CMP Design Space Exploration Subject to Physical Constraints

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

Abstract

This paper explores the multi-dimensional design space for chip multiprocessors, exploring the inter-related variables of core count, pipeline depth, superscalar width, L2 cache size, and operating voltage and frequency, under various area and thermal constraints. The results show the importance of joint optimization. Thermal constraints dominate other physical constraints such as pin-bandwidth and power delivery, demonstrating the importance of considering thermal constraints while optimizing these other parameters. For aggressive cooling solutions, reducing power density is at least as important as reducing total power, while for low-cost cooling solutions, reducing total power is more important. Finally, the paper shows the challenges of accommodating both CPU-bound and memory-bound workloads on the same design. Their respective preferences for more cores and larger caches lead to increasingly irreconcilable configurations as area and other constraints are relaxed; rather than accommodating a happy medium, the extra resources simply encourage more extreme optimization points.

1 Introduction

Recent product announcements show a trend toward aggressive integration of multiple cores on a single chip to maximize throughput. However, this trend presents an expansive design space for chip architects, encompassing the number of cores per die, core size and complexity (pipeline depth and superscalar width), memory hierarchy design, operating voltage and frequency, and so forth. Identifying optimal designs is especially difficult because the variables of interest are inter-related and must be considered simultaneously. Furthermore, trade-offs among these design choices vary depending both on workloads and physical (e.g., area and thermal) constraints.

We explore this multi-dimensional design space across a range of possible chip sizes and thermal constraints, for both CPU-bound and memory-bound workloads. Few prior works have considered so many cores, and to our knowledge, this is the first work to optimize across so many design variables simultaneously. We show the inter-related nature of these parameters and how the optimum choice of design parameters can shift dramatically depending on system constraints. Specifically, this paper demonstrates that:

- A simple, fast approach to simulate a large number of cores by observing that cores only interact through the L2 cache and shared interconnect. Our methodology uses single-core traces and only requires fast cache simulation for multi-core results.
- CPU- and memory-bound applications desire dramatically different configurations. Adaptivity helps, but any compromise incurs throughput penalties.
- Thermal constraints dominate, trumping even pin-bandwidth and power-delivery constraints. Once thermal constraints have been met, throughput is throttled back sufficiently to meet current pin-bandwidth and ITRS power-delivery constraints.
- A design must be optimized with thermal constraints. Scaling from the thermal-blind optimum leads to a configuration that is inferior, sometimes radically so, to a thermally optimized configuration.
- Simpler, smaller cores are preferred under some constraints. In thermally constrained designs, the main determinant is not simply maximizing the number of cores, but maximizing their power efficiency. Thermal constraints generally favor shallower pipelines and lower clock frequencies.
- Additional cores increase throughput, despite the resulting voltage and frequency scaling required to meet thermal constraints, until performance gains from an additional core is negated by the impact of voltage and frequency scaling across all cores.
- For aggressive cooling solutions, reducing power density is at least as important as reducing total power. For low-cost cooling solutions, however, reducing total power is more important.

This paper is organized as follows. Section 2 is the related work. We introduce our model infrastructure and validation in section 3. We present design space exploration results and explanations in section 4. We end with conclusions and proposals for future work in section 5.

2 Related Work

There has been a burst of work in recent years to understand the performance, energy, and thermal efficiency of different CMP organizations. Few have looked at a large numbers of cores and none, of which we are aware, have

0-7803-9368-6/06/$20.00/©2006 IEEE

jointly optimized across the large number of design parameters we consider while addressing the associated methodology challenges. Li and Martínez [17] present the most aggressive study of which we are aware, exploring up to 16-way CMPs for SPLASH benchmarks and considering power constraints. Their results show that parallel execution on a CMP can improve energy efficiency compared to the same performance achieved via single-threaded execution, and that even within the power budget of a single core, a CMP allows substantial speedups compared to single-threaded execution.

Kongetira et al. [12] describe the Sun Niagara processor, an eight-way CMP supporting four threads per core and targeting workloads with high degrees of thread-level parallelism. Chaudhry et al. [4] describe the benefits of multiple cores and multiple threads, sharing eight cores with a single L2 cache. They also describe the Sun Rock processor's "scouting" mechanism that uses a helper thread to prefetch instructions and data.

El-Moursy et al. [6] show the advantages of clustered architectures and evaluate a CMP of multi-threaded, multi-cluster cores with support for up to eight contexts. Huh et al. [10] categorized the SPEC benchmarks into CPU-bound, cache-sensitive, or bandwidth-limited groups and explored core complexity, area efficiency, and pin bandwidth limitations, concluding due to pin-bandwidth limitations that a smaller number of high-performance cores maximizes throughput. Ekman and Stenstrom [5] use SPLASH benchmarks to explore a similar design space for energy-efficiency with the same conclusions.

Kumar et al. [14] consider the performance, power, and area impact of the interconnection network in CMP architecture. They advocate low degrees of sharing, but use transaction oriented workloads with high degrees of inter-thread sharing. Since we are modeling throughput-oriented workloads consisting of independent threads, we follow the example of Niagara [12] and employ more aggressive L2 sharing. In our experiments, each L2 cache bank is shared by half the total number of cores. Interconnection design parameters are not variable in our design space at this time, and in fact constitute a sufficiently expansive design space of their own that we consider this to be beyond the scope of the current paper.

The research presented in this paper differs from prior work in the large number of design parameters and metrics we consider. We evaluate CMP designs for performance, power efficiency, and thermal efficiency while varying the number of cores per chip, pipeline depth and width, chip thermal packaging effectiveness, chip area, and L2 cache size. This evaluation is performed with a fast decoupled simulation infrastructure that separates core simulation from interconnection/cache simulation. By considering many more parameters in the design space, we demonstrate the effectiveness of this infrastructure and show the inter-relatedness of these parameters.

The methodologies for analyzing pipeline depth and width build on prior work by Lee and Brooks [16] by developing first-order models for capturing changes in core area as pipeline dimensions change, thereby enabling power density and temperature analysis. We identify optimal pipeline dimensions in the context of CMP architectures, whereas most prior pipeline analysis considers single-core microprocessors [8, 9, 22], Furthermore, most prior work in optimizing pipelines focused exclusively on performance, although Zyuban et al. found 18FO4 delays to be power-performance optimal for a single-threaded microprocessor [26].

Other researchers have proposed simplified processor models, with the goal of accelerating simulation. Within the microprocessor core, Karkhanis and Smith [11] describe a trace-driven, first-order modeling approach to estimate IPC by adjusting an ideal IPC to account for branch misprediction. In contrast, our methodology adjusts power, performance, and temperature estimates from detailed single-core simulations to account for fabric events, such as cache misses and bus contention. In order to model large scale multiprocessor systems running commercial workloads, Kunkel et al. [15] utilize an approach that combines functional simulation, hardware trace collection, and probabilistic queuing models. However, our decoupled and iterative approach allows us to account for effects such as latency overlap due to out-of-order execution, effects not easily captured by queuing models. Although decoupled simulation frameworks have been proposed in the context of single-core simulation (e.g., Kumar and Davidson [13]) with arguments similar to our own, our methodology is applied in the context of simulating multi-core processors.

3 Experimental Methodology

To facilitate the exploration of large CMP design spaces, we propose decoupling core and interconnect/cache simulation to reduce simulation time. Detailed, cycle-accurate simulations of multi-core organizations are expensive, and the multi-dimensional search of the design space, even with just homogeneous cores, is prohibitive. Decoupling core and interconnect/cache simulation dramatically reduces simulation cost with minimal loss in accuracy. Our simulation approach uses IBM's Turandot/PowerTimer, a cycle-accurate, execution-driven simulator, to generate single-core L2 cache-access traces that are annotated with timestamps and power values. We then feed these traces to Zauber, a cache simulator we developed that models the interaction of multiple threads on one or more shared interconnects and one or more L2 caches. Zauber uses hits and misses to shift the time and power values in the original traces. Generating the traces is therefore a one-time cost, while what would otherwise be a costly multiprocessor simulation is reduced to a much faster cache simulation. Using Zauber, it is cost-effective to search the entire multi-core design space.

