In the Proceedings of the 18th Symposium on High Performance Computer Architecture (HPCA 2012).

# Computational Sprinting

Arun Raghavan*, Yixin Luo†, Anuj Chandawalla†,
Marios Papaefthymiou†, Kevin P. Pipe†‡, Thomas F. Wenisch† and Milo M. K. Martin*
*Department of Computer and Information Science, University of Pennsylvania
†Department of Electrical Engineering and Computer Science, University of Michigan
‡Department of Mechanical Engineering, University of Michigan

## Abstract

*Although transistor density continues to increase, voltage scaling has stalled and thus power density is increasing each technology generation. Particularly in mobile devices, which have limited cooling options, these trends lead to a* utilization wall *in which sustained chip performance is limited primarily by power rather than area. However, many mobile applications do not demand sustained performance; rather they comprise short bursts of computation in response to sporadic user activity.*

*To improve responsiveness for such applications, this paper explores activating otherwise powered-down cores for sub-second bursts of intense parallel computation. The approach exploits the concept of* computational sprinting*, in which a chip temporarily exceeds its sustainable thermal power budget to provide instantaneous throughput, after which the chip must return to nominal operation to cool down. To demonstrate the feasibility of this approach, we analyze the thermal and electrical characteristics of a smart-phone-like system that nominally operates a single core (∼1W peak), but can sprint with up to 16 cores for hundreds of milliseconds. We describe a thermal design that incorporates phase-change materials to provide thermal capacitance to enable such sprints. We analyze image recognition kernels to show that parallel sprinting has the potential to achieve the task response time of a 16W chip within the thermal constraints of a 1W mobile platform.*

## 1. Introduction

Future technology generations are predicted to provide more transistors, but unfortunately Dennard scaling—the concomitant reduction of CMOS threshold and supply voltages—has stalled. Hence, power density is increasing with each new process generation on a trajectory that outstrips improvements in the ability to dissipate heat. The confluence of these trends has led to a phenomenon referred to as *the utilization wall* or *dark silicon*: a chip sized for the economic manufacturability sweet-spot will have far more transistors than can be used on a sustained basis [5, 9, 13, 15, 18, 30, 43]. This phenomenon is perhaps most acute in mobile devices, such as netbooks, tablets, and smart phones, which, despite relatively large chip area, have thermal budgets that are constrained by the poor heat dissipation of passive convection.

Processors today (and their heat sinks and energy delivery systems) are designed primarily for *sustained performance*. Although the focus on sustained performance is the right design choice for some applications (for example, batch-mode high-performance computing), many workloads are interactive in nature and thus demand *responsiveness*—how long does the user have to wait after initiating a command? In such settings, responsiveness may be more important than sustained performance [14]. In this work, we pose the question, "What would a system look like if designed to provide responsiveness during bursts rather than with a singular focus on sustained performance?"

Many current and emerging interactive applications are characterized by short bursts of intense computation punctuated by long idle periods waiting for user input [3, 45], especially in mobile settings [40]. For example, consider a camera-based visual search application in which a mobile device performs feature extraction on a high-resolution image and transmits a compact feature vector to the cloud for recognition. The limited compute capabilities of today's mobile devices combined with the responsiveness demands of the user (receiving a query response within a few seconds) preclude the use of the most advanced compute-intensive extraction algorithms [11, 16]. Search result quality could be improved if the mobile device supported an intense computational burst to perform better feature extraction. Other mobile applications that fit the pattern of bursty usage include image processing and computational photography tasks (such as panoramic stitching and image deblurring/noise reduction), navigation route planning, and natural language processing (speech recognition and translation).

To improve application responsiveness in today's thermal-limited context, this paper explores temporarily exceeding the chip's sustainable thermal budget (a.k.a. thermal design power or TDP). Even though the chip is generating heat faster than the system dissipates it, temperature does not spike instantaneously. Instead, *thermal capacitance* in the system causes the temperature to

rise over an extended—albeit still short—time interval. In fact, some systems today will exceed TDP by a small margin over tens of seconds by boosting voltage and clock frequency (such as Intel's Turbo Boost 2.0 technology [36]), but this approach is energy inefficient (as DVFS incurs a quadratic power cost for a linear frequency boost), which is especially unappealing for battery-powered devices.

This paper investigates *computational sprinting*, in which a chip improves application responsiveness by temporarily exceeding its sustainable thermal budget by an order of magnitude or more for sub-second durations, providing a brief but intense burst of computation. As today's systems lack sufficient thermal capacitance close to the chip to support such intense sprints, one aspect of sprinting we explore is designing the chip's thermal packaging to incorporate more thermal capacitance rather than only the conventional objective of maximizing thermal conductivity to the ambient environment (which in turn determines TDP).

Motivated by the parallelism potential of dark silicon, we explore *parallel sprinting* in which many "dark silicon cores" are activated for up to a second, resulting in an intense sprint with the potential to provide an order of magnitude improvement in responsiveness (*e.g.*, accelerating a five-second task to half a second). Parallel sprinting can achieve energy efficiency similar to that of sequential execution because power grows roughly linearly with the number of active cores, a more efficient approach than voltage boosting.

Sprinting improves responsiveness; it does not, however, improve sustained performance. Once sprinting capacity is exhausted, the chip must cool in non-sprint mode before it can sprint again. Thus, sustained performance remains limited by the sustainable TDP. In essence, sprinting shifts TDP budget from future moments of low utilization to compress the time of the present computation.

Computational sprinting raises design challenges spanning from packaging (solving thermal and electrical challenges) to end-user studies (how much do end users value the increase in responsiveness and tolerate the delay between sprints). In this paper, we focus on the technical feasibility of intense parallel sprinting. We identify and address challenges in the thermal, electrical, and software/runtime aspects of sprinting. Specifically, we explore increasing thermal capacitance by integrating phase-change materials close to the chip (Section 4), employing gradual core activation to avoid sprint-induced voltage instability (Section 5), augmenting batteries with ultracapacitors to provide sufficient electrical current during sprints (Section 6), and using a software runtime for activating and deactivating parallel sprints (Section 7).

We perform an initial evaluation of parallel computational sprinting via a combination of thermal modeling, SPICE, and detailed architectural simulations using a suite of image processing and vision-focused application kernels inspired by camera-based search. We consider a design for a smart-phone-like system architecture built from a conventional cache-coherent many-core chip with in-order throughput-optimized cores. The system nominally operates only a single core (1W peak), but can sprint with up to 16 cores for up to a second while still operating within the engineering constraints of a high-end smart phone (*e.g.*, heat dissipation only by passive convection). Broadly, our evaluation demonstrates that parallel sprinting can improve responsiveness by an average of 10.2× for sprints of up to a second.

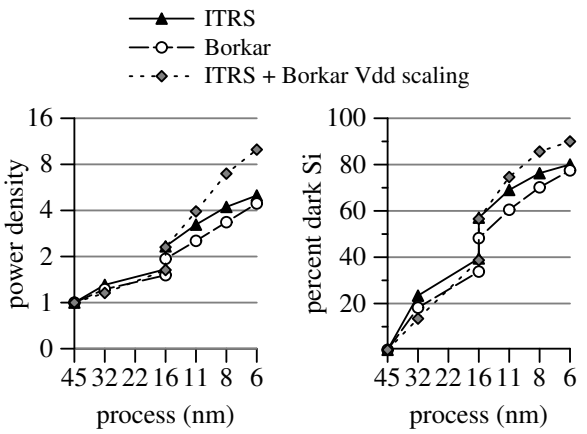
## 2. Dark Silicon and Mobile Chip Trends

An increasing chorus of academic and industry veterans warns that chip power density scaling trends are not sustainable and that only a fraction of the typical area of today's chip can be powered in future technologies, giving rise to the prediction of "dark silicon" (*i.e.*, that much of a chip must be powered off at any time) [5, 9, 13, 15, 18, 30, 43]. Mike Muller (CTO of ARM) has spoken publicly about the dark silicon problem, predicting that by 2019 only 9% of the transistors on a chip can be active at any time [30].

