PTSMT: A Tool for Cross-Level Power, Performance, and Thermal Exploration of SMT Processors

Deepa Kannan†, Aseem Gupta‡, Aviral Shrivastava†, Nikil D. Dutt‡, Fadi J. Kurdahi‡

† Compiler and Microarchitecture Laboratory, Arizona State University, Tempe, AZ 85281 USA
{deepa.kannan, aviral.shrivastava}@asu.edu
‡Center for Embedded Computer Systems, University of California Irvine, CA 92697 USA
{aseemg,dutt,kurdahi}@uci.edu

Abstract

Simultaneous Multi-Threading (SMT) processors are becoming popular because they exploit both instruction-level and thread-level parallelism by issuing instructions from different threads in the same cycle. However, the issues of power and thermal management hinder SMT processors fabricated in nano-scale technologies. Power and thermal issues in SMT processors not only limit the achievable performance, but also have a direct impact on the cost and viability of these processors. While several performance simulation tools to explore the performance aspect of SMT processors early in their design phase exist, there is a lack of early power and performance evaluation tools for SMT processors. To this end, we have developed PTSMT: a tightly coupled power, performance and thermal exploration tool for SMT processors. In this paper, we demonstrate that PTSMT can automatically and effectively accomplish power, performance and thermal exploration of SMT processors at various levels of design hierarchy, at the application level, microarchitecture level, and physical level. Our experimental results show that: at the application level, number of contexts into which an application is divided could affect performance by 2.2x, energy by 52%, and peak temperature by 35^oC; and at the microarchitecture level, context swapping during run time could reduce energy by 9% and improve performance by 8%. These observations indicate the size of the design space which can be explored using PTSMT.

1. Introduction

Simultaneous multithreading [1] is a processor design that combines hardware multithreading with superscalar processor technology to allow multiple threads to issue instructions each cycle. Unlike conventional superscalar processors, which suffer from a lack of per-thread instruction-level parallelism, simultaneous multithreading uses multiple threads to compensate for low single-thread ILP. The performance consequence is significantly higher instruction throughput and program speedups.

While SMT processors have become viable only due to the level of integration provided by technology scaling, technology scaling has resulted in high power density (power per unit area), and therefore high operating temperatures, both of which have a profound impact on SMT processors. While high power density leads to an increase in the cooling cost, [2], high operating temperatures results in increased probability of timing violations because of higher signal propagation delay and switching time, reduced lifetime because of phenomena such as electromigration [3]. High power densities lead to higher temperatures which in turn increase the leakage power. This close coupling can lead to thermal runaway and therefore requires a joint power, and thermal simulation approach to detect and avoid such situations. These thermal effects in modern sub-nanometer fabrication technologies necessitate a closely coupled power performance and thermal modeling of SMT processors. Furthermore, temperature-aware performance modeling [4, 5] becomes critical because the temperatures of individual blocks in the processor play a very important role in estimating the reliability and performance of the whole processor design. For example, the temperature of any block in a chip is not only dependent on its own power density but also on the power densities of adjacent blocks, due to thermal diffusion (i.e. the flow of heat from hot blocks to cold blocks).

While there exist several tools for performance exploration of SMT processors [6, 7, 8], there is a lack of closely coupled power, performance and thermal simulation tools, which can effectively detect thermal emergencies, and evaluate the impact of various thermal policies.

In this paper, we present **PTSMT**: A tool for closely coupled Power, Performance and Thermal simulation of SMT processors. We present experimental results to demonstrate the effectiveness of PTSMT in early exploration of SMT processor designs at various levels of design hierarchy: at the **Application level**, where decisions such as the number of contexts into which an application should be divided are made; **Microarchitectural level** at which optimal selection of hardware resources, instruction fetch policies from different threads is done; **Physical level**, where different floorplans and other technology parameters are examined. PTSMT empowers the designer with the capability to quantitatively evaluate various design alternatives and power and thermal management techniques at each of the three design levels, very early in the SMT processor design process.

2. Related Work

2.1. PPT Exploration

Temperature estimation of VLSI chips is a well defined area of research. Industrial tools (e.g., Flotherm [9]) and Firebolt [10]

1063-9667/08 $25.00 © 2008 IEEE
DOI 10.1109/VLSI.2008.84

are based on finite element analysis methods and have been in use for a long time. Such tools were primarily used by package and cooling system designers. A thermal model can be converted to an equivalent electrical model dual [5]. Temperature estimation tools [5], [11] based on this model are often used in academic settings. In our work we have used Hotspot [5].

