EnergySmart: Toward Energy-Efficient Manycores for Near-Threshold Computing*

Ulya R. Karpuzcu‡, Abhishek Sinkar†, Nam Sung Kim†, and Josep Torrellas‡
‡ University of Illinois Urbana-Champaign † University of Wisconsin-Madison
ukarpuzc@umn.edu {sinkar,nskim3}@wisc.edu torrella@illinois.edu

Abstract

While Near-Threshold Voltage Computing (NTC) is a promising approach to push back the manycore power wall, it suffers from a high sensitivity to parameter variations. One possible way to cope with variations is to use multiple on-chip voltage (V_{dd}) domains. However, this paper finds that such an approach is energy inefficient. Consequently, for NTC, we propose a manycore organization that has a single V_{dd} domain and relies on multiple frequency domains to tackle variation. We call it EnergySmart*. For this approach to be competitive, it has to be paired with effective core assignment strategies and also support fine-grain (i.e., short-interval) DVFS. This paper shows that, at NTC, a simple chip with a single V_{dd} domain can deliver a higher performance per watt than one with multiple V_{dd} domains.*

1. Introduction

Manycore scaling now faces a Power Wall. Successive technology generations have been increasing chip power density, resulting in a situation where more cores can now be placed on a chip than can be concurrently operating. We urgently need new ways to execute more power- and energy-efficiently.

One way to improve energy efficiency is to reduce the supply voltage (V_{dd}) to a value a bit higher than a transistor's threshold voltage (V_{th}). This environment is called Near-Threshold Voltage Computing (NTC) [7, 13, 23] — as opposed to the conventional Super-Threshold Voltage Computing (STC). V_{dd} is a powerful knob because it has a strong impact on both dynamic and static energy. According to initial estimates [7, 13], NTC can decrease the energy per operation by several times over STC. One drawback is a degradation in frequency, which may be tolerable through more parallelism in the application. Since many more cores can be executing concurrently within the chip's power envelope, the result is a higher throughput for parallel codes.

A major roadblock to NTC is parameter variations, namely the deviation of device parameters from their nominal specifications. Already at STC, a chip has regions with different speed and power consumption. At NTC, the same amount of variations causes larger changes in device speed and power due to the low V_{dd} [23].

*This work was supported in part by the National Science Foundation under grants CPA-0702501, CCF-095360, CCF-1012759, CCF-1016262, and CNS-1217102; DOE ASCR under Award Number DE-FC02-10ER2599; DARPA under UHPC Contract Number HR0011-10-3-0007; an IBM Faculty Award; and a generous gift from AMD. Ulya R. Karpuzcu is now with the Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities. Abhishek Sinkar is now with Oracle.

It is important to find techniques to cope with parameter variations in future NTC chips. Currently, there are some techniques being used to handle variations, such as Adaptive Body Biasing (ABB) and V_{dd} tuning — in the form of Adaptive Supply Voltage (ASV) or Dynamic Voltage Scaling (DVS). Their effectiveness increases with support for multiple V_{dd} and frequency (f) domains. With these domains, we can separately adapt to the parameter values in the different chip regions.

Unfortunately, simply applying these techniques will likely not work at NTC. There is consensus that ABB will be ineffective in new technologies. In addition, supporting many on-chip V_{dd} domains with conventional on-chip V_{dd} regulators is not energy efficient — hence hardly compatible with an energy-efficient environment such as NTC. The reason is that on-chip regulators — the only type we currently know how to build to provide the desired functionality — have a typical power efficiency of 75-90% and take up a lot of chip area [8, 15, 20]. Moreover, given the hundreds of cores that an NTC chip can have, we would need many such regulators, substantially increasing the complexity.

In this paper, we examine the energy inefficiency of multiple V_{dd} domains at NTC. We also propose a simpler alternative that tackles variation with a single chip-wide V_{dd} domain and multiple f domains. We call this new organization *EnergySmart*. However, for such an organization to be competitive, we need to address two issues. The first one is to provide core-to-job assignment algorithms that deliver high performance per watt without relying on multiple V_{dd} domains. The second is to support chip-wide DVFS that adapts V_{dd} in intervals short enough to be competitive with an environment with on-chip V_{dd} regulators.

This paper makes three contributions. First, it proposes EnergySmart and shows that, at NTC, it delivers higher performance per watt than a chip with multiple on-chip V_{dd} domains supported by conventional on-chip V_{dd} regulators — and, in addition, its hardware is simpler. The reasons are: (i) the power inefficiencies of the on-chip regulators, (ii) the increased V_{dd} guardbands needed by the fine-grain V_{dd} domains (to handle deeper V_{dd} droops due to the lower ability to average the current drawn), and (iii) the fact that the V_{dd} domains are physically large enough to include within-domain variations. Second, this paper introduces core assignment algorithms for the EnergySmart architecture that deliver high performance per watt and are simple to implement. Finally, the paper shows that the lower speed of V_{dd} changes without on-chip V_{dd} regulators does not hamper effective DVFS.

The paper is organized as follows: Section 2 provides a

background; Section 3 discusses multiple V_{dd} domains at NTC and presents EnergySmart; Sections 4 and 5 discuss core assignment and fine-grain DVFS for EnergySmart; Sections 6 and 7 evaluate the ideas; Section 8 discusses how our analysis relates to STC chips; and Section 9 covers related work.

2. Near-Threshold Computing and Variations

2.1. Near-Threshold Computing (NTC) Basics

NTC refers to an environment where V_{dd} is a bit higher than the transistors' V_{th} [7, 13, 23]. For current technologies, it roughly corresponds to $V_{dd} \approx$0.5V, while conventional (Super-Threshold Voltage Computing or STC) environments correspond to $V_{dd} \approx$1V. NTC is interesting because it reduces the energy per operation several times compared to STC — at the expense of degrading the frequency of operation [13]. As a result, the power is expected to reduce by over an order of magnitude, allowing more cores to operate simultaneously for the same manycore power envelope. If the application has parallelism, this is a major advantage.

Figure 1 compares the scaling of three parameters under NTC, STC, and classical CMOS theory [10]: V_{dd}, transistor delay and power density. The X axis shows gate length to characterize each technology generation. According to classical scaling, both V_{dd} and transistor delay reduce at each generation, giving rise to a constant power density. Under STC scaling, the decrease of the transistor's V_{th} has practically stopped, to keep subthreshold leakage under control, which in turn has prevented V_{dd} from scaling [16]. As a result, power density has been increasing. As we go from STC to NTC scaling, the curves experience vertical shifts: V_{dd} decreases (Figure 1(a)), hence power density goes down and transistor delay increases (Figure 1(b)).

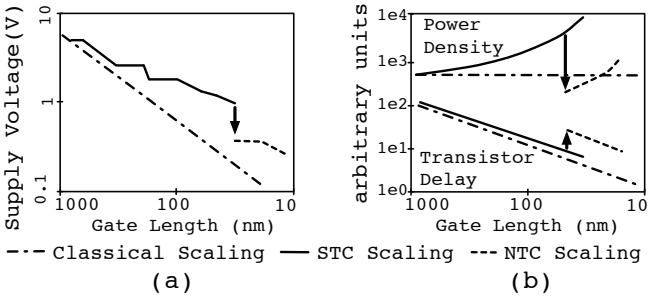


Figure 1: Scaling under three scenarios (from [7]).

In terms of energy and delay, NTC represents a desirable operating region. Figure 2 shows the inverse of energy per operation (labeled as energy efficiency) in MIPS/w (left Y axis) and the transistor delay (right Y axis) as a function of V_{dd}. At NTC, the energy efficiency remains high and the transistor delay is relatively low. Away from this region, higher V_{dd} quickly reduces energy efficiency. Lower V_{dd}, on the other hand, quickly leads to slower transistors.

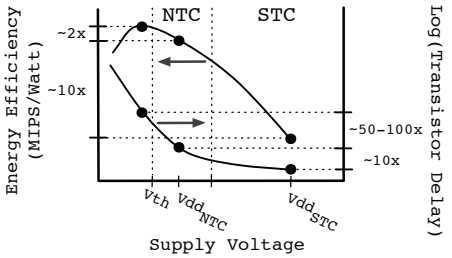


Figure 2: Impact of V_{dd} on efficiency and delay (from [13]).

2.2. Impact of Process Variations at NTC

Process variations, the deviation of device parameters from their nominal specifications, manifest across the chip as static, spatial fluctuations in transistor parameters [3, 4]. Within-die (WID) process variations are caused by systematic effects (e.g., lithographic irregularities) and random effects (e.g., varying dopant concentrations) [33]. Two key parameters affected by variations are V_{th} and the effective channel length (L_{eff}). The higher the V_{th} and L_{eff} variations become, the higher the variations in transistor switching speed and static power consumption get. The result is chips with increased variation in frequency and power consumption across cores and memories. Under variation, the average core has lower frequency than otherwise. This is because the slower transistors determine the core's frequency. Moreover, the average core consumes more static power. The reason is that low-V_{th} transistors consume more additional static power than high-V_{th} ones save.

