Evaluating the Thermal Efficiency of SMT and CMP Architectures

Yingmin Li[†], Zhigang Hu[††], David Brooks[‡], Kevin Skadron[†]
[†] Dept. of Computer Science, University of Virginia [††] IBM T.J. Watson Research Center
[‡] Dept. of Computer Science, Harvard University
{yingmin,skadron}@cs.virginia.edu, zhigangh@us.ibm.com, dbrooks@eecs.harvard.edu

Abstract

Simultaneous multithreading (SMT) and chip multiprocessing (CMP) both allow a chip to achieve greater throughput, but their thermal properties are still poorly understood. This paper uses Turandot, PowerTimer, and HotSpot to evaluate the thermal efficiency for a Power4/Power5-like core. Our results show that although SMT and CMP exhibit similar peak operating temperatures, the mechanism by which they heat up are quite different. More specifically, SMT heating is primarily caused by localized heating in certain key structures such as the register file, due to increased utilization. On the other hand, CMP heating is mainly caused by the global impact of increased energy output, due to the extra energy of an added core. Because of this difference in heat up machanism, we found that the best thermal management technique is also different for SMT and CMP. Finally, we show that CMP and SMT will scale differently as the contribution of leakage power grows, with CMP suffering from higher leakage due to the second core's higher temperature and the exponential temperature-dependence of subthreshold leakage.

1 Introduction

Simultaneous multithreading (SMT) [22] is a relatively new microarchitectural paradigm that has found industrial application [12, 15]. SMT allows instructions from multiple threads to be simultaneously fetched and executed in the same pipeline, thus amortizing the cost of many microarchitectural structures across more instructions per cycle. The promise of SMT is area-efficient throughput enhancement. But even though SMT has been shown to be energy efficient for most workloads [14, 19], the significant boost in instructions per cycle or IPC means increased power dissipation and possibly increased power density. Since the area increase reported for SMT execution is relatively small (10-20%), thermal behavior and cooling costs are major concerns.

Chip multiprocessing (CMP) [7] is another relatively new microarchitectural paradigm that has found industrial application [12, 13]. CMP instantiates multiple processor "cores" on a single die. Typically the cores each have private branch predictors and first-level caches and share a second-level, on-chip cache. For multi-threaded or multi-programmed workloads CMP architectures amortizes the cost of a die across two or more processors and allow data sharing within a common L2 cache. Like SMT, the promise of CMP is a boost in throughput. The replication of cores means that the area and power overhead to support extra threads is much greater with CMP than SMT, but the lack of contention between threads yields a much greater throughput for CMP than SMT [4, 7, 17]. Each core on a chip dramatically increases its power dissipation, so thermal behavior and cooling costs are also major concerns for CMP.

Because both paradigms target increased throughput for multi-threaded and multi-programmed workloads, it is natural to compare them. This paper provides a thorough comparison and analysis of thermal behavior and thermal efficiency of SMT and CMP in the context of a POWER4-like microarchitecture. We combine IBM's cycle-accurate Turandot [16] and PowerTimer [3, 9] performance and power modeling tools, modified to support both SMT and CMP, with University of Virginia's HotSpot thermal model [21]. Validation strategies for these tools have been discussed in [10, 14].

We find the thermal profiles are quite different between CMP and SMT architectures although their peak temperature is similar. With the CMP architecture, the heating is primarily due to the global impact of higher energy-output. For the SMT architecture, the heating is very localized, in part because of the higher utilization of certain key structures such as the register file. When we consider dynamic thermal management (DTM) strategies that seek to use feedback control to reduce the key hotspots, these different heating patterns are critical. In general, we find DTM strategies that target local structures are superior for SMT architectures and global DTM strategies work better with CMP architectures.

The rest of the paper is outlined as follows. In Section 2, we discuss the related work in comparing SMT and CMP processors from an thermal efficiency standpoint. Section 3 discusses the details of the performance, power, and temperature methodology that we utilize in this work. Section 4 discusses the baseline results for SMT and CMP archi-

tectures without DTM. Section 5 explores the more realistic case when microprocessors are DTM constrained and explores which strategies are best for CMP and SMT under performance and energy-constrained designs. Section 6 concludes the paper and discusses avenues for future research.

2 Related Work

We are only aware of two papers exploring thermal behavior of SMT and/or CMP. Heo et al. [8] look at a variety of ways to use redundant resources, including multiple cores, for migrating computation of a single thread to control hot spots, but find the overhead of core swapping is high. Donald and Martonosi [6] compare SMT and CMP and find that SMT produces more thermal stress than CMP. But their analysis assumes that the cores of the CMP system are simpler and have lower bandwidth than the single-threaded and SMT processors, while we assume that all three organizations offer the same issue bandwidth per core. They also consider a novel mechanism to cope with hotspots, by adding "white space" into these structures in a checkerboard fashion to increase their size and hopefully spread out the heat, but found that even a very fine-grained partitioning did not achieve the desired heat spreading. We adopt a similar idea for the register file, our key hotspot, but rather than increase its size, we throttle its occupancy. Simulations using an improved version of HotSpot in [11] suggest that sufficiently small structures will spread heat effectively.

3 Modeling Methodology

3.1 Microarchitecture & Performance modeling

We use Turandot/PowerTimer to model an out-of-order, superscalar processor with resource configuration similar to current generation microprocessors. The overall processor organization is shown in Figure 1. Table 3.1 describes the configuration of our baseline processor for the single-threaded design point.

SMT is modeled by duplicating data structures that correspond to duplicated resources and increasing the sizes of those shared critical resources like the register file. Round-robin policy is used at various pipeline stages to decide which threads should go ahead. More detail about SMT enhancement can be found in [14].