2.2. SMT - Performance Exploration

Many previous works have shown the importance of SMT architectures, analyzing and evaluating their performance characteristics. Early work by Goncales et al. [7] demonstrated the need for an efficient SMT simulator to model the multithreaded architecture including the memory hierarchy. They presented an SMT simulator that has been developed on top of the Simplescalar toolset [12]. Tullsen et al. presented the SMTSIM [6]-an instruction-level simulator for a detailed simulation of a pipelined SMT processor with all sources of latency modeled. Madon et al. [8] developed the SSMT simulator, to evaluate the performance aspects of SMT processors. The SSMT is an SMT simulator based on SimpleScalar and augmented to simulate an SMT pipeline.

2.3. SMT - Power and Thermal Exploration

The power consumption of SMT processors has been examined in [13]. They used PowerTimer [14] to estimate the power of the underlying hardware components of the SMT processor. [15] have carried out investigations on thermal management of SMT processors. They have proposed increasing the area of the blocks with high power densities by adding redundant devices in an effort to reduce the power densities and hence reduce temperatures. A taxonomy of different thermal management schemes for SMT processors is given in [16]. [17] presents an adaptive fetch algorithm where new instructions are fetched depending on history based profiling of threads as integer or floating-point intensive.

In all of the above works, SMT processors were simulated and different power estimators such as [14], [18] were used to estimate power. Then temperatures were estimated using tools like [5] or in many cases power densities were used as a indicative representation of temperatures. Such an approach ignores the effects of **thermal diffusion** among blocks with different temperatures. It has been observed that because of thermal diffusion, within a chip the temperature of a block with lower power density can be much higher than the temperature of another block with higher power density [19]. The floorplan of a processor significantly affects thermal diffusion and hence the temperature profile [20]. Our work is different from all the above in that we **tightly couple performance, power, and temperature estimation** for SMT processors. The reason for tight coupling is that *we should be able to examine the effect of thermal management policies on performance and power during run time*. For example, if a thermal emergency occurs and a thermal management technique kicks in at the N^{th} cycle, we should be able to observe the resulting changes (because of the thermal management technique) in the performance, power, and temperature traces from $(N+1)^{th}$ cycle onwards.

Another motivation behind coupling power and temperature estimation is **leakage power**. Leakage power of a transistor increases with temperature in a super-linear manner [21]. Increased leakage power raises the temperature further. This positive interdependency continues to raise both temperature and leakage power until the total power consumed is dissipated to the environment by the package and steady temperatures are reached. Leakage power and temperature estimations can be off by as much as 20% if this positive feedback between temperature and leakage power are ignored [19]. Existing SMT simulators do not take into account the power and thermal considerations during their performance simulation. In our work we have coupled performance, power, and temperature estimation, and presented a platform for concurrent PPT exploration of SMT processors across different levels in a design.

3. PTSMT: Power, Performance & Temperature Simulator for SMT Processors

Power and temperature are fast becoming the bottleneck in increasing the performance of SMT processors. Consequently, both power and thermal management techniques must be investigated in a tightly coupled manner for these technique to be effective. To this end we have developed a **P**ower-**T**emperature **S**imultaneous **M**ulti **T**hreading simulator called PTSMT.

3.1. Performance Model

At the heart of the PTSMT performance model is the Simplescalar [12] engine. The SSMT simulator used to model the performance of SMT processors supports out-of-order issue and execution. PTSMT models a very detailed out-of-order issue SMT processor with a two-level memory system and speculative execution support. This simulator is able to execute multiple contexts simultaneously i.e. dispatch instructions provided by multiple contexts to functional units.

3.2. Power Model

The power models are based on WATTCH [18], but also include the leakage energy models, similar to those in the PTScalar tool setPTSMT models both the dynamic power, and the leakage power, including the temperature dependence of the leakage power. Dynamic energy is independent of temperature, but has a quadratic dependence on supply voltage. In our power model, dynamic energy for a circuit block in each cycle is calculated as: $P_d = N_{gate} \cdot C \cdot V_{dd}^2$, where P_d represents the dynamic energy of the block per cycle, N_{gate} is the number of gates in the circuit block, C represents the switching capacitances and V_{dd} represents the supply voltage. Leakage power mainly consists of subthreshold and gate leakage power. In our power model, we consider both subthreshold and gate leakage for logic circuits and all memory-based units using Equation (1):

$$P_{leak} = N_{gate} \cdot I_{avg}(T, V_{dd}) \cdot V_{dd} \quad (1)$$

where, P_{leak} is the total leakage power, $I_{avg}(T, V_{dd})$ is the total leakage current per gate at a given temperature T and supply voltage V_{dd}.