This prediction arises because device density is increasing much faster than per-device capacitance is decreasing. For example, Borkar *et al.* [5] predict a 25% capacitance reduction versus 75% density increase per process generation. Historically, constant power density has been maintained by scaling down supply voltage, but technology assessments indicate drastically slowed scaling in future nodes [1, 5]. Thus, even if clock frequencies remain flat, the combined trends will result in power density increases; attempts to increase frequency exacerbate this effect. Figure 1 shows power density and dark silicon area projections for a fixed-area chip [1, 5].

Until recently, the challenge of increased power density was passed along to package designers by specifying higher TDP for chips, resulting in CPUs and GPUs that dissipate 100+ Watts. However, both high-end chips and mobile chips are reaching the thermal limits of active and passive cooling, respectively. Without innovation, either physical chip size must decrease each generation (thereby ending the decades of benefit the industry has reaped from Moore's Law) or the *active fraction* of the chip must be decreased.

This trend is particularly acute for emerging smart phone and tablet devices, which are already exhibiting several symptoms of power limitations and significant inactive silicon. In fact, mobile chips have already embraced heterogeneity and special-purpose accelerators,




(a) Power density (b) Percent dark silicon

**Figure 1. Power density and dark silicon trends.** Data from Borkar [4, 5, 13], ITRS [1], and ITRS with Borkar's more pessimistic voltage scaling assumptions [4].

two of the commonly suggested approaches for mitigating the dark silicon problem [43]. For example, examining the die photo of the Apple A5 chip shows that less than half the die area is used for the CPUs, GPU, and caches; much of the rest of the die is used for custom accelerators (*e.g.*, for energy-efficient multimedia playback). Moreover, mobile chips already employ sophisticated power management to dynamically limit maximum power dissipation (*i.e.*, only a subset of units may be concurrently active without exceeding TDP).

Another sign of the dark silicon problem in mobile chips is their much lower per-area power densities than desktop/server chips. Although limited to just a few watts due to both battery life and thermal constraints, today's mobile phone/tablet chips have substantial die area (NVIDIA's Tegra 2 is 49 $mm^2$, Apple's A4 is 53 $mm^2$, and Apple's A5 is 122 $mm^2$). In comparison, Intel's "Sandy Bridge" Core i7 chip is 149 $mm^2$ for a dual-core chip (216 $mm^2$ for a quad-core chip) with a TDP ranging from 17W at 1.8 Ghz to 65W at 3.3 Ghz. Thus, although the phone/tablet chips have 3× less area, their TDP is lower by an order of magnitude or more.

## 3. Computational Sprinting Overview

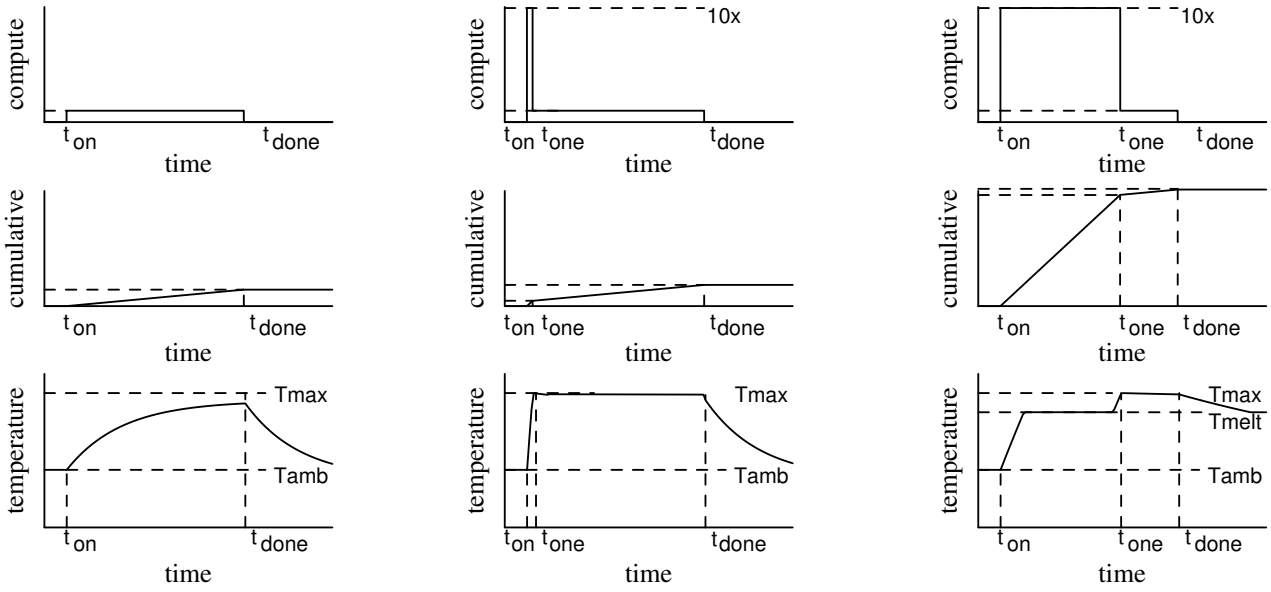
Rather than abandon the benefits of transistor density scaling, we instead make a case for embracing dark silicon, maintaining current chip sizes, and leveraging these excess transistors *transiently*, when performance really counts (*i.e.*, when the user is waiting).

For concreteness, we choose to target an order of magnitude or larger improvement in responsiveness (*i.e.*, completing a five-second computational task in less than half a second) by pursuing a design that can sprint with 16 1W cores in a system that can sustain operation of only a single 1W core. We therefore assume that activating these 16 cores will exceed the system's TDP by 16×. We select one second for the maximum sprint duration as response times slower than this threshold typically degrade the user experience. The scaling and mobile platform trends discussed above suggest that a future chip with the same die area as current mobile chips (and thus presumably roughly similar die cost) will have enough dark silicon to economically support these additional cores for use during sprinting.

To enable such intense sprints, we must address several design challenges. We begin by describing the high-level thermal response of both non-sprinting and sprinting systems. Figure 2(a) shows the thermal response of a conventional system starting from idle. The single core becomes active at time $t_{on}$, causing system temperature to rise asymptotically toward $T_{max}$. The system computes until time $t_{done}$, at which time the core becomes idle and the temperature begins to fall asymptotically toward $T_{ambient}$. For a given $T_{ambient}$ and $T_{max}$, the maximum sustainable thermal power is determined by total thermal resistance of the package and heatsink (but is independent of the thermal capacitance). In contrast, the rate at which the temperature rises (and falls) is determined by both the thermal resistance and capacitance.

