Scaling Power and Performance via Processor Composability

Department of Computer Sciences Technical Report TR-10-14

Madhu Saravana Sibi Behnam Robatmili Hadi Esmaeilzadeh
Bertrand Maher Dong Li Aaron Smith*
Stephen W. Keckler Doug Burger*

Computer Architecture and Technology Laboratory *Microsoft Research
The University of Texas at Austin
cart@cs.utexas.edu

Abstract

Power dissipation trends are leading high-performance processors to a regime in which all chip elements cannot be operated simultaneously at maximum frequency. Consequently, energy efficiency will increase even more in importance, and performance must be achieved within strict power budgets. Current designs employ techniques such as Dynamic Voltage and Frequency Scaling (DVFS) to provide power-performance tradeoffs for both single and multi-threaded workloads. In power-dominated regimes, processors will be run at or near the minimum voltage. Frequency can be reduced to save power, but there is no scaling strategy for increasing performance with high energy efficiency if the processor is being operated at its maximum frequency (and minimum voltage). In this paper, we evaluate the energy efficiency of processor composability–dynamically aggregating small energy-efficient physical cores into larger logical processors–as a method of scaling single-threaded performance up and down, comparing composability to the energy efficiency of voltage and frequency scaling. We measure the power breakdowns of the baseline composable microarchitecture (the TFlex microarchitecture, based on an EDGE ISA) and compare the energy efficiency and performance to one processor designed for power-efficiency (ARM/XScale) and another designed for high-performance (a variant of the Power-4) using normalized power models for as fair a comparison as possible. The study shows that composing multiple dual-issue cores (up to eight) provides performance scaling that is as energy efficient as frequency scaling in a balanced microarchitecture, and is considerably more efficient than scaling the voltage to achieve additional performance once the maximum frequency at the minimum voltage is attained.

1 Introduction

Power dissipation has become the most pressing concern facing microprocessor designers. Current and projected power trends point to a world where all transistors on a chip cannot be operated at the maximum frequency, and worse, they

may not even all be switched on. Consequently, the focus of academic and industrial research has shifted from pure performance to the best performance within a given power budget, or improving performance at some minimum level of energy efficiency.

The slowdown in frequency scaling and diminishing improvements in single-threaded superscalar performance has led to the industry shift to chip multi-processors (CMPs). While CMPs are well suited for multi-threaded workloads they do not directly address single-threaded performance. Additionally, CMP designers need to address the key issue of core granularity: how big should each core be and how many cores should be built on a chip. Larger cores offer better single-thread performance at lower power efficiency, limiting the number of cores that can be placed on a die. Recently, a number of researchers have investigated schemes for dynamically merging or splitting groups of processors to accelerate single threads, providing a large dynamic range of power and performance than single cores alone. *Core Fusion* [10] was designed for x86 processors, while *composability* was applied to small EDGE cores [15]. Both approaches showed how to provide a good range of ILP and TLP on one hardware substrate, but no thorough study has evaluated the power/energy benefits of this concept, nor has voltage and/or frequency scaling been compared as a competing approach to balance energy efficiency and performance.

The EDGE cores used in processor composition, called TFlex cores, were designed to provide a relatively high-performance baseline core as a composition primitive, while keeping the core relatively small and energy efficient (i.e. an efficient, narrow-width out-of-order issue core). Compared to a RISC or CISC architecture, a microarchitecture that implements an EDGE ISA [4] to support composability has some features that may make it potentially more power efficient, but some others that may add energy overhead. Composable EDGE microarchitectures are typically formed from a number of small cores or tiles, with no centralized structures and no global signaling wires [26, 15]. Since such microarchitectures are composed of multiple simpler structures with fewer ports–compared to their superscalar counterparts–they have lower energy to access individually but require more communication among structures. The conventional bypass networks in superscalar processors have been replaced by a lightweight operand network (OPN) in recent EDGE microarchitectures [26, 15], which could add power overheads compared to a superscalar core. Thus there is potential for high power efficiency using the EDGE cores, and a large dynamic range of power/performance using composition, but to date whether this approach can compete with conventional cores using voltage and frequency scaling was unknown.

This paper compares the energy and performance of lightweight EDGE cores to both low-end (ARM/XScale PXA255 [6])

and high-end (4-wide, out-of-order, single-threaded Power4 [13]) processors. Using that comparison as a baseline, the paper then compares the benefits of using composition, along with voltage and frequency scaling, to measure the dynamic range of EDGE cores compared to voltage and frequency scaling on the XScale[1] and Power4 cores. While DVFS has been highly effective in the past, due to the cubic reduction in power with only a linear reduction in performance as voltage is scaled, power constraints are moving processors to a point where they will typically run at V_{min}, scaling to higher voltages only when performance is paramount, and scaling down the frequency at V_{min} when lower power is required. Thus, composition of cores provides another option to use along with DVFS when running at the maximum frequency for a given voltage, if more performance is required.

This study shows that composition, coupled with frequency scaling, can provide a dynamic range of power and performance for single threads that other architectures cannot, scaling from low-performance, high power efficiency to power-efficient high performance. Across a suite of benchmarks including SPEC2000, a single TFlex core runs 60% faster than the ARM/XScale core with 12% worse power efficiency, and can be frequency scaled down to be only slightly slower at the same power. Compared to the Power4, a single (uncomposed) TFlex core runs 2.5 times slower, but consumes 4.2 times less power, showing approximately two-fold energy efficiency improvement. Using that efficiency as a baseline, composing a moderate number of cores (four to eight dual-issue cores) can provide increased performance at roughly the same energy efficiency as strict frequency scaling, thus providing a more efficient solution than increasing the supply voltage. To achieve the best power efficiency while scaling performance, one should first scale frequency while maintaining the voltage at V_{min} until the maximum frequency at that voltage (which we term f_{maxeff})is reached. One should then compose more cores while still at V_{min} and f_{maxeff} until diminishing returns are reached at eight or sixteen cores. As a last resort, voltage and frequency can be scaled together.

2 Background

Composability is a proposed architectural capability that permits dynamic, asymmetric [9] multicore execution to be run on a homogenous substrate, in effect permitting "dynamic heterogeneity." By allowing multiple physical cores to be aggregated or disaggregated, composability enables a wide range of execution modes for many workloads: single-threaded, multi-threaded, and multi-programmed workloads. Each of these workloads has different degrees of inherent parallelism,

[1]We use ARM and XScale interchangeably in this paper.

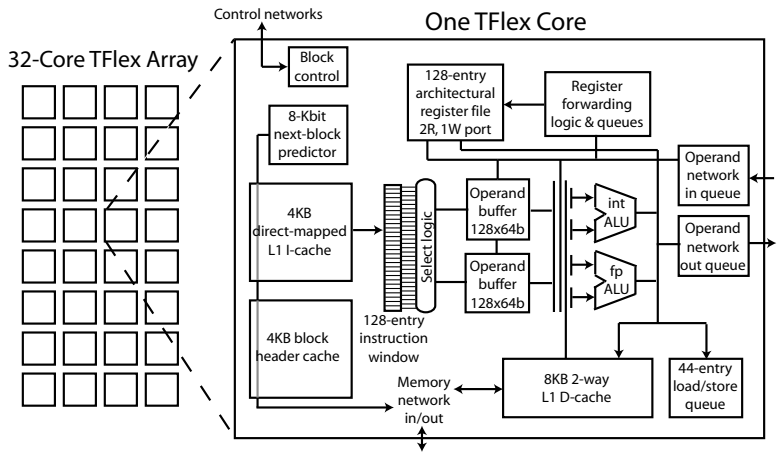


Figure 1: TFlex Microarchitecture

thereby requiring differing amounts of execution resources. To meet the execution requirements of the workload at hand, higher level software policies can effectively leverage the mechanism of composability to optimize for performance, power efficiency and system throughput. Researchers have proposed various flavors of composability, which include methods for composing various in-order cores into a larger core [32, 29] and out-of-order cores into even larger cores [10]. In the "Core Federation" technique [29], two neighboring in-order cores are "federated" to create an out-of-order processor. In "Core Fusion" [10], multiple dual-issue out-of-order cores are fused to create a wide-issue out-of-order core. Using these techniques, workloads with higher inherent parallelism can be executed on a larger logical processor composed of many physical cores. On the other hand, if the workload has low levels of parallelism, fewer cores are allocated to the workload to optimize system power efficiency.

Another example of a composable architecture is supported by Explicit Data Graph Execution (EDGE) architecture [4]. EDGE architectures. In addition to exploiting different types of parallelism [25], EDGE architectures support composable architectures through block-based ISA support [15]. EDGE architectures are characterized by two key features. First, they are block-oriented architectures – they fetch, execute, and commit instructions as a group. Second, within a block the dependence between a producing and a consuming instruction is explicitly encoded in the producing instruction. The direct instruction communication permits energy-efficient out-of-order execution, allowing single cores to serve as a high-performance primitive for composition. The block-atomic execution permits efficient composition by reducing control flow and register renaming.

The TFlex microarchitecture [15] is a fully composable design implementing an EDGE ISA. In this paper we use TFlex as the experimental vehicle to compare power and performance changes using composition versus (and in addition to) DVFS. Figure 1 details the microarchitecture of a single TFlex core. TFlex supports composability by physically

distributing various structures including the register file, instruction window, L1 caches, and operand bypass network. In addition to partitioning, the microarchitecture implements various distributed protocols using micronetworks to implement instruction fetch, execute, commit, speculation recovery, and other bookkeeping [15].