3.3. Thermal Model

The thermal model in PTSMT is based on the HotSpot [5] models. Temperature is modeled by equivalent RC thermal circuits, based on the duality between heat transfer and electrical phenomena where two parameters: thermal resistance R_t and thermal capacitance C_t are used to characterize thermal behavior. The equivalent RC thermal circuit consists of three layers: heatsink, heat spreader and chip die. Provided the average power within a time period computed using the power model, the transient temperature is calculated by solving the differential equations for the RC circuit with a fourth-order Runge-Kutta method.

3.4. Capabilities

PTSMT provides the designer with a fast, cycle-accurate simulation of SMT processors for a coupled evaluation of performance, power, and temperature. The SMT architecture is parameterizable and many of these features were inherited from the SimpleScalar and SSMT simulators. Some of the *parameterizable options* include: instruction and data caches configurations, number of integer and floating point ALUs, branch predictor configurations, size of Register Update Unit (RUU), latency for memory accesses, Translation Lookaside Buffer (TLB) configurations, and other execution parameters such as the maximum number of instructions to simulate etc. The output of PTSMT includes many *performance parameters* such as: Total number of instructions committed from each context; number of loads, stores, and branches; simulation statistics such as total number of cycles, instructions per cycle, time; branch predictor statistics on lookups, hits, misses, updates; cache behavior parameters such as accesses, hits, misses, replacements etc. for instruction and data L1 caches, L2 cache, and instruction and data TLBs; access counts, power, and temperature values at each cycle for all the SMT architectural components.

3.5. Construction

PTSMT is derived from the SSMT simulator. We have supplemented SSMT with a power estimator which is similar to WATTCH [18] and temperature estimator which is similar to HotSpot [5]. Figure 3.5 shows the basic construction of PTSMT. PTSMT performs a fast, cycle-accurate, power and temperature coupled simulation of SMT architecture. However, the architectural configuration can be easily changed which allows the tool to be used for architectural exploration.

The tool takes binaries compiled for the SimpleScalar architecture, floorplan of the processor, configuration files with parameters for the thermal and power models, package and other technology dependent libraries. PTSMT also inherited an advanced cache simulator [26] from Simplescalar which models behavior like cache misses etc. Outputs of PTSMT simulation include performance statistics, power (leakage + dynamic), and temperature profiles for all the components in the architecture at any cycle intervals.

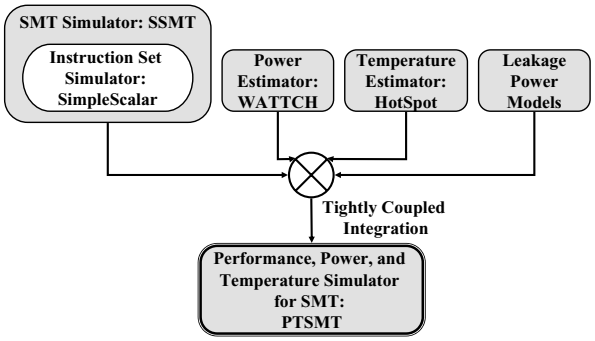

Figure 1. Construction of PTSMT

4. Cross-level PPT Exploration with PTSMT

4.1. Experimental Setup

We have used a SMT version of Alpha-21464 as the floorplan to conduct our experiments and demonstrate the need and usefulness of PTSMT. Alpha-21464 is not a SMT, therefore, we have added the required additional hardware to enhance the architecture for SMT. Then we used Parquet [22] to generate a floorplan for this variant. We used the benchmarks **benchmarks** from the MiBench [23] suite. We use 45nm technology, clock speed of 3 GHz and a relatively large L2 cache, and other parameters scaled accordingly. For our SMT, we allow complete sharing of the Instruction and Data L1 caches, functional units and all other blocks except the Integer and Floating point register files and the corresponding register alias tables and the RUU.

