The McPAT Framework for Multicore and Manycore Architectures: Simultaneously Modeling Power, Area, and Timing

SHENG LI, HP Labs
JUNG HO AHN, Seoul National University
RICHARD D. STRONG, University of California, San Diego
JAY B. BROCKMAN, University of Notre Dame
DEAN M. TULLSEN, University of California, San Diego
NORMAN P. JOUPPI, HP Labs

This article introduces McPAT, an integrated power, area, and timing modeling framework that supports comprehensive design space exploration for multicore and manycore processor configurations ranging from 90nm to 22nm and beyond. At microarchitectural level, McPAT includes models for the fundamental components of a complete chip multiprocessor, including in-order and out-of-order processor cores, networks-on-chip, shared caches, and integrated system components such as memory controllers and Ethernet controllers. At circuit level, McPAT supports detailed modeling of critical-path timing, area, and power. At technology level, McPAT models timing, area, and power for the device types forecast in the ITRS roadmap. McPAT has a flexible XML interface to facilitate its use with many performance simulators.

Combined with a performance simulator, McPAT enables architects to accurately quantify the cost of new ideas and assess trade-offs of different architectures using new metrics such as Energy-Delay-Area^2 Product (EDA^{2}P) and Energy-Delay-Area Product (EDAP). This article explores the interconnect options of future manycore processors by varying the degree of clustering over generations of process technologies. Clustering will bring interesting trade-offs between area and performance because the interconnects needed to group cores into clusters incur area overhead, but many applications can make good use of them due to synergies from cache sharing. Combining power, area, and timing results of McPAT with performance simulation of PARSEC benchmarks for manycore designs at the 22nm technology shows that 8-core clustering gives the best energy-delay product, whereas when die area is taken into account, 4-core clustering gives the best EDA^{2}P and EDAP.

Categories and Subject Descriptors: C.0 [**Computer Systems Organizations**]: General

General Terms: Performance, Verification

ACM Reference Format:
Li, S., Ahn, J. H., Strong, R. D., Brockman, J. B., Tullsen, D. M., and Jouppi, N. P. 2013. The mcpat framework for multicore and manycore architectures: Simultaneously modeling power, area, and timing. *ACM Trans. Archit. Code Optim.* 10, 1, Article 5 (April 2013), 29 pages.
DOI: http://dx.doi.org/10.1145/2445572.2445577

1. INTRODUCTION

It has always been true in this community that tools both limit and drive research directions. The power modeling framework Wattch [Brooks et al. 2000] has been such

Authors' addresses: S. Li, HP Labs, Palo Alto, CA; email: sheng.li@hp.com; J. H. Ahn, Seoul National University, Korea; email: gajh@snu.ac.kr; R. D. Strong, University of California at San Diego, CA; email: rstrong@cs.ucsd.edu; J. B. Brockman, University of Notre Dame, South Bend, IN; email: jbb@nd.edu; D. M. Tullsen, University of California at San Diego, CA; email: tullsen@cs.ucsd.edu; N. P. Jouppi, HP Labs, Palo Alto, CA; email: norm.jouppi@hp.com.

DOI: http://dx.doi.org/10.1145/2445572.2445577

a tool, enabling a tremendous surge in power-related architecture research. However, several factors drive the need for new tools to address changes in architecture and technology. This includes the need to accurately model multicore and manycore architectures, the need to model and evaluate power, area, and timing simultaneously, the need to accurately model all sources of power dissipation, and the need to accurately scale circuit models into deep-submicron technologies. This article describes a new power, area, and timing modeling framework called McPAT (Multicore Power, Area, and Timing), which addresses these challenges.

McPAT advances the current state-of-the-art in several directions. First, McPAT is an *integrated* power, area, and timing modeling framework that enables architects to study all three important metrics simultaneously on the same footing. Timing is a key element of the performance equation. The first goal of any processor design is to deliver its required performance, which imposes constraints on target cycle time or clock frequency. Power (or more accurately power density) and the resulting heat issues have become the most critical design constraint of modern and future processors. This concern only grows as the semiconductor industry continues to provide more transistors per chip in pace with Moore's Law. Area is also a key, often neglected, constraint to keep the cost of designs under control, as die costs are proportional to the fourth power of the die area [Rabaey et al. 2003] in theory, and proportional to the second power of the die area because of good yield in current process [Hennessy and Patterson 2011]. It is increasingly difficult for architects to consider any of these elements in isolation, as most architecture changes impact all three, as well as overall performance. By modeling power, area, and timing simultaneously, McPAT provides a solution for this serious challenge in computer architecture research.