Sprinting mode operation is shown in Figure 2(b). In this example, the system is initially idle and the system temperature matches that of the ambient environment ($T_{ambient}$). At time $t_{on}$, an external event (*e.g.*, user input) triggers demand for a burst of computation, and it initiates a parallel sprint by activating all cores. As the heat generation is greater than in sustained operation, the chip temperature rises more quickly. In this example, the temperature reaches $T_{max}$, and thus the computation exceeded the maximum sprint duration of the system. When the temperature reaches $T_{max}$, the system terminates the sprint by disabling all but one core (at time $t_{one}$); any remaining work is completed by this core. During this interval (from time $t_{one}$ to $t_{done}$), because the thermal system is designed to sustain the operation of a single core, temperature remains stable near $T_{max}$. When the computation is done ($t_{done}$), all cores become idle, so the system begins to cool.

Increasing sprint duration requires increasing the thermal capacitance of the system. One way to increase the thermal capacitance is by placing a block of phase change material close to the die (as will be described in Section 4). Adding such material increases the thermal capacitance both by its *specific heat* (the amount of energy to change the temperature of a gram of the material by one degree) and—more importantly—its *latent heat of fusion* (the amount of energy to melt a gram of the material). Figure 2(c) shows the same computation in




(a) Sustained execution (b) Sprint execution (c) Augmented sprint

**Figure 2. Sprinting operation.** Cores active (top row), cumulative computation (middle row) and temperature (bottom row) over time for three execution modes: (a) sustained, (b) sprint, and (c) sprint augmented with phase change material.

sprint mode on such an augmented system. The temperature rises as before, but when the temperature reaches the melting point of the material ($T_{melt}$), the extra thermal energy injected into the system is absorbed by the melting process, allowing the system to continue to execute for a period *without* increasing the temperature. Only once the material has fully melted does the temperature begin to rise again. Similarly, when the system cools, its temperature is constant as the material returns to its solid phase. Overall, in this example the additional thermal capacitance allows the system to perform significantly more computation during the sprint interval.

## 4. Thermal Design

As shown by the previous example, the thermal design is a key enabler. Conventional semiconductor heat sinks are designed primarily to maximize heat conduction for steady-state operation at peak power; their key thermal criterion is the TDP, which indicates the maximum steady-state power that can be dissipated through thermal management while keeping the junction below its maximum safe temperature ($T_{jmax}$). The bursty nature of heat generation under sprinting, however, suggests an architecture incorporating and optimizing thermal storage—characterized by thermal *capacitance*, rather than thermal conductivity alone.

Although exploiting thermal capacitance is not new (for example, prior work has used large thermal capacitance to allow a portable computer to exceed its nominal TDP by up to a factor of four for an hour using a half kilogram aluminum plate [8]), such published work has generally focused on using thermal capacitance over minute to hour time scales [19, 22, 38]. In sprinting, the key challenge is increasing the thermal capacitance over short timescales, which in large part determines the total heat (joules) that can be expended during a sprint. In particular, we seek a design to sprint with 16 1W-cores for up to a second, which requires sufficient thermal capacitance and heat spreading to absorb 16 joules over this time period. Although there is sufficient aggregate thermal capacitance for this small amount of heat in a typical mobile device (*e.g.*, in the case), low thermal conductivity and large distances from the die lead to long thermal time constants (several minutes [29]), and the system therefore cannot respond quickly enough to store heat during sprints. Hence, we explore incorporating thermal capacitance that is in close proximity to the die, under the tight volumetric and weight constraints of a mobile phone.

### 4.1. Heat Storage Using Solid Materials

The most straightforward approach to increase the thermal capacitance available for sprinting is to place a large (relative to the die) piece of copper or aluminum in close proximity to the die. For example, copper has a volumetric heat capacity of 3.45 J/cm$^3$ K, so absorbing 16 joules with a 7.2 mm thick block of copper (or a 10.3 mm thick block of aluminum, which has a volumetric heat capacity of 2.42 J/cm$^3$ K) over a 64 mm$^2$ die will result in a temperature rise of 10°C. There are

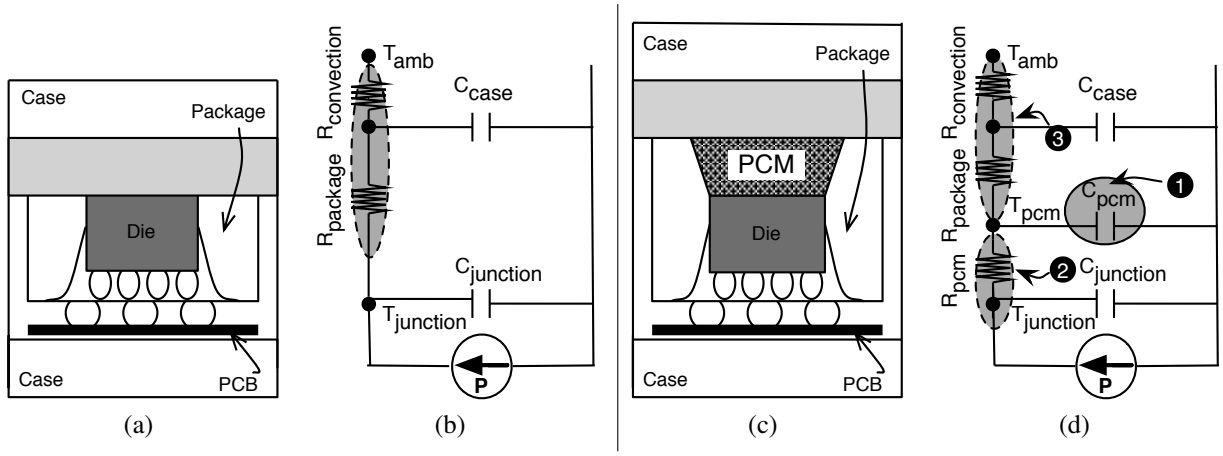



**Figure 3.** The thermal components of a mobile system (a) and its thermal-equivalent circuit model (b). In (c) and (d), the system is augmented with a block of phase change material (PCM). The amount of computation possible during a sprint is primarily determined by the thermal capacity of the PCM (❶). The maximum sprint power is determined by the thermal resistance into the PCM (❷), whereas the sustained power is determined by the total resistance to ambient (❷ + ❸). The thermal resistance from the PCM to ambient (❸) determines how quickly the system cools after a sprint.

two potential drawbacks, however, that might preclude such an approach for sprinting: (1) when the system initiates a sprint after operating a single core (at TDP) for an extended period, the metal temperature will already be elevated, limiting the potential headroom available for sprinting, and (2) the thermal resistance within the (comparatively thick) metal limits the rate at which heat can be absorbed, and thus could limit sprint intensity.

### 4.2. Heat Storage Using Phase Change

An alternative source of thermal capacitance is latent heat in a phase change material. Latent heat refers to the heat required to transition a material from one phase to another (*e.g.*, melting a solid). During a phase transition, the material's temperature remains constant. PCMs with thermal transients of tens of minutes to hours have previously been proposed for thermal management of mobile phones [41] and other portable electronic devices [2].

A wide range of nonflammable, non-corrosive PCMs exist that melt at a relatively low temperature. Icosane (candle wax) for example, has a melting point of 36.8°C and a large latent heat of 241 J/g [2]. Encapsulated solid-liquid PCMs and solid-solid organic PCMs offer reduced complexity as the fluid phase is either not present or is self-contained [44]. In our context, if we assume a PCM with a latent heat of 100 J/g and density of 1 g/cm$^3$, about 150 milligrams of PCM (2.3mm thick block of PCM in contact with a 64mm$^2$ die) can absorb approximately 16 J of heat, which is enough to allow 16 cores to operate at 1W each for 1s at a constant temperature.

