Building the Case for Temperature Awareness in Energy Consumption Models: an Application of the Energy-Frequency Convexity Rule

Kameswar Vaddina, Florian Brandner, Gérard Memmi
LTCI - TÉLÉCOM ParisTech - Paris, France
firstname.lastname@telecom-paristech.fr

Pierre Jouvelot
MINES ParisTech, PSL University, Paris, France
pierre.jouvelot@mines-paristech.fr

ABSTRACT

Optimizing computing and communication systems that host energy-critical applications is becoming a key issue for software developers. In previous work, we introduced and validated the Energy/Frequency Convexity Rule for CPU-bound benchmarks on recent ARM platforms. This rule states that there exists an optimal clock frequency that minimizes the CPU's energy consumption for non-performance-critical programs. We showed that the Energy/Frequency Convexity Rule is related to the non-linearity of power with respect to frequency and is not dependent on the supply voltage.

Here, we discuss the application of an analytical energy consumption model proposed previously to our target board, a TI AM572x EVM. We show that this non-linear analytical model can, for our experimental settings, be approximated by a frequency-linear variant, as our voltage is maintained constant. This, however, does not fit the measurements on the board, suggesting that a parameter is currently missing in the analytical model. We conjecture that accounting for temperature in the model would yield more accurate results that are in-line with our measurements. This builds the case for the inclusion of this important parameter in our energy models.

1 INTRODUCTION

Despite the advances in hardware technology, which have reduced the energy consumption of today's computing systems, designers are quickly learning that it is beneficial to consider a system-level view in order to minimize the overall energy consumption of the hardware/software platform instead of just striving for chips with very low power requirements. This *co-design* approach to system design involves making predictions of the energy consumption during the early stages of the design cycle – way before the final hardware and software versions are chosen/ready. Analytical tools can help system designers by providing energy consumption metrics up-front, which allows to streamline and optimize the energy budget during the initial phase of the project when decisions in the interest of energy consumption are likely to have less impact on the performance and cost criteria.

The application of this system-level approach in recent work lead to the discovery of the, so-called, Energy/Frequency Convexity Rule [4]. This rule states that the curve relating energy consumption to processing frequency exhibits convex behaviour, which implies the existence of an optimal frequency that minimizes energy consumption for compute-bound applications. This result has been validated on various platforms. Notably, the result was recently confirmed [10] for the TI AM572x evaluation module using a highly accurate on-line measurements that allow to perform precise energy monitoring and profiling of non-performance-critical applications.

In this paper, we build an analytical model that can be used by firmware and application developers to optimize their software for power consumption for a given target platform running at a particular technology node. We compare the predictions obtained from the analytical model with measurements obtained from on-line runs on the target platform and highlight some of its advantages and limitations. In particular with respect to temperature issues, this paper makes the following contributions.

- We provide an analytical energy consumption model for a typical central processing unit (CPU). The model consists of two separate submodels for power and execution time. The power model, proposed previously [3], is further decomposed into dynamic, short-circuit, and static components.
- We apply this analytical model to our target board, the TI AM572x evaluation module, considering controlled voltage settings. We confirm that the convexity in the experimental data is maintained, even with a fixed voltage.
- However, with a fixed supply voltage, the non-linear analytical model becomes linear w.r.t. the processing frequency. The model then no longer fits the measurements on the platform. Based on preliminary results, we conjecture that this is due to the fact that the model does not accurately consider temperature – which was masked in previous experiments by non-linear voltage terms.[1]

The remainder of this paper is structured as follows. In Section 2, we describe the analytical energy consumption model, using a small number of parameters. This theoretical framework, which was proposed earlier [3], forms the fundamental groundings for the Energy/Frequency Convexity Rule. In Section 3, we discuss how De Vogeleer et al. [3] derived this energy model and validated it on a test bed. Next, we describe the experimental setup for the TI AM572x EVM board used in this work, provide details on the data acquisition infrastructure, and explain how the host and the board under test are synchronized. The results obtained by running various benchmarks on our target board are discussed in Section 5, paving the way towards a refined model. Finally, we conclude in Section 6 and provide remarks on future work and research directions.