The explicit instruction-to-instruction communication within a block is necessary for scalable composability. In addition, it eliminates or reduces many of the associative structures required for out-of-order execution, including associative instruction wakeup and per-instruction renaming. It also eliminates the need for wide bypass networks used by superscalar processors for operand delivery. Additionally, since all temporary values produced within a block are directly communicated to the consuming instructions, the ISA greatly reduces the number of accesses to the global register file.

However, the block-oriented nature and the direct instruction communication of EDGE ISAs have their own set of overheads compared to traditional superscalar processors. Each EDGE block has a large header and a body [4]. The header contains the register read and write instructions along with other metadata needed for executing that block. The body contains the actual "compute" instructions from the block. The presence of the header block is a clear disadvantage of the EDGE ISA with respect to instruction cache efficiency and capacity. The distributed nature of TFlex necessitates various distributed network protocols for block fetch, commit and operand delivery, which are additional sources of overhead for TFlex. The direct instruction communication causes more overhead when there are many consumers for a given producing instruction. The compiler employs "move" instructions to build a software fan-out tree for delivering values to all consumers. These disadvantages of the EDGE ISA and TFlex implementation add to the energy overhead of the TFlex system. To understand the inherent advantages and disadvantages of EDGE architectures, we perform a detailed performance and power comparison of a sample EDGE architecture with a superscalar processor. The results of this comparison are presented in Section 4. The TFlex microarchitecture supports several policies for mapping instructions to cores [24]. In this work, we use *deep* mapping policy that maps all of the instructions in a block to a single core, but maps successive blocks to different cores in the composed system.

3 Experimental Methodology

3.1 Experimental Platforms

We use an execution-driven, cycle-accurate simulator for all our experiments on the TFlex system [15]. The TFlex simulator supports up to 32 physical cores, which can be composed into different configurations ranging from 1 to 32 cores in powers of 2, with a single thread (in this paper) running on some number of the cores. To assess the relative advantages and disadvantages of EDGE architectures in terms of power efficiency, we use a Power4 like [13] microarchitecture and XScale microarchitecture [6] as comparison platforms. We model the Power4 microarchitecture with a trace-driven simulator called Turandot, which incorporates parameterizable processor power models [21]. We leverage the XTREM performance and power simulator for modeling the XScale microarchitecture [6]. We compare TFlex against these two platforms for the following reasons. First, these platforms represent two categorical extremes in the performance and power spectrum – a high-end system optimized for performance and a low-end system optimized for power, providing a good range for comparisons to composed Tflex processors. Second, the TFlex power models are developed based on validated power models from the TRIPS prototype [8]. Thus, the baseline TFlex power models are based on the 130nm node used for the TRIPS prototype implementation. The original Power4 processor has been implemented at the 180nm technology [13]. Because the 180nm node is close to the 130nm node of TRIPS, we reduce some the scaling uncertainties of the baseline power models of TFlex and Power4 that would be present in comparing power models from disparate process technologies. Third, the Turandot simulator and the power models have been validated against low-level RTL models of the Power4 processor [21]. Similarly, the XTREM power models have been validated against real hardware [6]. Since all platforms have been through some form of validation, our methodology eliminates many uncertainties of unvalidated architectural power models. While it would be interesting to compare TFlex power and performance to other high-performance, power-optimized platforms like Intel Core-2 or AMD Opteron microarchitectures, we avoid such an attempt because comparing real hardware for what platform to a simulator for another injects many uncontrollable unknowns into the experiments. Furthermore, to understand the advantages and disadvantages of TFlex compared to these new platforms, detailed access to fine-grained power and performance measurements is essential and not readily available outside of simulation.

3.2 Power Models

A comparison of disparate platforms such as TFlex, Power4 and XScale is challenging because they all implement different instruction sets, and are based on different process technologies and design methodologies (custom vs. ASIC). To mitigate these challenges and to enable a fair comparison, we tune the baseline Power4 and XTREM power models to reasonably match that of TFlex. As a first step, we ensure the same process technology (65nm), supply voltage (1.0 Volt) and core frequency (1 GHz) in all platforms. Although TFlex and Power4 could operate at higher frequencies at 65nm we choose 1 GHz to be reasonable operating frequency for the XScale platform. The baseline TFlex power models at 130nm and Power4 and XTREM power models at 180nm are suitably scaled down to the 65nm using linear technology scaling. The Turandot simulator accepts the number of fan-out-of-4 (FO4) delays of the design as a parameter to adjust its performance and power models. Assuming the FO4 delay value of 32 picoseconds (which is equal to $0.5 * 65$nm (feature size)) and the desired 1 GHz frequency, we use 30 FO4 delays in the simulator.

TFlex Power Models: The TFlex power models are based on detailed TRIPS power models validated against a hardware prototype [8]. The power model uses CACTI [30] models for all major structures such as instruction and data caches, SRAM arrays, register arrays, branch predictor tables, load-store queue CAMs, and on-chip network router FIFOs to obtain a per-access energy for each structure. These per-access energies combined with access counts from the architectural simulator provides the energy dissipated in these structures. The power models for integer and floating point ALUs are derived from both Wattch [3] and the TRIPS design database. The combinational logic power in various microarchitectural units is modeled based on detailed gate and parasitic capacitances from the TRIPS design databases, and activity factor estimates from the simulator.

Turandot and ARM Power Models: The baseline Power4 models are based on detailed circuit-level simulations of various circuit macros of the processor. The XTREM models use CACTI models (older CACTI version) for various array structures. For a consistent comparison, we replace these in-built power models for all array structures in Turandot and XTREM with CACTI-based power models for the same structures, ensuring that TFlex, XScale and Power4 all utilize CACTI-based models for all array structures. To ensure further consistency, we replace built-in ALU power models of Power4 and XTREM with those of the TFlex model. The combinational logic power in Turandot is modeled similar to TFlex: capacitances come from the Power4 design database, and activity estimates come from the Turandot simulator.

L2 Caches: The three simulators support different L2 cache organizations. While TFlex implements a Non-Uniform

Cache Access (NUCA) L2 cache [14], Turandot and XTREM support a unified, banked L2 cache. In all the experiments of this paper, we ignore the L2 power – both dynamic and static – to focus solely on the efficacy of composability and EDGE architectures. We argue that this is a reasonable approach because the area for the same capacity L2 cache would be the same for each processor; the energy and influence on performance would be similar as well.

Clock Tree: We model global clock drivers, global clock tree interconnect, pre-charge transistors and pipeline latches as part of the TFlex clock tree. The TFlex clock models are based on validated latch count estimates from the TRIPS design and accurate per-latch capacitances from the design library. Since the TRIPS prototype chip does not implement clock gating [22], we add the clock gating support to the TFlex simulator for our experiments. Our clock gating model keeps track of utilization factors in all microarchitectural units. If a unit is active during a clock cycle, our model attributes to the full ungated clock power to that unit. However, if a unit is idle during a cycle our model does not completely clock-gate that unit to be realistic. Our model assigns a fixed percentage (10%) of the ungated clock power to the idle unit. Additionally, all the Operand network (OPN) routers in the TFlex core are not clock gated to allow correct operand communication.

The Turandot clock power models are similarly based on measurements on circuit-level macros [23]. However, the baseline Turandot clock model has two key differences with the TFlex model. First, the Turandot model assumes perfect clock gating – any gateable unit is completely clock gated. We adjust this clock gating style to match the realistic gating style of TFlex (fixed 10% overhead for idle units). Second, the power consumption of higher levels of the clock tree (clock buffers, and splitters) is not modeled in the Turandot model. Assuming the same clock-tree design style across the two systems, and using the clock tree power models from the TFlex design, we add a fixed percentage (20%) overhead to the baseline Turandot clock power. Our analysis of the TFlex clock tree power shows this 20% overhead is uniform across workloads with varying clock-gating patterns. We also add clock gating support to the XTREM simulator. The simulator keeps track of activity in the pipeline latches, and all idle latches and the associated last level of clock buffers are assumed to be clock-gated. The rest of clock tree is always left on.

Leakage Models: We obtain detailed transistor width estimates of each TFlex core from the TRIPS design database. We use the sub-threshold current values predicted by Zhao et al. [31]. For gate-leakage current, we use gate-oxide thickness values from [31] and the work that relates gate-oxide thickness to gate-leakage density [5]. These unit-width leakage current values combined with transistor width estimates provide a simple area-based leakage power models for the TFlex core. The total leakage power of the TFlex system is estimated as the sum of leakage powers of all cores in the composed

Parameter	Configuration
Hand-optimized Benchmarks	2 Kernels, 6 EEMBC benchmarks
Compiled Benchmarks	4 microbenchmarks, 6 kernels, 23 EEMBC benchmarks, 10 SPEC CPU 2000 benchmarks currently supported (6 Integer, 4 FP), simulated with single simpoints of 100 million instructions [28]

Table 1: Benchmarks.

system. We assume that when two TFlex cores are composed into a logical processor, the total power consumption is the sum of the dynamic and the static power dissipated in those two cores; we assume that unused TFlex cores can be completely power- and clock-gated. To normalize the leakage power models across all platforms, we replace the built-in Turandot model (which estimates leakage power as a fixed percentage of dynamic power) and XTREM models with our area-based leakage models after adjusting for the area estimates of the Power4 and the XScale core.