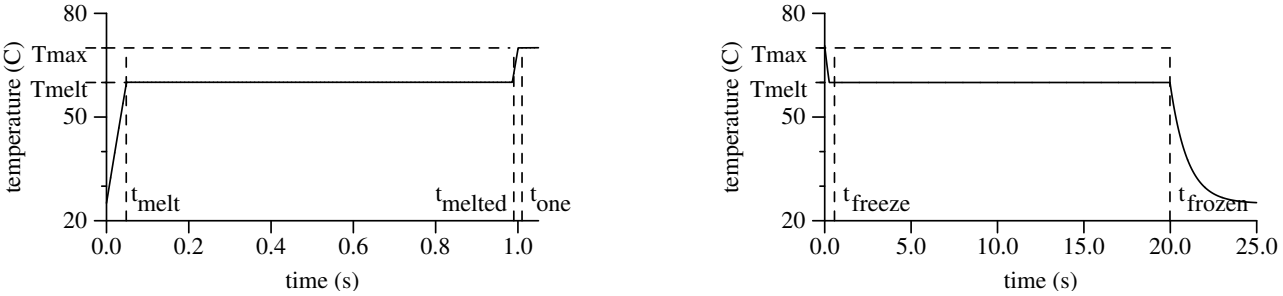
Because most PCMs have low thermal conductivity, a heat spreading network must be integrated to achieve the high rates of heat transfer required for sprinting. Previous work has examined the use of metal or diamond microchannels to achieve PCM-based heat sinking on a timescale of 100 $\mu$s [17], and fiber mesh carriers [10] could potentially be coated with or composed of copper to improve heat transfer. In addition to enhancing thermal conductivity, such an integrated mesh could improve the mechanical robustness of the PCM to avoid wearout effects such as the formation of cracks or voids that might compromise thermal conductivity [10].

### 4.3. Heat Flux Considerations

Although the peak heat flux (25W/cm$^2$) and rate of temperature increase that we envision is high for a mobile device, it is below the typical range for high-end processors [31], which has two important implications. First, the mechanical stresses incurred due to heating during a sprint are not extraordinary and should not fundamentally curtail the lifetime of a sprint-enabled chip. Second, the required thermal conductance between the junction and PCM falls within the range of conventional thermal interface materials (TIMs), and is therefore not expected to limit sprint intensity. Although such a peak heat flux is perhaps not a fundamental barrier to feasibility, it could necessitate a more expensive package. Integrating the PCM into the package and placing it close to the die limits the greater heat flux to only the TIM (since heat can be released from the PCM over a longer time period between sprints), thus ameliorating thermal demands on the package as a whole.

### 4.4. Thermal Modeling

To study the utility of a PCM for sprinting, we use the thermal models shown in Figure 3. We later couple these same models with our detailed architectural simulations (in Section 8). Figure 3(a) and Figure 3(b) illustrate




(a) Sprint initiation

(b) Post-sprint cooldown

**Figure 4.** Transient behavior of initiation and termination of sprinting

the thermal network of a conventional system in which heat flows from the die junction to the ambient through parallel paths that include a number of components. We base our model, in particular the values we use for the thermal resistances and capacitances of various components (*e.g.*, printed circuit board, battery, case), on a physically-validated model of a mobile phone [29].

Figure 3(c) and Figure 3(d) illustrate the thermal network of a PCM-based system. As discussed above, the PCM is placed close to the die. The melting point of the PCM is chosen to be sufficiently high (*e.g.*, 60°C) that it melts during sprints but not during sustained single-core operation (and, conversely, the sustained single-core power budget must be selected to limit junction temperature at most to just below the melting point of the PCM).

### 4.5. Thermal Modeling Results

Figure 4(a) illustrates the transient thermal behavior of a 16W sprint on a 1W TDP system under the thermal model and PCM assumptions described above. Junction temperature initially rises rapidly, then plateaus for 0.95s during the phase change, subsequently rising to the maximum safe temperature (set at 70°C for these simulations). Factoring in all sources of thermal capacitance, the sprint can be sustained for a little over 1s.

Between sprints, the PCM returns to its original phase as the system cools back down to the ambient temperature. Approximate cooldown duration can be calculated by multiplying the duration of the sprint by the ratio of sprint power and nominal TDP. Figure 4(b) shows the cooldown behavior based on our thermal model. The exponential nature of heat transfer results in the junction temperature being close to ambient after about 24s. The cooldown period is governed primarily by the amount of thermal capacitance, the thermal resistances between the PCM and the ambient (see Figure 3), and to a lesser degree by the melting point of the PCM. The higher the melting point of the PCM, the larger the temperature gradient between the PCM and ambient, which accelerates cooling.

## 5. Power Grid and Core Activation

In addition to thermal challenges discussed above, sprinting also introduces electrical challenges in its on-chip power distribution grid. When transitioning into or out of sprint mode, the on-chip voltage rails must remain within specified tolerance levels to preserve state and prevent timing errors despite the massive in-rush current when activating many cores [24]. However, the chip must be able to activate cores quickly enough to minimize the delay before useful parallel computation can occur during sprints. Whereas prior work has examined activation schedules for large blocks within monolithic cores [21], to our knowledge, minimum activation time has not been studied in the context of manycore processors.

### 5.1. Power Distribution Network Model

We model the power grid of a sprint-enabled processor using a network of RLC components (Figure 5). This model encompasses the supply regulator as well as board, package, and on-chip interconnect. We assume an ideal voltage regulator, supplying a steady 1.2V between the two rails. Wire resistance and inductance for ground and power lines are modeled separately at the board, package, and chip level. We include decoupling capacitance at the interfaces of successive levels to reduce voltage bounce. We assume cores are powered via a shared yet aggressively gated power grid structure [25], which allows for the selective control of power supplied to individual cores. To capture the separate ground and supply lines of each core, the impedance of the package and the on-chip interconnect are modeled as distributed networks. The power-gated cores are modeled as current sources. We model the shared interconnect of the power grid between adjacent cores as a grid of in-series resistance and inductance components. This grid is connected to a distributed RLC model of the package. We obtain the parameter values for the various components of the RLC model from prior work [35].

### 5.2. Abrupt Activation

Activating all cores at the same time causes a current change (dI/dt) that has a detrimental effect on power

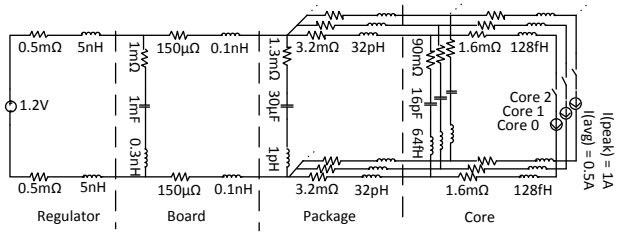



**Figure 5.** RLC power network model

supply integrity. Fluctuations in power and ground may substantially exceed acceptable tolerance levels (typically 1-2%), as shown in Figure 6(a). The graph shows SPICE simulation results of the power grid voltage when all cores are activated within 1ns. The results indicate that voltage fluctuation exceeds 2% of nominal supply voltage, which is likely to result in incorrect operation. Specifically, supply voltage bounces down to 1.171V, which is 97.5% of nominal. Furthermore, it takes 2.53$\mu$s for the supply voltage to come within 2% of its settling voltage.

