating-point operations per cycle [23].

In conclusion, the principles of low powered SoC based design used in Blue Gene/Q has delivered industry leading energy efficient HPC systems. The success of Blue Gene/Q systems shows that the current innovations and efforts of using SoC based solutions are capable of powering the future HPC systems with a greater energy efficiency. One the selected system for this MSc project is the ARM based Viridis system. It is SoC based and has similar dense architecture of the compute blade. Analysis of the architecture of both the will be beneficial to understand the areas of improvement, for the ARM SoC based processors in the performance domain.

## 2.4 Current HPC systems and Energy Efficiency

In this section a discussion made on the energy efficiency of the architecture of the current HPC systems and current progress made by them towards the energy efficiency. It covers the use of accelerators in today's systems and reasons behind their power efficiency.

Current HPC systems are dominated by the x86 architecture based processors. **Figure 2.3** below shows the architectures of the current top 500 systems (as per June 2013 Top500 list). It clearly shows that about 90% of top 500 systems contain x86 architecture based processors [24].

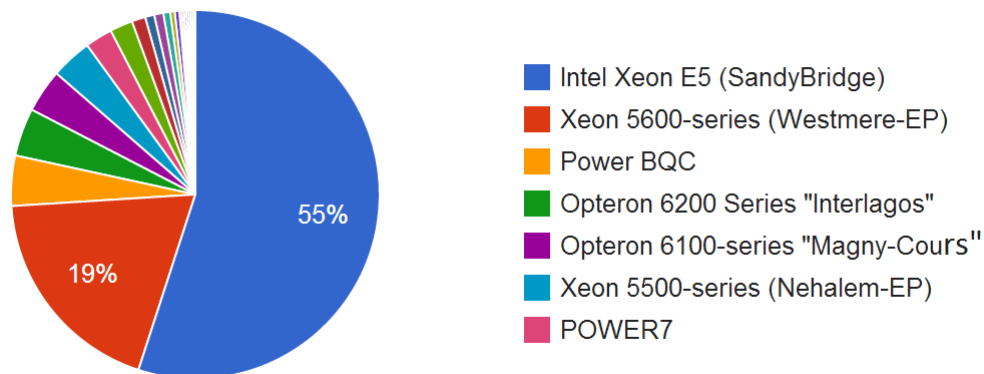


Figure 2.3: Processor Generation in the Top500 supercomputers June 2013 (Image reproduced from [24])

Question is always raised, why x86 based systems are not as energy efficiency as that of SoC based systems like ARM. One of the reasons behind this is that the performance of processor was the primary focus behind the design and evolution of the x86 processors. Another reason is that the x86 processors have very complex and large instruction set, which also maintains the backward compatibility. This causes the instruction translation system to occupy more space on the die. They have deep instruction pipelines and other circuitry like out-of-order execution, branch prediction or speculative execution. This makes the x86 based chips to have less area dedicated for the compute units. These units are also responsible for the better performance of the x86 processors. However, this performance gain is at the cost of a space on the die and more power consumption. On the other hand, SoC based systems like ARM are RISC based and having comparatively simpler instruction set [25]. They are having shallower pipelines. As an example, ARM Cortex-A9 have 8-stage pipeline, Intel Atom has 16-stage pipeline compared with upto 31-stage pipeline of Intel Pentium 4 [25, 26]. On the other hand, ARM Cortex-A9 does not have simultaneous multithreading (SMT), which gives simplicity of design at the cost of performance [25].

Now there are a huge amount of efforts made towards the energy efficiency of the x86 based processors. They are using RISC-like core design while ARM instruction set is getting complex. From Cortex-A9 ARM also started supporting out-of-order execution. These changes made it difficult to categorize systems on the basis of RISC and CISC architecture [25]. Intel improved the energy efficiency of its x86 based processors by using efficient decoding stages, which also reduced the pipeline stages to 16 for the Nehalem microarchitecture [27]. New approaches like dynamic voltage/frequency scaling (DVFS) has improved the energy efficiency of the x86 processors. In this technique, clock frequency of CPU is increased when processor is heavily used and vice-versa when it is idle [28]. These all enhancements made the modern x86 processors energy efficient along with the performance improvements.

It is seen that the simpler core design and more area for the compute units on a die are the success factors for the energy efficiency. However, sophistication of the x86 processors is required to run complex software like an operating system. This introduced the new era of accelerators. They are used along with the processors to achieve a better performance with energy efficiency.

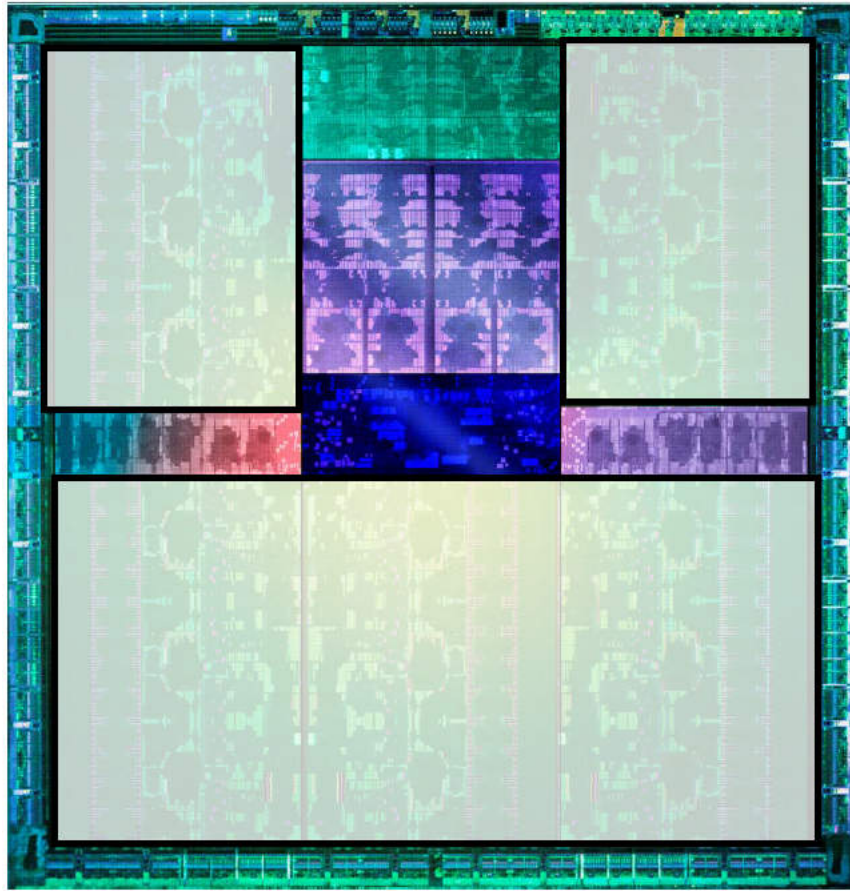


Figure 2.4: Kepler GK110 Die Photo (Image reproduced from [29])

Highlighted area in the **Figure 2.4** shows the area given for compute units on the Kepler Gk110 GPU die [29]. Accelerators like GPUs have large number of simple cores. GK110 in the K20 GPU has 2496 compute cores [30]. These cores are less sophisticated than the CPU cores. They do not have branch prediction units, they have less scheduling units. This allows them have more die area for the compute units to have a simple and energy efficient design. However, this makes GPUs unable to run on their own, they need to be used along with the CPUs. In this hybrid approach a key computational kernel is executed on the GPU to get performance boost while the other part of application runs on a CPU. It makes such hybrid architectures to provide better performance and greater energy efficiency [31]. This can be seen from the fact that the current top three most energy efficient systems have GPUs [4].

**Figure 2.5** shows the architecture of the streaming Multiprocessor (SMX) in the Kepler architecture based GPUs.

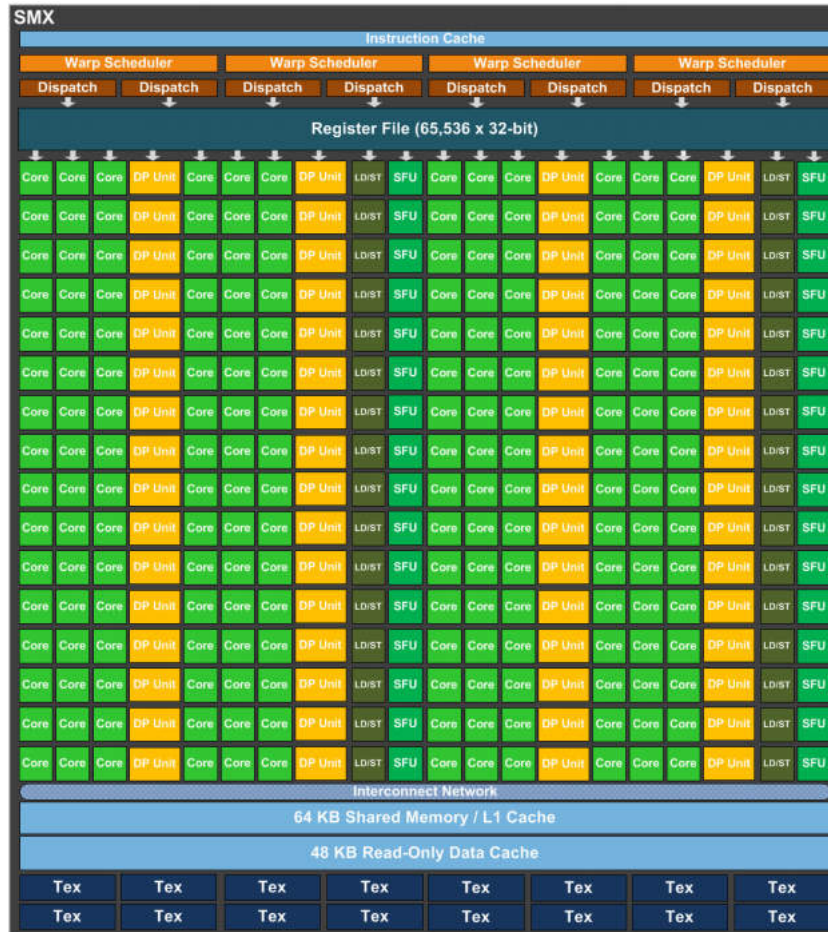


Figure 2.5: Kepler Streaming Multiprocessor in the Kepler architecture based GPU (Image reproduced from [29])

SMX is the basic building block of the NVIDIA GPUs [29]. In these NVIDIA GPUs a group of 32 cores is called as warp which executes a single instruction in lock-step mode. From the figure it can be seen that there are 192 cores capable of performing single precision operation, but there are only 64 double precision units, eight instruction dispatch units and four warp schedulers. Each SMX is having 64 KB of configurable memory that can be distributed as shared memory and L1 cache. Graphics memory provides very high memory bandwidth. These cores operate at low frequency. This architectural simplicity makes these GPUs to have more die area allocated for compute units which ultimately results in a better performance to power ratio [29].

However, these accelerators have down side as well which limits the benefits that can be achieved from them. Data for the computation needs to be offloaded to the accelerator memory. It happens on the slow PCIe bus. This often acts as a performance bottleneck for the applications that are requiring more such transfers between CPU and accelerator memory. They often require application to be developed in the programming models designed for them [31]. Intel Xeon Phi is an exception here [32]. It is a bit difficult task to maximize utilization of the available hardware and get the benefits that accelerators are capable of giving [31]. There are a lot of efforts are being done to address these issues. In spite of all these obstacles, accelerators have proved their important role to achieve greater energy efficiency.



## 2.5 Current HPC Projects towards Energy Efficiency

We have seen that the SoC based processors are capable of providing the energy efficient HPC systems. It was also seen that the use of accelerators for computationally intensive part can provide a better performance with the energy efficiency. There are some initiatives taken in the HPC community to achieve synergy between the low powered CPUs and accelerators to address the power challenge for the HPC systems. In this section we will discuss the Mont-Blanc project from Barcelona Supercomputing Centre (BSC) and various clusters built as part of it. In this, we will take a quick overview of the architectural choices made for these clusters to achieve the goal of energy efficiency.

### 2.5.1 Project Mont-Blanc

Mont-Blanc is the project funded by European Commission. Its purpose is to design and build a new type of computer architecture for the future HPC systems, and deliver an exascale performance with greater energy efficiency [9]. This project has three main objectives as below [9].

- By 2014, develop the energy efficient prototype using low powered embedded technology.
- Design a next generation of HPC system by 2017, by overcoming the limitations identified from the previous prototype.
- Develop portfolio of exascale application to run on this system.

**Figure 2.6** shows the roadmap of the ARM based prototypes from Mont-Blanc project. As part of Mont-Blanc project, first prototype of this series, "Tibidabo" was developed in 2011. It is having 256 node containing ARM based processors. These nodes had Q7 carrier modules developed by SECO that were containing NVIDIA Tegra 2 SoC [33].

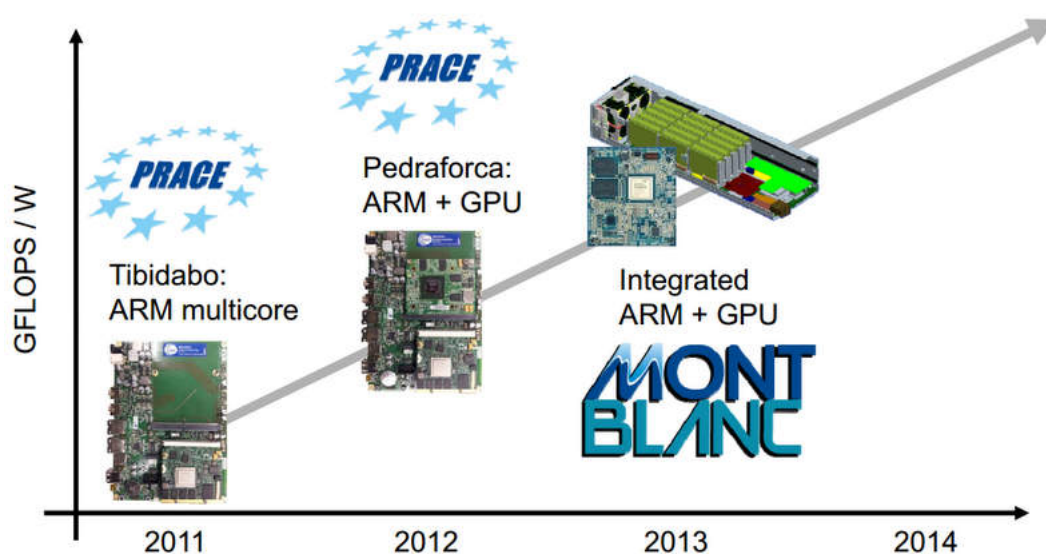


Figure 2.6: Road map of the ARM based prototypes (Image reproduced from [33])

Figure **Figure 2.7** shows building blocks of the Tibidabo cluster. Each node has a quad core ARM Cortex-A9 processor and Ultra Low Powered (ULP) GPU embedded in it. Two racks of this cluster delivered the performance of 512 GFLOPS from 3.4 kWatt, which is 0.15 GFOLPS/Watt [34].

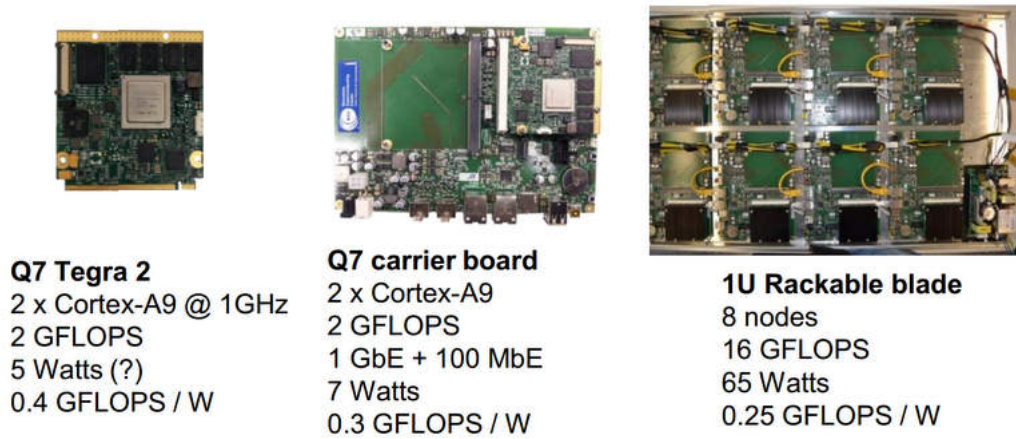


Figure 2.7: Building Blocks of Tibidabo Cluster (Image reproduced from [34])

Tibidabo cluster was designed on the concept to achieve energy efficient system by using power efficiency of the embedded SoC based processors. However, ULP GPUs supported only OpenGL, no CUDA was supported which made the applications unable to use GPUs for computation. It was also having an issue of power wastage of  $\sim 7$  Watts per container due to arrangement of the boards [35].

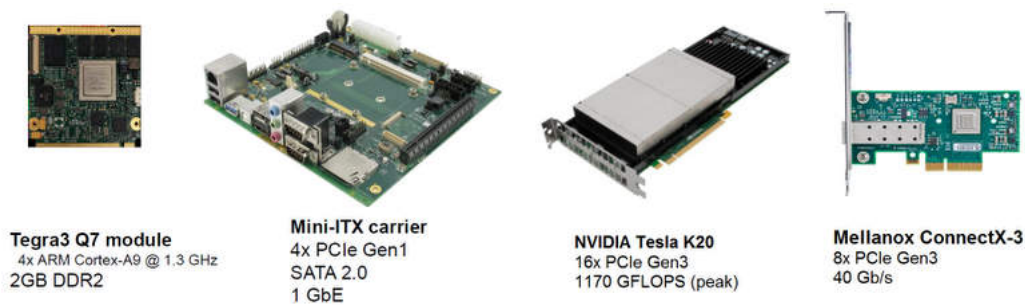


Figure 2.8: Building Blocks of the Pedraforca Cluster (Image reproduced from [36])

Second prototype of the project, "**Pedraforca**" deployed in June 2013 [36]. Figure **Figure 2.8** shows the building blocks of the Pedraforca cluster. Each node contains Tegra 3 Q7 module with quad core Cortex-A9 processor. It has one of the leading interconnect, Infiniband with a 40 Gb/s bandwidth. Interestingly, each node also contains NVIDIA Tesla K20 GPU. It is a server class GPU providing 1.17 TFLOPS of peak performance. Pedraforca is having 64 such nodes [34].

Using powerful GPUs it is clear that the Pedraforca cluster is trying to achieve a better energy efficiency. Results for the Pedraforca cluster are not yet published, but they are expected to show higher performance per watt ration than its predecessor. On the other hand, architectural choice of using server class GPUs with low powered CPUs will require the applications to have more GPU intensive computation in order to achieve better energy efficiency. For CPU intensive applications this cluster might not be an appropriate choice.

Design of the next prototype cluster in the Mont-Blanc series is heavily influenced by the current market condition. ARM based processors are lagging the x86 based processors in performance by about eight times, but they are two orders of magnitude cheaper than the x86 based processors. Due to huge growth in the smartphone market, performance gap between x86 and ARM based processor might reduce to 1:2 while the ARM processors will get cheaper. In

the performance domain, integrated ARM GPU is expected to grow faster in the future [35].

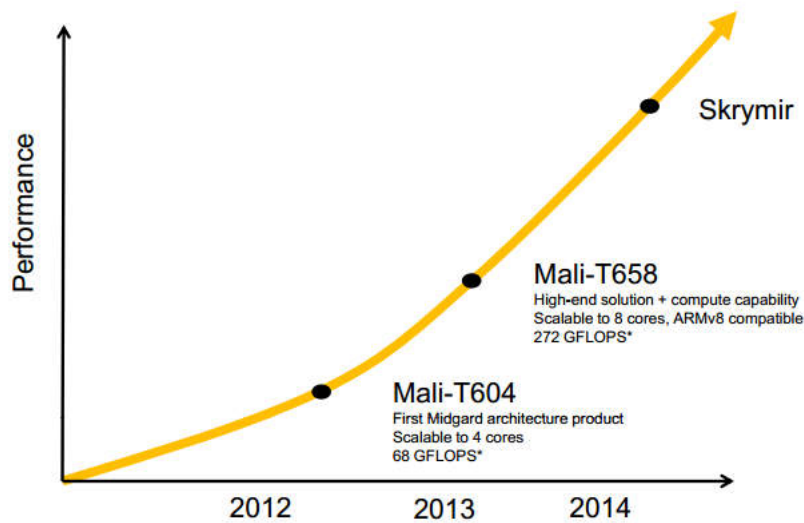


Figure 2.9: Expected performance growth of ARM Mali GPU (Image reproduced from [35])

As per predictions in Figure **Figure 2.9**, Mali-T604 delivered the performance of 68 GFLOPS for the single precision calculation. Therefore hybrid architecture containing Exynos 5 CPUs and a Mali GPU has been selected for the next Mont-Blanc prototype. Exynos 5 is ARM Cortex-A15 based dual core CPU operating at 1.7 GHz [35]. Mali T604 is having four GPU cores called the **Shader cores** [37].

**Figure 2.10** shows the building blocks of the next prototype cluster in the Mont-Blanc series. Similar to its predecessor Pedraforca, this cluster is also using combination of the CPU and GPU to achieve the better energy efficiency.

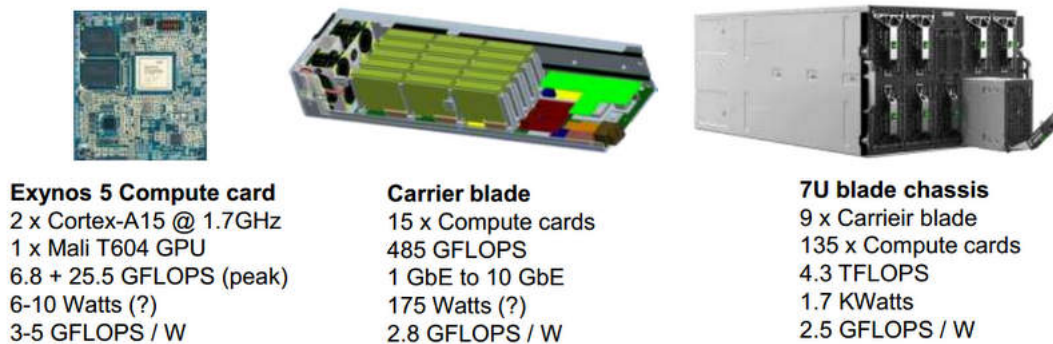


Figure 2.10: Building blocks of next Mont-Blanc Prototype (Image reproduced from [36])

This time a compute card will contain two dual core Cortex-A15 processors and integrated Mali T604 GPU. Selection of integrated GPU has interesting architectural benefits. One of them is that the CPU and GPU share the same memory. This avoids the data transfer over PCIe bus which is well known bottleneck for the traditional GPUs. Additionally, integrating GPU makes it cheaper as well as more energy efficient [37]. Mali T-604 provides a support for OpenCL 1.1 which makes it possible to develop applications leveraging GPUs. Although these GPUs are much less powerful than the previous K20s, they are maintaining the required energy efficiency by their less power consumption. Cortex-A15 processor contains a Vector Floating

Point (VFP) coprocessor which supports double precision floating point operations [38]. Each compute card will have 4GB DRAM to support bigger problem size per node [37]. With all these features the next prototype cluster is expected to provide better energy efficiency than its predecessors.

In conclusion, by realizing the benefits of using SoC based embedded devices, Mont-Blanc project developed various innovative prototypes. As per the goal of the project, each prototype learned from its predecessor and achieved the better energy efficiency than it. Delivering the exascale system on the planned schedule is another aspect, but the results delivered by the prototypes certainly indicate the positive signs of using the SoC based embedded technology in the future HPC systems.

## 2.6 Summary

This chapter provided the overview of the ISC'13 Student Cluster Challenge which was part of the dissertation. Significant amount of the dissertation work was done for the preparation and participation in the competition.