Transistor delay and power consumption are more sensitive to variations at NTC than at STC. Consider transistor delay. At low V_{dd}, the same ΔV_{th} causes a larger change in transistor delay [14]. At 11nm, for example, according to the NTC-specific delay model from Markovic *et al.* [23], as V_{th} changes from 0.20 to 0.26mV, the transistor delay increases by 50% at V_{dd}=0.7V and more than doubles at V_{dd}=0.5V.

Consider power now, both static and dynamic. As V_{dd} decreases to the NTC region, dynamic power reduces more than static power does. Hence, static power becomes a major contributor to the total power. Since static power depends exponentially on V_{th}, variations in V_{th} affect the variation in the transistor's static power exponentially. In addition, dynamic power variation also increases at NTC over STC. The reason is that dynamic power depends on the frequency and, as we have seen, at low V_{dd}, transistor delay (and hence frequency) is more sensitive to changes in V_{th}.

2.3. Modeling Process Variations at NTC

To model WID process variations at NTC, we use VARIUS-NTV [18], a recent model based on VARIUS [31] that works at NTC. For V_{th} and L_{eff}, it models the systematic component with a multivariate normal distribution with σ_{sys}, and the spatial correlation using a spherical function. At a distance ϕ, the correlation becomes 0. The random component is modeled with a multivariate normal distribution with σ_{ran}.

VARIUS-NTV plugs the V_{th} and L_{eff} variations into the EKV-based equation of Markovic *et al.* [23] to estimate tran-

sistor (and gate) delay variation, and in the equation for static power [9] to estimate transistor power variation.

VARIUS-NTV follows other work [2, 7] in using an 8-transistor cell for SRAM. Such a cell is easy to design reliably at NTC (unlike the 6T cell used by VARIUS) because it decouples the transistors used for reading and writing. Moreover, VARIUS-NTV models all the types of SRAM failure modes that can occur at NTC (read access, write stability, write timing, and hold) [25]. VARIUS only modeled read access failures. Finally, in SRAM timing and stability analysis, VARIUS-NTV models the impact of leakage, which is substantial at NTC and neglected by VARIUS.

To estimate the frequency of pipeline stages and cache modules, VARIUS-NTV uses gate delay variations and critical path distributions. The slowest pipeline stage determines the core's frequency. VARIUS-NTV finds the chip's static power by integrating the power over all the on-chip transistors. VARIUS-NTV logic and memory models are validated in [18] with variation measurements from Intel's 80-Core TeraFLOPS processor [11]. Moreover, the models use VARIUS parameters that were verified using experimental data from Razor publications [31].

3. Eschewing Multiple V_{dd} Domains at NTC

Having multiple on-chip V_{dd} domains with independent voltage scaling can increase the energy efficiency of a manycore — e.g., by executing memory-intensive programs in low-V_{dd} domains. Since NTC is an energy-conscious environment, such support appears to suit it. In addition, NTC has two additional reasons to benefit from multiple V_{dd} domains. First, since WID process variations have a higher impact at NTC, chip neighborhoods at NTC are expected to benefit more from the decoupling of V_{dd} values across the chip. The second reason is that, at NTC, V_{dd} is a particularly strong lever to affect the power and performance conditions of operation. Specifically, small changes in V_{dd} have a relatively large impact on the performance and energy consumption of a core.

However, we uncover several limitations that make the use of multiple V_{dd} domains at NTC less attractive. The presence of such limitations suggests a different type of manycore architecture at NTC, and two challenges that need to be overcome for cost-effective operation. Next, we describe these limitations, the architecture, and the challenges.

3.1. Limitations of Multiple V_{dd} Domains at NTC

There are several effects that limit the cost-effectiveness of multiple on-chip V_{dd} domains in an energy-conscious environment such as a future NTC chip. While some of these effects also apply to STC, their overall significance is lower at STC. The effects are shown in Table 1. The first one is the power loss in conventional on-chip V_{dd} regulators. Having V_{dd} regulators on chip is likely the only realistic way to support many domains (e.g., 36 in the hypothetical chip we consider) — utilizing many off-chip regulators is too expensive. Unfortunately, such regulators have typical power efficiencies of 75-90%, be they switching or low-dropout (LDO) regulators. For example, Ghasemi *et al.* [15] discuss LDO regulators with a range of efficiencies, while Kim *et al.* [20] discuss on-chip switching regulators that have a peak power efficiency of 77%. Chang *et al.* [8] developed a design with 90% efficiency, but this is a regulator with a fixed 2 to 1 Vin/Vout voltage conversion. As a result, while it is more efficient than a variable Vout regulator, it is quite limited. Moreover, the design takes a lot of area, which is a very valuable commodity. Overall, while we expect regulator design to improve with time, their efficiency will likely remain in the upper part of the range indicated. This is hardly compatible with a very energy-conscious environment like NTC.

Limitation	Reason
Power loss in on-chip V_{dd} regulators	Regulators have power efficiencies of 75-90%
Increased V_{dd} guardband to tolerate larger dynamic V_{dd} droops	Fine-grain domains lack averaging effects in the current drawn
Inability to fully fine-tune V_{dd} for individual cores in one domain	To reduce cost and complexity, a domain is physically large and has within-domain variation

Table 1: Why multiple V_{dd} domains become less attractive in a future NTC environment.

The second effect is the likely need to increase the V_{dd} guardband in the finer-grain V_{dd} domains, to guard against more accentuated dynamic V_{dd} droops. Such deeper droops can be induced by the lack of averaging effects in the current drawn by the small domain — compared to a large chip with a single V_{dd} domain. Indeed, in the latter, the current drawn tends to average out over the activity of many cores, inducing smaller swings. James *et al.* [17] discuss this problem for the IBM POWER6 processor, and observe a 5-10% increase in V_{dd} droop due to domain separation. Increased V_{dd} guardbands imply lower energy efficiencies, which are undesirable in an NTC environment. In an STC environment, V_{dd} guardbanding is relatively less critical, both because V_{dd} is higher and because process variation has a lower impact.

Finally, the low-power operation of NTC will result in physically-large chips, with possibly hundreds of cores. Attempting to provide a V_{dd} domain per core is unrealistic. On the other hand, given the physical size of the chip, dividing the chip into several V_{dd} domains will produce large, suboptimal domains. Each domain will likely include a sizable number of cores, with non-trivial within-domain differences — especially given the increased sensitivity to process variations. This will lead to a V_{dd} setting that is suboptimal for individual cores — hence missing out on part of the potential gains. This effect is less applicable to an STC environment.

While these effects relate to energy-efficiency considerations, on-chip regulators also consume substantial area and introduce design complexity. For all these reasons, we now propose an alternative architecture. Further discussion on how these issues impact or are related to STC chips is presented in Section 8.

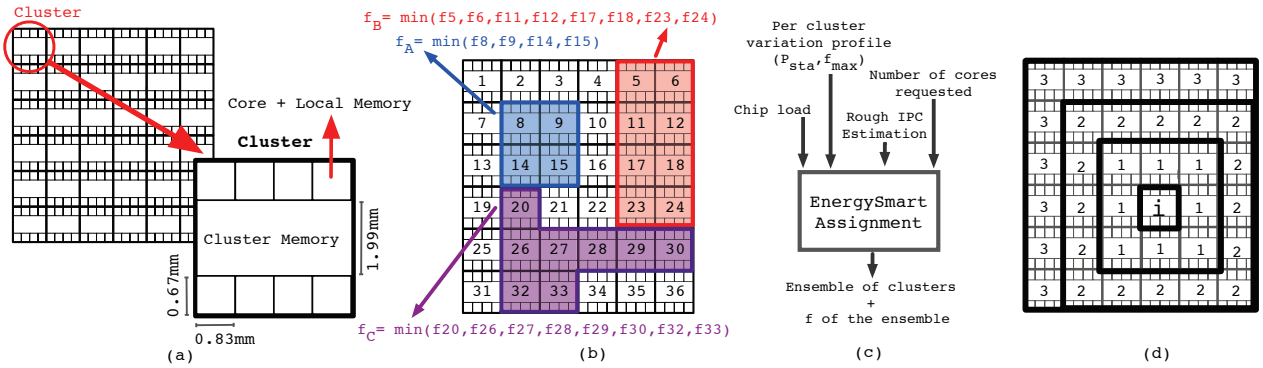


Figure 3: Example EnergySmart architecture (a), its operation (b), the core assignment algorithm (c), and distance of clusters to cluster *i* (d).

3.2. An Alternative for Future NTC: *EnergySmart*

We propose an alternative architecture for future NTC manycores. The architecture, called *EnergySmart*, eschews multiple on-chip V_{dd} domains for energy efficiency. It keeps a single V_{dd} in the whole chip. DVFS can be used, but is applied globally across the chip. By removing V_{dd} domains, EnergySmart also gains in hardware simplicity and saves chip area.

