Effects of Pipeline Complexity on SMT/CMP Power-Performance Efficiency

Benjamin Lee and David Brooks
Harvard University
Division of Engineering and Applied Sciences
Cambridge, Massachusetts, USA
{bclee, dbrooks}@eecs.harvard.edu

Abstract

We consider processor core complexity and its implications for the power-performance efficiency of SMT and CMP architectures, exploring fundamental trade-offs between the efficiency of multi-core architectures and the complexity of their cores from a power-performance perspective. Taking pipeline depth and width as proxies for core complexity, we conduct power-performance simulations of several SMT and CMP architectures employing cores of varying complexity. Our analyses identify efficient pipeline dimensions and outline the implications of using a power-performance efficiency metric for core complexity.

Collectively, our results suggest SMT architectures enable efficient increases in pipeline dimensions and core complexity. Furthermore, reducing pipeline dimensions in CMP cores is inefficient, assuming ideal power-performance scaling from voltage/frequency scaling and circuit re-tuning. Given these conclusions, we formulate guidelines for complexity effective design.

1. Introduction

We present an analysis of processor core complexity, quantified by pipeline depth and width, and its effects on the power-performance efficiency of simultaneous multi-threading (SMT) and chip multi-processing (CMP). We define efficiency in terms of $BIPS^3/W$, a voltage invariant power-performance metric that captures the cubic relationship between power and performance. Research in efficient computer architectures has been motivated by significant increases in power dissipation on high-performance systems. Increasing power dissipation also complicates thermal management and current/voltage stability in a system. SMT and CMP architectures are of particular interest as the microprocessor industry moves towards such systems to meet performance targets in mainstream computing [1, 2, 3].

SMT architectures amortize the cost of microarchitectural structures over a greater number of instructions per cycle drawn from multiple threads. Similarly, CMP architectures increase thread-level parallelism by constructing multiple processor cores on a single die. Prior work has examined the power-performance efficiency of such architectures for a particular core design [4, 5] or a particular class of applications [6]. In contrast, we examine SMT/CMP efficiency as a function of core complexity by taking pipeline depth and width as a proxy for complexity.

We specify pipeline depth by the number of FO4 inverter delays per pipeline stage[1] and pipeline width by the maximum number of instructions decoded per cycle. Identifying optimal pipeline parameters for SMT and CMP cores effectively optimizes the balance between instruction-level and thread-level parallelism to maximize efficiency.

Conducting simulations of several SMT and CMP system configurations with varying pipeline parameters (Section 2), we identify power-performance efficient pipeline dimensions and outline the implications of using an efficiency metric for core complexity (Section 3, Section 4). Overall, we draw the following conclusions:

1. SMT architectures enable power-performance efficient increases in pipeline dimensions toward deeper, wider pipelines (Section 3.3, Section 4.3).
2. Reducing pipeline dimensions in CMP architectures is power-performance inefficient (Section 3.4, Section 4.4) relative to alternatives from hardware tuning.

Collectively, these results support the conventional wisdom that SMT architectures are more effective with deeper, wider pipelines and refute the belief that CMP core complexity should decrease relative to uni-processor complexity. We employ these conclusions to formulate guidelines for complexity effective SMT and CMP design (Section 5).

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 latch overhead per pipeline stage is measured in terms of FO4 delay, deeper pipelines have smaller FO4 delays.

2. Experimental Methodology

2.1. Performance Modeling

We employ Turandot, a cycle-based microprocessor performance simulator [7, 8], to obtain data for varying pipeline designs in out-of-order superscalar processors. We evaluate the performance of a design in terms of its achieved instruction throughput measured in billions of instructions per second (BIPS). We compute throughput for n threads with Equation (1), where $Inst_i$ is the number of instructions committed by thread i, $max(Cy_i)$ is the number of cycles required to complete the execution of all threads, and f is the clock frequency. We also evaluate performance in terms of effective delay or inverse throughput ($BIPS^{-1}$).

$$BIPS = \frac{\sum_{i=1}^{n} Inst_i}{max(Cy_i)} \times \frac{f}{10^6} \quad (1)$$

2.2. Power Modeling

We employ PowerTimer, a Turandot based microarchitectural simulator with power modeling extensions, to examine the power implications of varying pipeline designs [9, 10]. The PowerTimer energy models are based on circuit-level power analyses performed on microarchitectural structures in a modern, high-performance PowerPC [11]. Each structure or subunit is comprised of multiple individually analyzed macros. The circuit-level power analyses determine each structure's power dissipation as the sum of hold power and switching power where switching power is a function of the structure's input switching factor. The unconstrained hold and switching power for each macro are combined to generate linear equations for each macro's power. A linear combination of power equations for the macros within a particular subunit produces the subunit's unconstrained power model (Figure 1). PowerTimer uses microarchitectural activity information from Turandot to scale down each subunit's unconstrained hold and switching power under a variety of clock gating assumptions on a per-cycle basis to get the final estimate of power.

