Viability of Low-Power Architectures as Parallel File Systems

Amanda B. Braga[1], Natália G. Rampon[1], Vinícius R. Machado[1], Jean L. Bez[1],
Francieli Z. Boito[2], Rodrigo V. Kassick[1], Edson L. Padoin[3], Philippe Navaux[1]

[1]Federal University of Rio Grande do Sul – Porto Alegre, Brazil
{amanda.binotto, natalia.rampon, vrmachado, jean.bez, rvkassick, navaux}@inf.ufrgs.br
[2]Federal University of Santa Catarina – Florianópolis, Brazil
francieli.boito@posgrad.ufsc.br
[3]Regional University of Northwest of Rio Grande do Sul (UNIJUI) – Ijuí, Brazil
elpadoin@inf.ufrgs.br

Abstract

This paper presents an I/O performance and energy efficiency analysis of low-power processors when compared to conventional architectures. This study aims at evaluating the viability of using such low-power architectures as servers to file systems in HPC environments. Results have shown that the low-power architecture could be an alternative to applications that perform many more read operations than write operations. The study also has shown that this architecture could lead to 93% of energy consumption decrease in read operations.

1. Introduction

The increase of processing power in architectures of High-Performance Computing (HPC) came along with the increase of a significant power demand on these systems. Great energetic demands like this are not the best scenario ecologically and economically speaking. In this regard, a DARPA report suggests that future HPC systems - from which is expected exaflops performance - should obey a 20MW limit on power demand [1].

Based on this premise, researchers have been seeking alternatives to respect such limits. A commonly used strategy is the use of low power architectures, changing regular processors to Advanced RISC Machines processors (ARM). Despite presenting less performance, these architectures give better power efficiency to some scientific applications [2].

However, processing is not the only one responsible for the big energetic consumption on High-Performance Computing Systems. Input and Output operations - known as I/O operations - also contributes to the high power demand, due to the crescent gap between processing and storage latencies. Therefore, it is common that applications spend a significant time on I/O operations and, for this reason, methods on how to decrease this power demand should be studied.

Considering the HPC scenario, I/O operations are processed by the parallel file system (PFS) and machines operate as servers for the data. These servers receive requests from the processing nodes and process them accessing the local storage system. Therefore, the processing capability of these machines is not well explored due to the elevated time spent with I/O operations. As an alternative, it is possible to consider that low demanding power architectures when used as storage servers could be an alternative that might lead to better energy efficiency.

With the possibility to use a low power architecture as a storage server in mind, the objective of this paper is to do a comparison between the mentioned architecture with a traditional computer, both working as parallel file system data servers. The rest of this paper is organized as follows. The next section discusses the experimental methodology involved and results are shown in Section 3. Lastly, the conclusion of this paper and future work are presented in Section 4.

2. Experimental Methodology

To achieve the objective of this research, two machines were used. The low power computer chosen was an A20 dual-core ARM Cortex-A7 processor produced by AllWinner running at 1GHz frequency. Besides, the machine has 2GB RAM memory and 8GB NAND storage. The storage device used was an SSD 840 Series MZ-7TD500BW by Samsung with a capacity of 500GB and a SATA 6Gb/s bus.

It was not possible to run a real file system on the ARM processor. The impossibility was due to the fact that the processor only accepts a modified Linux kernel as an operational system that does not support the file system. Attempts were made to install the module that supports it, but they were not successful. Instead, an MPI code was used to simulate the file servers' activity. The servers' simulation consists of them receiving requests that are staggered and there is no communication between the servers.

Requests are provided to the servers through trace files. Each one of them is organized as follows: there is one request per line and each line contains the application's timestamp, the request's size and the file's offset. The trace file approach was used because it was of this research's interests to isolate the network so a fair comparison could be made. By using the mentioned approach, it is possible to exclude the time needed to a client to make a request.

In addition, it would also have been unfair to compare the network used in our low power cluster with a regular cluster's one, because it is far slower and would harm performance. With that in mind, we decided to take the network off our experiments, because we wanted to focus on the processing capabilities of the low power architectures. Without the network in place, we were able to use a single desktop in the comparison, which allowed the usage of the same SSD memory in both architectures, simulating, the fact that in a conventional cluster we would not be able to change storage devices. By using the same storage device, a more fair comparison was made, taking into account particularities in our storage device and its characteristics.