3.3 Benchmarks and DVFS Support

Table 1 lists the various benchmarks used in this study, which fall under two categories. The first category of benchmarks are compiled by the TFlex compiler, and are additionally hand-optimized. These include 2 kernels and 6 EEMBC benchmarks. The second category of benchmarks are purely compiled by the TFlex compiler. These include 6 kernels, 4 microbenchmarks, 23 EEMBC benchmarks, and 10 SPEC benchmarks with single simpoints of 100 million instructions. We include both categories of benchmarks as hand-optimized benchmarks typically exhibit better performance on TFlex than compiled ones. For SPEC benchmarks, we generate the instruction traces corresponding to the same simpoints on an AIX machine, and feed these traces to the Turandot simulator. We were forced to exclude the rest of SPEC suite for one or more of the following reasons: (1) the compiler infrastructure currently fails to generate reliable binaries for four benchmarks (176.gcc, 197.parser, 200.sixtrack and 254.gap); (2) the ARM simulators (functional and performance) fail output checks for the SIMPOINT regions on others (171.swim, 183.equake, 188.ammp and 301.apsi); and (3) the Turandot simulator fails to execute one benchmark (168.wupwise).

In order to compare the performance/power efficacy of composability to that of DVFS, we add the needed support for model DVFS in all the simulators. We limit our experiments to "static" DVFS where the voltage and frequency are set *before* running the application and not "dynamic" DVFS where the settings are changed *while* the application is running. Thus, we do not need to model the reconfiguration costs associated with changing the settings. This approach is fair because we compare static DVFS to static composability, that is composing the cores *before* an application runs. Static composability also does not incur the reconfiguration costs. For modeling power with DVFS, we scale the dynamic power

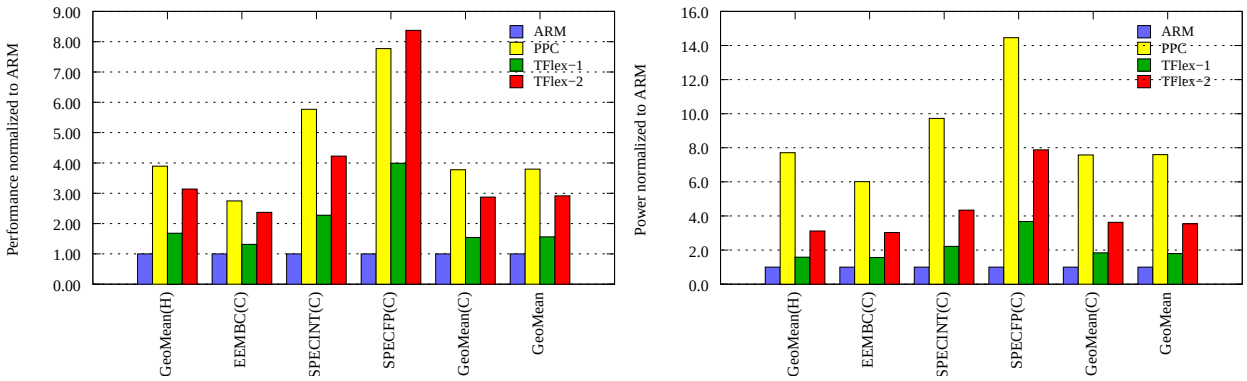
Structures	TFlex-1 (2-issue)		Arm (1-issue, in-order)		Power4 (4-issue)	
	Size	Area(mm^2)	Size	Area(mm^2)	Size	Area(mm^2)
Fetch (B.Pred., I-cache)	Block Predictor in [15] 8KB I-cache	0.73	BTB in [6] 32KB I-Cache	0.69	Predictor in [13] 64KB I-cache	1.72
Reg. Files	128 entries	0.25	16 entries	0.12	80-entry INT 72-entry FP	0.63
Exec. resources (issue window, ALUs)	128-entry SRAM-based issue window 2-INT ALU, 1-FP ALU	1.06	Single Issue 1 INT-ALU, 1 MAC	0.36	82-entry CAM-based issue window (partitioned) 1 combined, INT+MEM Unit 1 FP, 1 Branch 1 Logical	2.86
L1 D-cache (D-cache, LSQ)	8KB D-cache 44-entry LSQ	0.99	32 KB D-cache Write and Fill Buffers	0.62	32KB D-cache Two 36-entry queues	2.99
Routers	Dual OPN [15]	0.27	N/A	0.0	N/A	0.0
L2 Caches	4-MB NUCA Cache	–	4-MB Banked Cache	–	4-MB Banked Cache	–
Total		3.3		1.79		8.2

Table 2: Microarchitecture Comparison

by the applied voltage and frequency and leakage power by the applied voltage. We also adjust the memory latency (in cycles) based on the current processor frequency. We measure performance in these DVFS experiments as the inverse of total time taken (in seconds) by the benchmark.

4 EDGE Energy Efficiency Comparison

In this section we perform a detailed comparison of power and performance of EDGE architectures with ARM and PowerPC processors. This comparison highlights the power advantages and disadvantages of EDGE ISAs relative to conventional architectures. We use 1-core TFlex, the smallest composable unit in the TFlex system, a scaled-down Power4 processor, and the ARM/XScale PXA255 as our experimental baselines. Table 2 compares the microarchitectural features of these three architectures and their respective area estimates when scaled to 65nm. The area estimates for TFlex are obtained from [15], the Power4 area estimates are from the Turandot simulator, and the ARM estimates come from [6]. The original Power4 processor can issue up to eight instructions [13] whereas the Power4 in this study has been scaled down to issue up to four instructions. The scaled-down Power4 avoids the high complexity and the power overhead of the original Power4. The number of ports in the register files, and issue queues in Power4 have been suitably adjusted to match the new issue width of four.



(a) Relative performance

(b) Relative power

Figure 2: Comparison between baseline platforms and two TFlex configurations at the same voltage and frequency.

4.1 Comparison Results

Figure 2(a) shows the geometric mean of performance of Power4, TFlex 1-core and 2-core configurations all normalized to that of XScale ("ARM" bar) for different benchmark types. We show both TFlex 1-core and 2-core configurations here because each is similar in performance to XScale and Power4 respectively. We separate the benchmarks that are hand-optimized for TFlex from the purely compiled ones because hand benchmarks typically exhibit better performance on TFlex than compiled ones. However, for Power4, all these benchmarks are compiled with the XLC compiler on AIX, whose code quality exceeds that of the current TFlex compiler. Power4 out-performs 1-core TFlex by 2.4x on average and has better performance in all benchmark categories due to its larger level-1 caches and higher issue width. Additionally, Power4 supports a fused multiply-add instruction that combines a multiply and add operation into one instruction. On the other hand, dual-issue, out-of-order TFlex 1-core outperforms the single-issue, in-order XScale core by 60%. Power4 outperforms TFlex 2-cores only by 30%, but TFlex 2-cores outperforms XScale by 3x.

Figure 2(b) shows the geometric mean of normalized power for Power, TFlex-1, and TFlex-2 all normalized to XScale. As discussed in Section 3, we only report the power consumed by the respective processor cores in all platforms, and exclude the L2 power to limit the discussion to relative advantages and disadvantages of EDGE. While Power4 performs 2.4 times better than 1-core TFlex, Power4 also consumes 4.2 times more power than 1-core TFlex. TFlex 1-core outperforms XScale by 60% while consuming 80% more power, achieving 12% worse energy efficiency. The key reasons for higher power consumption of Power4 are its larger size and its use of many power-inefficient structures.

Raw power is not a good metric of energy efficiency, especially in high-end microprocessors like the Power4 and TFlex [2]. Since power depends on the processor voltage and frequency, we can reduce power dissipation by suitably

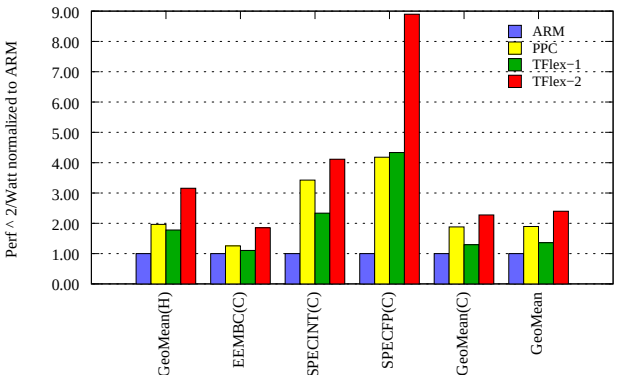


Figure 3: Normalized Inverse Energy-Delay Comparison

adjusting the voltage and frequency. However, reducing voltage and frequency also affects the processor performance, which is not captured by the raw power metric. To simultaneously compare both power and performance in high-end processors we use the voltage-independent metric energy-delay (or its inverse, performance^2/watt) instead of energy-delay^2 because the former is easier to reason about and is more appropriate in a regime in which power supply voltage has little dynamic range. Figure 3 compares performance^2/watt of Power4, TFlex-1 and TFlex-2 normalized to XScale. The y-axis of the graph plots performance^2/watt normalized to ARM and the x-axis shows various types of benchmarks. The higher the values of performance^2/watt for a system, the higher its energy efficiency. Other platforms outperform XScale in terms of energy-delay because of poor performance of the XScale platform. On average, Power4 achieves 40% better energy efficiency in terms of inverse energy-delay compared to TFlex 1-core, a direct result of better performance of Power4 compared to TFlex. However, Tflex is 5% more efficient than Power4 on SPEC-FP and within 15% on EEMBC and hand-coded benchmarks. Additionally, TFlex 2-cores which matches the issue width of Power4 achieves about 30% better energy efficiency than Power4.