To handle process variations more inexpensively than with fine-grain V_{dd} and frequency (f) domains, EnergySmart uses only f domains. EnergySmart is organized in clusters of cores, where each cluster is potentially a f domain. A cluster is characterized by the maximum frequency (f_{max}) that it can support at the lowest possible chip-wide safe voltage (V_{ddNOM}). V_{ddNOM} and the set of f_{max} are set at manufacturing-testing time. With many f domains, the chip still has many degrees of freedom to tackle process variation.

Figure 3(a) shows an EnergySmart manycore with measures for 11nm. The chip has 6×6 clusters, where each cluster has a cluster memory and 8 cores with local memories.

To determine V_{ddNOM} and the set of f_{max}, we proceed as follows. Each cluster's minimum sustainable voltage, V_{ddMIN}, is set after performing SRAM hold and write stability failure analyses to ensure reliable operation. Then, the chip-wide V_{ddNOM} becomes the maximum of all clusters' V_{ddMIN}. After V_{ddNOM} is set, timing tests in the SRAM and logic for each cluster i determine the maximum frequency f_{max_i} that the cluster can support at V_{ddNOM}. Such frequency will be the default frequency of the cluster. It can be increased if V_{dd} increases over V_{ddNOM}.

In EnergySmart, where we give up multiple V_{dd} domains and, instead, rely on fine-grain f domains, two challenges appear: the need to (1) carefully perform core-to-job assignment, and (2) effectively support fine-grain (i.e., short-interval) DVFS. We consider them next.

4. The Challenge of Core Assignment

4.1. Rationale: Simplicity and Effectiveness

Attaining energy-efficient performance in a manycore with hundreds of cores as enabled by NTC requires good core-assignment algorithms. Unfortunately, the number of degrees of freedom in the assignment is vast. Hence, the challenge is to provide effective assignment while keeping the algorithms simple so they are implementable. EnergySmart infuses simplicity by eliminating V_{dd} domains: the assignment algorithm only needs to select the f of the chosen cores, rather than both V_{dd} and f. However, the inability to set multiple V_{dd}s in the chip disables a knob for energy efficiency. Hence, EnergySmart's assignment algorithm needs careful design.

The chip is organized in clusters of cores to exploit systematic variations and to enhance scalability. For assignment simplicity, EnergySmart sets the cluster to be the smallest f domain. Clocking all the cores in a cluster at the same f is reasonable, since the whole cluster is likely to have a similar value of the systematic component of process variations. EnergySmart further simplifies core assignment by assigning all the cores in a cluster as a group to a job. Any resulting unused cores in the cluster are power gated. Leaving them unused is typically not a problem because, in our environment, there is likely a surplus of cores. However, if free cores are scarce, a cluster can take-in multiple jobs.

A single parallel job may grab multiple clusters. Such set of clusters is called an *Ensemble*. EnergySmart runs an ensemble at a single f which, at V_{ddNOM}, is equal to the lowest of the f_{max} of the constituting clusters. We pick a single f for the whole ensemble to keep the assignment algorithm simple and, therefore, implementable. In addition, running at a single f ensures that all the threads of the job make similar progress, which typically results in faster overall execution [21] — especially in highly-synchronized codes. While we could pick a different f for each of the clusters (and even cores) in the ensemble, the result would be a complex and slow assignment algorithm. In addition, since we do not know the structure of individual programs, assignment of different f to clusters or cores may result in higher energy for no performance gain.

When multiple jobs are running concurrently, each is assigned to a different ensemble, which forms a separate f domain. An example is shown in Figure 3(b), where f_{max} for cluster i is depicted as f_i. Three ensembles are allocated to three jobs (A, B and C), giving rise to three independent f domains characterized by frequencies f_A, f_B and f_C.

There are various design alternatives for the network that interconnects the clusters. One alternative is to have a single, separate f domain for the network [11]. This is a simple

design, but requires that every communication between clusters in an ensemble cross domains. An alternative is to break the network into multiple *f* domains, each including multiple clusters. This design could allow clusters within an ensemble to communicate without crossing *f* domains — although longer-distance messages would still have to cross domains. Moreover, the network is more complicated to design. Given that crossing a *f* domain only adds about 2 ns [12], we choose the simple, single *f* domain for the network. Hence, when clusters within an ensemble communicate, they cross *f* domains. However, a baseline chip that uses multiple V_{dd} domains also has the same issue: communicating clusters cross *f* domains.

One degree of freedom in EnergySmart's assignment algorithm is whether the clusters that form an ensemble have to be physically contiguous. Not worrying about contiguity simplifies the algorithm, but may result in ensembles where inter-thread communication is expensive. In our design, we consider two algorithms that give either high or low priority to choosing contiguous clusters for an ensemble. In practice, close-by clusters have similar systematic variation values. As a result, both algorithms try to pick contiguous clusters implicitly or explicitly. We consider the algorithms next.

4.2. Core Assignment Algorithm: M_Assign

We call our core assignment algorithm *M_Assign* (for many-core assignment). When a new job arrives, *M_Assign* assigns an ensemble of clusters to it at a single *f* and, typically, does not revisit the assignment during the job's lifetime. In the following discussion, *M_Assign* maximizes MIPS/w; other related metrics can also be used.

M_Assign uses information from both hardware and application. The hardware information includes each cluster's static power (P_{sta}) and maximum frequency supported (f_{max}) at V_{ddNOM} and reference temperature (T). This information is generated at manufacturing-testing time. Providing this information for a single T may be enough, as at NTC, T is lower than at STC and does not vary much. However, for higher precision, the manufacturer may provide *M_Assign* with a table of P_{sta} and f_{max} values for different T. Then, based on on-line measurement of the T, *M_Assign* could use the most appropriate value. Finally, another piece of information is the load of the chip (which clusters are busy).

The application information is the number of cores requested (equal to the number of threads) and an estimate of the average IPC of the application's threads. The IPC is provided for a few *f* values, and is interpolated for the others. It can be obtained from previous runs of the application or from the current run.

The output of *M_Assign* is the chosen ensemble of clusters for the job, plus the *f* these clusters should run at — equal to the minimum of the f_{max} of the chosen clusters (Figure 3(c)).

To see the simplicity of *M_Assign*, assume that a job requests n cores. *M_Assign* must return an ensemble E of size $|E| = \lceil n/ClSize \rceil$ clusters, where *ClSize* is the cluster size. Naively, *M_Assign* could simply check all the possible groups of $|E|$ free clusters, and pick the group that delivers the maximum MIPS/w at V_{ddNOM}. In our design, *M_Assign* relies on an intelligent exhaustive search, where the search space gets pruned and the runtime complexity reduced significantly. Specifically, *M_Assign* repeatedly picks one free cluster i (which can cycle at most at f_{max_i}), and combines it with the best selection of $|E-1|$ clusters among those that can cycle *faster* than i, to arrive at the ensemble E which maximizes MIPS/w:

$$\begin{aligned} & max_E(\frac{MIPS}{watt}) \equiv \\ & \equiv min_E(\frac{watt}{MIPS}) \equiv min_E(\frac{\sum_E P_{sta} + \sum_E P_{dyn}}{IPC \times |E| \times ClSize \times f_{max\_i}}) \\ & \equiv min_E(\frac{\sum_E P_{sta} + C \times V_{ddNOM}^2 \times |E| \times f_{max\_i}}{IPC \times |E| \times ClSize \times f_{max\_i}}) \end{aligned} \quad (1)$$

At the time cluster i is considered, all variables of this formula are known except $\sum_E P_{sta}$, the total P_{sta} of the ensemble E to be formed. We know the *f* of E, f_{max_i}, as set by the slowest cluster, namely cluster i. The operating V_{dd} is fixed chip-wide to V_{ddNOM}. The number of cores requested determines the ensemble size $|E|$. An estimate of $IPC(f_{max\_i})$ is also available. Finally, C, the average cluster capacitance, is proportional to the area, and does not depend on the selection. $\sum_E P_{sta}$, on the other hand, changes with the selection of the clusters to form E. Thus, for each cluster i considered, the ensemble that maximizes MIPS/w reduces to the ensemble of the clusters that deliver $min(\sum_E P_{sta})$. The pseudo-code for *M_Assign* is given in Algorithm 1, where E^* is the optimal ensemble of $|E|$ clusters and E is the selected candidate ensemble.

Algorithm 1 Pseudo-code for *M_Assign*.

```
1: E* ← ∅
2: for each free cluster i in the chip do
3:    /* cluster i cycles at f_max_i, and so we use IPC(f_max_i) */
4:    find the |E| − 1 free clusters faster than i that have the minimum Σ_{E−1} P_sta
5:    /* these clusters can all cycle at f_max_i or higher */
6:    E ← { i ∪ these |E|-1 clusters }
7:    if (MIPS/watt(E) > MIPS/watt(E*)) then
8:       E* ← E
```

M_Assign runs very fast if the clusters are ordered offline from lowest to highest P_{sta}, and from highest to lowest f_{max}. As *M_Assign* picks one cluster i at a time, it only needs to select, among those with higher f_{max}, the $|E|-1$ ones that have the lowest P_{sta}. It then computes the MIPS/w of the ensemble. This process is repeated once for each available cluster i, and the ensemble with the highest MIPS/w is picked.