In this chapter we discussed the SoC based approach and reasons behind its energy efficiency. Case study of Blue Gene/Q showed the potential of the SoC based technology to deliver the energy efficient HPC systems in future. We took an overview of the efforts made to improve the energy efficiency of current HPC systems. It was also seen that the accelerators played key role in achieving the better performance per watt ratio. In the discussion over the architecture of current accelerators some challenges were identified which include the different instruction sets, different programming models, transfer of data between CPUs and accelerator. These challenges are found to be getting addressed by the current initiatives towards energy efficiency. Discussion on the Project Mont-Blanc showed it. It was seen that there are efforts going on to improve the energy efficiency by using a SoC based technology. Evolution of the clusters indicated the use of low powered CPU and low powered GPU on a same die for higher energy efficiency. This shows that there is a huge possibility of the embedded SoC based processors in the future HPC systems.

# Chapter 3

## Project Hardware

In this chapter we will discuss about the hardware used in this project. Discussion will be made on the architecture and other features that are impacting the performance of the applications running on it.

### 3.1 Hardware Selection

In the previous chapter we saw the benefits of SoC based processors and possibility of their usage in the future HPC systems. Role of the accelerators and benefits of having both of them on a same die were also noticed. However, about 90% of the systems in Top500 list do not contain the accelerators [24]. This shows the importance of CPU only systems in the current and future HPC systems. Considering this scenario a focus is given on comparison of the energy efficiency of the current SoC based processors with the x86 based processors. Availability of the hardware and background of cluster competition also played very important role in the hardware selection.

For this project there are three types of processors selected from the available hardware. They are, a single-board ARM processor in the Pandaboards, server class ARM processor in the Boston Viridis system and the x86 based Intel Xeon processor.

### 3.2 Hardware details

#### 3.2.1 Pandaboard ES

Pandaboard ES is a single board SoC manufactured by the Texas Instruments [39]. In this project latest version of the Pandaboard is used which contains OMAP4460 SoC as shown in **Figure 3.1**. It has ARM Cortex-A9 processor which is based on ARMv7 instruction set. OMAP4460 is Dual-core processor and operates at 1.2 GHz. It has slightly higher frequency than its predecessor, but it is still low to maintain the lower power consumption of the system. Both the cores share a 1 MB of L2 cache and 1 GB dual channel LPDDR2 RAM [40]. LPDDR2 stands for a Low Powered DDR2 RAMs which are specifically made for the low power consumption and higher memory bandwidth. They have reduced supply voltage from 2.5V to 1.2V compared to DDR2 ones. They use temperature-compensated refresh approach due to which at low temperature it refreshes less often [41].

Pandaboard has onboard 10/100 Mbps Ethernet port. It is comparatively small in capacity and consumes less power. However, it is potential bottleneck for the communication intensive applications. Pandaboard has onboard slot for the high speed SD card and it allows the SD cards up to 32 GB. It also has 2 USB 2.0 ports through which other devices like SSDs, keyboards can

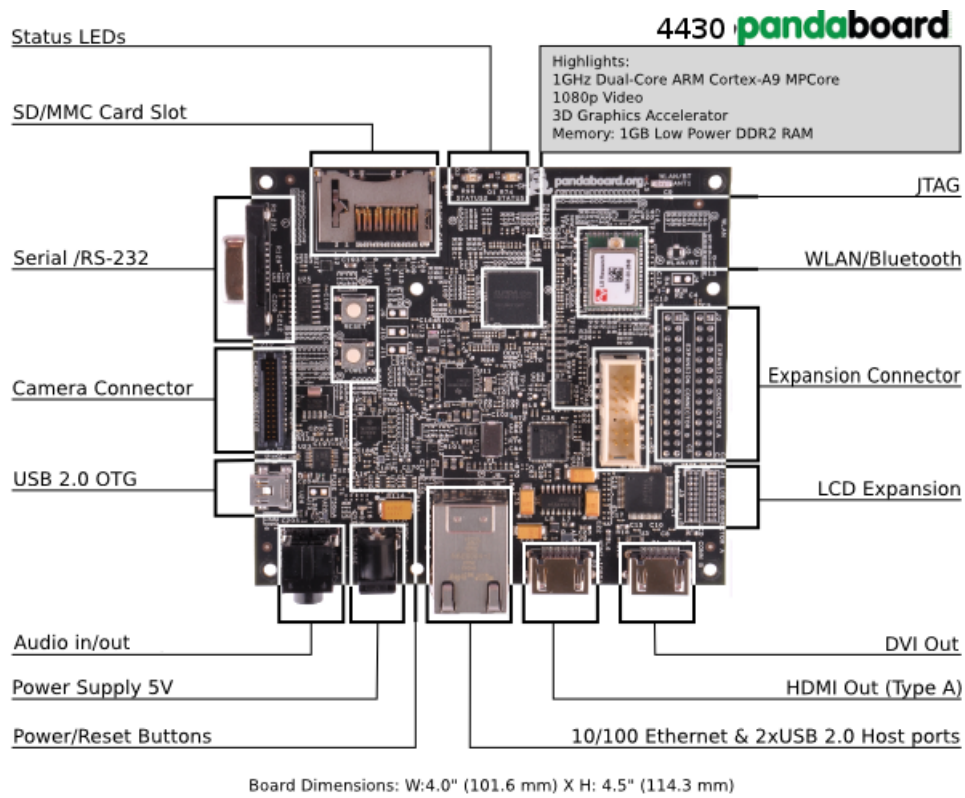


Figure 3.1: Pandaboard ES 4460 and its Components(Image reproduced from [42])

be connected to the board. As this OMAP processor was developed for the mobile devices, it also has support for audio, camera, Bluetooth and Wi-Fi [40]. These features are not used during this project.

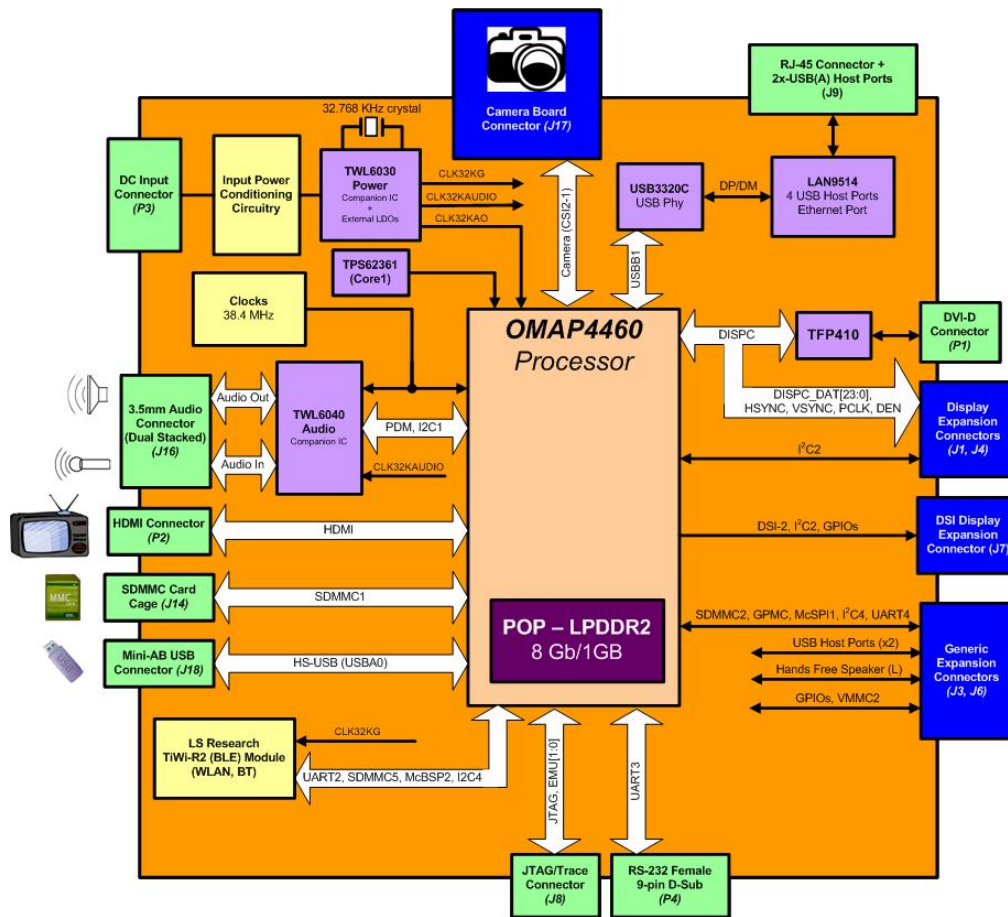


Figure 3.2: Pandaboard ES 4460 Block Diagram(Image reproduced from [42])

Cortex-A9 processors in Pandaboard are 32-bit processors and such processors can have maximum of four cache coherent cores. They can operate between 0.8 GHz to 2 GHz. Cortex-A9 processors supports Advanced SIMD extension, it is also known as NEON extension. They have 128-bit vectors but executes with 64-bits at a time. This extension is useful to improve performance of the applications by vectorization. NEON extension is optional in Cortex-A9 and it uses same registers as Vector Floating Point (VFP) unit [21].

VFP unit was initially introduced to support a short vector mode, but it operated on each vector sequentially. Hence they did not give the real benefits of SIMD vector parallelism [21]. It shows that specifying the VFP support will not be useful while optimizing the applications or benchmarks.

### 3.2.2 Boston Viridis

Boston Viridis is the first ARM architecture based server solution containing SoC based processors, networking and IO which is fully integrated on a single server chip. It is one of the products from the Boston Viridis project. Viridis project prepared the solutions mainly for the web servers, cloud and data analytics market [11]. It has novel architecture which is discussed in this section.

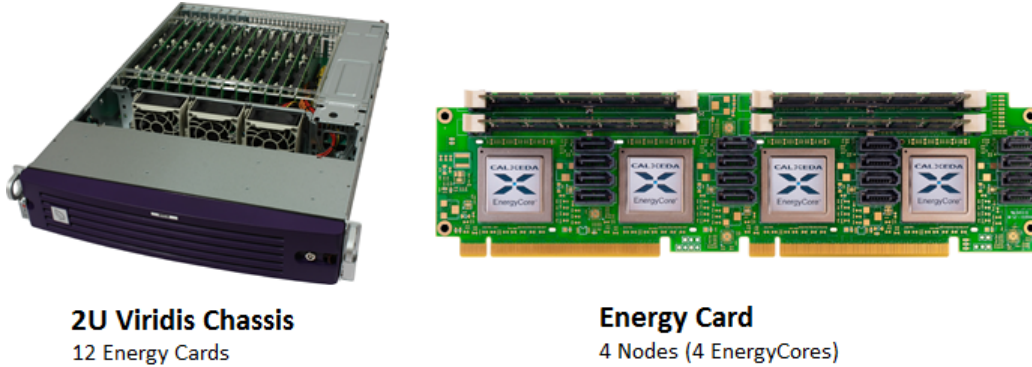


Figure 3.3: Viridis Chassis and Viridis Energy Card (Image reproduced from [43])

**Figure 3.3** shows the Viridis Chassis which contains the twelve energy cards into it. Each Energy Card contains four server nodes. Each server node is a quad core ARM based processor manufactured by Calxeda. This makes each energy card to have four server nodes and 16 cores, and a chassis to have 192 cores in total. It is very interesting to see that the power consumption of the whole chassis is ~300W which makes it very different from the conventional servers in the market [43].

Each Viridis server node contains the Calxeda EnergyCore processor, shown in Figure **Figure 3.4**.

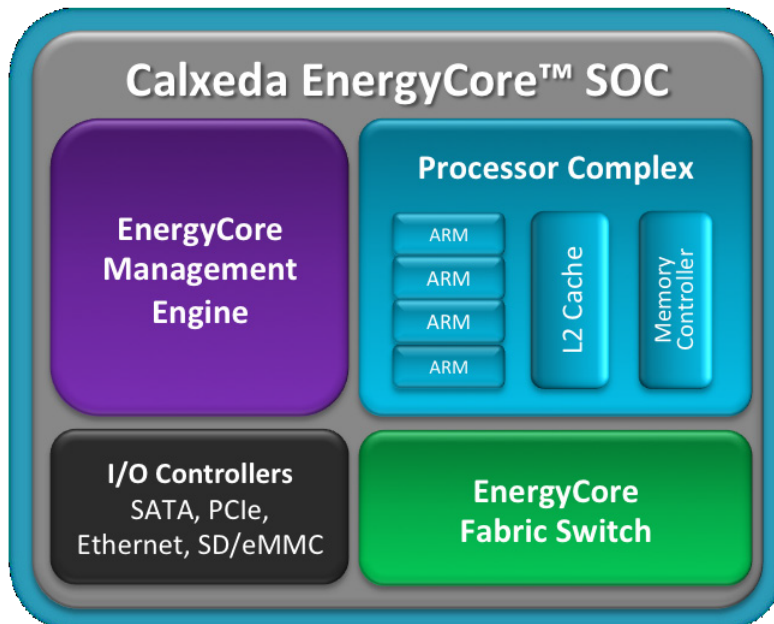


Figure 3.4: Block Diagram of a Viridis Node containing Calxeda EnergyCore (Image reproduced from [43])



Each server node has a quad core ARM Cortex-A9 processor operating at 1.4GHz. They share the 4MB of L2 cache. Each node has 4GB of DDR3 DRAM connected via miniDIMM connector. Cortex-A9 based core in Viridis node is same as that of the core in the Pandaboard ES processor. Except the core count, both Pandaboard and Viridis processors share the same other architectural features of the VFP unit and NEON extension.

Viridis nodes are having higher capacity network onboard, it is 10Gb Ethernet. Interesting feature of this system is that each Viridis node connects to every other Viridis node through 10GbE fabric. Onboard EnergyCore fabric switch handles the Ethernet switching part. It has bandwidth of 80Gb. Use of such switch removed the requirement of an additional switch which ultimately saves the power consumption of the system [43].

Another interesting feature is the EnergyCore Management Engine. Each Viridis node has this management engine which contains a separate, dedicated processor that is responsible for optimizing the energy efficiency of a node. It turns off the power of various components when they are not in use. This engine also performs the additional tasks of optimizing routing in the fabric by avoiding the failed and congested routes. It also provides an interface for the remote system management such as Intelligent Platform Management Interface 2.0 (IPMI 2.0) which is used in this MSc project [43].

### 3.2.3 Intel Xeon

The third type of processor selected for this project is the Intel Xeon E5-2670. Intel Xeon is a family of multi-socket capable x86 based processors targeted for the workstation and server market [44]. About 55% of the current top 500 systems have processors from the Xeon E5 family [24]. Intel Xeon E5-2670 are based on the Intel Sandy Bridge microarchitecture. They have eight processing cores operating at 2.6 GHz and capable of supporting 2-way SMT (called as hyper threads or HT). Each processing core has 32 KB of separate instruction and data cache along with 256 KB of L2 cache. These cores share the 20 MB of L3 cache [45]. These processors have two load units which are capable of performing two 128-bit loads per cycle. Using these load units the AVX unit in a processor can perform two floating point operations per cycle [46]. AVX stands for the Advanced Vector extension which is an extension to the x86 instruction set to execute SIMD instructions. It is the next version of the Streaming SIMD Extensions (SSE) [47]. These processors have TDP of the 115W at the peak performance [45]. TPD is the thermal design power. It is the maximum amount of the power the cooling system for a computer needs to dissipate. Processors are never expected to cross this limit even at their peak [48]. In the Xeon family many processors are provided by Intel with slight change in their specification. Further discussion is focused on the details of Xeon E5-2670 from the Xeon E5-2600 family of processors that are used for this MSc project.

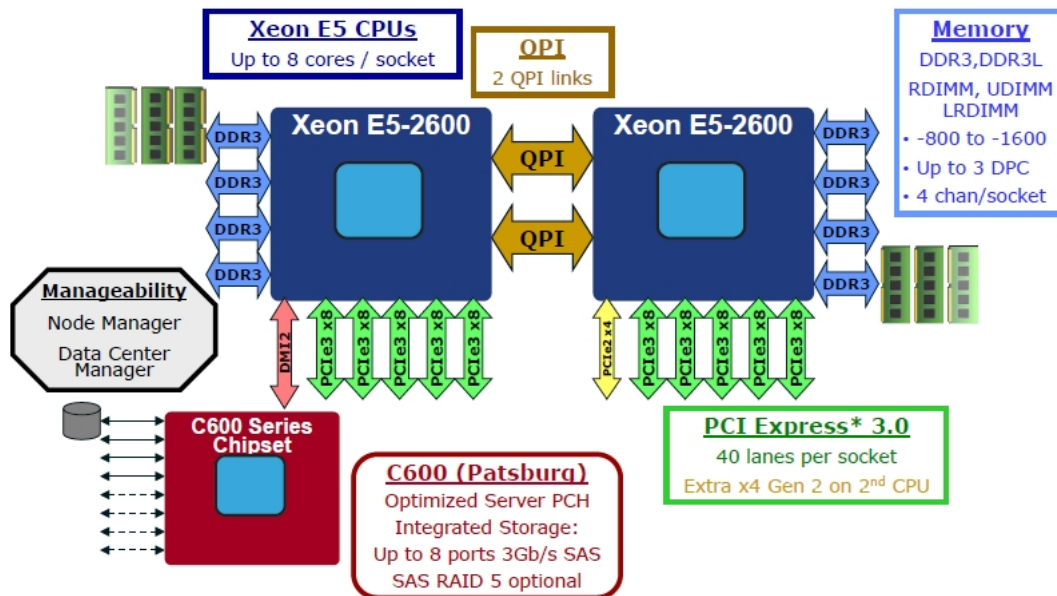


Figure 3.5: Compute Blade with Dual Socket Intel Xeon E5-2600 series Processors (Image reproduced from [48])

**Figure 3.5** shows the block diagram compute blade used for this project. Each compute blade had two sockets containing Intel Xeon E5-2670 processors connected through two Intel QPI (Quick Path Interconnect) links with the speed of 8 GT/s. Each node had 64GB of DDR3 RAM. Infiniband Quad Data Rate (QDR) interconnect from Mellanox was used to communicate with the other nodes. 4xQDR provided bandwidth of 32 Gbps with extremely low latency of 1.29 microseconds [49].



Figure 3.6: Compute Blade used at ISC'13 SCC (Image reproduced from [48])

**Figure 3.6** is showing the compute blade of type 1027GR-TRF from Supermicro [48]. Four compute blades of this type were used in the cluster built for the ISC'13 SCC. This project used two such blades.

### 3.3 Table of Comparison

Specification	OMAP 4460	Calxeda EnergyCore	Intel Xeon E5-2670
Core	2	4	16
Clock Freq.	1.2 GHz	1.4 GHz	2.6 GHz
Instruction Set	ARMv7	ARMv7	x86-64
L1 Cache	32 KB Instruction 32 KB Data	32 KB Instruction 32 KB Data	32 KB Instruction 32 KB Data
L2 Cache	1 MB (shared)	4MB (shared)	256KB per core
L3 Cache	NA	NA	20 MB
RAM	1 GB LPDDR2	4 GB DDR3	64 GB DDR3
FLOP per cycle	1	2	32

Table 3.1: Details of hardware specifications of the three types of processors

As discussed in the section 2.3 that the SoC based cores are very energy efficient for the integer calculations but they are having a poor floating point performance [21]. It can also be seen from the comparison in the **Table 3.1**. Cortex-A9 core is capable of executing one floating point instruction per two clock cycles. This makes Pandaboards to perform one FLOP per clock cycles due to two processing cores. Similarly two FLOPs performed by the Viridis node. On the other hand, each core in the Xeon E5-2600 series is capable of performing two FLOPs. Which make each Intel node to perform 32 FLOPs per clock cycle [21]. This is expected to impact the performance of the ARM based processors for the scientific applications.

### 3.4 Summary

From the hardware specifications it is observed that the Pandaboard and Calxeda nodes are computationally less powerful but they are having less power consumption as well. They use memory, network and specialized hardware for reducing the power consumption. Calxeda nodes are having a larger memory which support a bigger problem sizes and they are having fast networking with 10Gb Ethernet. Comparing with the Pandaboards it seems that the Calxeda nodes have addressed the performance bottlenecks of memory, networking and core count to run a server class application. On the other hand, Intel Xeon processors are very powerful and Infiniband interconnect contributed towards their performance by reducing the communication latency. It was also seen that the Intel Xeon processing cores are capable of performing four times more floating point operations than the ARM cores. This will make the Intel Xeons to show higher performance for the scientific applications.

Intel Xeons are clear winners in the performance domain but at the cost of power consumption. This will make it interesting to see how the Intel Xeon processors perform on the performance per watt metric compared to the ARM SoC based low powered processors.

# Chapter 4

## Project Setup

Cluster building was the significant portion of the dissertation work. This chapter gives an overview of the design and setup of the clusters built from the hardware discussed in the previous chapter. Discussion also covers the power measurement hardware and techniques used for each cluster. This chapter is divided into two parts. Section 4.1 is focused on the hardware setup. Section 4.2 describes the configured software stack and its importance for the cluster and the project as a whole.

### 4.1 Hardware Setup

#### 4.1.1 Hardware Setup Overview

Three clusters were used for this project, which were Pandaboard cluster, Calxeda Viridis cluster and Intel Cluster. Intel cluster was built in a group by the ISC'13 SCC team with the help of Boston staff. It was made available through a remote access after the competition. Remaining two clusters were built individually during the course of project, out of them Pandaboard cluster was built from the scratch. Viridis cluster had expensive hardware and different arrangement of the cards in the chassis. Therefore Viridis cluster was built with the help of staff from the Boston Ltd. This was done during the training trip at vendor location. Cluster was then made available through the remote access.

#### 4.1.2 Pandaboard Cluster

This cluster was built using six Pandaboards. One board was used as a head node and remaining five were compute nodes. Netgear GS608 was an eight port GigaBit Ethernet switch used to connect these boards. Each board was having 16 GB SD card.

**Figure 4.1** shows a design of the cluster. It can be seen that only the compute nodes were connected to the power meter. **What's up Pro ES** power meter was used for this cluster [48]. Panda1 to Panda5 are the five compute nodes and Pandahead is the head node.

Power consumption of the switches is also one of the important parts of the today's HPC system. However, this project was more focused towards the power consumption of the embedded SoC based processors and a board. Switch was external and not on-board. Additionally, power consumption of the switch was about 8-10W which was roughly equivalent to the power consumption of the two Pandaboards. This power consumption was always in the range of 8-10W even if the single node or whole system was used. Due to this when applications and benchmarks were scaled across the nodes, they were showing better performance per watt ratio. Calculating the power consumption of the switch would not have shown the true impact

of the scaling on the power consumption of the boards. It might also have derived the misleading conclusions about the energy efficiency of the cluster. Therefore, decision was made to calculate the power consumption of the compute nodes only.

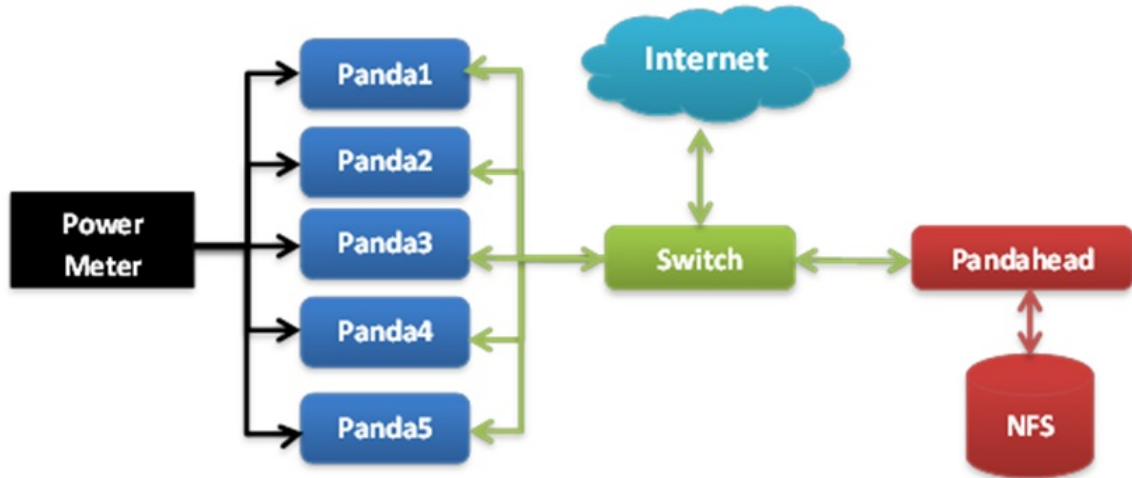


Figure 4.1: Block Diagram of the Pandaboard Cluster

Internet access was provided to the cluster from the external system (personal laptop) by sharing the internet connection through an Ethernet cable. Network File System (NFS) was configured on the head node to keep shared data and applications.