4.2 Power Breakdown Analysis

Table 3 presents detailed power statistics of XScale, Power4 and TFlex 1-core configurations for SPEC2000 Integer and FP benchmarks. The table breaks down the total processor power into categories including leakage, clock tree, and different microarchitectural units. The power for each category is reported in milliWatts (mW). All the power categories are common to both TFlex and Power4 except for “Block-Control” and the “OPN-Router”. Block control refers to the power spent in the block fetch and commit protocols in the distributed TFlex microarchitecture, and OPN-Router refers to the power consumed by the operand network router. Additionally, since XScale is an in-order, single-issue microarchitecture, many

Category	SPEC-INT (mW)			SPEC-FP (mW)		
	XScale	Power4	TFlex-1	XScale	Power4	TFlex-1
CLOCK	95.98 (19.10%)	1693.37 (37.69%)	215.72 (20.78%)	95.85 (23.81%)	1802.40 (32.44%)	229.80 (17.37%)
LEAKAGE	36.67 (7.30%)	197.53 (4.40%)	74.02 (7.13%)	36.67 (9.11%)	197.53 (3.56%)	74.02 (5.59%)
CLOCK-TREE	95.98 (19.10%)	282.23 (6.28%)	43.11 (4.15%)	95.85 (23.81%)	300.40 (5.41%)	45.83 (3.46%)
FETCH	110.12 (21.92%)	270.00 (6.01%)	124.18 (11.96%)	76.79 (19.07%)	268.02 (4.82%)	122.18 (9.23%)
DECODE	55.84 (11.12%)	311.59 (6.93%)	25.77 (2.48%)	40.46 (10.05%)	362.43 (6.52%)	28.06 (2.12%)
DCACHE	70.23 (13.98%)	447.19 (9.95%)	66.24 (6.38%)	57.77 (14.35%)	457.21 (8.23%)	70.90 (5.36%)
FP-ALUs	0.00 (0.00%)	90.11 (2.01%)	3.73 (0.36%)	0.00 (0.00%)	1150.96 (20.72%)	311.69 (23.56%)
INT-ALUs	132.61 (26.39%)	1723.23 (38.35%)	514.43 (49.55%)	94.42 (23.45%)	1600.32 (28.81%)	455.19 (34.40%)
ISSUE-Q	0.00 (0.00%)	445.91 (9.92%)	23.44 (2.26%)	0.00 (0.00%)	437.89 (7.88%)	25.01 (1.89%)
REGFILE	0.96 (0.19%)	118.72 (2.64%)	5.33 (0.51%)	0.62 (0.15%)	125.24 (2.25%)	8.19 (0.62%)
REG-RENAMER	0.00 (0.00%)	299.21 (6.66%)	30.76 (2.96%)	0.00 (0.00%)	333.43 (6.00%)	47.28 (3.57%)
LSQ	0.00 (0.00%)	307.69 (6.85%)	51.27 (4.94%)	0.00 (0.00%)	322.18 (5.80%)	58.28 (4.40%)
BLOCK-CONTROL	0.00 (0.00%)	0.00 (0.00%)	17.40 (1.68%)	0.00 (0.00%)	0.00 (0.00%)	18.05 (1.36%)
OPN-ROUTER	0.00 (0.00%)	0.00 (0.00%)	58.53 (5.64%)	0.00 (0.00%)	0.00 (0.00%)	58.53 (4.42%)
TOTAL POWER	502.41	4493.42	1038.20	402.58	5555.60	1323.19

Table 3: Power Breakdown: 1-Core TFlex and Power4

structures including the load-store queues, issue windows and register renamers are absent. Such categories are attributed 0% power for XScale. We split the total clock power into two categories. The first category is the power consumed by the clock tree (clock buffers and splitters), which is shown as the "CLOCK-TREE" category. The second category is the power consumed by the last level of the clock tree – latches – which is subsumed into the respective microarchitectural units. Total clock power is the sum of these categories, and is shown in the table under the name "CLOCK". However, since the number of pipeline latches in XScale is relatively small, their power is also included in the "CLOCK-TREE" category. We split the clock power in this way to compare the raw clock power of both TFlex and Power4 designs, and to compare microarchitectural components including their fraction of clock power.

Clock Tree and Leakage: The clock tree power of Power4 is about 8 times that of TFlex which in turn is 2 times that of XScale. Since Power4 is the largest of the three designs, its leakage power is 2.5 times that of TFlex-1, which is about 2 times that of XScale. The Power4 processor is a larger design than a 1-core TFlex and an XScale with larger caches, multiple register files and CAM-based issue queues, which all require clocking support and consume a larger area compared to their counterparts. The Power4 ALUs are larger than TFlex ALUs because Power4 supports a fused multiply add instruction. XScale, on the other hand, does not have a hardware floating point unit and emulates FP code in software. This contributes to its lower power relative to TFlex and Power4 albeit with reduced performance levels.

As described in Section 3 we perform two normalization steps to the baseline clock models of Power4 – realistic clock gating and clock tree overhead. Realistic clock gating adds a fixed percentage of ungated clock power as overhead to an idle unit. Clock tree overhead adds the power dissipated in the clock tree (clock buffers) to the baseline clock power. These

two normalization steps cause Power4's clock power to be higher than when these steps are not applied.

Register Files: Each TFlex core implements a unified architectural register file containing 128 architectural registers [15]. The TFlex register file in each core has only 1 read and 1 write port. The Power4 baseline, on the other hand, implements separate floating point and integer register files. The FP and the Integer register files have 3 read and 1 write, and 2 read and 1 write ports respectively, which in turn increases the per-access power for the Power4 register files. Furthermore, the EDGE ISA eliminates many register accesses due to direct intra-block instruction communication. For example, on average the Power4 baseline experiences 50% more register file access (combined Integer and FP) compared to TFlex. Due to these factors the register file power in Power4 is almost 10 times that of TFlex. Simple and partitioned register files and direct dependence encoding clearly contribute to lower power of TFlex. On the other hand, the register file of XScale is much smaller compared to that of TFlex, and hence, consumes less power than TFlex.

Register Renamer: The Power4 microarchitecture maps architectural register names to physical registers using integer and FP register renamers. Since the Power4 baseline can dispatch up to 4 integer operation and 4 FP operations per cycle to the issue queues, the individual register renamers should have the ability to map a maximum of 12 integer (4 * 2 Sources + 4 * 1 Destination) and 12 FP registers, which greatly increases the complexity of the renamer. The TFlex microarchitecture implements register renaming between blocks, using the renaming read and the write queues in each TFlex core [15]. On average the register renamers (integer and FP) of Power4 consume almost 8 times the power of read and write queues in TFlex. Eliminating register renaming with many more architectural registers and simplifying register forwarding between blocks is a power advantage for TFlex.

Issue Windows: The Power4 microarchitecture dynamically rediscovers dependencies between producer and consumer instructions in a program using the wake-up/select logic in the issue windows. To do this, the Power4 microarchitecture employs an associative, content-addressable memory (CAMs) in each of its integer and FP issue windows. Every result broadcasts its tag to all instructions in the issue window and an associative search is performed to find the correct dependent instructions. The EDGE ISA of TFlex explicitly encodes the dependence between a producer and a consumer and avoids the need for a hardware associative search at runtime. TFlex has a simple, RAM-based issue window which significantly reduces the energy consumption. The associative issue window of Power4 consumes almost 18 times more power as the TFlex issue window, a key advantage of the explicit encoding feature of the EDGE ISA, which in turn simplifies the TFlex microarchitecture.

