Precise ECG Platform on Modern Processors

S. Normalized Infloop
Department of Calculator Science
Theoretical Abstract Interpretation Testing Society
Pitsston, PA, United State Monads of America
yueyao@andrew.cmu.edu

Ivybridge N. Skylake
Institute of Nondeterministic Interpretation
Theoretical Abstract Interpretation Testing Society
Bosburgh, PA, United State Monads of America
yuningzh@andrew.cmu.edu

***Abstract*—The authors shivers in the howling wind on Pausch Bridge, wondering why the processors in their backpacks gets to sit comfortably and doing nothing instead of switching their tiny transistors to keep their owners, who paid BIG money to buy them, warm.**

***Index Terms*—Thermal Systems, Ambient Heat Modulating Technologies, Parallel Heating, 2D Computer Graphics, Edible Content Generation**

I. Introduction

Of all the things computing machines brought to the world, there is one thing that people have consistently tried hard to get rid of since the dawn of this field. Dissipating heat in computing system has become a major design problem in all levels of computer engineering. Modern processors is on the verge of, if not already, hitting on one of the major design boundaries known as the power wall (the chip's overall temperature and power consumption) [1]. Some people believe that "the power wall is now arguably the defining limit of the power of the modern CPU" [2].

If we are unable to curse of this everyday intensifying brownian motion within the computing machines, why not take advantage of it? Heat, as a resource, can be very useful in a number of ways. In fact, heat is a very important resource widely used in *Edible Content Generation (ECG)* process. Edible Content Generation takes (usually) biological specimens and apply a sequence of chemical and physical decorations to them, generating human digestible contents. This process is more commonly known as *cooking*.

It has been over a decade since people has tried to utilize the heat generated from computing devices to facilitate ECG. However, a vast majority of those attempts has failed and results in either overheated computing devices or unsuccessful ECG process. The authors believe the problem is that operators of such process have very little control on the power of the heating device, compared to tradition ECG platforms (more commonly known as stoves). This work addresses this problem by proposing a way to turn modern processors into precise ECG platforms, which users have precise control of power down to every second.

II. Computing Device Based ECG Platforms

A. Graphics Device Based ECG Platforms

Graphics card users has a long history of trying to use their graphics devices as an ECG platform. In [3], the author tried to use an Geforce GTX 480 graphics card to cook an egg. The ECG software used to control the ECG process are games and benchmarking software of unknown name. The resulting edible content is shown in Fig. 1.

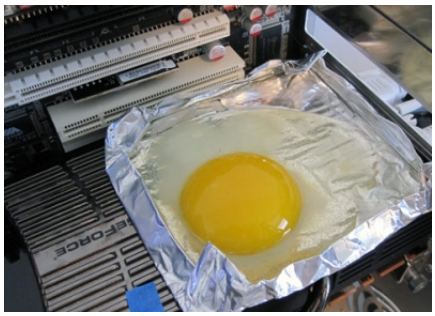

Fig. 1. Egg fried on an GTX480 graphic card.

The authors reports that the resulting edible content is, in fact, too close to its primitive form to be edible (an effect more commonly known as being "too raw"). The researchers blame the ECG controlling software for the failure of the experiment and suggests *FurMark* would give a better result. This research further demonstrates the urgent need to design a precise and easy to use ECG platform.

The authors also believe MVidia is in fact secretly encouraging users to use their product as an ECG platform to attract more customers. We acquired two pieces of advertising material (Fig. 2 and Fig. 3) from an unnamed source familiar with the company media promotion strategy. Around a decade a ago, GPGPU computation started off as an hack into graphics rendering pipeline and now its' one of the primary use of GPUs in the inductry. We believe history will repeat itself in the case of graphics device based ECG. Our work, although focusing on CPUs, generalizes well to GPUs (TODO: give this wild claim some justification!).

B. CPU Based ECG Platforms

Users has also tried use CPU as ECG platforms. C. C. Channel [4] demonstrated in a YourTube Video series the possibility of using Ontel processors to generate various kinds of edible contents. In their experiments, bacon, spaghetti, and even pop corns are processed through an Ontel processor. The software used to control the ECG platform is not known.

Fig. 2. Early promotion material used for MVidia GTX480.

Fig. 3. Promotion material for MVidia RTX2080 for Chinese market.