#### 4.1.3 Viridis Cluster

Two energy cards containing eight Viridis nodes were used to build this cluster. **Figure 4.2** is shows the arrangement of these components in the cluster.

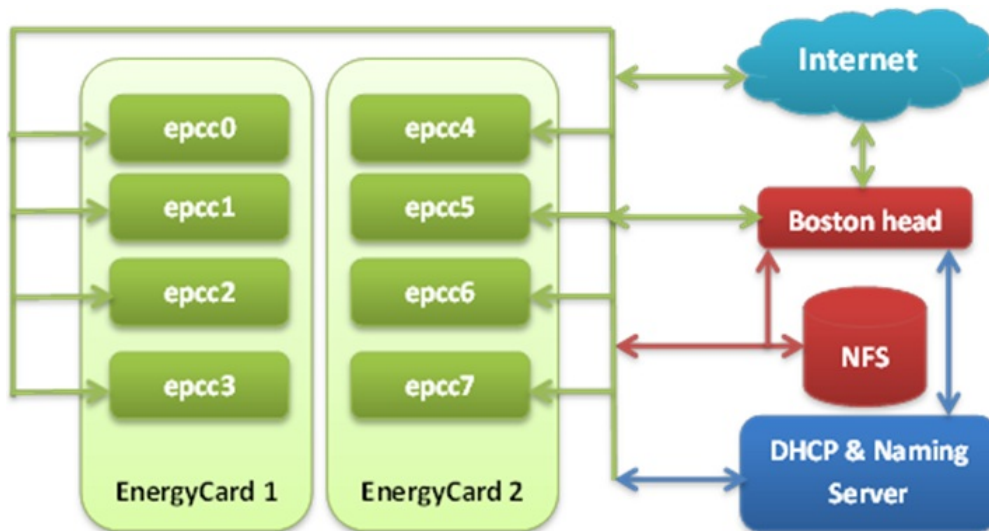


Figure 4.2: Block Diagram of the Viridis Cluster

As discussed earlier, each Viridis node has a management engine which monitors and optimizes the power consumption of the system on-the-fly. It supported a feature to manage the nodes remotely by using an Intelligent Platform Management Interface 2.0 (IPMI) service.

IPMI tool configured on the Boston head node used this service to provide the power consumption details of the Viridis nodes remotely. Sample output and command to measure the power consumption of the epcc0 node is shown below.

**Command**

```
ipmitool -H epcc0-ipmi -U admin -P admin -I \
lanplus sensor get 'Node Power'
```

**Output**

```
Locating sensor record...
Sensor ID : Node Power (0x62)
Entity ID : 19.2
Sensor Type (Analog) : Power Unit
Sensor Reading : 4.400 (+/- 0) Watts
Status : ok
Lower Non-Recoverable : na
Lower Critical : na
Lower Non-Critical : na
Upper Non-Critical : na
Upper Critical : na
Upper Non-Recoverable : na
```

There was script written in Shell and Perl to process this output from all the eight nodes and create a Comma Separated Values (CSV) file. Script is given in the Appendix A. Each record in the file contained comma separated values for the timestamp and power consumption all the eight nodes as below.

```
20130727184713,4.30,3.90,4.60,3.90,4.00,4.00,4.80,3.40
20130727184714,4.30,3.80,4.60,3.90,4.00,4.00,4.80,3.40
```

Boston staff provided the information about the IPMI tool and power consumption of the various components of the system because it was not feasible to measure it separately. IPMI tool measured the power consumption of the processor, memory and other circuitry on SoC of a node. IPMI measured the power consumption of the cluster from the Boston head node.

#### 4.1.4 Intel Cluster

Cluster containing the four nodes, each with a dual socket Intel Xeon E5-2670 and two NVIDIA Tesla K20 GPUs was built for ISC'13 SCC. Compute blades of this cluster were separated after the competition and put on the network at Boston Ltd. Remote access was provided to them. However, only three nodes were made available for use during the dissertation period. They had 48 processing cores in total. Some selected benchmarks which were performing Fourier Transfer, Conjugate Gradient, Multigrid calculations and selected MILC application required number of processors equal to the power of two. Therefore, decision was made to use only two compute nodes for the purpose of this project.

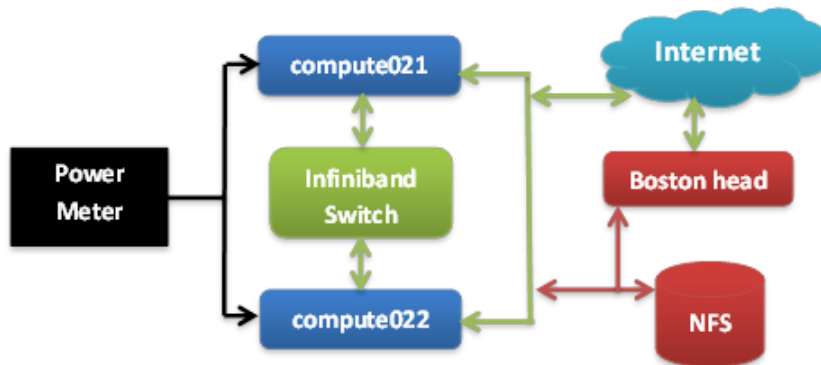


Figure 4.3: Block Diagram of the Intel Cluster

**Figure 4.3** shows the design of the Intel Cluster. Two Intel nodes were connected to the power meter and connected to each other using Infiniband switch. This cluster was also on the same network as that of Viridis cluster and sharing a same NFS and internet gateway. Unlike the Viridis cluster, Intel cluster was having static IP allocated to it by the provisioning mechanism. Server provisioning is a mechanism through which server is installed with OS, device drivers and other softwares to make it available network operations [50].

Node Name	Ethernet IP Address	Infiniband IP Address
compute021	172.16.1.11	192.168.0.2
compute022	172.16.1.12	192.168.0.3
head	172.16.1.1	NA

Table 4.1: Network Configuration of the Intel Cluster

Compute nodes were also capable of communicating over Infiniband interconnect. However, it was interesting to learn that while distributing MPI job or doing Secured SHell (SSH) they used the Ethernet connection but while performing MPI communication call they used the Infiniband interconnect. The reason behind this was the selected MPI was optimized for the Infiniband. It is discussed further in the next section, in the discussion on the software structure.



## 4.2 Software Setup

### 4.2.1 Software Setup Overview

This section covers the configuration of the software stack of all the three clusters. Intel cluster had a proprietary OS and software suite from the Red Hat and Intel respectively. Both the software suite and OS were specifically made for the cluster environment. Additionally, provisioning procedure was also provided by the Boston Staff. This made the task of configuring the cluster and softwares much easier. On the other hand, remaining two clusters were built using a generic OS available in the market. They required a huge amount of efforts for configuring a software stack. In this section a complete installation procedure for the Pandaboard cluster is explained. Viridis cluster was based on the Ubuntu based OS as that of Pandaboard cluster, so it required similar approach for the configuration. However, there were many other configurations required to be done specifically for the Viridis cluster.

This section also provides the brief information about the installed packages and their significance. It was very important to understand their purpose, because it helped to resolve many issues occurred during the installation of the applications and benchmarks. Some of these issues are discussed here which were critical and provided a good learning experience.

### 4.2.2 Overview

#### Methodology

To build the Pandaboard cluster, first a single node was built with a minimum software stack that is required for the compute nodes. Then five copies of this SD card were made to create compute nodes. Then one copy was selected for the head node. Then the head node was equipped with management softwares. Further discussion explains this methodology in detail.

#### Operating System

Pandaboards support many Linux distributions. Ubuntu and its derivative Linaro is recommended by the Pandaboard community. Linaro is non-profit organization that is developing softwares and tools for the ARM based SoCs [51]. The benchmarking results of NASA Advanced Supercomputing (NAS) suite for the Pandaboards were published on the Phoronix website. They showed that the Linaro 12.12 distribution performed better than the available Ubuntu 12.10 and Linaro predecessors. It was due to the optimizations in the latest Linaro kernel that are specifically made for the ARM based platforms [52]. There were four versions of Linaro from 13.01 to 13.04 released after the Linaro 12.12 [53]. Unfortunately the installation attempts of these versions were failed. Boards were not booting up after installation of these versions. Therefore after trying out different versions Linaro 12.12 got installed successfully. Considering scope and objective of the project there were not many efforts spent on identifying the root cause of the issue.

#### Network Configuration

Further installation of the software stack required the internet access to download packages from the repository. Therefore network configuration was done as the next step. **Table 4.2** shows the network configuration of the cluster.

Node Name	IP Address
pandahead	10.42.0.100
panda1	10.42.0.31
panda2	10.42.0.32
panda3	10.42.0.33
panda4	10.42.0.34
panda5	10.42.0.35
Gateway	10.42.0.1

Table 4.2: Network Configuration of the Pandaboard Cluster

All the nodes were allocated the static IP addresses for the sake of simplicity. Gateway was the external system (personal laptop) used to share the internet connection. To reflect these changes the `/etc/network/interfaces` file was changed.

### Configuring SSH

Open SSH server was installed so that a node can able to accept the connection requests through SSH which SSH client was installed to make such requests. It was important to configure a passwordless SSH within the nodes so that the MPI jobs can communicate without any OS interventions. For this purpose key pair of public and private rsa keys was generated using `ssh-keygen` utility. Public keys were copied into a `~/.ssh/authorized_keys` file. Since the SD card was duplicated, it did not require doing this configuration for every node. However, they shared the private keys, but from the scope and purpose of this project it was acceptable.

### Configuring Environment Modules

Environmental modules are very useful to do a dynamic configuration of the user environment [54]. Loading and unloading of these modules changes the environment variables like `PATH`, `MANPATH`, `LD_LIBRARY_PATH` dynamically. They are very useful in the scenarios while using multiple compilers, changing between different versions of MPI. Therefore environment modules were configured using `environment-modules` package. There were scripts written in TCL scripting language to create the module files for the OpenMPI-1.6.5 and MPICH2-3.0.4.

### Configuring Compilers and MPI

Compilers provided by GNU were configured. It included `gcc-4.7`, `g++-4.7`, `gfortran-4.7`. OpenMPI-1.6.5 and MPICH2-3.0.4 were installed to provide the two commonly used MPI distributions.

Installation of the MPI libraries was having some subtleties involved. Either order of their installation affected the final outcome. It was observed that the second installation overwritten the symbolic links in the `/usr/bin` directory created by the first installation. To fix this issue, it required a careful creation of the module files in order to select the proper libraries, header files and executable. Incorrect choice of `mpirun` caused `n` independent executions of the applications when it was expected to have collaborated run of `n` processes. Incorrect use of header file `mpi.h` cause program to give error message of the invalid communicator [55]. It was interesting to see that the OpenMPI used the shared MPI libraries like `libmpi.so` while MPICH2 used static libraries like `libmpich.a`. Static libraries are linked at compile time while the shared libraries or shared object files are linked at the run time [56]. Therefore shared libraries required to be present on all the nodes at proper location, wherever the executable built using them is going

to execute. Environment modules played a critical role while handling such issues.

It was observed that when the jobs are started by any MPI executable across the nodes, all the login scripts were not getting executed on the compute nodes. Only few of them like `~/ .bashrc` were getting executed. Since the environmental modules installed externally, configuring the modules and its commands was done through other scripts at boot time. This caused the MPI runs across the compute nodes unable to recognize the environment modules and its loading. It required a configuration of the modules to be brought into the login scripts by running a `/etc/profile.d/modules.sh` script. Fixing this issue required the analysis and understanding the internals of the module loading procedure and its source scripts. Finally, this configuration made the seamless switching between the MPI versions.

### Configuring Libraries and Utilities

Automatically Tuned Linear Algebra Software (ATLAS) library is commonly used package in the scientific computing. It provides the implementation of Basic Linear Algebra Subroutine (BLAS) and Linear Algebra Package (LAPACK) API which is used in the benchmarks like Linpack [57]. Package for the ATLAS library was installed from a repository. There is an implementation of ATLAS library from the Vesperix which is optimized for the ARM platform. These optimized libraries contain the routines written in assembly language that makes use of NEON SIMD extensions to provide a better performance [58]. These libraries were installed and special environment module was written, so that these optimized libraries will get preference over unoptimized libraries from the repository during linking.

### Configuring Network File System (NFS)

NFS was configured to have a shared location for keeping files from the applications, benchmarks and environment modules. NFS client was configured which used while mounting and accessing the NFS. New partition was created on the head node, and NFS server was installed on it after cloning the compute nodes. It required additional changes to the `/etc/fstab` file to mount NFS automatically while booting up the compute nodes.

There were some subtleties involved while mounting the NFS. It required a NFS server on the head node to be up before the NFS clients contact it. Otherwise NFS was not able to mount it. This required manual mounting or rebooting of compute nodes for remounting the NFS. There was a script written to perform a missing mount operation which was put into at the end of the boot sequence of a head node using `update-rc.d` utility in Linux. This script is provided in the Appendix A. There were other utilities like `autofs` are present which has background process running to take care of automatic mounting and unmounting the NFS [59]. They were not used in order to keep compute nodes with minimum background processes.

There were some other permission issues while using a NFS when the files were created from a root account. Some of them are, user was not able to edit files with root account, files created by a root account were created with `nobody` user name, root was not able to change the permissions of the file. To avoid these issues a new user account was created with the name `pandauser`. Additionally, it is not a good practice to use root account for the non-administrative operations.

### Configuring Compute Nodes

After performing the previously explained setup, SD cards were cloned using `dd` utility in Linux. Cloning procedure had to be done carefully because incorrect use of parameters caused the corrupt file system and its content.

All cloned SD cards were updated with appropriate hostnames and network configurations to make them suitable for the compute nodes. As shown in the **Table 4.2**, hostnames were

selected as `panda1` to `panda5` for the compute nodes and `pandahead` for the head node. Hostnames were updated using `hostname` command. `/etc/hosts` file was updated with list of hostnames and their IP addresses. This was used during a hostname to IP resolution. Minor typos in this file gave different issues in the installation. For example, other hosts were able to ping a particular host but that host was not able to ping others, batch scheduler was not registering the particular node as a valid compute node.

### Configuring Batch Scheduler

Batch scheduler provides the unified interface to submit jobs and manage the cluster resources. Batch scheduler framework provides features to dynamically add, remove the cluster resources. It guarantees an exclusive access to the resources. Batch scheduler coordinates the MPI processes running on the dynamically allocated resources. It avoids the requirement from the users to provide the information about the compute nodes and available resources on them. It is also convenient to provide an appropriate information to load the environment modules which guarantees to have consistent execution environment across the compute nodes.

Torque-4.2.4 and Maui-1.7.1 were used to provide the batch scheduling functionality. Torque provided the resource management framework which included registering the compute nodes, creating queues, getting jobs submitted to the queue, queue management, compute node management and many other resource management functionalities. It accepted the submitted jobs through Portable Batch Scheduler (PBS) scripts. Torque suite also came with a `pbs_sched` batch scheduler [59]. However, developers of the Torque suggested using Maui or Moab for an efficient scheduling [60]. Maui scheduler was used for the Pandaboard cluster. Maui works comfortably with the resource management framework provided by the Torque. Maui uses different services provided by the Torque to pull out the submitted jobs from the queues, gets the details about the resources from the Torque and allocates them to a job, starts job on the cluster and maintains its state during a complete lifecycle.

The Torque clients had a bit different installation procedure. Installation of the Torque server provides the shell scripts that needs to be executed on the client systems. Execution of the script creates an executable `pbs_mom` on the client machine which communicates with the `pbs_server` process on the Torque server. To create the executable, the provided shell script copies the binary content of the executable into the `pbs_mom` file and changes its permission. It also copies the required libraries in similar fashion. This approach is different from the usual build approach of building from source files. It caused `pbs_mom` to give unexpected errors when tried to execute. Copying of libraries required the `ldconfig` command to be executed in order to update runtime linker information about the newly copied libraries. It took significant amount of time and efforts to identify this issue and its fix.

### 4.2.3 Viridis Cluster

#### Methodology

Viridis cluster contained eight nodes which were provisioned using a tool named Puppet. Each node had OS and networking configured as part of provisioning. Provisioned nodes had a minimum OS set up which required further configuration of the software stack from the scratch. There was no remote access provided for the cluster management tools used for provisioning. Only IPMI access was provided for the purpose of power measurement. Hence the installation had to be replicated for each node manually. There was a script written to consolidate all installation commands and then it ran on each node to complete installation.

There was no separate head node used. This was done because many benchmarks required processor count as order of two. It would have caused only four out of remaining seven nodes to be used for such benchmarks. There was also no sudo access provided on the other system to install the cluster management softwares like batch scheduler.

#### Operating System and Networking

Ubuntu 13.04 was used as an OS for the Viridis compute nodes. It was suggested by the Boston staff and its image was already configured with the puppet tool. This helped to make a cluster available for use very quickly. Puppet configured the compute nodes to use DHCP and naming server available on the Boston network. It also mounted the NFS containing the home directory of the remote login provided by Boston team.

#### Configuring Environment Modules

Environment modules were installed similar to the Pandaboard cluster. Module files were kept on the shared NFS.

#### Configuring Compilers and MPI

Viridis nodes were also installed with GNU compilers for C, C++ and FORTRAN. Unlike the Pandaboards, installation of the MPI was done from the source tar ball due to installation issues from the repository. Cluster was installed with the OpenMPI (1.6.5) and MPICH2 (3.0.4). As this was the external installation, it was made on the shared NFS so that all the eight nodes shared same installation along with their environment module files.

As there was no head node and no batch scheduler installed, all the MPI jobs were executed using command line. Both versions of the MPI accepts the list of host files using `"-hostfile"` option with slight difference in the way of writing these files. However, this approach caused several interesting issues which highlighted the importance of using a sophisticated mechanism like batch scheduler. One of the issues was, while running an application across the nodes with MPICH2 the applications were crashing. They were running for a single node but for the multiple nodes they were giving error that they were unable to communicate with processes on the other nodes. Reason behind this was the use of naming server caused MPICH2 unable to resolve the hostnames. To fix the issue it required passing `"-disable-hostname-propagation"` flag [61]. Environment modules were useful during building the applications but while executing them across the nodes it was necessary to load these modules. It was not happening automatically. It was easier in case of a batch scheduler but in this case it required manual change to a `~/ .bashrc` file to load the appropriate modules. It was not good solution but used as a quick work around due to time constraints.

**Configuring Libraries and Utilities** Similar to Pandaboard cluster, ATLAS libraries were configured for the Viridis cluster. During course of the configuration, different applications and benchmarks failed for the different reasons. It required installation of different libraries

to fix these issues which also consumed significant amount of time during configuring the applications and benchmarks. For example, clang package was required to provide collection of the Low-Level Virtual Machine (LLVM) libraries required by compilers. `gcc-multilib` support was required to compile the applications for 32-bit ARM processors. Installation of the packages containing `make`, `cmake`, `automake`, `gawk`, `bash`, `ksh` utilities was also done on encountering with errors during building the applications.

#### 4.2.4 Intel Cluster

##### Brief Overview

Configuring Intel cluster was easier compared with the other two clusters. This task was carried out within a team and with the help of staff members at Boston Ltd. This section gives a brief overview of the software stack of this cluster.

This cluster was also not having any head node. It was mainly because the ISC SCC was having a limitation on the power consumption of the cluster. Therefore no extra node was selected and no compute node was modified with the cluster management softwares to avoid their impact on the performance.

CentOS-6.2 was used as an OS for all the compute nodes. It is an Enterprise Linux distribution from the Red Hat. This OS contains many tools and utilities commonly used in the server environments like environment modules, libraries and compilers [62]. There was a custom installation image of the CentOS was provided by the Boston team and it was configured in the kickstarter provisioning tool. Kickstarter tool from the Red Hat is specifically created to automate the Red Hat Enterprise Linux installation [63]. This custom image contained a full software stack including Intel Parallel Studio XE 2013. Intel Parallel Studio is a software development suite developed by the Intel. It contains the commonly used softwares for the parallel computing. It contains compilers, debuggers, optimized libraries like Intel Math Kernel Library (Intel MKL) and many other tools bundled within it. These tools are specially tuned to take advantage of features unique to the Intel microprocessors [64]. Provisioning software was configured to perform all these installations.

### 4.3 Summary

This chapter provided details about various steps and issues encountered during the cluster building and configuring the software stack. It was observed that building the Intel cluster became considerably simple due to the cluster management softwares provided by the Red Hat. Additionally, use of enterprise standard software suite from the Intel made the task of software stack configuration easier. On the other hand, building and configuring the Pandaboard cluster was much more challenging. Viridis cluster required more efforts while configuring the software stack. It was also observed that there was comparatively less support from the scientific forums and communities for the issues with ARM based systems.

In conclusion, cluster building from ARM based processors provided a great learning experience about the different aspects of the cluster building, networking and cluster software stack.

## Chapter 5

# Experimental Methodology

One of the objectives of this project is to perform the energy efficiency analysis of the embedded SoC based processors and the servers commonly used in the current HPC systems. This chapter provides the details about the experimental methodology selected to use these clusters to achieve the objectives of this project.

### 5.1 Methodology

Different benchmarks were selected to perform the energy efficiency analysis of the processors in the three types of clusters. The selected benchmarks were divided into three categories as shown in the **Figure 5.1**.

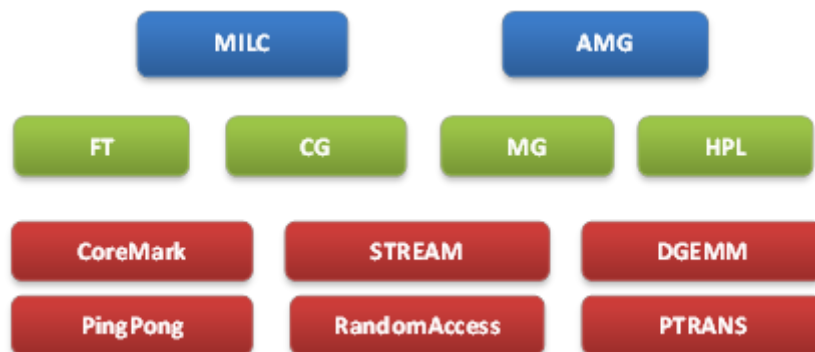


Figure 5.1: Set of selected benchmarks for the experiment

The performance of the hardware used in the cluster was measured using the synthetic benchmarks. These are shown by red color (last two rows) in the **Figure 5.1**. Each benchmark in this category tested a particular component of the cluster. Performance of the clusters on these benchmarks was used to analyze the behavior of the application specific benchmarks in the next category. Initially the performance of a core from each type of processor from the different clusters was measured using CoreMark Benchmark. Then performance of the clusters was measured using HPC Challenge (HPCC) benchmark suite. HPCC is commonly used to measure the performance of the HPC systems [65].

Next category of benchmarks was of the application specific benchmarks with an exception of the High Performance Linpack (HPL). Apart from HPL the other benchmarks were selected from the NAS Parallel Benchmarks (NPB) suite. They are shown in green color (second row) in the **Figure 5.1**. These benchmarks contained the calculation which was building block of

the selected real-world applications. There were three application from this suite were selected. They were significant for the selected scientific applications MILC and AMG. These benchmarks measured the performance along with the energy efficiency of the clusters. HPL from the HPCCC suite was used to measure the energy efficiency of the cluster on the GFLOP/Watt metric. It is the same metric used for the systems in Green500 list [66].

Purpose of selecting the applications from NPB suite was to present the combination of the results of these benchmarks and performance of the real-world applications based on them. These results can be used as a reference, to get an idea about the performance of the other applications on these clusters, which are having similar building blocks. The real-world applications MILC and AMG were benchmarked as part of the top level of the benchmarks. They are shown in the blue color (Top row) in the **Figure 5.1**.

All the benchmarks were optimized where it was possible. Optimizations are explained in detail for the HPL alone. Similar optimizations were tried for the other benchmarks as well. All the benchmarks and applications except AMG provided results at the end to confirm the success of the benchmark. It included benchmarks from the HPCCC, NPB suite and MILC application. Since there was no test provided for the AMG, output was compared with the unoptimized version.

## 5.2 Power Measurement

Power measurement was a critical task. Incorrect approach of power measurement may lead to false conclusions. Results become more sensitive in case of the low powered systems like Pandaboard and Viridis nodes.