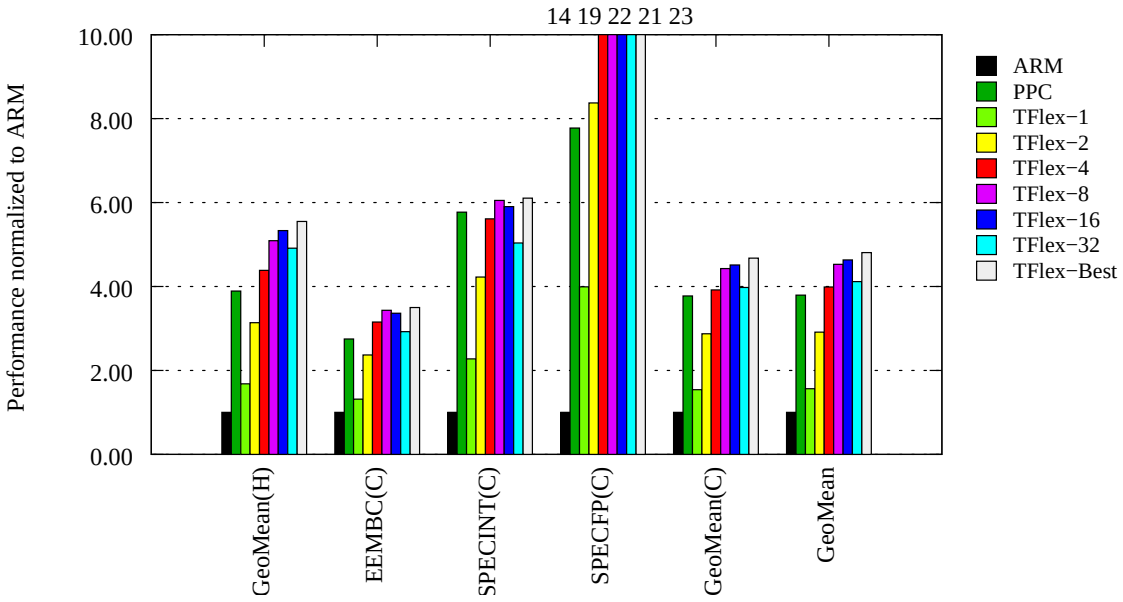


Figure 4: Performance Comparison with Composability

Operand Network and Block Control: The OPN replaces the conventional superscalar bypass networks in the TFlex microarchitecture and transports results of an instruction to all of its consumers. The OPN is a significant overhead for the TFlex microarchitecture both in terms of performance and power. Although the 1-core TFlex configuration does not communicate over the OPN with other cores, our power models do not clock gate the OPN routers, and hence, the routers consume power even in the 1-core configuration. We observe that the OPN router consumes about 6% of the total power in TFlex (the row named "OPN-Router"). We also observe that the power consumed by block fetch and commit protocols are not a huge power overhead for the TFlex design (2% of total power). The analysis shows that these block protocols do not add more overhead even in higher TFlex core configurations.

To summarize, Power4 performs about 2.4 times better than a single TFlex core but at the expense of 4.2 times more power than TFlex. At the other end, TFlex 1-core performs 60% better than an XScale core expending 80% more power than XScale. On a head-to-head power breakdown comparison, TFlex runs with lower power because of partitioned register files, RAM-based issue windows, and direct producer-consumer operand consumption. However, we note that the OPN consumes a reasonable fraction (6%) of the overall power for TFlex and is a disadvantage for the TFlex microarchitecture.

4.3 Composability

In this section we examine the effects of composability on performance, power and inverse energy-delay of the TFlex system, scaling from 1 to 32 cores. In addition, detailed power breakdowns from all TFlex configurations highlight the

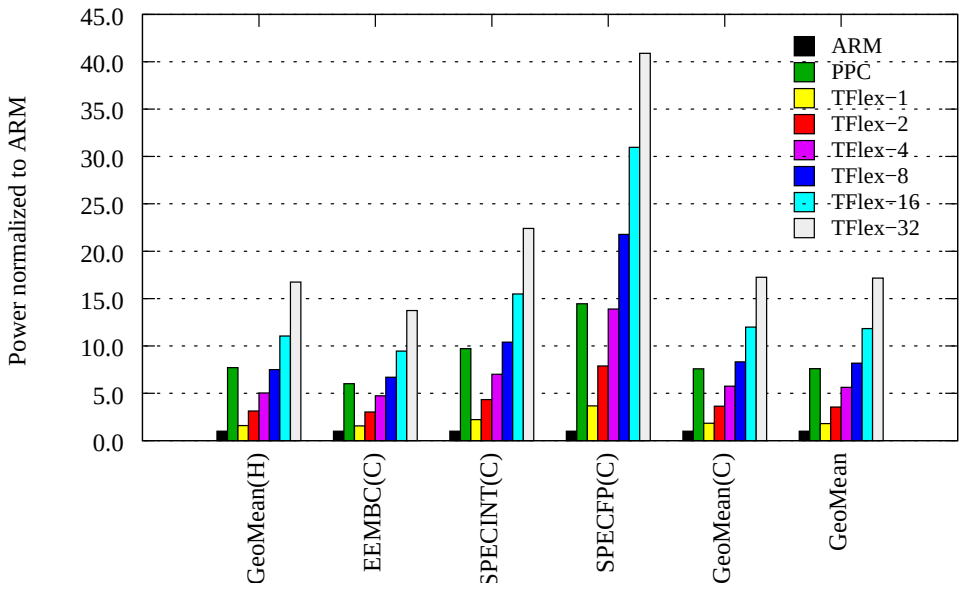


Figure 5: Power Comparison with Composability

Category	Average Power Breakdown (mWatts)					
	1-Core	2-Core	4-Core	8-Core	16-Core	32-Core
CLOCK	222.76 (18.87%)	454.54 (18.80%)	841.32 (20.53%)	1495.86 (23.52%)	2608.56 (28.44%)	4678.57 (35.69%)
LEAKAGE	74.02 (6.27%)	148.04 (6.12%)	296.07 (7.22%)	592.15 (9.31%)	1184.30 (12.91%)	2368.60 (18.07%)
CLOCK-TREE	44.47 (3.77%)	90.68 (3.75%)	168.26 (4.11%)	300.43 (4.72%)	526.82 (5.74%)	949.57 (7.24%)
FETCH	123.18 (10.43%)	271.97 (11.25%)	476.94 (11.64%)	718.75 (11.30%)	924.43 (10.08%)	978.56 (7.46%)
DECODE	26.92 (2.28%)	49.82 (2.06%)	80.83 (1.97%)	119.56 (1.88%)	164.08 (1.79%)	218.15 (1.66%)
DCACHE	68.57 (5.81%)	144.16 (5.96%)	251.88 (6.15%)	407.43 (6.41%)	622.71 (6.79%)	965.77 (7.37%)
FP-ALUs	157.71 (13.36%)	321.58 (13.30%)	534.65 (13.05%)	828.19 (13.02%)	1118.54 (12.19%)	1496.57 (11.42%)
INT-ALUs	484.81 (41.06%)	942.00 (38.97%)	1468.92 (35.85%)	2008.98 (31.58%)	2420.26 (26.38%)	2581.36 (19.69%)
ISSUE-Q	24.22 (2.05%)	49.96 (2.07%)	79.41 (1.94%)	109.59 (1.72%)	132.56 (1.45%)	139.83 (1.07%)
REGFILE	6.76 (0.57%)	14.75 (0.61%)	25.88 (0.63%)	39.46 (0.62%)	53.97 (0.59%)	64.95 (0.50%)
REG-RENAMER	39.02 (3.30%)	85.14 (3.52%)	149.40 (3.65%)	227.78 (3.58%)	311.49 (3.40%)	374.85 (2.86%)
LSQ	54.77 (4.64%)	112.67 (4.66%)	175.23 (4.28%)	241.12 (3.79%)	292.28 (3.19%)	313.54 (2.39%)
BLOCK-CONTROL	17.72 (1.50%)	37.70 (1.56%)	69.08 (1.69%)	117.46 (1.85%)	188.88 (2.06%)	326.44 (2.49%)
OPN-ROUTER	58.53 (4.96%)	148.65 (6.15%)	321.38 (7.84%)	649.87 (10.22%)	1232.58 (13.44%)	2330.74 (17.78%)
TOTAL POWER	1180.70	2417.11	4097.92	6360.76	9172.90	13108.92

Table 4: TFlex Power Breakdown with Composability

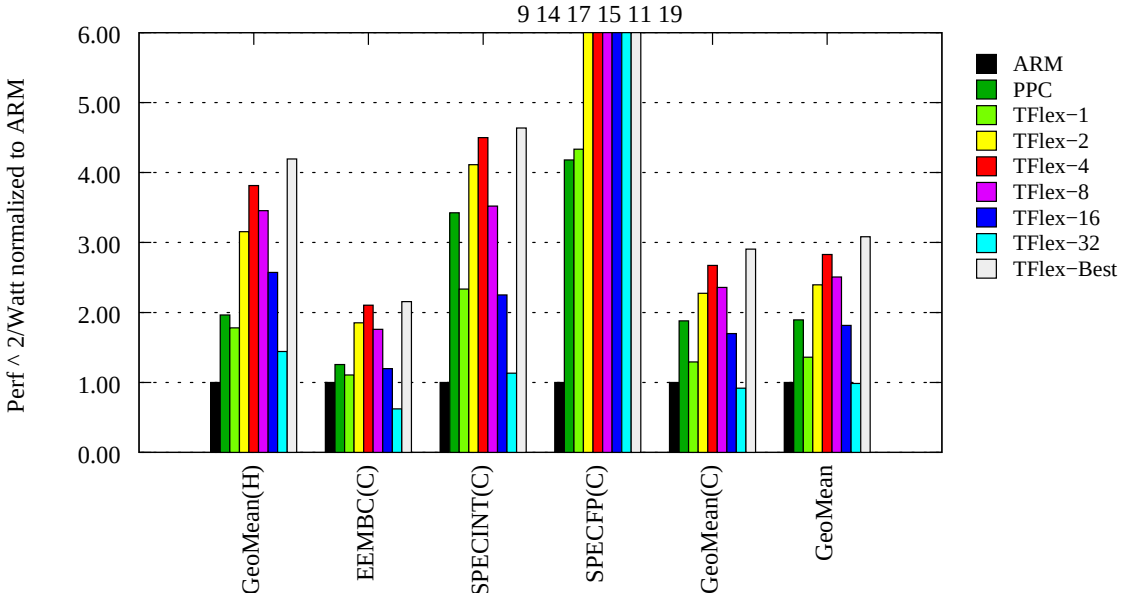
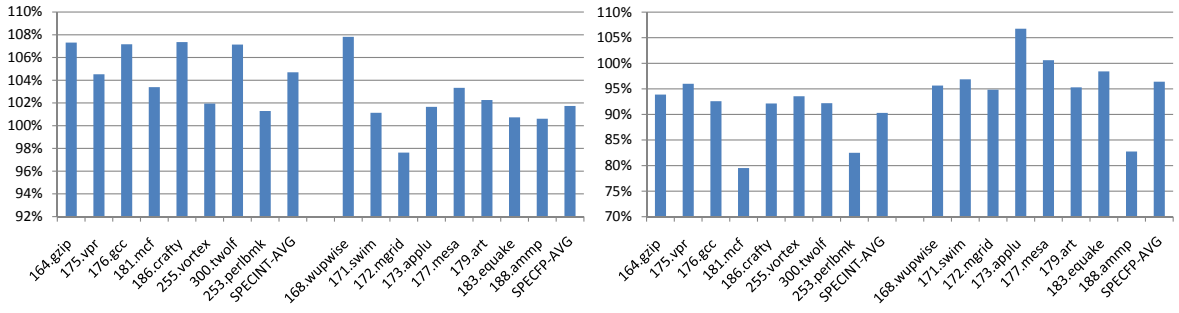