2 ENERGY CONSUMPTION MODEL

Earlier research endeavours put forward some intuition-based motivations for the convex behaviour of energy consumption by computer programs, but few introduce analytical frameworks based on proper physical principles. De Vogeleer et al. [4] have proposed

[1] These terms are derived indirectly from measurements through fitting, as direct measurements are impossible behind the sensitive voltage regulators.

a physics-based analytical model for energy consumption of processors. It was then validated on an actual test bed under realistic conditions. Part of this present work assesses the accuracy of this analytical model with energy data obtained from a new and much more accurate test bed.

The energy consumption of a typical CPU over an execution time period t is equal to the integral of its power over time. The relationship between the power (in Watt or Joule/s) and energy (Joule) consumed over the time period $[0, t]$ is as follows:

$$E_{cpu}(t) = \int_0^t P_{cpu}(t)dt = \int_0^t I(t)V(t)dt, \tag{1}$$

where $P_{cpu}(t)$, $I(t)$, $V(t)$ represents the instantaneous power, current, and operating voltage of the CPU.

For a discrete time interval Δt, the energy consumption can be deemed quasi constant, which, in practice, also means that the parameters that define energy consumption are also constant during this time interval. According to the Riemann sum approximation technique, the integral in Equation 1 can then be approximated by a finite sum, as denoted in the following equation, with $t = n\Delta t$:

$$E_{cpu}(n\Delta t) = \sum_{i=0}^{n-1} P_{cpu}(i\Delta t)\Delta t. \tag{2}$$

2.1 Power Model

When power requirements are reduced, end-user devices become longer-lasting and more reliable. Hence, it is important to know where the power is being dispatched and how to calculate it. An analytical model may help us to understand how various factors, like power constraints, capacitance, input voltage, switching activity, and temperature, affect the power characteristics of a device. Such a model can thus be used to calculate the power features of a device and determine the maximum reliable operating frequency, current requirements, power-supply sizing, cooling/heat-sink requirements, and, eventually, help in coming up with a criterion for selecting devices for future systems.

A typical CMOS digital circuit always requires power, independent of whether its logic states undergo transitions due to dynamic switching or remain unchanged. Its power is mainly composed of three components, namely 1) dynamic (switching) power, 2) short-circuit power, and 3) static (leakage) power. The total power of a CPU at a given time instant t (omitted in the formula) can then be denoted by the following equation:

$$P_{cpu} = P_{dynamic} + P_{short-circuit} + P_{static}, \tag{3}$$

2.1.1 Dynamic power. This is the power required for the charging and discharging of capacitances in the integrated circuit. It is represented by an effective switching capacitance, C_{sc}, and is proportional to the switching clock frequency (f_{clk}) and power supply voltage (V_{DD}). The dynamic power of a logic gate can be represented by the following relationship:

$$P_{dynamic} = \frac{1}{2}\, C_{sc}\, V_{DD}^2\, f_{clk}\, N_t, \tag{4}$$

where the switching capacitance C_{sc} is the sum of the power-related capacitance (C_{pd}), which is the equivalent circuit capacitance, and the external load capacitance (C_{load}). N_t denotes the total number of logic state transitions.

2.1.2 Short-circuit power. This is the power required by the CMOS circuit due to the short-circuit currents flowing from the voltage supply to the ground. This occurs only in static CMOS circuits when both the NMOS and PMOS field effect transistors for an inverter are conducting simultaneously. The short-circuit power is then given by the following equation [2]:

$$P_{short-circuit} = \frac{\beta}{12}\,(V_{DD} - 2V_T)^3\,\tau\, f_{clk}, \tag{5}$$

with β the MOS transistor gain factor, τ the input transition time and V_T the threshold voltage.

2.1.3 Static power. This is the power induced by the flow of leakage currents, which usually occur when the circuit is in steady-state with no switching activity. There are two sources of leakage currents in typical CMOS circuits: the diode- and sub-threshold leakage currents.

A diode-leakage current occurs when the leakage current pass through the reverse-biased diode junctions of the transistors, located between the source/drain and the substrate. The diode-leakage current is given by the Shockley diode equation, which is represented as follows [7]

$$I_{dl} = i_s(e^{qV_d/kT} - 1), \tag{6}$$

with i_s, the reverse bias saturation current, k, Boltzmann's constant, V_d, the diode voltage, T, the absolute temperature of the p-n junction, and q, the electrical charge of the electron.