We design two *M_Assign* algorithms that differ in the priority given to choosing contiguous clusters for an ensemble. *M_Assign_NC* (for non-contiguous) is the algorithm just described, which neglects contiguity. *M_Assign_C* (for contiguous) gives priority to picking contiguous clusters. It does so by picking the ensemble of the clusters that deliver $min(\sum_{E-1}(D_{ij} \times P_{sta\_j}))$ as opposed to $min(\sum_{E-1} P_{sta})$. In this

formula, the summation is done over all component clusters j of E except i, and D_{ij} is the wavefront distance between cluster i and component cluster j (Figure 3(d)). Consequently, clusters that are far apart from i are penalized.

M_Assign is an *exact* algorithm rather than heuristic-based. However, it has low overhead and is scalable. The reason is that it is optimized for the EnergySmart architecture: it leverages EnergySmart's properties of no V_{dd} domains, single f per ensemble (and per cluster), and whole-cluster assignment. These characteristics make it different from past work on assignment algorithms (discussed in Section 9). Section 7.3 quantifies *M_Assign*'s instruction count. It scales with O(N^2), where N is the number of clusters per chip.

4.3. Other Assignment Algorithms

We consider two simple, greedy algorithms. A non-contiguous version, *Greedy_NC*, picks free clusters in increasing order of P_{sta}/f_{max}. A contiguous version, *Greedy_C*, first picks the free cluster of minimum P_{sta}/f_{max} as the center, and expands the ensemble along wavefronts of increasing distance from the center. At each wavefront, it picks clusters in increasing order of P_{sta}/f_{max}. These algorithms scale with O(N).

In contrast, a baseline manycore with per-cluster V_{dd} and f domains needs more complicated assignment algorithms. Specifically, assume that a job needs an ensemble of $|E|$ clusters. We need to find the set of clusters for the ensemble, and the ensemble's V_{dd} and f that maximize MIPS/w, assuming a single V_{dd} and f domain per ensemble to reduce complexity. To do so, we repeatedly pick one cluster i (which needs at least V_{ddMIN_i} and can only cycle at f_{max_i} at V_{ddMIN_i}). Then, we try to combine it with all of the possible groups of $|E|-1$ clusters among those that have a V_{ddMIN} lower than V_{ddMIN_i}. For each of these combinations, we set the ensemble's V_{dd} to V_{ddMIN_i} (which is the highest one), raise the f of each of the $|E|-1$ clusters accordingly, and set the ensemble's f to the minimum of the resulting f of all of the $|E|$ clusters. We then compute the MIPS/w. As we proceed, we pick the best selection for this i, and then the best selection over all possible i. We call this algorithm *MultipleVf_NC*. It has a higher overhead than *M_Assign* and scales with O(N^3).

5. The Challenge of Applying Fine-Grain DVFS

A second challenge of not using multiple V_{dd} domains is that, without on-chip voltage regulators, the speed at which a DVFS algorithm can change V_{dd} is lower. Specifically, according to Kim *et al.* [19], on-chip regulators can change V_{dd} at a rate of about 30mV/ns. On the other hand, off-chip regulators take over two orders of magnitude longer to change the same V_{dd} magnitude. For example, a very conservative estimate is given by Intel's guidelines, which assume that off-chip regulators take 1.25µs to change V_{dd} by 6.25mV [1, 27]. The reason for the lower speed is the higher latency of the communication between the CPU and the off-chip V_{dd} regulator. A V_{dd} regulator must sense the V_{dd} applied to cores to adjust its output V_{dd} accordingly; however, sensing the V_{dd} from off-chip is much slower than from on-chip. This limits the speed of V_{dd} regulators and thus requires a bulkier inductor and capacitor to support high efficiency. Overall, this inability to change V_{dd} fast could result in less energy-efficient DVFS operation under EnergySmart than under multiple V_{dd} domains.

In practice, however, there are several reasons why this issue is not likely to have a significant effect on the execution's energy efficiency. First, the V_{dd} changes needed at NTC are small most of the time. This is because even modest V_{dd} changes quickly bring the execution to regimes that are either too energy-inefficient or too slow. Moreover, the value of V_{dd} at NTC likely needs to be capped for reliability reasons. Secondly, the algorithm for selecting the DVFS levels in an environment with many V_{dd} domains will be complicated and have non-negligible overhead, discouraging very frequent use. Finally, the speed of DVFS changes is limited by the PLL relocking time for f change, which is at least 10µs [27]. Very fast V_{dd} regulators do not eliminate this critical path.

Aggressive PLL designs to hide the dead time due to relocking do exist [5]. The idea is to switch between multiple PLLs that are already preset to different frequencies. Unfortunately, this approach is not applicable to DVFS: we would need to provide a number of PLLs equal to the number of possible values that f can take, which is impractical. Overall, all the effects mentioned may hide the higher latencies of off-chip regulators. Section 7.4 quantifies the tradeoffs.

6. Evaluation Setup

We evaluate EnergySmart by modeling an 11nm NTC chip with 288 cores. The chip is organized in clusters. A cluster has 8 cores (each with a private memory) and a cluster memory. Table 2 shows the technology and architecture parameters. The nominal values of V_{dd} and f are 0.55 V and 1.0 GHz (which we estimate correspond to 0.77 V and 3.0 GHz for STC). To model variation, we use VARIUS-NTV [18]. Our baseline parameter values are $(\sigma/\mu)_{Vth} = 15\%$, $(\sigma/\mu)_{Leff} = 7.5\%$, and $\phi = 0.1$, although we vary the values for sensitivity analysis. These and other technology parameters are derived from ITRS and from projected trends from industry at 11nm. Every experiment is repeated for 100 chips with the same variation parameters but different variation profiles, and we present the average.

A cluster is the smallest clock domain. We use per-cluster PLLs. An abundant number of PLLs is already placed on chip to handle clock skew. The clock is routed as a tree.

Each core is a single-issue engine where memory accesses can be overlapped with each other and with computation. The per-core memories are private L1 caches, while the cluster memories are shared L2 caches. We use a full-mapped directory-based MESI coherence protocol where each pointer corresponds to one cluster. The network is a bus inside a cluster and a 2D-torus across clusters. The chip size is 400mm^2, which is comparable to large processor chips today [32].

To evaluate performance and power consumption, we interface Pin over a user-level pthreads library to the SESC [29] cycle-level architectural simulator. The power analysis relies

System Parameters	
Technology node: 11nm	P_{MAX} = 100W
Num. cores: 288	$T_{MAX} = 80^oC$
Num. clusters: 36 (8 cores/clus)	Chip area ≈ 20mm x 20mm
Variation Parameters	
Correlation range: $\phi = 0.1$	Sample size: 100 chips
Total $(\sigma/\mu)_{Vth} = 15\%$	Total $(\sigma/\mu)_{Leff} = 7.5\%$
Equal contribution syst. & rand.	Equal contribution syst. & rand.
Technology Parameters	
V_{ddNOM} = 0.55V	On-chip V_{dd} regulator: 5-25% loss
V_{thNOM} = 0.33V	V_{dd} guardband for V noise:
f_{NOM} = 1.0GHz	5% base
f network = 0.8GHz	+ 5% if multiple V_{dd} domains[17]
Architectural Parameters	
Core-private mem: 64KB WT,	Cluster mem: 2MB WB,
4-way, 2ns access, 64B line	16-way, 10ns access, 64B line
Network: bus inside cluster	Coherence: directory-based MESI
and 2D-torus across clusters	Avg. mem round-trip access time
Cross a f domain boundary: 2ns	(without contention): ≈80ns
Num. memory controllers: 8	

Table 2: Technology and architecture parameters.

on McPAT [22] scaled to 11nm. HotSpot is used to model the temperature. The algorithms for core assignment in the manycore are implemented in R [35].

For our experiments, we run multi-programmed workloads that contain the following PARSEC applications: blackscholes, ferret, fluidanimate, raytrace, swaptions, canneal, dedup, and streamcluster. Each application runs in parallel with a thread count that can range from 8 to 64 threads. We measure the complete parallel sections of the applications (i.e., region of interest) running the standard simsmall input data set. We report the average performance (e.g., in MIPS) or average energy efficiency (e.g., in MIPS/w).

7. Evaluation

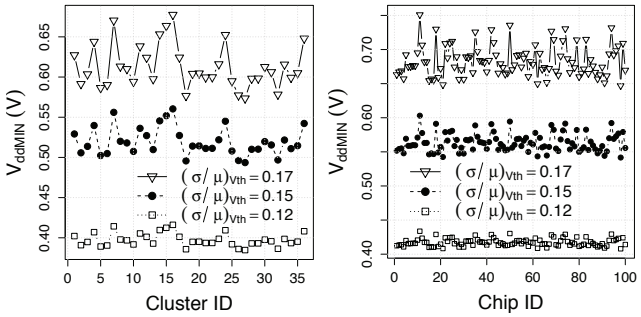
In this section, we first show the variation observed in NTC EnergySmart chips, and then examine the effect of not having multiple V_{dd} domains, the impact of the core assignment algorithms, and the implications on fine-grain DVFS. Finally, we perform a sensitivity analysis of the architecture.