Figure 6: Perf^2/Watt with Composability

power overheads of composability. Figures 4, 5, and 6 compare the performance, power and inverse energy-delay of different TFlex configurations normalized to 1-core TFlex. TFlex 16-core achieves the best average performance, and performance scales up to 8 or 16 cores. Total power increases as we scale up the cores due to increase in static power, higher clock tree power, and distributed communication overheads. Table 4 shows the average power breakdown in milliwatts for all TFlex configurations. The leakage power doubles as we double the number of cores because transistor counts grow with area. With increasing cores clock tree power increases as well, but grows sub-linearly due to fine-grained clock gating in the cores. The largest source of power overhead for TFlex is operand communication over the network. Including the clock power, the OPN router consumes roughly 20% of the total core power at the 32-core configuration.

In terms of inverse energy-delay, the best configuration switches to 4 cores from 16 cores. This shift to a smaller configuration occurs because performance increases at a slower rate than power as the number of cores increases. Because the composable TFlex microarchitecture allows a system to choose the best possible configuration on a per-program basis, we include a “TFlex-Best” measurement in Figures 4 and 6. This measurement shows the performance and inverse energy-delay when using the best configuration for each benchmark. TFlex-Best configuration achieves an average 3x performance improvement and 2.3x inverse energy-delay improvement over 1-core Tflex.



(a) Speedup. (b) Relative Power.

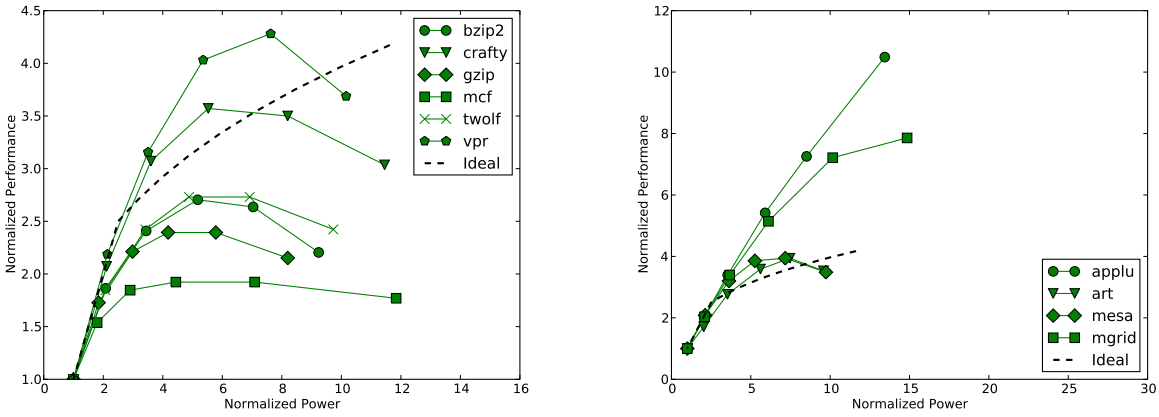
Figure 7: Limited broadcast support to elide move overheads in TFlex Architectures.

4.4 Further EDGE Optimizations

While other architectures, such as ARM and PowerPC have enjoyed numerous design generations that have driven performance and energy-efficiency up, EDGE architectures are still in their nascency and have more than a few opportunities for improvement. One particular overhead in the EDGE architecture implemented in TFlex stems from move instructions required to fan out values that are used by many consumers. Unlike a conventional architecture which can employ the register file to broadcast values, a compiler for an EDGE architecture must build a software fanout tree of move instructions which can result in instruction execution overheads of up to 20% [20].

To target this overhead, we have examined a compiler-assisted hybrid instruction communication mechanism that augments a token-based instruction communication model with a small number of architecturally exposed broadcasts within the instruction window. A narrow CAM allows high-fanout instructions to send their operands to their multiple consumers. All other instructions, which have low-fanout, rely on the point-to-point token communication model. The compiler determines statically which instructions use tokens and which use broadcasts. Few tags are required as the compiler can reuse broadcast identifiers for non-overlapping live range broadcasts.

Figure 7 shows the performance and power of the hybrid broadcast scheme running single-threaded code on a composed 16-core TFlex system relative to a system without support for broadcast. With four overlapping broadcasts per hyperblock, the hybrid scheme achieves about a 5% speedup and 10% reduction in power for SPEC-INT. For SPEC-FP, the speedup and power reductions are mixed, due largely from the fewer number of move fanout trees in these benchmarks. While we do not include hybrid broadcasts in the performance and power results in the rest of this section, we expect that this and other mechanisms can provide a steady improvement for EDGE architectures.



(a) SPEC-INT (b) SPEC-FP

Figure 8: Increasing composable core count from 1 to 32.

5 Performance/Power Trade-Offs: DVFS vs. Composability

Obtaining high performance and energy efficiency simultaneously has become substantially more challenging for two primary reasons. First, threshold voltages for CMOS circuits are reaching the minimum limits meaning that voltage scaling, and the quadratic reduction in power it delivers is coming to an end. Second, processor designers have backed away from ultra-high frequency designs because power grows superlinearly as circuits are successively optimized for higher and higher clock frequency. Thus the maximum frequency (f_{max}) for integrated circuits is likely to scale slowly in future technologies.

To achieve single-thread performance beyond the range of f_{max}, systems need new microarchitectural mechanisms that provide the same or better performance/power trade-offs as frequency scaling. In this section we measure the efficacy of composability as an augmentation to frequency scaling that can extend the power/performance trade-offs beyond today's capabilities. By aggregating or disaggregating cores, composability offers a power-performance tradeoff along a different dimension from frequency scaling. In this section we quantitatively address the following questions: (1) how do power and performance trade off in composable processors, (2) in what regimes are DVFS, frequency scaling, and composability the most beneficial, and (3) what benefits can a system that supports both DVFS and composability bring.

5.1 Composability Results

To examine the relationship between performance and power for the TFlex composable core processor, we varied the core count from 1–32 (while running a single thread that employs all of the provided cores) and measured the improvement in

performance and power relative to a 1-core configuration. Figure 8 plots the results for SPECINT and SPECFP applications respectively. Each graph includes a line labeled "Ideal" which, for a single-core TFlex architecture, represents linear power/performance scaling while at V_{min} (pure frequency scaling regime) and quadratic cubic power/performance scaling (DVFS scaling regime) according to the power states of Table 5.

For most of the SPECINT benchmarks, performance and power increase at equal rates in the range of 1–4 cores. Performance reaches a peak at 8 cores and then drops off at 16 and 32 cores. These applications have insufficient instruction level parallelism to effectively use more than 8 cores; beyond that point, the increasing distribution of the architecture increases communication overheads and causes performance to drop. In this regime, power increases but sublinearly relative to core-count increases because clock gating reduces power consumption in the idle cores. The benchmark mcf performs particularly poorly as it is memory bound and reaches its limit with just a few cores. The SPECFP benchmarks show superior, with most benchmarks able to use 16 cores effectively. In particular applu has ample parallelism and easily uses even 32 cores. Power and performance scale nearly linearly in the regime of 1–16 cores.