Subthreshold leakage occurs when the leakage currents flow from the drain to the source when the gate-source voltage is smaller than the threshold voltage. The closer the threshold voltage is to 0V, the greater the leakage current [2]. The subthreshold leakage current is expressed by the following equation [1]:

$$I_{sl} = Ke^{(V_{GS}-V_{th})/(nV_T)}(1 - e^{-V_{DS}/V_T}), \tag{7}$$

with K, a parameter dependent of the technology node, V_{GS}, the gate-to-source voltage, V_{DS}, the drain-to-source voltage, V_{th}, the threshold voltage, $V_T = kT/q$, the thermal voltage, where T is the absolute temperature in Kelvin, $n = 1 + \Omega t_{ox}/D$, where t_{ox} is the thickness of gate oxide, D is the channel depletion width, and $\Omega = \epsilon_{si}/\epsilon_{ox}$, where ϵ_{si} and ϵ_{ox} are the permittivity of the silicon and gate oxide, respectively.

The resulting static power for both sources of leakage currents is expressed by the following equation:

$$P_{static} = I_{leakage}V_{DD}, \tag{8}$$

where $I_{leakage}$ is the sum of I_{dl} and I_{sl}. The total power of the CPU is then the sum of the dynamic, short-circuit, and static power – as expressed in Equation 3.

Based on the ITRS measurement traces, Skadron et al. [9] have established an interesting relationship between dynamic and leakage power. They established a relationship where leakage power has a dependence on the temperature T. The short-circuit power contribution is ignored in their model. It is represented as follows:

$$R_T = \frac{P_{leakage}}{P_{dynamic}} = \frac{R_0}{V_0 T_0^2}\, e^{\frac{B}{T_0}}\, V_{DD}\, T^2\, e^{\frac{-B}{T}} \tag{9}$$

where R_0, V_0, T_0, and B are constants that are obtained through fitting from a series of measurements on actual target platforms.

If T is considered to be constant across different operating voltages, then R_T is a function of the voltage multiplied by a constant γ, which includes temperature-dependent variables and other constants. The total power of the CPU is then simply the combination of the dynamic and leakage current formulas from Skadron et al. [9]. It is presented by the following equation:

$$\begin{aligned} P_{cpu} &= P_{dynamic} + P_{leakage} \\ &= (1 + \gamma V_{DD})\, \xi f_{cpu} V_{DD}^2, \end{aligned} \tag{10}$$

where γ is a parameter describing the magnitude of the leakage currents due to capacitor-based circuits and ξ a parameter defining the power requirements of the microprocessor.

2.2 Execution Time Model

Applications usually run on top of an operating system. So, in order to get an estimate of an application computing time, we need to account for the time consumed by the OS to perform periodical tasks, like process scheduling, interrupt handling, and processing kernel events. De Vogeleer et al. [3] have proposed to model the total time needed to complete a program (t) as follows:

$$t = \frac{cc_b}{f_{cpu} - f_{cck}} + t_{cck} + t_{idle}, \tag{11}$$

where t is the total time needed to complete an application, cc_b the number of clock cycles to complete the instructions of an application, f_{cck} the inverse of the number of clock cycles per time unit required by the OS kernel, and f_{cpu} the CPU's clock frequency.

The time accounting for the kernel overhead to perform periodical tasks (t_{cck}) and the operating system's idling (t_{idle}) is (generally) small compared to the time that the CPU is able to perform useful work, i.e., execute applications. Hence, Equation 11 can be rewritten as follows:

$$t = cc_b\left(\frac{1}{f_{cpu} - f_{cck}} + \beta\right), \tag{12}$$

where β is a system architecture-dependent scaling factor.

2.3 Energy Consumption Model

Typical compute-bound non-performance-critical applications incur a constant load on the CPU. The power requirements of the CPU increase linearly with the increase in CPU utilization. The energy consumption of a benchmark is thus given by the product of its average power and execution time.