In our results, we have used a **base case** for comparison, which we define as follows: To do a fair comparison we assume that the SMT processor in base case has a thermal control mechanism. The temperatures of all the blocks in the processor are continuously monitored during the execution of the program. If the temperature of any block goes beyond a defined thermal threshold of 100^oC (can be different for other designs or technologies), we stop fetching instructions till the temperature wanes (i.e., stall the complete processor pipeline) in order to avoid a thermal runaway.

4.2. Physical level Exploration

The phenomenon of power and temperature are very closely related to physical design choices such as selection of cell libraries, floorplan etc. The floorplan of the processor affects the temperature profile because it changes the thermal diffusion among neighboring components. As a result, different floorplans have different temperature profiles. Since the leakage power of a transistor has a super-linear dependency on temperature, the floorplan also has an affect on the leakage power dissipation.

Effect of Floorplan

We simulated our benchmarks using five different SMT floorplans with PTSMT. Figure 2 shows the energy consumption (normalized to the base case) and peak temperatures of the processor for different floorplans. The energy consumption values are normalized to the energy for the first floorplan. It was ob-

served that there was a 13% variation in the overall energy consumption of the processor and a 21^oC variation in the maximum temperature of the chip across different floorplans. This variation in the peak temperature has a foremost effect on the cost of the package and the cooling mechanism which increases significantly with temperature.

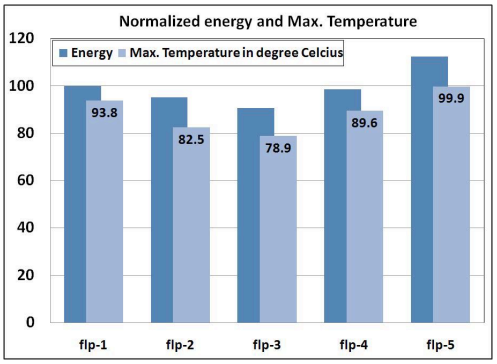

Figure 2. Energy and maximum temperatures for different floorplans of an SMT Processor

If the cooling mechanism includes a fan, a lower peak temperature will warrant less use of the fan and thus save power consumed by fans (which can range from 4W for hand-held devices to 30W for fixed devices). Other benefits of a lower peak temperature include: improved reliability, reduced leakage current, less delay for transistors and wires, etc. Different floorplans have different thermal profiles because the placement of the blocks affects thermal diffusion. This experiment motivates the need for a coupled power and temperature estimation tool which takes into account the effect of floorplans on temperature and hence on the power consumption. If we consider only those floorplans which meet the timing constraints of the processor, there is a design space with multiple floorplans having different power dissipation and temperature profiles.

4.3. Microarchitecture level Exploration

SMT processors need replication of certain hardware so that multiple threads can issue instructions in the same cycle. There are multiple ALUs of integer and floating point type so that multiple instructions can execute in one cycle. The extra hardware offers increased performance but at a cost of increased area and power. Thus, the choice of microarchitecture and the analysis of the accompanying tradeoffs must be done very early in the design cycle of SMT processors, because they have a direct impact on the PPT metrics.

Effect of Context Swapping

If the temperature of any of the resources corresponding to one of the contexts rises above a defined threshold temperature, we swap the execution of this context with another context. The context responsible for the increased register file temperature now uses another register file which was at a lower temperature, thereby allowing the original register file to cool down. This runtime swapping is repeated again when needed. We simulate the benchmarks and compare the runtime, energy, and temperature of the processor against the case when no context swapping was employed. From Figure 3 we observe that on an average there is a 9% reduction in energy consumption with a 8% reduction in runtime (performance improvement) because of this swapping of contexts compared to the base case of stalling the processor.

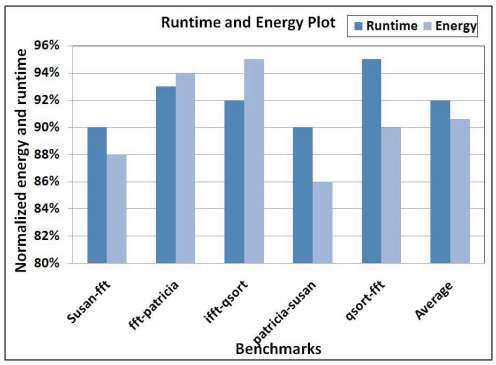

Figure 3. Energy and performance for context swapping to do thermal management

Effect of Architectural Choices

The number of functional units, Integer ALUs (IALUs) and Floating Point ALUs (FALUs), is one of the microarchitectural parameters that can be used to optimize power and performance. We simulate our benchmarks using PTSMT with varying number of IALUs and analyze the energy and runtime in each case. Figure 4 plots the processor's total energy consumption and runtime for the representative susan corners benchmark as the number of IALUs in the microarchitecture vary from 1 to 8.

