Modeling the energy consumption of programs: thermal aspects and Energy/Frequency Convexity Rule

Karel De Vogeleer[#], Kameswar Rao Vaddina*, Florian Brandner*, Pierre Jouvelot°, Gerard Memmi*

[#]Hypervirtu, Brussels, Belgium
*LTCI - TELECOM ParisTech - University of Paris-Saclay, Paris, France
°MINES ParisTech, PSL Research University, Paris, France

Email: k.de.vogeleer@gmail.com,
{kameswar.vaddina,florian.brandner,gerard.memmi}@telecom-paristech.fr,
pierre.jouvelot@mines-paristech.fr

Abstract: *This article summarizes our current studies aiming at a better understanding of the energy consumption of a microprocessor during the execution of an application through a combination of theoretical results and experimental validations, The analysis of the transient thermal behavior and energy gains (ranging from 20 to 40% in some cases) via the adaptation of the clock frequency are of obvious practical interest. A general Passive Cooling Rule (PCR) for an isothermal object subjected to radiation, convection and internal heat generation is proposed. This power-temperature model is observed on an Exynos 5410 processor. Several approximations to this cooling rule are formulated for practical use, particularly online. They are accompanied by general rules for assessing when passive cooling becomes non-negligible compared to active cooling in embedded systems. On another hand, a theoretical framework for the existence of an Energy/Frequency Convexity Rule (EFCR) of program consumption is established. It is validated both by the state of the art and by experimental measurements where the impacts of variation of multiple parameters are studied. Power requirement models are then explained for the Exynos 5410 integrating the clock frequency, temperature and number of active cores. The novelty of these models is that they take into account certain characteristics of the running programs and that they can be directly reused in any simulation for other processors of similar architecture.*

Keywords: Application energy profiling, energy-critical, energy consumption, frequency-energy convexity rule, passive cooling rule, Exynos processor, Cortex processor, DVFS, TMU, embedded systems

1. Introduction and motivations[1]

1.1. Motivations and energy-critical applications

A hardware and software system or an application are called “energy-critical” if the slightest change in available energy can have a significant impact on its results and use. From this point of view, any program running on a battery-powered system can prove to be energy-critical; this is the case for many embedded systems, wireless sensor networks or mobile phone applications. Whatever its origin, any portable device exhibits this same Achilles’s heel: the battery. This class of programs or applications is vast and justifies by itself to study and better understand the energy and thermal behaviors of programs associated with the hardware platform on which they will be executed. We could add at the other end of the spectrum, high-performance computing systems (HPC) which, if they are not energy-critical per se, are also constrained by both temperature and available power supply. These systems often have to work within a constrained budget including energy consumption.
Numerous and important works focus on the chemistry of batteries to improve their capacity and longevity; other works focus on energy harvesting; other ones, on electronic design computer architecture: all these works aim at increasing application autonomy. Microprocessor manufacturers have well understood the economic and ecological stakes and propose architectures allowing to play with clock frequencies and load balancing in order to better manage energy consumption. For example, the heterogeneous "big little" architecture proposed by ARM (Cortex A53) and then by Samsung (Exynos 5210) combines a low-power processor with a high-power

[1] This paper is an extended version of De Vogeleer, K., Memmi, G., and Jouvelot, P. *Modélisation de la consommation énergétique des programmes : aspects thermiques et loi de convexité énergie-fréquence*. Génie logiciel, 6, 2016.

processor to better adjust the computation power to a given application needs, optimizing the energy consumption more efficiently than by simply choosing a clock frequency.
However, regardless of the progress of this work, the need for optimizing the efficiency of the energy consumed by a battery-powered system will persist. One reason among others is the constant miniaturization of electronic communicating objects combined with a reduction in the volume and weight of batteries, with, icing on the cake, a need for a restlessly increasing level of autonomy. This has been confirmed by academic studies [20] as well as marketing surveys, since the creation of portable devices. The energy efficiency of battery-based systems remains critical, as demand for power continuously outperforms developments designed to increase battery capacity ([34], [45]).
Better management of clock frequency as well as better control over thermal effects of microprocessors will have significant impacts on energy behavior. It is this fundamental idea that we pursue and we seek, in the first place, to understand and model these relations with accuracy, considering both physical parameters and software characteristics. To this end, it is first necessary to be able to experiment, to measure with accuracy, to collect data, to deduce a model and to validate it. Then, it will be possible to exploit statically or dynamically this model by using a set of controllers allowing to play with clock frequencies or to balance the load of the various processors of an embedded system. Knowledge of the hardware architecture of a system and its behavior with regard to physical quantities is, of course, of paramount importance, in particular, its thermal behavior. However, we believe that knowledge about the software that will run on the given hardware platform must be added for an optimized resource allocation process and, furthermore, to be able to build an energy program profile for, then, predict and control the system energy consumption. Today, operating systems can act more and more efficiently on the clock frequency or the voltage of microprocessors in order to slow them down or accelerate them and thus save on the total consumed energy or, on the contrary, to obtain the best possible performance ([42], [38]). Another important factor is the influence of temperature on power. Numerous publications attempt to measure or model these factors under various situations (mobile, high-performance computing, datacenter management). Published earnings range from 10% to 40%.

1.2. Energy profiling

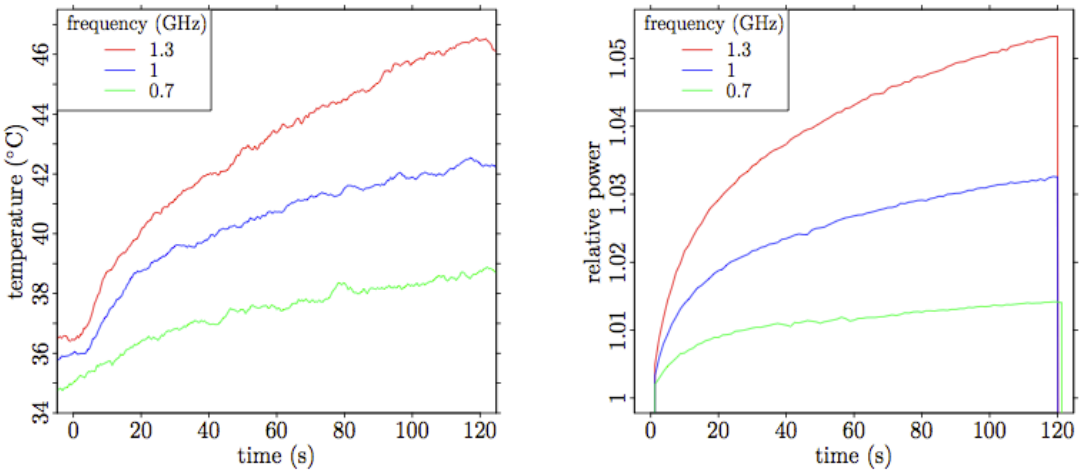
Clock frequency, processor temperature and program run time are among the important factors influencing energy consumption and are key parameters of the energy profile of a program execution on a given hardware platform. Providing energy consumption models will pave the way for optimizing energy for combined software and hardware systems such as embedded systems. Current and future technologies will allow to select a subset of its logic and to keep it active at a given time to remain compatible with the acceptable levels of maximum power dissipation; the deactivated fraction of the logic is called *black silicon*. In addition, power and physical transmission time limits have kept the maximum clock frequencies of current and future technologies [17] constant during the execution of a program. Consequently, to improve performance, microprocessor architectures employ solutions with increasing degrees of parallelism. For example, multi-core simultaneous multithreading (SMT) has provided substantial energy savings with recent hardware [16].
In addition to optimizing energy efficiency at the hardware level, software can also be designed to minimize energy consumption. For embedded systems, at least six traits of software energy optimization can be considered: energy-oriented operating systems, efficient resource management, the impact of configuring user interaction with mobile devices and applications, wireless interfaces and radio transmission, sensor management, and cloud computing services [45]. All these facets must be optimized globally to achieve the greatest energy gain, which is an ambitious and complex task.
At the software level, energy and power optimization techniques can be divided into static or dynamic methods. Static methods include, among others, efficient software design and optimization of the compiler backend. For these approaches to efficiently optimize energy and power, an energy profile of the hardware on which the software will be run is required. Of course, the software generated will only be optimal for a given hardware platform. Some environment variables may not be fully known ahead of time, such as the ambient temperature or the level of humidity. Furthermore, the design of an energy-efficient architecture is very sensitive to the necessarily variable load of the processor [16]. This makes us understand the limitations of static optimization techniques, which can only be overcome by dynamic techniques.
Dynamic optimization will therefore use a set of contextual information available only at run time. The outside temperature is a typical example. The hardware and software system can dynamically adapt to improve energy efficiency and power. Online methods include dynamic compilation optimization, byte-code optimization, and optimization techniques at the operating-system level. As with static optimization, knowledge of the energy profiles of hardware and software helps to improve the decisions made by dynamic energy and power optimization techniques and are necessary to be able to anticipate such decision making.