Thus, from Equation 10 and Equation 12, the energy consumption of the CPU for running a given benchmark for t_{bench} seconds at fixed temperature and power (P_{bench}) can be modelled as follows:

$$\begin{aligned} E_{CPU}(t_{bench}) &= P_{bench} \cdot t_{bench} \\ &= ((1 + \gamma V_{DD}) \xi f_{cpu} V_{DD}^2) cc_b \left(\frac{1}{f_{cpu} - f_{cck}} + \beta\right). \end{aligned} \tag{13}$$

3 PARAMETER ESTIMATION AND VALIDATION

The energy consumption model defined by Equation 13 has earlier been validated by De Vogeleer et al. using measurements obtained from a Samsung Galaxy S2 test bed. This smartphone sports a 1.2GHz dual core ARM Cortex-A9 processor, which uses Samsung's own Exynos 4210 SoC [4]. They used the Gold-Rader implementation [5] of the bit-reverse algorithm, which is the part of the ubiquitous Fast Fourier Transform (FFT) algorithm that deterministically rearranges elements in an array; this kernel is often considered the reference algorithm for FFT applications.

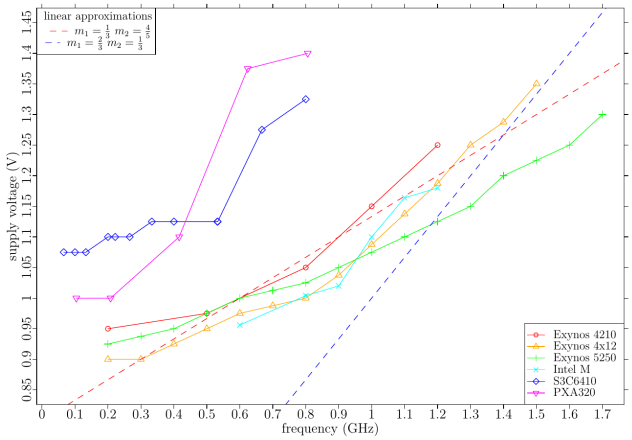

Figure 1: Frequency/voltage relationships of multiple application microprocessors as found in the Linux kernel. [3]

Different array sizes N were tested (N = 6, 8, 10, 12, 14, 16), with various frequencies. Figure 1 shows the Frequency/Voltage relationships for various application microprocessors found in the Linux kernel. The red dotted line is fitted on the depicted data from the first three microprocessors (Exynos 4210, Exynos 4x12 and Exynos 5250). It represents the linear relationship between supply voltage and frequency of operation and has been used in their curve fitting process.

The experimental data (see Figure 2) show their measured and modelled energy E_{CPU} for the benchmark kernel with different operating frequencies. The curve-fitting process used for fixing the parameter values used in Equation 2 invokes the non-linear least squares (*nls*, in R) technique, which minimizes the sum of the squares of the discrepancies between the curve and the data. This technique fits the power and execution time equations to the experimental data and yield the unknown parameters from Equation 13. The data shows that the analytical model for the energy consumption fits nicely with the measured data. Indeed, the absolute error between the fitted model and the measured data for the energy consumption stays well below 6%.

4 EXPERIMENT

To provide a detailed analysis of the energy consumption model described in Section 2, we selected a more advanced experimental setup than the one used by De Vogeleer et al. [4]. We also focused on an extended set of benchmark kernels on which to perform minute power and execution time measurements.

Test Platform. To provide a detailed analysis of the energy consumption model described in Section 2, we chose the AM572x EVM development board from Texas Instruments (TI). It is equipped with a high-performance Sitara™ SoC running GNU/Linux with kernel release version 4.9.59. The SoC is implemented using a 28-nm

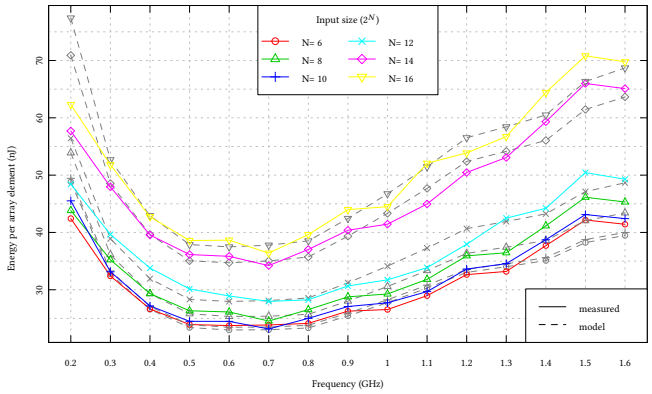

Figure 2: Energy required by the CPU at 37°C to complete our benchmark kernel given an input size. The dashed lines denote the theoretical curve as per Equation 13.