We extended Turandot to model a CMP configuration. So far, only multi-programmed workloads without inter-thread synchronization are supported. This essentially consists of simulating two separate cores, except that cache and cache-bus conflicts in the shared L2 cache must be modeled, as they are important determinants of performance.

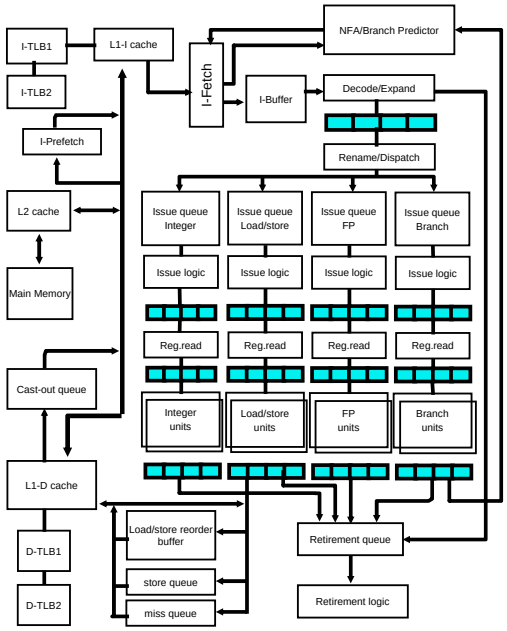


Figure 1: **Modeled processor organization for a single core.**

3.2 Benchmark Pairs

We use 15 SPEC2000 benchmarks as our single thread benchmarks. They are compiled by the *xlc* compiler with the -O3 option. First we used the Simpoint toolset to get representative simulation points for 500-million-instruction simulation windows for each benchmark, then the trace generation tool generates the final static traces by skipping the number of instructions indicated by Simpoint and then simulating and capturing the following 500 mil-

Processor Core	
Dispatch Rate	5 instructions per cycle
Reservation stations	mem/fix queue (2x20), fpq (2x5)
Functional Units	2 FXU, 2 FPU, 2 LSU, 1 BRU
Physical registers	80 GPR, 72 FPR
Branch predictor	16K-entry bimodal, 16K-entry gshare, 16K-entry selector, all with 1-bit entries
Memory Hierarchy	
L1 Dcache Size	32KB, 2-way, 128B blocks
L1 Icache Size	64KB, 2-way, 128B blocks
L2 I/D	1MB, 4-way LRU, 128B blocks 9-cycle latency
Memory Latency	77 cycles

Table 1: **Configuration of simulated processor.**

lion instructions.

We use pairs of single-thread benchmarks to form dual-thread SMT and CMP benchmarks. There are many possibilities for forming the pairs from these 15 benchmarks. We follow the following methodology to form our pairs. First, we let each single thread benchmark combine with itself to form a pair. We also form several SMT benchmarks by combining different single thread benchmarks. We first categorize the single thread benchmarks into eight major categories: high IPC (> 0.9) or low IPC (< 0.9), high temperature (peak temperature $> 355K$) or low temperature (peak temperature $< 355k$), floating benchmark or integer benchmark. We then form eighteen pairs of dual-thread benchmarks by selecting unique combinations of benchmarks with these categorizing criteria. In the following paper, we will divide the formed pairs into two categories, pairs that have high L2 cache miss ratio, and pairs that have low L2 cache miss ratio. When 1 benchmark in a pair has a high L2 cache miss ratio, we will categorize that pair as a high L2 cache miss pair.

3.3 SMT and CMP Speedup Metric

Comparison of different SMT or CMP configurations, or comparison of an SMT or CMP configuration against a single-threaded configuration, is difficult. As Sazeides and Juan [18] have shown, IPC can be misleading unless exactly the same instruction count for each thread is used in all experiments. Both groups propose similar metrics for computing an "SMT speedup". The goal is to distinguish between configurations that achieve high throughput at the expense of a single thread from those that do so with balanced throughput from both threads.

Snavely et al. propose that

$$\text{SMT speedup} = \sum \frac{IPC_{SMT}[i]}{IPC_{nonSMT}[i]} \qquad (1)$$

where $IPC_{SMT}[i]$ is the IPC of just the i'th thread during an SMT execution and $IPC_{nonSMT}[i]$ is its IPC during single-threaded execution. This considers how each thread performs under SMT relative to its non-SMT performance, so we choose this metric for our speedup computations. All speedups are computed relative to the IPC of each workload on the baseline, non-SMT machine.

In contrast to evaluating performance, evaluating energy efficiency should use traditional, simple unweighted metrics.

3.4 Power Model

PowerTimer differs from existing academic microarchitectural power-performance simulators primarily in energy-model formation. The base energy-models are derived from circuit-level power analysis that has been performed on structures in a current, high-performance PowerPC processor. This analysis has been performed at the macro level, and in general, multiple macros will combine to form a microarchitectural level structures corresponding to units within our performance model. PowerTimer models over 60 microarchitectural structures which are defined by over 400 macro-level power equations.

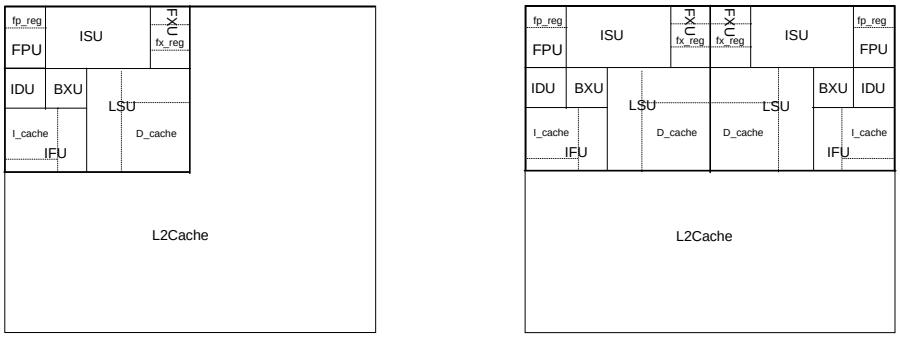
3.5 Temperature Model

To model operating temperature, we use the newly released HotSpot 2.0 (http://lava.cs.virginia.edu/HotSpot), which accounts for the important effects of the thermal interface material (TIM) between the die and heat spreader and has been validated against a test chip [10].

