Appears in the 3rd Workshop on Dependable Architectures (WDA-3)
In Conjunction with the 41st International Symposium on Microarchitecture (MICRO-41)
Lake Como, Italy, November 2008

Multicore Power Management: Ensuring Robustness via Early-Stage Formal Verification

Anita Lungu[1], Pradip Bose[2], Daniel J. Sorin[3], Steven German[2], and Geert Janssen[2]

[1]Dept. of Computer Science
Duke University
anita@cs.duke.edu

[2]IBM T.J. Watson Research Center
{pbose,sgerman,geert}@us.ibm.com

[3]Dept. of ECE
Duke University
sorin@ee.duke.edu

Abstract

Power management is important for multicore architectures. One important challenge for multicore DPM schemes is verifying that they are both safe (cannot lead to power or thermal catastrophes) and efficient (achieve as much performance as possible without exceeding power constraints). The verification difficulty varies among designs, depending, for example, on the particular power management mechanisms utilized and the algorithms used to adjust them. However, verification effort is often not considered in the early stages of DPM scheme design, leading to proposals that can be extremely difficult to verify.

To address this problem, we propose using formal verification (with probabilistic model checking) of a high-level, early-stage model of the DPM scheme. Using the model checker, we estimate the required verification effort, providing insight on how certain design parameters impact this effort. Furthermore, we supplement the verifiability results with high-level estimates of power consumption and performance, which allow us to perform a trade-off analysis between power, performance, and verification. We show that this trade-off analysis uncovers design points that are better than those that consider only power and performance.

1. Introduction

The prevalence of multicore architectures coupled with demands for low power systems motivate the development and evaluation of efficient power management solutions targeted specifically at multicores. Power is managed for several reasons, including to: improve power-efficiency, avoid power spikes, increase battery life, reduce the cost of providing power to the chip, and manage temperature. In this work, we investigate dynamic power management (DPM) schemes that can cap the peak power usage of a multicore. Providing a DPM scheme that caps the peak power can reduce system cost by decreasing the cooling and packaging requirements, or it can relax the power constraints placed on other system components.

One critical aspect in the development of a new DPM scheme is its verification. There are three properties that we wish to verify. First, we want to verify that the DPM scheme is *safe*. A DPM scheme can be unsafe, for example, if it allows the power usage to often exceed the allocated budget, or if it allows a core to be assigned a voltage or frequency outside of the desired range. Second, we wish to verify that the DPM scheme is *efficient* in achieving as much performance as possible while not exceeding power constraints or violating priority rules for provisioning power. A buggy DPM scheme might sacrifice more performance than expected. Third, we want to verify that the DPM scheme is *functionally correct*, such that the same results are obtained with and without the DPM scheme. In this paper, we consider verification of the first two features. As a concrete example of the importance of DPM verification, concerns over Intel's Foxton DPM scheme [16] led to it being disabled in the first Montecito chips [4].[1]

The current industrial workflow in the development of a new DPM scheme is illustrated in the unshaded portion of Figure 1. At an early stage, the focus is restricted to maximizing the efficiency of the DPM scheme, with limited consideration of its verification. Later, the scheme is implemented in detailed, low-level simulators, and verification[2] primarily checks whether the scheme achieves its efficiency goals.

The problem with this current workflow is that it is prone to missing bugs. First, simulation is by definition incomplete as a verification solution, because only the states that are reached in a particular simulation path are ascertained to be bug-free. Second, if verification feasibility is not considered at design time, the reachable state space of the resulting DPM scheme can be enormous, which is problematic. Workflows often have goals for achieving minimum coverage, so having more states

1. Intel has not officially stated whether the concerns were over safety, efficiency, or functionality bugs.

2. Using a simulator to "verify" a design is sometimes referred to as "validation" instead of verification.

requires more simulation cycles. If no coverage goal is specified, having more states increases the probability that undiscovered bugs remain in the design and decreases confidence in DPM correctness.

To address the above concerns, we propose the introduction of an additional, early step in the development of a new DPM scheme. We illustrate this added step in the shaded portion of Figure 1. This additional step creates, at an early design stage, a high-level model of the proposed power management policy which is then verified for efficiency and safety using probabilistic model checking, an exhaustive formal verification method. By performing a high-level verification early in the development process, we identify problems when they are easier to solve. A high-level model is also much easier to develop and modify than a detailed simulator, so we can quickly explore numerous designs.

With the use of the model checker, we estimate the effort required to verify the DPM scheme (measured as number of reachable states and transitions) enabling a better understanding of the impact on verification effort of scaling certain design parameters. Furthermore, we supplement the verifiability results with a high-level estimate of power consumption and performance, which enables us to perform a trade-off analysis between reaching power, performance, and verification goals.