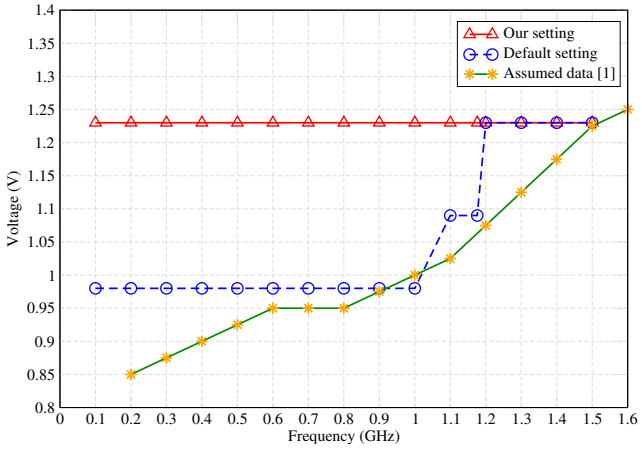

Figure 3: Frequency/Voltage relationship for TI AM572x evaluation module. The plot also includes the assumed voltage data for [4].

process technology and is comprised of several subsystems. The MPU subsystem incorporates two ARM Cortex-A15 cores. It also incorporates individual level 1 (L1) and level 2 (L2) caches which are shared between the cores and includes various other shared peripherals. The subsystem supports a configuration to completely shut off one core and run the other at low voltage and low frequency to achieve low-power operation [6].

The AM572x EVM board offers current-sense resistors for all its submodules, including the MPU. These resistors provide access to the power supply rail and allow continuous power monitoring in real time during software execution. We modified the board by soldering male headers across the current-sense resistors to easily connect probes. The resistor's value is 0.01 Ω and has been chosen to provide the best possible dynamic range during data acquisition.

We use a compact data acquisition device from National Instruments (NI cDAQ-9174) for all our data acquisition needs. NI CompactDAQ is a portable platform that integrates connectivity and signal conditioning into I/O modules that can directly interface with many different sensors. We use an NI 9215 voltage input module to measure the voltage drop across the sense resistor. We then calculate the current using Ohm's law and multiply it with common-mode voltage to get power values.

Finally, our experimental setup includes a Windows host machine running the LabVIEW software, the NI cDAQ with the NI 9215 voltage input module plugged in, and the TI AM572x board.

Protocol. We put our platform to the test using different workloads and measured three key characteristics (execution time, average power, and energy consumption) while varying the clock frequency in steps of 100 MHz between 100 MHz and 1500 MHz, while keeping the supply voltage at a fixed setting. We evaluated the power and execution time models presented in [4] with the data obtained from our experimental platform.

For our experiments, we use two cryptographic benchmarks, namely SHA and Blowfish, from the BEEBS suite [8] and the GoldRader *bit-reverse* algorithm. The benchmarks are run 3 times in similar environmental conditions and it has been found that there are no significant variations in terms of energy consumption.

5 ANALYSIS

Based on the experimental data acquired as described in Section 4, the adequacy of the analytical model of energy consumption of Section 2 was analysed. Here, we present our main findings.

5.1 Frequency/Voltage Relationship

The underlying assumption of [3] is that the relationship between frequency and supply voltage in a DVFS process is approximately linear for modern microprocessors. The documentation for the TI AM572x EVM board used here, for instance, indicates three operating modes for frequencies from 100MHz to 1GHz, from 1GHz to 1.176GHz, and from 1.2GHz to 1.5GHz. For each of these modes, a reference voltage (1.06 V, 1.16 V, 1.21 V respectively) is indicated with a range of ±10% at run time. This would roughly match the hypothesis underlying the analytical model.

There is, however, a caveat. The voltage does not vary dynamically at run time with regard to these reference voltages. Instead, a fixed value is preset by the manufacturer by automated testing. The determined voltage is stored in the AVS Class 0 registers and cannot be modified (Table 18-26 of the AM572x TRM [6]). The frequency-voltage relation in the DVFS process is then no longer linear. For example, on our TI AM572x EVM board, the operating voltage is preset to 0.98 V, 1.09 V, and 1.23 V respectively.

For our experiments, we set the CPU target voltage to 1.23 V for the entire frequency range to observe the effect of frequency scaling in isolation. Figure 3 depicts the Frequency/Voltage relationship for the TI AM572x EVM, the assumed voltage data for [4] and the voltage setting used in our experiments.

