# HPC Power Management on Haswell CPU Architecture

Ariel Young, Ioan Raicu
Illinois Institute of Technology
Computer Science Department
ayoung11@hawk.iit.edu, iraicu@cs.iit.edu

## ABSTRACT

**As science continues to generate large sets of data, society expects nothing less than a machine capable of performing intensive calculations in a matter of microseconds. The technology necessary to fulfill these demands consume a great amount of power, leading to a greater ecological footprint. The primary objective is to investigate altering different components of the machine in ways that would, in effect, decrease the power consumption with minimal to zero impact on application performance, starting with the processor. In order to do so, we placed the cores in different p-states and c-states to reduce the power consumption while running a variety of applications.**

## General Terms

Management, Measurement, Performance, Experimentation.

## Keywords

Power Management, Processor States.

## 1. INTRODUCTION

Today, computer parts have not only become a burden on our wallets but also have greatly increased the amount of power needed to operate at consumer expected speeds. A primary motivator of this work is the Student Cluster Competition at the IEEE/ACM Supercomputing/SC 2015 conference [5], which focuses on obtaining the best performance possible from within a specific power budget.

A study showed that the main consumer of power within a machine was the memory and processor components [1]. Manufacturers such as Intel, have begun implementing technology to reduce unnecessary power consumption. With Intel's latest release, they have managed to reduce the idle states' power consumption overall reducing the required energy to run the processor by 40% [4]. When entering into an idle state, there is an expected latency (which increases as you enter higher idle states) in order to resume activity. However, the Haswell processor now experiences shorter latencies than previous generations, reducing the opportunity cost of entering these deep sleep states [4].

Two ways to adjust the core's power usage is through utilization of its P-states and C-states. P-states are states of activity that have varying voltage and frequency pairings of the processor, the higher the P-state, the lower the frequency that it runs at. A core is always in one of the C-states, idle or active. C-states provide different levels of idleness, as cores advance to a higher C-State, the amount of energy saved increases but there also is a higher latency to resume activity. If a core is active, it is in C0, as it becomes idle it will enter C1 and if it is needed for a task it will either demote back to C0 or enter into another C-state. [Cite a source here]

According to Intel, power has the following relationship between frequency, voltage and the capacitance of the chip:

$$P = C * V^2 * F + Leakage \qquad [2]$$

We are unable to control the power leaked by the processor, but we are able to change the frequency and voltages. By increasing the P-states we lower the frequency and voltage that the processor runs at which in turn lowers the power consumption of the processor overall.

Taking advantage of individual core control and the operational states, we can optimize the power usage through advancing certain cores to deeper sleep states or higher P-states depending on whether or not CPU usage is heavy.

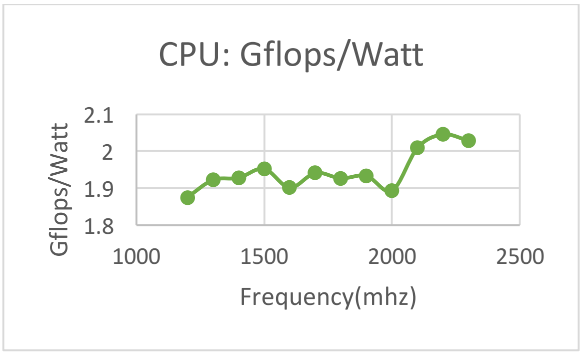
## 2. DESIGN AND IMPLEMENTATION

The primary motivation for this research is for the Student Cluster Competition which focuses on limiting power usage while running a series of applications that resemble everyday high performance computing. The applications rely heavily on a variety of resources, for our preliminary research we chose to study High Performance LINPACK which is very CPU intensive so were able to see the effect of the various P-states on the application's performance. Since LINPACK spends the majority of its time computing, changing the C-states is meaningless as the application would not have finished running.

## 3. PERFORMANCE EVALUATION

The preliminary data gathered in this report was conducted on a node within Joint Laboratory for System Evaluation (JLSE) cluster at Argonne National Laboratory.

Figure 1: The effect of deeper P-states on CPU intensive application LINPACK.



As expected, the application's performance was impeded by the lower frequencies that the processor ran at. The relationship between the ratio of Gflops to Watts and frequency is not clear, but overall they are directly related which justifies the use of less nodes with running at a higher CPU frequency.