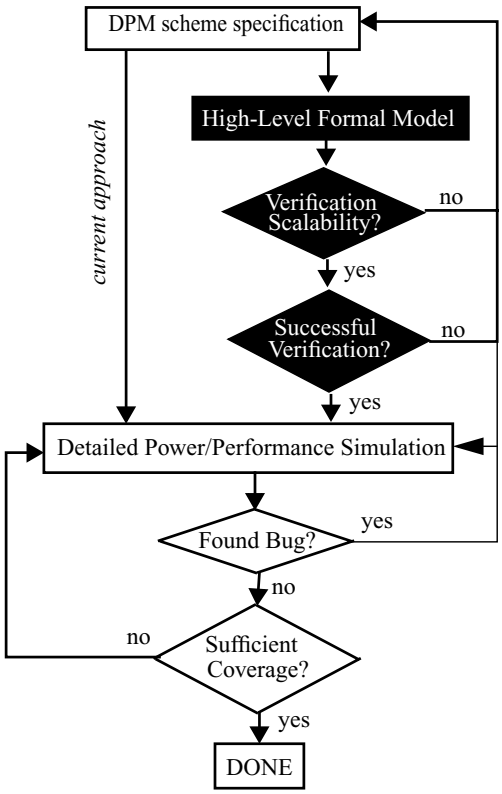


Figure 1. Workflow for Development of New DPM Scheme. Shaded portions indicate proposed additions.

Model checking does not eliminate the need to later simulate a detailed implementation of the DPM scheme, but it can catch bugs early and help the simulation reach desired state coverage goals.

Our main contributions are the following:

- We propose the use of verification effort as an additional metric to be considered, together with performance, in the early stages of DPM scheme design.
- We investigate and compare the effort necessary to verify different DPM algorithms as a function of the available mechanisms for adjusting power usage.
- We evaluate the trade-offs between verification effort, efficiency, and safety of the DPM schemes mentioned above.

The rest of this paper is organized as follows. In Section 2, we discuss related work. In Section 3, we present the type of DPM scheme we investigate and its parameters of interest. In Section 4, we explain our experimental methodology. In Section 5, we present our results, and we conclude in Section 6.

2. Background and Related Work

Power management is an important issue and thus there has been a significant amount of prior work in this area. In this section we first present multicore-specific power management schemes (Section 2.1). We then discuss prior work in power management verification (Section 2.2). Lastly, we discuss verification-aware design in general (Section 2.3).

2.1 Multicore Power Management

The most straightforward way to manage power in a multicore chip is to simply apply well-known single-core techniques to every core. However, Isci et al. [5] observed that such "local" (per-core) management was potentially inefficient because it could not take advantage of peak power averaging effects that occur across multiple cores. They introduce global schemes in which a single, centralized, "global" controller determines the power budget and settings (e.g., voltage and frequency) for every core. Sharkey et al. [18] provide a more detailed evaluation of these global schemes in terms of their efficiency. Sartori and Kumar [17] present a proactive scheme for managing peak power in multicore chips. They observe that distributed algorithms can be used to select the power level allocation for cores and that they would be more scalable than algorithms based on having a centralized global controller. However, no multicore DPM scheme has been analyzed to determine its verification effort and to trade-off verifiability against other design goals.

2.2 Verifying Power Management Schemes

There has been a limited amount of prior work in verifying DPM schemes. One representative piece of

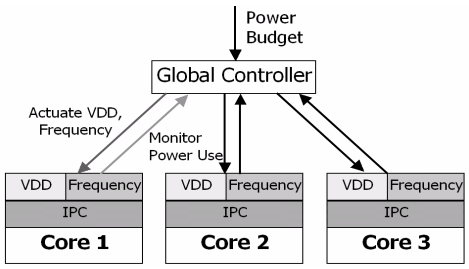


Figure 2. DPM Scheme with Global Controller

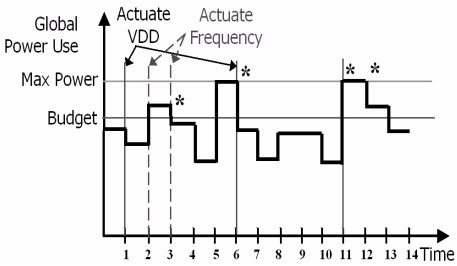


Figure 3. DPM Scheme Power Utilization

work by Shukla and Gupta [20] uses the SMV model checker [12] to verify a DPM scheme. We are interested in DPM for multicores, whereas their focus is on solutions for unicore systems. Furthermore, we use model checking to estimate verification effort and verify a set of correctness properties, while they use it to stress the optimality bounds of the DPM scheme by constructing a worst case task trace. Dubost et al. [3] present a high-level argument for specifying power management schemes in the Esterel language, which facilitates using a model checker to verify the designs. They do not discuss any specific DPM scheme or verification.

One interesting approach to DPM verification is the use of *probabilistic model checking*. With a traditional model checker, such as Murphi [2], one can prove absolute invariants. For example, one can prove that the power never exceeds a 50W power budget. However, with DPM, it may be tolerable that a 40W "soft power budget" is occasionally exceeded if that happens infrequently. Two recent research papers [15, 7] have used the PRISM probabilistic model checker [6] to analyze DPM schemes. They target unicore systems and use PRISM to find optimal power management policies for given task arrival distributions and constraints on expected wait queue size. In contrast, we are interested in analyzing the trade-off between verifiability and other metrics for multicore schemes.