Some of the experiments do provide valid results. The authors are able to obtain a few edible content with great quality. However, in "Cooking with Ontel 5 - The Nearly Indestructible Celeroff D", the authors lost one of their experimentation platform due to overheating.

This highlights the need for a precise ECG platform. If the user sets the power too high for too long, the user might end up losing the hardware. On the other hand, if the ECG platform provides insufficient power, the edible content may be too close to its primitive form to be safely consumed by the user.

The fact that modern processors are multi-core and superscalar presents more challenge to building a precise ECG platform. The ECG platform will have to orchestrate work across different cores so that the overall power stays constant.

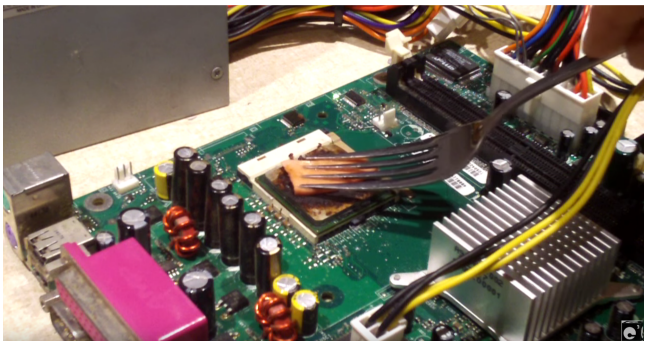

Fig. 4. Fork facilitated ECG with Ontel Celeroff D processors.

III. System Architecture

The goal of this work is to implement a precise ECG platform on a modern processor. Modern processors refers to processors with many cores and possibly SMT support. By saying precise we wish to grant users very fined grained control of ECG platform power output for every second. Controlling power consumption of processors roughly translates into controlling CPU utilization rate [5]. In conclusion, we need to design a system that gives user control on CPU utilization, down to every second, on modern multi-core processors.

A. Load generation by frequency modulation.

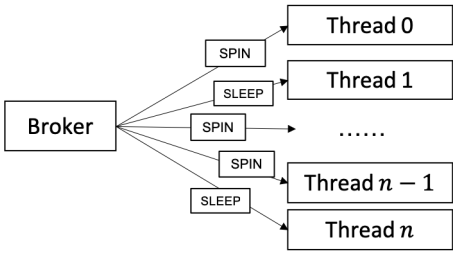

Fig. 5. Broker dispatches tiny chunk of work to threads.

Our system is built upon a standard broker-worker architecture. A globally shared broker will generate work for

each thread. Each thread independently obtains jobs from the broker, executes it, and starts over. There are two types of jobs:

- **SLEEP**. The thread will sleep for a small period of time.
- **SPIN**. The thread will spin in a tight loop for a small period of time.

Usually the size of both jobs is set to 1 ms. The broker randomly generates jobs based on a desired system load. For example, if the desired CPU utilization is 10%, the broker has a 1/10 probability of generating a `SPIN` work. It should be self evident that this strategy indeed achieves the required CPU utilization. The proof is left as an exercise for the reader. Essentially this scheme achieves a certain system load by modulating frequency of processor spinning. This scheme is thus termed (job) frequency modulation. The probability of generating a spin job is termed *spin rate*, denoted by s.

This scheme is better than the length modulation scheme, where we adjust the length of spinning jobs relative to sleep job. Frequency modulation provides a more stable CPU load over time (less variation in utilization). For every (large enough) time window, the average CPU utilization will be identical regardless where the window is. On the other hand, frequency modulation helps to achieve precise control of CPU utilization.

B. Load compensation by PI controller.

It is often the case where people need to work while using ECG platforms. This causes the problem if the user is working on the computing device while its being used as an ECG platform. The workload generated by the user will very likely affect overall CPU utilization and increase power output, which may overcook edible contents, or worse, destroys the device. It is essential for the ECG platform to be able to compensate for the load.

Due to the unpredictable nature of the user workload, it will be very hard to model (even reliably measure) user workload online. Here we took a feedback control system approach. We attach what is known as an *proportionate-integral controller* (PI controller) to the output. For those unfamiliar with the concept, we provide the following explanation.

For every time step, there exists a desired CPU utilization L_0. It also measures the current CPU utilization, which denoted as L. The error e as this time step is defined as $e \triangleq L - L_0$. Clearly e is a function of time t, The compensation factor s' is calculated as

$$s' \triangleq P\ e(t) + I \int_0^t e(t)\ dt$$

where P and I are two coefficients termed proportionate coefficient and integral coefficient. Finally, if the spin rate dictated by desired CPU utilization is s_0, then the actual spin rate used by the broker will be $s = s_0 + s'$.