Finally, note that memory is clocked at a fixed, independent frequency. For the compute-bound benchmarks used in our experiments, the static energy consumption is hence proportional to the CPU clock speed.

5.2 Execution Time Model Validation

Figure 4 shows the performance of the benchmarks, i.e., execution time, depending on the CPU's clock frequency on the TI AM572x

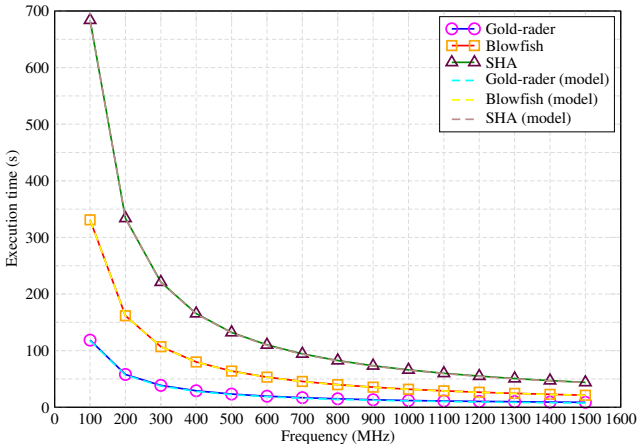

Figure 4: Performance of the benchmarks with varying clock frequency on the TI AM572x platform.

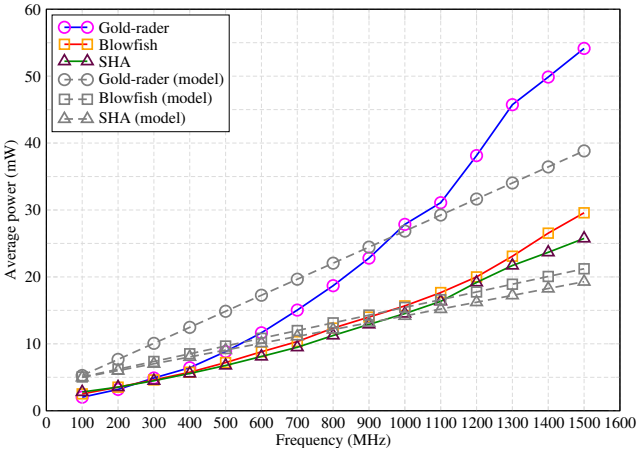

Figure 5: Average power of the benchmarks with varying clock frequency on the TI AM572x platform

EVM. The execution time decreases in a clear non-linear fashion for our CPU-bound applications as the frequency increases. This is hardly surprising and underlines the importance of frequency scaling, including Dynamic Frequency Scaling, for performance. While the energy consumption of the SoC, when running a benchmark, also depends on the benchmark's execution time (recall that leakage power and other factors that are independent of CPU load may play a role here), the clock frequency may have a more significant impact. Figure 4 also includes the non-linear least squares fit of the execution time model (Equation 12) on our experimental data. It can be seen that the curve fits the data points very well.

5.3 Power Model Validation

The power requirements change depending on the presence or absence of a heatsink on the CPU die and on whether or not it has been connected to a fan. In our experiments, the power values are measured when the board under test has the heatsink attached with no fan connected. Figure 5 shows that the time-averaged power (using 1 ms time resolution) increases steadily with the frequency for all three benchmarks (SHA, Blowfish and Gold-Rader) in an almost-linear fashion. This accounts for the power of the entire MPU subsystem, i.e., both ARM cores, where one core is running the benchmark and the other is idle – but not powered off. Note that the subsystem also includes the core's private L1 caches, a shared L2 cache, and the necessary interconnect. These components cannot be controlled separately in software, but may transition themselves into low-power modes independently from the CPUs. Reducing the clock frequency minimizes the average power, but this may not be the best strategy, as the non-linear increase in execution time may offset potential gains.

Figure 5 also includes a non-linear least squares fit of the power model (Equation 10) on our experimental data. It can be seen that the curve does not fit the data points very well. This is because, for our experimental voltage setting, which is set to a constant 1.23 V, Equation 10 becomes linear. Hence, it does not neatly fit our non-linear experimental curve.