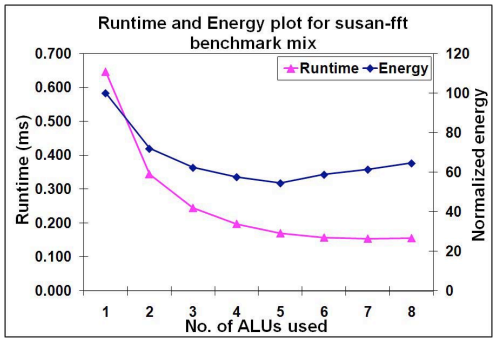

Figure 4. Energy and performance for a benchmark at different number of IALUs in the SMT architecture

It can be seen from the curve that the runtime decreases steeply as the number of IALUs increase from 1 to 4. This is because of higher the number of IALUs. But as the number of IALUs increases to 8, runtime decreases slightly before stabilizing because there are not too many integer operations in the benchmark. Functional units such as IALUs, being regions of very high activity, are one of the major contributors to the processors total energy. Consequently, we observe a significant difference in the total energy with the different number of IALUs used. The energy consumption curve shows a decrease in energy as the number of IALUs increases from 1 to 5 because: the instructions are distributed among all the IALUs, thereby decreasing

the dynamic energy consumed by each of them. The energy consumption increases as the number of IALUs increases from 5 to 8 because: though there are more number of IALUs, there are lower number of instructions committed to them, which is also implicit from a corresponding halt in the decrease of runtime. Hence, though the additional IALUs consume only small dynamic power, they consume a significant standby power. Thus different number of IALUs provide a tradeoff between energy consumption and runtime.

Effect of Fetch Policies

PTSMT allows the exploration of different policies for fetching instructions from the ready queues of hardware contexts, for SMT processors. The 'fetch policy' governs the way in which multiple instructions are fetched from contexts which are ready with executable instructions. The fetch policy has a direct impact on the PPT metrics trio. We implement two such fetch policies and compare the power and performance of each of them with that of the base case.

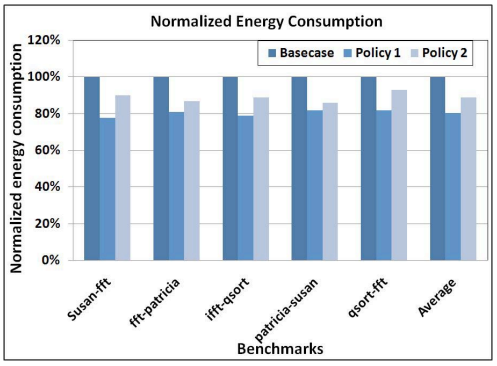

Figure 5. Energy for different fetch policies to regulate maximum temperature

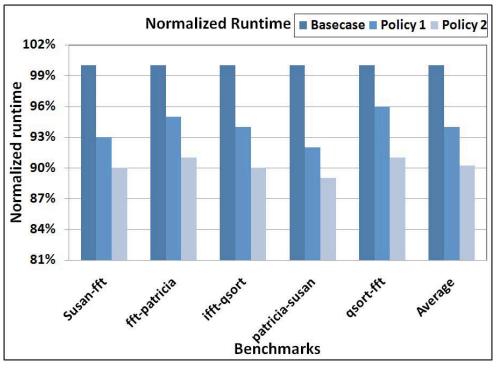

Figure 6. Performance for different fetch policies to regulate maximum temperature

Policy 1: The temperature of any of the resource corresponding to each context (such as the Register Files and the Register Update Unit) corresponding to all contexts is monitored. If the temperature of any of them goes beyond the defined threshold, instructions corresponding to that context are stopped from being fetched, until the temperature of the hottest block belonging to that context drops below the lower threshold temperature.

Policy 2: In this policy, similar to policy 1, the temperature of any of the resource corresponding to each context is monitored. If the temperature of any of them goes beyond the defined threshold the number of instructions being fetched from the hottest context in each cycle is reduced.