HotSpot models temperature using a circuit of thermal resistances and capacitances that are derived from the layout of microarchitecture units. The thermal package that is modeled consists of the die-to-spreader TIM (thickness 0.05mm), the heat spreader (thickness 1mm), another TIM, the heat sink (thickness 6.9mm), and a fan. Removal of heat from the package via airflow takes place by convection and is modeled using a single, equivalent thermal resistance of 0.8K/W . This assumes the fan speed and the ambient temperature inside the computer "box" ($40°$C) are constant, both of which are true for the time scales over which our benchmarks are simulated.

Due to lateral heat spreading, thermal behavior is sensitive to the layout of the microarchitecture units. We use the floorplans shown in Figure 2, which have been derived by inspection from the die photo of the Power5 in [5]. Note that Figure 2 only shows floorplans for the single-threaded and CMP chips. The SMT floorplan is identical to the single-threaded case, except that the increase in resources to accommodate SMT means that the core is 12% larger.

According to [5], the POWER5 offers 24 sensors on chip. Accordingly, we assume it is reasonable to provide at least one temperature sensor for each microarchitecture block in the floorplan, and that these sensors can be placed reasonably close to each block's hot spot, or that data fusion among multiple sensors can achieve the same effect. We also assume that averaging and data fusion allow dynamic noise to be ignored , and that offset errors can be removed by calibration [1]. We sample the temperature every 100k cycles and set our DTM experiments' thermal emergency threshold at 356K. This threshold is carefully chosen so for single thread single core architecture it will normally lead to less than 5% performance loss due to DTM control. At the beginning of the simulation, we set the steady state temperature for each unit as the initial temperature so the whole simulation's thermal output will be meaningful.



(a) ST & SMT (b) CMP

Figure 2: **Floorplans used for thermal simulation. Note that the SMT core is actually 12% larger than the ST core shown above**

4 Baseline Results

In this section, we discuss the temperature implications of SMT and CMP designs *without* dynamic thermal management. In the next section, we consider thermally limited designs.

When we compare the three architectures (ST, SMT, and CMP), we hold the chip area as a constant at 210 mm^2 including the on-chip level two cache. This means CMP will have the smallest L2 cache, since its core area is largest among the three. In our experiment, the L2 cache sizes for ST, SMT, and CMP are 2MB, 2MB, and 1MB respectively.

4.1 SMT and CMP Temperature

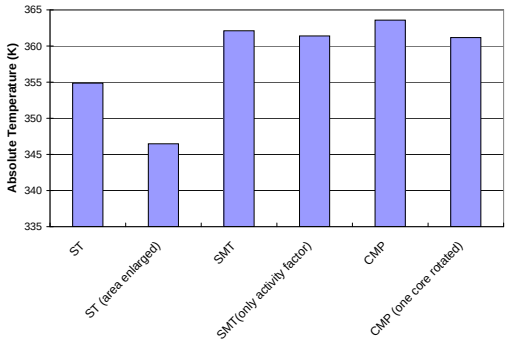


Figure 3: **Temperature of SMT and CMP vs. ST.**

Figure 3 compares the maximum measured temperature[1] for several different microprocessor configurations. We see that the single-threaded core has a maximum temperature of nearly 355K. When we consider the SMT processor, the temperature increases around 7 degrees and for the CMP processor the increase is around 8.5 degrees.

With such a small difference in temperature, it is difficult to conclude that either SMT or CMP is superior from a temperature standpoint. In fact, if we rotate one of the CMP cores by 180 degrees, so the relatively cool IFU of core 1 is adjacent to the hot FXU of core 0, the maximum CMP processor temperature will drop by around 2 degrees, which makes it slightly cooler than the SMT processor.

Despite the fact that the SMT and CMP processors have relatively similar absolute temperature ratings, the reason for the SMT and CMP hotspots are quite different. In order to better understand underlying reasons behind the temperature increases in these machines, we have performed additional experiments to isolate the important effects.

We have taken the SMT core and only scaled the power dissipation with increased utilization (omitting the increased power dissipation due to increased resources and leaving the area constant). From Figure 3 we can see that the SMT temperature will rise to nearly the same level as when all three factors are included. This makes sense when we consider that the *unconstrained power density* of most of the scaled structures in the SMT processor (e.g. register files and queues) will likely be relatively constant because the power and area will both increase with the SMT processor, and in this case the utilization increase becomes the key for SMT hotspots. From this we can conclude that for the SMT processor, the temperature hotspots are largely due to the higher utilization factor of certain structures like the integer register file.

The reasoning behind the increase in temperature for the CMP machine is quite different. For the CMP machine, the utilization of each individual core is nearly the same as for the single-thread architecture. However, on the same die area we have now integrated two cores and the total power of the chip nearly doubles (as we saw in Figure 5) and hence the total amount of heat being generated nearly doubles. Because of the large chip-level energy consumption, the CMP processor heats up the TIM, heat spreader, and heat sink, thus raising the temperature of the overall chip. Thus the increased temperature of the CMP processor is due to a global heating effect, quite the opposite of the SMT processor's localized utilization increase. This

[1] Temperatures are reported in degrees Kelvin, subtract 273.15 for degrees Celsius

fundamental difference in thermal heating will lead to substantial differences in thermal trends as we consider future technologies and advanced dynamic thermal management techniques.