3.1 Simulator Infrastructure

Our framework decouples core and interconnect/cache simulation to reduce simulation time. Detailed core simulation provides performance and power data for various

Fetch		Decode	
NFA Predictor	1	Multiple Decode	2
L2 I-Cache	11	Millicode Decode	2
L3 I-Load	8	Expand String	2
I-TLB Miss	10	Mispredict Cycles	3
L2 I-TLB Miss	50	Register Read	1
Execution		Memory	
Fix Execute	1	L1 D-Load	3
Float Execute	4	L2 D-Load	9
Branch Execute	1	L3 D-Load	77
Float Divide	12	Float Load	2
Integer Multiply	7	D-TLB Miss	7
Integer Divide	35	L2 D-TLB Miss	50
Retire Delay	2	StoreQ Forward	4

Table 1. 19FO4 Latencies (cycles).

	8D	4D	2D	1D
Functional Units				
FXU	4	2	1	1
MEM	4	2	1	1
FXU	4	2	1	1
BR	4	2	1	1
CR	2	1	1	1
Pipeline Stage Widths				
FETCH	16	8	4	2
DECODE	8	4	2	1
RENAME	8	4	2	1
DISPATCH	8	4	2	1
RETIRE	8	4	2	1

Table 2. Width Resource Scaling.

Structure	Energy Growth Factor
Register Rename	1.1
Instruction Issue	1.9
Memory Unit	1.5
Multi-ported Register File	1.8
Data Bypass	1.6
Functional Units	1.0

Table 3. Energy Scaling.

core designs, while interconnect/cache simulation projects the impact of core interaction on these metrics.

3.1.1 Core Simulation

Our detailed core simulation infrastructure consists of Turandot, PowerTimer, and HotSpot 2.0. Turandot is a validated model of an IBM POWER4-like architecture [19]. PowerTimer implements circuit-extracted, validated power models, which we have extended with analytical scaling formulas based on Wattch [1, 2]. HotSpot 2.0 is a validated, architectural model of localized, on-chip temperatures [21]. Each of these components in our detailed simulation infrastructure is modular so that any particular simulator can be replaced with an alternative. We extended Turandot and PowerTimer to model the performance and power as pipeline depth and width vary using techniques from prior work [16].

Depth Performance Scaling: Pipeline depth is quantified in terms of FO4 delays per pipeline stage.[1] The performance model for architectures with varying pipeline depths are derived from the reference 19FO4 design by treating the total number of logic levels as constant and independent of the number of pipeline stages. This is an abstraction for the purpose of our analysis; increasing the pipeline depth could require logic design changes. The baseline latencies (Table 1) are scaled to account for pipeline depth changes according to Eq. (1). These scaled latencies account for latch delays ($FO4_{latch} = 3$) and all latencies have a minimum of one cycle. This is consistent with prior work in pipeline depth simulation and analysis for a single-threaded core [26].

$$Lat_{target} = \left\lfloor Lat_{base} \times \frac{FO4_{base} - FO4_{latch}}{FO4_{target} - FO4_{latch}} + 0.5 \right\rfloor \quad (1)$$

Depth Power Scaling: Each factor in the standard equation for dynamic power dissipation, Eq. (2), scales with pipeline depth. The clock frequency f increases linearly with depth as the delay for each pipeline stage decreases. The clock gating factor CGF decreases by a workload dependent factor as pipeline depth increases due to the increased number of cycles in which the shorter pipeline stages are stalled. As the true switching factor α is independent of the pipeline depth and the glitching factor β decreases with pipeline depth due to shorter distances between latches, switching power dissipation decreases with pipeline depth. The latch count, and consequently hold power dissipation, increases linearly with pipeline depth. We refer the reader to prior work for a detailed treatment of these scaling models [26].

$$P_{dyn} = CV^2 f(\alpha + \beta) \times CGF \quad (2)$$

Width Performance Scaling: We quantify the pipeline width in terms of the maximum number of instructions decoded per cycle. Performance data for architectures with varying pipeline widths are obtained from the reference 4-decode design (4D) by a linear scaling of the number of functional units and the number of non-branch instructions fetched, decoded, renamed, dispatched, and retired per cycle (Table 2). All pipelines have at least one instance of each functional unit. As pipeline width decreases, the number of instances of each functional unit is quickly minimized to one. Thus, the decode width becomes the constraining parameter for instruction throughput for the narrower pipelines we consider (e.g., 2D).

Width Power Scaling: We employ a hybrid approach to model the power impact of scaling the width of the pipeline. Our baseline microarchitecture, based on the POWER4, includes a clustered backend microarchitecture for structures like the functional units, issue queues, and register files. This approach is effective at managing complexity, cycle time, and power dissipation in wide-issue superscalar cores [3, 20, 25]. An analogous technique is used to construct the dual-ported data cache. When scaling the width of these structures, we assume that unconstrained hold and switching power increases linearly with the number of functional units, access ports, and any other parameter that must change as width varies.

[1] Fan-out-of-four (FO4) delay is defined as the delay of one inverter driving four copies of an equally sized inverter. When logic and overhead per pipeline stage is measured in terms of FO4 delay, deeper pipelines have smaller FO4 delays.

In certain non-clustered structures, however, linear power scaling may be inaccurate and, for example, does not capture non-linear relationships between power and the number of SRAM access ports since it does not account for the additional circuitry required in a multi-ported SRAM cell. For this reason, we apply superlinear power scaling with exponents (Table 3) drawn from Zyuban's work in estimating energy growth parameters [25]. Since these parameters were experimentally derived through analysis of non-clustered architecture, we only apply this power scaling to the non-clustered components of our architecture.

3.1.2 Interconnection/Cache Simulation

The core simulators are supplemented by Zauber, a much faster simulator that performs interpolation on L2 cache traces provided by the core simulators. Zauber decouples detailed core simulation and the simulation of core interaction. The cores in a CMP architecture usually share one or more L2 caches through an interconnection fabric. Therefore, resource contention between cores occurs primarily in these two resources. We find it possible to simulate cache and fabric contention independent of core simulations without losing too much accuracy. The impact of contention on the performance and power of each core may then be evaluated quickly using interpolation.

First, we collect L2 access traces based on L1 cache misses through one pass of single-core simulations with a specific L2 cache size (0.5MB in our experiment). We found these L2 traces to be independent of the L2 cache size. In these traces we record the L2 cache address and access time (denoted by the cycle) information for every access. We also need to sweep through a range of L2 cache sizes for each benchmark and record the performance and microarchitectural resource utilization every 10k instructions as this information will be used in the interpolation. These L2 traces are fed into an cache simulator and interconnection-contention model that reads the L2 accesses of each core from the traces, sorts them according to time of access, and uses them to drive the interconnection and L2 cache simulation. This interconnection/cache simulator outputs the L2 miss ratio and the delay due to contention for every 10k instruction segment of the thread running on each core.

With this L2 miss ratio and interconnection contention information we calculate the new performance and power number for each 10k instruction segment of all the threads. Since we know the performance and microarchitectural resource utilization for several L2 miss ratio values, we are able to obtain new performance and utilization data for any other L2 miss ratio produced by the cache simulator via interpolation. Power numbers can be derived from the structure utilization data with post-processing.

When we interleave the L2 accesses from each thread, we are using the cycle information attached with each access to sort them by time of access. However, each thread may suffer different degrees of performance degradation due to interconnection and L2 cache contention. Therefore, sorting by time of access may not reflect the real ordering. In our model, we iterate to improve accuracy. In particular, given the performance impact from cache contention for each thread, we can use this information to adjust the time of each L2 access in each L2 trace and redo L2 cache emulation based on this new L2 access timing information. Iterating to convergence, we find three iterations are typically enough to reach good accuracy.