2.3 Verification-Aware Design

Lungu and Sorin [8] quantified the effort required to formally verify parts of microprocessors. Martin [9] and Marty et al. [10] discussed the verification effort required for different cache coherence protocols. Our work differs from this prior work by focusing on power management schemes.

3. DPM Design Space Exploration

A wide variety of DPM solutions have been proposed in response to different requirements. In this section we describe the particular type of solution we analyze and its design parameters.

3.1 High Level View of DPM Design Space

We target DPM schemes that can cap the peak power usage of a multicore chip by using dynamic voltage and frequency scaling (DVFS). Figure 2 depicts the system we consider. The overall goal of the global DPM controller is to maintain the power usage of the system below the budget target set by a user (which could be the OS) with a minimum performance penalty. We use the expression "power budget" in a manner similar to prior work [5, 18]. The budget is the desirable power consumption level for the chip (shown in Figure 3). The budget differs from the Maximum Power for the chip, in that the budget is a somewhat soft limit. Exceeding the hard Maximum Power limit could lead to a thermal emergency and even burn the chip. However, exceeding the power budget occasionally, while still keeping the power below Maximum Power, can be tolerated. Budget overshoots cause the policy's goal to be temporarily unmet, but they cause no thermal emergencies. Recently developed DPM schemes also allow temporary budget overshoots [5, 18].

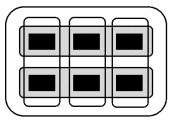
To keep the chip under its budget, the global controller periodically monitors the power usage of all cores and actuates their voltages and frequencies such that the total power consumption is maintained below the specified budget. We consider two actuation intervals: one for changing both voltage and frequency and one for changing only the frequency.

Figure 3 illustrates the power consumption of the chip over time. The Max Power horizontal line represents the maximum power the chip can consume given the worst case activity factors of all cores. The Budget line represents the constraint imposed on the power use of the chip. The global controller uses this power budget value as the target for its feedback mechanism. In setting the voltage and frequency levels, the global controller makes the prediction that the cores will maintain their current activity factors for the next interval. When this is a misprediction, the actual power use can temporarily overshoot, as shown in Figure 3 at the times marked with stars. On the next actuation point the controller tries again to bring the power use below budget.

3.2 Design Goals and Parameters

Of the multiple design goals that such a DPM scheme can target, we investigate *efficiency* (reducing the performance hit induced by decreasing core fre-

Figure 4. Possible Assignments of Cores to Controllers



quency through DVFS), *safety* (decreasing time and power spent over budget) and *verifiability* (decreasing required verification effort).

To reach these goals, designers can make decisions on many parameters. We consider here only a subset of them to keep our analysis tractable. Specifically, we compare a heterogeneous policy, which allows the controller to assign different voltage and frequencies across the cores, to a homogeneous policy, where the same voltage and frequency is set for all cores. For both policies, we analyze the design space along 3 parameters: number of voltage levels (**VL**) into which the voltage range is split, number of frequency levels (**FL**) that can be allocated for a given voltage level, and number of cores assigned to a single DPM controller. Figure 4 illustrates this cores per controller (**CPC**) design parameter. If we consider a 6-core chip, a DPM solution might use a single controller assigned to all chips (the outer boundary), or 2 controllers each monitoring 3 cores (the two horizontal groupings), or 3 controllers each supervising 2 cores (the three vertical groups).

3.3 Motivating Early Formal Analysis

Designers certainly have some intuitive *a priori* understanding of how choosing different design points in the above parameter space affects their goals. For example, one might expect that a heterogeneous solution with more CPC will outperform a solution with fewer CPC, because the peak power use of more cores should be decreased due to averaging effects. But what is the *quantitative* gain in performance when going from 2 CPC to 3 CPC, for example? Is that performance gain worth the impact on verification effort? How does the safety of the solution change in response to CPC? Do the answers vary between homogeneous and heterogeneous policies? In addition to questions about CPC, designers want to answer similar questions about other parameters, such as VL and FL, and possible interactions between parameters. Will a change in VL impact design goals differently depending on the value of CPC?

These are the type of questions to which we seek answers via performing the proposed early stage formal analysis. These answers enable designers to make more informed decisions, and we show concrete examples of these benefits in Section 5.

4. Methodology for Formal Analysis

We begin this section with our motivation for using probabilistic model checking to verify the analyzed DPM schemes and a brief overview on this method. Then we provide details on the particular methodology we use to conduct our experiments.

4.1 Probabilistic Model Checking

We use probabilistic model checking with PRISM [6] to explore the design space of our DPM schemes and analyze trade-offs between efficiency, safety, and verifiability.