7.1. Variation Observed in NTC EnergySmart Chips

We assess the variation in V_{ddMIN}, frequency (f), and static power (P_{sta}) in EnergySmart chips. We use three values of process variation, namely $(\sigma/\mu)_{Vth}$ = 12, 15, and 17%. For each value, we examine the variation across the 36 clusters of a representative EnergySmart chip, and the variation across 100 EnergySmart chips.

Figure 4(a) shows the variation in cluster V_{ddMIN} across the 36 clusters of an EnergySmart chip. Higher values of $(\sigma/\mu)_{Vth}$ cause not only a higher V_{ddMIN}, but also a higher variation in V_{ddMIN} across clusters. Figure 4(b) shows the variation in chip V_{ddMIN} (chip-wide maximum across all cluster V_{ddMIN}) across the 100 EnergySmart chips. Both the chip V_{ddMIN} and its variation across chips increase with $(\sigma/\mu)_{Vth}$.

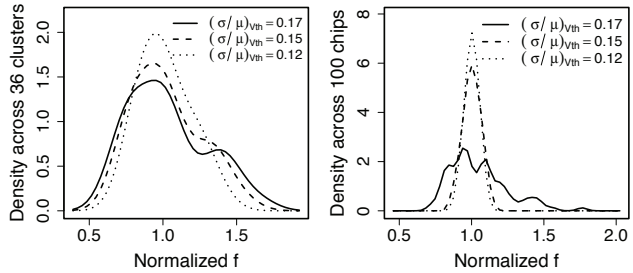
The variation in f and P_{sta} depends on the V_{dd} applied. In the next experiments, to show f and P_{sta} variation, we use as



(a) V_{ddMIN} variation within a chip (b) V_{ddMIN} variation across chips

Figure 4: Variation of V_{ddMIN}.

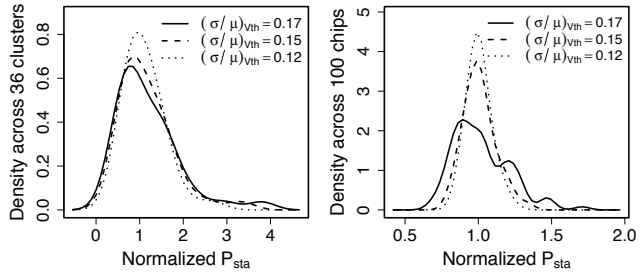
the safe, fixed V_{dd} per chip, the chip V_{ddMIN} for $(\sigma/\mu)_{Vth} = 0.17$. Figures 5 and 6 show the resulting f and P_{sta} variation, respectively. The plots show kernel density estimates for each parameter. They can be regarded as smooth correspondents of histograms, with an area under the curve equal to 1.



(a) f variation within a chip (b) f variation across chips

Figure 5: Kernel density of f.

In each figure, the leftmost plot shows the variation (of cluster f or of cluster P_{sta}) across the 36 clusters of a chip, while the rightmost one shows the variation (of chip-wide median cluster f or of chip P_{sta}) across the 100 chips. In each plot, the x axis shows the f or P_{sta} normalized to the non-variation afflicted counterpart. We observe that, as $(\sigma/\mu)_{Vth}$ increases, the spread of the distributions increases, which means that the variation in f or P_{sta} goes up.

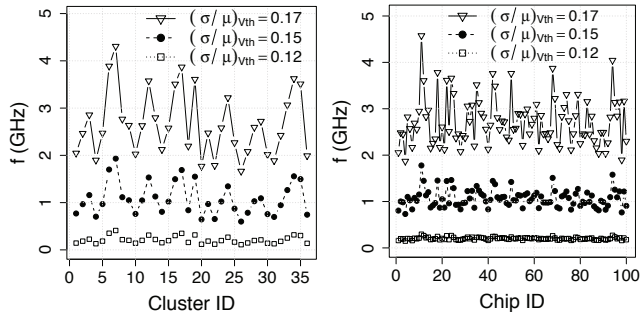


(a) P_{sta} variation within a chip (b) P_{sta} variation across chips

Figure 6: Kernel density of P_{sta}.

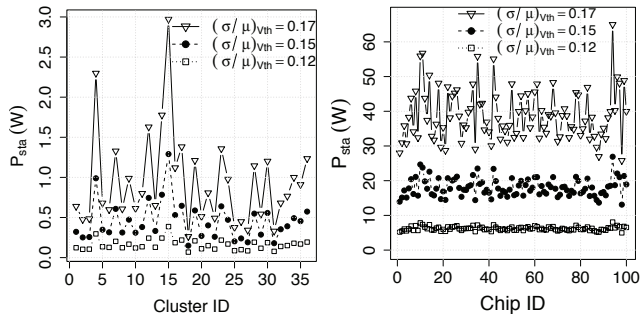
We now show the actual f and P_{sta} values, considering the V_{dd} at the corresponding $(\sigma/\mu)_{Vth}$, as opposed to the fixed V_{dd} across all the configurations, as used above. Recall that, as we increase $(\sigma/\mu)_{Vth}$, the value of chip V_{ddMIN} increases, to ensure correct operation.

Figures 7 and 8 show the variation in f and P_{sta}, respectively. In both cases, the leftmost plot shows variation within a chip while the rightmost one shows variation across chips. We see that progressively higher values of $(\sigma/\mu)_{Vth}$ induce both higher values and higher variation of f and P_{sta}. The rest of the paper uses $(\sigma/\mu)_{Vth}$ = 15% as the default.



(a) f variation within a chip (b) f variation across chips

Figure 7: Variation of *f*.



(a) P_{sta} variation within a chip (b) P_{sta} variation within chip

Figure 8: Variation of P_{sta}.

7.2. Impact of Not Having Multiple V_{dd} Domains

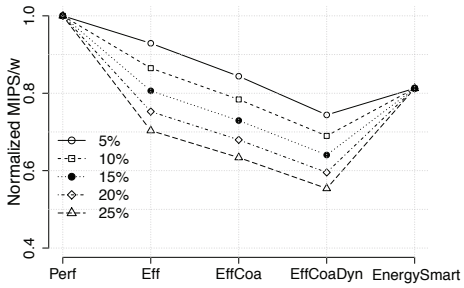
To see the impact of not having multiple V_{dd} domains at NTC, we compare the environments of Table 3. *Perf* is a perfect environment with a V_{dd} and an f domain per cluster, and no overheads to support the domains. *Eff* is *Perf* plus the power inefficiency of the on-chip V_{dd} regulators. While the power loss of on-chip regulation changes as a function of the output V_{dd}, we use the peak efficiency, independently of the output V_{dd}. We examine minimum inefficiencies in the 5–25% range. *EffCoa* is *Eff* with coarser V_{dd} domains (four clusters per domain) to reduce cost and complexity. *EffCoaDyn* is *EffCoa* plus a larger V_{dd} guardband. This is to tolerate deeper dynamic V_{dd} droops — resulting from the likely lack of averaging effects in the current drawn by the small domain, compared to a large chip with a single V_{dd} domain. Extrapolating from James *et al.* [17], we add an additional 5% V_{dd} guardband over the base 5% V_{dd}-noise guardband used in the chip with a single V_{dd} domain [19]. Finally, EnergySmart has a single V_{dd} domain.

All of the environments have off-chip V_{dd} regulators. Using on-chip regulators can contribute to higher efficiency for the off-chip regulators. However, correctly modeling such effect requires an analysis of many factors that would lengthen this paper substantially. Hence, we neglect it, and consider this effect as included in the final on-chip regulator's power inefficiency that we evaluate in our sweep.

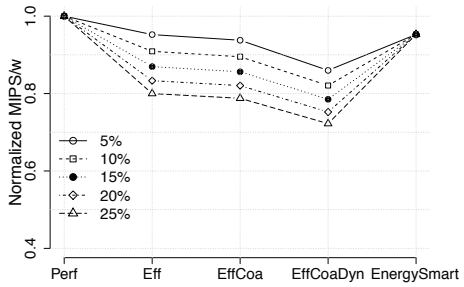
Environm.	Multiple V_{dd} Domains?	Description
Perf	Yes	Perfect environment with per-cluster V_{dd} and f domains. No overheads.
Eff	Yes	*Perf* + power loss due to on-chip V_{dd} regulation.
EffCoa	Yes	*Eff* with coarse V_{dd} domains of 4 clusters each.
EffCoaDyn	Yes	*EffCoa* + a larger V_{dd} guardband to handle deeper dynamic V_{dd} droops.
EnergySmart	No	Single V_{dd} domain. Each cluster is an f domain.

Table 3: Environments analyzed to assess V_{dd} domains at NTC.