Power dissipated by a processor consists of dynamic and leakage components, $P = P_{dynamic} + P_{leakage}$. The dynamic component may be expressed as

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

where α is the true switching factor required for logic functionality, β is the glitching factor that accounts for spurious transitions resulting from differing delays between logic paths. Thus, $(\alpha + \beta)$ is the average number of actual transitions. The clock gating factor, CGF, is the fraction of cycles during which microarchitectural structures are not clock gated. The remaining variables are the effective switching capacitance (C), the supply voltage (V), and the clock frequency (f).

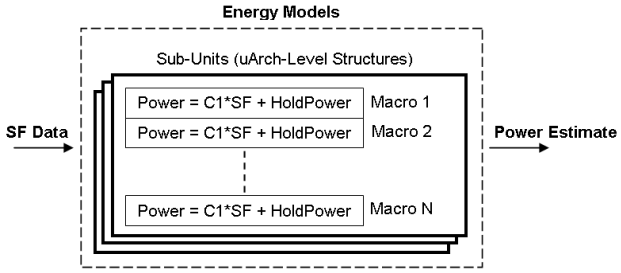

Figure 1. PowerTimer Energy Models.

2.3. Power-Performance Efficiency

We use $\frac{BIPS^3}{W}$ as our power-performance efficiency metric for comparing different pipeline designs. This metric is derived from the cubic relationship between power dissipation and the supply voltage, given a fixed logic/circuit design. Since $P_{dynamic} \approx CV^2 f\alpha \times CGF$ and V is roughly proportional to f, $P_{dynamic} \approx CV^3\alpha \times CGF$. Thus, power is roughly proportional to V^3. This suggests $P \times D^3$, where P is power and D is delay, is an appropriate voltage invariant power-performance metric for server-class microprocessors [9]. Lastly, note that $\frac{BIPS^3}{W} \equiv (P \times D^3)^{-1}$.

2.4. Microarchitectural Modeling

The baseline configuration is a single-threaded pipeline with 19 FO4 delays per stage capable of decoding four non-branch instructions per cycle (Table 1). We also analyze several SMT and CMP architectures (Table 2) with extended versions of Turandot and PowerTimer[5]. The SMT cores are modeled by increasing the sizes of shared and critical resources (*e.g.* register file and issue queues) by 50 percent and implementing round-robin thread scheduling. CMP architectures are simulated as separate cores, tracking conflicts in the shared L2 cache and cache bus. Inter-thread synchronization is not currently supported.

We refer to a particular design point as [*arch*,*depth*,*width*], where *arch* is an architecture from Table 2 and *depth*, *width* refer to the pipeline dimensions. Depth is quantified in terms of FO4 delays per pipeline stage and width is quantified in terms of the number of non-branch instructions decoded per cycle. For example, the baseline is denoted as [ST,19,4].

Processor Core	
Decode Rate	4 non-branch instructions per cycle
Dispatch Rate	9 instructions per cycle
Reservation Stations	FXU(40),FPU(10),LSU(36),BR(12)
Functional Units	2 FXU, 2 FPU, 2 LSU, 2 BR
Physical Registers	80 GPR, 72 FPR
Branch Predictor	16k 1-bit entry BHT
Memory Hierarchy	
L1 DCache Size	32KB, 2-way, 128B blocks, 1-cy latency
L1 ICache Size	32KB, 1-way, 128B blocks, 1-cy latency
L2 Cache Size	2MB, 4-way, 128B blocks, 9-cy latency
Memory	77-cy latency
Pipeline Dimensions	
Pipeline Depth	19 FO4 delays per stage
Pipeline Width	4-decode

Table 1. Baseline Microarchitectural Parameters.

Architecture	Description
ST	Single-threaded baseline (Table 1)
SMT1c1t	SMT-expanded core running 1 thread
SMT1c2t	SMT-expanded core running 2 threads
CMP2c1t	CMP with 2 SMT-expanded cores, each running 1 thread
CMP2c2t	CMP with 2 SMT-expanded cores, each running 2 threads

Table 2. SMT,CMP Architectures.

2.5. Benchmarks

We report experimental results from a suite of 21 traces from the SPEC2000 benchmarks in Table 3. The traces were generated with the tracing facility *Aria* [8] using the full reference input set. However, sampling reduced the total trace length to 100 million instructions per benchmark program. The sampled traces were validated against the full traces before finalizing the traces [12].

We present power and performance data as an average of the SPEC2000 benchmarks. The single-threaded core configurations (*i.e.*, ST and SMT1c1t) were run with every benchmark in the suite. The multi-threaded core configurations (*i.e.*, SMT1c2t and CMP*c*t) were run with multiple copies of the same benchmark for every benchmark in the suite. For example, we simulated CMP2c2t running four copies of *ammp*. Identifying an interesting subset of heterogeneous benchmarks for parallel execution is future work.