Using a model checker allows us to quantify the verification effort for a system. We chose a model checking tool over a simulator because a model checker is a complete verification solution which traverses the entire reachable state space of a design in ascertaining correctness. In contrast, a simulator is incomplete because it touches only a limited subset of all reachable states. We obtain a better verifiability measure for a design when we can exercise its entire reachable state space and all state transitions. The choice of probabilistic model checking over traditional, non-probabilistic model checking was motivated by characteristics of the problem we want to verify. For the verification of a DPM scheme we are not only interested whether a power overshoot can happen, but also how often this is expected to happen under typical conditions. These types of correctness characteristics depend on the changing activity factor of the workloads, which can be captured in a probabilistic framework.

The inputs to the probabilistic model checker are: the *state elements* of the system, the *probabilistic transition rules* (a description of how the behavior can change from one state to the next), and the *correctness properties* (the requirements which, if met, assure the system's correctness). In addition, it is possible to evaluate the expected values of certain quantities in the system, such as power and performance, by associating *rewards* with system states. Rewards are similar to tokens, in that the states that satisfy a certain condition are assigned tokens. It is not our goal to use model checking for a better estimate of power usage and performance impact; rather, we use the rewards to obtain high-level measures of power and performance and analyze their trade-off with verifiability. Based on the probabilistic state machine description, the model checking tool traverses the entire reachable state space of the design and verifies whether the correctness properties are met. When rewards are specified it also calculates their expected values over a certain bounded number of system transitions.

4.2 DPM Model Construction

For our DPM scheme, the *state elements* are: the current voltage, frequency, and activity factor of each core and an incrementing counter triggering when the global controller should actuate both voltages and frequencies as opposed to only frequencies.

Table 1. Microprocessor Configuration

Feature	Description
pipeline width	4 decode/issue/commit
ROB/LSQ sizes	150 entries / 32 entries
branch pred.	2 level, 3 16K-entry BHTs
functional units	4 FXU, 4 FPU, 1 BR
L1I cache	64KB, 2-way, 16B blocks, 1cycle
L1D cache	64KB, 2-way, 16B blocks, 1cycle
L2 cache	1MB, 8-way, 64B blocks, 9 cycles
memory	100 cycles

Table 2. Benchmarks

	Low Ave IPC	High Ave IPC
Stable IPC	*mcf*	*eon, crafty*
Variable IPC	*art, parser*	*bzip2*

The *probabilistic transition rules* specify how the activity factor changes for the cores and how the voltages and frequencies change in response to controller actuations. We approximate each core's activity factor using its instructions per cycle (IPC), because IPC is strongly correlated with the activity factor and it is easy to obtain. This correlation is not perfect, but obtaining the exact activity factor would require a low-level implementation that is unlikely to exist early in the design cycle. To make our analysis tractable with PRISM, we quantize the IPC values into four distinct ranges, and we choose the mean IPC of a range to represent the activity factor of a core in that range.

We obtain the transition probabilities using Turandot [13], a detailed, cycle-accurate simulation model. The microprocessor's configuration is shown in Table 1. For benchmarks, we chose six SPEC 2000 benchmarks, shown in Table 2, that have very different behavior, both in terms of their average activity factor and in how much their activity factor changes over time. The appropriate SimPoint [19] intervals for these benchmarks were traced using Aria [14]. For each benchmark, the simulator produces the average IPC for each time quantum of 100µs (400,000 cycles at 4GHz). The sampling period of 100µs reflects the safe specification parameter of the power manager, in terms of the longest duration of allowable power spikes. Given that chip-level thermal time constants are in the range of milliseconds or tens of milliseconds [1], 100µs is a very safe, conservative setting of this parameter.

We wish to point out that we obtain the benchmark IPC values from a simulation of a single-core processor, rather than from a simulation of a multicore processor. The intuitive reason for this decision is that PRISM will inherently construct all possible combinations of IPCs and IPC transitions for all cores running the benchmarks.[3] Moreover, it is not obvious that we even *could* simulate every possible combination, since it is extremely difficult to compel the simulated system into each combination of core states.

4.3 DPM Scheme Properties

We verify the behavior of the system against a set of correctness properties that must be true in every state. We also specify a set of reward structures that enable us to quantify performance, power use, and safety.

Correctness properties. The correctness properties we consider for our DPM scheme are:

- No deadlock state can ever be reached.
- The voltages and frequencies for all cores are always maintained within a pre-specified range.
- There is no mismatch between the voltage and frequency assigned to a core (e.g., we never match a very high frequency with a very low voltage).

Reward structures. We use rewards to keep track of power, performance, and the states in which the system is over budget. PRISM computes the expected rewards over a bounded interval, and we set the bound to 1000 transitions in our experiments.

4.4 Quantifying Performance, Power, Safety, and Verifiability

We now describe the models and metrics we use to quantify performance, power, safety, and verifiability for our early stage formal analysis.