### 5.3. Gradual Activation

With a sufficiently gradual activation schedule, supply and ground bounce can remain within tolerance at all times during activation. Figure 6(b) and Figure 6(c) show the results of a gradual uniform linear activation schedule for the cores over a ramp of 1.28$\mu$s and 128$\mu$s, respectively. With a 1.28$\mu$s ramp, the chip fails to meet a 2% tolerance on supply fluctuation. However, when the ramp is slowed down to 128$\mu$s, voltage fluctuations remain within tolerance. Notice that the supply voltage settles at a value that is approximately 10mV lower than the nominal supply voltage of 1.2V. This reduction in supply voltage is due to the resistive drop of the current draw on the power distribution network.

Overall, although we find that activating all the cores abruptly is infeasible, when activation is spread over a 128$\mu$s ramp the voltage fluctuations are within tolerance. As this interval is much smaller than our target sprint duration, the impact on attainable parallel speedup is negligible.

## 6. Power Source

Conventional smart phone batteries and associated voltage regulators are designed to sustain currents of at most a few amps—ample for a 1W peak-power system. However, our goal of 16× sprinting calls for far higher power—up to 16W for up to a second. A typical battery found in a mobile phone provides some headroom for sprinting, *e.g.*, a representative Li-Ion battery can provide bursts of 10W (2.7A at 3.7V); higher currents are typically precluded by internal thermal constraints [6]. Such a battery would limit the sprint intensity (*e.g.*, to fewer than ten 1W cores). In contrast, laptop-class batteries can provide sufficient current for sprinting, but they do not meet the form-factor constraints of smart phones.

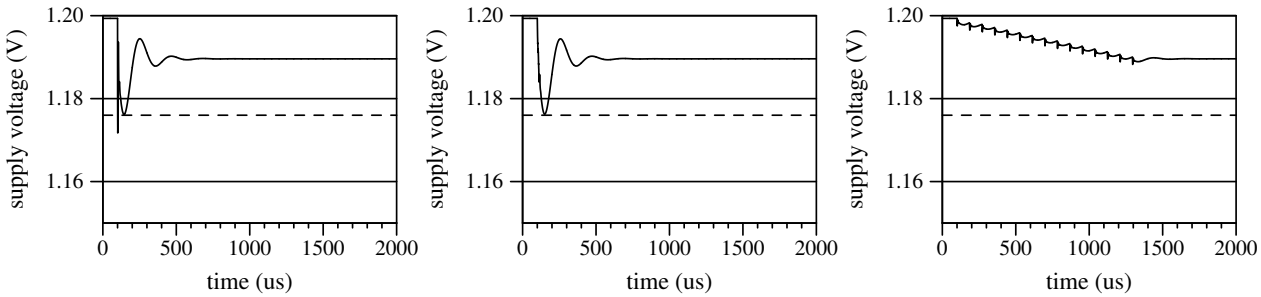
Alternatively, high-discharge Li Polymer battery technologies that were specifically designed to provide large currents (*e.g.*, for applications such as power tools and electric vehicles) can easily meet the current demands of 16 1W cores. For example, a representative 51g Li Polymer battery, such as Dualsky GT 850 2s, can supply 43A at 7V. Moreover, improving the energy density of high-discharge-rate battery cells is an active research area with promising results from initial prototypes [23].

The high discharge rate of ultra-capacitors offers another approach for meeting peak current demands of sprinting. Prior work has explored hybrid battery–ultracapacitor power sources to support burst currents in larger electronic systems [34] and to improve battery efficiency [32, 33]. The latest ultra-capacitors have the potential to meet our requirements for energy capacity, peak current, and form factor with negligible energy losses due to leakage. For example, a 25F NESSCAP ultra-capacitor, weighing 6.5g, can store 182 joules and provide a peak current of 20A at a rated voltage of 2.7V with a total leakage current below 0.1mA. As with batteries, ultracapacitor technology is also an area of active research [37].

A further challenge lies in delivering the necessary peak currents from the off-chip power source over the chip pins. Whereas 100A peak currents are commonly sustained in desktop and server package/socket designs, 16A peak currents exceed the norm for mobile devices. Providing such peak currents will likely require more power and ground pins. Today's phone and tablet processors have both smaller packages and narrower pin pitches than desktop chips (for example, Apple's A4 has a 14mm-square package, 0.5mm pitch and 531 pins; the Qualcomm MSM8660 has a 14mm-square package with a 0.4mm pitch and 976 pins). If a pair of power/ground pins provides a peak current of 100mA, 16A at 1V requires 320 pins, likely increasing the cost of the package. On-chip voltage regulators [26] combined with higher pin voltages might also assist in satisfying peak current demands.

## 7. Sprint Activation and Deactivation

Software activates sprinting whenever there is sufficient thread-level parallelism in the application. When there are more active threads than cores, the operating system or runtime informs the hardware to wake-up idle cores and migrates threads to the newly activated cores. As seen in Section 3, while computing in parallel, the power dissipation of the system exceeds TDP. Because




(a) Instantaneous activation (b) Linear ramp over 1.28$\mu$s (c) Linear ramp over 128$\mu$s

**Figure 6.** Supply voltage versus time for activating cores over three ramp-up times

continued activity at this rate would eventually cause overheating, sprinting requires additional mechanisms to determine the thermal state of the system to prevent overheating.

To construct an accurate view of the system's thermal status, our proposed design monitors energy dissipation since sprint initiation. Based on the dynamic energy consumption and a thermal model of the system, the hardware estimates when the available thermal budget is nearly exhausted. Such an activity-based mechanism is similar to dynamic thermal management employed in systems today to close the gap between worst-case thermal budgets and average-case power dissipation [7, 39].

If all computation is completed while in sprint mode, the software ends the sprint by simply turning off the the now-idle cores and placing them into a deep sleep mode. However, when the computation exceeds the sprint capabilities of system, the hardware intervenes and informs the software that the thermal sprint capacity is nearing exhaustion. The software is then responsible for migrating all active threads to a single core and turning off all other cores. Any remaining computation then proceeds in non-sprint mode by multiplexing the threads on a single core. To prevent thermal emergencies even if the operating system is unable to migrate threads and deactivate cores in time, as a last resort the hardware will throttle down the frequency of all active cores to remain under sustainable TDP. As dynamic power dissipation is linearly related to frequency, the hardware must throttle the frequency by at least a factor equal to the number of active cores. Once the software has migrated the threads and deactivated the cores, remaining computation can continue at nominal operating clock frequency.

## 8. Architectural Evaluation

The goal of this evaluation is to show that sub-second sprints can provide significant responsiveness benefit by reducing execution time. We use simulation of a many-core system to analyze the performance of parallel sprinting for a set of vision and image analysis kernels

| Kernel | Description |
|---|---|
| `sobel` | Edge detection filter; parallelized with OpenMP |
| `feature` | Feature extraction (SURF) from [12] |
| `kmeans` | Partition based clustering; parallelized with OpenMP |
| `disparity` | Stereo image disparity detection; adapted from [42] |
| `texture` | Image composition; adapted from [42] |
| `segment` | Image feature classification; adapted from [42] |

**Table 1.** Parallel kernels used in the evaluation

(Table 1) and a feature extraction application (`feature`) that is representative of the processing performed in camera-based search [16]. We explore a range of input sizes, thermal design points, and number of cores used to sprint.