Second, McPAT provides comprehensive power models. It models more than just dynamic power; this is critical for deep-submicron technologies because static power has become comparable to dynamic power [Semiconductor Industries Association 2007]. McPAT models all three types of power dissipation—dynamic, static, and short-circuit power—to give a complete view of the power envelope of multicore processors. Moreover, McPAT is the first architectural modeling framework that supports clock-gating and power-gating models, which enables architects to study advanced power management techniques such as P- and C-state [Naveh et al. 2006] power management of modern processors. It can interact with a performance simulator to study various power management alternatives.

Third, McPAT provides a complete, integrated solution for multithreaded and multicore/manycore processor power. Some researchers have combined core power models with a router power model [Kahng et al. 2009], but even that is an incomplete solution. Multicore and manycore processors are complex SoCs (System on Chip) that have powerful and sophisticated on-chip components including cores, caches, interconnects, and system controllers (such as memory controllers and Ethernet controllers, as can be seen in the Intel Sandybridge processor). McPAT models power, area, and timing of most of the important parts of multi/manycore processors, including all of the components listed before. McPAT supports more detailed and realistic models based on existing Out-Of-Order (OOO) processors. McPAT can model both a reservation-station model and a physical-register-file model based on real architectures, including the Intel P6 [Intel 1998], Netburst [Hinton et al. 2001], and SandyBridge [Yuffe et al. 2011]. Since McPAT already contains models for base single-threaded processors, multithreading support is included by modeling the sharing and duplication of hardware resources, as well as the extra hardware overhead. Thus McPAT can model the power, area, and timing of multithreaded processors, whether in-order CMT (Chip-MultiThreading) (e.g., Sun Niagara) or out-of-order SMT (Simultaneous MultiThreading) (e.g., Intel SandyBridge).

Fourth, McPAT handles technologies that can no longer be modeled by the linear scaling assumptions used by Wattch. The simple linear scaling principles are no longer valid because device scaling has become highly nonlinear in the deep-submicron era. McPAT uses technology projections from ITRS [Semiconductor Industries Association 2007] for dynamic, static, and short-circuit power; as a result, this tool will naturally evolve with ITRS even beyond the end of the current roadmap.

Fifth, McPAT is efficient and flexible. It can perform automatic optimizations on low-level design parameters while enabling the user to focus on high-level architectural investigation. This approach enables users, if they choose, to ignore many of the low-level details of the components being modeled. Moreover, rather than being hardwired to certain simulators, McPAT is the first architectural modeling framework that uses an XML-based interface to enable easy integration with various performance simulators. Thanks to the contributions from the McPAT user community, McPAT currently works with almost all major performance simulators including M5 [Binkert et al. 2006], GEMS [Martin et al. 2005], GEM5 [Binkert et al. 2011], Graphite [Miller et al. 2010], SST [Rodrigues et al. 2011], and Multi2Sim [Ubal et al. 2007].

Finally, McPAT enables architects to evaluate manycore processor designs from a new perspective by using new metrics. Introduced by McPAT, the metrics of Energy-Delay-Area^2 Product (EDA^{2}P) and Energy-Delay-Area Product (EDAP) are examples of comprehensive metrics that consider both performance and cost, including both the operational cost (energy) and the capital cost (area). While a chip vendor may favor EDA^{2}P as area^2 provides an approximation to die cost in practice, a system vendor could prefer EDAP as other fixed system costs such as memory and I/O reduce the overall system cost dependence on chip multiprocessor cost. The new metrics are shown to reveal new design sweet spots that cannot be found using other current metrics.