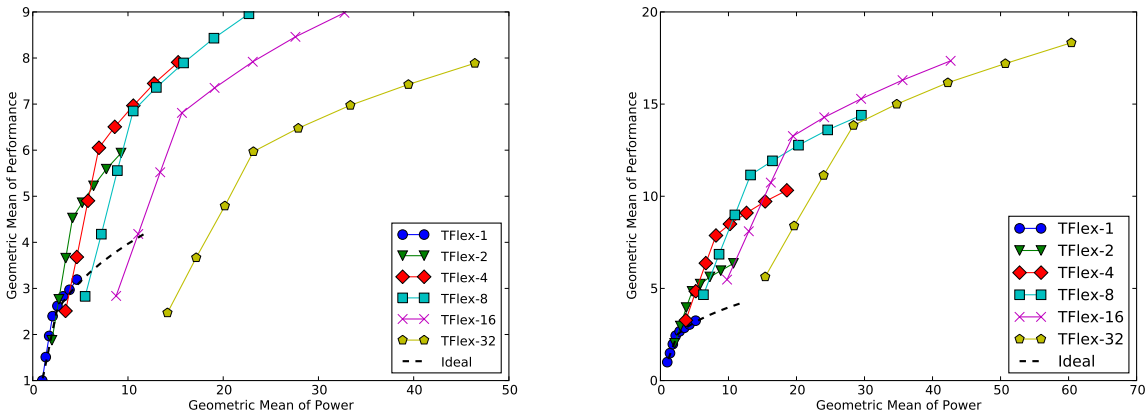
5.2 Composability and DVFS

To model voltage and frequency scaling, we assume a 65nm process and the power states listed in Table 5 for our experiments. The table shows that we assume a moderate f_{max} of 1 GHz. Indices from 1 through 5 represent the DVFS regime, where both the operating voltage and frequency can change. Indices from 5 through 8 represent the frequency scaling regime where only the operating frequency changes and the voltage remains constant at 0.75 volts. We include both voltage/frequency scaling and just frequency scaling regimes to account for limited further reduction in V_{min}. We operate Power4 and XScale models at all the states in Table 5. The TFlex processor is operated at all composable configurations – 1, 2, 4, 8, 16 and 32 cores – at each of the DVFS states, resulting in a total of 48 configurations (6 core configurations x 8 DVFS states). While we recognize that high performance processors such as Power4 and TFlex can scale to higher frequencies, we use this limited set of power states to be able to compare TFlex to both ARM and Power4.

Figure 9 shows normalized performance (Y-axis) and power (X-axis) of various TFlex configurations for the SPEC-INT and SPEC-FP applications. Since performance is plotted on the y-axis and power on the x-axis, configurations that have higher y values and lower x values are more energy efficient than other configurations – such configurations expend less energy and achieve better performance compared to other configurations. Each TFlex curve represents a different core

Index	Vdd (volts)	Frequency (MHz)
1	1.0 (V_{max})	1000 (f_{max})
2	0.937	937
3	0.875	875
4	0.812	812
5	0.75	750 (f_{maxeff})
6	0.75	600
7	0.75	450
8	0.75 (V_{min})	300 (f_{min})

Table 5: DVFS Configurations for Power4 and TFlex



(a) SPEC-INT (b) SPEC-FP

Figure 9: Performance versus power with DVFS and composition.

count (1–32 cores) and the points on each curve cover all the DVFS settings in table 5. For all configurations, power and performance are normalized to a single TFlex core at the (V_{min} and f_{min} configuration, which corresponds to index 8 in table 5).

In the frequency scaling regime, each TFlex configuration achieves approximately a one-for-one improvement in performance when additional power is applied (higher frequencies). To reach f_{max} requires voltage increases, which then demands more incremental power to provide an incremental performance improvement. However, composability provides a better performance/power trade-off compared to voltage scaling, and also enables a scale down in power and performance, providing power efficiency when lower performance is required. For SpecInt and SpecFP, all the composable TFlex configurations combined with frequency and DVFS scaling offer a richer tradeoff space and better opportunity for power efficiency than provided by frequency scaling or DVFS alone. Higher-level software policies attempting to maximize system throughput under a given power budget can effectively exploit this trade-off space offered by composition and DVFS. Composition combined with DVFS clearly offers multiple operating points at a given performance level but a flexible power budget or vice-versa.

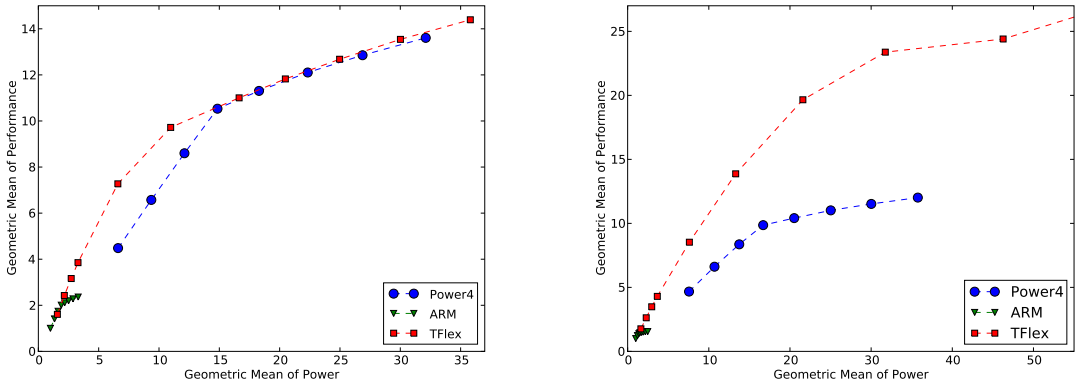
In these graphs, one can consider the points that form the frontier toward the upper left in the graphs as a "Pareto-optimal" curve illustrating the best configurations. Note that each option of applying power to achieve performance (frequency scaling, voltage/frequency scaling, and composition) are represented along this frontier. The graphs indicate that in general the first best choice is scaling frequency only. The second best option that provides the biggest performance benefit for the power cost is composition. This concurs with the observation from Section 5.1 that performance and power trade nearly one-for-one for at least a subset of the composable configurations. Once the tipping point of composability is reached after which adding cores ceases to improve performance, voltage and frequency scaling combined should be applied. Such scaling should be reserved for last as reaching f_{max} through voltage increases is power inefficient.

5.3 Summary

Obtaining higher performance generally requires expending more power. For a conventional microprocessor chip that supports voltage and frequency scaling, improving performance with the best efficiency first comes by scaling the frequency while keeping the processor at its minimal possible voltage (V_{min}). Eventually, scaling will reach the maximum possible frequency attainable at this lowest voltage, which we term f_{maxeff}. If the system requires greater performance, frequency and voltage must increase in concert, with each step being less power efficient than the last. Likewise, when aiming for lower power with the least drop in performance, conventional systems should perform the reverse of these steps: first scale down voltage and frequency to V_{min}, and then scale frequency down alone.

Composability offers an additional step, namely keeping voltage and frequency the same and changing the number of cores. Figure 10 shows the power and performance scaling curves for ARM, PowerPC, and TFlex, normalized to the ARM core at the lowest voltage and frequency we examine (0.75V and 300MHz from Table 5). Both ARM and PowerPC show a linear relationship between power and performance while frequency is scaled up at V_{min}. The TFlex curve shows configurations (voltage, frequency, cores) that produce best measured power and performance. Like ARM and PowerPC, frequency scaling at V_{min} and one core produces the best power and performance until reaching f_{maxeff}. At this point, keeping voltage and frequency constant while scaling the core count up to eight produces the best results. When core scaling reaches diminishing returns, returning to voltage and frequency scaling produces greater performance, albeit at a higher power cost.

The graph also shows the potential for composability as a means of bridging across the spectrum of architectures with



(a) SPEC-INT (b) SPEC-FP

Figure 10: Best measured power and performance scaling.

different power and performance optimization points. Although ARM and PowerPC are designed for very different points in the spectrum, the composabilty of TFlex allows different configurations to operate in either regime.

6 Related Work

DVFS has been extensively studied in the architecture community for performance-power trade-offs. We limit our discussion to DVFS studies that are most relevant to this study. Martinez et. al. [18] argue that using DVFS as the only optimization dimension limits the power-performance optimization options for CMPs. They propose to combine DVFS and different number of cores as two mechanisms for optimizing power and performance. Parallel regions run on different number of cores, while different DVFS settings are applied to all participating cores. Their results show properly choosing the number of cores and chip-wide DVFS settings at run time can lead to significant power savings. To explore this relatively large optimization space, the authors propose several search heuristics producing near optimum results. As TFlex is also a CMP, their work is directly applicable to the TFlex microarchitecture. However, this paper limits the experiments to single-threaded workloads only, since there is so much multi-threaded research occurring. Several researchers have used DVFS to trade performance for power in CMPs and in Multiple Clock-Domain (MCD) processors [11, 12, 19, 27]. Isci et al. [11] assume a fine-grained per-core DVFS mechanism on a multi-core architecture to maximize overall system throughput within a given power budget. In [12], the authors propose multiple-clock domain processors to eliminate global clock skew effects and show a slight reduction in overall power and performance.

Using heterogeneous multicores (also known as asymmetric multicore) systems is another way to improve power or

performance over symmetric multicore systems. Kumar et al. [16] use a single-ISA heterogeneous multicore architecture in which each core implements the Alpha ISA, but belongs to a different generation microarchitecture. Ghiasi and Grunwald [7] show an asymmetric dual-core design will operate under heavy loads at a performance close to that of two symmetric cores. However, using a reconfigurable component in the asymmetric design achieves much lower power consumption compared to the symmetric design. Other studies evaluate the benefits of using asymmetric cores in improving multiprogramming [17] and multithreaded programing [1]. Asymmetric multicores avoid the overheads incurred by composable architectures but are inflexible because the choice of various asymmetric cores has to be made at design time and not at run time as provided by composability.

7 Conclusion

Increasing power dissipation in computing systems has become the critical issue facing architects, now that supply voltage scaling is no longer keeping pace with transistor integration. Future gains will come from increasing performance at the same power, or reducing power at the same level of performance. To achieve single-thread performance beyond the range of the maximum frequency, f_{max}, systems direly need new microarchitectural mechanisms that provide the same or better performance/power trade-offs as frequency scaling.