The measurement of power was made with a P4460 Kill A Watt EZ power meter, which has an accuracy of 0.5% and a refresh rate of 1 second. However, since the equipment used does not have any means of communication with external devices, it was impossible to gather data directly from it. We then devised a method for data gathering, which consisted of filming the data being shown in the display of the power meter and then synchronizing it with the timestamps of beginning and end of each test. Thus, we were able to manually get the data and analyze it.

Trace files were created to emulate a single client contiguously accessing a 6GB file from the parallel file system server. This access is separated in small (32KB) or large (4MB) requests.

The size of all executed tests was 6GB. Tests were made considering the requests' size and the type of operation; they were executed in both worked architectures. In order to increase the reliability of the tests, each one of them was repeated five times. Afterward, the arithmetical mean of the five tests was made and the obtained value was used to analyze the data. Therefore, there were made forty experiments and there were eight different configurations of tests.

3. Results

In this section, the results obtained will be presented. Firstly, we will analyze how the size of requests and operation influence the power demand. Afterward, in Section 3.2 the tests' runtimes in both architectures will be compared.

3.1. Power demand analysis

In order to investigate the influence of requests on power demand, the average power will be used. This value is obtained by the arithmetic mean of instant power measurements.

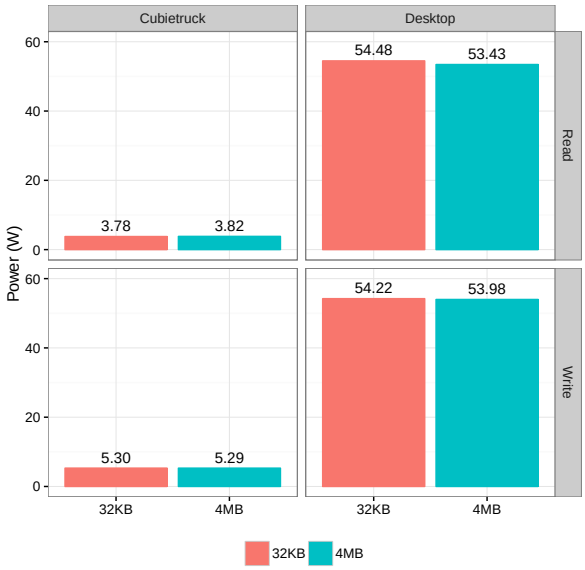

Figure 1. Power demand

Figure 1 shows the power demand on each architecture, separated by size and type of operation (write or read). The results for each architecture are in different columns and the rows discriminate between the operation performed.

From this data, it is clear that the requests' size does not have any impact whatsoever on the demand for power, neither in the Cubietruck nor in the Desktop configuration. However, in the Cubietruck device the type of operation performed alters the demand: write operations consume 40.2% more power than read operations in this architecture. The Cubietruck alternative also decreases power demand in 90.2% in comparison to the Desktop architecture in write operations and in 92.8% in readings.

3.2. Runtime analysis

Another important measure of this research was the runtime analysis of the conducted tests. The measures were made by running the same tests on the low power and the regular one architectures.

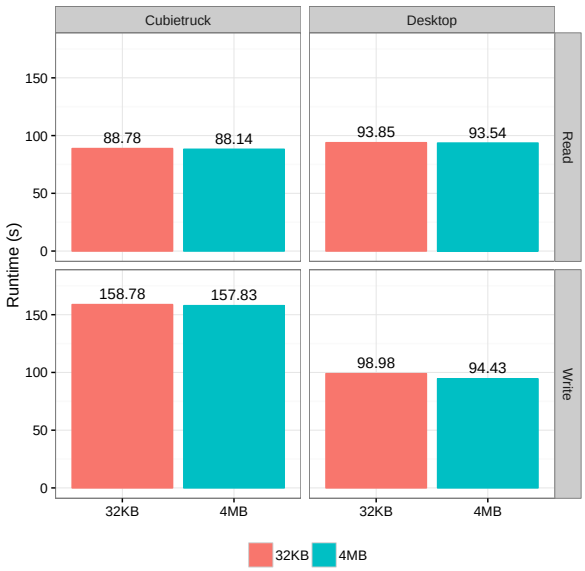

Figure 2. Execution time

Figure 2 shows the runtime analysis on each architecture, separated by size and type of operation (write or read).