Power measurements were done by using the power meter or IPMI tool as shown in the cluster setup. In case of the Pandaboard cluster, nodes that were not used for the benchmarking purpose were turned off. Similar arrangement was requested for the Intel cluster. IPMI tool provided a feature to measure the power consumption of the single node, so it did not require any specific arrangement.

There are some factors which can affect the performance and power consumption of the processor and system. OS noise and heating of processors are some of the examples. To minimize the errors due to noise, all the benchmarks were ran ten times. Average power consumption was calculated for each run. The results repeated for more than 70% of time were considered as final ones. Power consumptions of the systems were recorded by the external devices like power meter and IPMI tool. These devices were recording readings before and after the benchmark run. It required calculating exact runtime of the benchmarks and then taking the corresponding readings from the output of measurement device.

There were other factors like the sensitivity of the meter and sampling frequency would also have impacted the results. However, it was not feasible to use consistent power measurement techniques or device. Importantly, the impact of these factors was not expected to be significant enough to affect the conclusions of the experiment. Objective of the project was to understand the performance of the SoC based systems for the scientific applications at broader level. Therefore, these factors were ignored.



## Chapter 6

# Synthetic Benchmarks

This chapter contains the discussion on the selected synthetic benchmarks and their results. Synthetic benchmarks are used to understand the performance of the hardware used in the cluster. This chapter does not contain the analysis on their energy efficiency except for the HPL. Energy efficiency analysis of HPL is done as it is used by the Green500 list to determine energy efficiency of the supercomputers [66].

### 6.1 CoreMark

CoreMark Benchmark is widely used in the embedded systems to measure the performance of the CPUs. It is also possible to measure the performance of a single core in a multiprocessor system using this benchmark [67]. This benchmark is having negligible communication involved in it which makes it to scale perfectly. Different processors in the three clusters were having the different core counts. Therefore, for the fair comparison only the performance of a single core was considered for analysis.

As discussed earlier, both Pandaboard and Viridis nodes were used the processors based on ARMv7 architecture. This benchmark was selected to analyze the factors affecting the performance of systems containing these two types of processors. Additionally, it will provide information about the processing capability of the processors.

CoreMark benchmark provides a single number as score for the benchmark. This makes it easier to compare the performance of the CPU cores. This single figure for the performance is derived from the performance of the CPUs for the operations for which CPUs are commonly used. These operations involve search and sort on lists, cyclic redundancy check (CRC) and manipulations on matrix [67].

## Results and Analysis

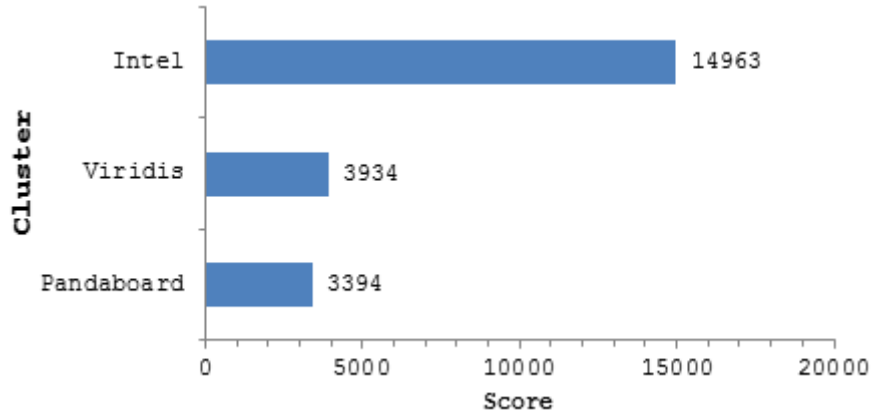


Figure 6.1: Performance of the cores in different clusters for the CoreMark benchmark

**Figure 6.1** shows the comparison of the performance of a single core used in the Pandaboard, Viridis Calxeda nodes and Intel Xeon processors. It can be seen that the Xeon core performed much better than the cores in the embedded SoC based ARM processors. One of the main reasons was that they were operating at higher clock frequency (2.6 GHz) compared to the Pandaboard (1.2 GHz) and Viridis Calxeda processors (1.4 GHz). It was interesting to observe that the performance of the Xeon core was not directly proportional to their operating frequency. At  $\sim 1.8$  times higher clock frequency than the Viridis nodes the Xeons performed  $\sim 3.8$  times better. In case of the Pandaboard, Xeons had  $\sim 2.2$  times higher clock rates, but they performed  $\sim 4.4$  times better. This was because the Xeon cores were having more register count, deeper cache and memory hierarchy. Each core was capable of performing four times more floating point operations than ARM cores. Additional circuitry like branch prediction also helped Xeon cores to achieve higher performance. The results for the CoreMark benchmark showed that the Xeon processors were much powerful for the general purpose CPU operations.

On the other hand, both Pandaboard and Viridis Calxeda processors were based on the same ARMv7 processor architecture with difference in their operating frequency and memory. CoreMark is not memory intensive benchmark [67]. The obtained results were directly proportional to their operating frequency. This showed that both Pandaboard and Viridis cores are equally powerful.

## 6.2 HPCC Benchmark Suite

One of the goals of the project was to analyze the suitability of the embedded SoC based systems for the scientific applications and future HPC systems. HPCC benchmark suite consists of seven tests in total. These tests measure various attributes of the system which significantly impacts the performance of the real-world scientific applications [65]. Therefore the HPCC benchmark suite was selected to measure the performance of the ARM processors based clusters. Results from the Intel Xeon processors based cluster will represent the performance and the requirements of the current HPC systems.

Compilation of the hpcc-2.1 benchmark suite produced a single executable file [65]. This executable runs all the benchmarks in the HPCC suite. These benchmarks produce results for a single core as well for whole cluster. All the clusters were not having a same core count. To make the fair comparison the performance of single core is discussed for some of the benchmarks. Each benchmark discussion contains the information about the core count with which

it was run.

### 6.2.1 DGEMM

DGEMM stands for the Double-precision General Matrix Multiply (GEMM). This benchmark measures the floating point rate for the matrix multiplication containing double precision floating point numbers [65]. Matrix multiplication is used as a building block in many scientific calculations. Therefore, higher performance on the DGEMM benchmark is important attribute of the HPC systems.

BLAS libraries contain the subroutine for the matrix multiplication [57]. Use of optimized BLAS libraries can play a key role in the results of DGEMM benchmark as well as the scientific applications using the matrix multiplications.

#### Results and Analysis

**Figure 6.2** is showing the performance for the single core of the different processors for a DGEMM benchmark. Intel processors showed very high performance of 19.12 GFlops/sec which was about 17 times the performance of the Viridis cores and 27 times the Pandaboard cores. It clearly showed the Intel cores were much more powerful in doing the double-precision matrix multiplication operations.

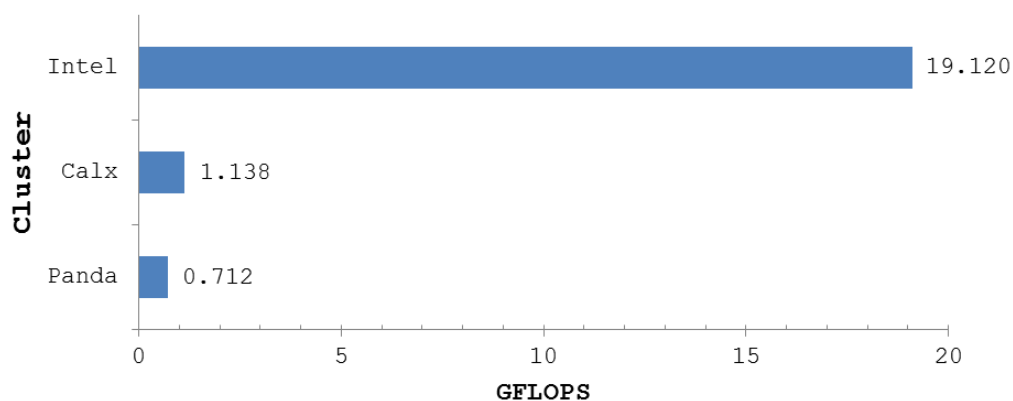


Figure 6.2: Performance Panda, Viridis and Intel Clusters on DGEMM benchmark

As discussed earlier, Intel Xeon cores could perform four times more floating point operations than the ARM cores. They also had about double clock frequency. These were the major reasons behind the better performance of the Intel processors.

### 6.2.2 STREAM

STREAM benchmark is used to measure the sustainable memory bandwidth of the system. Memory bandwidth is one the important attribute of the HPC systems. It is well known fact that the CPU speed is much higher than the memories. This makes CPU to stall for the data to arrive from the memory to perform calculations. Therefore memory bandwidth plays a critical role and act as a potential performance bottleneck for many memory bound applications. STREAM uses GB/s as a metric to measure the memory bandwidth. It performs four types of operations Copy, Scale, Sum and Triad. It provides the memory bandwidth each of them separately [65].

**Results and Analysis** Figure 6.3 is showing the results for the single process STREAM benchmark for the three clusters.

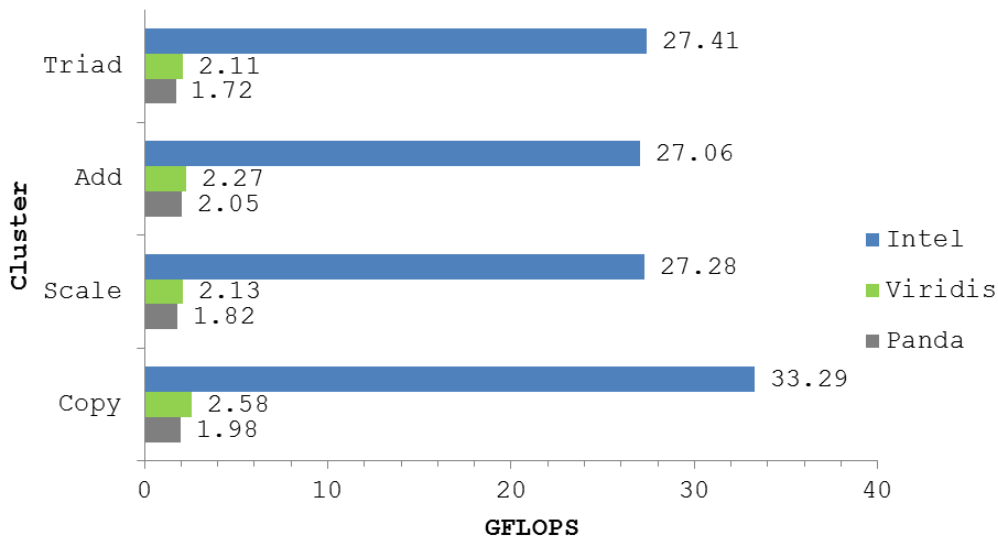


Figure 6.3: Performance Panda, Viridis and Intel cores on STREAM benchmark

It can be clearly seen that the Intel core had very high memory bandwidth compared with the other two types of the ARM based processors, for all four types of operations. Reason behind this was the Intel cores had eight memory banks containing DDR3 RAM sticks of 8GB each. On the other hand, a Viridis node contained single DDR3 RAM stick of 4GB per node. Energy card contained four sticks for four nodes on it. Pandaboard contained Low Powered DDR2 RAM of 1GB.

It was interesting to see that the Pandaboard core achieved almost equal bandwidth as that of the Viridis core with less and old version of the memory. It was because the memory was present on the board. This highlights the benefits of SoC arrangement of the components over the conventional one.

### 6.2.3 RandomAccess

RandomAccess Benchmark measures the frequency of the system to issue updates to the randomly generated memory addresses. It is measured in Giga-updates per second (GUPS) [65].

#### Results and Analysis

**Figure 6.4** shows the results for the RandomAccess benchmark for the single process.

It can be seen that the frequency of updating the randomly generated addresses was much higher for the Intel cores. Similar to the STREAM benchmark, Pandaboard core showed almost 75% of the performance of a Viridis core. The reason behind this was same that the Pandaboards had memory on the same chip as that of processor.

Intel cores perform lot better than the other two clusters, because they had larger DDR3 memory but it had eight memory banks to access it. Additionally, they were operating at the higher memory clock frequency (2.2GHz) than the Pandaboard (0.4GHz) and Viridis core (0.6GHz). These were some of the reasons behind the higher update frequency of the Intel cores.

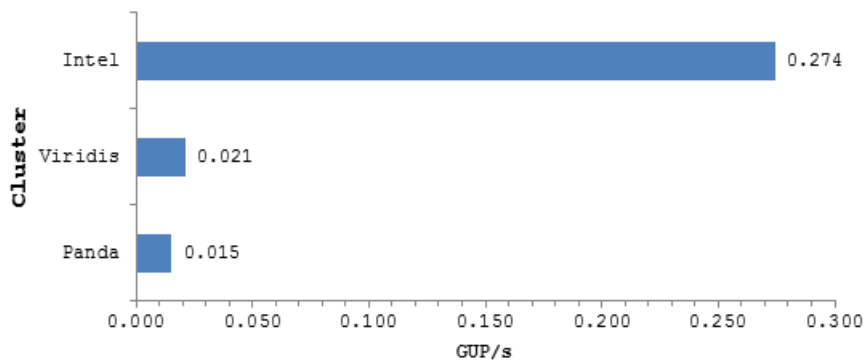


Figure 6.4: Performance Panda, Viridis and Intel Clusters on RandomAccess benchmark

### 6.2.4 PTRANS

PTRANS stands for the Parallel matrix TRANSpose. In this the pair of processors communicates with each other simultaneously. They transfer the data in the large arrays which tests the communication capacity of the network. It is measured in GB/s [68].

While running this benchmark from the HPC suite it uses all the processing cores with which benchmark is run. Although the Pandaboard cluster was having less core count (ten cores) than the Intel and Viridis cluster (both 32 cores), the benchmark involved communication between the pair of processors. This makes the results of the benchmark depend on the network capacity than the core count [68]. Therefore results for the Pandaboard are also presented along with the other two cluster.

#### Results and Analysis

**Figure 6.5** is showing the performance of the clusters on the PTRANS benchmark which tested the capacity of the network. Considering the significance of the benchmark for the scientific applications, all the available processing cores within each cluster were used for this benchmark. Both Intel and Viridis cluster had 32 cores which were used for this benchmark. Only available ten cores of the Pandaboard cluster were used for this benchmark.

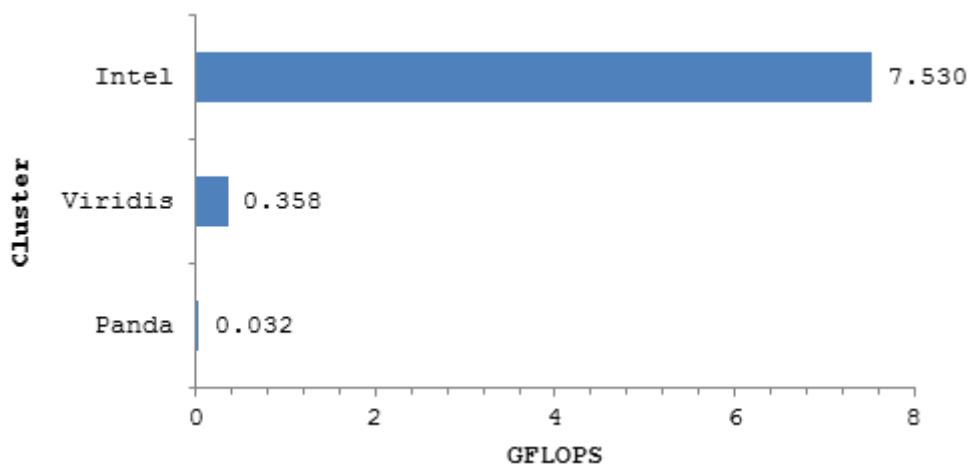


Figure 6.5: Results of the PTRANS benchmark on all the clusters

From the **Figure 6.5** it is clear that the Intel cluster had very high capacity network. It showed about 25 times capacity of the Viridis cluster. Infiniband interconnects (40Gb/s) had

only four times more bandwidth than the Viridis cluster with (10Gb/s). The low latency interconnects would have resulted in the high performance of the Intel cluster on the PTRANS benchmark. Pingpong benchmark would provide more details about the latency and bandwidth of the network.

From these results Intel nodes were expected to perform better on the communication dominating application and benchmarks. On the other hand, Pandaboard cluster showed the least capacity about ten times lesser than Viridis cluster. This was proportional to their network capacity. Therefore the Pandaboards were expected to show poor scaling for the communication critical applications and benchmarks.

### 6.2.5 Pingpong

Pingpong benchmark from the HPCC suite performs the transfer of messages across the nodes of the cluster. Algorithm of the benchmark forms different topologies from the available processes to tests the bandwidth and latency across the nodes. In case of the multiple nodes the topologies are formed in such way that the communication between the processes goes through the interconnect connecting the nodes. It also makes parallel message transfers when more than two processes are involved []. This makes the benchmark independent of the core count or the processes involved in the benchmarks. It calculates the communication bandwidth and latency for the small (8Bytes) and large (2000000Bytes) MPI messages. MPI uses different protocols Rendezvous and Eager protocol to transfers a large and small message respectively. This benchmark clearly indicates the performance of the cluster for the communication intensive benchmark.

#### Results and Analysis

**Figure 6.6** is showing the results for the Pingpong benchmark. For small messages the Intel cluster showed an extremely low latency due to use of Infiniband interconnect. It showed 1.36 microseconds which was about 100 times lower than the latency for the Viridis cluster. Use of 10Gb Ethernet benefited the Viridis cluster and gave about ten times lower latency. Intel nodes performed transfer of the larger messages about 33 times faster than the Viridis nodes. Viridis nodes maintained their ten times lower latency for larger messages as well.

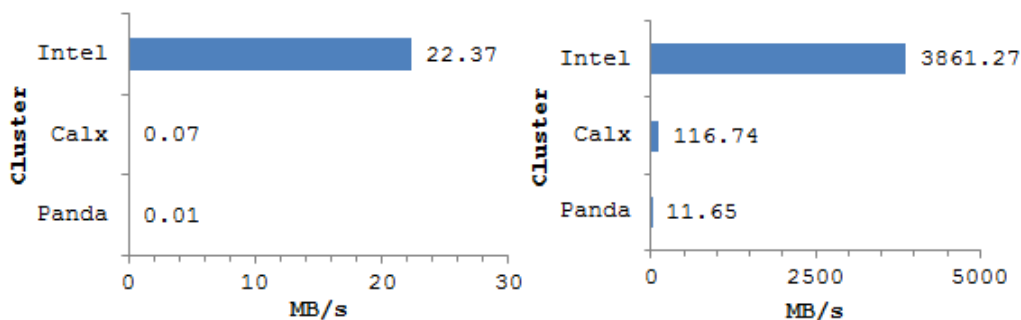


Figure 6.6: Latency for the Small and Large Size message on the clusters

Intel nodes were benefited by the Remote Direct Memory Access (RDMA) functionality for the larger nodes. This feature is used for the larger messages by the modern interconnects to achieve better performance [49]. Exact impact of it could not be seen from the results in **Figure 6.6**.

Figure 6.7 shows the communication bandwidth achieved by the different cluster.

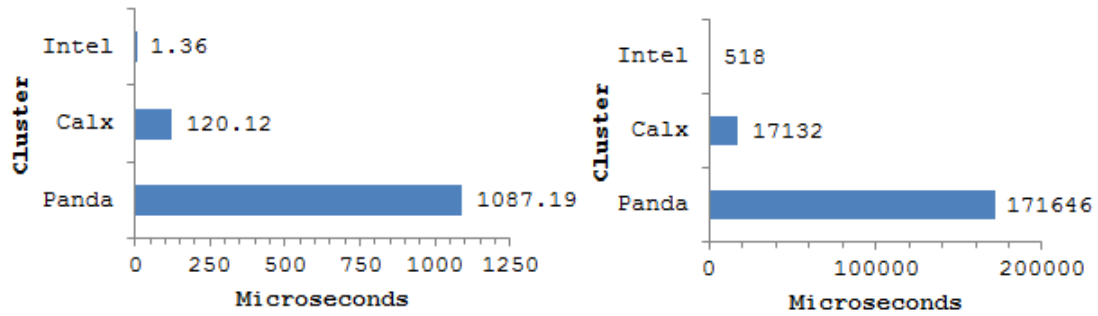


Figure 6.7: Network bandwidth for the Small and Large Size message on the clusters

Results for the communication bandwidth were similar to the results for the latency. Intel nodes showed the highest communication bandwidth for both small and large messages.

## 6.3 HPL

High Performance Linpack (HPL) benchmark is used to rank systems in the Top500 list [5]. It generates, solves, checks and times the process of solving randomly generated dense system of linear equations. HPL also tests the result of the benchmark after the run and provides pass or fail information. HPL provides many options to express the problem for a given system. It includes selection of the grid of processes, size of the problem to be solved and many other options to tune the benchmark for a particular system [69]. Linpack can be highly optimized and problem size can be selected large enough to make maximum use of the available memory in the system. It scales well and performs many floating point operations per memory access which leads to almost peak performance of the system [70]. Linpack is criticized for the same reasons. Many real-world applications get limited by the bottlenecks like memory bandwidth, communication bandwidth, disk IO operations.

HPL is also part of HPCCC benchmark suite. It is described in separate section due to scaling and energy efficiency analysis performed on this benchmark.

### 6.3.1 Pandaboard Cluster

Linpack performance on multiple nodes depends on the performance of a single node. Therefore, initially the Linpack was optimized for a single node and then the scaling test was done with the two available MPI versions.

**Single Node Performance Optimization Table 6.1** is showing the performance optimization results for a single Pandaboard node containing two processing cores. Optimization level O3 was selected. It was the highest level of optimization supported by the available gcc-4.7 compiler.

Optimization Type	Performance (GFLOPS)
Atlas library and -mfloat-abi=hard -march=armv7-a -funroll_loops -O3	1.775
-mfpu=vfpv3-D16 mfloat-abi=hard -march=armv7-a -funroll_loops -O3	1.285
-mfloat-abi=hard -march=armv7-a -funroll_loops -O3	1.287
-march=armv7-a -funroll_loops -O3	1.284
-funroll_loops -O3	1.284
-O3	1.282

Table 6.1: Performance optimization results for a single Pandaboard Node

Loop unrolling was done using a compiler flag `-funroll_loops` which showed a slight improvement in the performance. By using this flag the compiler unrolls the loops whose size can be determined at the compile time [71]. Loop unrolling causes more number of instructions available for the compiler to perform the instruction reordering. It ultimately tries to avoid the stalling of the pipelines. It is also beneficial at the hardware level where out-of-order execution units optimize the sequence of execution. They reduce the bubbles in the pipelines which ultimately yields more performance. ARM Cortex-A9 supports the out-of-order execution in the hardware. This justifies the reasons behind the improved performance with the loop unrolling.



It was interesting to see that there was no performance improvement by providing the information about the underlying architecture. Reason behind this might be the use of operating system and compiler from the Linaro. They were specifically optimized for the ARM architecture based processors.

```
gcc -Q -help=target -march=native
```

Above command was used to query the GNU compiler to retrieve information about the target platform and architecture. Output of this command helped to select the appropriate architecture for the "-march" flag as "armv7-a" and vector floating point unit (VFPU) using "-mfpu=vfpv3-D16" flag.

It was interesting to see that by specifying the VFPU reduced the performance of a node. One of the possible reasons behind this was that the support for VFPU was later removed by the ARM because these instructions were getting processed sequentially. These instructions did not give the intended performance gain. VFPU are now used by the NEON SIMD extensions [42]. By specifying "-mfpu" flag the compiler would have forced to generate the vector instructions for this unit which would have reduced the performance due to their known drawback. This analysis was restricted here by considering the scope of the project and this flag was not used during further optimizations.

Querying gcc also provided the information about the support for the floating-point ABI (Application Binary Interface). Use of the type of ABI as "hard" with the "-mfloat-abi" flag showed improvements in the performance. Reason behind this was that the compiler used FPU specific conventions while generating the floating-point instructions. This is ARM architecture specific flag provided by the GNU compilers [72]. There were other choices of ABI can be provided instead of "hard" with this option. They are "soft" and "softfp" and this might be the reason that the compiler did not select option "hard" automatically.