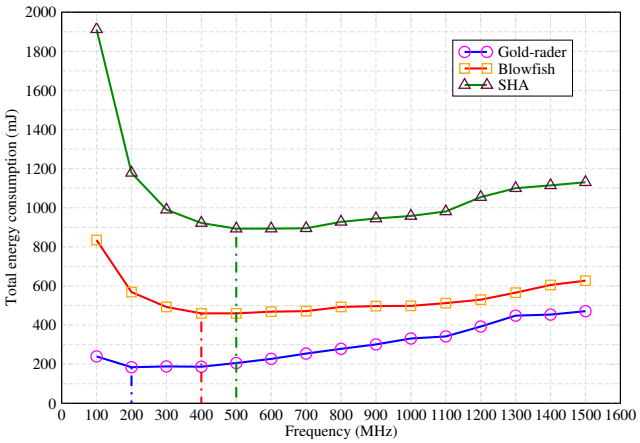

Figure 6: Total energy vs. Frequency: energy consumption of our three benchmarks with varying clock frequency on the TI AM572x platform. The dotted lines represent the optimum frequency points for each benchmark.

5.4 Energy/Frequency Convexity Rule

Energy-management approaches often neglect that frequency scaling is highly dependent on workload characteristics. Figure 6 compares the accumulated total CPU energy consumption for each of our three benchmarks with varying frequency on the TI AM572x platform (accounting for the same subsystems as for Figure 5). The energy consumption curves are clearly convex, each having an optimal frequency point (f_{opt}) where the energy consumption is minimized. Our experiments thus further support De Vogeleer et al. [4]'s previous work revealing the existence of the Energy/Frequency Convexity Rule for compute-intensive applications running on an Exynos-based platform in a Samsung Galaxy S2 phone. Their convexity curves can be seen in Figure 2.

The convexity of the energy curve is in part due to the processor's static power, which is independent of the core frequency. As the processor runs with a faster clock, it requires less total time, and the total static energy consumption is less in those cases. On the other

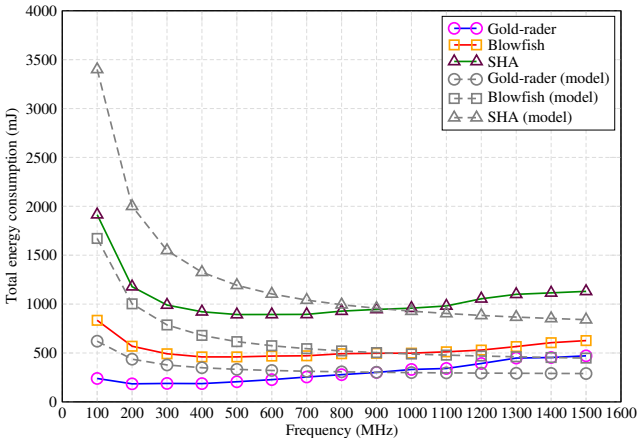

Figure 7: Total energy consumption of our three benchmarks with varying clock frequency on the TI AM572x platform. The modelled energy curves do not fit the obtained experimental measured data.

hand, our experimental data and Equation 12 show that the time it takes to execute a workload increases more than linearly when decreasing the CPU's frequency, while leakage power continues to be drawn during the time the CPU is active. This leakage contributes to the convexity at the lower end of the frequency range, i.e., when the operating frequency f is lower than the optimal frequency f_{opt}. For the region where the operating frequency is greater than the optimal frequency ($f > f_{opt}$), the energy consumption is attributed to the increase in frequency, which contributes to dynamic power.

5.5 Temperature Impact

The estimated experimental energy consumption values above were obtained by multiplying the data in the power traces with those in the execution time traces for each frequency. A similar computation was also performed with the fitted power and execution time values. Figure 7 shows the energy consumption for all the three benchmarks. Clearly, the energy consumption predictions obtained from the model do not describe convex curves and also do not fit the experimental data measured on the board. This is a direct consequence of the linear relationship between frequency and power in Equation 10 (recall that our experiments use a constant V_{DD}).