We validate Zauber against our detailed cycle-accurate simulator, Turandot. Figure 1 shows the average performance and power data from Turandot simulation and Zauber simulation for 2-way and 4-way CMPs. From these figures, the average performance and power difference between Turandot and Zauber is within 1%. For a 2-way CMP, Zauber achieves a simulation time speedup of 40-60x, with detailed Turandot simulations requiring 1-2 hours and the decoupled simulator requiring 1-3 minutes.

Since we are modeling throughput-oriented workloads consisting of independent threads, we consider a relatively high degree of cache sharing Niagara [12]. Each L2 cache bank is shared by half the total number of cores. The interconnection power overheads are extrapolated from [14].

We assume the L2 cache latency does not change when we vary the L2 cache size. We also omit the effects of clock propagation on chip throughput and power when core number increases.

3.2 Analytical Infrastructure

We use formulas to vary and calculate parameters of interest in the CMP design space exploration. The design parameters we consider include core count, core pipeline dimensions, thermal resistance of chip packaging, and L2 cache size. As we vary these parameters, we consider the impact on both power and performance metrics.

3.2.1 Performance and Power Modeling

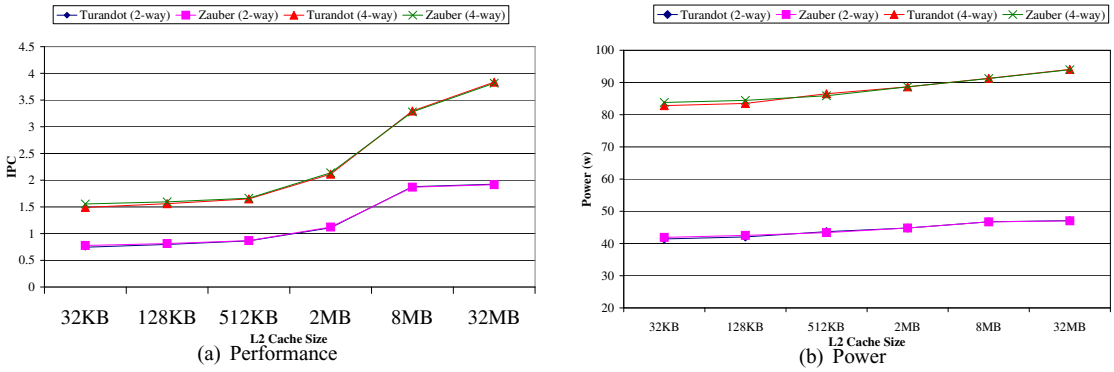
The analytical model uses performance and dynamic power data generated by Zauber simulation. Leakage power density for a given technology is calculated by Eq. (3), where A and B are coefficients determined by a linear regression of ITRS data and T is the absolute temperature. $A = 207.94$ and $B = 1446$ for 65nm technology.

$$P_{\text{leakage density}} = A \cdot T^2 \cdot e^{-B/T} \quad (3)$$

3.2.2 Temperature Modeling

We use steady-state temperature at the granularity of each core to estimate the chip thermal effects. This neglects localized hotspots within a core as well as lateral thermal coupling among cores. Addressing these is important future work, but employing a simple analytical temperature formula instead of the more complex models in HotSpot reduces simulation time and allows us to focus on how heat-removal limitations constrain core count and core type.

We observe that the heat spreader is almost isothermal for the range of the chip areas and power values we investigate, so we can separate the global temperature rise across the thermal package due to total chip power dissipation from localized temperature rise above the package



(a) Performance (b) Power

Figure 1. The validation of Zauber model.

due to per-core power dissipation. This is described by Equations 4, 5, and 6. Suppose we want to calculate the temperature of a specific core on the chip, where P_{glo} and P_{core} are the global chip and single core dynamic power, respectively. Similarly, L_{glo} and L_{core} are the global chip and single core leakage power, respectively. The chip's total dynamic power is the sum of the dynamic power dissipated by all the cores on chip, the L2 cache and the interconnect. The chip leakage power is summed in a similar manner. The sum of R_{spread} and R_{hsink} denotes the thermal resistance from the heat spreader to the air and the sum of $R_{silicon}$ and R_{TIM} denotes the thermal resistance from the core. Collectively, these parameters specify the chip's thermal characteristics from the device level to the heat spreader, ignoring the lateral thermal coupling above the heat spreader level.

We categorize the CMP heatup into local and global effects. The former is determined by the local power dissipation of any given core and the effect on its temperature. The latter is determined by the global chip power.

$$H_{glo} + H_{loc} = T_{core} - T_{amb} \quad (4)$$

$$H_{glo} = (P_{glo} + L_{glo}) \cdot (R_{spread} + R_{hsink}) \quad (5)$$

$$H_{loc} = (P_{core} + L_{core}) \cdot (R_{silicon} + R_{TIM}) \quad (6)$$

This distinction between local and global heatup mechanisms is first qualitatively introduced by Li et al. in [18]. They observed that adding cores to a chip of fixed area increases chip temperature. This observation may not be evident strictly from the perspective of per-core power density. Although power density is often used as a proxy for steady-state temperature with each core exhibiting the same power density, core or unit power density is only an accurate predictor of the temperature increases in the silicon relative to the package. Per-unit or per-core power density is analogous to one of the many thermal resistances comprising the entire network that represents the chip.

Adding cores does indeed increase temperature, because it increases the total amount of power that must be removed. The current primary heat removal path is convection from a heat sink. Although accurate expressions for convective heat transfer are complex, a first-order approximation is:

$$q = hA(T_{sink} - T_{air}) \quad (7)$$

where q is the rate of convective heat transfer, h is the convective heat transfer coefficient that incorporates air speed and various airflow properties, A is the surface area for convection, and to first order we can assume T_{air} is fixed. At steady-state, the total rate of heat P generated in the chip must equal the total rate of heat removed from the chip. If we hold h and A constant, then as we add cores and P exceeds q, T_{sink} must rise to balance the rates so that $P = q$. This increases on-chip temperatures because the sink temperature is like an offset for the other layers from the sink-spreader interface through the chip.

Alternative heat removal mechanisms also warrant consideration. For example, fan speed may be increased, but this approach is often limited by acoustical limits and various board-layout and airflow factors that lead to diminishing returns (e.g. increased pressure drop across a larger heat sink). We can lower the inlet air temperature, but this is not an option in many operating environments (e.g. a home office), or may be extremely costly (e.g. in a large data center). We could also increase heat sink area, but this is where Eq. (7) breaks down. That expression assumes that the heat source is similar in size to the conductive surface. In reality, increasing the heat sink surface area does not improve convective heat transfer in a simple way. Increasing fin height and surface area is limited by airflow constraints that dictate an optimal fin configuration. Increasing the total size of the heat sink (i.e. increasing the area of its base), leads to diminishing returns as the ratio of sink to chip area increases due to limitations on how well the heat can be spread. In the limit, the heat source looks like a point source and further increases in the sink area will have no benefit, as heat will not be able to spread at all to the outermost regions of the heat sink.

With regard to the single thermal resistance in the HotSpot model, adding cores is equivalent to adding current sources connected in parallel to a single resistor, the sink-to-air resistance. The increased current leads to a larger IR drop across this resistor and a proportionally larger heat-sink temperature.

Equations 4, 5, and 6, quantify the contributions from global and local heatup. Figure 2 presents results from validating this simple model against HotSpot, varying the heat sink resistance and fixing the power distribution to test different temperature ranges. The temperature difference be-

tween these two models is normally within 3°.