There was a huge performance gain achieved by using the ATLAS libraries from the Vesperix. This library is optimized specifically for the ARM platform. The main reason for the better performance was the use of NEON SIMD extension which ultimately vectorized the computationally intensive part for the platform [58].

**Linpack Scaling on multiple Nodes** After optimizing the Linpack for a single node, the scaling test was done whose results are shown in **Figure 6.8**.

Nodes Count	Core Count	OpenMPI (GFLOPS)	MPICH2 (GFLOPS)
1	2	1.775	1.772
2	4	3.015	3.011
3	6	4.038	4.037
4	8	4.982	4.978
5	10	5.829	5.823

Table 6.2: Linpack scaling for the Pandaboard Cluster

From **Figure 6.8** and **Figure 6.4** it can be seen that the Linpack scaled consistently but not linearly. Significant performance drop was observed when Linpack was scaled across the nodes. Consistent scaling was observed when the extra nodes were added. This behavior indicates towards the potential communication bottleneck. This might be due to slow network (1Gb Ethernet) used for the Pandaboard.

Different types of MPI libraries may cause the difference in the performance, especially for the codes having different communication pattern. This is mainly due to the different

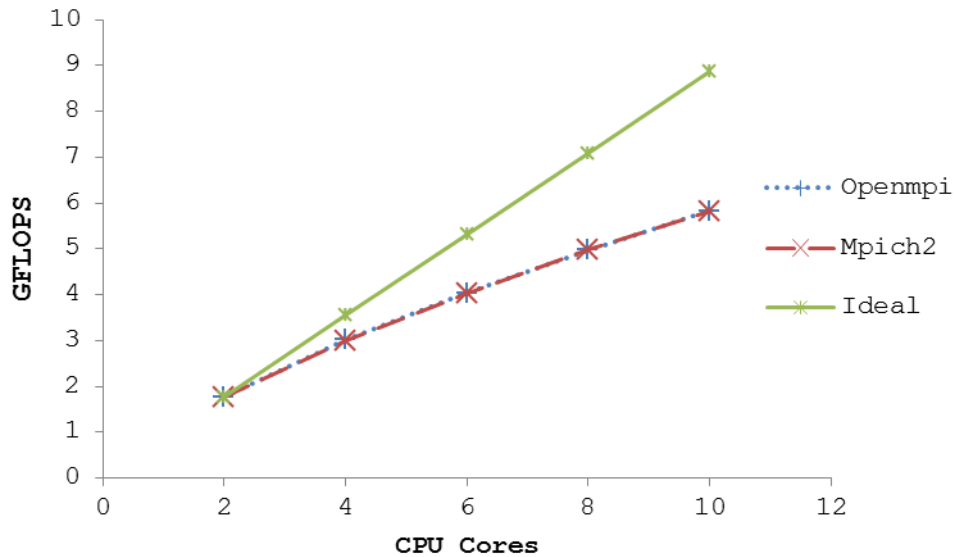


Figure 6.8: Linpack scaling on the Pandaboard Cluster

algorithms and implementation of the MPI library routines is done by the different library vendors. From the **Figure 6.8** it can be seen that the performance of the cluster was almost the same for both types of the MPI libraries. It might be because the Linpack performance is not dominated by the communication [70]. Additionally, both the types of libraries were from the Linaro repository and there was no specific optimization was known to be done for either of them. There was no special feature like support for the collective calls was provided by the hardware. Hence for the other benchmarks only one of them was selected. For Pandaboard OpenMPI was selected. Considering the aim of the project and to avoid the ignorance towards the impact of MPI library choice the decision was made. Only the selected applications MILC and AMG were benchmarked using available MPI libraries. Similar approach was used for benchmarking the other clusters as well.

### 6.3.2 Viridis Cluster

Viridis cluster contained the processors with similar cores as that of Pandaboard cluster, but the whole system was assembled in a unique way. It was targeted for the server market so additional focus was given on the performance of the system along with the energy efficiency. This makes it more interesting to see the performance of this system along with the Pandaboards.

#### Single Node Performance Optimizations

**Table 6.3** is showing the performance optimization results for the Viridis Calxeda node containing four processing cores.

It can be seen that the performance optimizations for the Viridis Nodes also showed the similar results as that of Pandaboards. As discussed previously, reason behind this was the underlying processors had same ARMv7 architecture. From simple calculation it could be seen that the performance of single Viridis node was proportional to the increased core count and clock frequency.

Optimization Type	Performance (GFLOPS)
ATLAS Library and -march=armv7-a -mfloat-abi=hard -funroll-loops -O3	4.376
-march=armv7-a -mfloat-abi=hard -funroll-loops -O3	3.236
-mfloat-abi=hard -funroll-loops -O3	3.196
-mfpu=vfpv3-d16 -funroll-loops -O3	3.177
-funroll-loops -O3	3.193
-O3	3.176

Table 6.3: Performance optimization results for a single Viridis Node

Unlike the Pandaboards, the performance of the Viridis cluster showed some improvements when the machine architecture was specified. "-march" flag was used. It informed the compiler about the underlying "armv7-a" architecture. This behavior was expected. As discussed previously, Viridis node used the generic OS Ubuntu 13.04. This caused the softwares including the GNU compiler from the repository to have generic features, not specific for the ARM platform. Therefore the compiler did not generate the platform specific code without specifying explicitly as in case of the Pandaboards.

#### Linpack Scaling on multiple Nodes

Figure 6.9 shows the scaling results for the Linpack benchmark on the Viridis cluster. It can easily be seen that the Linpack scaled very well on the Viridis nodes compared to the Pandaboards. Viridis nodes showed the proportional performance to the Pandaboards for a single node. These observations indicate towards the efficient network in the Viridis nodes as the reason behind the better scaling of the Linpack on them.

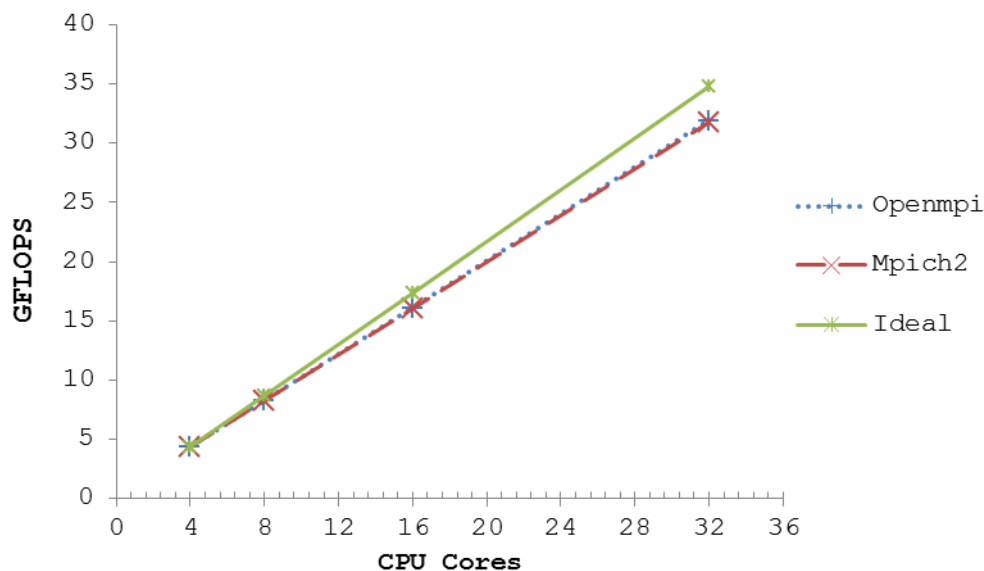


Figure 6.9: Linpack scaling on the Viridis Cluster

Nodes Count	Core Count	OpenMPI (GFLOPS)	MPICH2 (GFLOPS)
1	4	4.35	4.34
2	8	8.23	8.20
4	16	16.09	16.05
8	32	31.94	31.79

Table 6.4: Linpack scaling for the Viridis Cluster

From the **Figure 6.9** and **Figure 6.4** it can be seen that there was no significant impact of using different MPI libraries. Reason behind this was similar to the Pandaboard cluster. Neither of the libraries was providing or exploiting any special feature of the hardware. For the further benchmarking OpenMPI was selected for the Viridis Cluster.

### 6.3.3 Intel Cluster

Similar to previous two clusters, performance of a single node was optimized and then the scaling test was performed. Intel cluster contained the x86 based processors operating at much higher frequency than the processors in the other two clusters. Hence they were expected to achieve much higher Linpack score than the other clusters.

**Single Node Performance Optimization Table 6.5** is showing the performance optimization results for a single node in the Intel Cluster containing 16 processing cores.

Different optimizations were attempted for the Intel node. "-unroll-aggressive" is the Intel compiler flag for performing a loop unrolling. Newer versions of the Intel microprocessors adapt the evolving x86 instruction set and its new extensions. Use of "-xHost" flag informs the compiler to generate instructions with the highest available instruction set on the host machine [73].

Optimization Type	Performance (GFLOPS)
-xHost -O3	128.80
-fomit-frame-pointer -O3	128.70
-unroll-aggressive -O3	128.80
-O3	128.70

Table 6.5: Performance optimization results for a single Intel Node

From the figure it can be seen that there was no performance gain was achieved by any of the compiler flags. Reason behind this was the Intel nodes were having Intel Parallel Studio installed on them. It was specially tuned for the Intel microprocessors [64]. Therefore compiler picked up these optimizations implicitly.

As discussed previously, software suite also bundled with Intel MKL, the libraries specially optimized for the Intel platforms. This also contributed towards overall Linpack performance. The generic version of these libraries was not available on the platform, so the exact impact of the MKL could not be determined. Published benchmark results from the Intel and experience with ARM based clusters supports the importance of the optimized libraries for the platform [74].

**Linpack Scaling on multiple Nodes** **Figure 6.10** shows the Linpack scaling results for the Intel cluster. Scaling results clearly map to the discussion in the hardware and cluster setup sections 3.2.3 and 4.2.4 respectively. It can be seen that the Linpack scaled almost linearly

till the eight cores. It was because each node was dual socket containing Xeon processor with eight cores in each socket.

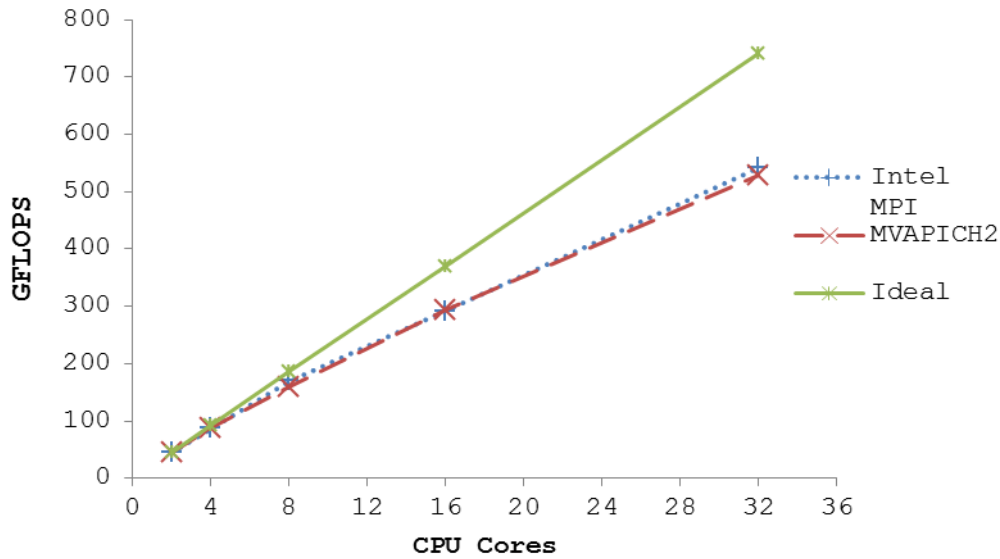


Figure 6.10: Linpack scaling on the Intel Cluster

Further scaling results were proportional to the type of communication medium used between the processors. Communication between two processors in a node had to pass through the QPI interconnect connecting them. This was of course slower than the communication within a processor. Therefore a performance drop can be observed while scaling from 8 to 16 cores. While scaling from 16 to 32 cores the communication had to be done over the Infini-band interconnect connecting two compute nodes. This showed the further performance drop for scaling across the nodes.

From the **Figure 6.6** it can be seen that the use of different MPI libraries also did not show any significant improvement.

Nodes Count	Core Count	Intel MPI (GFLOPS)	MVAPICH2 (GFLOPS)
1	2	46.29	45.03
1	4	89.48	90.62
1	8	171.10	165.51
1	16	293.40	292.73
2	32	528.79	541.60

Table 6.6: Linpack scaling for the Intel Cluster

Unlike to the other two clusters, all the used MPI libraries were specially optimized to take benefit of the Infiniband interconnect. There was slight better performance was observed for the MPI library for Intel. Therefore, Intel MPI was used for the further benchmarking.

In this project used only two nodes, so there was not much impact of the MPI library was expected, especially when all of them were interconnect aware. During the benchmarking for the MILC application at ISC'13 SCC, it was observed that while scaling across the four nodes there was slightly more impact of the MPI library. Intel MPI scaled better than the other ones. However, the impact was not significant enough to influence the conclusion of this project.

### 6.3.4 Linpack Energy Efficiency

This discussion is more focused on the objective of this project to compare the energy efficiency of the processors in different clusters. **Figure 6.11** shows the performance of the clusters on the GFLOPS/Watt metric.

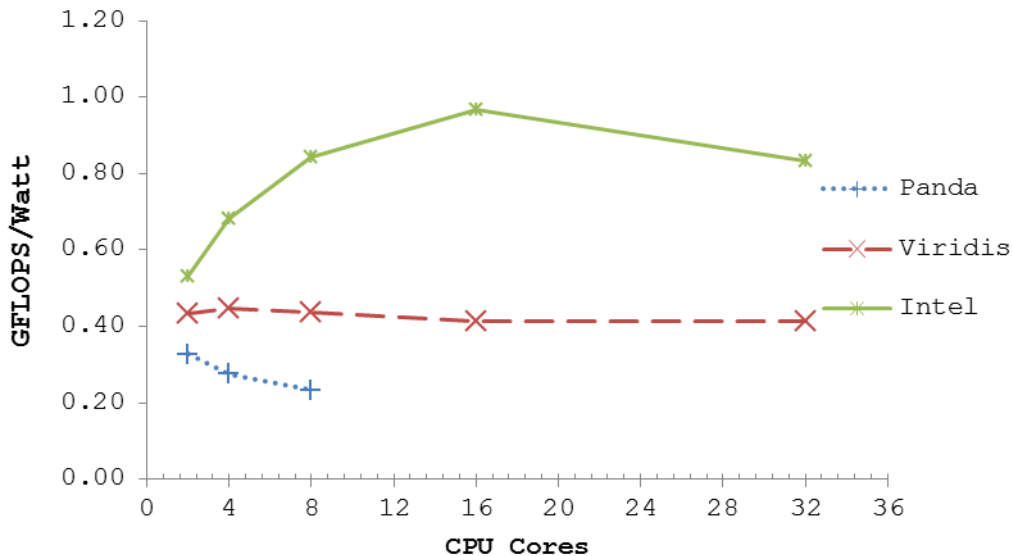


Figure 6.11: Energy efficiency comparison of Panda, Viridis and Intel Cluster

From the **Figure 6.11** it can clearly be seen that the Intel processors showed higher energy efficiency than the ARM based processors. Intel processors consumed more power but provided better returns in terms of performance and so the energy efficiency.

Both Intel and Viridis cluster showed improvement in the energy efficiency while scaling within the node. This was mainly because of the idle power consumption of the unused processors. As discussed previously, both the processors have made improvements to reduce their idle power consumption. Viridis nodes did not show drastic improvement in the energy efficiency compared to the Intel nodes. This showed the better power management of the Viridis nodes. The reason behind this was the energy management engine which tries to turn off the maximum unused circuitry on a chip, while Xeon processors under-clock them [43, 28].

It was interesting to see that the Viridis nodes showed much consistent energy efficiency compared to other clusters. It was mainly due to the unique arrangement of the four nodes on a single card and multiple such cards within a chassis. This made Viridis nodes to achieve good scaling and better energy efficiency. It was also seen that the Intel cluster showed drop in the energy efficiency when it was scaled across the nodes. This was mainly due to increased power consumption of the network cards and this scaling was done within the chassis which impacted the performance. Pandaboard cluster showed the least energy efficiency and it also reduced considerably while scaling across the nodes. This was mainly due to the slow network card on the Pandaboards which acted as a communication bottleneck. These observations indicate towards the benefits of using an efficient network along with the SoC arrangement to achieve better energy efficiency with performance.

In overall comparison, Intel nodes turned out to be twice energy efficient as that of the Viridis nodes and about four times the Pandaboard nodes for the eight cores. This ratio was almost the same when Viridis and Intel clusters used with their full capacity of 32 cores.

## 6.4 Summary

This chapter provided detailed discussion on the results obtained for the synthetic benchmarks. Performance of the selected hardware is measured from core level till the cluster level using CoreMark and HPCC benchmark suite. Synthetic benchmarks showed that the Intel processors were much more powerful than the other two types of the ARM based processors. Performance for the STREAM benchmark highlighted the benefits of SoC, by having on chip memory. Performance of the Pingpong benchmark showed the impact of having low powered network card. It was interesting to see that the unique packaging of the Viridis cluster made it to achieve the best scaling results on the HPL benchmark among the selected clusters. It was interesting to see that the Intel cluster showed higher energy efficiency than the ARM based processors. Four times more capacity to perform the floating point operations was one of the main the reasons behind it. From the results on the selected synthetic benchmarks it is clear that the Intel processors were clear winner in the performance domain. HPL scores also showed the Intel processors had higher energy efficiency. Considering the drawbacks of the HPL, it will be interesting to see the energy efficiency of the ARM based processors for the real-world scientific applications.





## Chapter 7

# Application Specific Benchmarks

This chapter contains the results and analysis of the application specific benchmarks selected from the NPB suite. It contains the discussion focused on the energy efficiency of the application specific benchmarks.

### 7.1 NAS Parallel Benchmark Suite (NPB)

NPB suite is more focused towards measuring the performance of the system, which can closely be mapped to the performance of the real-world scientific applications. NPB is developed and maintained by the NASA Advanced Supercomputing (NAS) Division. This benchmark consists of the applications derived from the Computational Fluid Dynamics (CFD). These applications mimic the computation which is building block for the applications in the CFD domain. They are also commonly used in other scientific applications [75].

NPB version 3.3 contains eight applications in the suite. Selection of the applications from NPB suite was driven by the choice of MILC and AMG as the scientific applications. Three out of eight applications were selected for the purpose of this project. They are Conjugate Gradient (CG), Multi-Grid (MG) and Fourier Transform (FT). CG and MG were selected because they are one the major building blocks of the selected applications MILC and AMG respectively [76, 77]. FT was selected because of its wide use in many scientific applications.

All the applications in the NPB suite give their performance in the form of million operations per second (Mops/sec). To calculate the energy efficiency, the Mops/sec value is divided by the average power consumption during the benchmark run. This value is used as a performance/Watt metric. These benchmarks also have integrated tests to check correctness of the answer at the end.

#### 7.1.1 Fourier Transform (FT)

FT contains the calculation of discrete 3-D Fast Fourier Transforms (FFT). It contains all-to-all communication which makes this benchmark communication intensive [75]. In spite of having 1-D FFT as part of HPC suite, this benchmark was selected. Reason behind this was to perform energy efficiency analysis of the benchmark involving 3-D FFT calculation. As these FFT calculations are common in scientific applications, these results would be helpful to understand the behavior of the applications, on these processors and clusters, which involve such calculation.

### Results and Analysis

**Figure 7.1** shows the energy efficiency achieved with the different clusters for the Fourier Transform benchmark. **Table 7.1** contains the values for both raw performance and performance/Watt metric for all the three clusters.

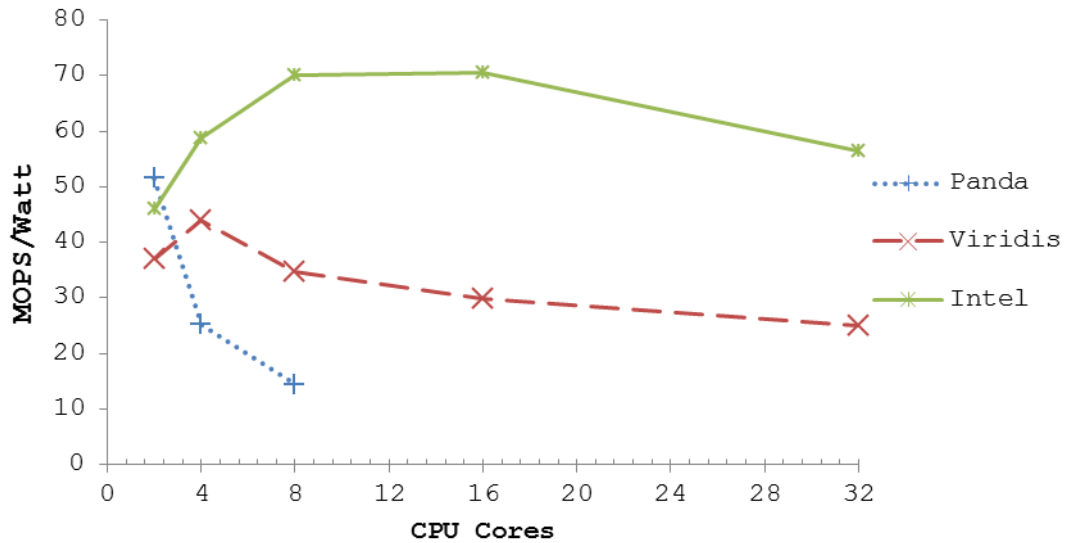


Figure 7.1: Energy Efficiency for the NAS 3D-FFT Benchmark

Similar to the Linpack benchmark, it can be seen that the scaling within the node showed improved energy efficiency. It was due to the power consumption of idle processors. While scaling across the nodes it could be seen that the Pandaboard showed a sudden drop in the energy efficiency.

Core Count	Intel MOPS	Intel MOPS/Watt	Viridis MOPS	Viridis MOPS/Watt	Panda MOPS	Panda MOPS/Watt
2	3449.8	46.0	284.5	37.0	226.5	51.5
4	6566.0	61.8	418.9	43.9	248.6	25.2
8	9166.1	77.0	625.8	34.7	295.9	14.4
16	17282.7	90.5	1072.5	30.0	-	-
32	28404.2	85.0	1654.6	24.9	-	-

Table 7.1: Energy Efficiency for the NAS 3D-FFT Benchmark

From **Table 7.1** it can be seen that the Pandaboard showed extremely poor scaling. It is because the 3-D FFTs contain all-to-all communication. This requires an efficient communication channel between the communicating processors. From the results of the Pingpong benchmark from the HPCC suite, it was observed that the Pandaboard had much slower interconnects. Therefore, communication acted as a performance bottleneck for the Pandaboard. On the other hand, Viridis and Intel nodes showed better scaling results due to their efficient interconnects. Energy efficiency for both the nodes reduced when scaled across nodes, but at much slower rate. Intel nodes showed energy efficiency about twice as that of the Viridis cluster.

From **Table 7.1** it can also be seen that the Intel cluster provided about 17 times performance of the performance of Viridis cluster with 32 cores. From the results, for the calculation involving 3D-FFT one can expect to see reduced performance while scaling across the nodes.

Pandaboards are expected to impact severely while scaling. It can be said that for the application containing FFT calculations, low latency and higher bandwidth network are the critical factor for scaling.

### 7.1.2 Conjugate Gradient (CG)

This benchmark used to calculate smallest eigenvalue of large, sparse and definite matrix. During this calculation a vector matrix multiplication of the sparse matrix and long distance irregular communication occurs. This makes the benchmark to have irregular memory accesses and communications [75].

#### Results and Analysis

**Figure 7.2** is showing the performance of the clusters for the CG benchmark on the energy efficiency metric with corresponding data in **Table 7.2**.