3. Pipeline Depth Analysis

3.1. Depth: Performance Scaling

Models for architectures with varying pipeline depths are derived from the reference 19 FO4 design by treating the total number of logic levels as constant independent of the number of pipeline stages. This is an abstraction for the purpose of our analyses; increasing the pipeline depth could require logic design changes. The baseline latencies (Table 4) are scaled to account for pipeline changes according to

$$Latency_{target} = \left\lfloor Latency_{base} \times \frac{FO4_{base}}{FO4_{target}} + 0.5 \right\rfloor \quad (3)$$

SPEC2000 Benchmarks			
ammp	applu	apsi	art
bzip2	crafty	equake	facerec
gap	gcc	gzip	lucas
mcf	mesa	mgrid	perl
sixtrack	swim	twolf	vpr
wupwise			

Table 3. SPEC2000 Benchmark Suite.

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 4. [ST,19,4] Latencies (cy).

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 [14].

3.2. Depth: Power Scaling

Each factor in Equation (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 take leakage power as 30 percent of dynamic power dissipation. We refer the reader to prior work for a detailed treatment of these scaling models.

3.3. Depth: Power-Performance Evaluation

Figure 2 presents the power and performance trends as the pipeline depth increases for [SMT1c1t,*,4] as an average of the 21 benchmarks in our suite (Table 3).[2] Note that pipeline depth is quantified in FO4 delays per stage and smaller FO4 delays are equivalent to deeper pipelines.

The performance maximizing pipeline depth is the 10 FO4 design point, where performance is measured in BIPS.

2 [ST,*,4] and [SMT1c1t,*,4] have similar trends despite differences in resource sizing. The CMP architectures, [CMP2c*t,*,4], effectively double the power and throughput of their SMT counterparts and are not evident in relative trends.

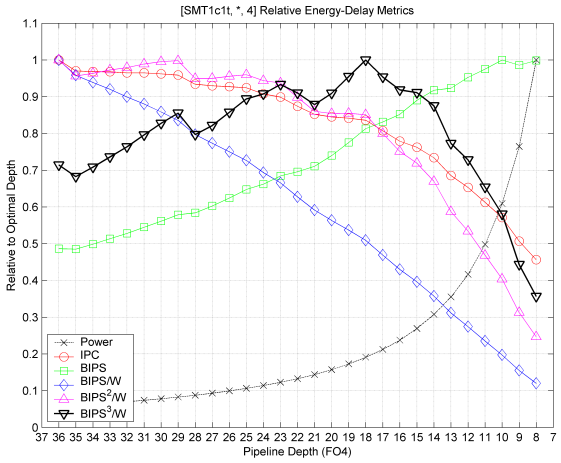

Figure 2. [SMT1c1t,*,4] Power-Performance Metrics.

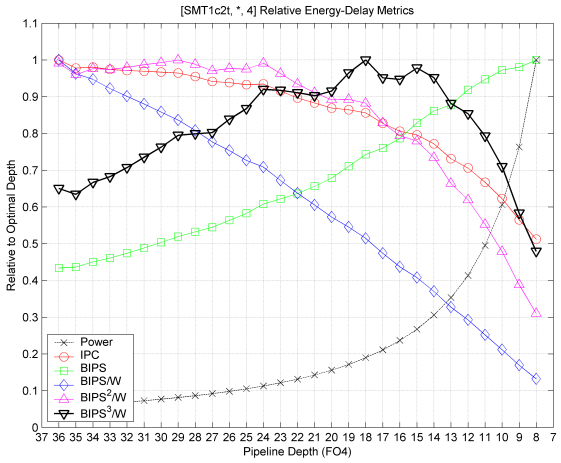

Figure 3. [SMT1c2t,*,4] Power-Performance Metrics.

Since power increases superlinearly and performance increases sublinearly with pipeline depth, the performance to power ratio $\frac{BIPS}{W}$ decreases with depth. In contrast, the 18 FO4 design point is power-performance efficient, maximizing the $\frac{BIPS^3}{W}$ metric. Note $\frac{BIPS^3}{W}$ decreases significantly as the number of pipeline stages increases beyond this design point. In particular, the performance maximizing 10 FO4 design point is 40 percent less efficient than the power-performance optimal 18 FO4 design point.

Figure 3 performs a similar power-performance analysis for the same architecture simultaneously executing two threads, [SMT1c2t,*,4]. Relative to the [SMT1c1t,*,4] results from Figure 2, we find executing a second thread mitigates the efficiency penalties associated with deeper pipelines. Although SMT1c1t and SMT1c2t share the same optimal pipeline depth of 18 FO4 delays per stage, deeper SMT1c2t pipelines up to the 14 FO4 design point achieve $\frac{BIPS^3}{W}$ within 5 percent of the optimal. In contrast, increasing SMT1c1t pipeline depth to 14 FO4 delays per stage reduces $\frac{BIPS^3}{W}$ by more than 10 percent. If performance were the only objective and the 8 FO4 design point were chosen, the 52 percent loss in efficiency for SMT1c2t is significantly smaller than the 64 percent loss for SMT1c1t.