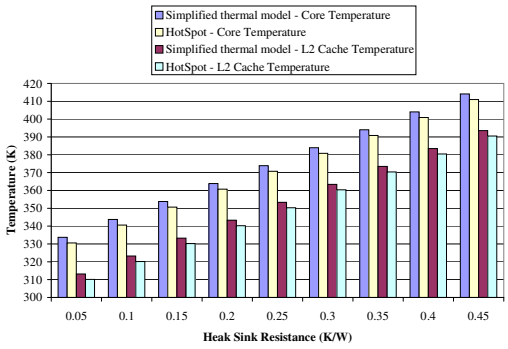


Figure 2. Simplified temperature model validation.

3.2.3 Area Modeling

We assume a 65nm technology. Based on a Power5 die photo, we estimate the baseline core area to be $11.52mm^2$, equivalent to the area of 1MB of L2 cache. We assume each $n/2$ cores share one L2 cache through a crossbar routing over the L2 and estimate the total crossbar area to be $6.25n \cdot mm^2$ [14], where n is the number of cores. As pipeline dimensions vary, we scale the core area to account for additional structures and overhead.

Depth Area Scaling: Given our assumption of fixed logic area independent of pipeline depth, latch area constitutes the primary change in core area as depth varies. Let w_{latch} be the total channel width of all pipeline latch transistors, including local clock distribution circuitry. Let w_{total} be the total channel width for all transistors in the baseline microprocessor, excluding all low-leakage transistors in on-chip memories. Let the latch growth factor (LGF) capture the latch count growth due to logic shape functions. In our analysis, we take the latch ratio (w_{latch}/w_{total}) to be 0.3 and the LGF to be 1.1, assuming superlinear latch growth as pipeline depth increases [26]. Assuming changes in core area are proportional to the total channel width of latch transistors in the pipeline, we scale the portion of core area attributed to latches superlinearly with pipeline depth using Eq. (8).

$$A_{target} = A_{base}\left(1 + \frac{w_{latch}}{w_{total}}\left(\left(\frac{FO4_{target}}{FO4_{base}}\right)^{LGF} - 1\right)\right) \quad (8)$$

Width Area Scaling: Table 4 presents area scaling factors for varying pipeline width. We consider each unit and its underlying macros. To first-order, the core area attributed to the fixed point, floating point, and load store units scale linearly due to clustering. We also assume the area of multi-ported SRAM array structures is wire dominated and scales linearly with the number of ports [23, 24]. This assumption applies to SRAM memories (*e.g.* register files), but may be extended to queues (*e.g.* issue queues), tables (*e.g.* rename mappers), and other structures potentially implemented as an SRAM array.

Note the area of the instruction fetch and decode units (IFU, IDU) are independent of width. Within the fetch unit, the instruction cache, instruction TLB, program counter, and branch handling hardware dominate the fetch unit's total power dissipation and, to first-order, these structures are independent of the fetch width. Within the decode unit, the instruction decoder ROM used to crack complex instructions dominates decode power and, to first-order, this ROM is independent of decode width. Also note that only a subset of the macros for the instruction sequencing unit (ISU) scale as width increases, resulting in a sublinear area dependence on width for this unit. For the sequencing unit, only area associated with issue queues and tables for register renaming scale with pipeline width. The total scaling factors, a weighted average of the unit scaling factors, suggest a sublinear relationship between area and width.

Unit/Macro	2D	4D	8D
FXU	0.5	1.0	2.0
FPU	0.5	1.0	2.0
ISU	0.6	1.0	1.8
IFU	1.0	1.0	1.0
LSU	0.5	1.0	2.0
IDU	1.0	1.0	1.0
Total	**0.7**	**1.0**	**1.7**

Table 4. Pipeline Width Area Scaling.

3.2.4 DVFS Scaling and Reward Functions

Using a large number of cores may lead to thermal runaway due to high chip power and the positive feedback of leakage power and temperature. We must employ a thermal control mechanism to prevent this behavior and to account for the resulting performance impact. We take this control into consideration by emulating voltage and frequency scaling for steady-state temperature control. Our dynamic simulations do not model the dynamic control aspect of DVFS. Instead, we only simulate workloads in which all cores are occupied — "worst typical-case" workloads that are likely to dictate thermal design. Then, for a given workload, we calculate its steady-state temperature and infer the voltage and frequency settings necessary to prevent the steady-state temperature from pushing the chip above 100°. These settings could represent the maximum steady-state or nominal settings that are safe for "worst typical-case" workloads, or could represent steady-state V/f values with DVFS when these workloads are running. In reality, the DVFS settings would fluctuate around these values with such workloads, permitting higher settings when fewer cores are occupied.

$$\begin{aligned} T_{thr} - T_{amb} &= (P_{global}R_{below} + P_{core}R_{above})V_{sc}^2 F_{sc} \\ &\quad + (L_{global}R_{below} + L_{core}R_{above})V_{sc} \quad (9) \end{aligned}$$

For a given core number n, L2 cache size l, pipeline depth d, and pipeline width w, we obtain the dynamic power consumption and performance from Zauber and the leakage power with Equation 3. For a given temperature threshold, we calculate the voltage and frequency scaling factors from Equation 9, which is deduced from Equation 4, assuming that the leakage power is mainly subthreshold leakage power and is linearly dependent on voltage. Using 0.9V and 2.0 GHz as the nominal voltage and clock

frequency and their scaling factors as V_{sc} and F_{sc}, we use a nonlinear voltage/frequency relationship obtained from HSPICE circuit simulation. After determining the voltage and frequency scaling required for thermal control, we calculate our reward functions, BIPS and BIPS^3/W, with Equations 10 and 11.

$$BIPS(n,l,d,w) = BIPS_{base} \cdot F_{sc} \quad (10)$$

$$\frac{BIPS^3}{W}(n,l,d,w) = \left(\frac{{BIPS_{base}}^3}{P_{dyn} + \frac{P_{leak}}{V_{sc}F_{sc}}}\right)\left(\frac{F_{sc}}{V_{sc}}\right)^2 \quad (11)$$

3.3 Workloads

We characterize all SPEC2000 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-point or integer benchmark. We employ eight of the SPEC2000 benchmarks (art, mcf, applu, crafty, gcc, eon, mgrid, swim) as our single thread benchmarks, spanning these categories. We further categorize benchmarks according to their L2 miss ratios, referring to those with high and low miss ratios as memory- and CPU- bound, respectively.

To generate static traces, we compile with the *xlc* compiler and -O3 option. We use Simpoint [7] to identify representative simulation points and generate traces by capturing 100 million instructions beginning at the Simpoint.

For both CPU-bound and memory-bound benchmarks, we use pairs of single-thread benchmarks to form dual-thread benchmarks and replicate these pairs to form multiple benchmark groups of each benchmark category for CMP simulation with more than two cores. We only simulate workloads consisting of a large pool of waiting threads to keep all cores active, representing the "worst typical-case" operation likely to determine physical limits.

4 Results

We present the results from the exploration of a large CMP design space that encompasses core count, pipeline dimensions, and cache size. We consider optimizing for performance (BIPS) and power-performance efficiency (BIPS^3/W) under various area and thermal constraints. In addition to demonstrating the effectiveness of our experimental methodology for exploring large design spaces, our results also quantify significant CMP design trends and demonstrate the need to make balanced design choices.

4.1 Optimal Configurations

Table 5 presents optimal configurations that maximize BIPS and BIPS^3/W for a fixed pipeline depth while Table 6 presents optima for a fixed superscalar width. Configurations are presented for various combinations of area and thermal constraints. The area constraint can take on one of four values: no constraint ("nolimit"), $100mm^2$, $200mm^2$, or $400mm^2$. Similarly, packaging assumptions and hence thermal constraints can take on one of three values: no constraint (NT), low constraint (LR=0.1, low thermal resistance, i.e. aggressive, high-cost thermal solution), and high constraint (HR=0.5, high thermal resistance, i.e. constrained thermal solution, such as found in a laptop). The tables differentiate between CPU- and memory-bound benchmarks and specify the required voltage and frequency ratios needed to satisfy thermal constraints.