1.3. Contributions

This paper summarizes the main results set out in K. De Vogeleer's PhD thesis [14] whose objective is to focus on the energy profiles of programs and, more precisely, on the way in which two fundamental parameters, namely temperature and clock frequency of microprocessors, affect energy consumption of applications or programs as well as the optimum operating conditions of a computer system. After this introduction, Section 2 discusses two relationships addressing the influence of temperature. A power-temperature analytical model is illustrated by numerous examples from the literature. This model is useful for canceling the temperature bias in power measurements and increasing their accuracy. A cooling rule for a supposedly isothermal object subjected to radiation, convection, and internal heat generation as well as two approximations better suited for practical use are given in Section 2.2. The Energy/Frequency Convexity Rule (EFCR) of a given program energy consumption on a single-core microprocessor is developed in Section 3; it was first published in [11] and has been validated experimentally with different microprocessors ([11], [46]). The article concludes with a look at future work.

2. Two relationships describing the influence of temperature

When studying energy consumption on the basis of power measurements, it is imperative to experiment accurately in order to obtain reliable power samples and reproducible measurement traces by controlling a set of physical and environmental variables likely subject to noise measures. Temperature (T) has a significant impact on the energy consumption of microprocessors as well as on their performance. The example in Figure 1 shows an increase in energy consumption of 5% for a temperature increase of 10 °C at a clock frequency of 1.3 GHz. It also shows that, the higher the temperature, the higher the demand for power. It is this last relation between power and temperature that we will study. Then, we will consider a passive cooling rule which is the only one that has to be taken into consideration in the case of small sensors or in mobile phones without active cooling circuits.



(a) Temperature (b) Power

Figure 1: Temperature (a) and power (b) of an Exynos 4210 under a constant workload, operational at different microprocessor clock rates. It is observed that the energy consumption increases rapidly due to the increasing temperature of the microprocessor, all other parameters being kept constant.

2.1. Temperature-power relationship

The electric currents in a microprocessor, or in any integrated circuit (IC) for that matter, respect the fundamental laws of electricity (Ohm, Kirchhoff, ...). Thus, they produce heat dissipation that is proportional to I^2R, where R is the resistance and I the overall electrical current of the system studied. The first law of thermodynamics indicates that, in regular operation, the energy input of a system is equal to the energy consumption of the system. Thus, in the absence of other energy interactions and neglecting information related to energy itself, the only form of energy emanating from a microprocessor is the heat generated by the currents going through resistive elements [8]. Therefore, the heat dissipation of the microprocessor is very close to its energy consumption.

Moreover, a microprocessor exhibits a transient thermal behavior where the cooling and heating times depend on its heat capacity; the higher the heat capacity of the systems, the slower their transient thermal behavior is. These systems are traditionally modeled by RC circuits, based on the current/heat equivalence, i.e., a low-pass filter, where the temperature of the system is proportional to the voltage across the capacitors [10]. In fact, thermal (transient) behavior is more complex, because the resistance R and the current I vary with the evolution of the microprocessor temperature. In general, experience shows that the power grows super-linearly with the temperature of the system under study. One can, however, remind the reader about the linear bound in kT (where k is the Boltzmann constant) of heat dissipation during a machine cycle. This lower bound based on information theory and developed by Brillouin and then by Landauer and Bennet [4] remains theoretical and very far from the measures performed on the best hardware technologies.

In addition to other physical properties, leakage currents certainly are the greatest contributors to this relation between the microprocessor power and its temperature, which may contribute to explain the exponential relationship between power and temperature. To put things into perspective, we have experimented with the ARM A15 processor, which is used extensively in embedded systems; in the most extreme cases, the energy consumption at 85 ^{o}C were 20% higher than at 25 ^{o}C. More fundamentally, the work we carried out ([13, 14]) concluded that an exponential model of power as a function of temperature provided a high level of accuracy over a temperature ranging from 25 °C to 85 °C, but also that a quadratic model was quite sufficient over a smaller temperature interval. This model is of the type:

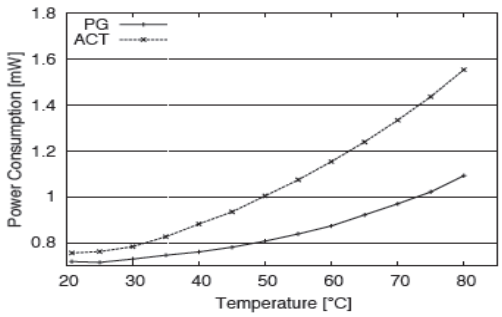
$$P = a_1 e^{T/a_2} + a_0,$$

where the parameters a_0, a_1 and a_2 depend directly on the type of microprocessor used. This power-temperature relationship can be used to suppress the bias of the temperature over a power measurement trace.

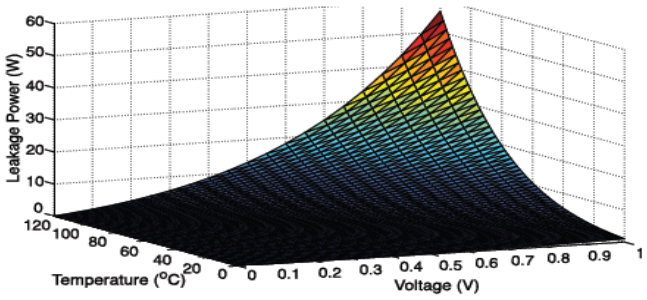
The distance of a temperature probe from a heat source can introduce also errors on power measurements, such as a time shift or a reduced temperature amplitude. A somewhat ad hoc transformation function can be used to compensate for such behavior. Such models are of interest for the following reasons.