Since, having a constant voltage did not affect the Energy/Frequency convexity property of our experimental data (see also [10]), this shows that another parameter needs to be taken into account in our model, which is not voltage but one which has a non-linear relation with power. Interestingly, during our experiments, we have noticed that the core temperature can increase by about 10°C for the SHA benchmark when run using Linux's *ondemand* frequency governor. This is a direct consequence of the long execution time, in the order of several hundreds of seconds, and the high CPU demand of this compute-bound kernel. Note that, on the contrary, the Gold-Rader kernel has a short execution time. Hence, De Vogeleer et al. have rightly assumed that the temperature can be deemed constant during the entire execution cycle of their short benchmark, thus not seeing the discrepancy between data and model that our more advanced protocol put to light. Our own use of Gold-Rader, which we are looping 1024 times in each run, induces significant temperature changes along the way.

Hence, temperature variation cannot, in general, be totally ignored in the energy modelling, as was done in Equation 13. We conjecture that the most important missing variable in our power model is thus more than likely temperature, and that a formal relationship between power and temperature needs to be established. A first approach to this question is the work of Skadron et al. [9], which shows that power is proportional to the square of the temperature. Of course, simply inserting the temperature variable in the model does not mean that the regression analysis will be able to resolve the relationship between temperature and power. More research is thus clearly warranted to tackle this issue, the importance of which has been put into light by our experiments.

6 CONCLUSION AND FUTURE WORK

In this work, we evaluate the analytical model of the CPU energy consumption introduced by De Vogeleer et al [4]. Based on a reconfirmation of the Energy/Frequency Convexity Rule for CPU-bound benchmarks using a highly accurate workbench, our results show that the convexity persists despite the experimental voltage setting being set to a constant value. We found that, when benchmarks have longer execution time, their junction temperature of the CPU is not constant, rendering the fit with energy consumption models more challenging. We propose to introduce the temperature variable in the power consumption model to handle this issue.

Future work will address the temperature issues put into light here. In particular, we intend to perform more accurate temperature-dependent performance experiments, using an industry-grade temperature control chamber. This would help assessing the importance of this parameter and provide food for thought to design a temperature-aware formal model for energy consumption.

REFERENCES

[1] Anantha Chandrakasan, Isabel Yang, Carlin Vieri, and Dimitri Antoniadis. 1996. Design considerations and tools for low-voltage digital system design. In *Proceedings of the 33rd annual Design Automation Conference*. ACM, 113–118.

[2] Yi-Kan Cheng, Ching-Han Tsai, Chin-Chi Teng, and Sung-Mo Steve Kang. 2000. *Electrothermal analysis of VLSI systems*. Springer Science & Business Media.

[3] Karel De Vogeleer, Gerard Memmi, and Pierre Jouvelot. 2017. Parameter sensitivity analysis of the Energy/Frequency Convexity Rule for application processors. *Sustainable Computing: Informatics and Systems* 15 (2017), 16–27.

[4] Karel De Vogeleer, Gerard Memmi, Pierre Jouvelot, and Fabien Coelho. 2013. The energy/frequency convexity rule: Modeling and experimental validation on mobile devices. In *International Conference on Parallel Processing and Applied Mathematics*. Springer, 793–803.

[5] Bernard Gold and Charles M Rader. 1983. *Digital processing of signals*. Krieger Publishing Co., Inc.

[6] Texas Instruments and AM572x SitaraTM Processors Silicon Revision. 2014. 2.0, 1.1 Technical Reference Manual. *Literature Number SPRUHZ6H* (2014).

[7] Nam Sung Kim, Todd Austin, David Baauw, Trevor Mudge, Krisztián Flautner, Jie S Hu, Mary Jane Irwin, Mahmut Kandemir, and Vijaykrishnan Narayanan. 2003. Leakage current: Moore's law meets static power. *computer* 36, 12 (2003), 68–75.

[8] James Pallister, Simon Hollis, and Jeremy Bennett. 2013. BEEBS: Open benchmarks for energy measurements on embedded platforms. *arXiv preprint arXiv:1308.5174* (2013).

[9] Kevin Skadron, Mircea R Stan, Karthik Sankaranarayanan, Wei Huang, Sivakumar Velusamy, and David Tarjan. 2004. Temperature-aware microarchitecture: Modeling and implementation. *ACM Transactions on Architecture and Code Optimization (TACO)* 1, 1 (2004), 94–125.

[10] Kameswar Rao Vaddina, Florian Brandner, Gérard Memmi, and Pierre Jouvelot. 2017. Experimental energy profiling of energy-critical embedded applications. In *25th international conference SoftCOM 2017*.