Overall, these results are consistent with the conventional wisdom that SMT architectures perform better in deeper pipelines. The larger number of pipeline stages allows interleaving multiple simultaneous threads at a finer granularity than those afforded by shallower pipelines. Although SMT enables power-performance efficient increases in pipeline depth, voltage/frequency scaling and circuit retuning may provide many of the same benefits for significantly less engineering effort.

3.4. Depth: Complexity Implications

Figure 4 depicts *microarchitectural tuning curves* for each architecture[14]. The average effective delay (or inverse throughput[3]) of the traces in our benchmark suite is plotted on the x-axis in units relative to the baseline [ST,19,4]. The average power dissipation is similarly plotted on the y-axis. The scatter plots indicate the location of each design point in this power-performance space, capturing trends in power-performance trade-offs at each point for a fixed supply voltage. The deeper pipelines are positioned in the high-power, low-delay region of the tuning space.

The downward sloping plots are *hardware tuning curves* representing the power-performance trade-offs achievable from varying the supply voltage and tuning the circuits to meet the frequency target. These curves are an abstraction of the techniques employed and design decisions made at various stages in the microprocessor design (*e.g.* voltage/frequency scaling, transistor sizing, and circuit styles). As the delay budget tightens, greater gate-level parallelism and transistor sizes are needed and more power is dissipated. Hardware intensity, η, quantifies these power and performance trade-offs by specifying the percentage change in energy required to achieve a 1 percent improvement in the critical path delay by restructing the logic and re-tuning the circuits for a given power supply [13]. Mathematically, $\eta = -\frac{\%E}{\%D}$. Assuming a typical hardware intensity of two, these curves represent a 1 percent reduction in effective delay by changing the power supply and tuning the hardware, at a cost of 3 percent in power. This is consistent with the cubic relationship between power and performance.

The efficient 18 FO4 design point from Section 3.3 is also the point on the microarchitectural tuning curve where a 1 percent performance gain from increasing the pipeline

3 Zyuban et al. originally tracked delay on this axis, but inverse throughput ($BIPS^{-1}$) is a more accurate description of this metric when considering multiple thread contexts.

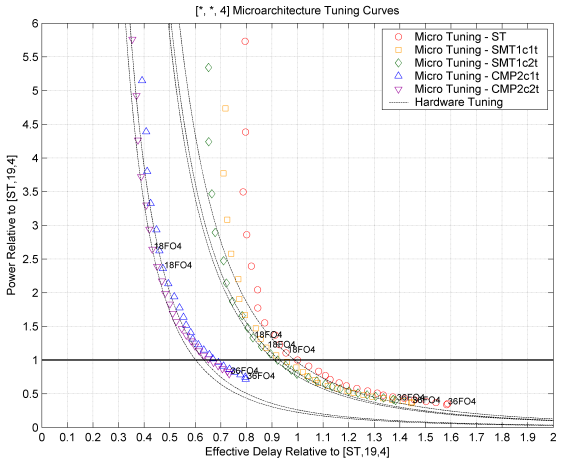

Figure 4. [*,*,4] Micro Tuning Curves.

depth requires a 3 percent increase in power dissipation. Graphically, this implies the point of tangency between the microarchitectural and hardware tuning curves is the efficient design point that delivers a given level of performance while minimizing power.

Let the power budget be the relative power dissipation of the original design [ST,19,4]. Alternative designs that meet this power budget include [SMT1c1t,20,4], [SMT1c2t,21,4], [CMP2c1t,30,4] and [CMP2c2t,31,4]. Note that each of these designs incorporate fewer pipeline stages while reducing effective delay. In particular, choosing [CMP2c2t,31,4] over [ST,19,4] reduces the number of pipeline stages by approximately two-thirds and decreases effective delay by approximately 35 percent.

[CMP2c2t,31,4] is inefficient because it is possible to achieve a greater reduction in delay for the same power dissipated, as illustrated in Figure 4. First, increase the pipeline depth to the 18 FO4 design point, the power-performance optimal for this architecture. This corresponds to moving up the CMP2c2t tuning curve from [CMP2c2t,31,4] to [CMP2c2t,18,4]. Second, tune the hardware until the power budget is met. This corresponds to moving down the hardware tuning curve until relative power dissipation is one. At this new, efficient design point, the power budget is met and effective delay is reduced by approximately 40 percent relative to [ST,19,4] and 8 percent relative to [CMP2c2t,31,4]. Thus, reducing core complexity in multi-core architectures are not power-performance efficient since hardware tuning more complex cores yields greater performance for the same power budget.

	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 5. [ST,19,*] 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 6. Energy Growth Parameters.

4. Pipeline Width Analysis

4.1. Width: Performance Scaling

Performance data for architectures with varying pipeline widths are obtained from the reference 4-decode (4D) design 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 5). 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 (*i.e.* 2D and 1D).