4.2 Impact of Technology Trends

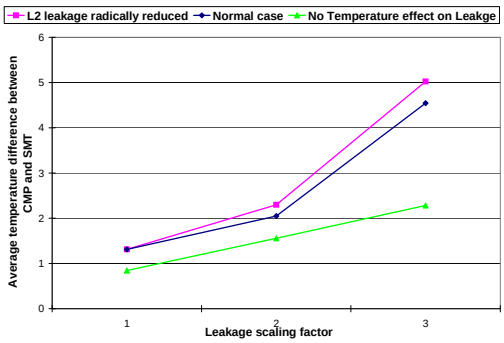


Figure 4: **Temperature Difference between CMP and SMT with different leakage scaling factors.**

As we move towards the 65nm and 45nm technology nodes, there is universal agreement that leakage power dissipation will become a substantial fraction of the overall chip power. Because of the basic difference in the reason for increased thermal heating between the SMT and CMP processors, we expect that these processors will scale differently as leakage power becomes a more substantial portion of total chip power.

Figure 4 shows the impact of technology scaling on the temperature of SMT and CMP processors. In this figure, we show the difference in absolute temperature between the CMP and SMT core for three generations of leakage (roughly corresponding to 130nm, 90nm, and 65nm technologies). As the we increase the leakage scaling factor, there are several important trends to note. The most important trend is that the temperature difference between the CMP machine (hotter) and SMT machine (cooler) increases from 1.5 degrees with our baseline leakage model to nearly 5 degrees with the most leaky technology. The first reason for this trend is that the increased utilization of the SMT cores becomes muted by higher leakage. The second reason is that the SMT machine's larger L2 cache tends to be much cooler than the second CMP core. This, coupled with the exponential temperature dependence of subthreshold leakage on temperature, causes the CMP processor's power to increase more than the SMT processor. This aggravates the CMP processor's global heat up effect. From Figure 4, we can see that if we remove the temperature dependence of leakage in our model, the temperature difference between the CMP and SMT machine grows much less quickly. Figure 4 also shows how the trend is amplified when we consider the case where aggressive leakage control is applied to the L2 cache (perhaps through high-Vt transistors). In this case, the SMT processor is favored because a larger piece of the chip is eligible for this optimization.

5 Aggressive DTM constrained designs

To reduce packaging cost, current processors are usually designed to sustain the thermal requirement of typical workloads, and engage some dynamic thermal management techniques when temperature exceeds the design set point. Because SMT and CMP dissipate more power and run hotter, a more accurate comparison of their relative benefits requires data on their cooling costs, whether those costs are monetary in terms of more expensive packaging, or performance losses from DTM. This section explores the impact of different DTM strategies upon the performance and energy efficiency of SMT and CMP, and how these DTM results explain the different thermal behavior of these two organizations.

It is important to note that peak temperature is not indicative of cooling costs. A benchmark with short periods of very high temperature, separated by long periods of cooler operation, may incur very low performance overhead from DTM, while a benchmark with more moderate but sustained thermal stress may engage DTM often continuously.

To make an equal comparison of DTM performance among ST, SMT, and CMP chips, we continue to use the same thermal package for all three configurations (see Section 3).

5.1 DTM Techniques

We implemented four DTM strategies in this paper:

- Dynamic voltage scaling (DVS): DVS cuts voltage and frequency in response to thermal violations and restores the high voltage and frequency when the temperature drops below the trigger threshold. The low voltage is always the same, regardless of the severity of thermal stress; this was shown in [20] to be just as effective as using multiple V/F pairs and a controller. For these workloads, we found that a voltage of 0.87 (79% of nominal) and frequency of 1.03GHz (77% of nominal) were always sufficient to eliminate thermal violations. Because there is not yet a consensus on the overhead associated with switching voltage and frequency, we test both 10 and 20 μsec. stall times for each change in the DVS setting.

- Fetch-throttling: Fetch-throttling limits how often the fetch stage is allowed to proceed, which reduces ac-

tivity factors throughout the pipeline. The duty cycle is set by a feedback controller.

- Rename-throttling: Rename throttling limits the number of instructions renamed each cycle. Depending on which register file is hotter with the outcome of the previous sampling period, either floating-point register file renaming or integer register file renaming will be throttled. This reduces the rate at which a thread can allocate new registers in whichever register file has overheated, and is thus more localized in effect than fetch throttling. But if the throttling is severe enough, this will has the side effect of slowing down the thread that is causing the hot spot. This can degenerate to fetch throttling, but when it is the FP register file being throttled, the slowdown can be valuable for mixed FP-integer workloads by helping to regulate resource use between the two threads.

- Register-file occupancy-throttling: We find the register file is usually the hottest spot of the whole chip, and its power is proportional to the occupancy. One way to reduce the power of the register file is to limit the number of register entries to a fraction of the full size. To distribute the power density, we propose to interleave the on and off registers, so that the heat can be more evenly spread across the whole register file. It is important to note that our modeling of this technique is idealistic, assuming that the reduction in power density across the register file is proportional to the number of registers that have been turned off. This assumes an ideal interleaving and ideal heat spreading and neglects power dissipation in the wiring, which will not be affected with occupancy throttling. This technique is included to demonstrate the potential value of directly reducing power density in the structure that is overheating, rather than reducing activity in the whole chip.