Figures 3–5 present performance trade-offs between core count, L2 cache size, and pipeline dimensions for a $400mm^2$ chip subject to various thermal constraints.

4.1.1 No Constraints

In the absence of area and thermal constraints (nolimit+NT+CPU, nolimit+NT+MEMORY), the throughput maximizing configuration for both CPU- and memory-bound benchmarks employs the largest L2 cache and number of cores. Although the optimal pipeline width for all benchmarks is eight (8W), CPU-bound benchmarks favor deeper pipelines (12FO4) to take advantage of fewer memory stalls and higher instruction level parallelism. Conversely, memory-bound benchmarks favor relatively shallow pipelines (18FO4).

For BIPS^3/W, the optimal depth shifts to shallower pipelines; 18FO4 and 30FO4 delays per stage are optimal for CPU and memory-bound benchmarks, respectively. The optimal width shifts to shallower, narrower pipelines for memory-bound benchmarks due to the relatively high rate of memory stalls and low instruction level parallelism.

4.1.2 Area Constraints

Considering area constraints ({100,200,400}+NT+*), we find core number and L2 cache size tend to decrease as area constraints are imposed. Although both techniques are applied in certain cases (100+NT+CPU, 100+NT+MEMORY), decreasing the cache size is naturally the most effective approach to meet area constraints for CPU-bound benchmarks, while decreasing the number of cores is most effective for memory-bound benchmarks.

With regard to pipeline dimensions, we find the optimal width decreases to 2W for all area constraints on memory-bound benchmarks (*+NT+MEMORY) except 100+NT+MEMORY. According to the area models in Section 3.2.3, changes in depth scale the latch area (only 30% of total area) whereas changes in width scale the area associated with functional units, queues, and other width-sensitive structures. Thus, shifting to shallower widths provides greater area impact (Table 5). Although pipeline depths may shift from 12 to 18/24FO4 delays per stage, they are never reduced to 30FO4 delays per stage to meet area constraints (Table 6).

As in the case without constraints, the bar plots in Figure 3, which vary pipeline depth, shows CPU-bound

	BIPS						$BIPS^3/W$				
	L2 (MB)	Core Number	Pipeline Width	Voltage Scaling	Frequency Scaling		L2 (MB)	Core Number	Pipeline Width	Voltage Scaling	Frequency Scaling
nolimit+NT+CPU	32	20	8	1.00	1.00		16	20	8	1.00	1.00
nolimit+LR+CPU	8	20	4	0.75	0.63		8	20	4	0.75	0.63
nolimit+HR+CPU	2	18	2	0.59	0.39		2	16	2	0.61	0.43
400+NT+CPU	4	20	4	1.00	1.00		4	20	4	1.00	1.00
400+LR+CPU	4	20	4	0.75	0.64		2	20	4	0.76	0.65
400+HR+CPU	2	18	2	0.59	0.39		2	16	2	0.61	0.43
200+NT+CPU	2	10	4	1.00	1.00		2	10	4	1.00	1.00
200+LR+CPU	2	10	4	0.87	0.80		2	12	2	0.90	0.85
200+HR+CPU	2	12	2	0.67	0.51		2	12	2	0.67	0.51
100+NT+CPU	2	4	4	1.00	1.00		2	4	4	1.00	1.00
100+LR+CPU	2	4	4	0.97	0.96		2	4	4	0.97	0.96
100+HR+CPU	2	4	4	0.79	0.70		2	4	4	0.79	0.70
nolimit+NT+MEMORY	32	20	8	1.00	1.00		32	20	4	1.00	1.00
nolimit+LR+MEMORY	16	20	4	0.73	0.61		16	20	4	0.73	0.61
nolimit+HR+MEMORY	8	10	2	0.62	0.45		8	10	2	0.62	0.45
400+NT+MEMORY	16	16	2	1.00	1.00		16	16	2	1.00	1.00
400+LR+MEMORY	16	12	4	0.81	0.73		16	16	2	0.81	0.72
400+HR+MEMORY	8	10	2	0.62	0.45		8	10	2	0.62	0.45
200+NT+MEMORY	8	8	2	1.00	1.00		8	8	2	1.00	1.00
200+LR+MEMORY	8	6	4	0.93	0.90		8	8	2	0.92	0.88
200+HR+MEMORY	8	8	2	0.66	0.51		8	8	2	0.66	0.51
100+NT+MEMORY	2	4	4	1.00	1.00		4	4	2	1.00	1.00
100+LR+MEMORY	2	4	4	1.00	1.00		4	4	2	0.98	0.98
100+HR+MEMORY	2	4	4	0.81	0.73		4	4	2	0.81	0.73

Table 5. Optimal Configurations with Varying Pipeline Width, Fixed Depth (18FO4)

	BIPS						$BIPS^3/W$				
	L2 (MB)	Core Number	Pipeline Depth	Voltage Scaling	Frequency Scaling		L2 (MB)	Core Number	Pipeline Depth	Voltage Scaling	Frequency Scaling
nolimit+NT+CPU	32	20	12	1.00	1.00		16	20	18	1.00	1.00
nolimit+LR+CPU	8	20	18	0.75	0.63		8	20	18	0.75	0.63
nolimit+HR+CPU	2	14	24	0.62	0.44		2	14	24	0.62	0.44
400+NT+CPU	4	18	12	1.00	1.00		4	20	18	1.00	1.00
400+LR+CPU	4	20	18	0.75	0.64		2	20	24	0.85	0.78
400+HR+CPU	2	14	24	0.62	0.44		2	14	24	0.62	0.44
200+NT+CPU	2	10	18	1.00	1.00		2	10	18	1.00	1.00
200+LR+CPU	2	10	18	0.87	0.80		2	10	24	0.97	0.95
200+HR+CPU	2	10	18	0.63	0.45		2	10	24	0.69	0.55
100+NT+CPU	2	4	18	1.00	1.00		2	4	18	1.00	1.00
100+LR+CPU	2	4	18	0.97	0.96		2	4	18	0.97	0.96
100+HR+CPU	2	4	18	0.79	0.70		2	4	18	0.79	0.70
nolimit+NT+MEMORY	32	20	18	1.00	1.00		32	20	30	1.00	1.00
nolimit+LR+MEMORY	16	20	30	0.85	0.78		16	20	30	0.85	0.78
nolimit+HR+MEMORY	8	10	30	0.65	0.48		8	10	30	0.65	0.48
400+NT+MEMORY	16	12	18	1.00	1.00		16	12	30	1.00	1.00
400+LR+MEMORY	16	12	30	0.94	0.91		8	12	30	1.00	1.00
400+HR+MEMORY	8	10	30	0.65	0.48		8	10	30	0.65	0.48
200+NT+MEMORY	8	6	24	1.00	1.00		8	6	30	1.00	1.00
200+LR+MEMORY	8	6	24	1.00	1.00		8	6	30	1.00	1.00
200+HR+MEMORY	4	6	30	0.83	0.75		4	6	30	0.83	0.75
100+NT+MEMORY	2	4	24	1.00	1.00		2	4	30	1.00	1.00
100+LR+MEMORY	2	4	24	1.00	1.00		2	4	30	1.00	1.00
100+HR+MEMORY	2	4	30	0.96	0.95		2	4	30	0.96	0.95

Table 6. Optimal Configurations with Varying Pipeline Depth, Fixed Width (4D)