Figure 9 compares the MIPS/w of our 288-core NTC chip for the different environments. We consider two scenarios: one where we use all 36 clusters (Figure 9(a)) and one where we use only 18 (Figure 9(b)). The workload consists of combinations of our PARSEC applications, where each PARSEC code runs with 8 threads on one cluster. We report the geometric mean of MIPS/w across clusters. In each plot, we perform a sensitivity analysis of different power inefficiencies for the on-chip V_{dd} regulators (5%, 10%, 15%, 20%, and 25%). To be fair, the total power consumed by the chip and on-chip V_{dd} regulators in all of the design points is kept constant.



(a) Full utilization: All 36 clusters



(b) 50% utilization: 18 clusters

Figure 9: Normalized MIPS/w in the different environments for workloads that use all 36 clusters (a) or only 18 (b). We consider different on-chip V_{dd} regulator inefficiencies.

For each environment, we compute the per-domain V_{dd} and f as per Section 3.2: SRAM hold and write stability determine V_{dd}, and then timing tests in the SRAM and logic determine f. In this section, we are after the MIPS/w that each environment delivers. To exclude sub-optimal operation due to algorithmic imperfections, we use oracular core assignment algorithms. The combinatorial optimization algorithms are based on the

Hungarian method [26]. We ignore any algorithmic overhead. Implicitly, this approach favors the non-EnergySmart environments, which require more complex core assignment algorithms due to the higher number of degrees of freedom (i.e., the multiple voltages per chip). Finally, after we compute the MIPS/w, we plot it normalized to that of *Perf*.

Consider first the fully-utilized chip (Figure 9(a)). EnergySmart only attains 81% of the MIPS/w of *Perf*. This is because it does not exploit the multiple V_{dd} domains of *Perf* and, therefore, operates less energy-efficiently. However, in the environments with multiple V_{dd} domains, as we go from *Perf* to *Eff*, *EffCoa* and *EffCoaDyn*, progressively adding more realistic assumptions, the MIPS/w keep decreasing. By the time we reach a realistic environment (*EffCoaDyn*), its MIPS/w at 15% regulator power loss is 64% of *Perf*'s — significantly lower than EnergySmart's.

As we vary the regulator power losses between 5% and 25%, EnergySmart always has a higher MIPS/w than the realistic *EffCoaDyn*. For the multiple V_{dd} domain chip to beat EnergySmart, it must not require any increase in V_{dd} guardband (*EffCoa*) and use regulators with only 5% power losses. Alternatively, it must support per-cluster V_{dd} domains, no increase in V_{dd} guardband (*Eff*), and use regulators with at most 10% power losses.

Figure 9(b) repeats the experiment when we only use half of the clusters. We see similar trends for all the non-*Perf* environments, except that the drop in MIPS/w is not as large. The reason is that each environment now picks a subset of clusters — leaving the most energy-inefficient ones idle. EnergySmart attains 95% of the MIPS/w of *Perf*, while *EffCoaDyn* only attains 79% at 15% regulator power loss.

7.3. Core Assignment in EnergySmart

The previous section showed that EnergySmart delivers higher MIPS/w than a realistic implementation of multiple V_{dd} domains. In this section, we explore how to best use an EnergySmart environment, by evaluating the EnergySmart core assignment algorithms of Section 4: *M_Assign_NC*, *M_Assign_C*, *Greedy_NC*, and *Greedy_C*. As a reference, we also show *MultipleVf_NC* applied to *EffCoaDyn* for three different V_{dd} regulator inefficiencies (5%, 10%, and 15%). In the previous section, the workload contained mixes of 8- threaded programs only, to expose the impact of per-cluster V_{dd} and f domains. Now, we consider a flexible workload composed of a mix of applications from PARSEC, such that one runs with 64 threads, one with 32, one with 16, and one with 8. The mix uses a total of 15 clusters, and we measure MIPS/w. We run many experiments, forming similar mixes with all of the possible permutations of the PARSEC programs, and report the average MIPS/w. The power budget is set to the worst-case power consumed by all 36 clusters.

Our EnergySmart algorithms are light-weight. For these experiments, *M_Assign_NC*, *M_Assign_C*, *Greedy_NC*, and *Greedy_C* execute, on average, 7835, 7912, 233, and 326 assembly instructions per run, respectively. The corresponding number for *MultipleVf_NC* is 153100.

Figure 10 shows the MIPS/w attained by the algorithms for three different manycore utilization scenarios. In the first one, all the clusters are available at the time the workload is launched (0% busy clusters). Hence, the whole power budget is available. The second scenario assumes that 25% of the clusters are already busy running another, existing load when we launched our workload. Such load was 8-threaded runs of our highest-IPC application, namely dedup. In the third scenario, 50% of the clusters are already busy. For the 25% and 50% scenarios, the figure augments the bars with ranges. The range top corresponds to when the busy clusters were the least MIPS/w-efficient ones; the range bottom when they were the most MIPS/w-efficient ones. This choice matters because it sets the power budget available. In the figure, all bars are normalized to *M_Assign_NC* with 0%.

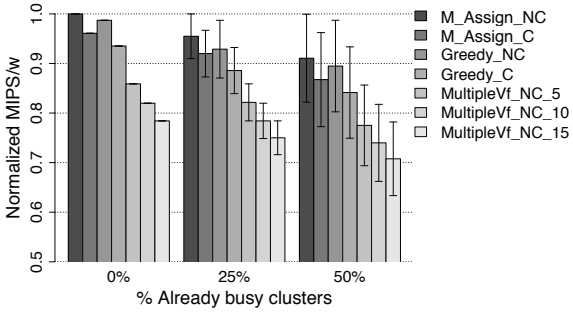


Figure 10: MIPS/w attained by different core assignment algorithms if 0%, 25%, 50% of the clusters were already busy initially.

Figure 10 shows that *M_Assign_NC* delivers the highest MIPS/w. *M_Assign_C* is about 4% worse. The reason is that the advantages of picking close-by clusters are offset by the fact that some of these clusters are not the most energy-efficient ones available. We also see that the greedy algorithms deliver only slightly lower MIPS/w than their *M_Assign* counterparts. Therefore, they can also be safely used.

The algorithm for the multiple V_{dd} domains (*MultipleVf_NC*) delivers significantly lower MIPS/w than *M_Assign*. For the 15% inefficiency (*MultipleVf_NC_15*), it delivers around 78% of the MIPS/w of *M_Assign_NC*. Even though *MultipleVf_NC*'s overhead is still tolerable for our chip size, it suffers because the energy efficiency of the *EffCoaDyn* manycore is lower. Overall, *M_Assign_NC* on EnergySmart is the most effective environment.

As we increase the fraction of the chip that was already busy with other work, the MIPS/w delivered by our workload goes down. This is because the power budget available is lower. However, the trends across environments remain. Not surprisingly, the MIPS/w delivered strongly depends on what clusters were already busy.

7.4. Implications on Fine-Grain DVFS

We now apply fine-grain DVFS to the environments of Table 3. We use a regulator inefficiency of 15%, which is in the ballpark of the designs of Kim *et al.* [20]. We consider different durations of the intervals between DVFS adaptations (from

0.1ms to 10ms) and two classes of workloads (one that uses 16 clusters and one 24 clusters). A workload consists of a group of 8-threaded applications from PARSEC, where each application runs on a cluster. For the experiments, we pick the most MIPS/w-efficient 16 or 24 clusters, since these are expected to respond with the highest f increase to a given V_{dd} increase (and to the corresponding power increase). Hence, these clusters are expected to deliver the highest MIPS with DVFS. Next, the applications are statically assigned to these clusters using the Hungarian algorithm [26].

Since using a specific DVFS algorithm can produce sub-optimal operating points, we use a best-case Oracle DVFS algorithm that relies on non-linear optimization in finding fs and V_{dd}s [6]. Moreover, we disregard any overhead involved in computing the next interval's V_{dd} and f — which favors the more complex multi-V_{dd} environments such as *EffCoaDyn*. Specifically, at the beginning of each interval, we assume we know the average IPCs of the applications in the interval. Then, we find the f and V_{dd} for each cluster (or a single V_{dd} in EnergySmart) that delivers the highest overall MIPS — while remaining within the chip's power envelope and within the V_{dd} cap (set to 20% higher than the V_{ddNOM} of EnergySmart).

We model the physical overheads of DVFS adaptation as described in Section 5. Specifically, in all environments, every interval that needs a f change induces a 10µs stall due to PLL relocking [27]. In addition, for the EnergySmart environment, we conservatively assume it takes 1.25µs to change V_{dd} by 6.25mV [1, 27]. We favor the multi-V_{dd} environments by assuming they can change V_{dd} instantaneously.

Figure 11 shows the resulting performance of fine-grain DVFS for the different environments and adaptation intervals. Charts (a) and (b) correspond to workloads that use 16 or 24 clusters, respectively. The performance for all the environments is normalized to *Perf* for 0.1ms and 16 clusters.