By limiting the resources available to the processor, all these policies will cause the processor to slow down, thus consuming less power and finally cooling down to below the thermal trigger level. DVS has the added advantage that reducing voltage further reduces power density; since $P \propto V^2 f$, DVS provides a cubic reduction in heat dissipation relative to performance loss, while the other techniques are linear. But the other techniques may be able to hide some of their performance loss with instruction-level parallelism. Of the three policies, fetch-throttling has more of a global effect over the whole chip by throttling the front end. Register-file occupancy throttling targets the specific hot units (the integer register file or the floating point register file) most directly and thus is the most localized in effect. This may incur less performance loss but also may realize less cooling. Rename throttling is typically more localized then fetch throttling and less so than register-file throttling.

DVS's cubic advantage is appealing, but as operating voltages continue to scale down, it becomes more difficult to implement a low voltage that adequately cuts temperature while providing correct behavior and reasonable frequency. Another concern with DVS is the need to validate products for two voltages rather than one. Finally, our assumption that both frequency and voltage can change in 10–20 μs may be optimistic. If voltage and frequency must change gradually to avoid circuit noise, the latency to achieve adequate temperature reduction may be prohibitively long.

Our register-occupancy throttling is limited to register files based on a latch-and-mux design. Power dissipation in SRAM-based designs is likely to be much more heavily dominated by the decoders, sense amplifiers, and word and bit lines. Furthermore, our technique may be idealistic, because it assumes that reducing register file occupancy uniformly reduces power density, when in fact those registers that remain active will retain the same power dissipation. But this does not mean that the temperature of active registers remains unchanged, because neighboring areas of lower power density can help active registers to spread their heat. Whether a register is small enough to spread enough heat laterally is an open question and requires further analysis. However, results in [11] using HotSpot 2.0 suggest that, below about 0.2–0.25 mm and for a 0.5mm die with a typical high-performance package, the ratio of vertical to lateral thermal resistance is so high that heat spreads out very quickly, without raising the localized temperature. This result differs from the findings of [6], who used HotSpot 1.0 to find that much smaller sizes are needed to spread heat. But HotSpot 1.0 omits the TIM's very high thermal resistance and performs less detailed thermal modeling of heat flow in the package. Clearly the granularity at which spreading dominates, and alternative layouts and organizations can reduce hotspots, is an important area requiring further research. But almost all prior DTM research has focused on global techniques like fetch gating, voltage-based techniques, or completely idling the hot unit, all of which suffer from significant overheads. What is needed are techniques that can reduce power density *in situ*, without introducing stalls that propagate all the way up the pipeline. Our register-occupancy throttling illustrates that such an approach offers major potential benefits, and further research in this direction is required.

5.2 Baseline SMT and CMP performance and Energy results

Because we find the performance and energy conclusions are quite different for workloads with high L2 miss rate vs. those with lower miss rates, we normally report results for

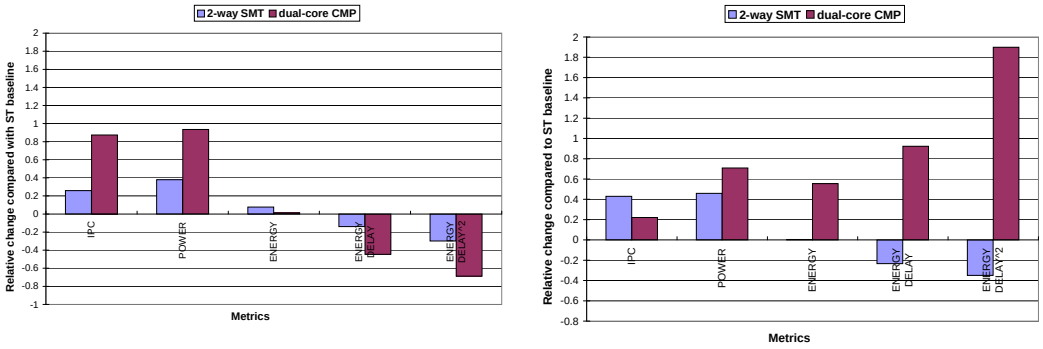


Figure 5: **Performance and Energy efficiency of SMT and CMP compared to ST, for low L2 cache miss workloads (Left) and high L2 cache miss workloads (Right)**

these categories separately.

Figure 5 breaks down the performance benefits and energy efficiency of SMT and CMP for our POWER4-like microarchitecture. The results in this figure are divided into two classes of benchmarks – those with relatively low L2 miss rates (left) and those with high L2 cache miss rates (right). This figure shows that CMP dramatically outperforms SMT for workloads with low to modest L2 miss rates, with CMP boosting throughput by 87% compared to only 26% for SMT. But the CMP chip has only half the L2 cache as SMT, and for workloads with high L2 miss rate, CMP only affords a throughput benefit of 22% while SMT achieves a 42% improvement.

The power and energy overheads demonstrated in Figure 5 are also enlightening. The power overhead of SMT is 38–46%. The main reasons for the SMT power growth are the increased resources that SMT requires (e.g. replicated architected registers), the increased resources that are needed to reduce new bottlenecks (e.g. additional physical registers), and the increased utilization due to additional simultaneous instruction throughput [14]. The power increase due to CMP is even more substantial: 93% for low-L2-miss-rate workloads and 71% for the high-miss-rate workloads. In this case the additional power is due to the addition of an entire second processor. The only reason the power does not double is that L2 conflicts between the two cores lead to stalls where clock gating is engaged, and this explains the lower power overhead of the L2-bound workloads.

Combining these two effects with the energy-delay-squared metric (ED^2) [23], we see that CMP is by far the most energy-efficient organization for benchmarks with reasonable L2 miss rates, while SMT is by far the most energy-efficient for those with high miss rates. Indeed, for L2-bound workloads, from the standpoint of ED^2, a single-threaded chip would be preferable to CMP, even though the single-threaded chip cannot run threads in parallel. Of course, this is at least in part due to the reduced L2 on the CMP chip.