4.2. Width: Power Scaling

A relatively optimistic power scaling technique assumes 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. We expect linear power scaling to effectively estimate changes in power dissipation for functional units since we employ a clustered architecture. Superlinear power scaling effectively reduces to an approximate linear scaling for clustered structures[15, 16, 17]. Furthermore, cache port scaling by replicating a 1-read, 1-write port cache to obtain a 2-read, 1-write port cache for the Power-4 architecture, modeled by [ST,19,4], also suggests linear power scaling is applicable for this microarchitectural structure [18, 19].

In certain cases, however, linear power scaling is an optimistic first-order approximation and, for example, does not capture non-linear relationships between power and the number of register file access ports since it does not ac-

count for the additional circuitry required in a multi-ported SRAM cell. For this reason, we formulate a relatively pessimistic estimate of power dissipation trends as pipeline width varies by applying superlinear power scaling with exponents (Table 6) drawn from Zyuban's work in estimating energy growth parameters [15]. These parameters form a pessimistic power estimate since the values are experimentally derived from non-clustered architectures and tend to overestimate energy growth for clustered architectures.

4.3. Width: Power-Performance Evaluation

As in the pipeline analysis, Figures 5–6 depict microarchitectural tuning curves with linear and superlinear power scaling, respectively. These figures demonstrate the effects of tuning pipeline width for each architecture given a fixed pipeline depth. Note that pipeline width is quantified by the number of non-branch instructions decoded per cycle and faster decode rates correspond to wider pipelines.

Hardware tuning curves are drawn through power-performance efficient design points. Under linear power scaling, a pipeline width of 8D is found to maximize $\frac{BIPS^3}{W}$ for all architectures except the ST baseline; the efficient ST width is 4D. As in the depth analysis, the optimality of these design points can be demonstrated by considering an alternative design point and showing it cannot achieve lower delay for the same power budget.

Comparing the architectures executing one thread per core (*i.e.* ST, SMT1c1t, CMP2c1t) and those executing multiple simultaneous threads per core (*i.e.* SMT1c2t, CMP2c2t) with linear power scaling, we find executing a second thread motivates increasing pipeline width to improve efficiency. In particular, we find the 4D and 8D design points offer comparable efficiency when executing one thread, but choosing the wider 8D pipeline for architectures executing multiple threads can reduce effective delay by approximately 6 to 8 percent after hardware tuning to meet power constraints imposed by their respective original 4D design points. This observation may be drawn from Figure 5 by noting that microarchitectural tuning curves for single-threaded workloads track the hardware tuning curves more closely than their multi-threaded counterparts around the 4D and 8D design points.

As expected, superlinear power scaling decreases the power-performance optimal width to 4D for architectures executing a single thread per core and reduces the benefits of using the wider 8D design for architectures executing multiple threads per core. The hardware tuning curves in Figure 6 track the microarchitectural tuning curves more closely from 4D to 8D compared to those in Figure 5. This implies smaller efficiency penalties for choosing the 4D design over the 8D design.

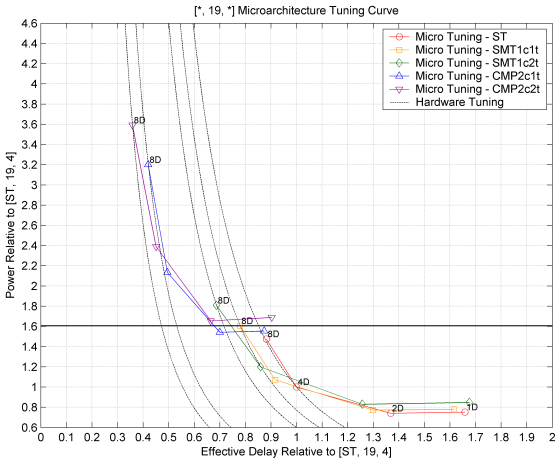

Figure 5. [*,19,*] Micro Tuning Curves - Linear.

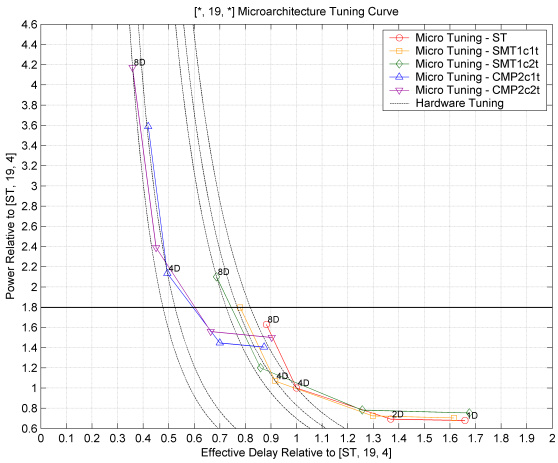

Figure 6. [*,19,*] Micro Tuning Curves - Superlinear.

Overall, these results are consistent with the conventional wisdom that SMT architectures perform better in wider pipelines. Conceptually, the larger number of instructions capable of entering a wider pipeline is better exploited when multiple thread contexts are executing simultaneously.