- Measurements are essential for understanding and optimizing energy-critical systems ([16], [45]). A lack of detailed power measurements will undermine efforts to reduce energy consumption on modern software, embedded systems, or the Internet of Things (IoT) [16].
- Several research studies have attempted to describe the power-temperature relationship by focusing on a subset of the leakage currents described by BSIM (Berkeley Short-channel IGFET Model) ([19], [30], [31], [41], [44]).
- Such models assume that the leakage currents are only temperature-dependent. However, the currents are also a function of time, because the voltages applied across transistor terminals change over time, which introduces additional modeling complexity. The leakage currents also depend on a multitude of factors specific to each integrated circuit technology, which are not all yet accurately known. Therefore, it may be interesting to propose a general modeling of the power-temperature relationship, as an alternative approach to models derived from BSIM.
- It is essential to control error rates due to power measurements according to the different temperature variations of the system under study. For example, some of our power measurements exhibit up to 10% deviation due to transient temperature effects [14]. Thus it is important that the effects of temperature can be understood and controlled so as to obtain a fair and accurate basis of analysis in order to efficiently compare different power measurements.
- From the point of view of measurement, it is also important to understand the impact of internal heat generation on temperature and the effect of remote sensors. Precise and repeatable power measurement protocols are difficult to establish due to the transient thermal behavior of electrical components. More precisely, for a given test program, measurement of the power supply of the microprocessor can lead to different values for different temperatures of the microprocessor. For the sake of precision and fair comparison between different power measurements, it is therefore critical to know how to control, compensate or even cancel the effects of transient thermal behavior.

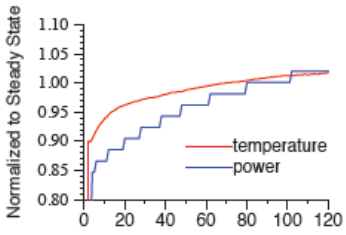
Moreover, it is important to constantly study the various power-temperature models of the literature in order to optimize the DVFS (Dynamic Voltage and Frequency Scaling) modules and of course the TMU (Thermal Management Unit). Figure 2 contains a set of results presented in various publications which all show an increase in the required power as a function of temperature. Weissel and Bellosa [48] developed a TMU for a computing center based on a set of measurements over a temperature ranging from 35 °C to 60 °C.



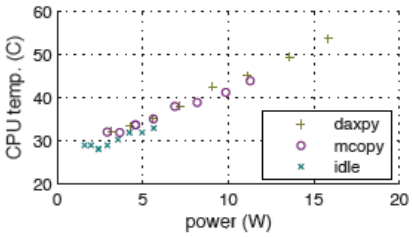
(a) [26]



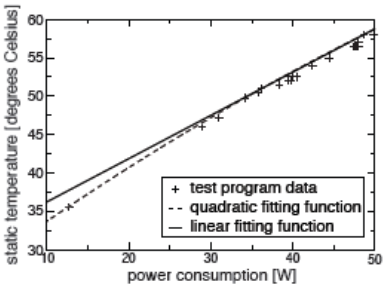
(b) [25]



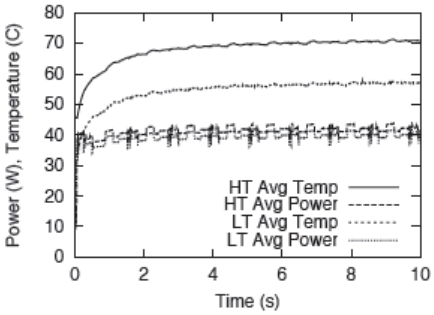
(c) [40]



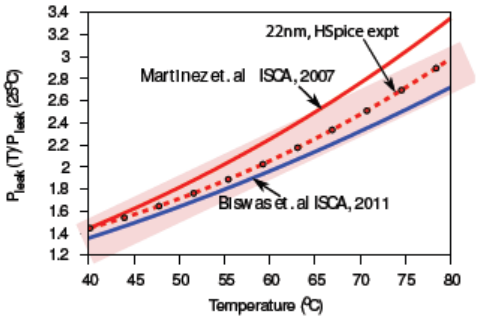
(d) [23]



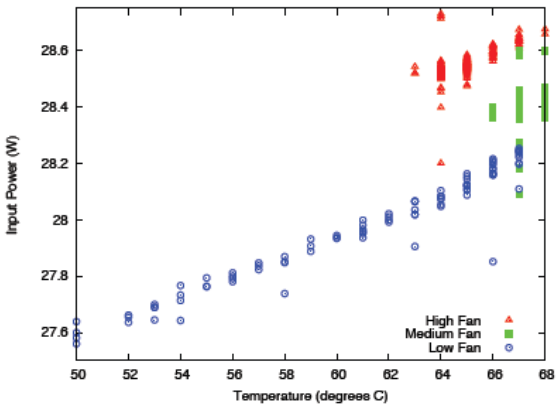
(e) [48]



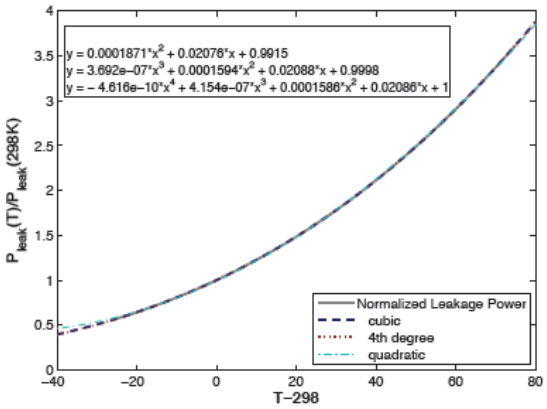
(f) [33]



(g) [35]



(h) [43]



(i) *[6]*

Figure 2: Figures showing the required power as an increasing function of temperature.

2.2. Passive cooling of microprocessors

The reliability of electronic products is strongly influenced by spatial and temporal temperature gradients, as well as by their absolute values [28]. Thermal gradients, which occur in both space and time, induced by the variability of the microprocessor load and the operations, are generating thermal cycles that have negative effects on the Mean Time Between Failures (MTBF) of systems [27]. The International Technology Roadmap for Semiconductors (ITRS) even claims that processing costs and performance specifications may be limited by service life, with reliability becoming the primary concern in the design phase of a microprocessor [24]. The Mean Time To Failure (MTTF) or end of operation of an electronic equipment decreases exponentially with temperature; possible causes of failure are electro-migration, chemical reactions, dielectric rupture, or hardware creep [8]. An increase in temperature from 10 ^{o}C to 15 ^{o}C can reduce the life of a microprocessor by half [47]. Therefore, the temperature of a system is often limited to control heat dissipation at maximum power, optimizing MTTF, minimizing energy consumption, avoiding self-destruction (especially when playing with overclocking), as well as for user safety reasons. Smartphones are often thermally capped around 50 ^{o}C, so users do not burn themselves, but also in order to efficiently use the electrical capacity of the battery. It should be pointed out that the surface temperature in contact with electronic systems should be limited to 41 ^{o}C or 45 ^{o}C, depending on the material, to ensure a comfortable user handling [5].