5.3 DTM Results: Performance

For many traditional computing design scenarios, performance is the most critical parameter, and designers primarily care about power dissipation and thermal considerations because of thermal limits. In these cases, designers would like to optimize performance under DTM constraints. These include systems such as traditional PC desktops and certain high-performance server environments where energy utility costs are not critical.

To evaluate architectures viable for these situations, Figure 6 shows performance of SMT and CMP architectures with different DTM schemes. As we observed in the previous section, the results are again dependent on whether the workloads have high L2 miss ratio. For workloads with low or moderate miss ratios, CMP always gives the best performance, regardless of which DTM technique is used. On the other hand, for workloads that are mostly memory bound, SMT always gives better performance than CMP or ST. When comparing the DTM techniques, we found that DVS10, the DVS scheme assuming an optimistic 10 μs voltage switch time, usually gives very good performance. This is because DVS is very efficient at reducing chip power consumption, thus bringing chip temperature down very quickly and allowing the chip to quickly revert back to the highest frequency. When assuming a more pessimistic switching time of 20 μs, the performance of DVS degrades a lot, but is still among the best of the the DTM schemes.

However, in a system where energy consumption is not a primary concern, DVS may not be available due to the high implementation cost, while the relatively easier-to-implement throttling mechanisms are available. Therefore in the rest of this section, we only compare the performance of the various throttling strategies. Looking at the low L2 miss workloads (Left of Figure 6) and the high L2 miss workloads (Right of Figure 6), we find that SMT and CMP diverge with regards to the optimal throttling scheme. For CMP, fetch-throttling and register-occupancy throttling work equally well, and both outperform the local rename-

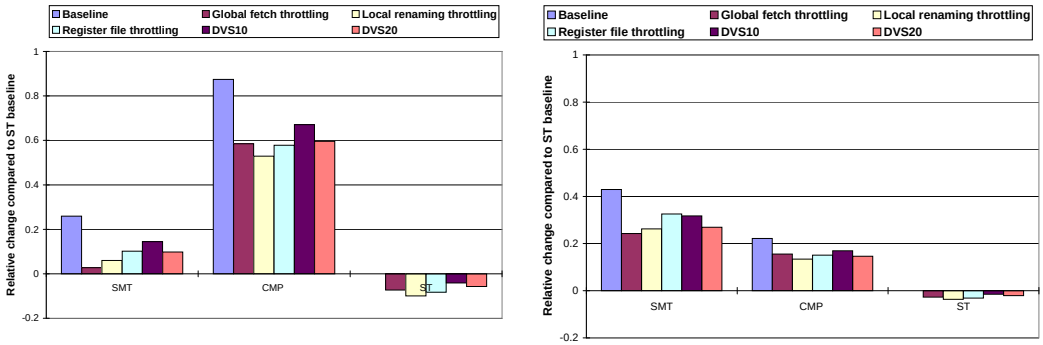


Figure 6: **Performance of SMT and CMP vs. ST. with different DTM policies, all with threshold temperature of 356K. Workloads with low L2 cache miss rate are shown on the left. Workloads with high L2 cache miss rate are shown on the right.**

throttling. For SMT, register throttling is the best performing throttling scheme, followed by rename-throttling and global fetch-throttling. In fact, for SMT running high L2 miss workloads, the local register occupancy throttling performs better than all of the other DTM techniques including DVS. The relative effectiveness of the DTM techniques illustrates the different heating mechanisms of CMP and SMT, with heating in the CMP chip a more global phenomenon, and heating in the SMT chip localized to key hotspot structures. For example, by directly resizing the occupancy of the register file, register-throttling is very effective at reducing the localized power density of the register file, and bringing down the temperature of the register file. In other words, the match-up between the mechanism of register-throttling and the inherent heat-up mechanism makes register-throttling the most effective DTM scheme for SMT. On the other hand, CMP mainly suffers from the global heat up effects due to the increased power consumption of the two cores. Thus global DTM schemes that quickly reduce the total power of the whole chip performs the best for CMP.

5.4 DTM Results: Energy

In many emerging high-performance computing environments, designers must optimize for raw performance under thermal packaging constraints, but energy consumption is also a critical design criteria for battery life or for energy utility costs. Examples of these systems are the high-performance mobile laptops, and servers designed for throughput oriented data centers like the Google cluster architecture [2].

In this scenario, designers often care about joint power-performance system metrics after DTM techniques have been applied. Figure 7 through Figure 9 shows the power and power-performance metrics (energy, energy-delay, and energy-delay^2) for the ST, SMT, and CMP architectures after applying the DTM techniques. All of the results in these figures are compared against the baseline ST machine without DTM. From these figure, we see that the dominating trend is that global DTM techniques, in particular DVS, tend to have superior energy-efficiency compared to the local techniques for most configurations. This is true because the global nature of the DTM mechanism means that a larger portion of the chip will be cooled, resulting in a larger savings. This is especially obvious for the DVS mechanism, because the DVS's cubic power savings is significantly higher than the power savings that the throttling techniques provide. The two local thermal management techniques, rename and register file throttling, do not contribute to a large power savings while enabled, as these techniques are designed to target specific temperature hotspots and thus have very little impact on global power dissipation. However, from an energy-efficiency point of view, these local technique can be competitive because in some cases they offer better performance than global schemes.

Figure 7 shows the results for the ST machine. Because DTM is rarely engaged for the ST architecture, there is a relatively small power overhead for these benchmarks. These ST results provide a baseline to decide whether SMT and CMP are still energy-efficient after DTM techniques are applied.