Performance. In our model, the performance of a core is a linear function of its frequency, *f*. That is, if we increase *f* by X%, then the performance is also improved by X%. This is an approximation, because the performance benefit of a large increase in *f* is limited by the unchanged memory performance. Nevertheless, for a high-level model that is considering small adjustments in *f*, we think this assumption is reasonable.

Our model considers the latency required to transition between voltage levels, and it assumes that a core functions at its lowest frequency during a voltage transition (1µs per 10mV). The latency of transitioning between frequency levels is much shorter—on the order of one or two processor cycles [11]—because it can be done with on-chip digital PLL mechanisms. This latency is orders of magnitude shorter than a 100µs actuation interval, and thus we do not model it.

3. One caveat is that a simulation of a multicore chip might (a) exhibit transitions that are *never* exhibited by a single-core chip, or (b) *never* exhibit transitions that are exhibited by a single-core chip. These scenarios, although unlikely, could result from contention for resources that occurs in multicore chips.

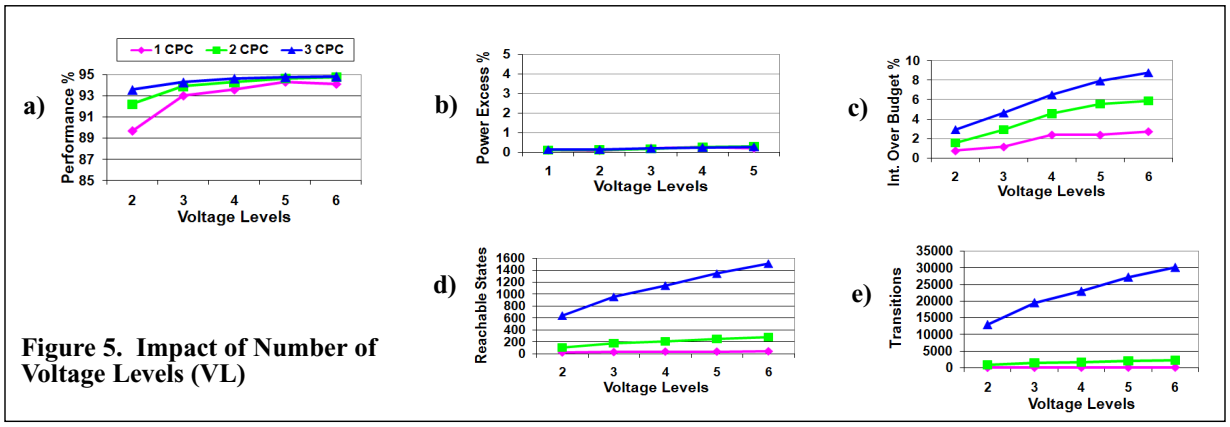


Figure 5. Impact of Number of Voltage Levels (VL)

Power. In our model, the power consumption of a core is a function of the core's frequency (f), voltage (V), and activity factor (A). We model both active and leakage power, with active power consumption formulated using the usual $\sim f*A*V^2$ dependence equation. The leakage power is modeled approximately as a cubic function of V, as this has been found to capture the behavior quite well for the particular supply and threshold voltage ranges appropriate for current CMOS technologies (65nm or 45nm). The power model used is admittedly abstract, but deemed to be good enough for the DVFS-driven power management policies considered in this paper (as in Isci et al. [5] or Sharkey et al. [18]).

Safety. We consider two safety metrics: the percentage of time the system is expected to be over budget, and the percentage of power used over budget.

Verifiability. We consider two metrics for quantifying verification effort. The first is the total number of reachable states of the design. The second is the number of possible transitions between states.

Because we use a simulator to generate the state transition probabilities, our performance and safety results are a function of the benchmark suite, because they depend on rewards computation. The verifiability results are also a function of benchmark suite as the number of reachable states and transitions depends on the changing behavior of the applications. For benchmarks with radically different behavior, these results might be different. We state this perhaps obvious characteristic of our work—after all, benchmark dependence is common in microarchitectural studies—because it differs from traditional (non-probabilistic) model checking. Note that the correctness properties mentioned in Section 4.3 are proved correct independent of the benchmark suite.

5. Experimental Evaluation

We now detail the two specific DPM schemes we modeled for our analysis and their design parameters. Then we describe the performance, safety, and verifiability trade-offs we find in this design space.

5.1 Scope of Analysis

We analyze *heterogeneous* and *homogeneous* DPM schemes. For the heterogeneous schemes, the controller uses a priority based greedy algorithm for distributing the power budget. It allocates the largest voltage that fits in the power budget for the first core (while provisioning enough power to run the rest of the cores at lowest voltage) then allocates the largest possible voltage for the second core and so on. This heterogeneous policy is very similar to current state-of-the-art DVFS policies, such as the "Priority" scheme analyzed by Isci et al. [5]. For homogeneous schemes, the controller allocates the single greatest voltage level that keeps the chip below the power budget, assuming all cores maintain their current activity factors. This homogeneous policy is very similar to the "Chip-Wide DVFS" scheme proposed by Isci et al. [5].