Advanced electronic equipment uses Thermal Management Units (TMU) or Dynamic Thermal Management (DTM) techniques that are capable of slowing down or adapting systems in order to meet their sometimes stringent constraints. There exists a vast array of thermal control methods for microprocessors and Systems-on-Chip (SoC) ([3], [27]); they act on the compromises between thermal profile, frequency parameters, energy consumption, and thermal management device implementation complexity [51]. Some complex microprocessors may employ hardware-based TMUs, as some Intel microprocessors ([15], [32]) do, while software TMUs are frequently implemented in embedded systems. To effectively use TMUs, it is important to understand the transient thermal behavior of the system under study. This transient thermal behavior is driven by the way the microprocessor dissipates its heat in the surrounding environment. Active cooling systems release their heat into the environment by forced convection, e.g., air or other types of fluid, which surpasses other heat transfer modes. The transient behavior of these systems is well described by an exponential cooling rule.

However, passive cooling devices, such as smartphones, wireless mobile radio sensors or many other types of objects used in IoT, rely on the natural dissipation of heat, including radiation. The complex effects of radiative cooling on transient thermal behavior have been explored through an experimental and analytical framework as accurate as possible in [14]. To control this complexity, some simplifying hypotheses are necessary; in particular, a microprocessor will be represented by an isothermal part of silica oxide that will be placed in an infinite open space subjected to (natural) convection and radiation. It is clear that this assumption is not always verified.

Under the hypothesis of the sole presence of convection, radiation, and internal heat generation, the so-called Passive Cooling Rule (PCR) in the presence of radiation can be expressed by means of a temporal variable t as a function of the temperature T (it would be natural to have the inverse, i.e., the temperature as a function of time, but this function is even more complex), as follows:

$$t = -\frac{1}{\kappa_4}(A\ln|T-\omega_1| + B\ln|T-\omega_2| + \frac{C}{2}\ln|(T-\alpha)^2+\beta^2| + \frac{\alpha C + D}{\beta}\arctan\left(\frac{T-\alpha}{\beta}\right) + c_o). \quad \text{(PCR)}$$

A use case applied to a microprocessor-like object shows that the radiation-related component cannot be neglected for objects with a cooling surface area greater than 1 dm^2, which could be the cooling surface typical of a smartphone. For smaller cooling surfaces, the exact passive cooling rule approximates an exponential cooling rule. Under such conditions, an exponential cooling rule must be favored, since it is much less complex than the exact PCR. Approximations to the exact cooling rule have also been provided in [14]; they are intended for use in practical applications and, even, online applications (in particular in embedded systems). It has been shown that the following approximation

$$T = \frac{\sqrt{\kappa_1^2 - 4\kappa_2\kappa_0}(1 + c_o e^{-\frac{\kappa_2}{A}t})}{2\kappa_2(1 - c_o e^{-\frac{\kappa_2}{A}t})} - \frac{\kappa_1}{2\kappa_2} \quad \text{(PCR2)}$$

works best in most circumstances. For small deviations at room temperature, O'Sullivan's second-order

approximation method

$$T = \frac{w}{2m}\tanh\left(\frac{w}{2C}t + c_o\right) - \frac{n}{2m} + T_a, \qquad \text{(PCR3)}$$

can also be used satisfactorily. The three PCR rules for the passive cooling of objects are increasingly easy to efficiently compute (which is important for real-time or near-real-time applications), but, on the other hand, less and less precise (see [14] for comparisons of PCR2 or PCR3 error rates relative to PCR). These models can be useful in thermal management (TMUs) and dynamic (DTM) components to evaluate and predict the thermal behavior of a given microprocessor.

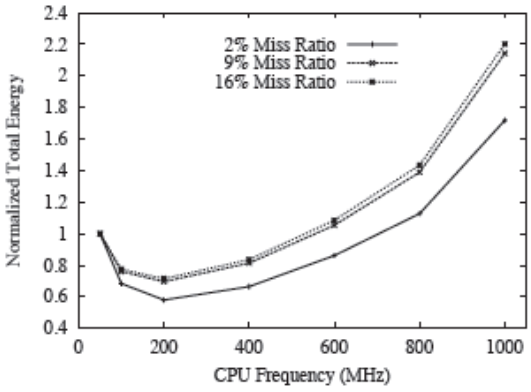
3. EFCR, the Energy/Frequency Convexity Rule

In addition to temperature, other parameters influence the energy profile of a system. The program's run time and the various characteristics of power demand of the software and hardware system are the main factors that define energy consumption. Considering them in a global manner is one of the original points of our work.
A program execution time is influenced by the type and cost of the various operations performed by the software in question, including the instructions accessing the external memory and calling the routines of the operating system. Each functional unit within a microprocessor and system component has its own power profile and run time. As a result, different code sequences exhibit different energy and run-time demands. For instance, Carroll and Heiser [7] showed that, for an embedded system, by running `equake`, `vpr` and `gzip` from the SPEC CPU2000 suite, the microprocessor energy consumption exceeds the consumption due to RAM alone, whereas other software such as `crafty` and `mcf`, from the same suite, exhibit a greater energy consumption from the RAM than from the microprocessor. Minimizing the number of memory access operations is a common energy optimization technique. For example, Intel has introduced, with the E5 Xeon chip (DDIO), the Ethernet network card (NIC), which can load data directly into the microprocessor cache, thus minimizing access to RAM. By avoiding input/output (I/O) operations, the performance, but also the energy consumption of the system, is improved.
An interesting feature of the energy consumption of a code sequence is that, under certain assumptions, the product of its execution time by the energy consumption of the microprocessor possesses convex properties; this rule is called the Energy/Frequency Convexity Rule of program consumption (EFCR). The existence of such a convexity property is based on both experimental and theoretical works and also relies on a vast survey of the state of the art. This rule thus shows that there is a clock frequency f_{opt} for the execution of each code sequence that minimizes the energy consumption of said code sequence. Under certain conditions, this optimum clock frequency, which reduces energy consumption, lies between the minimum f_{min} and maximum f_{max} clock frequencies of operation of the microprocessor. The choice of the clock frequency results thus in a compromise between performance in terms of execution time and the need in energy savings.

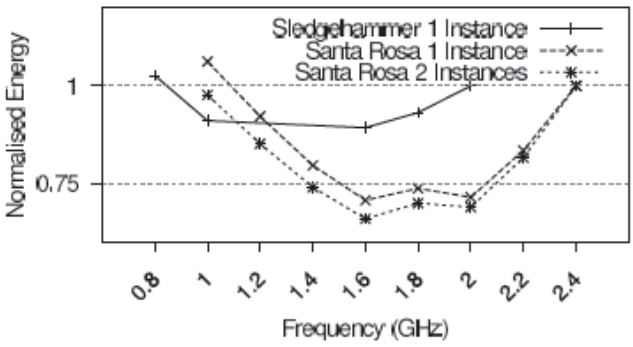
3.1. Related work on the convexity of the energy/frequency relationship