4.4. Width: Complexity Implications

Since no other design point improves performance while meeting the power constraints imposed by [ST,19,4] with linear power scaling in Figure 5, we instead consider a power budget of 1.7 defined by the relative power dissipation of [SMT1c1t,19,8]. Alternative designs that meet this power budget are [SMT1c2t,19,$4 < w < 8$] and [CMP2c1t,19,$2 < w < 4$].[4] Note each of these designs em-

ploy cores with narrower pipelines while reducing delay.

As in the pipeline analysis, these alternative designs with less complex cores may not be power-performance efficient. [CMP2c1t,19,$2 < w < 4$] is inefficient since it is possible to achieve higher performance for the same power dissipated by increasing the pipeline width to 4D or 8D and tuning the hardware to meet power constraints. This corresponds to moving up the CMP2c1t tuning curve from [CMP2c1t,19,$2 < w < 4$] to [CMP2c1t,19,4] or [CMP2c1t,19,8], then moving down the hardware tuning curve until the relative power dissipation is 1.7. At this new, efficient design point, the power constraints are satisfied and delay is reduced by approximately 33 percent relative to [SMT1c1t,19,8], compared to an approximately 12 percent reduction achieved from [CMP2c1t,19,$2 < w < 4$].

With the relatively pessimistic superlinear power scaling in Figure 6, however, the efficiency differences between the 8D and 4D designs are negligible for both SMT and CMP architectures. In these cases, the 4D design is preferrable since it is likely much less complex than the 8D design point.

Note the analysis thus far has emphasized the relative efficiency of the 8D and 4D design points. In no case, however, does the data support moving to cores narrower than the baseline 4D design. The [CMP2c2t,19,8] design point reduces effective delay by 45 percent over the baseline [ST,19,4] after hardware tuning with linear power scaling. In contrast, the [CMP2c2t,19,2] design point only achieved a 30 percent reduction in delay without hardware tuning. Tuning the hardware for the 2D design would only further reduce performance gains. Thus, these analyses suggest reducing core complexity in multi-core architectures by employing narrower pipelines are not power-performance efficient. As with pipeline depth, hardware tuning more complex cores with wider pipelines to meet power budgets may be more efficient.

5. Complexity Effective Design

The preceding analyses suggest limited opportunities for efficiently reducing core complexity in CMP architectures. In particular, we draw the following conclusions:

1. Employing cores with shallower or narrower pipelines is power-performance inefficient since hardware tuning an efficient design achieves higher performance for the same power dissipation.
2. In the cases where multiple design points are efficient, designers are able to choose a complexity effective design among these efficient alternatives.
3. Given a need to reduce complexity, the efficiency penalties for shallower pipelines are less than those for narrower pipelines.

The analyses from Section 3 and Section 4 find simply reducing pipeline depth or width is inefficient. Superior alternatives, from the perspective of power and performance, employ an efficient core in a multi-core architecture and tune the hardware to meet power constraints. Thus, designers should not rely on trends toward a larger number of less complex cores to meet power or performance targets, because such trends are likely to be inefficient. Overall, this conclusion implies microprocessor core design continues to play a significant role in CMP architectural development.

Multiple efficient design points exist in both depth and width analyses. In particular, designs ranging from 14 to 24 FO4 track the 3 percent, 1 percent power-performance trade-off of the hardware tuning curves, suggesting these designs are approximately equivalent from an efficiency perspective. Similarly, the 4D and 8D designs offer approximately the same efficiency. Choosing less complex designs from these efficient choices, enables designers to manage complexity among efficient alternatives.

The analyses in this paper seeks to maximize performance for a given power budget or minimize power for a given performance target. In the case where complexity constraints must also be met at the expense of power-performance efficiency, decreasing pipeline depth incurs less of an efficiency penalty relative to penalties incurred from decreasing pipeline width. For example, [CMP2c2t,36,4] incurs a 12 percent performance penalty relative to the hardware tuned [CMP2c2t,18,4]. In contrast, [CMP2c2t,19,2] incurs a 25 percent performance penalty relative to the hardware tuned [CMP2c2t,19,4]. Thus, designers seeking to reduce complexity should focus on pipeline depth to minimize the impact on efficiency.

Although we only consider two cores per chip, the complexity, power, and area of interconnect between cores in a CMP architecture becomes increasingly relevant as the number of cores increase [16, 28]. This suggests a fundamental trade-off between core and interconnect complexity. Employing many low complexity cores necessarily implies a more complex interconnect network. Conversely, as core complexity increases, fewer cores per chip are needed to achieve the same performance targets and a less complex interconnect network is required. Thus, complexity is effectively transferred from the interconnect to the CMP cores. Understanding this complexity trade-off is future work.

This work also neglects area effects which may become increasingly significant as the number of cores increase. Reducing core complexity while increasing the number of cores per chip may produce a net increase in throughput per

4 We employ this interval notation due to limited granularity in our design exploration space.

unit area, a metric not considered in this paper. Accounting for these effects is future work.

6. Related Work

The experimental work in this paper combines prior research in optimizing pipeline depths and power-performance analyses for SMT and CMP architectures.