From Figure 8 we can see that the SMT architecture is superior to the ST architecture for all DTM techniques except for rename throttling. As expected, the DVS techniques perform quite well, although with high-L2 miss rate benchmarks register file throttling does nearly as well as DVS for ED^2, due to performance advantages.

Figure 9 allows us to compare CMP to the ST and SMT machines for energy-efficiency after applying DTM. When comparing CMP and SMT, we see that for the low-L2 miss rate benchmarks, the CMP architecture is always superior to the SMT architecture for all DTM configurations. In general, the local DTM techniques do not perform as well for CMP as they did for SMT. We see the exact opposite behavior when considering high-L2 miss rate benchmarks. In looking at the comparison between SMT and CMP ar-

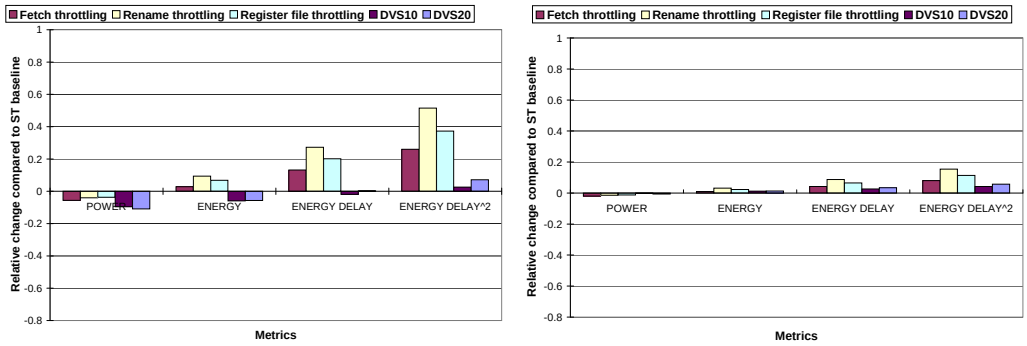


Figure 7: **Energy-efficiency metrics of ST with DTM, compared to ST baseline without DTM, for low-L2-miss-rate workloads (Left) and high-L2-miss-rate workloads (Right).**

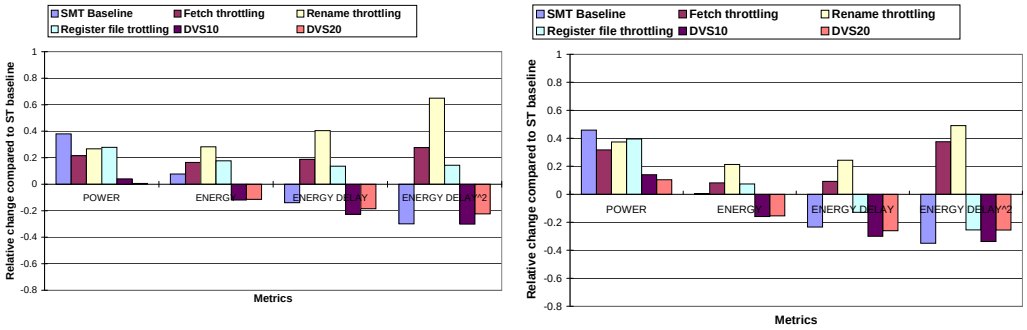


Figure 8: **Energy-efficiency metrics of SMT with DTM, compared to ST baseline without DTM, for low-L2-miss-rate benchmarks (Left) and high-L2-miss-rate benchmarks (Right).**

chitectures, we see that for the high-L2 miss rate benchmarks, CMP is not energy-efficient relative to *either* the baseline ST machine or the SMT machine—even with the DVS thermal management technique.

In conclusion, we find that for many, but not all configurations, global DVS schemes tend to have the advantage when energy-efficiency is an important metric. The results do suggest that there could be room for more intelligent localized DTM schemes to eliminate individual hotspots in SMT processors, because in some cases the performance benefits could be significant enough to beat out global DVS schemes.

6 Future Work and Conclusions

This paper provides an in-depth analysis of the performance, energy, and thermal issues associated with simultaneous multithreading and chip-multiprocessors. Our broad conclusions can be summarized as follows:

- CMP and SMT have similar thermal characteristics within current generation process technologies, but the thermal heating effects are quite different. SMT heating is primarily caused by localized heating within certain key microarchitectural structures such as the register file, due to increased utilization. CMP heating is primarily caused by the global impact of increased energy output.
- In future process technologies in which leakage power is a significant percentage of the overall chip power CMP machines are clearly hotter than SMT machines. For the SMT architecture, this is primarily due to the fact that the increased SMT utilization is overshadowed by additional leakage power. With the CMP machine, replacing the relatively cool L2 cache with a second core causes additional leakage power due to the temperature-dependent component of subthreshold leakage.
- CMP and SMT cores tend to perform better with different DTM techniques. In general, in performance-oriented systems, localized DTM techniques work better for SMT cores and global DTM techniques work better for CMP cores. For energy-oriented systems, global DVS thermal management techniques offer significant energy savings. However, the performance benefits of localized DTM make these techniques competitive for techniques for energy-oriented SMT machines.