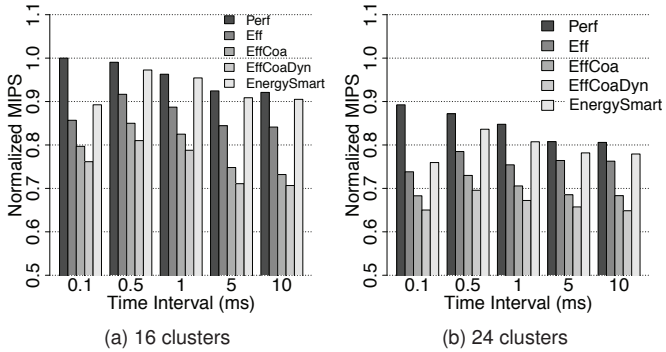


Figure 11: Performance of fine-grain DVFS.

The figure shows that, while EnergySmart's performance is lower than *Perf*'s, it is higher than that of the realistic environments with multiple V_{dd} domains, including *EffCoaDyn*. This is the case for all the adaptation intervals and the two chip utilization scenarios. For example, for 0.5–5ms intervals, EnergySmart's performance is 20-28% higher than *EffCoaDyn*'s for 16 clusters. This means that the higher latency of V_{dd} changes in EnergySmart due to having off-chip V_{dd} regulation is not able to outweigh EnergySmart's great advantage in energy efficiency. There are two reasons for this. First, the magnitude of the V_{dd} changes needed in NTC is modest. Second, the 10µs overhead of PLL re-locking is in the critical path of all the environments.

As the DVFS interval decreases, the DVFS algorithm tracks the application characteristics more accurately and the MIPS attained increases in all environments. However, as we reach very short intervals (0.1ms), the physical overheads of DVFS adaptation become noticeable, especially in EnergySmart, and the performance decreases. Still, with 0.1ms intervals, EnergySmart has higher performance than all the other non-*Perf* environments.

Finally, as we go from 16 to 24 clusters, the performance decreases across the board. The reason is that V_{dd} cannot go as high because more processors are now consuming power. Still, EnergySmart's performance is higher than the other non-*Perf* environments.

7.5. Sensitivity Analysis

Finally, we assess the impact of cluster granularity (core count per cluster) for a fixed chip core count, and the impact of chip core count for a fixed cluster granularity. We use a regulator inefficiency of 15%. The number of coarse-grain V_{dd} domains per chip (for *EffCoa* and *EffCoaDyn*) is fixed to 9.

Figure 12 compares a 288-core chip with either 4, 8 (default) or 16 cores per cluster. The results are organized as in Figure 9. We use all the clusters in Figure 12(a), and only half in Figure 12(b). Each environment is normalized to its *Perf*.

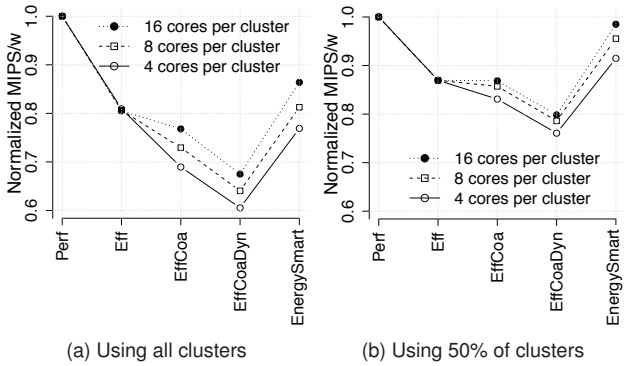


Figure 12: MIPS/w across different environments for a 288-core chip with 4, 8 (default) and 16 cores per cluster.

In all cases, EnergySmart's energy efficiency remains closer to *Perf* when compared to *EffCoaDyn*. As the cluster granularity increases, the difference in MIPS/w between *Perf* and the rest of the environments reduces, since there is less room for optimization. Under 50% utilization, the MIPS/w drop over *Perf* in all the environments reduces, since each environment can now pick an energy-efficient subset of clusters, leaving energy-inefficient ones idle.

Figure 13 compares 288-, 144- and 72-core chips with 8 cores per cluster. The results are organized as in Figure 9. We use all the clusters in Figure 13(a), and only half in Figure 13(b). Each environment is normalized to its *Perf*.

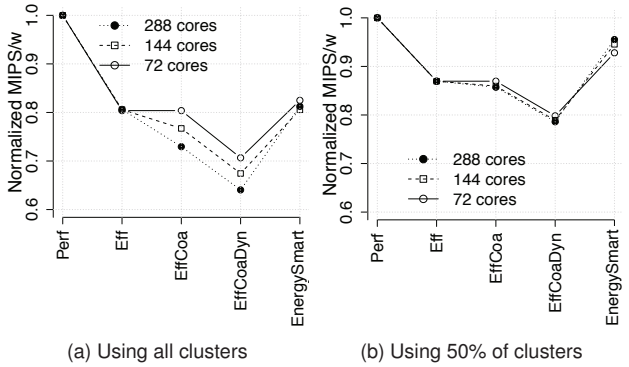


Figure 13: MIPS/w across different environments for 72-, 144- and 288-core chips with 8 cores per cluster.

In all cases, EnergySmart's energy efficiency is higher than *EffCoaDyn*'s. The MIPS/w of the environments tends to decrease relative to *Perf* with increasing core count. This is because, with more cores, there are more optimizations to be made. As usual, under 50% utilization, the MIPS/w drop over *Perf* in all the environments reduces.

8. Relationship to STC Chips

To understand how the ideas in this paper apply to STC, we considered an equivalent STC chip at the same 11nm node and 100W power envelope. Our analysis indicates that it contains 64 cores organized in 8 clusters, has a size of around 90mm^2, and its nominal V_{dd} and f are 0.77V and 3.0GHz. To enable DVFS, V_{dd} can change from 0.70V to 1.10V. Further, we estimate that in the *EffCoa* environment, a V_{dd} domain includes 2 clusters — for a total of 4 domains in the chip. Finally, we assume that the V_{dd} guardbands (the base one and the additional one for *EffCoaDyn*) have the same absolute value as in NTC. The workload is still one 8-threaded PARSEC application per cluster.

We see that the STC chip is quite different than its equivalent NTC one. In addition, the STC chip and its V_{dd} domains are used differently — which provides a different justification for the multiple V_{dd} domains. Specifically, at STC, we are not interested in maximizing the MIPS/w of the workload. That would result in an execution that is too slow. Instead, we want to maximize the performance (MIPS) of the workload for the given power budget of 100W. In addition, we are not interested in throughput computing, where all the threads are largely equally important. Instead, we assume that we are interested in speeding up one particular 8-threaded application (e.g., dedup, which is the one with the highest IPC). Hence, for that application, we use high V_{dd} and f, while we run the other applications at the lowest available V_{dd} (0.70V) and f (given by the variation profile). In this case, having multiple V_{dd} domains is especially useful. We consider this to be a realistic STC environment.

Figure 14 compares the performance of the application mix under the different environments. The performance is normalized to that of *Perf* and is shown for different on-chip V_{dd} regulator inefficiencies. We can see that the environments with multiple V_{dd} domains attain better performance than EnergySmart. This is because they drop the V_{dd} of the domains that are not running dedup to 0.70V, and increase the one running dedup to the maximum V_{dd} allowed by the power budget. The fs are changed accordingly. In *EffCoa*, the high V_{dd} domain runs one application at high f and the other at low f. EnergySmart is not competitive because it has to apply a high V_{dd} to all the clusters, wasting energy in the process.

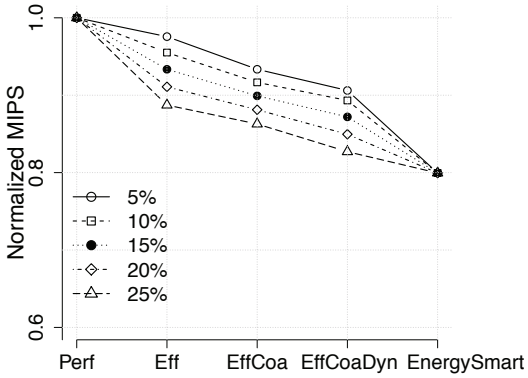


Figure 14: Normalized MIPS in different environments under STC.

9. Related Work

There is substantial work related to V_{dd} domains, especially in STC environments. Dighe *et al.* [11] studied variation-aware DVFS to optimize the energy efficiency of an Intel 80-core processor. They proposed V_{dd} domains with four cores per domain. However, they used a very different environment than ours, namely a 65nm STC chip with V_{dd}=1.2-0.8V. In their simulations, they did not consider any power loss due to V_{dd} regulation, or any increased V_{dd} guardband for multiple V_{dd} domains. Our emphasis on energy efficiency at NTC results in tighter V_{dd} guardbands and power budgets.