6.1. Optimizing Pipeline Depth

Zyuban, *et al.*, [14] found 18 FO4 delays to be the power-performance optimal pipeline design point for a single-threaded microprocessor. The authors also introduced the microarchitectural tuning curves for graphically analyzing a design's relative position in the power-performance space.

Most prior work in optimizing pipeline depth focused exclusively on improving performance. Kunkel, *et al.*, [20] demonstrated that vector code performance is optimized on deeper pipelines while scalar codes perform better on shallower pipelines. Dubey, *et al.*, [21] developed a more general analytical pipeline model to show that the optimal pipeline depth decreases with increasing overhead from partitioning logic between pipeline stages.

More recent research includes finding optimal pipeline designs from simulation. In particular, Hartstein, *et al.*, [22] performed detailed simulations of a four-way superscalar, out-of-order microprocessor with a memory execute pipeline to identify a 10.7 FO4 performance optimal pipeline design for the SPEC2000 benchmarks. Similarly, Hrishkesh, *et al.*, [23] performed simulations for an Alpha 21264-like machine to identify 8 FO4 as a performance optimal design running the SPEC2000 benchmarks.

6.2. SMT, CMP Power-Performance Analyses

Li, *et al.*, [5] performed a comparative performance, power, and temperature analysis on SMT and CMP architectures. The authors found CMP architectures to be more power-performance efficient for CPU bound benchmarks and SMT architectures to be more efficient for memory bound benchmarks. The latter conclusion follows from the fact that SMT architectures are able to have larger L2 caches given a fixed area budget.

Other related work has examined the power-performance efficiency of SMT architectures. Li, *et al.*, [4] studied the efficiency of a POWER4-like architecture while Seng, *et al.*, [24] studied power optimizations for a multi-threaded Alpha processor. Sasanka, *et al.*, [6] and Kaxiras, *et al.*, [25] compare the relative efficiencies of SMT and CMP architectures for multimedia and signal processing workloads, respectively. Similarly, work by Kumar, *et al.*, [26] with heterogeneous CMP cores demonstrates these architectures produce a net increase in efficiency.

Prior studies have also considered hybrids of SMT and CMP designs (*e.g.* two CMP cores, each supporting two-way SMT), concluding that hybrid organizations with N thread contexts are generally inferior to pure CMP architectures with N full cores [6, 26, 27]. This conclusion is also supported in our work (SMT1c2t versus CMP2c1t).

The complexity, power dissipation, and area of interconnect between cores in a CMP architecture become increasing relevant as the number of cores increase [16, 28]. These considerations do not significantly impact the work presented in this paper since we consider CMP architectures with only two cores.

7. Conclusions and Future Directions

SMT architectures offer opportunities to efficiently increase pipeline dimensions and, consequently, core complexity. In contrast, reducing pipeline dimensions in CMP cores is potentially power-performance inefficient, assuming ideal power-performance scaling from hardware tuning.

We will continue to develop power scaling techniques for pipeline analysis. Preliminary work in dividing microarchitectural structures into primitive building blocks and performing circuit-level power analyses on these blocks may improve existing analytical power models. We expect this hierarchical modeling scheme will enable faster and more accurate characterization of power scaling trends.

We also intend to consider heterogeneous trace pairs for simultaneous execution in our continuing work. Pairing CPU bound traces with memory bound traces might better utilize architectural resources and demonstrate higher power-performance efficiency.

We take power and performance to be the primary metrics in this study, but area and interconnect effects will become significant as we continue our work in CMP architectures for a larger number of cores. Accounting for these other design parameters and metrics is future work.

We intend to integrate the pipeline depth and width analyses for a more comprehensive understanding of the design space. This is a first step towards developing statistical regression models that will enable architectural designers and researchers to interpolate the power-performance effects of varying a pipeline design parameter without a large number of simulations.

References

[1] R. Kalla, B. Sinharoy, J. Tendler. Power5: IBM's Next Generation Power Microprocessor. In *Proc. 15th Hot Chips Symposium*, Aug 2003.

[2] D.T. Mar, F. Bins, D.L. Hill, G. Hinton, D.A. Koufaty, J.A. Miller, M. Upton. Hyper-Threading Technology Architecture and Microarchitecture. *Intel Technology Journal*, 6(1), Feb 2002.

[3] K. Krewell. UltraSPARC IV Mirrors Predecessor: Sun Builds Dual-Core Chip in 130nm. *Microprocessor Report*, Nov 2003.

[4] Y. Li, D. Brooks, Z. Hu, K. Skadron, P. Bose. Understanding the Energy Efficiency of Simultaneous Multithreading. In *Proc. ISLPED*, Aug 2004.

[5] Y. Li, D. Brooks, Z. Hu, K. Skadron. Performance, Energy, and Thermal Considerations for SMT and CMP Architectures. In *Proc. HPCA*, Feb 2005.

[6] R. Sasanka, S.V. Adve, Y.K. Chen, E. Debes. The Energy Efficiency of CMP vs SMT for Multimedia Workloads. In *Proc. ICCD*, Sep 2000.