Efforts by researchers as well as engineers in order to optimize energy management on a system have mostly focused on the hardware level. Energy management at the level of software applications is much more recent and is linked to the access, via the operating system, to various key parameters such as the temperature or the energy consumed by the integrated circuit or the card on the printed circuit board. At this level, it was initially a question of putting to sleep, or even switching off, the components that have nothing to perform, which can be deduced from the fact that queues at the components' inputs are empty [1]. You *et al* [49] have observed an average reduction in energy consumption of 23%.
A large number of authors have observed, with some exceptions, that the energy consumed by a microprocessor is not a linear function of its clock frequency but exhibits a convex curve and, therefore, an optimum. This convexity is a feature that has been observed in the literature (see Figure 3) but to the best of our knowledge, is not yet in usage in operating systems. On the analytical front, however, related work is rather rare and fairly approximate. Indeed, authors rarely go so far as modeling by considering the fundamental laws of physics. The published curves are thus mostly obtained by experimentation or even simulation. For example, Senn *et al* [37] and then Austin and Wright [2] present a heuristic model. Other authors ([22], [21]) discuss the consequences and possible exploitation of this convex behavior, but remain at a theoretical level. Yet other authors provide measurements of energy consumption under DVFS, but without exhibiting convexity (see [41], [39]. [2], or [42, 43]), which showed that, for specific applications and over a given frequency window, the convex behavior of energy as a function of frequency was not observable. Yet, Senn *et al* [37] showed a convex curve of energy as a function of frequency, and established a simplified model for their Texas Instruments C55, C62, C64 and C67 platforms. Figure 3 shows various graphs taken from the publications cited above.

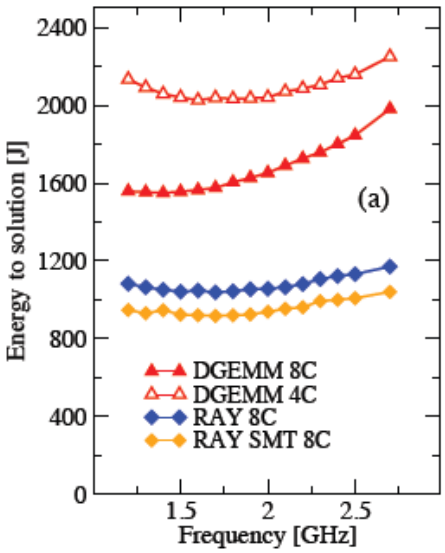
There are other theoretical and experimental works on the convex relation between frequency and dissipated energy. For example, Yuki and Rajopadhye [50] study the energy consumption of high-performance computers, as well as Austin and Wright [2], who examine the energy consumption on Cray CX30; the authors note that the value of the minimum is specific to the application. Cho and Chang [9] measure an optimum for a microprocessor-memory set; they arrive at a relatively complex model, but show the feasibility of using optimal frequencies. A more complete state of the art can be found in [12].



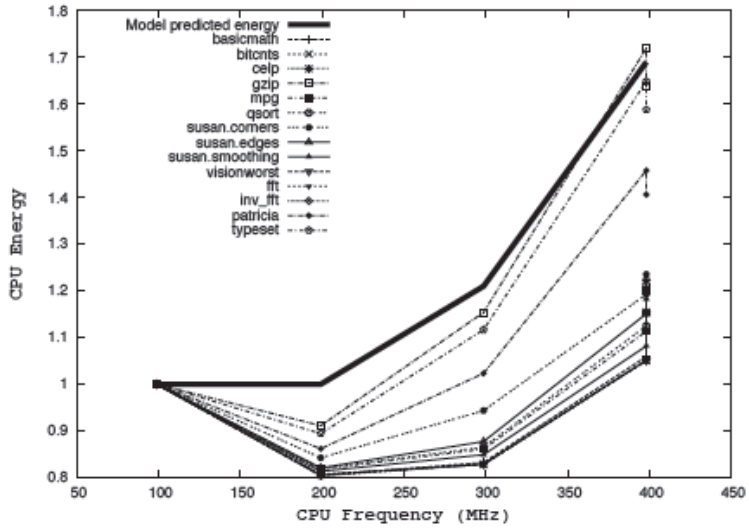
(a) [18]



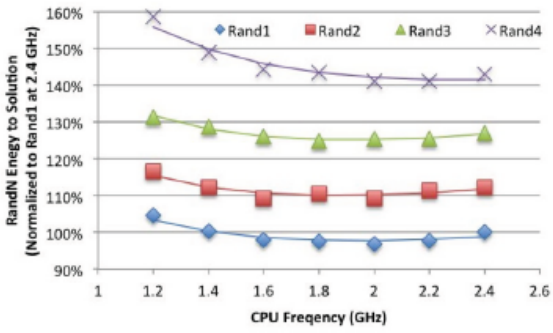
(b) [29]



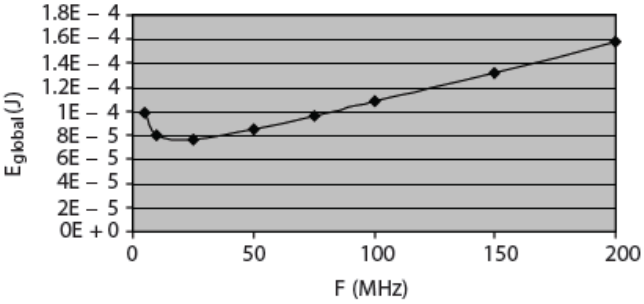
(c) [22]



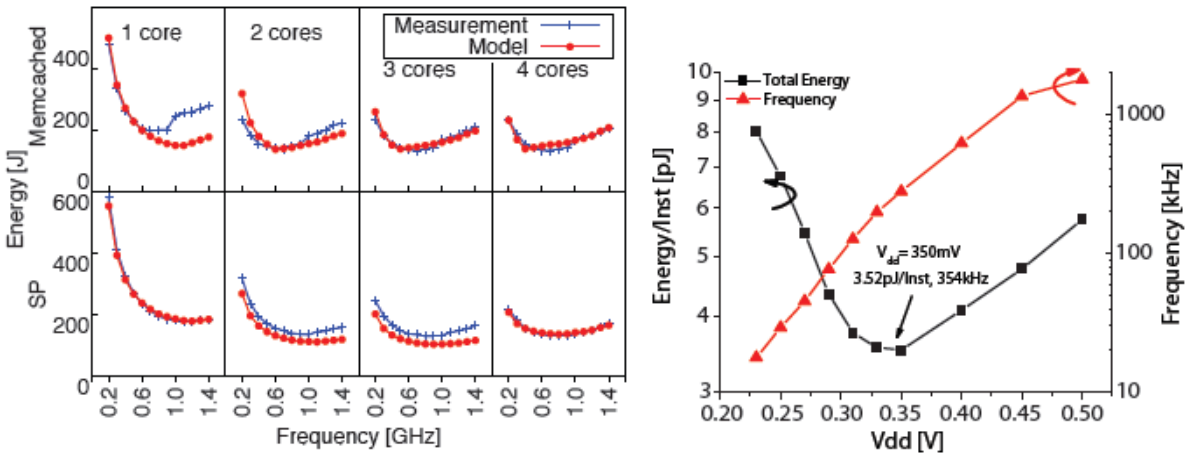
(d) [42]



(e) [2]



(f) [37]



(g) [Tudor…13] *(h) [38]*

Figure 3: Some graphs on the energy/frequency curve found in different independent publications. We observe the convexity of the energy/frequency curve for most of them.