Figure 5 and Figure 6 show the normalized (to base case) energy consumption and run time for the two policies respectively. On an average there is up to 20% and 11% energy reduction for fetch policies 1 and 2 respectively compared to the base case. Thus there is an average 6% and 10% performance gain respectively for the two fetch policies compared to the base case. Using PTSMT, the designer can choose to implement several other power and thermal management techniques for SMT processors at the micro architecture level, for a combined power, performance and temperature exploration.

4.4. Application level Exploration

Since SMT processors offer the capability to execute instructions from multiple threads in a single cycle, the top level application is divided into multiple threads or *contexts*. Each context has its own set of registers and program counters to maintain its state. While it is desirable to partition an application into the maximum number of threads, it is not always beneficial to do so. The maximum number of threads that an application can be broken into is limited by the hardware resources available on the SMT processor. The number of threads running on the SMT processor will not only affect the performance of the application but they also alter the power consumption and thermal profiles.

Selection of Number of SW Contexts for an Application

The division of an application into a suitable number of contexts may be done explicitly at functional level by the designer or it may be done using special compiler solutions.

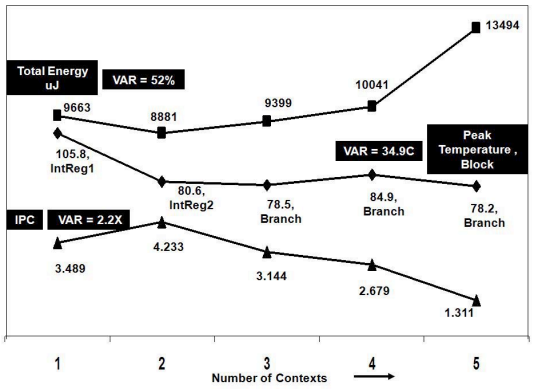

Figure 7. Variation in Performance, Energy & Temperature metrics when an application is run as different number of contexts on SMT

We have taken a large application based on MiBench benchmarks. We divided this application into a different number of sub-applications ranging from 2 to 5 for which the results are in Figure 7. We observe that maximum IPC is obtained when the application is divided into 2 contexts because the architecture has two hardware replications for context registers, program counters etc. When the application is divided into more con-

texts the performance goes down because there is a penalty of swapping contexts. We observe large variations (VAR) in energy, peak temperature, and performance. Thus, there is an exploration space which needs to be traversed in order to identify the right number of threads for an application in order to meet PPT budgets.

4.5. Overhead of PTSMT

In order to allow designers to take full advantage of PTSMT to do an early evaluation of design choices at the three levels it is necessary to have a low simulation time for PTSMT. On an average, PTSMT takes 71 seconds to simulate 5 million instructions. On the other hand, the SSMT simulator takes 29 seconds to simulate the same 5 million instructions. However, SSMT simulator only does performance simulation and not a tightly coupled power and temperature estimation as done by PTSMT. This execution overhead of PTSMT is justified by the additional power and temperature estimation for SMT processors at each cycle. The overhead can be reduced greatly if the temperature estimation is done after every tens of thousands of cycles instead of each cycle. This extends the utility of PTSMT beyond research on performance enhancement techniques to optimization of the PPT metrics.

5. Conclusion

There has been extensive research on analyzing the performance characteristics of SMT processors and evaluating their tradeoffs. However, current generation of technologies have imposed additional constraints of power dissipation and temperature on designs. There is a need for simultaneous evaluation of Performance, Power, and Temperature (PPT) metrics for SMT processors because of inter-relation between these metrics. In this paper we attempted to bridge this gap using a platform which does a tightly coupled performance, power, and temperature estimation for SMT processors and also allows the designer to evaluate choices at three different levels of design: Application, Microarchitecture, and Physical. We demonstrated that such a cross-level exploration will give the designer an opportunity to explore different choices at all the above levels to do early trade-off analysis for PPT.

Acknowledgment

This work was partially supported by NSF grant: CCF-0702797. The authors would like to thank Kaiyu Chen from Princeton University and Prof. Sarma Vrudhula and Vipin Mohan from Arizona State University.

References

[1] D. Tullsen et al., " Simultaneous Multithreading: Maximizing On-Chip Parallelism," *Proceedings of International Symposium on Computer Architecture*, 1995.

[2] S. Velusamy et al., "Monitoring Temperature in FPGA based SoCs," *Proceedings of International Conference on Computer Design*, 2005.

[3] K. Banerjee, A. Sangiovanni-Vincentelli et al., "On Thermal Effects in Deep Sub-Micron VLSI Interconnects," *Proceedings of Design Automation Conference*, 1999.