### 8.1. Simulation Methodology

To show the feasibility and utility of parallel sprinting, we utilize a many-core instruction-level simulator to model sprint initiation and also behavior when the sprint interval is exhausted. Covering a full sprint requires many-core simulations on timescales of up to one second—billions of instructions (1 billion cycles for 16 1-IPC cores at 1 Ghz is 16 billion instructions). As the software is parallel and our runtime must react to the system's thermal behavior, we are unable to apply sampling or other simulation acceleration techniques in a straightforward manner.

To allow tractable simulation times of these timescales, we model in-order x86 cores with a CPI of one plus cache miss penalties. The cores have private 32KB 8-way set-associative caches, a shared 4MB 16-way last-level cache with 20 cycle hit latency, and a dual-channel memory interface with 4GB/sec channels and an uncontended 60ns round-trip latency. We model a standard invalidation-based cache coherence protocol with the directory co-located with the last-level cache. When sprinting begins, the L1 caches are initially empty and the cores are enabled only after the power supply is stable.

We augment this performance model with a dynamic energy model that associates energy with the type of instruction being executed. We derive energy estimates from McPAT [28], configured at a 1GHz, 1W core, 22nm

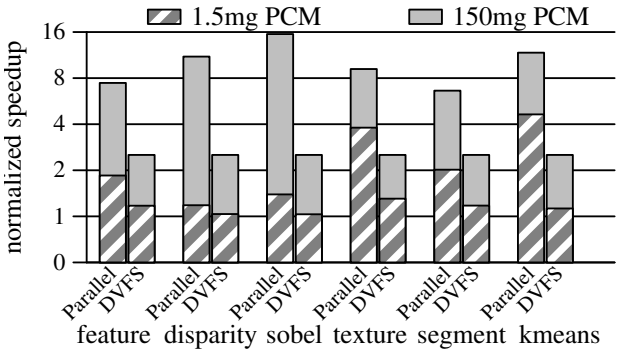



**Figure 7.** Parallel speedup on different workloads for 16 cores compared with idealized DVFS with the same maximum sprint power.

LOP (low operating power) technology node. During execution, we sample the energy consumed by each core every 1000 cycles to drive the thermal RC network model of the PCM-augmented heatsink described earlier in Section 4. We assume the chip to be a uniform heat source and do not account for temperature gradients across the die; recent work has shown tiled many-core architectures to be less susceptible to hot-spots [20]. To mitigate energy losses due to load imbalance and busy-waiting [27], the software runtime inserts `PAUSE` instructions on barriers, spinning on locks, and repeated failed task-stealing attempts. When a `PAUSE` instruction is encountered from a thread in sprint mode, the hardware puts that core to sleep for 1000 cycles. We assume the dynamic power dissipation of a sleeping core to be 10% that of an active core.

### 8.2. Increased Responsiveness

To demonstrate the improvement in responsiveness (*i.e.*, reduction in time to complete a short task), we simulated the parallel workloads on 16 cores with the full-provisioned thermal configuration (150 milligrams of PCM). The total height of the bars in Figure 7 shows an average parallel speedup of 10.2× over a single-core non-sprinting baseline. The sprinting and non-sprinting configurations both have the same TDP, last-level cache, and memory bandwidth constraints, but the sprinting configuration is able to use the additional cores (assumed to have been dark silicon) to improve responsiveness.

### 8.3. Thermal Capacitance Design Point

To measure the effect of limited sprint duration with tractable simulation times, we reduced the thermal capacitance of the system by reducing the amount of PCM by 100×. The bottom segment of the bars in Figure 7 represent the speedup for this design point. These simulations show smaller speedups, because under this more constrained thermal configuration all workloads exhaust

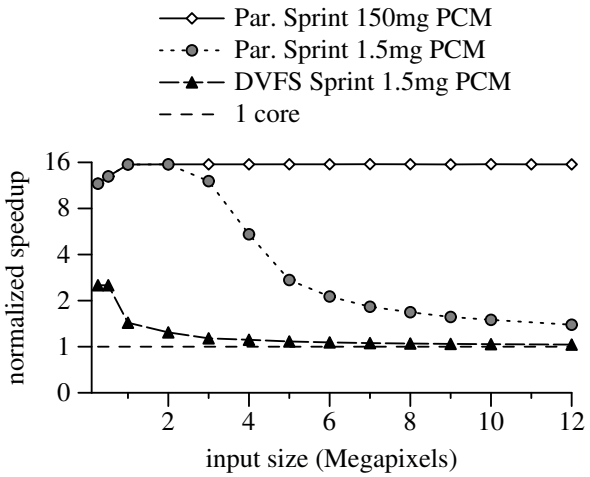



**Figure 8.** Parallel speedup for `sobel` with increasing computational demand for 16 cores compared with idealized DVFS with the same maximum sprint power.

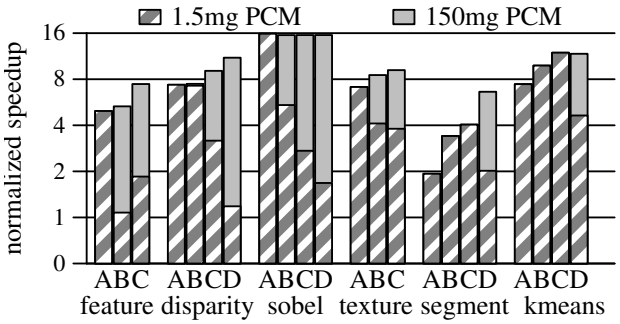



**Figure 9.** Speedup on 16 cores with varying input sizes

the sprint duration and are forced to execute some of the computation in the post-sprint single-core mode.

Figure 8 shows the impact of the thermal design on speedup (y-axis) for the `sobel` kernel as the amount of computation per sprint is increased by increasing image resolution (x-axis). Except for the lowest resolution images, `sobel` scales linearly up to 16 cores. For the fully sized PCM, the system is able to sustain the sprint for the entire computation at all image resolutions. However, for the artificially limited design point (1.5mg of PCM), the graph shows that speedup drops off as the fixed-sized sprint can handle less of the total computation.

Figure 9 shows speedups for all the workloads with varying problem sizes, reinforcing the trend of larger problems sizes exhibiting higher parallel speedup but also requiring larger thermal capacitance to complete during the sprint window. As seen in the `feature` application, parallel sprinting achieves an 8× speedup with the largest input size (HD image, bar C)—which allows the user to process an image with 8× the amount of detail than would be possible in a traditional (non-sprinting) device.