In our future work, we hope to tackle the challenging problem of considering significantly larger amounts of thread-level parallelism and considering hybrids between CMP

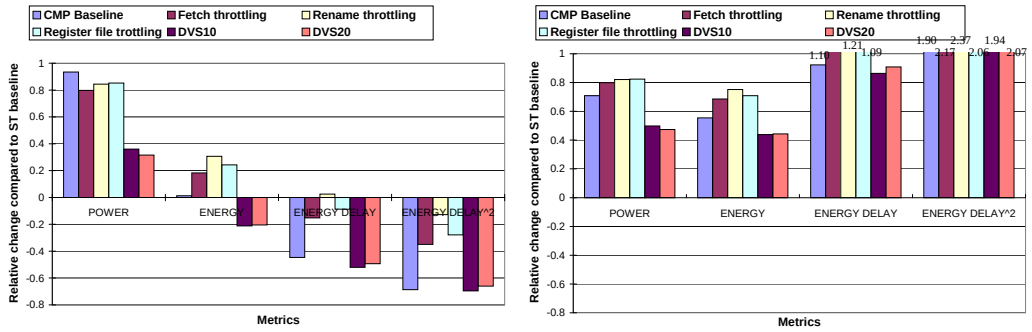


Figure 9: **Energy-efficiency metrics of CMP with DTM, compared to ST baseline without DTM, for low-L2-miss-rate benchmarks (Left) and high-L2-miss-rate benchmarks (Right).**

and SMT cores. There is also signifi cant opportunity to explore tradeoffs between core-level ILP and TLP exploitation from an energy and thermal standpoint. We also would like to explore server-oriented workloads which are likely to contain characteristics that are most similar to the memory-bound benchmarks from this study.

References

[1] A. Bakker and J. Huijsing. *High-Accuracy CMOS Smart Temperature Sensors*. Kluwer Academic, Boston, 2000.

[2] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: The google cluster architecture. *IEEE Micro*, 23(2):22–28, April 2003.

[3] D. Brooks, P. Bose, V. Srinivasan, M. Gschwind, P. G. Emma, and M. G. Rosenfi eld. New methodology for early-stage, microarchitecture-level power-performance analysis of microprocessors. *IBM Journal of Research and Development*, 47(5/6), 2003.

[4] J. Burns and J.-L. Gaudiot. Area and system clock effects on SMT/CMP processors. In *Proceedings of the 2001 International Conference on Parallel Architectures and Compilation Techniques*, pages 211–18, Sep. 2001.

[5] J. Clabes et al. Design and implementatin of the power5 microprocessor. In *ISSCC Digest of Technical Papers*, pages 56–57, Feb. 2004.

[6] J. Donald and M. Martonosi. Temperature-aware design issues for smt and cmp architectures. In *Proceedings of the 2004 Workshop on Complexity-Effective Design*, June 2004.

[7] L. Hammand, B. A. Nayfeh, and K. Olukotun. A single-chip multiprocessor. *IEEE Computer*, 30(9):79–85, Sep. 1997.

[8] S. Heo, K. Barr, and K. Asanovic. Reducing power density through activity migration. In *Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, Aug. 2003.

[9] Z. Hu, D. Brooks, V. Zyuban, and P. Bose. Microarchitecture-level power-performance simulators: Modeling, validation, and impact on design, Dec. 2003. Tutorial at the 36th Annual IEEE/ACM International Symposium on Microarchitecture.

[10] W. Huang, M. R. Stan, K. Skadron, S. Ghosh, K. Sankaranarayanan, and S. Velusamy. Compact thermal modeling for temperature-aware design. In *Proceedings of the 41st Design Automation Conference*, June 2004.

[11] W. Huang, M. R. Stan, K. Skadron, K. Sankaranarayanan, S. Ghoshy, and S. Velusamy. Compact thermal modeling for temperature-aware design. Technical Report CS-2004-13, University of Virginia Department of Computer Science, Apr. 2004.

[12] R. Kalla, B. Sinharoy, and J. Tendler. Power5: Ibm's next generation power microprocessor. In *Proc. 15th Hot Chips Symp*, pages 292–303, August 2003.

[13] K. Krewell. UltraSPARC IV mirrors predecessor: Sun builds dual-core chip in 130nm. *Microprocessor Report*, pages 1, 5–6, Nov. 2003.

[14] Y. Li, D. Brooks, Z. Hu, K. Skadron, and P. Bose. Understanding the energy effi ciency of simultaneous multithreading. In *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, Aug. 2004. To appear.

[15] D. T. Marr, F. Binns, D. L. Hill, G. Hinton, D. A. Koufaty, J. A. Miller, and M. Upton. Hyper-threading technology architecture and microarchitecture. *Intel Technology Journal*, 6(1):4–15, Feb. 2002.

[16] M. Moudgill, J.-D. Wellman, and J. H. Moreno. Environment for powerpc microarchitecture exploration. *IEEE Micro*, 19(3):15–25, 1999.

[17] R. Sasanka, S. V. Adve, Y. K. Chen, and E. Debes. The energy effi ciency of cmp vs. smt for multimedia workloads. In *Proceedings of the 18th Annual ACM International Conference on Supercomputing (ICS '04)*, June 2004.

[18] Y. Sazeides and T. Juan. How to compare the performance of two smt microarchitectures. In *Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software*, Nov. 2001.

[19] J. Seng, D. Tullsen, and G. Cai. Power-sensitive mutli-threaded architecture. In *Proceedings of the 2000 International Conference on Computer Design*, Sept. 2000.

[20] K. Skadron. Hybrid architectural dynamic thermal management. In *Proceedings of the 2004 Design, Automation and Test in Europe Conference*, Feb. 2004.

[21] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *Proceedings of the 30th Annual International Symposium on Computer Architecture*, pages 2–13, Apr. 2003.

[22] D. M. Tullsen, S. Eggers, and H. M. Levy. Simultaneous multithreading: Maximizing on-chip parallelism. In *Proceedings of the22th Annual International Symposium on Computer Architecture*, 1995.

[23] V. Zyuban and P. Strenski. Unifi ed methodology for resolving power-performance tradeoffs at the microarchitectural and circuit levels. In *Proc. of Int'l Symposium on Low-Power Electronics and Design*, August 2002.