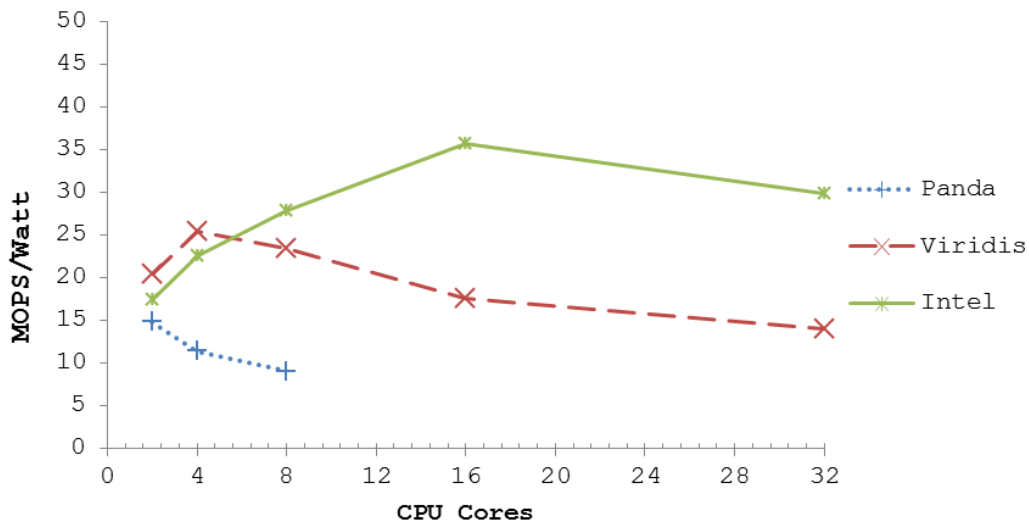


Figure 7.2: Energy Efficiency for the NAS CG Benchmark

For CG benchmark as well, as expected the scaling within the node showed better energy efficiency. Unlike the FT benchmark, Pandaboard showed better scaling for the CG benchmark. This was because the CG benchmark is not heavily dominated by the communication. Energy efficiency seems to be reducing for the for the Viridis and Intel cluster while scaling across the nodes.

Core Count	Intel MOPS	Intel MOPS/Watt	Viridis MOPS	Viridis MOPS/Watt	Panda MOPS	Panda MOPS/Watt
2	1718.2	23.2	169.9	20.4	77.9	14.8
4	3133.7	30.2	254.5	25.4	116.2	11.4
8	4489.3	36.2	455.4	23.4	185.3	9.1
16	8633.6	50.5	650.7	17.6	-	-
32	17029.6	47.6	973.9	14.0	-	-

Table 7.2: Energy Efficiency for the NAS CG Benchmark

**Table 7.2** shows that the Intel cluster again showed the performance about 17 times of the performance of Viridis cluster with 32 cores. It can also be seen that the Viridis cluster

showed twice performance as that of the Pandaboards. CG benchmark contains long distance communication and random memory access. These performance results clearly map to the combined results on the RandomAccess and Pingpong benchmark for the clusters.

From the results it can be said that the applications involving a significant conjugate gradient calculation are expected to show consistent scaling results on the Intel and Viridis nodes. Pandaboards can show a good single node performance but it is not expected to show as good scaling as the other two clusters.

### 7.1.3 MultiGrid (MG)

MG involves calculation on the sequence of meshes involving both short and long distance communications. Unlike to CG the MG benchmark has highly structured communication pattern. It is memory intensive benchmark [75].

#### Results and Analysis

Similar to the previous benchmarks, **Figure 7.3** and **Table 7.3** are showing the information about the energy efficiency of the clusters for the MG benchmark.

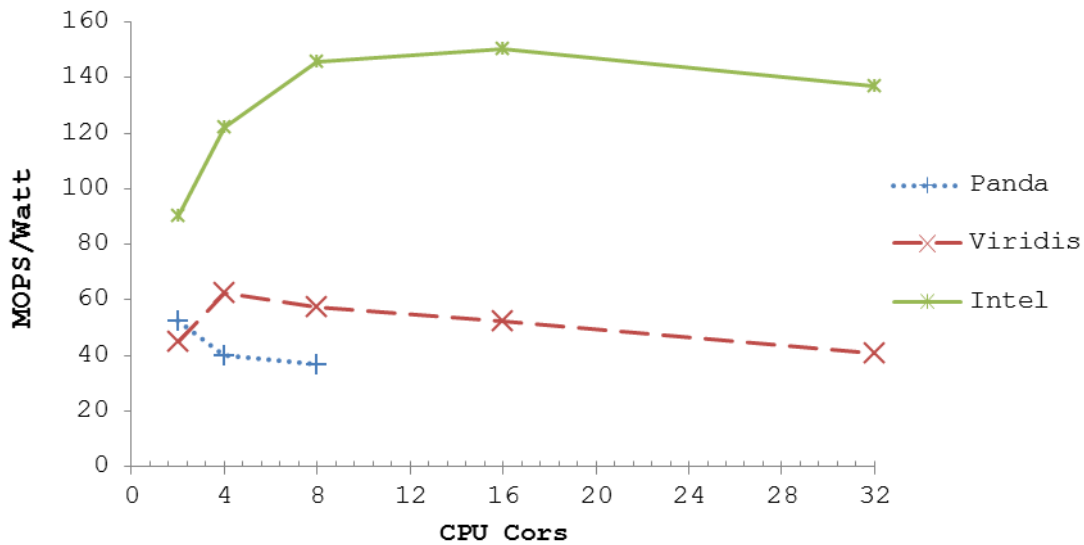


Figure 7.3: Energy Efficiency for the NAS MG Benchmark

Core Count	Intel MOPS	Intel MOPS/Watt	Viridis MOPS	Viridis MOPS/Watt	Panda MOPS	Panda MOPS/Watt
2	6980.3	90.3	355.4	44.9	288.7	52.5
4	12456.4	122.0	590.4	62.6	407.7	40.0
8	18527.2	145.5	1060.9	57.4	744.6	36.7
16	25216.3	150.5	1926.0	52.4	-	-
32	46659.9	136.7	2811.0	40.8	-	-

Table 7.3: Energy Efficiency for the NAS MG Benchmark

For MG benchmark as well Intel nodes provided better performance and the higher energy efficiency. From the result it can be seen that the performance of the Pandaboard dropped significantly by about 20%, when scaled across two nodes. This was due to slow network of

the Pandaboards, but then there just about 10% performance drop, when it scaled across four nodes from two nodes.

Similar to the CG benchmark, Intel cluster showed the performance about 17 times the performance of the Viridis cluster. Unlike the CG benchmarks, Pandaboards showed more than 70% performance as that of the Viridis cluster with eight cores on MG benchmark. As MG is memory intensive benchmark, the use of on-chip memory was useful to maintain the performance.

Energy efficiency results for both the Intel and Viridis cluster showed the best consistency among the selected application specific benchmarks. Results from **Table 7.3** also shows that both the Intel and Viridis cluster showed a very good scaling across the nodes. Efficient low latency interconnect of the Intel cluster and unique packaging of the Viridis cluster were the main reasons behind this. Applications involving the Multi-grid calculation are expected scale quite nicely on the Intel and Viridis cluster with quite consistent energy efficiency.

## 7.2 Summary

From the results for the application specific benchmarks it could be seen that the applications having FT calculations will be dominated by the communication. The energy efficiency of the cluster would heavily depend on the type of interconnects they have used. Inversely, the applications involving the Multi-grid calculations are expected to scale well and also show the better energy efficiency. Applications involving the CG calculations are in the middle of the stack. They are expected to show better scaling and energy efficiency than the communication dominated applications.

It was also seen that the performance of the selected benchmarks on the Intel cluster was about 17 times compared with the performance on the Viridis cluster. The performance of the memory access, bandwidth and interconnects were the key factors behind this. It was also seen that the on-chip memory of the Pandaboards helped them achieve a better performance score comparable to the Viridis cluster.



# Chapter 8

## Scientific Applications

To understand the role of embedded SoC based processors and systems for the HPC systems, it is necessary to analyze how these systems perform for the real-world HPC applications. For this purpose MIMD Lattice Computation (MILC) and Algebraic Multigrid (AMG) applications were selected. These two applications were used during ISC'13 Student Cluster Challenge for configuring and executing on the built cluster. Quick overview of these applications is given below. It is followed by the results and analysis on the energy efficiency of these applications.

### 8.1 MILC

MIMD Lattice Computation (MILC) is a set of applications from Quantum Chromodynamics (QCD) domain. Application named `su3_rmd` was selected for the purpose of benchmarking as it is most commonly used approach for benchmarking the MILC[71]. The `su3_rmd` application uses R algorithm to create a simple gauge configuration that are used in the physics projects [76]. There are different algorithms and methods available to perform same set of calculations. However, objective of the project is to measure the energy efficiency of the systems for a real-world scientific application. Therefore, the `su3_rmd` was used.

Applications in the MILC can be configured to display its performance to perform conjugate gradient calculation, which is represented in the MFLOPS. This approach is commonly used for benchmarking the MILC applications [71]. Same metric is used to calculate energy efficiency of all the clusters in the MFLOPS/Watt format, for a selected fixed problem size for the `su3_rmd` application.

#### 8.1.1 MILC Overview

There were similar compiler optimizations done for the MILC as that of the Linpack benchmark. They included a loop unrolling, providing information about the underlying platform, ABI unit and optimization level. They applied incrementally and then the final combination was selected. Detailed discussion on this was not provided as the reasons behind these compiler optimizations were already discussed during the Linpack optimization.

There is a suite of SciDAC libraries specially built for the applications in lattice QCD domain. They are provided by the USQCD, a collaboration of US scientists [78]. These libraries were used for the optimization of the MILC. This section contains the optimization results and quick overview of these libraries. It is followed by the scaling and energy efficiency discussion on the different clusters. All the results and discussion in this section is for the selected `su3_rmd` application in the set of applications from MILC.

**Figure 8.1** shows the execution time for the different types of optimizations. These results were for a single core and with the smaller problem size of  $8 \times 8 \times 8$ .

"None" represents the execution time with optimization `-O0`. Such low optimization level is less frequently used in practice. Here they it is used to show the impact of compiler optimizations. "Compiler optimizations" represents the performance of the application after performing the compiler optimizations. "SciDAC" optimizations represent the results after using SciDAC libraries.

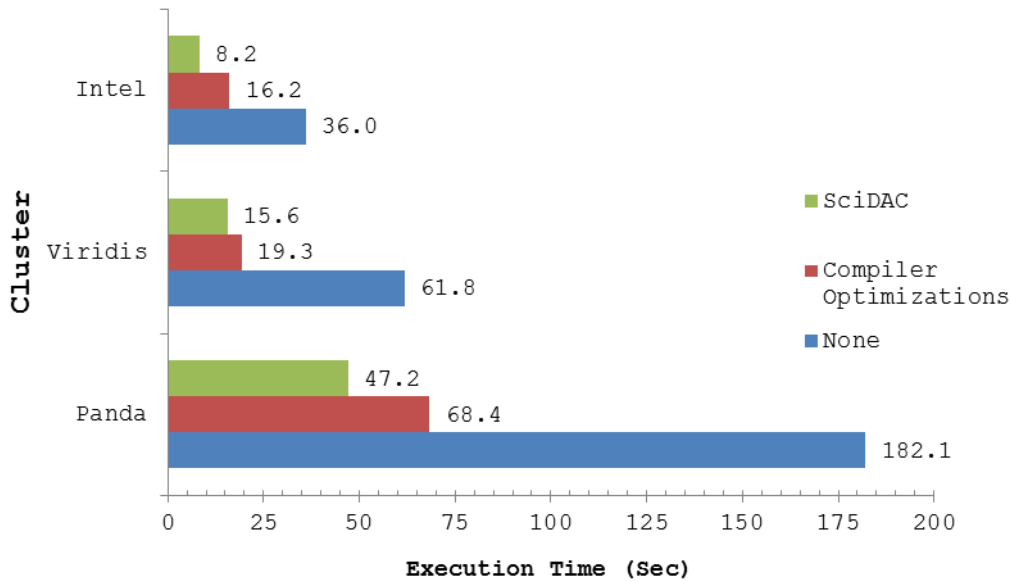


Figure 8.1: Execution time for the different optimization for the `su3_rmd` application

It can be seen that the compiler optimization gave significant performance gain. Similar to the Linpack benchmark, use of `-O3` flag performed most of the optimizations. It was interesting to see that the Viridis core achieved the performance ( $\sim 19.3$  sec) which was very close to performance of the Intel core ( $\sim 16.2$ ). After performing optimizations with the SciDAC libraries, Intel core got benefited much more than the Viridis ones. This was mainly because Intel cores had complex circuitry and AVX instruction set which was exploited by the SciDAC libraries to achieve higher performance. While ARM based core contained simpler cores with NEON extension to achieve higher energy efficiency. SciDAC library used these features which caused Intel cores to get more benefited. Due to these reasons in overall comparison the SciDAC libraries gave a significant amount of the performance gains for all the three types of clusters. The used SciDAC libraries and the reason behind the improved performance are given in detail below.

There were five SciDAC libraries used for the `su3_rmd` application. They were QMP, QIO, QLA, QDP and QOQDP.

- QMP library provides a communication layer which is specially designed for the lattice QCD applications. It is implemented over MPICH2 and optimized for the repetitive and nearest neighbor communications [78].
- QIO library provides the routines to support different types of IO functionalities for the lattice data [78].
- QLA library provides an interface to the linear algebra routines. Some of these routines in this library are highly optimized to achieve a high performance on the known architec-



tures. During installation of this library, special care was taken to provide all the platform specific optimization information to the compiler [78].

- QDP library provide the data parallel interface that is optimized for many architectures. These optimizations are done by providing the routines that can be optimized by SSE extensions. Routines from this library are used by the QLA library. QDP uses QMP library for the communication [78].
- QOQPD is high level library that contains various force and inverter terms that are used in QCD applications. They are optimized using low level QDP library [78].

### 8.1.2 MILC Scaling

Figure 8.2, 8.3 and 8.4 are showing the scaling results for the MILC on the Pandaboard, Viridis and Intel clusters respectively.

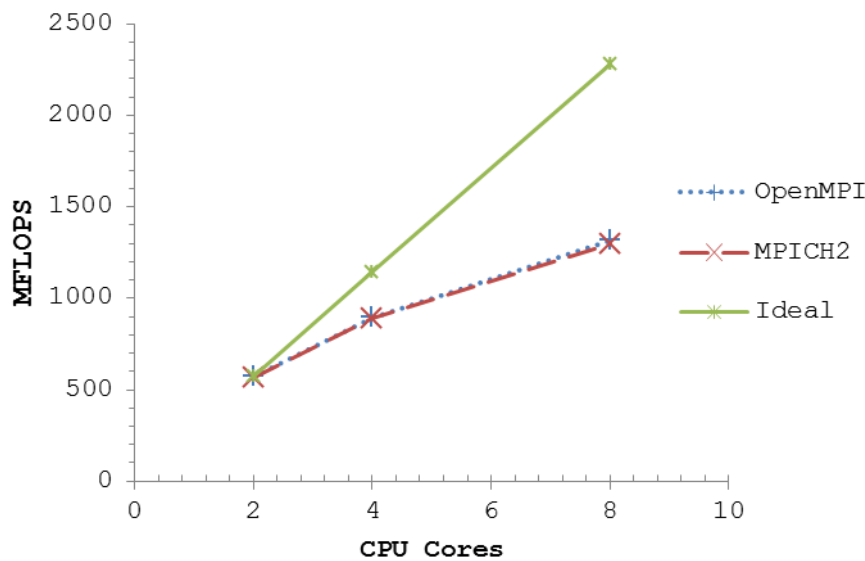


Figure 8.2: Scaling of MILC on Pandaboard Cluster

From the figures it can be seen that the Pandaboard showed very poor scaling for the MILC. On the other hand, the Viridis cluster showed better scaling results compared to other two clusters. It can also be seen that the scaling results for the Viridis nodes were reflecting the architecture of the cluster. While scaling from 4 to 32 cores the consistent performance drop was observed. The obvious reason behind performance drop was the communications done over the network. It was interesting to observe that for Viridis cluster the scaling across the energy card was as good as the scaling within the energy card. The reason behind this was that each node in the Viridis chassis was connected to every other node using 10GbE. On card switch handled the communication between the nodes. This made each node to communicate with every other node directly as a single homogeneous system.

**Table 8.1** shows the values in MFOP/sec for the MPI which performed the best for a particular cluster.

Similar to Viridis nodes the Intel nodes also showed good scaling across nodes but within a node the scaling was not perfect. One of the possible reasons behind this was the communication overhead. Due to less communication overhead with the less core count the lesser core count showed a better performance. All MPI libraries showed almost the same performance

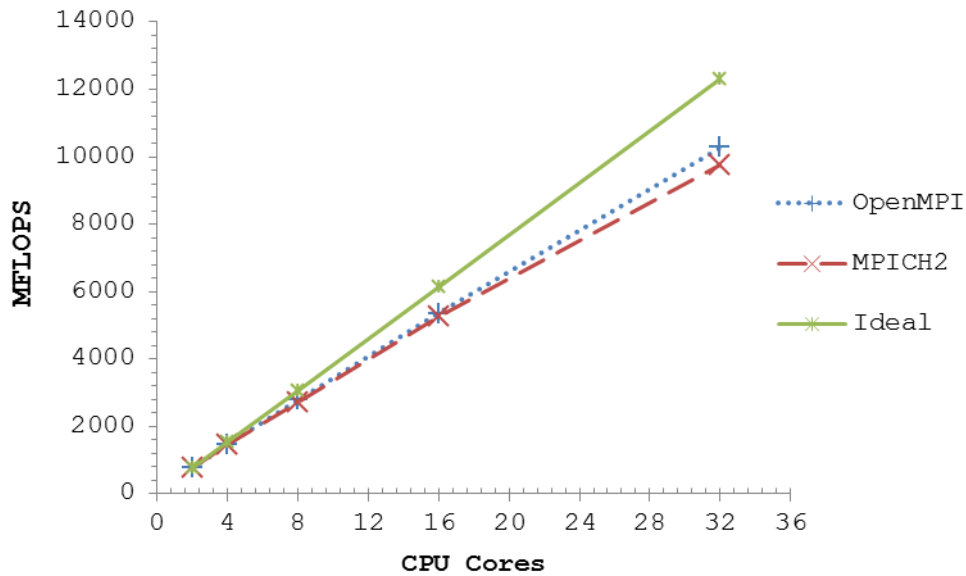


Figure 8.3: Scaling of MILC on Viridis Cluster

Core Count	Intel MFOPS	Intel MFOPS/Watt	Viridis MFOPS	Viridis MFOPS/Watt	Panda MFOPS	Panda MFOPS/Watt
2	10154.5	137.2	824.3	90.05	570.4	102.6
4	19352.2	186.2	1471.2	156.6	894.4	68.5
8	33873.3	262.6	2772.1	131.4	1315.1	41.3
16	49894.9	280.4	5340.2	105.3	-	-
32	96308.8	261.0	10270.1	80.7	-	-

Table 8.1: Performance of MILC scaling in MFLOP/sec for different clusters

within the node. There were slight better results provided by the Intel MPI while scaling across the nodes.

It could also be seen that for the MILC `su3_rmd` application the Intel cluster showed about ten times performance (MFLOPS) as that of the Viridis cluster, both with 32 cores. Viridis cluster showed a performance about twice the performance of the Pandaboard while compared with the eight cores. Results for the Pandaboard and Viridis clusters were quite similar to the results of the CG benchmark. However, results for the Intel cluster were quite lesser than the results of the CG benchmarks.

### 8.1.3 MILC Energy Efficiency

**Figure 8.5** shows the energy efficiency results for the `su3_rmd` application from the MILC application set.

It was discussed earlier that the application contains a significant amount of conjugate gradient calculation. The selected application `su3_rmd` also showed the similar results as that of the conjugate gradient (CG) benchmark from NPB suite. It can be seen that the Intel nodes again showed the higher energy efficiency along with the better performance compared to ARM based nodes. They showed approximately 2.5 times energy efficiency of the Viridis nodes and more than about six times the energy efficiency of the Pandaboards.

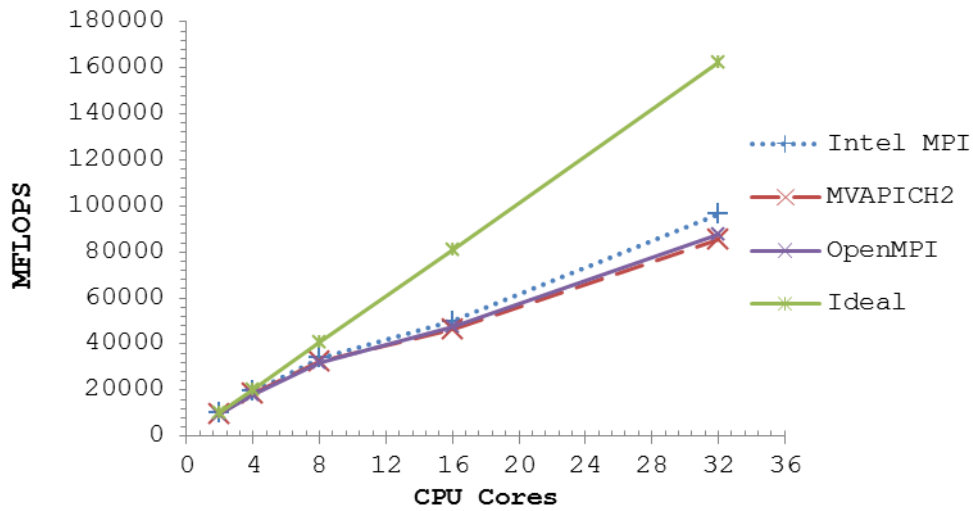


Figure 8.4: Scaling of MILC on Intel Cluster

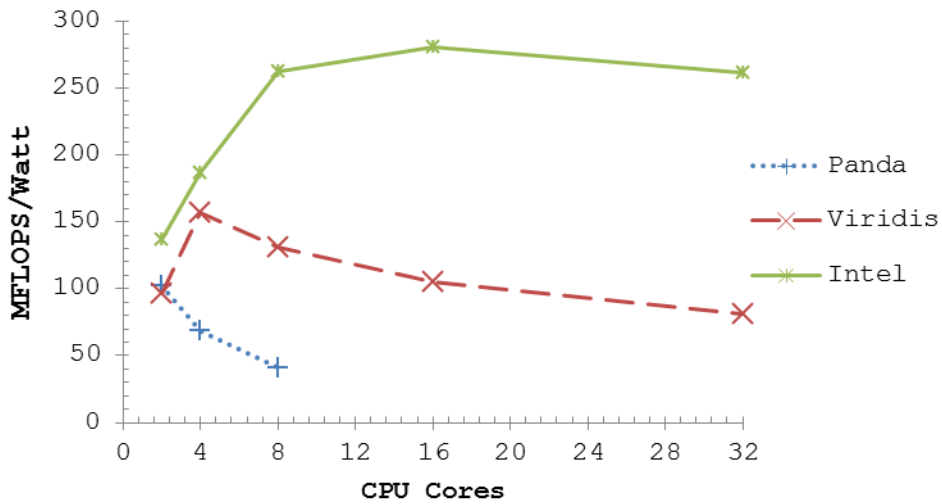


Figure 8.5: Results on Performance/Watt matrix for the MILC application

Pandaboard nodes showed significant drop in the performance when scaled across the multiple nodes. From the performance on the CG benchmark this behavior was expected. The reason behind this was the same, slow network speed of the Pandaboard. Good scaling of the application for the Viridis cluster helped it to maintain the energy efficiency while scaling across the nodes.

## 8.2 AMG

Algebraic Multigrid (AMG) application is used to calculate the discretized differential equations. It is used in many applications like oil reservoir simulation, problems from CFD domain, multi-pass interpolation and electromagnetic systems. It has Multigrid (MG) calculation as one of the building blocks [77].

To measure the performance of the AMG application it provides a single number at the end of its run. There is no metric provided for it. It uses below formula to calculate this number.

$$\text{System Size} * \text{Iterations} / \text{Solve Phase Time}$$

In the formula, a system size is the core count. Iterations are selected by the application depending on the input values of the process count and problem size. Solve phase time is the time taken for the actual computation. This performance number is then converted into performance/Watt matrix by dividing it by average power consumed during execution of the benchmark.

### Results and Analysis

Similar to the MILC application there were optimization and scaling tests done. In this section only the energy efficiency results are discussed. Results for the scaling tests are provided in the Appendix C. Similar optimizations as that of the Linpack benchmarks were used for optimizing AMG. They included a loop unrolling and information about the underlying hardware. To avoid repetition this section does not contain the discussion on them. Unlike MILC, there were no special libraries used for optimizing AMG.