In this paper, we measure the efficacy of processor composition as an alternative to frequency scaling that can extend the power/performance trade-offs beyond today's capabilities. We first examine the overheads of composable architectures, through a detailed comparison of a 1-core TFlex system with ARM XScale and Power4 processors. Compared to the Power4, the results indicate that composition can permit much more energy-efficient execution for all but the highest part of the performance range. Furthermore, composition permits scaling from extreme power efficiency with a small penalty, to equivalent power and performance on SPECINT, and considerably improved power and performance on SPECFP. Thus, composing cores provides three main benefits: (1) composition enables a single architecture to span a wide range of architectures designed for different points in the power/performance spectrum; (2) composability is more efficient than voltage scaling as a means to achieve higher performance from a given architecture; and (3) composability can extend performance beyond maximum frequency limits.

We expect that additional mechanisms and improvements, such as the broadcast optimization shown herein, better predicate prediction, and higher quality code generation from a production compiler, will push the performance and energy

efficiency for all benchmarks beyond what the compared processors are able to achieve. However, raw performance or efficiency benefits are likely insufficient to justify the expensive adoption of a new instruction set paradigm. It remains to be seen whether the large dynamic range of composable processors will provide sufficient impetus to justify the expense of adoption.

References

[1] M. Annavaram, E. Grochowski, and J. P. Shen. Mitigating amdahl's law through epi throttling. In *Proceedings of the 32nd International Symposium on Computer Architecture*, pages 298–309, June 2005.

[2] D. Brooks, P. Bose, S. Schuster, H. Jacobson, P. Kudva, A. Buyuktosunoglu, J. Wellman, V. Zyuban, M. Gupta, and P. Cook. Power-aware Microarchitecture: Design and Modeling Challenges for Next-Generation Microprocessors. *IEEE Micro*, 20(6):26–44, November/December 2000.

[3] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A Framework for Architectural-level Power Analysis and Optimizations. In *Proceedings of the 27th Annual International Symposium on Computer Architecture*, pages 83–94, May 2000.

[4] D. Burger, S. Keckler, K. McKinley, M. Dahlin, L. John, C. Lin, C. Moore, J. Burrill, R. McDonald, and W. Yoder. Scaling to the End of Silicon with EDGE Architectures. *IEEE Computer*, 37(7):44–55, July 2004.

[5] M. G. Chris Bowen, Gerhard Klimeck and D. Chapman. Dopant fluctuations and quantum effects in sub-0.1um cmos, 1997.

[6] G. Contreras, M. Martonosi, J. Peng, G.-Y. Lueh, and R. Ju. The xtrem power and performance simulator for the intel xscale core: Design and experiences. *ACM Transactions in Embedded Computing Systems*, 6(1):4, 2007.

[7] S. Ghiasi and D. Grunwald. Aide de Camp: Asymmetric Dual Core Design for Power and Energy Reduction. Technical Report CU-CS-964-03, The University of Colorado, Department of Computer Science, 2003.

[8] M. S. S. Govindan, S. W. Keckler, and D. Burger. End-to-end validation of architectural power models. In *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, pages 383–388, 2009.

[9] M. Hill and M. Marty. Amdahl's Law in the Multicore Era. *IEEE Computer*, 41(7):33–38, July 2008.

[10] E. Ipek, M. Kirman, N. Kirman, and J. F. Martínez. Core Fusion: Accommodating Software Diversity in Chip Multiprocessors. In *Proceedings of the 34th International Symposium on Computer Architecture*, pages 186–197, 2007.

[11] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi. An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance For A Given Power Budget. In *Proceedings of the 39th Annual International Symposium on Microarchitecture*, pages 347–358, December 2006.

[12] A. Iyer and D. Marculescu. Power and Performance Evaluation of Globally Asynchronous Locally Synchronous Processors. In *Proceedings of the 29th Annual International Symposium on Computer Architecture*, pages 158–168, May 2002.

[13] J.M.Tendler, J.S.Dodson, J. J.S.Fields, H.Le, and B.Sinharoy. POWER4 System Microarchitecture. *IBM Journal of Research and Development*, 46(1), 2002.

[14] C. Kim, D. Burger, and S. W. Keckler. An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 211–222, October 2002.

[15] C. Kim, S. Sethumadhavan, M. S. Govindan, N. Ranganathan, D. Gulati, D. Burger, and S. W. Keckler. Composable Lightweight Processors. In *Proceedings of the 40th Annual International Symposium on Microarchitecture*, pages 381–394, December 2007.

[16] R. Kumar, K. I. Farkas, N. P. Jouppi, P. Ranganathan, and D. M. Tullsen. Single-ISA Heterogeneous Multi-Core Architectures: The Potential for Processor Power Reduction. In *Proceedings of the 36th International Symposium on Microarchitecture*, pages 81–92, December 2003.

[17] R. Kumar, D. M. Tullsen, P. Ranganathan, N. P. Jouppi, and K. I. Farkas. Single-ISA Heterogeneous Multi-Core Architectures for Multithreaded Workload Performance. In *International Symposium on Computer Architecture*, pages 64–75, June 2004.

[18] J. Li and J. Martinez. Dynamic Power-performance Adaptation of Parallel Computation on Chip Multiprocessors. *Proceedings of the 13th Annual International Symposium on High-Performance Computer Architecture (HPCA)*, pages 77–87, 2006.

[19] G. Magklis, M. L. Scott, G. Semeraro, D. H. Albonesi, and S. Dropsho. Profile-based Dynamic Voltage and Frequency Scaling for a Multiple Clock Domain Microprocessor. In *Proceedings of the 30th Annual International Symposium on Computer Architecture*, pages 14–27, June 2003.

[20] Mark Gebhart, Bertrand A. Maher, Katherine E. Coons, Jeff Diamond, Paul Gratz, Mario Marino, Nitya Ranganathan, Behnam Robatmili, Aaron Smith, James Burrill, Stephen W. Keckler, Doug Burger, and Kathryn S. McKinley. An Evaluation of the TRIPS Computer System. In *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, August 2009.

[21] M. Moudgill, P. Bose, and J. Moreno. Validation of Turandot, a Fast Processor Model for Microarchitecture Exploration. *International Performance, Computing and Communications Conference*, pages 451–457, February 1999.

[22] M.S.Govindan, D. Burger, S. W.Keckler, and the TRIPS Team. TRIPS: A Distributed Explicit Data Graph Execution Microprocessor. *Hot Chips 19: A Symposium of High Performance Chips*, August 2007.

[23] J. Neely, H. Chen, S. Walker, J. Venuto, and T. Bucelot. CPAM: A Common Power Analysis Methodology for High-Rerformance VLSI Design. pages 303–306, 2000.

[24] B. Robatmili, K. E. Coons, D. Burger, and K. S. McKinley. Strategies for Mapping Dataflow Blocks to Distributed Hardware. In *Proceedings of the 40th Annual International Symposium on Microarchitecture*, pages 23–34. IEEE Computer Society, November 2008.

[25] K. Sankaralingam, R. Nagarajan, H. Liu, C. Kim, J. Huh, D. Burger, S. W. Keckler, and C. R. Moore. Exploiting ILP, TLP, and DLP with the Polymorphous TRIPS Architecture. In *Proceedings of the 30th Annual International Symposium on Computer Architecture*, pages 422–433, June 2003.

[26] K. Sankaralingam, R. Nagarajan, R. McDonald, R. Desikan, S. Drolia, M. S. S. Govindan, P. Gratz, D. Gulati, H. Hanson, C. Kim, H. Liu, N. Ranganathan, S. Sethmadhavan, S. Sharif, P. Shivakumar, S. W. Keckler, and D. Burger. Distributed Microarchitectural Protocols in the TRIPS Prototype Processor. In *Proceedings of the 39th International Symposium on Microarchitecture*, pages 480–491, December 2006.

[27] G. Semeraro, G. Magklis, R. Balasubramonian, D. H. Albonesi, S. Dwarkadas, and M. L. Scott. Energy-Efficient Processor Design Using Multiple Clock Domains with Dynamic Voltage and Frequency Scaling. In *Proceedings of the 8th Annual International Symposium on High-Performance Computer Architecture*, pages 29–40, February 2002.

[28] T. Sherwood, E. Perelman, and B. Calder. Basic Block Distribution Analysis to Find Periodic Behavior and Simulation Points in Applications. In *Proceedings of the Internationl Symposium on Parallel Architectures and Compilation Techniques (PACT)*, pages 3–14, September 2001.

[29] D. Tarjan, M. Boyer, and K. Skadron. Federation: Out-of-Order Execution using Simple In-Order Cores. Technical Report CS-2007-11, University of Virginia, Department of Computer Science, August 2007.

[30] D. Tarjan, S. Thoziyoor, and N. Jouppi. Cacti 4.0. HP Labs Technical Report, HPL-2006-86. http://www.hpl.hp.com/techreports/2006/HPL-2006-86.pdf.

[31] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm design exploration. In *ISQED '06: Proceedings of the 7th International Symposium on Quality Electronic Design (ISQED'06)*, pages 585–590, 2006.

[32] H. Zhong, S. A. Lieberman, and S. A. Mahlke. Extending Multicore Architectures to Exploit Hybrid Parallelism in Single-thread Applications. In *Proceedings of the International Conference on High Performance Computer Architecture*, pages 25–36, February 2007.