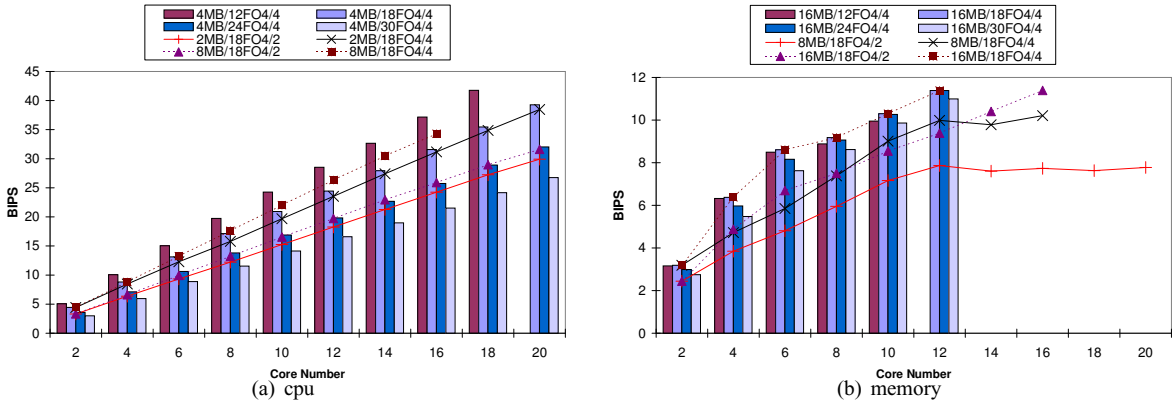
benchmarks favor deeper pipelines (4MB/12FO4/4 is optimal) and memory-bound benchmarks favor shallower pipelines (16MB/18FO4/4 or 16MB/24FO4/4 are optimal). The line plots in Figures 3–4 also present performance for varying widths for modest thermal constraints. In this case, the optimal pipeline width is 4W for a fixed depth of 18FO4 delays per stage.

4.1.3 Thermal Constraints

We find thermal constraints (nolimit+{NT,LR,HR}+*), also shift optimal configurations to fewer and simpler cores. The optimal core number and L2 size tends to decrease with heat sink effectiveness. For example, the optimum for nolimit+HR+MEMORY is 8MB L2 cache and 10 cores. Again, CPU-bound benchmarks favor decreasing cache size to meet thermal constraints while memory-bound benchmarks favor decreasing the number of cores.

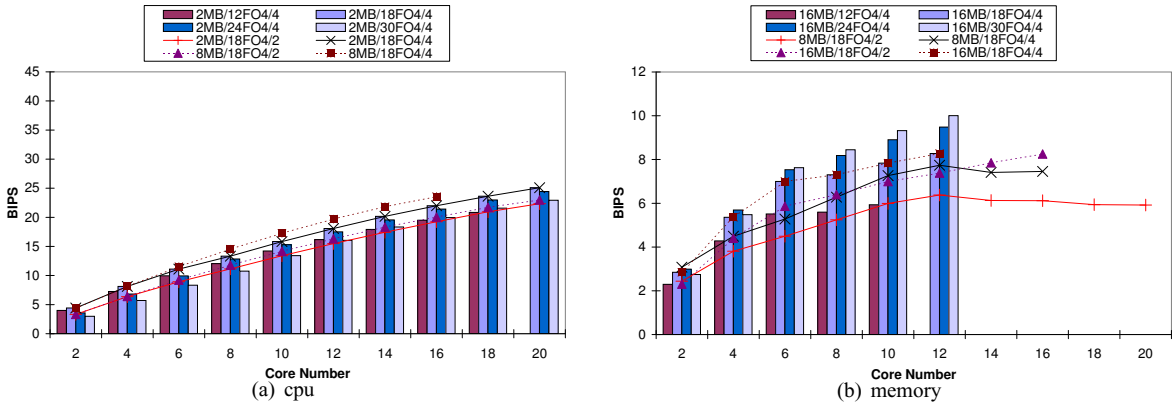
Figure 5 also illustrates the impact of global heating on optimal pipeline configurations. As the number of cores increase for CPU-bound benchmarks, the optimal delay per stage increases by 6FO4 (i.e., from 18 to 24FO4) when twelve cores reside on a single chip. The increasing core count increases chip temperature, leading to shallower pipelines that lower power dissipation, lower global temperature, and meet thermal constraints.

Simpler cores, characterized by smaller pipeline dimensions, tend to consume less power and, therefore, miti-



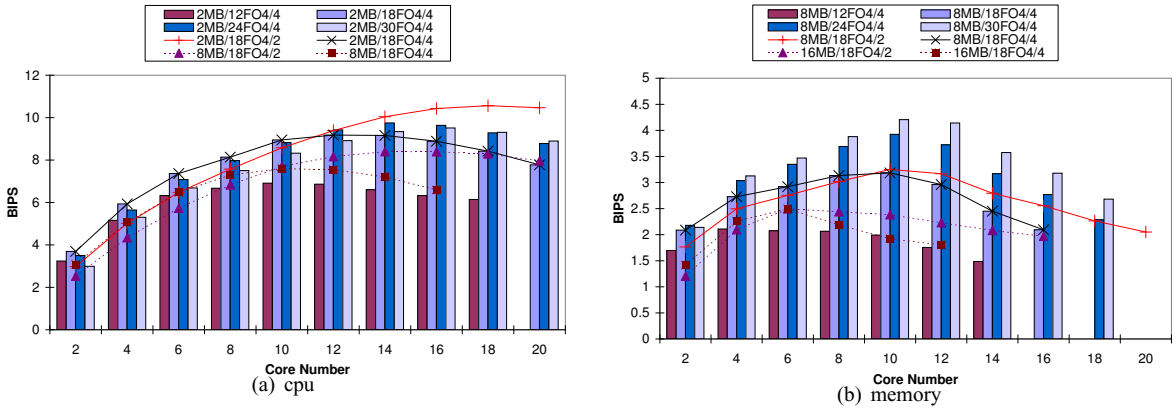
(a) cpu (b) memory

Figure 3. Performance of various configurations with chip area constraint at 400mm^2 (without thermal control).



(a) cpu (b) memory

Figure 4. Performance of various configurations with chip area constraint at 400mm^2 (R = 0.1 heat sink).



(a) cpu (b) memory

Figure 5. Performance of various configurations with chip area constraint at 400mm^2 (R = 0.5 heat sink).

gate the core's thermal impact. In particular, the optimal pipeline depth shifts to 24 and 30FO4 delays per stage for CPU and memory-bound benchmarks, respectively, when comparing nolimit+NT+* to nolimit+HR+* in Table 6. Similarly, the optimal width shifts to 2W for all benchmarks when comparing the same entries in Table 5.

Figures 4–5 show imposing thermal constraints shifts the optimal depth to shallower design points. The performance for CPU and memory-bound benchmarks are maximized for 18-24 and 24-30FO4 delays per stage, respectively. Pipeline power dissipation increases superlinearly with depth while pipeline area increases sublinearly according to Section 3.2.3. Thus, growth in power dissipation exceeds area growth and the overall power density increases with depth. Thus, optimal designs must shift to shallower pipelines to meet thermal constraints. Similarly, more aggressive thermal constraints, shown in Figure 5 shifts the optimal width to the narrower 2W, especially as the number of cores increases. These results also suggest thermal constraints will have a greater impact on pipeline configurations than area constraints.

4.1.4 Area and Thermal Comparison

Comparing the impact of thermal constraints (nolimit+NT+* versus nolimit+HR+*) to the impact of area constraints (nolimit+NT+* versus 100+NT+*) demonstrates larger shifts towards smaller pipeline dimensions. In general, thermal constraints exert a greater influence on the optimal design configurations.

Applying a more stringent area constraint reduces the trend towards simpler cores. With a smaller chip area, resulting in fewer cores and smaller caches, total power dissipated and the need for thermal control is diminished. As this occurs, pressure towards simpler cores with smaller pipeline dimensions also fades.

4.1.5 Depth and Width Comparison

Consider a baseline configuration 2MB/18FO4/4W. As thermal constraints are imposed, the configuration may either shift to a shallower core (2MB/24FO4/4W) or shift to a narrower core (2MB/18FO4/2W). Since changes in width scale area for both functional units and many queue structures, whereas changes in depth only scale area for latches between stages, width reductions have a greater area impact relative to depth reductions. Thus, the 2MB/24FO4/4W core is a larger core relative to the 2MB/18FO4/2W and exhibits lower dynamic power density. However, the smaller 2MB/18FO4/2W core benefits from less leakage power per core and, consequently, less global power (since dynamic power dissipation is comparable for both cores).