By the results' analysis, it is possible to see, in the first place, that the requests' size does not affect the tests' runtime. Also, it is clear to note that on read operations the ARM processor and the regular processor present very similar performance. Yet, we can see that write operations on the low power architecture present a worse performance than on the regular architecture. With 32KB request size, by instance, the Cubietruck had a 68,8% increase of time by comparing with the Desktop. Therefore, the ARM processor could be an alternative to regular processors when dealing with an application that performs a lot more read operations than write ones.

3.3. Energy consumption analysis

The energy consumed in Joules by the experiments is obtained multiplying the median power by the execution time. Moreover, besides the impact caused by the type of operation because of the change in runtime, it is also expected to be found differences in energy consumption by the tests between the different architectures. This happens because the devices have different power demands.

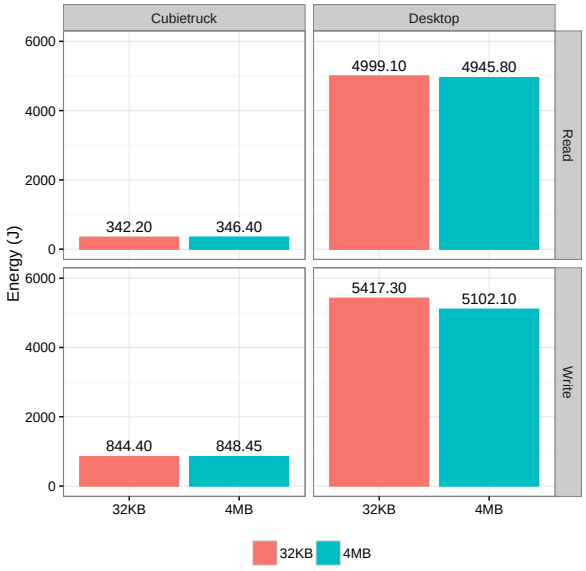

Figure 3. Energy consumption

Figure 3 shows the energy analysis on each architecture, separated by size and type of operation (write or read). It is again seen that the requests size have little to no influence in the results. By the graphics, it's clearly observed that the Cubietruck has great energy efficiency. The low-power architecture consumes 93% less energy in read operations and 83.4% in write ones in comparison to the usual one.

Considering the fact that both architectures have similar runtimes in read operations, the ARM processor is a feasible low-power alternative to regular processors in this aspect. Even when taking into account the 68.8% increase in execution time seen during write operations in Section 3.2, the alternative still stands as a possibility, since the energy consumption decline in this scenario is of 83.4%. Hence, the Cubietruck device would be a good substitute for regular architectures in applications that use more read operations than write ones.

4. Conclusions and Future Work

This study showed that low power processors could be an alternative to applications that perform many more read operations than write ones. The substitution would lead to an energy economy of up to 83.4% in write operations and 93% in read ones.

As future work, we plan to emulate real scientific applications and include more workloads in the study, such as non-contiguous accesses to files, more clients accessing concurrently the file system and different access patterns. Moreover, we will include the network in the future analysis.

Acknowledgements

The authors of this paper would like to thank Rodrigo Kassick from the Federal University of Rio Grande do Sul for the code used to simulate the file system activities.

References

[1] P. Kogge, K. Bergman, S. Borkar, D. Campbell, W. Carlson, M. Denneau, P. Franzon, W. Harrod, K. Hill, and et. al. Exascale computing study: Technology challenges in achieving exascale systems. pages 1–297, 2008.

[2] E. L. Padoin, F. Z. Boito, L. L. Pilla, M. B. Castro, P. O. A. Navaux, and J.-F. Mehaut. Performance/energy trade-off in scientific computing: The case of arm big.little and intel sandy bridge. *IET Computers & Digital Techniques*, pages 1–14, 2014.