3.2. Analytical model of EFCR

Analytically, we obtain the system energy E_{sys} as a function of the clock frequency f and voltage V of the processor. This function depends on a certain number of parameters related not only to the architecture of the processor and the rest of the system as can be found in the literature, but also to software-dependent issues: (1) the program directly linked to the application (via the number of clock cycles of its execution, cc_b) and (2) other programs or services that can run on the same processor as the application and that behave as thieve clock cycles (f_k), typical on a multitasking operating system. EFCR is expressed by the following equation:

$$E_{\mathrm{sys}} = \left((1+\gamma V)\xi f V^2 + P_{\mathrm{back}}\right) \cdot cc_{\mathrm{b}} \left(\frac{1}{f - f_{\mathrm{k}}} + \beta\right), \qquad \text{(EFCR)}$$

where γ is a parameter associated with the leakage currents, ξ characterizes the CPU power, P_{back} is the power needed outside the CPU (screen, sensors, ...), cc_b is the number of clock cycles of the application, f_k corresponds to a frequency related to the cycles stolen by the various overheads, in particular of the operating system, and β is related to the periods of waiting time when the central processing unit is not active. This formula (the physical parameters being all in positive) justifies the convexity of the energy curves as a function of the frequency. It also makes it possible to establish that the optimal frequency is independent of the size of the application (linked to cc_b), which we have already observed experimentally in Figure 4.
A sensitivity analysis of the parameters was carried out ([12, 14]). The P_{back} background power is the parameter that has the greatest influence on the optimum clock frequency, which minimizes the power usage of the system. In general, f_{opt} can be used, i.e., $f_{min} < f_{opt} < f_{max}$, if the power requirements of the microprocessor are smaller than P_{back}. For some types of applications, however, f_{opt} is independent of P_{back}, for example for the processing of repetitive tasks within computer systems. The number of clock cycle thieves f_k also affects f_{opt} significantly, in the sense that f_{opt} increases when fewer clock cycles are available for the actual calculation.

3.3. Experimental validations of EFCR

Data acquisition campaigns were obtained from the Bristol Energy Efficiency Benchmark Suite (BEEBS) and the Golden-Rader bit-reverse algorithm. These programs were run on several different platforms, two of them made up of multimedia SoCs. The power profiles used were recorded on the same SoCs, one of which is composed of a Cortex A9 and the other of a dual microprocessor Cortex A7 and A15. The results suggest that the energy consumed per input element is strongly correlated with the clock frequency of the microprocessor and, even more interestingly, that the corresponding curve presents a clear minimum on a frequency window specific to the computer system and the program under study (Figure 4). An analytical model of this behavior is also motivated, which fits well with the data presented. A sensitivity analysis of the parameters has also been carried out in order to evaluate the influence of the parameters on the optimal frequency ([12, 14]). It is also shown that the optimum frequency increases when the needs in power of the system, excluding microprocessors, increase. The presence of clock cycles dedicated to the operating system also increases the optimum frequency.

It is observed that the optimum frequency as derived from the theoretical framework presented here is, however, independent of the number of instructions to be executed, depending yet on the size N of the input data.

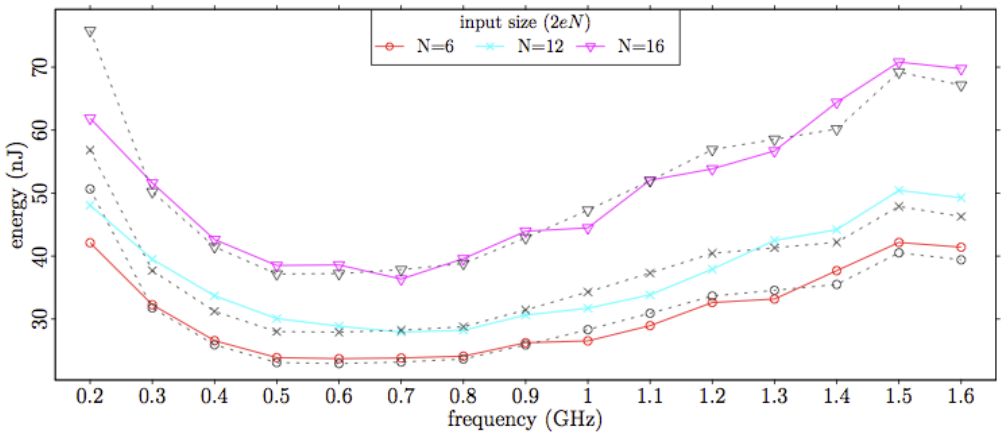


Figure 4: Experimental and theoretical energy consumption data of a program running on ARM Cortex A9 microprocessors. The energy consumption for various characteristic points with different input sizes is shown for the Rader algorithm as well as the BEEBS benchmark. The solid lines represent the measured data, while the dashed lines represent the energy/frequency computed curve (EFCR).

This experiment has been recently further validated by a more accurate measurement campaign using a TI AM572x platform plugged to Labview equipment [46] (see Figure 5).

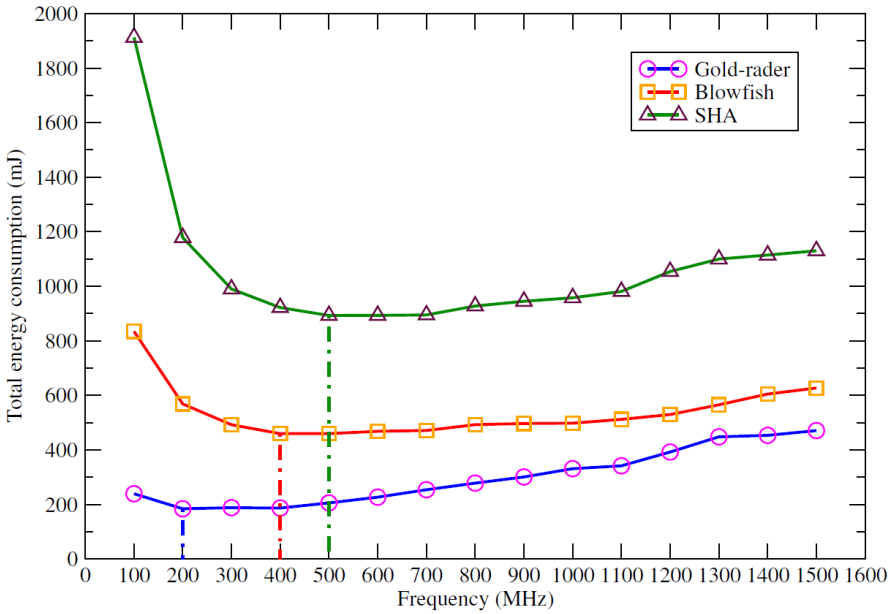


Figure 5: Energy consumption of three different programs running on a TI AM572x platform showing different profiles with different f_{opt}.