From our temperature models in Section 3.2.2, total power output, P_{global}, has greater thermal impact for a chip with a poor heat sink (i.e., high thermal resistance, $R_{heatsink}$). Similarly, the thermal impact is dominated by the local power density, P_{core}, for a chip with a good heat sink. In this case, the transfer of heat from the silicon substrate to the spreader dominates thermal effects. Thus, to minimize chip heatup, it is advantageous to reduce width and global power in the context of a poor heat sink and advantageous to reduce depth and local power density in the context of a more expensive heat sink.

4.2 Hazards of Neglecting Thermal Constraints

Thermal constraints should be considered early in the design process. If a chip is designed without thermal constraints in mind, designers must later cut voltage and clock frequency to meet thermal constraints. The resulting voltage and frequency, and hence performance, will likely be cut more severely than if a thermally-aware configuration were selected from the beginning. Figure 6 demonstrates the slowdown incurred by choosing a non-thermally optimal design with voltage and frequency scaling over the thermally-optimal design. The y-axis plots the thermal-aware optimal performance minus the performance of the configuration without thermal considerations, normalized to the optimal performance. This figure summarizes the slowdown for all combinations of die sizes, heat-sink configurations, application classes, and for both pipeline depth and width optimizations. The average difference for varying depth is around 12-17% and 7-16% for varying width.

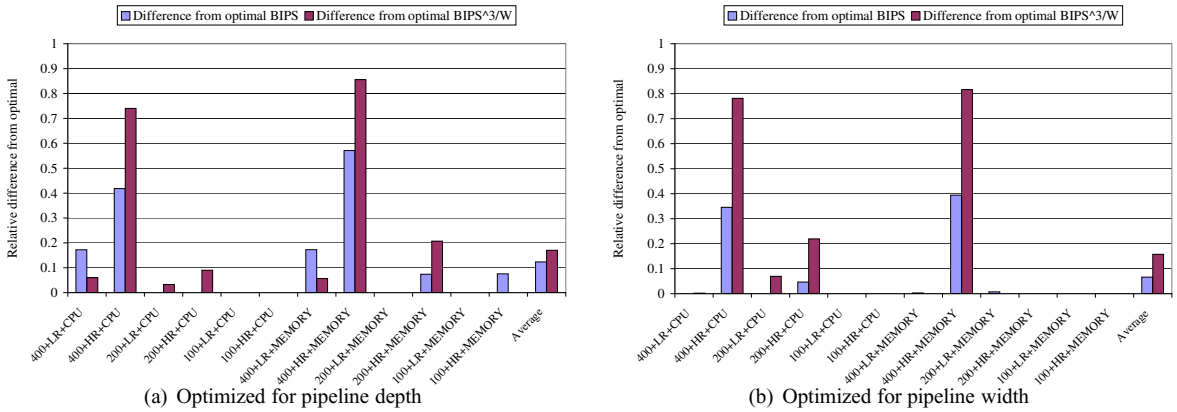
However, we find that for large, $400mm^2$ chips, omitting thermal consideration may result in huge performance degradations. For example, the 400+HR+CPU and 400+HR+MEMORY configurations result in a 40% – 90% difference in performance for BIPS and BIPS^3/W. As area constraints are relaxed, the optimal point tends to include more cores and larger L2 caches. However, if the chip has severe thermal problems, DVFS scaling must scale aggressively to maintain thermal limits, into a region with significant non-linear voltage and frequency scaling, producing large performance losses. For smaller chips with fewer cores and smaller L2 caches, the difference may be negligible because there are very few configurations to choose from. As future CMP server-class microprocessors target $400mm^2$ chips with more than eight cores, it will be essential to perform thermal analysis in the early-stages of the design process when decisions about the number and complexity of cores are being performed.

4.3 DVFS Versus Core Sizing

In meeting thermal constraints for large CMP machines where global heat-up and total chip power is a concern, designers may be forced to choose among implementing fewer cores, smaller L2 caches, or employing aggressive DVFS scaling. We find DVFS superior to removing cores for CPU-bound applications as long as reductions in frequency are met by at least an equal reduction in dynamic and leakage power. Additional cores for CPU-bound applications provide linear increases in performance with near-linear increases in power dissipation. However, because of the strongly non-linear relationship between voltage scaling and clock frequency at low voltages, voltage scaling at some point stops providing super-linear power savings to make up for the performance (clock-frequency) loss. At this point, designers must consider removing cores and L2 cache from the design to meet thermal constraints.

For example, a chip with 30% leakage power no longer achieves super-linear power-performance benefit from DVFS scaling after roughly 0.55x Vdd scaling; frequency of the chip drops to 0.18x and power dissipation also to 0.18x (dominated by leakage power, which only scales linearly with Vdd). Further reductions in Vdd lead to greater performance loss than power savings. (In future process technologies, more than 0.55x Vdd scaling may also approach reliability limits of conventional CMOS circuits.)

Figure 5 shows an example of this behavior with the 2MB/18FO4/4W design. When this design exceeds 14 cores, further increases in core count lead to performance degradation. Vdd scaling has exceeded 0.55x, and the additional DVFS scaling necessary to meet thermal constraints costs more performance than is gained by adding these additional cores. On the other hand, the 2MB/18FO4/2W design only requires Vdd scaling of



(a) Optimized for pipeline depth (b) Optimized for pipeline width

Figure 6. The difference from the optimal when no thermal consideration is made in early design.

0.57x out to 20 cores, which is why this design is attractive even with the additional cores.

Similar analyses hold for memory-bound applications. In this case, the tradeoff is more complex, because the performance benefit from adding cores may be non-linear.

4.4 Accommodating Heterogeneous Workloads

Figures 3–5 also highlight the difficulty of accommodating a range of workload types under area constraints.This is less of a concern when looking at a small number of cores like most prior studies. Prior studies have also neglected the role of pipeline dimensions, which we find to play a major role. And for large numbers of cores, radically different configurations are possible.

CPU-bound and memory-bound workloads have different, incompatible optima. The performance loss from using the CPU-bound optimum with the memory-bound workload and vice-versa is severe, 37–41% and 26–53% respectively, depending on thermal constraints. Even if we try to identify compromise configurations, it is surprising how poorly they perform for one or the other workload. Of course, the best compromise depends on how heavily each workload is weighted. We tried to minimize the performance loss on both workloads.

With no thermal limits, the best configuration is 16 4-wide, 18FO4-deep cores with 8MB of cache, incurring an 18% penalty for the CPU-bound workload. If we turn off 8 cores, it incurs 10% penalty for the memory-bound workload. Moving to 16MB improves memory-bound performance, but hurts CPU-bound performance because it sacrifices 8 cores with an area constraint of 400 mm^2.

With thermal limits, the optimal configurations begin to converge, as the maximum possible number of cores and the L2 cache size is constrained, as the BIPS benefit of extra cores is reduced for CPU-bound benchmarks, and as the benefit of additional cache lines is reduced for memory-bound benchmarks. For low thermal resistance, the best compromise is 18 4-wide cores and 8 MB. This incurs only a 4% performance loss for CPU-bound benchmark and a 10% loss for the memory-bound case. With high thermal resistance, the best compromise is 14 4-wide, 30FO4-deep cores with 8 MB of cache. Turning off 4 cores we reach the optimal configuration for memory-bound case, but this configuration incurs 12% penalty for the CPU-bound case.

Although the discrepancy between the needs of CPU- and memory-bound workloads narrows with increasing thermal constraints, some penalty seems inevitable, because CPU-bound benchmarks prefer more cores while memory-bound benchmarks prefer larger L2 caches. It is interesting to note that we do not see a simple heuristic for identifying good compromise configurations.