All of our DPM schemes use two actuation intervals: a 500µs one to change both voltage and frequency of cores (the frequency is set to the highest value permitted for the voltage level selected) and a 100µs one to change only the frequency. We vary the voltage range from 1.05V to 0.78V and we scale the frequencies linearly with the voltage from 4.2GHz to 3.15GHz.

When analyzing the impact of increasing VL, we maintain the same voltage range and divide it into more levels (from 2 to 6 in our experiments). When varying FL, we divide the frequency range corresponding to a particular voltage level into more values (from 1 to 5). We also vary CPC from 1 to 3. Note that this is different from comparing a 1-core chip to a chip with 2 or 3 cores; we consider a chip with the same number of cores, 6 for example, which has 6, 3 or 2 controllers.We do not model a 6-core system with a single controller (having CPC of 6) because the associated state explosion makes the verification through model checking

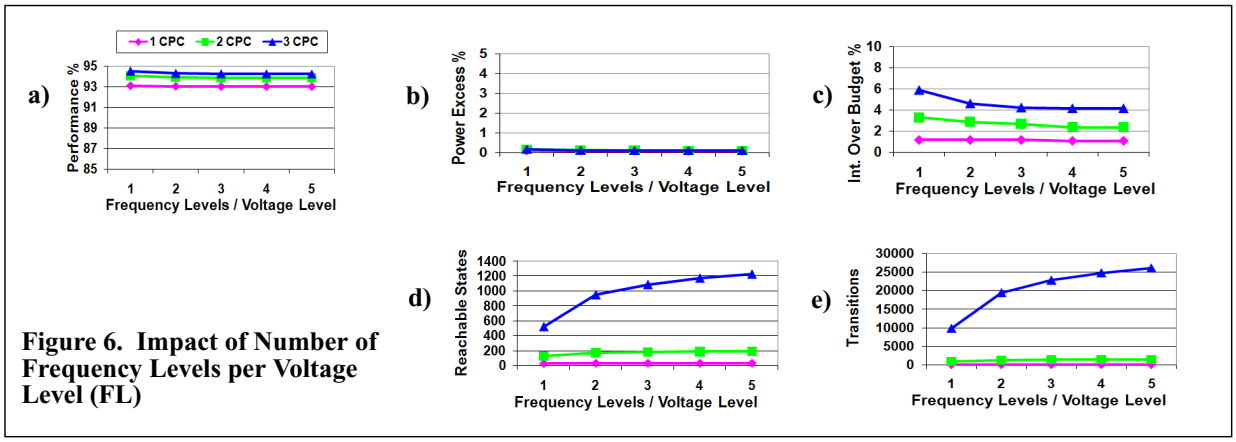


Figure 6. Impact of Number of Frequency Levels per Voltage Level (FL)

impractical and our results show little overall performance improvement beyond 3 CPC.

In our analysis, the global controller uses the power model described in Section 4.4 to estimate the power use of the system (a function of activity factor, voltage and frequency). The global controller predicts that the cores will maintain their current activity factor during the next interval.

We perform a range of experiments setting the power budget to 25, 40, 50, 70 and 100% of the maximum power the chip can consume (corresponding to a 4 IPC activity factor across all cores). The results we present are averaged across the different budget levels and benchmarks.

5.2 Impact of Number of Voltage Levels

The first design parameter we explore is VL. We consider a heterogeneous scheme and fix FL to 2 for clarity (the results were similar for the other FL values). Figure 5(a) shows the impact of VL on performance with respect to a chip without DPM. Figure 5(b,c) show safety, and Figure 5(d,e) show verifiability. We notice a strong interaction between VL and CPC; on many of our metrics of interest, the impact of increasing VL varied across different levels of CPC. Hence we present data for CPC=1, 2 and 3 on the same graph.

We notice several interesting phenomena. First, in terms of performance, the trend corroborates our intuition that increasing VL benefits performance. However, we notice a saturation around VL=5 and performance remains almost flat afterwards. Prior work [17] proposed using VL=10 in an experimental setup that used 4 cores, simulating various SPLASH benhmarks. Our results, albeit in a different setup, suggest that such a large value of VL offers little marginal benefit.

The impact of CPC on performance also matches our intuition in that we achieve better performance by increasing CPC. In fact the CPC=1 solution lags behind the CPC=2 and CPC=3 solutions at all voltage levels. However, the difference between the CPC=2 and CPC=3 solutions is minimal. They differ somewhat for low values of VL (2 or 3) but after that point there is very little difference in performance. The intuition is that the presence of 2 cores with activity factors that differ achieves a good enough average effect on the aggregate peak power to make throttling unnecessary. In prior work [5], the authors foresaw the motivation and need for centralization of the multicore power management problem. In this work we have seen that centralization is indeed better than local per-core control, but clustering of cores per controller beyond two may not yield additional performance. This insight is an important additional input to future architectural design of multicore power management protocols.