**Table 8.2** contains the values for both performance and performance/Watt metric for the AMG application. Problem size 100x100x100 was selected for all the tests. As discussed earlier, AMG application provided performance as single number at the end of application run. This number was used as a measure of performance of the application run. Same number was used as a metric to measure the performance of AMG during the ISC'13 SCC. There was no unit like MFLOP/sec provided for this number. This number was very big so it was divided by 1000 to convert it into the kUnits of performance along with better readability. Values in the **Table 8.2** contain the figures in kUnits.

Core Count	Intel Perf.	Intel Perf./Watt	Viridis Perf.	Viridis Perf./Watt	Panda Perf.	Panda Perf./Watt
2	8332.6	66.2	318.9	42.6	288.8	54.4
4	15030.9	113.7	585.6	66.9	551.8	55.0
8	21104.3	142.7	1165.3	69.2	1055.9	54.5
16	43913.7	246.8	2011.1	62.6	-	-
32	88532.6	229.9	3752.9	59.7	-	-

Table 8.2: Linpack scaling the Intel Cluster

**Figure 8.6** is showing the energy efficiency of the clusters for the AMG application. Similar to other benchmarks, scaling within node showed the improved energy efficiency. Intel cluster again outperformed the other two clusters. They showed about twice energy efficiency compared to the Viridius cluster when used with 32 cores.

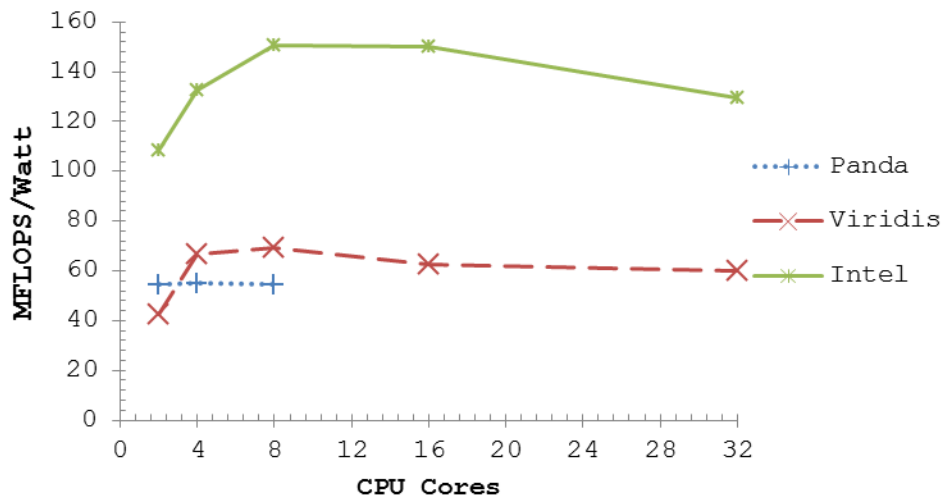


Figure 8.6: Results on performance/Watt metric for the AMG application

It can be seen from the **Figure 8.6** and **Table 8.2** that the AMG application scaled very well on all the three clusters. Results for the MG benchmark from NPB suite showed similar prediction about the performance and energy efficiency. AMG application showed better scalability than the MG benchmark. Unlike the other benchmark and application results, Pandaboard cluster also showed almost consistent energy efficiency while scaling across the nodes. This was clearly because the application was not dominated by the communication.

### 8.3 Summary

In this chapter we have seen the performance of the clusters on the real-world scientific applications. Pattern of the energy efficiency was quite similar to the energy efficiency pattern, generate by the cluster for the application specific benchmarks from NPB suite. It was interesting to see that the Intel nodes outperformed on the energy efficiency matrix as well.

Pandaboards showed a good single node performance but they got impacted by its slow network while scaling across the nodes. However, they showed a good scaling and energy efficiency for the AMG application which was not communication intensive.

Viridius cluster was in the middle of the stack, both in terms of performance and energy efficiency. It was observed that the Viridius nodes had improvements in the key areas where the Pandaboards faced the performance bottlenecks. Faster interconnects were one them which helped the Viridius nodes achieve a good scaling results compared to other two clusters. Higher operating frequency also benefited the Viridius nodes to achieve a higher performance than the Pandaboards.



## Chapter 9

# ISC'13 Student Cluster Challenge Work

This chapter describes the work done for the ISC'13 Student Cluster Challenge [13].

### 9.1 Overview

During the preparation phase (from 27<sup>th</sup> May to 14<sup>th</sup> of June) four of us worked in a team. The work involved the performing analysis of power consumption of the different hardware, selecting new architecture and hardware for the cluster. During this phase there was a hardware training provided by the vendors at their location. There were different experiments conducted to finalize the cluster configuration. Author was mainly responsible for the MILC application. However, there was a huge amount of administrative work which was done whenever required. There was a GPU version of the MILC configured and installed. All this work is explained in detail in this chapter.

### 9.2 Training Trip

There was a training scheduled for the three days at the location of the vendors. On the first and second day the training was scheduled at Boston Ltd. There we learned about the different hardware components and assembling them into the compute blade of a cluster. Training also covered the server provisioning, cluster management software suite and regular administrative activities of cluster maintenance.

On first and second day of the training there was an attempt done to configure Ubuntu 13.04 server as an OS. The main reason behind this was that the provided CentOS 6.2 was consuming about 1.75GB out of 64GB memory for running the OS and related processes. For Linpack benchmark we required maximum memory as we were using about 95% of memory. Installation of Ubuntu required configuring complete software stack for it. Additionally, there was a huge amount of effort spent to configure the Infiniband interconnect on the Ubuntu. After configuring the software stack the Linpack benchmark was installed and tested. It was found that the performance of the Linpack was not as good as that of the CentOS. Therefore decision was taken to rollback to CentOS. After few days it was found that the problem was not with the OS but with one of the GPU. This caused the compute node to give low performance on the Linpack.

Third day of the training was at the Viglen office. There was an effort spent on the applications and Linpack benchmark. There were application experts for the GROMACS and Linpack

benchmark from Viglen and NVIDIA respectively coordinated the effort.

### 9.3 Cluster and software Configuration

After the training trip there were efforts spent by the team on installing different versions of MPI and MKL. There were different articles about the libraries and optimizations which claimed to have higher performance were attempted. It included installation of cublas from NVIDIA [79]. These experiments involved configuring benchmarks, applications for these libraries and MPI versions for which individual was responsible for. These experiments were continued till disassembling the cluster for shipping to Germany.

### 9.4 MILC on GPUs

Along with the work mentioned above, there were efforts done to achieve a higher performance for the MILC application. It was done independently as author was responsible for the application.

There was an alpha version of the MILC 7.7.10-a8 released on 1<sup>st</sup> June 2013, just two weeks before the competition date. This version was containing the code capable of running on the GPUs [80]. Since the selected cluster was containing four nodes with two GPUs each, the GPU version was installed on the cluster. This installation was extremely challenging compared to other MILC installations. It required configuring another library QUDA 0.5.0 containing the QCD routines running on GPU. It also required installing the QUDA to MILC interface [80]. To compile MILC with GPU, the make file was configured with the changes and installation was performed.

QUDA installation was a challenging task in itself. It was identified that the alpha release of the MILC was not fully compatible with the QUDA library. There was an issue with the conditional macros. To resolve this it required changing the order in which header files were included in the several source files. After this installation it was identified that make file QUDA MILC was broken and missed to compile some of the source files. These files were compiled manually. After all these issues, finally, the MILC GPU compatible version was installed successfully.

**Figure 9.1** is showing the scaling results for the MILC CPU only and GPU version.

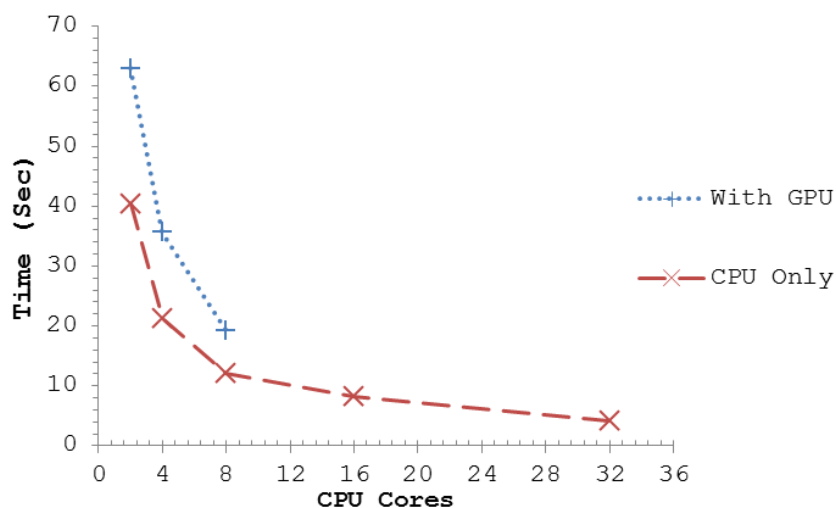
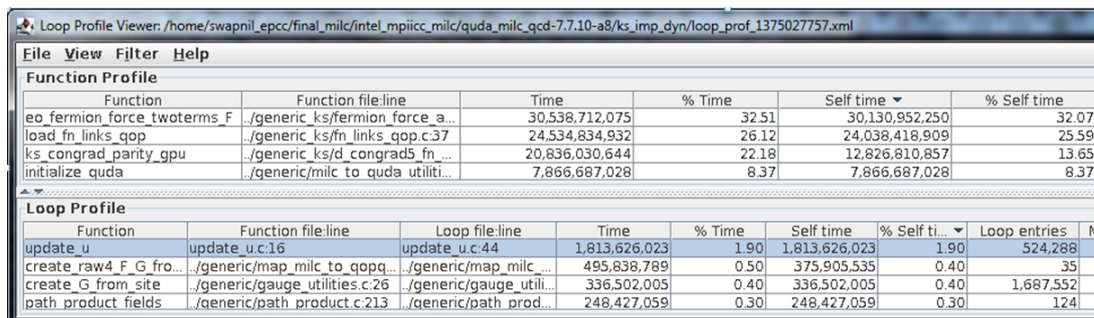


Figure 9.1: Scaling with and without GPU version of the MILC



MILC GPU version turned out to be slower than the CPU only version. Reason behind this was the QUDA library was containing a single GPU version of the routines [81]. The QUDA library was using a single MPI process per GPU. This caused the MILC GPU version to use only eight MPI processes while using all the available GPUs. On the other hand, CPU version of the code used 64 MPI processes. Additionally, only part of the MILC calculation was moved to the GPUs using QUDA library. The remaining part was just using CPUs to perform the calculation. For a CPU only approach there was no overhead of offloading data to the GPU. This made the GPU version of the MILC to run slower than the CPU version.

There was also a support for the OpenMP, added in this alpha release of the MILC [81]. It was very small portion which was ported to use OpenMP threads for the calculation. There was also investigation done to add the OpenMP threads to the remaining code. It was done in order to maximize the use of CPUs when only two MPI processes were running with a GPU version. It was also a difficult task to figure out the use of an OpenMP in the code using the existing framework. The code contained wrapper macros defined around the OpenMP directives. After figuring out the appropriate use of wrappers, the profiling of the code was done to identify the computationally intensive loops. **Figure 9.2** is the screenshot of the Intel profiler which is showing the functions and loops consuming the maximum amount of time.



The screenshot shows the Intel Profiler Loop Profile Viewer interface. The title bar indicates the file path: /home/swapnil\_epcc/final\_milc/intel\_mpiicc\_milc/quda\_milc\_qcd-7.7.10-a8/ks\_imp\_dyn/loop\_prof\_1375027757.xml. The interface has a menu bar with 'File', 'View', 'Filter', and 'Help'. There are two main sections: 'Function Profile' and 'Loop Profile', each with a table of data.

Function	Function file:line	Time	% Time	Self time	% Self time
eo_fermion_force_twoterms_F	./generic_ks/fermion_force_a...	30,538,712,075	32.51	30,130,952,250	32.07
load_fn_links_qop	./generic_ks/fn_links_qop.c:37	24,534,834,932	26.12	24,038,418,909	25.59
ks_congrad_parity_gpu	./generic_ks/d_congrad5 fn...	20,836,030,644	22.18	12,826,810,857	13.65
initialize_quda	./generic/milc_to_quda_utiliti...	7,866,687,028	8.37	7,866,687,028	8.37

Function	Function file:line	Loop file:line	Time	% Time	Self time	% Self ti...	Loop entries	M
update_u	update_u.c:16	update_u.c:44	1,813,626,023	1.90	1,813,626,023	1.90	524,288	
create_raw4_F_G_fro...	./generic/map_milc_to_gopq...	./generic/map_milc...	495,838,789	0.50	375,905,535	0.40	35	
create_G_from_site	./generic/gauge_utilities.c:26	./generic/gauge_utili...	336,502,005	0.40	336,502,005	0.40	1,687,552	
path_product_fields	./generic/path_product.c:213	./generic/path prod...	248,427,059	0.30	248,427,059	0.30	124	

Figure 9.2: Intel profiler screenshot showing the functions and loops taking the maximum time

Output of this profiler was used to identify the loops and function to focus the efforts to change them to use OpenMP threads. Adding OpenMP threads broke the code. It produced inconsistent output solution. Considering the risk of breaking the code and required amount of time for the changes the plan of adding OpenMP directives was dropped.

## 9.5 MILC Benchmarking

There were standard benchmarks available for the other applications. As MILC was the set of applications, different people used different applications for the benchmarking purpose. Exact application for the benchmarking was not known till the day of competition. This required cumbersome task of making sure that all the applications were running and documenting the steps to run. During the preparation phase, developers of the application helped to focus the efforts on few applications. They did not disclose the actual application to maintain the fairness of the competition. MILC developers suggested the applications from the `ks_imp_dyn` and `ks_imp_rhmc` application set [82]. These sets contained roughly eight to ten applications. All these applications were installed and tested. GPU version for these applications was also tested. Different applications required different set of inputs which required the understanding its significance in the application run. During the benchmarking an actual data set was provided which was expected to be distributed over the cluster. However, this fact was unknown initially. Therefore it required a lot of scaling tests and different combinations of the input

for each possible application to get the best performance out of cluster. This was extremely cumbersome and time consuming task.

## 9.6 HPCC Hack

We used the GPU version of a Linpack for the highest Linpack award competition. During the other part of competition the score of the HPCC benchmark suite was going to be calculated. HPCC benchmark suite also contained a Linpack calculation. However, the HPCC code was integrated with the Linpack code which was using CPUs only. HPCC suite produced a single executable which ran all the benchmarks in the HPCC suite. The analysis of the code was done to understand the internal working of the code. It was found that when the HPCC executable was started with some MPI processes then those processes were directly entering into the Linpack code. To get benefited by the GPU version of a Linpack for the HPCC score, an attempt was made to integrate external binary with the existing code. It required the code changes to the Linpack code. There was separate "system" system call made from the HPCC code to call the external binary. These hacky changes were made to achieve a better score for the HPCC suite.

## 9.7 Summary

Participation in the ISC'13 Student Cluster Challenge provided an experience with the real-world applications and hardware. Working in a team made us learn from each other by sharing thoughts and solving problems. Team managed to finish second the highest Linpack score with 8.321 TFLOPS. Additionally, in spite of using the minimum number of nodes, we managed to come first on one and second on the other MILC dataset [83]. This made the team from EPCC to make successful debut in the cluster competition.

# Chapter 10

## Project Conclusion

This Chapter provides the conclusion derived from the results obtained from this project. It is followed by the future work and project evaluation sections.

### 10.1 Conclusion

Use of SoC based mobile processors for the HPC systems is widely discussed topic in the HPC community. Considering this background, an analysis of the energy efficient SoC based architectures was done in this project. It involved case study of Blue Gene/Q and current initiatives by the Project Mont-Blanc. The trend was observed towards combining the embedded SoC based architectures and accelerators for the higher performance and energy efficiency. For this project the focus was given on the energy efficiency of SoC based processors.

To perform the energy efficiency analysis there were three clusters built during this project. Intel and Viridis cluster were built with the help of Boston Staff. It gave a learning experience of setting up and configuring a cluster environment. Two of the clusters were from ARM based SoCs, the Pandaboards and Boston Viridis cluster. Intel Xeon cluster presented the requirements of the current HPC systems. To analyze the energy efficiency for the HPC application the benchmarks were selected from three categories. Firstly, synthetic benchmarks were used to measure the performance critical aspects of the system. Secondly, three benchmarks from the NPB suite were selected. They were more application specific. Finally, the real-world scientific applications AMG and MILC were selected.

Intel nodes were expected to perform better on the performance metric. It was interesting to see them leading the table of energy efficiency as well. On the benchmarks as well as for the real-world applications the Intel cluster showed about twice energy efficiency as that of the Viridis nodes. Higher energy efficiency of the Intel processors was due to their excellent floating point performance and sophisticated hardware. While ARM based processors were lagging the floating point performance. This shows that it will take some more time to deliver HPC suitable solutions with SoC based processors. Next ARM architecture ARMv8 is also focused on the performance along with the energy efficiency.

Another important aspect of the project was to understand the novel architecture of the Boston Viridis system and its performance and energy efficiency. Viridis product targeted for the server market of web servers, cloud and data analytics, not the HPC market. However, its study revealed many interesting aspects of SoC based system and their potential to serve the future HPC systems. They showed a better idle power management using special on-chip microprocessor. Their unique EnergyCard architecture and its packaging showed very good scaling results. In comparison with the Pandaboard cluster, it was observed the Viridis nodes showed higher performance by using efficient interconnect, on-board switch, more core count

and higher operating frequency. On the other hand, Pandaboard showed the benefits of having on-chip memory for better performance. Dense packaging of the Viridis nodes showed the best scaling results among all the processors. This dense packing reminds the packaging of the BlueGene/Q and importance of it for the energy efficiency. ARM based processors have benefit of integrating ARM based GPUs on-chip. From the analysis it could be said that by having processors, memory and accelerators with homogenous instruction set on a same chip, the SoC based processors can provide much higher performance and energy efficiency.

Currently SoC based processors are moving with energy efficiency towards the performance while the x86 based processors are moving with performance to the energy efficiency. Additionally, the profitable and fast growing market of mobile processors is funding the research for the powerful and energy efficient chips. This will make the SoC based chips cheaper as well. Anticipating from the current driving force from the market it could be said future HPC systems are expected to be more energy efficient. There is huge possibility for embedded SoC based mobile processors to power the HPC system by combining them with the on-chip accelerators.

## 10.2 Future Work

In this section we will discuss the work that can be done to extend the analysis done during this project.

This project was focused towards understanding the power efficiency and the architectural benefits of SoC based processors for the HPC system. This analysis can further be extended to the analysis of processors and accelerators on the same card. As discussed earlier, Mont-Blanc project is also planning to build next cluster on the same concept. Unlike the current systems, such hybrid systems will be having both CPU and accelerator like GPU based on the same instruction set. It will be interesting to see how these SoC based hybrid systems address the challenges for the current hybrid architectures. During this project it was also identified that the SoC based system has capacity to do better power management. Analysis can be done to understand the idle power consumption of these hybrid architectures and to identify ways to improve them.

It will be ambitious but interesting as well, to perform research on the programming models to achieve a better performance out of on-chip processors and accelerator architecture. It can also be extended to use architecture compliant compilers to achieve a better energy efficiency along with the performance.

It was not discussed but it was noticed that the optimizations had an impact on the power consumption of the system. It was obviously due to more hardware usage by the instructions like SIMD extensions. These instructions use more integer and floating point units to perform more operations in a single clock cycle. It was also observed that the less sophisticated hardware provides lower performance but higher energy efficiency. Research can be made to combine these two aspects. One could analyze the possible ways in which a compiler could generate the energy efficient code for less sophisticated processors. It was discussed that there are features provided by modern processors to perform dynamic power management. Research can also be performed to use compiler to execute special instructions to optimize the energy efficiency of the application run.

There were different patterns of energy efficiency observed for the different applications on different hardware. This could be extended to create a performance model to predict the performance along with the energy efficiency for a particular application and dataset.

There were a lot of difficulties faced during the cluster building with the Pandaboard. It was mainly due to there is no hardware support provided like IPMI which is used by the commonly used cluster management software. There is also support available by the community to configure these boards for the network booting. One could use these features to customize the existing cluster management tool or create new one to build and manage the cluster using ARM based boards.

### 10.3 Project Evaluation

This section contains the discussion about the distribution of the efforts spent during the dissertation. There were a huge number of challenges faced during the different phases of the dissertation. This chapter contains a quick overview of some them.

**Figure 10.1** is showing the effort distribution during the dissertation.

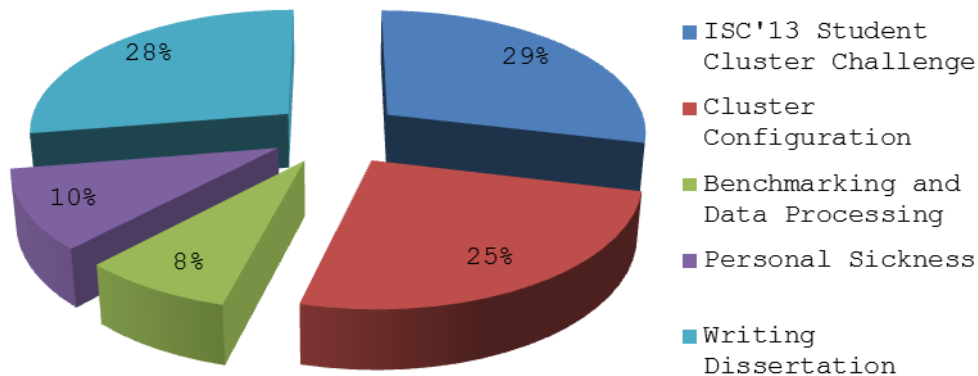


Figure 10.1: Dissertation work distribution

This project was based on the background of ISC'13 Student Cluster Challenge. Significant amount of time (about one third) during the dissertation was spent on the preparation and participation in the competition. There were a large number of issues handled by the team. They ranged from fixing the fuse of the cluster, till the installation and provisioning of the cluster.

After the competition a Pandaboard cluster was built from the scratch. Both Pandaboard and Viridis cluster were configured with software stack, benchmarks and applications. It took another significant portion of the time during the dissertation. Initially, cluster from the seven Pandaboards and three CARMA DevKits was planned to be built. It was identified that one of the Pandaboards was dead. Hence only six boards were used in the cluster. Original plan was to perform analysis on combining the low powered CPU and low powered GPU for higher energy efficiency. Unfortunately, one of the three available CARMA board had damaged root account. Account name was changed to 'oot' from 'root'. As there was only one sudo user available which was damaged, this board was not ready to be used. One of the remaining two boards had an issue that it was not detecting the keyboard. This made only one CARMA board available for use. To fix these errors there was a procedure to flush these boards with new OS was provided. By coordinating with SECO (manufacturers of the CARMA kit) the new kernel and flushing procedure was obtained. Flushing did not yield the expected results and it was consuming a lot of time. After the competition and all these attempts there was a very short time remaining for the experiment. Considering the risk of completing project on time, the decision was taken not to use CARMA boards. Viridis nodes were used instead of CARMA boards and focus was given the low powered CPUs alone.

There were many difficulties faced while using the remote Viridis and Intel cluster. Since there were many people using the system, it required a lot of scheduling and planning to use the systems. There were issues like crashing of the systems, user account expiring, unexpected power cut and issues after rebooting the node happened quite often. Impact of them was re-

duced by dynamically planning the activities and coordinating with the Boston Team.

Actual benchmarking and processing the results took comparatively less (5-6 days) amount of time during the whole project. There was significant amount of impact of personal sickness of about nine days on the project. Due to this many interesting extensions to the project had to be dropped. Finally, the writing part, it was another major portion of the dissertation. It took about three weeks of time as expected and planned.

Two of the main extensions are explained here which were dropped. There was a plan to create custom provisioning software for the Pandaboards. The PoC (proof-of-concept) for the PXE booting the Pandaboards was not complete. It was done partially but the plan had to be dropped due to insufficient time. Installation of the MILC, SciDAC libraries and MILC GPU version was extremely challenging to finish. There were many issues like insufficient documentation, broken make files, incompatible interfaces and their different versions. After successfully performing this installation, there was a plan to port application to the "**easybuild**" project. It is an open-source project which automates the building, installation of the application and its dependencies using a very few (usually one) commands [84].