[4] W. Liao et al.,"Microarchitecture level power and thermal simulation considering temperature dependent leakage model,"*Proceedings of International Symposium of Low Power Electronic Devices*, 2003.

[5] W. Huang et al.,"HotSpot: Thermal Modeling for CMOS VLSI Systems,"*IEEE Transactions on Component Packaging and Manufacturing Technology*, 2005.

[6] D. Tullsen et al., "The SMTSIM Multithreading simulator", 1996.

[7] Ronaldo Goncalves et al., "SS SMT - A Simulator for SMT Architectures", 1998

[8] Dominik Madon et al., "The SSMT Simulator", 1999

[9] Flotherm, http://www.flomerics.com/flotherm/

[10] Firebolt,http://www.gradient-da.com/tech/firebolt.htm

[11] Y. Yang et al.,"Adaptive Chip-Package Thermal Analysis for Synthesis and Design,"*Proceedings of Design, Automation, and Test in Europe*, 2006.

[12] T. Austin et al., "The SimpleScalar Tool Set, Version 2.0," *University of Wisconsin-Madison Computer Sciences Department Technical Report*, June 1997.

[13] Y. Li et al., "Understanding the Energy Efficiency of Simultaneous Multithreading," *Proceedings of International Symposium on Low Power Electronics and Design*, 2004.

[14] D. Brooks et al., "Microarchitectural-Level Power-Performance Analysis: The PowerTimer Approach," *IBM Journal of Research and Development*, Oct/Nov, 2003.

[15] J. Donald et al.,"Temperature-Aware Design Issues for SMT and CMP Architectures,"*Proceedings of the Wkshop on Complexity-Effective Design*, 2004.

[16] J. Donald et al.,"Techniques for Multicore Thermal Management: Classification and New Exploration,"*Proceedings of International Symposium on Computer Architecture*, 2006.

[17] J. Donald et al.,"Leveraging Simultaneous Multithreading for Adaptive Thermal Control,"*Workshop on Temperature-Aware Computing Systems*, 2005.

[18] D. Brooks et al.,"Wattch: A Framework for Architectural-Level Power Analysis and Optimizations,"*Proceedings of International Symposium on Computer Architecture* , 2000.

[19] Aseem Gupta et al.,"STEFAL: A System Level Temperature- and Floorplan-Aware Leakage Power Estimator for SoCs," *Proceedings of VLSI Design Conference*, 2007.

[20] I. Koren et al.,"Temperature Aware Floorplanning," *Proceedings of Workshop on Temperature Aware Computer Systems,*, 2005.

[21] A. Agarwal, K. Roy et al., "Leakage in nano-scale technologies: mechanisms, impact and design considerations," *Proceedings of Design Automation Conference*, 2004.

[22] S. Adya et al., "Fixed-outline Floorplanning: Enabling Hierarchical DesignEfficient," *IEEE Trans. on VLSI Systems*, 2003.

[23] M. Guthaus et al., "MiBench: A free, commercially representative embedded benchmark suite," *IEEE Workshop in workload characterization*, 2001.

[24] D. Tullsen et al., " Simultaneous Multithreading: A Platform for Next Generation Processors," *Proceedings of International Symposium on Computer Architecture*, 1995.

[25] T. N. Vijaykumar et al.,"Heat Stroke: Power-Density-Based Denial of Service in SMT,"*Proceedings of International Symposium on High-Performance Computer Architecture* , 2005.

[26] R. Sugumar et al., "Efficient Simulation of Caches under Optimal Replacement with Applications to Miss Characterization," *Proceedings of ACM Sigmetrics Conference on Measurements and Modeling of Computer Systems*, 1993.

[27] Joachim Clabes et al., "Design and implementation of the POWER5 microprocessor", *Proceedings of the 41st annual conference on Design automation*, 2004.

[28] Shailender Chaudhry et al., "High-Performance Throughput Computing", *IEEE Micro*, 2005.

[29] Intel Hyper-threading technology, http://www.intel.com/technology/platformtechnology/hyper-threading/index.htm

[30] M. Onouchi et al., "A System-level Power-estimation Methodology based on IP-level Modeling, Power-level Adjustment, and Power Accumulation," *Proceedings of Asia South Pacific Design Automation*, 2006.

[31] J.Zejda et al., "Gate-Level Power Estimation Using Transition Analysis," *Workshop on Design Methodologies for Microelectronics*, 1995.