4. Future work and conclusion

The accuracy of the measurement protocol of the power-temperature relationship in embedded systems could be improved, mainly by using several more accurate temperature probes. This is an arduous task because the temperature probes on the circuits have low resolution and can lead to high error rates. External temperature measuring devices, such as infrared sensors, can provide improved accuracy but must be used in a controlled environment and preferably with the circuit under study. The ambient temperature and the temperature of the circuit must also be controlled so as to avoid the effects of temperature hysteresis, which may interfere with the

power-temperature correlation. In particular, it seems important to us to verify when the assumption of isothermality constitutes a legitimate approximation for our thermal relationships.
From a theoretical point of view, it might be interesting to evaluate the exact impact factor of processes that depend on the temperature, in addition to the leakage current, and that affect the power profile of the microprocessors. Using this knowledge, a more advanced model of the power-temperature relationship for microprocessors could be designed, based on physical principles, in addition to the heuristic power-temperature model presented previously. In addition, the transformation model to account for distant temperature sensor syndrome could be deduced from a more robust theoretical analysis, replacing an approximate polynomial, although, as has been shown, such an effort would probably lead to a very complex mathematical formulation that could constitute an important stumbling block for online uses.
From the point of view of experimental power measurements, more precise data would also be beneficial to refine and validate our models, as we already started to do in [46]. A higher sampling rate would allow for more detailed observation of specific parts of the instruction sequences. This would lead to an estimation of the useful power of the microprocessor with a finer grain, which could be beneficial for obtaining information about the different functional units of the microprocessor, better than the estimation at the level of application that was assumed in [14]. Our work has used traces of power measurements at 4 kHz. High-end data acquisition tools have sampling rates of 100 kHz to 1 GHz, which shows that progress is possible in this direction in spite of higher and more expensive post-measurement data processing requirements. More precise energy profiles can then also lead to more aggressive DVFS optimization, which would yield far greater energy gains than those provided by the interactive frequency governors present in typical Linux distributions.
Finally, the rule of energy/frequency convexity of program consumption has been studied by adopting a theoretical point of view, without taking into account human factors. For HPC systems that must produce results as quickly as possible, human factors are of little value: programs must run as fast as possible. On the other hand, devices designed with rich human-machine interactions (HMI), such as smartphones or tablets, must absolutely take into account the human emotional aspects in the use of the energy/frequency convexity rule. As has been presented in this work, EFCR is a compromise between energy consumption, on the one hand, and execution time, on the other. If an HMI is involved, the user experience will potentially be affected by too great a change in execution time. As the execution time increases, the computer system becomes less reactive and the user experience will deteriorate, non-linearly. However, for applications requiring many human interactions, it has been shown that clock frequency can be reduced without affecting user experience [36]. Understanding the emotional impact of EFCR should therefore be a key factor when designing a system optimized for quality of human experience. User experience can then be seen as an additional constraint to define the optimum clock frequency. This constraint would induce a lower limit on the clock frequency, which would indicate the moment at which the user would no longer tolerate any slowdown of the system.

5. References

[1] M. Abramovici, P. A. Bradley, P. L. Levin, and G. Memmi: Integrated circuit with autonomous power management. *US Patent Applicaton* #20060218424, published in September 2006.
[2] B. Austin and N. J. Wright: Measurement and interpretation of microbenchmark and application energy use on the Cray XC30. In *Proceedings of the 2nd International Workshop on Energy Efficient Supercomputing* (Piscataway, NJ, USA, 2014), E2SC '14, IEEE Press, pp 51–59.
[3] L. Benini, A. Bogliolo, and G. De Micheli: A survey of design techniques for system-level dynamic power management. *IEEE Trans. on very large-scale integration (VLSI) SYSTEMS*, vol. 8, no. 3, June 2000, pp 299–316.
[4] C. Bennett: The Thermodynamics of Computation-A Review; *Int. J. of Theoretical Physics*, vol. 21, n°12, (1982), pp 905-940.
[5] M. K. Berhe: Ergonomic temperature limits for handheld electronic devices. *In ASME InterPACK Conference* (July 2007), vol. 2, pp 1041-1047.
[6] S. Biswas, M. Tiwari, T. Sherwood, L. Theogaraja, and F. Chong: Fighting fire with fire: Modeling the datacenter-scale effects of targeted superlattice thermal management. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on* (June 2011), pp 331–340.
[7] A. Carroll and G. Heiser: An analysis of power consumption in a smartphone. In Proceedings of the USENIX conference on USENIX (Berkeley, CA, USA, 2010).
[8] Y. Cengel and A. Ghajar: Heat and Mass Transfer: Fundamentals and Applications. McGraw-Hill Education, 2010.
[9] Y. Cho and N. Chang: Energy-aware clock-frequency assignment in microprocessors and memory devices for dynamic voltage scaling. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26*, 6 (June 2006), pp1030–1040.
[10] A. Cohen, F. Finkelstein, A. Mendelson, R. Ronen, and D. Rudoy: On estimating optimal performance of