Figure 2: Varying core usage on WRF

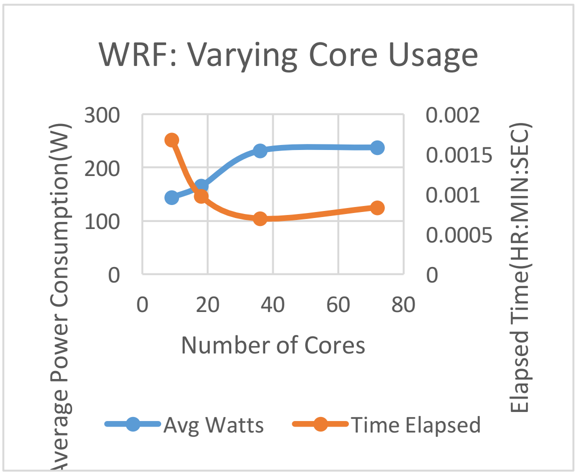
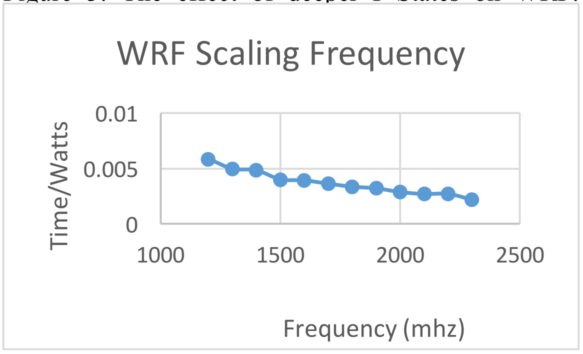


Figure 3: The effect of deeper P-States on WRF.



Although WRF is not as CPU-intensive as Linpack, changing the frequency that it is run at does affect its performance as the time it takes to run increases while the frequency decreases. Allocating less/more than the amount of cores of the physical machine for the WRF application shows an increase in the application's run time. Since the CPU does not draw as much power as when LINPACK is run, changing the C-states of the cores while allocating less cores for WRF would greatly impact performance.

In addition to studying the varying frequencies of the CPU on a few applications, we have also created a script for automatic power control. An upper and lower bound limit were set to create a gray zone to signal when to stop increasing the frequency (the lower bound) as to prevent spikes in power usage and also to signal when to turn down the frequency. Studying WRF and LINPACK shows that we cannot treat every application the same as the applications rely heavily on other components of the machine. For LINPACK, we were more specific as to how much to change the frequency based on how low/high the power was from the line. For WRF, leaving the frequency steps at 100 mhz, no matter the distance from the safety/power limit, seemed to work perfectly fine.

Figure 4: Linpack's power track while running the automated script

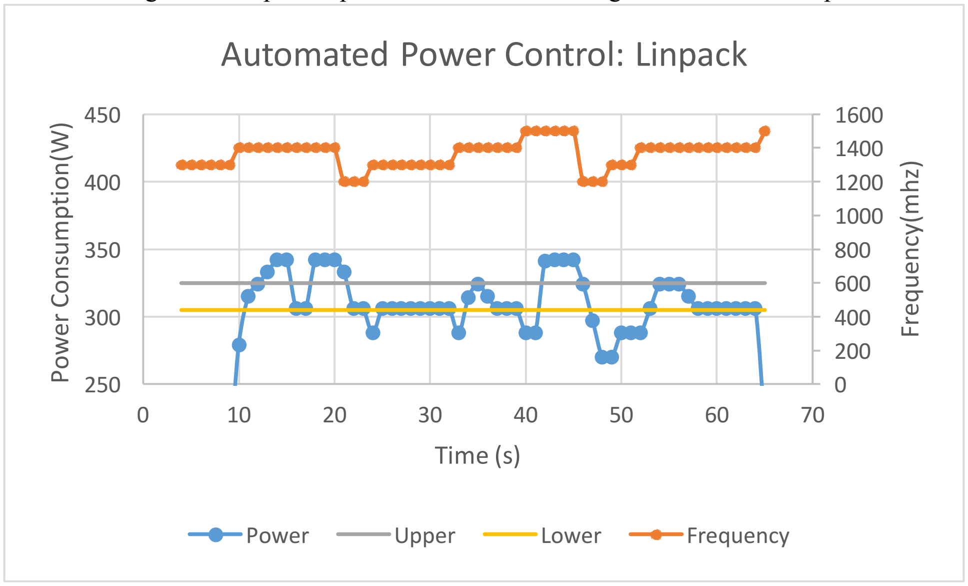
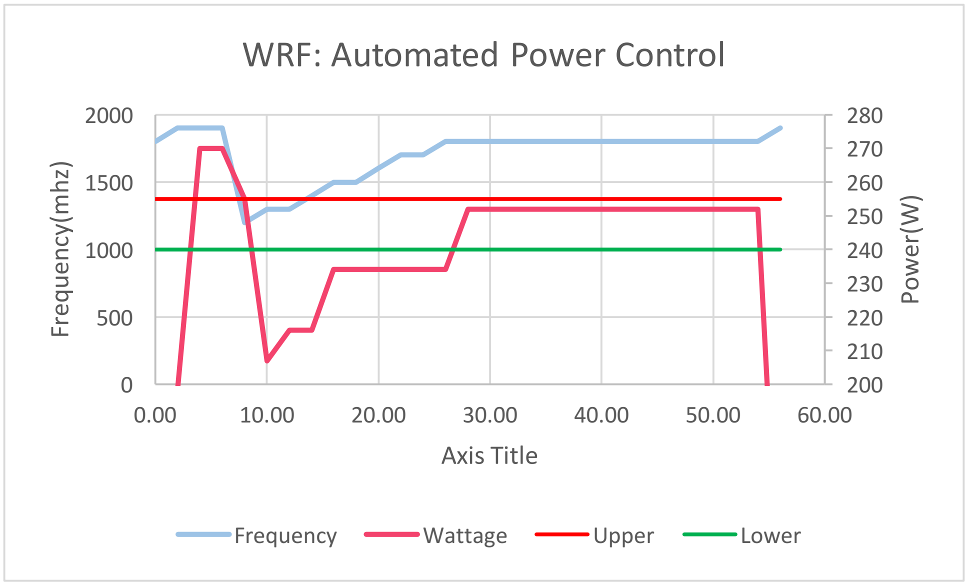