Objective this project was to learn building the cluster from the SoC based processor, and performing its energy efficiency analysis for the real-world scientific application. Project has successfully achieved this objective. During the project preparation phase the plan was made to perform the analysis on the energy efficiency of the low powered CPUs along with the low powered GPUs. Due to issues with the hardware this objective had to be dropped. This risk was anticipated one so an immediate action was taken. Using Viridis system was identified as a backup plan for the hardware issues which turned out to be useful. There were other risks related to hardware shipping, no physical access for the power measurement and access to the cluster was identified. Impact of these risks was mitigated by coordinating within the team and Boston staff. Boston staff booked times slots for the team. They also provided feature get power measurements for the remote systems.

In conclusion, the cluster building activity turned out to be most challenging and time consuming. The unexpected issues with the hardware also shaped the final project. As the cluster competition was part of this project, it left very short time and scope to plan the project deliverables. Personal sickness impacted the deliverables heavily. However, all the issues and challenges provided a huge learning experience. In spite of occurrence of many risks, careful planning and experience from previous dissertations helped to deliver project on time.





# Appendices



# Appendix A

## Scripts

### Power Measurement Script

```
#!/usr/bin/perl

use Time::HiRes qw(usleep);
sub getTimeStamp{

    my $timeStamp;
    my $sec, $hr, $min, $mon, $day, $year, $other;

    ($sec, $min, $hr, $day, $mon, $year, $other)=localtime;
    $year+=1900;          #To compansate year diff.
    $mon+=1;             #To compansate month diff
    $timeStamp= sprintf \
    "%04d%02d%02d%02d%02d", $year, $mon, $day, $hr, $min, $sec;

    return $timeStamp;
}

my $currTime, $reading, $record, $filename;
$currTime=getTimeStamp();
$filename="Reading_".$currTime;
print "Power Measurement Started at ",$currTime,"\n";

for (;;) {
    #Sleep for half a second
    usleep(500000);

    $currTime=getTimeStamp(); # Get Current timestamp

    $record=$currTime;      # Initialize new record
    for( $i=0; $i<=7; $i++){
        #Get power reading of particular node
        $command="ipmitool -H epcc$i-ipmi -U admin -P admin -I
lanplus sensor get 'Node Power'|grep 'Sensor Reading'|awk -F':' '{print
\$2}'|awk '{print \$1}'";
        $reading=`$command`; # Execute command to return power
readings of particular node
        chomp($reading);      # Remove carrige return at the end
        $reading=sprintf "%.2f",$reading; # print reading in xxx.xx
format. i.e. 2 decimal places
        $record=$record.", ".$reading;
    }
    print $record."\n";
    `echo $record >>$filename`;
}
$currTime=getTimeStamp();
```

```
print "Power Measurement Stopped at ", $currTime, "\n";
```

## **NFS Auto-mounting Script**

```
#!/usr/bin/ksh
#
#Source : nfsAutomount.ksh

for i in {1..5}
do

    #Login to all Panda compute nodes and
    #mount all entries in the /etc/fstab

    ssh panda$i sudo mount -a

done

# Above Script was placed at the end of boot sequence of
# the panda head node using below command.

# update-rc.d nfsAutomount.ksh start 21 2 3 4 . start 31 5 \
    . stop 80 0 1 6
```

# Appendix B

## Linpack Sample output

### 1. Intel Cluster

```
=====
===
T/V          N    NB    P    Q          Time
Gflops
-----
---
WR00C2C1     32768  128    8    4          43.31
5.416e+02
HPL_pdgesv() start time Fri Aug  9 20:11:34 2013
HPL_pdgesv() end time   Fri Aug  9 20:12:18 2013
-----
---
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=      0.0020610 .....
PASSED
=====
===
```

### 2. Viridis Cluster

```
=====
===
T/V          N    NB    P    Q          Time
Gflops
-----
---
WR00C2L2     57344  128    8    4          3936.35
3.194e+01
HPL_pdgesv() start time Sat Jul 27 06:57:18 2013
HPL_pdgesv() end time   Sat Jul 27 08:03:43 2013
-----
---
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=      0.0005923 .....
PASSED
=====
```

### 3. Pandaboard Cluster

```
=====
===
T/V          N    NB    P    Q          Time
Gflops
-----
---
WR00C2L2     20480  192    5    2          982.50
5.829e+00
HPL_pdgesv() start time Tue Jul  9 13:12:25 2013

HPL_pdgesv() end time   Tue Jul  9 13:28:48 2013

-----
---
||Ax-b||_oo/(eps*(||A||_oo*||x||_oo+||b||_oo)*N)=      0.0012002 .....
PASSED
=====
===
```

# Appendix C

## Scaling Results for AMG

Core #	Intel MPI	Intel MVAPICH2	Intel OpenMPI	Viridis OpenMPI	Viridis MPICH2	Panda OpenMPI	Panda MPICH2
2	108.22	104.29	103.92	42.60	42.37	54.38	54.23
4	132.55	126.57	124.44	66.90	66.70	55.04	54.76
8	150.43	143.05	140.75	69.16	68.18	54.54	53.92
16	150.26	139.39	142.60	62.59	61.78	-	-
32	129.67	115.25	117.71	59.74	56.64	-	-





# Bibliography

- [1] "science prospects and benefits with exascale computing".  
[online]. Available From: [http://www.olcf.ornl.gov/wp-content/uploads/2010/03/Science-Case-\\_012808-v3\\_\\_final.pdf](http://www.olcf.ornl.gov/wp-content/uploads/2010/03/Science-Case-_012808-v3__final.pdf). [Accessed: 29th July 2013].
- [2] Summary report of the advanced scientific computing advisory committee (ascac) sub-committee.  
[online]. Available From: [http://science.energy.gov/~media/ascr/ascac/pdf/reports/exascale\\_subcommittee\\_report.pdf](http://science.energy.gov/~media/ascr/ascac/pdf/reports/exascale_subcommittee_report.pdf). [Accessed: 29th July 2013].
- [3] Kogge p. et al. exascale computing study: Technology challenges in achieving exascale systems.  
[online]. Available From: <http://www.cse.nd.edu/Reports/2008/TR-2008-13.pdf>. [Accessed: 29th July 2013].
- [4] Green 500 list. published at isc june 2013.  
[online]. Available From: <http://www.green500.org/lists/green201306>. [Accessed: 29th July 2013].
- [5] Top 500 list. published at isc june 2013.  
[online]. Available From: <http://www.top500.org/lists/2013/06/>. [Accessed: 29th July 2013].
- [6] Wikipedia article, powerpc\_400 architecture.  
[online]. Available From: [http://en.wikipedia.org/wiki/PowerPC\\_400](http://en.wikipedia.org/wiki/PowerPC_400). [Accessed: 29th July 2013].
- [7] Armv8 architecture.  
[online]. Available From: <http://www.arm.com/products/processors/armv8-architecture.php>. [Accessed: 29th July 2013].
- [8] Amd will build 64-bit arm based opteron cpus for servers, production in 2014.  
[online]. Available From: <http://www.anandtech.com/show/6418/amd-will-build-64bit-arm-based-opteron-cpus-for-servers-production-in-2014>. [Accessed: 29th July 2013].
- [9] Mont-blanc project objectives.  
[online]. Available From: <http://www.montblanc-project.eu/project/objectives>. [Accessed: 29th July 2013].
- [10] Mont-blanc hardware ready to test in july 2013, mont-blanc press release.  
[online]. Available From: [http://www.montblanc-project.eu/sites/default/files/sites/default/files/press-releases/\\_\\_primeurmagazine.com\\_live\\_LV-PL-06-13-26.pdf](http://www.montblanc-project.eu/sites/default/files/sites/default/files/press-releases/__primeurmagazine.com_live_LV-PL-06-13-26.pdf). [Accessed: 29th July 2013].

- [11] Boston viridis - arm® microservers.  
[online]. Available From: <http://www.boston.co.uk/solutions/viridis/default.aspx>. [Accessed: 29th July 2013].
- [12] Milkyway-2 supercomputer hardware configuration.  
[online]. Available From: <http://www.top500.org/system/177999>. [Accessed: 29th July 2013].
- [13] Isc'13 student cluster challenge, competition rules.  
[online]. Available From: <http://www.hpcadvisorycouncil.com/events/2013/ISC13-Student-Cluster-Competition/rules.php>. [Accessed: 30th July 2013].
- [14] Novel energy efficient compute architectures on the path to exascale, epcc dissertation 2012.  
[online]. Available From: <http://www.epcc.ed.ac.uk/sites/default/files/Dissertations/2011-2012/Submission-1146632.pdf>. [Accessed: 30th July 2013].
- [15] Tibidabo (prace prototype), system overview.  
[online]. Available From: <http://www.bsc.es/marenostrium-support-services/tibidabo-prototype>. [Accessed: 30th July 2013].
- [16] Low-power high performance computing, epcc dissertation 2012.  
[online]. Available From: <http://www.epcc.ed.ac.uk/sites/default/files/Dissertations/2011-2012/Micheal%20Holiday.pdf>. [Accessed: 30th July 2013].
- [17] Building and benchmarking a low power arm cluster, epcc dissertation 2012.  
[online]. Available From: <http://www.epcc.ed.ac.uk/sites/default/files/Dissertations/2011-2012/Submission-1126390.pdf>. [Accessed: 30th July 2013].
- [18] Low-power high performance computing, epcc dissertation 2011.  
[online]. Available From: <http://www.epcc.ed.ac.uk/sites/default/files/Dissertations/2010-2011/PanagiotisKritikakos.pdf>. [Accessed: 30th July 2013].
- [19] Wikipedia article, system on a chip.  
[online]. Available From: [http://en.wikipedia.org/wiki/System\\_on\\_a\\_chip](http://en.wikipedia.org/wiki/System_on_a_chip). [Accessed: 30th July 2013].
- [20] Arm company profile.  
[online]. Available From: <http://www.arm.com/about/company-profile/index.php>. [Accessed: 30th July 2013].
- [21] Wikipedia article, arm architecture.  
[online]. Available From: [https://en.wikipedia.org/wiki/ARM\\_architecture](https://en.wikipedia.org/wiki/ARM_architecture). [Accessed: 30th July 2013].
- [22] Rahul garg, exploring the floating point performance of modern arm processors.  
[online]. Available From: <http://www.anandtech.com/show/6971/exploring-the-floating-point-performance-of-modern-arm-processors>. [Accessed: 30th July 2013].
- [23] R.A. Haring, M. Ohmacht, T.W. Fox, M.K. Gschwind, D.L. Satterfield, K. Sugavanam, P.W. Coteus, P. Heidelberger, M.A. Blumrich, R.W. Wisniewski, A. Gara, G.L.-T. Chiu, P.A. Boyle, N.H. Chist, and Changhoan Kim. The ibm blue gene/q compute chip. *Micro, IEEE*, 32(2):48–60, 2012. [Accessed: 30th July 2013].

- [24] List statistics, top500.  
[online]. Available From: <http://www.top500.org/statistics/list>. [Accessed: 30th July 2013].
- [25] What gives arm processors a lower power footprint than intel counterparts?  
[online]. Available From: <http://www.quora.com/What-gives-ARM-processors-a-lower-power-footprint-than-Intel-counterparts#>. [Accessed: 30th July 2013].
- [26] Wikipedia article, instruction pipeline.  
[online]. Available From: [http://en.wikipedia.org/wiki/Instruction\\_pipeline](http://en.wikipedia.org/wiki/Instruction_pipeline). [Accessed: 30th July 2013].
- [27] Wikipedia article, intel core (microarchitecture).  
[online]. Available From: [http://en.wikipedia.org/wiki/Intel\\_Core\\_\(microarchitecture\)](http://en.wikipedia.org/wiki/Intel_Core_(microarchitecture)). [Accessed: 30th July 2013].
- [28] Nvidia's next generation cuda compute architecture: Kepler tm gk110.  
[online]. Available From: <http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf>. [Accessed: 31th July 2013].
- [29] Wikipedia article, nvidia tesla.  
[online]. Available From: [http://en.wikipedia.org/wiki/Nvidia\\_Tesla](http://en.wikipedia.org/wiki/Nvidia_Tesla). [Accessed: 31th July 2013].
- [30] Alan gray, epcc lecture notes, parallel programming languages.
- [31] Intel official site, expectations for user source code changes for intel mic.  
[online]. Available From: <http://software.intel.com/en-us/articles/expectations-for-user-source-code-changes>. [Accessed: 31th July 2013].
- [32] Hardware specification for the seco q7 carrier boards quadmo747-x/t30.  
[online]. Available From: [http://www.seco.com/en/item/quadmo747-x\\_t30/](http://www.seco.com/en/item/quadmo747-x_t30/). [Accessed: 1st August 2013].
- [33] Nikola puzovic, alex ramirez, pedraforca: a first arm + gpu cluster for hpc.  
[online]. Available From: <http://on-demand.gputechconf.com/gtc/2013/presentations/S3064-Pedraforca-ARM-GPU-Cluster-HPC.pdf>. [Accessed: 1st August 2013].
- [34] Nikola rajovic, gabriele carteni, arm-based systems at bsc.  
[online]. Available From: <https://www.hpc2n.umu.se/sites/default/files/PSS2013-Presentation.pdf>. [Accessed: 1st August 2013].
- [35] Bsc press release, bsc building first supercomputer to combine arm cpus, gpu accelerators and infiniband.  
[online]. Available From: <https://www.bsc.es/about-bsc/press/bsc-in-the-media/bsc-building-first-supercomputer-combine-arm-cpus-gpu-accelerators>. [Accessed: 1st August 2013].
- [36] Mali-t604 overivew.  
[online]. Available From: <http://www.arm.com/products/multimedia/mali-graphics-plus-gpu-compute/mali-t604.php>. [Accessed: 1st August 2013].

- [37] Vector floating-point (vfp), arm1176jzf-s technical reference manual.  
[online]. Available From: <http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0301h/Cegdejhh.html>. [Accessed: 1st August 2013].
- [38] Wikipedia article, pandaboard.  
[online]. Available From: <http://en.wikipedia.org/wiki/PandaBoard>. [Accessed: 2nd August 2013].
- [39] Pandaboard es technical specs.  
[online]. Available From: <http://pandaboard.org/content/platform>. [Accessed: 2nd August 2013].
- [40] Wikipedia article, mobile ddr.  
[online]. Available From: [http://en.wikipedia.org/wiki/Mobile\\_DDR](http://en.wikipedia.org/wiki/Mobile_DDR). [Accessed: 2nd August 2013].
- [41] Pandaboard architecture and block diagram.  
[online]. Available From: <http://pandaboard.org/content/resources/references>. [Accessed: 3rd August 2013].
- [42] Viridis datasheet, boston viridis project.  
[online]. Available From: <http://download.boston.co.uk/downloads/9/3/2/932c4ecb-692a-47a9-937d-a94bd0f3df1b/viridis.pdf>. [Accessed: 3rd August 2013].
- [43] Wikipedia article, intel xeon family of processors.  
[online]. Available From: <http://en.wikipedia.org/wiki/Xeon>. [Accessed: 3rd August 2013].
- [44] Intel official site, intel® xeon® processor e5-2670.  
[online]. Available From: <http://ark.intel.com/products/64595>. [Accessed: 3rd August 2013].
- [45] Xeon e5-2600 series processor architecture.  
[online]. Available From: [http://www.theregister.co.uk/2012/03/06/intel\\_xeon\\_2600\\_server\\_chip\\_launch](http://www.theregister.co.uk/2012/03/06/intel_xeon_2600_server_chip_launch). [Accessed: 3rd August 2013].
- [46] Wikipedia article, advanced vector extension.  
[online]. Available From: [http://en.wikipedia.org/wiki/Advanced\\_Vector\\_Extensions](http://en.wikipedia.org/wiki/Advanced_Vector_Extensions). [Accessed: 4th August 2013].
- [47] Wikipedia article, thermal design power.  
[online]. Available From: [http://en.wikipedia.org/wiki/Thermal\\_design\\_power](http://en.wikipedia.org/wiki/Thermal_design_power). [Accessed: 4th August 2013].
- [48] Wikipedia article, infiniband.  
[online]. Available From: <http://en.wikipedia.org/wiki/InfiniBand>. [Accessed: 4th August 2013].
- [49] Wikipedia article, server provisioning.  
[online]. Available From: [http://en.wikipedia.org/wiki/Provisioning#Server\\_provisioning](http://en.wikipedia.org/wiki/Provisioning#Server_provisioning). [Accessed: 4th August 2013].

- [50] Operating system recommendations, pandaboard wiki, software resources.  
[online]. Available From: <http://pandaboard.org/node/225/#ubuntu>. [Accessed: 4th August 2013].
- [51] Michael larabel, linaro 12.12 enhances arm linux performance.  
[online]. Available From: [http://www.phoronix.com/scan.php?page=article&item=linaro\\_1212\\_linux&num=1](http://www.phoronix.com/scan.php?page=article&item=linaro_1212_linux&num=1). [Accessed: 5th August 2013].
- [52] Linaro available versions.  
[online]. Available From: <http://www.linaro.org/downloads/>. [Accessed: 5th August 2013].
- [53] Environmental manual page.  
[online]. Available From: <http://modules.sourceforge.net>. [Accessed: 5th August 2013].
- [54] Mpvapich2 forum mail thread, mvapich2 invalid communicator errors.  
[online]. Available From: <http://mail.cse.ohio-state.edu/pipermail/mvapich-discuss/2007-September/001112.html>. [Accessed: 5th August 2013].
- [55] Wikipedia article, library (computing).  
[online]. Available From: [http://en.wikipedia.org/wiki/Library\\_\(computing\)](http://en.wikipedia.org/wiki/Library_(computing)). [Accessed: 5th August 2013].
- [56] Wikipedia article, automatically tuned linear algebra software.  
[online]. Available From: [http://en.wikipedia.org/wiki/Automatically\\_Tuned\\_Linear\\_Algebra\\_Software](http://en.wikipedia.org/wiki/Automatically_Tuned_Linear_Algebra_Software). [Accessed: 5th August 2013].
- [57] Vesprix wiki, atlas 3.8.4 arm.  
[online]. Available From: <http://www.vesprix.com/arm/atlas-arm/index.html>. [Accessed: 5th August 2013].
- [58] Ubuntu documentation, autofs.  
[online]. Available From: <https://help.ubuntu.com/community/Autofs>. [Accessed: 6th August 2013].
- [59] Torque documentation, chapter 5: Integrating schedulers for torque.  
[online]. Available From: <http://docs.adaptivecomputing.com/torque/4-1-3/help.htm#topics/5-scheduler/integratingSchedulers.htm>. [Accessed: 6th August 2013].
- [60] Mpich2 developer's mail thread, mpiexec to call launcher for all names in machines file without hostname resolution (was -nolocal option).  
[online]. Available From: <http://lists.mcs.anl.gov/pipermail/mpich2-dev/2011-July/000870.html>. [Accessed: 6th August 2013].
- [61] Wikipedia article, centos.  
[online]. Available From: <http://en.wikipedia.org/wiki/CentOS>. [Accessed: 6th August 2013].
- [62] Kickstarter documentation, 31.1 what are kickstart installations?  
[online]. Available From: [https://access.redhat.com/site/documentation/en-US/Red\\_Hat\\_Enterprise\\_Linux/5/html/Installation\\_Guide/ch-kickstart2.html#s1-kickstart2-what-is](https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Installation_Guide/ch-kickstart2.html#s1-kickstart2-what-is). [Accessed: 6th August 2013].

- [63] Intel® parallel studio xe 2013.  
[online]. Available From: [http://software.intel.com/sites/billboard/sites/default/files/downloads/Intel\\_Parallel\\_Studio\\_XE\\_2013\\_PB.PDF](http://software.intel.com/sites/billboard/sites/default/files/downloads/Intel_Parallel_Studio_XE_2013_PB.PDF). [Accessed: 6th August 2013].
- [64] Wikipedia article, hpcc benchmark suite.  
[online]. Available From: [http://en.wikipedia.org/wiki/HPC\\_Challenge\\_Benchmark](http://en.wikipedia.org/wiki/HPC_Challenge_Benchmark). [Accessed: 7th August 2013].
- [65] Run rules for the green500, power measurement of supercomputers.  
[online]. Available From: [http://www.green500.org/docs/pubs/RunRules\\_Ver0.9.pdf](http://www.green500.org/docs/pubs/RunRules_Ver0.9.pdf). [Accessed: 7th August 2013].
- [66] Wikipedia article, coremark benchmark.  
[online]. Available From: <http://en.wikipedia.org/wiki/Coremark>. [Accessed: 7th August 2013].
- [67] Hpc challenge benchmark.  
[online]. Available From: <http://icl.cs.utk.edu/hpcc>. [Accessed: 7th August 2013].
- [68] Jack j. dongarra et al., the linpack benchmark: Past, present, and future.  
[online]. Available From: <http://www.netlib.org/utk/people/JackDongarra/PAPERS/hplpaper.pdf>. [Accessed: 8th August 2013].
- [69] David henty, epcc lecture notes, advanced parallel programming lecture 1.
- [70] Milc qcd code benchmarks.  
[online]. Available From: <http://physics.indiana.edu/~sg/milc/benchmark.html>. [Accessed: 9th August 2013].
- [71] Gnu manual page, 3.17.3 arm options.  
[online]. Available From: <http://gcc.gnu.org/onlinedocs/gcc/ARM-Options.html>. [Accessed: 9th August 2013].
- [72] Intel compiler vectorization.  
[online]. Available From: [https://computing.llnl.gov/?set=code&page=intel\\_vector](https://computing.llnl.gov/?set=code&page=intel_vector). [Accessed: 10th August 2013].
- [73] Intel official site, intel mkl benchmark results.  
[online]. Available From: <http://software.intel.com/en-us/intel-mkl>. [Accessed: 10th August 2013].
- [74] S. Herbert and D. Marculescu. Analysis of dynamic voltage/frequency scaling in chip-multiprocessors. In *Low Power Electronics and Design (ISLPED), 2007 ACM/IEEE International Symposium on*, pages 38–43, 2007. [Accessed: 31th July 2013].
- [75] David h. baily, nas parallel benchmark results.  
[online]. Available From: <http://www.davidhbailey.com/dhbpapers/npb-ipdt.pdf>. [Accessed: 8th August 2013].
- [76] Greg bauer et al., performance modeling and comparative analysis of the milc lattice qcd application su3 rmd.  
[online]. Available From: [http://htor.inf.ethz.ch/publications/img/ccgrid\\_performance\\_modeling\\_milc.pdf](http://htor.inf.ethz.ch/publications/img/ccgrid_performance_modeling_milc.pdf). [Accessed: 8th August 2013].

- [77] K stuben, algebraic multigrid amg: An introduction with applications.  
[online]. Available From: [http://www.scai.fraunhofer.de/fileadmin/download/samg/paper/AMG\\_Introduction.pdf](http://www.scai.fraunhofer.de/fileadmin/download/samg/paper/AMG_Introduction.pdf). [Accessed: 8th August 2013].
- [78] Us lattice quantum chromodynamics.  
[online]. Available From: <http://usqcd.jlab.org/usqcd-software>. [Accessed: 11th August 2013].
- [79] Nvidia developer zone, cublas library.  
[online]. Available From: <https://developer.nvidia.com/cublas>. [Accessed: 12th August 2013].
- [80] Milc home page, milc code version 7.  
[online]. Available From: [http://www.physics.utah.edu/~detar/milc/milc\\_qcd.html](http://www.physics.utah.edu/~detar/milc/milc_qcd.html). [Accessed: 12th August 2013].
- [81] Guochun Shi, S. Gottlieb, A. Torok, and V. Kindratenko. Design of milc lattice qcd application for gpu clusters. In *Parallel Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 363–371, 2011. [Accessed: 12th August 2013].
- [82] Email communication within the cluster challenge team and authors of the milc ( carlton detar et al.).
- [83] Isc'13 klusterkamph official results! record number of lins packed in leipzig.  
[online]. Available From: <http://studentclustercomp.com/?p=1526>, note = [Accessed: 12th August 2013].
- [84] Project homepage, easybuild-framework.  
[online]. Available From: <http://hpcugent.github.io/easybuild>. [Accessed: 11th August 2013].