[7] M. Moudgill, P. Bose, J. Moreno. Validation of Turandot, a Fast Processor Model for Microarchitecture Exploration. In *Proc. IEEE International Performance, Computing, and Communications Conference*, Feb 1999.

[8] M. Moudgill, J. Wellman, J. Moreno. Environment for PowerPC Microarchitecture Exploration. *IEEE Micro*, May/Jun 1999.

[9] D. Brooks, P. Bose, S. Schuster, H. Jacobson, P. Kudva, A. Buyuktosunoglu, J. Wellman, V. Zyuban, M. Gupta, P. Cook. Power-Aware Microarchitecture: Design and Modeling Challenges for Next-Generation Microprocessors. *IEEE Micro*, Nov/Dec 2000.

[10] D. Brooks, P. Bose, V. Srinivasan, M. Gschwind, P. Emma, M. Rosenfield. Microarchitecture-Level Power-Performance Analysis: The PowerTimer Approach. *IBM J. Research and Development*, vol. 47, nos. 5/6, 2003.

[11] J.S. Neely, H.H Chen, S.G. Walker, J. Venuto, T. Bucelot. CPAM: A Common Power Analysis Methodology for High-Performance VLSI Design. In *Proc. Ninth Topical Meeting Electrical Performance of Electronic Packaging*, 2000.

[12] V. Iyengar, L.H. Trevillyan, P. Bose. Representative Traces for Processor Models with Infinite Cache. In *Proc. HPCA-2*, Feb 1996.

[13] V. Zyuban, P. Strenski. Balancing Hardware Intensity in Microprocessor Pipelines. *IBM J. Research and Development*, vol. 47, nos. 5/6, 2003.

[14] V. Zyuban, D. Brooks, V. Srinivasan, M. Gschwind, P. Bose, P. Strenski, P. Emma. Integrated Analysis of Power and Performance for Pipelined Microprocessors. *IEEE Transactions on Computers*, Aug 2004.

[15] V. Zyuban. Inherently Lower-Power High-Perfomrance Superscalar Architectures. Ph.D. Thesis, University of Notre Dame, Mar 2000.

[16] K. Ramani, N. Muralimanohar, R. Balasubramonian. Microarchitectural Techniques to Reduce Interconnect Power in Clustered Processors. *5th Workshop on Complexity-Effective Design (WCED), in conjunction with ISCA-31*, Jun 2004.

[17] P. Chaparro, J. Gonzlez and A. Gonzlez. Thermal-Aware Clustered Microarchitectures. *Proc. of the IEEE International Conference on Computer Design*, Oct 2004.

[18] J.M. Tendler, J.S. Dodson, J.S. Fields,Jr., H. Le, and B. Sinharoy. POWER4 System Microarchitecture. *IBM J. of Research and Development*, vol. 46, no. 1, 2002.

[19] J.D. Warnock, J.M. Keaty, J. Petrovick, J.G. Clabes, C.J. Kircher, B.L. Krauter, P.J. Restle, B.A. Zoric, and C.J. Anderson. newblock The Circuit and Physical Design of the POWER4 Microprocessor. newblock *IBM J. of Research and Development*, vol. 46, no. 1, 2002.

[20] S.R. Kunkel, J.E. Smith. Optimal Pipelining in Supercomputers. In *Proc. ISCA-13*, Jun 1986.

[21] P. Dubey, M. Flynn. Optimal Pipelining. In *J. Parallel and Distributed Computing*, 1990.

[22] A. Hartstein, T.R. Puzak. The Optimum Pipeline Depth for a Microprocessor. In *Proc. ISCA-29*, May 2002.

[23] M.S. Hrishikesh, K. Farkas, N.P. Jouppi, D.C. Burger, S.W. Keckler, P. Sivakumar. The Optimal Logic Depth per Pipeline Stage is 6 to 8 FO4 Inverter Delays. In *Proc. ISCA-29*, May 2002.

[24] J. Seng, D. Tullsen, G. Cai. Power-Sensitive Multi-threaded Architecture. In *Proc. ICCD 2000*, 2000.

[25] S. Kaxiras, G. Narlikar, A.D. Berenbaum, Z. Hu. Comparing Power Consumption of SMT DSPs and CMP DSPS for Mobile Phone Workloads. In *International Conference on Compilers, Architectures, and Synthesis for Embedded Systems*, Nov 2001.

[26] R. Kumar, K.I. Farkas, N.P. Jouppi, P. Ranganathan, D.M. Tullsen. Single-ISA Heterogeneous Multi-core Architectures: The Potential for Processor Power Reduction. In *Proc. ISCA-31*, Jun 2004.

[27] J. Burns, J.L. Gaudiot. Area and System Clock Effects on SMT/CMP Processors. In *Proc. PACT 2001*, Sep 2001.

[28] R. Kumar, V. Zyuban, D. Tullsen. Interconnections in Multi-Core Architectures: Understanding Mechanisms, Overheads and Scaling. In *Proc. ISCA-32*, Jun 2005.