Rotem *et al.* [30] analyzed the impact of multiple V_{dd} and f domains under power delivery limitations, and proposed a clustered topology to maximize performance, but ignored variation. Yan *et al.* [37] used fast but power-inefficient on-chip V_{dd} regulation for applications requesting fast DVFS, and slow but power-efficient off-chip regulation otherwise. They used a 16–64 core chip and justified the area overhead of on-chip V_{dd} regulation by using dark silicon. EnergySmart is an NTC chip that has several hundred cores and no dark silicon.

Booster [24] eliminates on-chip V_{dd} regulators in an NTC chip with dual V_{dd} rails. In a 32-core chip, cores can switch between the two rails, running at two different per-core f. An on-chip governor determines how long each core spends on each rail, depending on its variation profile. The goal is to attain the same effective f across all the cores over a finite interval. EnergySmart targets a different environment, namely a several-hundred core chip, typically running multiple applications. In here, having all the cores appear homogenous is not the goal; instead, it is to attain the best variation-aware schedule. In addition, EnergySmart is simpler and more scalable, since it does not need the dual rails, or the circuitry and analysis to switch between them.

There is substantial work on application scheduling in a

variation-afflicted multicore (e.g., [28, 34, 36]). *M_Assign*'s contribution does not come from being used at NTC. It comes from being optimized for the EnergySmart architecture: it leverages EnergySmart's lack of V_{dd} domains, single f per ensemble (and per cluster), and whole-cluster assignment. Thanks to this, it is exact (not heuristic-based), has low overhead, and is scalable.

It is therefore more cost-effective than generic algorithms that scale to hundreds of cores. One such algorithm is the scalable hierarchical Hungarian algorithm by Winter *et al.* [36]. Even if we take such algorithm and set its number of V_{dd} domains to 1, it is not optimal for EnergySmart: (i) its core assignment neglects any inter-thread communication and (ii) it processes groups of cores independently and relies on locally-optimal schedules within each group. *M_Assign* uses parallel codes and performs a global schedule.

10. Conclusions

This paper proposed a new organization called *EnergySmart* for future energy-efficient NTC manycores. The idea is to keep a single V_{dd} domain and rely on multiple f domains to tackle variation. The removal of on-chip V_{dd} regulation results in energy efficiency, hardware simplicity, and area savings. EnergySmart can deliver higher performance per watt than a chip with multiple V_{dd} and f domains for three reasons: (i) conventional on-chip V_{dd} regulators have power losses; (ii) fine-grain V_{dd} domains need increased V_{dd} guardbands; and (iii) the resulting large domains include within-domain variations.

This paper also introduced core assignment algorithms tailored to EnergySmart that deliver high performance per watt and are simple. Finally, the paper showed that the higher latency of V_{dd} changes during DVFS without on-chip V_{dd} regulators is tolerable.

References

[1] *http://www.intel.com/Assets/en_US/PDF/designguide/321736.pdf.*

[2] J. Abella *et al.*, "High-Performance Low-Vcc In-Order Core," in *Intl. Symposium on High Performance Computer Architecture*, January 2010.

[3] K. Bernstein *et al.*, "High-Performance CMOS Variability in the 65-nm Regime and Beyond," in *IBM Journal of Research and Development*, July/September 2006.

[4] S. Borkar *et al.*, "Parameter Variations and Impact on Circuits and Microarchitecture," in *Design Automation Conference*, 2003.

[5] K. Bowman *et al.*, "Energy-Efficient and Metastability-Immune Resilient Circuits for Dynamic Variation Tolerance," *Journal of Solid-State Circuits*, January 2009.

[6] R. H. Byrd *et al.*, "A Limited Memory Algorithm for Bound Constrained Optimization," *SIAM Journal of Scientific Computing*, 1995.

[7] L. Chang *et al.*, "Practical Strategies for Power-Efficient Computing Technologies," *Proceedings of the IEEE*, February 2010.

[8] ——, "A Fully-Integrated Switched-Capacitor 2:1 Voltage Converter with Regulation Capability and 90% Efficiency at 2.3A/mm2," in *Symposium on VLSI Circuits*, June 2010.

[9] Y. Cheng and C. Hu, *MOSFET Modeling and Bsim3 User's Guide*. Kluwer Academic Publishers, 1999.

[10] R. Dennard *et al.*, "Design of Ion-Implanted MOSFETs with Very Small Physical Dimensions," in *Journal of Solid-State Circuits*, October 1974.

[11] S. Dighe *et al.*, "Within-Die Variation-Aware Dynamic-Voltage-Frequency-Scaling With Optimal Core Allocation and Thread Hopping for the 80-Core TeraFLOPS Processor," *Journal of Solid-State Circuits*, January 2011.

[12] R. R. Dobkin and R. Ginosar, "Two-phase Synchronization with Sub-cycle Latency," *Integration, the VLSI Journal*, June 2009.

[13] R. G. Dreslinski *et al.*, "Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits," *Proceedings of the IEEE*, February 2010.

[14] M. Eisele *et al.*, "The Impact of Intra-die Device Parameter Variations on Path Delays and on the Design for Yield of Low Voltage Digital Circuits," *IEEE Transactions on VLSI Systems*, December 1997.

[15] H. A. Ghasemi *et al.*, "Cost-Effective Power Delivery for Supporting Per-Core Voltage Domains for Power-Constrained Processors," in *Design Automation Conference*, June 2012.

[16] M. Horowitz *et al.*, "Scaling, Power, and the Future of CMOS," in *Intl. Electron Devices Meeting*, December 2005.

[17] N. James *et al.*, "Comparison of Split-Versus Connected-Core Supplies in the POWER6 Microprocessor," in *Intl. Solid-State Circuits Conference*, February 2007.

[18] U. R. Karpuzcu *et al.*, "VARIUS-NTV: A Microarchitectural Model to Capture the Increased Sensitivity of Manycores to Process Variations at Near-Threshold Voltages," in *Intl. Conference on Dependable Systems and Networks*, June 2012.

[19] W. Kim *et al.*, "System Level Analysis of Fast, Per-core DVFS Using On-chip Switching Regulators," in *Intl. Symposium on High Performance Computer Architecture*, February 2008.

[20] ——, "A Fully-integrated 3-Level DC/DC Converter for Nanosecond-scale DVS with Fast Shunt Regulation," in *Intl. Solid-State Circuits Conference*, February 2011.

[21] J. Lee and N. S. Kim, "Optimizing Total Power of Many-core Processors Considering Voltage Scaling Limit and Process Variations," in *Intl. Symposium on Low Power Electronics and Design*, March 2009.

[22] S. Li *et al.*, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," in *Intl. Symposium on Microarchitecture*, December 2009.

[23] D. Markovic *et al.*, "Ultralow-Power Design in Near-Threshold Region," *Proceedings of the IEEE*, February 2010.

[24] T. Miller *et al.*, "Booster: Reactive Core Acceleration for Mitigating the Effects of Process Variation and Application Imbalance in Low-voltage Chips," in *Intl. Symposium on High Performance Computer Architecture*, February 2012.

[25] S. Mukhopadhyay *et al.*, "Modeling of Failure Probability and Statistical Design of SRAM Array for Yield Enhancement in Nanoscaled CMOS," *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, December 2005.

[26] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[27] J. Park *et al.*, "Accurate Modeling and Calculation of Delay and Energy Overheads of Dynamic Voltage Scaling in Modern High-Performance Microprocessors," *Symposium on Low-Power Electronics and Design*, August 2010.

[28] K. Rangan *et al.*, "Achieving Uniform Performance and Maximizing Throughput in the Presence of Heterogeneity," in *Intl. Symposium on High Performance Computer Architecture*, February 2011.

[29] J. Renau *et al.*, "SESC Simulator," January 2005, http://sesc.sourceforge.net.

[30] E. Rotem *et al.*, "Multiple Clock and Voltage Domains for Chip Multiprocessors," in *Intl. Symposium on Microarchitecture*, December 2009.

[31] S. Sarangi *et al.*, "VARIUS: A Model of Process Variation and Resulting Timing Errors for Microarchitects," *Transactions on Semiconductor Manufacturing*, February 2008.

[32] J. L. Shin *et al.*, "A 40 nm 16-Core 128-Thread SPARC SoC Processor," *Journal of Solid-State Circuits*, vol. 46, no. 1, 2011.

[33] A. Srivastava *et al.*, *Statistical Analysis and Optimization for VLSI: Timing and Power*. Springer, 2005.

[34] R. Teodorescu and J. Torrellas, "Variation-Aware Application Scheduling and Power Management for Chip Multiprocessors," in *Intl. Symposium on Computer Architecture*, June 2008.

[35] The R Project for Statistical Computing:, *http://www.r-project.org/.*

[36] J. A. Winter *et al.*, "Scalable Thread Scheduling and Global Power Management for Heterogeneous Many-core Architectures," in *Intl. Conference on Parallel Architectures and Compilation Techniques*, September 2010.

[37] G. Yan *et al.*, "AgileRegulator: A Hybrid Voltage Regulator Scheme Redeeming Dark Silicon for Power Efficiency in a Multicore Architecture," in *Intl. Symposium on High Performance Computer Architecture*, February 2012.