Figure 5: WRF's power track while running the automated script.



In the preliminary runs, there was always a spike above the power limit set which comes from the processor running at full potential (2300mhz) in the beginnings. In hopes of preventing this, we adjusted our power management script and set the processor to a low frequency which still resulted in a dramatic spike in the power consumption. While it is somewhat upsetting, we will accept tiny slips in the power control as these slips last for a short amount of seconds.

## CONCLUSION AND FUTURE WORK

By altering the different operation states of the processor we were able to see the affect it had on the application performance as well as any significant impact on the power consumption.

Although we were unable to investigate changing the C-states of the cores, a future experiment would be finding an application that does not favor more cores to run and forcing idle cores into deeper sleep states.

Additionally, studying the applications further to understand the sudden spikes in power consumption would help in molding the generic power script to hopefully get the best out of the machines resources without going over the power budget.

As the CPU is only one component of a machine, the next steps would be to observe the power saving capabilities of other parts, such as memory and disk. The parts of a machine often have safety limits quoted by manufacturers to insure a stable machine. We look forward to testing the limits quoted by manufacturers to see if there are ways to reduce the power consumption of different components and allocate it elsewhere.

## 4. ACKNOWLEDGEMENTS

I would like to thank my mentor, Ioan Raicu, for guiding me through this research and providing constructive criticism towards improving my methods. I would also like to thank the NSF Career award for funding this research opportunity for me.

## 5. REFERENCES

[1] Economou, D., Rivoire, S., Kozyrakis, C., Ranganathan, P. 2006. Full-system power analysis and modeling for server environments. In Proceedings of the 2nd Workshop on Modeling, Benchmarking, and Simulation (MoBS), held at the International Sym- posium on Computer Architecture (Boston, MA, June 2006), pp. 70{77).

[2] Intel Power Management Technology. Intel® Power Management Technologies for Processor Graphics, Display, and Memory: White Paper. For 2010 - 2011 Desktop and Notebook Platforms, August 2010.

[3] Intel Power Management Technology. Intel® Xeon® Processor E7-8800 and 4800 v3 Datasheet, Vol. 1.: White Paper. 2015.

[4] Jain, Tarush, and Tanmay Agrawal. "The Haswell microarchitecture–4th generation processor. "*International Journal of Computer Science and Information Technologies* 4.3 (2013): 477-480.

[5] Kevin Brandstatter, Ben Walters, Alexander Ballmer, Adnan Haider, Andrei Dumitru, Serapheim Dimitropoulos, Ariel Young, William Scullin, Ben Allen, Ioan Raicu. "Experiences in Optimizing Cluster Performance For Scientific Applications: Controlling Configuration, Utilization, and Power Consumption", GCASR 2015