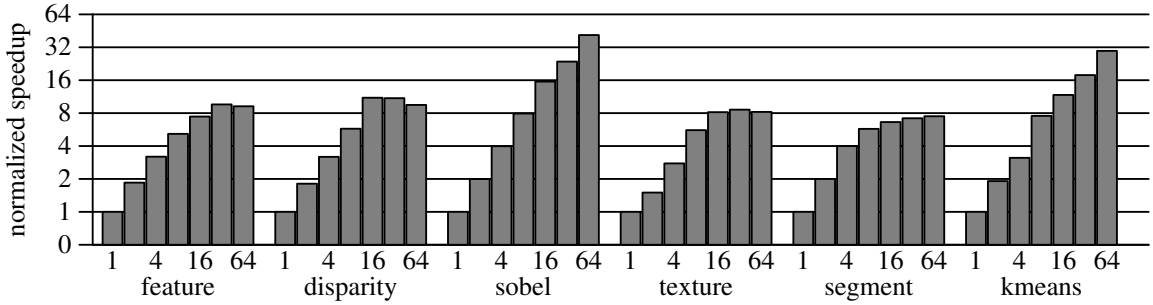



**Figure 10.** Parallel speedup with varying core counts at fixed voltage and frequency

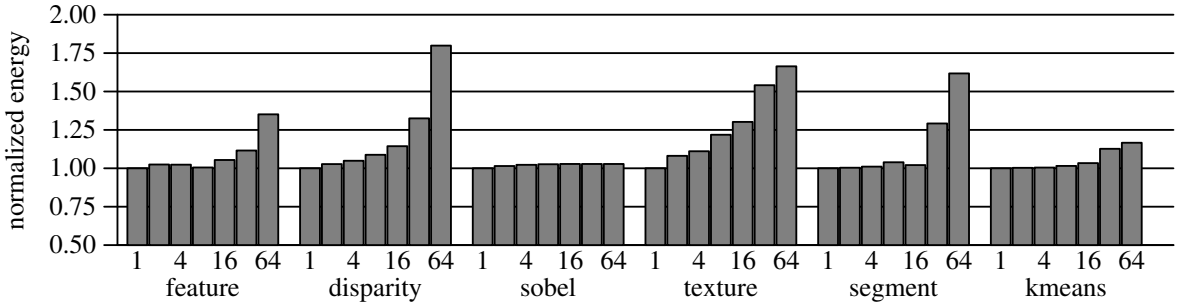



**Figure 11.** Dynamic energy with varying core counts at fixed voltage and frequency

### 8.4. Comparison with Voltage Boosting

Sprinting via boosting voltage and frequency is an alternative mechanism that exploits thermal capacitance. Our thermal and electrical design for a 16× sprint can use that headroom ether for 16 cores or a corresponding single-core frequency boost. A linear increase in voltage increases overall performance by a similar factor, but voltage has a quadratic impact on power, so we assume a 16× TDP headroom allows for a maximum frequency boost (and thus performance boost) of $\sqrt[3]{16} \approx 2.5\times$.

In Figure 7, the DVFS bars show the speedup achieved by single-core voltage-based sprinting over the single core non-sprint baseline. With sufficient thermal capacitance to sustain the duration of the sprint, the speedup scales as expected. However, with the more limited thermal configuration (1.5mg of phase-change material), voltage sprinting exhausts the available thermal capacitance after having completed less computation—because the rate of work completed is slower at the same power dissipation. This effect is visible in the lower component of the DVFS bar for each of the workloads. A similar effect is observed in Figure 8, in which the responsiveness of such a scheme falls rapidly due to early exhaustion of thermal capacitance as the image size increases to two megapixels and beyond.

### 8.5. Varying Intensity of Parallel Sprinting

Our results reported thus far assume 16 cores. However, changes in scaling trends could result in fewer or more cores available for sprinting on a future chip. Figure 10 shows how changing the number of sprinting cores affects the responsiveness (speedup) of each workload (for largest input size) over a single core baseline. As expected, configurations with fewer cores exhibit smaller speedups but are able to extract a higher percentage of peak throughput. With more cores, scaling drops off, but `kmeans` and `sobel` continue to scale well all the way up to 64 cores. The scaling of the other workloads are either limited by the available parallelism (`segment` and `texture`) or are limited by available memory bandwidth (`feature` and `disparity`). Doubling the memory bandwidth per channel allows `feature` and `disparity` to both achieve a 12× speedup on 64 cores.

### 8.6. Dynamic Energy Analysis

We next consider energy efficiency. Figure 11 shows the total dynamic energy consumption for each workload on varying numbers of cores. When operating in a region of linear speedup, the dynamic energy of the parallel sprint is unsurprisingly the same as the dynamic energy when executing on a single core; the same amount of work is performed, but just by many cores in less time. Even when operating in modes beyond linear scaling (*e.g.*, 6.6× speedup on 16 cores for `segment`), the runtime system's use of sleep modes is effective in avoiding dynamic energy increases. On 16 cores, the energy overhead due to parallel sprinting is less than 10% on five out of six workloads and only 12% on average. However, beyond sixteen cores, non-linear scaling and parallel execution overheads results in energy overheads of up to 1.8× over sequential execution (`disparity` on 64 cores). Sprinting

via voltage boosting results in a less energy efficient execution (not shown) because of the quadratic impact on power (approximately 6× more energy when using the 16× TDP headroom for voltage boosting).

## 9. Conclusion

In this paper we have presented a case for leveraging dark silicon on mobile platforms through parallel computational sprinting, targeting an order-of-magnitude gain in responsiveness. We explored the feasibility of computational sprinting in general and more specifically sprinting via parallelism, addressing several potential thermal and electrical barriers. Although numerous engineering challenges remain (in cost, thermal materials, packaging, and power supply), our study indicates that it is feasible to capture the responsiveness of a 16W chip within the engineering constraints of a 1W mobile device via parallel computational sprinting.

## Acknowledgments

The authors thank Andrew Hilton for use of his simulation infrastructure, Jason Clemons for providing the `feature` workload, and Dan Sorin, Karu Sankaralingam, and the anonymous reviewers for their comments on this work. This material is based upon work supported by the National Science Foundation under Grant No. CCF-0644197, CCF-0811320, CCF-0815457, and CCF-0916714.

## References

[1] International Technology Roadmap for Semiconductors, 2010 update, 2010. URL http://www.itrs.net.

[2] E. Alawadhi and C. Amon. PCM Thermal Control Unit for Portable Electronic Devices: Experimental and Numerical Studies. *IEEE Trans. on Components and Packaging Technology*, 26:116–125, 2003.

[3] G. Blake, R. G. Dreslinski, T. Mudge, and K. Flautner. Evolution of Thread-Level Parallelism in Desktop Applications. In *Proc. of the 37th Annual Int'l Symposium on Computer Architecture*, June 2010.

[4] S. Borkar. Major Challenges to Achieve Exascale Performance. *Salishan Conf. on High-Speed Computing*, 2009.

[5] S. Borkar and A. A. Chien. The Future of Microprocessors. *Communications of the ACM*, 54(5):67–77, 2011.

[6] R. J. Brodd, K. R. Bullock, R. A. Leising, R. L. Middaugh, J. R. Miller, and E. Takeuchi. Batteries, 1977 to 2002. *Journal of The Electrochemical Society*, 151(3): K1–K11, 2004.

[7] D. Brooks and M. Martonosi. Dynamic Thermal Management for High-Performance Microprocessors. In *Proc. of the 28th Annual Int'l Symp. on Computer Architecture*, July 2001.

[8] L. Cao, J. P. Krusius, M. A. Korhonen, and T. S. Fisher. Transient Thermal Management of Portable Electronics Using Heat Storage and Dynamic Power Dissipation Control. *IEEE Trans. on Components, Packaging, and Manufacturing Technology*, 21(1), Mar. 1998.

[9] K. Chakraborty, P. M. Wells, and G. S. Sohi. A Case for an Over-provisioned Multicore System: Energy Efficient Processing of Multithreaded Programs. Technical Report Technical Report #1607, Computer Sciences Department, University of Wisconsin–Madison, Oct. 2007.

[10] C.-P. Chiu, G. L. Solbrekken, V. LeBonheur, and Y. E. Xu. Application of Phase-Change Materials in Pentium III and Pentium III Xeon TM Processor Cartridges. *Intl. Symposium on Advanced Packaging Materials*, 2000.

[11] J. Clemons, A. J. R. Perricone, and a. T. A. Silvio Savarese. EFFEX: An Embedded Processor for Computer Vision Based Feature Extraction. In *Proceedings of the 45th Design Automation Conference*, June 2011.