Intuitively, the P-term compensates for sudden changes in user load, while the I-term compensates for long running user load. To ensure negative feedback, both P and I must be negative.

C. Intuitive load specification.

Our proposed ECG system features a very intuitive way for the user to specify the desired load. Since many users of ECG systems are not tech-savvy, it's very important to keep the interface simple. The users of our system may specify a "program" by supplying a textual file, whose contents mimics the desired shape of CPU utilization graph (except the time is the Y axis).

For example, the following config file specifies that the system should run a loop that utilization is 40%, 80%, 20%, 70% for the first, second, third, and fourth second of each iteration.

```
||||
||||||||
||
|||||||
```

This interface is intuitive and very easy to use. It is so simple that only one character is involved, and no formal specification or documentation is needed to understand this format. We name this form `YAMMY` as in ***Y**et **A**nother **M**arkup-lang? **M**y **G**od!* format. The authors are convinced that this format will be as popular in the field of ECG platform research as JSON in machine learning research.

IV. Results

We implemented our work and tested the work on one of GHC machines. Unfortunately the authors are denied physical access to GHC machines the moment the told the administration staff that want to perform ECG experiment on one of their computers. The authors have no idea why the administration staff holds such hostility to legit scientific research and edible contents. The best we have is running our program while monitoring CPU utilization.

Fig. 6. Experimenting a sine like ECG control function.

We specify an sine-like control function to test our implementation. The testing platform comes with an Ontel Xeon E5-1660 CPU, which contains 8 cores, each with 2-way SMT. As you can see, our results proves the effectiveness of our

approach. Our ECG program is able to provide precise control on the CPU.

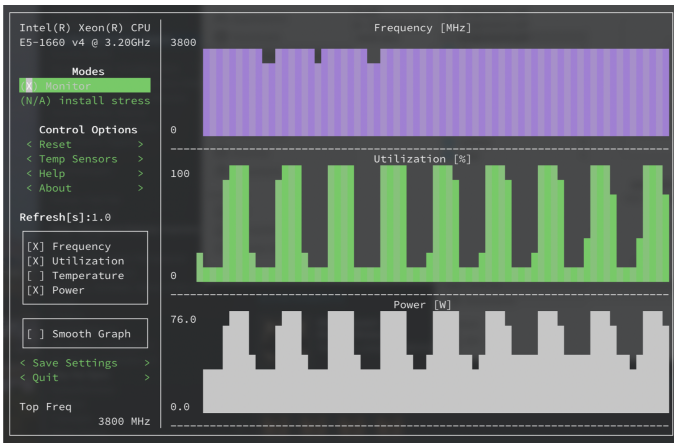

Fig. 7. Swift change in power output is also supported.

Fig. 7 tests the system's response time under quickly changing load specification. The result shows that the ECG program is able to precisely and quickly response to change in required power. The authors believe preparing edible content using our platform will be a pleasant and worry free process.

V. Conclusion

The code for this work is available on GitHub at `https://github.com/codeworm96/heat`. Since the authors are denied physical access to GHC machines, the authors believes there is no future of ECG platform research unless administration staff stop discriminating ECG platform researchers. Whats the point of doing ECG research when you cannot conduct proper ECG experiments? This field is DOOMED, dude! Get out! Learn you a Haskell for greater good [6].

References

[1] D. A. Patterson, "Future of computer architecture," in *Berkeley EECS Annual Research Symposium (BEARS), College of Engineering, UC Berkeley, US*, 2006.

[2] C. Mims. (2010) Why cpus aren't getting any faster. [Online]. Available: https://www.technologyreview.com/s/421186/why-cpus-arent-getting-any-faster/

[3] JEGX. (2010) Cook your eggs with a geforce gtx 480. [Online]. Available: https://www.geeks3d.com/20100331/cook-your-eggs-with-a-geforce-gtx-480/

[4] C. C. Channel. (2010) Cooking with intel. [Online]. Available: \url{https://www.youtube.com/results?search_query=Cooking+with+Intel}

[5] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *ACM SIGARCH computer architecture news*, vol. 35, no. 2. ACM, 2007, pp. 13–23.

[6] R. Harper, *Practical foundations for programming languages*. Cambridge University Press, 2016.