In terms of safety, the percentage power spent over budget is minimal, ranging from 0.1% to < 0.5% of the power usage of a solution without DVFS. The percentage of intervals spent over budget varies from ~0.5% to ~9%. An increase in CPC allows the controller to make more aggressive decisions in matching the power budget resulting in more mispredictions. The same can be said about increasing VL. Whether the amount of time spent over budget is deemed tolerable or not depends on the particular constraints of the application. However, considering the tiny percentage of power spent over budget, we conclude that VL does not greatly impact safety.

Given only the performance and safety analysis of the design space, one might conclude that the greatest difference can be noticed when going from CPC=1 to CPC=2 and that there is a minimal difference between CPC=2 and CPC=3. However, if we add verifiability to the picture, the conclusion changes dramatically. The verification effort, measured both in number of reachable states and transitions, increases dramatically with CPC. We see a strong interaction between CPC and VL in terms of verifiability effects. For both the CPC=1 and CPC=2 solutions, the verification effort does not

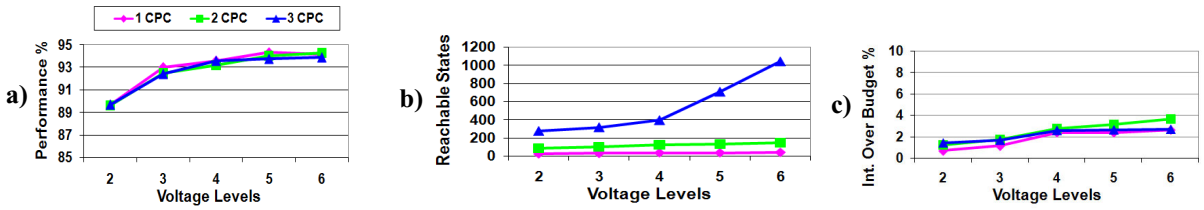


Figure 7. Impact of Homogeneous Policy

increase significantly with VL, unlike the case for the CPC=3 solution.

In conclusion, the performance improvement gained from going from CPC=2 to CPC=3 is insignificant (particularly for larger VL) while the increase in verification effort is extremely large. Our data suggest that the better design solution consists of having multiple controllers each assigned to a small number of cores (2) which can be set to 4-5 voltage levels as opposed to a design with a large CPC at low VL.

5.3 Impact of Number of Frequency Levels

The second design parameter we address is FL, the number of frequency levels that can be set for a given voltage level. Our hypothesis was that the 100µs actuation of the controller can take advantage of the increased frequency granularity and better track the power budget between consecutive voltage actuations.

Figure 6 shows our results when we consider a heterogeneous policy and fix VL=3 for performance with respect to a chip without DPM (a), safety (b, c) and verifiability (d, e). Our results indeed show a slight improvement in safety due to the increased flexibility in frequency levels. However, this improvement is minimal and accomplished with a performance penalty. The reason is that the frequency decrease is a lot less efficient in decreasing the overall power usage than the voltage. The impact of FL on verification, however, is very large both in reachable states and transitions. We conclude that the frequency knob should be used only when the safety margins of being over budget are tight, because a significant cost in verifiability will be paid. Also, FL=2 seems to suffice for getting most of the safety benefit. Our conclusion is specific to the type of system we analyzed, where it is possible to set both voltage and frequency of individual cores at different levels. For this case, using many frequency levels for one voltage level does not seem to represent a good design alternative from a verifiability, performance, and safety trade-off. For the class of systems that allocate the same voltage across all cores, the impact of frequency levels is likely to be more beneficial.

5.4 Impact of Using a Homogeneous Policy

We now explore the impact of choosing a homogeneous policy. We wish to discover whether homogeneity helps or hurts our pursuit of better design points. Figure 7 shows the results for a homogeneous policy when we vary VL. We notice a slight decrease in performance for an increase in CPC. This result is due to the fact that the homogeneous policy is more restrictive and all cores assigned to the controller are throttled to a single voltage level to match the budget. Second, the performance impact of increasing VL is more significant compared to the heterogeneous case. The safety is improved for the homogeneous solution as the percentage of intervals spent over budget decreases significantly.

6. Conclusions

Power management is important for multicore processors, and DPM scheme designers would like to have confidence that their schemes are both safe and efficient. We have shown the insight that can be gained by using formal methods—in this case, probabilistic model checking—to analyze high-level descriptions of DPM schemes. We have used PRISM to determine the effort required to verify DPM schemes, and we have compared these schemes with respect to their efficiency.

One conclusion we draw from this work is that global schemes (i.e., CPC>1) offer significant benefits in performance due to the ability to balance power across more cores. However, we must be careful to avoid scaling them to more cores than necessary. Linear increases in CPC cause exponential increases in the size of the reachable state space. Thus it is important to find the system configuration where both the verification is tractable and we obtain the majority of the benefits of a global solution. Our data shows that much of the benefit is achieved at just CPC=2; increasing CPC further provides little additional performance gain. In terms of safety, we found no significant difference between percentage energy spent over budget as a function of CPC, but a larger value of CPC resulted in the system spending more time over budget. Thus we recommend designs

in which chips are divided into small clusters of cores, where each cluster uses a global control scheme.