[12] J. Clemons, H. Zhu, S. Savarese, and T. Austin. MEVBench: A Mobile Computer Vision Benchmarking Suite. In *Proc. of the IEEE Int'l Symp. on Workload Characterization*, Sept. 2011.

[13] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger. Dark Silicon and the End of Multicore Scaling. In *Proc. of the 38th Annual Int'l Symposium on Computer Architecture*, June 2011.

[14] K. Flautner, S. K. Reinhardt, and T. N. Mudge. Automatic Performance-Setting for Dynamic Voltage Scaling. In *Proceedings of the 7th Conference on Mobile Computing and Networking MOBICOM*, 2001.

[15] S. H. Fuller and L. I. Millett. Computing Performance: Game Over or Next Level? *IEEE Computer*, 44(1):31–38, 2011.

[16] B. Girod, V. Chandrasekhar, D. M. Chen, N.-M. Cheung, R. Grzeszczuk, Y. Reznik, G. Takacs, S. S. Tsai, and R. Vedantham. Mobile Visual Search. *IEEE Signal Processing Magazine*, July 2011.

[17] S. P. Gurrum, Y. K. Joshi, and J. Kim. Thermal Management of High Temperature Pulsed Electronics Using Metallic Phase Change Materials. *Numerical Heat Transfer, Part A: Applications: An Int'l Journal of Computation and Methodology Issue 8*, 42:777–790, 2002.

[18] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki. Toward Dark Silicon in Servers. *IEEEMICRO*, 31(4):6–15, July 2011.

[19] M. Hodes, R. D. Weinstein, S. J. Pence, J. M. Piccini, L. Manzione, and C. Chen. Transient Thermal Management of a Handset Using Phase Change Material (PCM). *Journal of Electronic Packaging*, 124(4):419–426, 2002.

[20] W. Huang, M. R. Stan, K. Sankaranarayanan, R. J. Ribando, and K. Skadron. Many-Core Design from a Thermal Perspective. In *Proceedings of the 45th Design Automation Conference*, June 2008.

[21] D.-C. Juan, Y.-T. Chen, M.-C. Lee, and S.-C. Chang. An Efficient Wake-Up Strategy Considering Spurious Glitches Phenomenon for Power Gating Designs. *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, 18(2):246 –255, Feb. 2010.

[22] R. Kandasamy, X.-Q. Wang, and A. S. Mujumdar. Application of Phase Change Materials in Thermal Management of Electronics. *Applied Thermal Engineering*, 27 (17-18):2822 – 2832, 2007.

[23] B. Kang and G. Ceder. Battery Materials for Ultrafast Charging and Discharging. *Nature*, 458(7235):190–193, Mar. 2009.

[24] S. Kim, S. V. Kosonocky, and D. R. Knebel. Understanding and Minimizing Ground Bounce during Mode Transition of Power Gating Structures. In *Proc. of the 2003 Int'l Symp. on Low Power Electronics and Design*, 2003.

[25] S. Kim, S. V. Kosonocky, D. R. Knebel, K. Stawiasz, and M. C. Papaefthymiou. A Multi-Mode Power Gating Structure for Low-Voltage Deep-Submicron CMOS ICs. *IEEE Trans. on Circuits and Systems II*, 54(7), July 2007.

[26] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks. System Level Analysis of Fast, Per-Core DVFS Using On-Chip Switching Regulators. In *Proc. of the 14th Symposium on High-Performance Computer Architecture*, Feb. 2008.

[27] J. Li, J. F. Martinez, and M. C. Huang. The Thrifty Barrier: Energy-Aware Synchronization in Shared-Memory Multiprocessors. In *Proc. of the Tenth Symposium on High-Performance Computer Architecture*, Feb. 2004.

[28] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi. McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures. In *Proc. of the 42nd Int'l Symp. on Microarchitecture*, Nov. 2009.

[29] Z. Luo, H. Cho, X. Luo, and K. il Cho. System Thermal Analysis for Mobile Phone. *Applied Thermal Engineering*, 28(14-15):1889 – 1895, 2008.

[30] R. Merritt. ARM CTO: Power Surge Could Create 'Dark Silicon'. *EE Times*, Oct. 2009. URL http://www.eetimes.com/electronics-news/4085396/ARM-CTO-power-surge-could-create-dark-silicon-.

[31] F. J. Mesa-Martinez, E. K. Ardestani, and J. Renau. Characterizing Processor Thermal Behavior. In *Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems*, Mar. 2010.

[32] A. Mirhoseini and F. Koushanfar. HypoEnergy: Hybrid Supercapacitor-Battery Power-Supply Optimization for Energy Efficiency. In *Proceedings of the Conference on Design, Automation and Test in Europe*, Mar. 2011.

[33] L. Palma, P. Enjeti, and J. Howze. An Approach to Improve Battery Run-time in Mobile Applications with Supercapacitors. In *34th Annual IEEE Power Electronics Specialist Conference*, volume 2, June 2003.

[34] M. Pedram, N. Chang, Y. Kim, and Y. Wang. Hybrid Electrical Energy Storage Systems. In *Proc. of the 2010 Int'l Symp. on Low Power Electronics and Design*, 2010.

[35] M. Popovich, A. Mezhiba, and E. Friedman. *Power Distribution Networks with On-Chip Decoupling Capacitors*. Springer, 2008.

[36] E. Rotem, A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann. Power Management Architecture of the 2nd Generation Intel Core Microarchitecture, Formerly Codenamed Sandy Bridge. In *Hot Chips 23 Symposium*, Aug. 2011.

[37] J. Schindall. The Charge of the Ultracapacitors. *IEEE Spectrum*, 44(11):42–46, Nov. 2007.

[38] G. Setoh, F. Tan, and S. Fok. Experimental Studies on the use of Phase Change Material for Cooling Mobile Phones. *International Communications in Heat and Mass Transfer*, 37(9):1403 – 1410, 2010.

[39] B. Shi, Y. Zhang, and A. Srivastava. Dynamic Thermal Management for Single and Multicore Processors under Soft Thermal Constraints. In *Proc. of the 2010 Int'l Symp. on Low Power Electronics and Design*, 2010.

[40] A. Shye, B. Scholbrock, and G. Memik. Into the Wild: Studying Real User Activity Patterns to Guide Power Optimizations for Mobile Architectures. In *Proc. of the 42nd Int'l Symp. on Microarchitecture*, Nov. 2009.

[41] F. Tan and S. Fok. Thermal Management of Mobile Phone Using Phase Change Material. In *Proceedings of the Ninth Electronics Packaging Technology Conference*, Dec. 2007.

[42] S. K. Venkata, I. Ahn, D. Jeon, A. Gupta, C. Louie, S. Garcia, S. Belongie, and M. B. Taylor. SD-VBS: The San Diego Vision Benchmark Suite. In *Proc. of the IEEE Int'l Symp. on Workload Characterization*, Sept. 2009.

[43] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor. Conservation Cores: Reducing the Energy of Mature Computations. In *Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems*, Mar. 2010.

[44] R. Wirtz, N. Zheng, and D. Chandra. Thermal Management Using Dry Phase Change Materials. In *Proceedings of the Fifteenth Annual IEEE Semiconductor Thermal Measurement and Management Symposium*, 1999.

[45] L. Yan, L. Zhong, and N. Jha. User-Perceived Latency Driven Voltage Scaling for Interactive Applications. In *Proc. of the 41st Design Automation Conf.*, June 2005.