CPU dynamic thermal management. IEEE Computer Architecture Letters 2, 1 (2003), 6.
[11] K. De Vogeleer, G. Memmi, P. Jouvelot, and F. Coelho: The Energy/Frequency Convexity Rule: modeling and experimental validation on mobile devices. In *Proceedings of the 10th Conference on Parallel Processing and Applied Mathematics* (September 2013), Springer Verlag.
[12] K. De Vogeleer, G. Memmi, and P. Jouvelot: Parameter Sensitivity Analysis of the Energy/Frequency Convexity Rule for Application Processors *J. of Sustainable Computing, Informatics and Systems*, Elsevier B.V., Vol. 15, pp16-27, September 2017.
[13] K. De Vogeleer, G. Memmi, P. Jouvelot, and F. Coelho: Modeling the temperature bias of power consumption for nanometer-scale cpus in application processors. In *XIVth International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation* (July 2014), pp 172–180.
[14] K. De Vogeleer: La loi de convexité énergie-fréquence de la consommation des programmes: modélisation, thermosensibilité et applications .*Thèse Télécom ParisTech*, (September 2015).
[15] D. Duarte, G. Geannopoulos, U. Mughal, K. Wong, G. Taylor: Temperature sensor design in a high-volume manufacturing 65nm CMOS digital process. In Custom Integrated Circuits Conference, 2007. CICC '07. IEEE (2007), pp 221–224.
[16] H. Esmaeilzadeh, T. Cao, X. Yang, S. M. Blackburn, K. S. McKinley: Looking back and looking forward: Power, performance, and upheaval. Commun. ACM 55, 7 (July 2012), pp105–114.
[17] H. Esmaeilzadeh, T. Cao, X. Yang, S. M. Blackburn et K. S. McKinley: What is happening to power, performance, and software? IEEE MICRO 32, 3 (March 2012).
[18] X. Fan, C. S. Ellis, A. R. Lebeck: The synergy between power-aware memory systems and processor voltage scaling. In *Proceedings of the Third international conference on Power - Aware Computer Systems* (Berlin, Heidelberg, 2004), Springer-Verlag, pp 164–179.
[19] A. Ferre and J. Figueras: Characterization of leakage power in CMOS technologies. In Electronics, Circuits and Systems, 1998 IEEE International Conference on (1998), vol. 2, pp 185 –188.
[20] D. Ferreira, A. Dey, V. Kostakos: Understanding human-smartphone concerns: A study of battery life. In Pervasive Computing, K. Lyons, J. Hightower, et E. Huang, Eds., LNCS 6696. Springer Berlin Heidelberg, 2011, pp 19–33.
[21] V. W. Freeh, D. K. Lowenthal, F. Pan, N. Kappiah, R. Springer, B. L. Rountree and M. E. Femal: Analyzing the energy-time trade-off in high-performance computing applications. *IEEE Trans. Parallel Distrib. Syst. 18*, 6 (June 2007), pp 835–848.
[22] G. Hager, J. Treibig, J. Habich, and G. Wellein: Exploring performance and power properties of modern multicore chips via simple machine models. *CoRR abs/1208.2908* (2012).
[23] H. Hanson, S. Keckler, S. Ghiasi, K. Rajamani, F. Rawson, and J. Rubio: Thermal response to DVFS: analysis with an Intel Pentium m. In *Low Power Electronics and Design, ACM/IEEE Symposium on* (2007), pp 219–224.
[24] V. Hanumaiah and S. Vrudhula: Reliability-aware thermal management for hard real-time applications on multi-core processors. *In Design, Automation Test in Europe Conference Exhibition (DATE)*, 2011 (March 2011), pp 1–6.
[25] V. Hanumaiah and S. Vrudhula: Temperature-aware DVFS for hard real-time applications on multicore processors. *IEEE Transactions on Computers 61*, 10 (2012), pp 1484–1494.
[26] D. Ikebuchi, N. Seki, Y. Kojima, M. Kamata, L. Zhao, H. Amano, T. Shirai, S. Koyama, T. Hashida, Y. Umahashi, H. Masuda, K. Usami, S. Takeda, H. Nakamura, M. Namiki, and M. Kondo: Geyser-1: a MIPS R3000 CPU core with fine-grained run-time power gating. In *ASP-DAC* (2010), IEEE, pp 369–370.
[27] J. Kong, S. W. Chung, and K. Skadron: Recent thermal management techniques for microprocessors. *ACM Comput. Surv.* 44, 3 (June 2012), 13:1–13:42.
[28] P. Lall, M. Pecht, and E. Hakim: Influence of Temperature on Microelectronics and System Reliability: A Physics of Failure Approach. The electronic packaging series. CRC Press, 1997.
[29] E. Le Sueur and G. Heiser: Dynamic voltage and frequency scaling: the laws of diminishing returns. In *Proc. of the 2010 Int. Conf. on Power aware computing and system* (Berkeley, CA, USA, 2010), HotPower'10, pp 1–8.
[30] W. Liao, J. M. Basile, and L. He: Leakage power modeling and reduction with data retention. *In Proc. of the 2002 IEEE/ACM Int. Conf. on Computer-Aided Design*, New-York, 2002, ACM, pp 714–719.
[31] W. Liao, L. He and K. M. Lepak: Temperature and supply voltage aware performance and power modeling at microarchitecture level. *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys*. 24,7 (Nov. 2006), pp 1042–1053.
[32] R. M. McGowen, C. Poirier, C. Bostak, J. Ignowski, M. Millican, W. Parks, S. Naffziger: Power and temperature control on a 90-nm itanium family processor. *IEEE J. of Solid-State Circuits*, 41,1 (2006), pp 229–237.
[33] F. Mesa-Martinez, M. Brown, J. Nayfach-Battilana, J. Renau: Measuring power and temperature from real processors. In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on* (April 2008), pp 1–5.

[34] F. Rezzi, L. Collamati, M. Costagliol, and M. Cutrupi: Battery management in mobile devices. *In Frequency References, Power Management for SoC, and Smart Wireless Interfaces*, A. Baschirotto, K. A. Makinwa, and P. Harpe, Eds. Springer International Publishing, 2014, pp147–168.
[35] S. Sarangi, G. Ananthanarayanan, and M. Balakrishnan: Lightsim: A leakage aware ultrafast temperature simulator. in *Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific* (Jan. 2014), pp 855–860.
[36] V. Seeker, P. Petoumenos, H. Leather, and B. Franke: Measuring QOE of interactive workloads and characterising frequency governors on mobile devices. *In 2014 IEEE International Symposium on Workload Characterization*, IISWC 2014, Raleigh, NC, USA, 26-28 octobre, 2014, pp 61–70.
[37] E. Senn, J. Laurent, N. Julien, and E. Martin: Softexplorer: Estimating and optimizing the power and energy consumption of a C program for DSP applications. *EURASIP J. Adv. Sig. Proc. 2005*, 16 (2005), pp 2641–2654.
[38] M. Seok, D. Jeon, C. Chakrabarti, D. Blaauw, and D. Sylvester: Pipeline strategy for improving optimal energy efficiency in ultra-low voltage design. In *Proceedings of the 48th Design Automation Conference* (New York, NY, USA, 2011), DAC '11, ACM, pp 990–995.
[39] T. Simunic, L. Benini, and G. De Micheli: Cycle-accurate simulation of energy consumption in embedded systems. In *Design Automation Conference, 1999. Proceedings. 36th* (1999), pp 867–872.
[40] K. Singh, M. Bhadauria, and S.A. McKee: Real-time power estimation and thread scheduling performance counters. *SIGARCH Comput. Archit. News 37*, 2 (July 2009), pp 46–55.
[41] A. Sinha and A. P. Chandrakasan: Jouletrack: a web-based tool for software energy profiling. In *Proceedings of the 38th annual Design Automation Conference* (New York, NY, USA, 2001), DAC '01, ACM, pp 220–225.
[42] D. Snowdon, S. Ruocco, and G. Heiser: Power management and dynamic voltage scaling: Myths and facts. In *2005 WS Power Aware Real-time Comput.* (New Jersey, USA, Sept. 2005).
[43] D. C. Snowdon: *OS-Level Power Management*. PhD thesis, School of Computer Science and Engineering, University of NSW, Sydney 2052, Australie, March 2010.
[44] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif: Full chip leakage estimation considering power supply and temperature variations. In Proceedings of the 2003 international symposium on low power electronics and design (New York, NY, USA, 2003), ISLPED '03, ACM, pp 78–83.
[45] N. Vallina-Rodriguez, and J. Cowcroft; Energy management techniques in modern mobile handsets. *IEEE Communications Surveys & Tutorials*, vol 15, 1 (2013), pp 179–198.
[46] K. Rao Vaddina, F. Brandner, G. Memmi, and P. Jouvelot: Experimental Energy Profiling of Energy-Critical Embedded Applications. In *IEEE SoftCom'17*, Split, Croatia, September 2017.
[47] R. Viswanath, V. Wakharkar, A. Watwe, and V. Lebonheur: Thermal performance challenges from silicon to systems. *Intel Technology Journal*, Q3 (August 2000), pp 1–16.
[48] A. Weissel and F. Bellosa: Dynamic thermal management for distributed systems. In *Proc. of the First Workshop on Temperature-Aware Computer Systems (TACS'04)* (Munich, Germany, June 2004).
[49] Y.-P. You, C. Lee, and J. K. Lee: Compiler analysis and supports for leakage power reduction on microprocessors. In *Proceedings of the 15th international conference on Languages and Compilers for Parallel Computing* (Berlin, Heidelberg, 2005), LCPC'02, Springer-Verlag, pp 45–60.
[50] T. Yuki and S. Rajopadhye: Folklore confirmed: Compiling for speed = compiling for energy. In *Proc. of the 26th International Workshop on Languages and Compilers for Parallel Computing* (2013).
[51] F. Zanini, F. Atienza, C. N. Jones, L. Benini, and G. De Micheli; Online thermal control methods for multiprocessor systems. *ACM Trans. Des. Autom. Electron. Syst*. 18, 1 (Jan. 2013), 6, pp 1–6:26.