A second conclusion is that the use of fine-grained frequency tuning is likely not worth its costs for systems where it is possible to set both voltage and frequency of individual cores at different levels. The results show that having a large FL has an extremely large impact on verification effort. It is not clear that its modest safety benefits justify these verification costs.

Acknowledgments

This work was initiated as a 2007 summer internship project at IBM T. J. Watson Research Center. The work at IBM was supported in part by the Defense Advanced Research Projects Agency under its Agreement No. HR0011-07-9-0002. At Duke University this research was supported by the National Science Foundation under Grants CCF-0444516 and CCF-0811920. We thank Alvy Lebeck and Costi Pistol for helpful discussions about this work.

References

[1] J. Choi et al. Thermal-aware Task Scheduling at the System Software Level. In *Proc. of the Int'l Symposium on Low Power Electronics and Design*, Aug. 2007.

[2] D. L. Dill, A. J. Drexler, A. J. Hu, and C. H. Yang. Protocol Verification as a Hardware Design Aid. In *1992 IEEE Int'l Conference on Computer Design: VLSI in Computers and Processors*, pages 522–525, 1992.

[3] G. Dubost, S. Granier, and G. Berry. An Esterel-Based Formal Specification Methodology for Power Manager Development. Presented at the SAME Forum, Oct. 2007.

[4] D. Dunn. Intel Delays Montecito in Roadmap Shakeup. *EE Times*, October 24 2005.

[5] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi. An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance for a Given Power Budget. In *Proc. of the 39th Annual IEEE/ACM Int'l Symposium on Microarchitecture*, Dec. 2006.

[6] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 2.0: A Tool for Probabilistic Model Checking. In *Proc. of the 1st Int'l Conference on Quantitative Evaluation of Systems*, pages 322–323, Sept. 2004.

[7] M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic Model Checking and Power-Aware Computing. In *Proc. of the 7th Int'l Workshop on Performability Modeling of Computer and Communication Systems*, pages 6–9, Sept. 2005.

[8] A. Lungu and D. J. Sorin. Verification-Aware Microprocessor Design. In *Proc. of the Int'l Conference on Parallel Architectures and Compilation Techniques*, pages 83–93, Sept. 2007.

[9] M. M. K. Martin. Formal Verification and its Impact on the Snooping versus Directory Protocol Debate. In *Proc. of the Int'l Conference on Computer Design*, Oct. 2005.

[10] M. R. Marty et al. Improving Multiple-CMP Systems Using Token Coherence. In *Proc. of the Eleventh Int'l Symposium on High-Performance Computer Architecture*, pages 328–339, Feb. 2005.

[11] R. McGowen et al. Power and Temperature Control on a 90-nm Itanium Family Processor. *IEEE Journal of Solid-State Circuits*, 41(1):229–237, Jan. 2006.

[12] K. L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, 1993.

[13] M. Moudgill, P. Bose, and J. H. Moreno. Validation of Turandot, a Fast Processor Model for Microarchitecture Exploration. In *Proc. of the IEEE Int'l Performance, Computing and Communications Conference*, pages 451–457, Feb. 1999.

[14] M. Moudgill, J.-D. Wellman, and J. H. Moreno. Environment for PowerPC Microarchitecture Exploration. *IEEE Micro*, 19(3):15–25, May/June 1999.

[15] G. Norman, D. Parker, M. Kwiatkowska, and S. Shukla. Using Probabilistic Model Checking for Dynamic Power Management. *Formal Aspects of Computing*, 17(2):160–176, Aug. 2005.

[16] C. Poirier, R. McGowen, C. Bostak, and S. Naffziger. Power and Temperature Control on a 90nm Itanium-family Processor. In *Proc. of the IEEE Int'l Solid-State Circuits Conference*, Feb. 2005.

[17] J. Sartori and R. Kumar. Proactive Peak Power Management for Many-Core Architectures. Technical Report CRHC-07-04, UIUC CRHC, Oct. 2007.

[18] J. Sharkey, A. Buyuktosunoglu, and P. Bose. Evaluating Design Tradeoffs in On-Chip Power Management for CMPs. In *Proc. of the Int'l Symposium on Low Power Electronics and Design*, Aug. 2007.

[19] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder. Automatically Characterizing Large Scale Program Behavior. In *Proc. of the Tenth Int'l Conference on Architectural Support for Programming Languages and Operating Systems*, Oct. 2002.

[20] S. Shukla and R. K. Gupta. A Model Checking Approach to Evaluating System Level Dynamic Power Management Policies for Embedded Systems. In *Proc. of the High-Level Design Validation and Test Workshop*, pages 53–57, 2001.