5 Conclusions

Our major conclusions include:

- Joint optimization across multiple design variables is necessary. Even pipeline depth, typically fixed in architecture studies, may impact core area and power enough to change the optimal core count. Optimizing without thermal constraints and then scaling to a thermal envelope leads to dramatically inferior designs compared to those obtained from including thermal constraints in the initial optimization.
- Thermal constraints appear to dominate other physical constraints like pin-bandwidth and power delivery. Once thermal constraints are met, at least within the design space we studied, power and throughput have been throttled sufficiently to fall safely within current off-chip I/O bandwidth capabilities and ITRS power-delivery projections.
- Thermal constraints tend to favor shallower pipelines and narrower cores, and tend to reduce the optimal number of cores and L2 cache size. Nevertheless, even under severe thermal constraints, additional cores benefit throughput despite aggressive reductions in operating voltage and frequency. This is true until performance gains from an additional core is negated by the impact of the additional voltage and frequency scaling required of all the cores. This inflection occurs at approximately 55% of the nominal Vdd, well

into the range of non-linear frequency scaling (18% of nominal!).

- For aggressive cooling solutions, reducing power density is at least as important as reducing total power. For low-cost cooling solutions, however, reducing total power is more important because raising power dissipation (even if power density is the same) raises a chip's temperature.

These results raise a range of questions for future work, such as the need for adaptive chip architectures that can dynamically accommodate the full range of workloads, from heavily CPU-bound to heavily memory-bound. Examining how our findings here might change with other workloads (e.g., scientific parallel applications or communication-heavy commercial server workloads) and other architectures (e.g., in-order processors) is future work. Further research on L2/L3/Memory interconnect/ hierarchy for CMP and on the impact of clock propagation on CMP throughput and power is also necessary.

While CMPs may optimize for throughput-oriented application workloads at the expense of single-thread performance, single-thread performance will still be an important consideration for many application domains. Addressing single-thread performance will likely require additional design tradeoffs. This does not necessarily require aggressive superscalar cores running at full voltage and frequency. Future research in this direction must consider speculative multithreading, heterogeneous cores, dynamic core adaptation, run-ahead execution/scouting, and so forth.

Acknowledgments

This work was funded in part by the National Science Foundation under grant nos. CAREER CCR-0133634, CAREER CCF-0448313, CCR-0306404, CCF-0429765, a Faculty Partnership Award from IBM T.J. Watson, a gift from Intel Corp., and an Excellence Award from the Univ. Fund for Excellence in Science and Technology. We would also like to thank Dee A. B. Weikle and the anonymous reviewers for their helpful feedback.

References

[1] D. Brooks, P. Bose, V. Srinivasan, M. Gschwind, P. Emma, and M. Rosenfield. Microarchitecture-level power-performance analysis: the powertimer approach. *IBM J. Research and Development*, 47(5), 2003.

[2] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. In *Proc. of the 27th Int'l Symp. on Computer Architecture*, Jun. 2000.

[3] P. Chaparro, J. Gonzalez, and A. Gonzalez. Thermal-aware clustered microarchitectures. In *Proc. of the Int'l Conf. on Computer Design*, Oct. 2004.

[4] S. Chaudhry, P. Caprioli, S. Yip, and M. Tremblay. High performance throughput computing. *IEEE Micro*, 25(3):32–45, May/June 2005.

[5] M. Ekman and P. Stenstrom. Performance and power impact of issue-width in chip-multiprocessor cores. In *Proc. of the Int'l Conf. on Parallel Processing*, Oct. 2003.

[6] A. El-Moursy, R. Garg, D. Albonesi, and S. Dwarkadas. Partitioning multi-threaded processors with a large number of threads. In *Proc. of the 2005 Int'l Symp. on Performance Analysis of Systems and Software*, Mar. 2005.

[7] G. Hamerly, E. Perelman, J. Lau, and B. Calder. Simpoint 3.0: Faster and more flexible program analysis. In *Proc. of the Wkshp on Modeling, Benchmarking and Simulation*, June 2005.

[8] A. Hartstein and T. R. Puzak. The optimum pipeline depth for a microprocessor. In *Proc. of the 29th Int'l Symp. on Computer Architecture*, pages 7–13, May 2002.

[9] M. S. Hrishikesh et al. The optimal logic depth per pipeline stage is 6 to 8 FO4 inverter delays. In *Proc. of the 29th Int'l Symp. on Computer Architecture*, pages 14–24, May 2002.

[10] J. Huh, D. Burger, and S. W. Keckler. Exploring the design space of future CMPs. In *Proc. of the Int'l Conf. on Parallel Architectures and Compilation Techniques*, Sep. 2001.

[11] T. Karkhanis and J. E. Smith. A first-order superscalar processor model. In *Proc. of the 31st Int'l Symp. on Computer Architecture*, Jun. 2004.

[12] P. Kongetira, K. Aingaran, and K. Olukotun. Niagara: A 32-way multithreaded sparc processor. *IEEE Micro*, 25(2):21–29, Mar./Apr. 2005.

[13] B. Kumar and E. S. Davidson. Computer system design using a hierarchical approach to performance evaluation. In *Communications of the ACM, 23(9)*, Sep. 1980.

[14] R. Kumar, V. Zyuban, and D. M. Tullsen. Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In *The 32nd Int'l Symp. on Computer Architecture*, June. 2005.

[15] S. R. Kunkel, R. J. Eickemeyer, M. H. Lipasti, T. J. Mullins, B. O.Krafka, H. Rosenberg, S. P. Vander-Wiel, P. L. Vitale, and L. D. Whitley. A performance methodology for commercial servers. In *IBM J. of Research and Development, 44(6)*, 2000.

[16] B. Lee and D. Brooks. Effects of pipeline complexity on SMT/CMP power-performance efficiency. In *Proc. of the Workshop on Complexity Effective Design*, Jun. 2005.

[17] J. Li and J. F. Martinez. Power-performance implications of thread-level parallelism on chip multiprocessors. In *Proc. of the 2005 Int'l Symp. on Performance Analysis of Systems and Software*, pages 124–34, Mar. 2005.

[18] Y. Li, D. Brooks, Z. Hu, and K. Skadron. Performance, energy and thermal considerations for SMT and CMP architectures: Extended discussion and results. Technical Report CS-2004-32, Univ. of Virginia Dept. of Computer Science, Oct. 2004.

[19] M. Moudgill, J. Wellman, and J. Moreno. Environment for powerpc microarchitecture exploration. *IEEE Micro*, 19(3), May/Jun. 1999.

[20] K. Ramani, N. Muralimanohar, and R. Balasubramonian. Microarchitectural techniques to reduce interconnect power in clustered processors. In *Proc. of the Workshop on Complexity Effective Design*, Jun. 2004.

[21] K. Skadron, K. Sankaranarayanan, S. Velusamy, D. Tarjan, M. R. Stan, and W. Huang. Temperature-aware microarchitecture: Modeling and implementation. *ACM Trans. on Architecture and Code Optimization*, 1(1), Mar. 2004.

[22] E. Sprangle and D. Carmean. Increasing processor performance by implementing deeper pipelines. In *Proc. of the 29th Int'l Symp. on Computer Architecture*, pages 25–34, May 2002.

[23] M. Tremblay, B. Joy, and K. Shin. A three dimensional register file for superscalar processors. In *Proc. of the 28th Hawaii Int'l Conf. on System Sciences*, 1995.

[24] N. Weste and D. Harris. CMOS VLSI design: A circuit and systems perspective. Addison-Wesley, 2005.

[25] V. Zyuban. *Inherently lower-power high-performance superscalar architectures*. PhD thesis, Univ. of Notre Dame, Mar. 2000.

[26] V. Zyuban, D. Brooks, V. Srinivasan, M. Gschwind, P. Bose, P. Strenski, and P. Emma. Integrated analysis of power and performance for pipelined microprocessors. *IEEE Transactions on Computers*, 53(8), Aug. 2004.