In this article, we also explore the scaling trends and the interconnect options of future manycore processors. We first evaluate the scaling trends of power, area, and timing of proposed manycore architectures across five technologies—90, 65, 45, 32, and 22nm—which covers years 2004 to 2016 of the ITRS roadmap. Several important inflection points on power, power density, and EDP have been revealed as technology advances into the nanoscale regime. We then study the trade-offs of core clustering in manycore processors by organizing cores into clusters with local interconnect. For both common in-order and out-of order manycore designs at 22nm technology node, by combining power, area, and timing results of McPAT with performance simulation of PARSEC benchmarks, our experiments show that when die cost is not taken into account clustering 8 cores together gives the best Energy Delay Product (EDP) whereas when cost is taken into account configuring clusters with 4 cores gives the best EDA^{2}P and EDAP.

McPAT first appeared in Li et al. [2009], after which it has successfully attracted the attention of, and has been widely used by, the computer architecture research community. After its first release, we have been improving McPAT. This article reflects the major upgrades of McPAT beyond its previous releases in the following aspects.

— We provide models for power-efficient embedded processor architectures including the Intel Atom and the ARM cortex A9. These new models enable architects to study power-efficient embedded processors over a wide range, from mobile computing to green data center applications. Validations on the two embedded processors show accurate modeling results.
— We give models for system components including on-chip network controllers, disk I/O controllers, and on-chip north bridges. These new models closely follow the latest processor designs such as Intel SandyBridge which have higher levels of integration.

— We include details and examples about the hierarchical modeling framework at all levels, from architecture to circuit and device level. We also describe newly implemented models for power-gating techniques.

The remainder of this article is organized as follows. After discussing related work in Section 2, we describe the overall structure of McPAT in Section 3. Sections 4 and 5 discuss the hierarchical, integrated model of power, area, and timing. Section 6 uses an example on an OOO processor renaming unit as a vehicle to navigate through McPAT's modeling framework. This work also presents the validation results in Section 7. In Section 8, we combine McPAT with performance simulators and explore the interconnect options of future manycore processors by varying the degree of clustering over generations of process technologies. We conclude in Section 9.

2. RELATED WORK

CACTI [Wilton and Jouppi 1994] was the first tool to address the need for rapid power, area, and timing estimates for computer architecture research, focusing on RAM-based structures. The most recent release of the tool [Thoziyoor et al. 2008] supports SRAM- and DRAM-based caches as well as plain memory arrays. It uses device models based on the industry-standard ITRS roadmap [Semiconductor Industries Association 2007], using MASTAR [Semiconductor Industries Association 2007] to calculate device parameters at different technology nodes. CACTI uses the method of logical effort to size transistors. It contains optimization features that enable the tool to find a configuration with minimal power consumption, given constraints on area and timing.

The complexity-effective approach [Palacharla et al. 1997] was one of the first attempts to use analytic models to obtain rapid estimates for processor timing, focusing on control, issue, selection, and bypass logic. Using generic circuit models for pipeline stages, it estimates the RC delay for each stage and determines the critical path.

Wattch [Brooks et al. 2000] is a widely used processor power estimation tool. Wattch calculates dynamic power dissipation from switching events obtained from architectural simulation and capacitance models of components of the microarchitecture. For array structures, Wattch uses capacitance models from CACTI, and for the pipeline it uses models from Palacharla et al. [1997]. When modeling out-of-order processors, Wattch uses the synthetic Register Update Unit (RUU) model that is tightly coupled to the SimpleScalar simulator [Burger and Austin 1997]. SimplePower [Vijaykrishnan et al. 2000] is another power estimator that uses a combination of analytical and transition-sensitive energy models. There were also similar but proprietary power estimation tools such as the IBM PowerTimer [Brooks et al. 2003], Intel ALPS [Gunther et al. 2001], and the Cai-Lim model [Cai and Lim 2012]. These tools have enabled the computer architecture research community to explore power-efficient design options, as technology has progressed; however, their limitations have recently become apparent.

Orion [Kahng et al. 2009] is a tool for modeling power in Networks-On-Chip (NoC). Version 2.0 includes models for area, dynamic power, and gate leakage, but does not consider short-circuit power or timing. It uses repeated wire models for interconnect, as well as device parameters for future technology nodes obtained from the ITRS roadmap using MASTAR and other methods. Kumar et al. provided further details on NoC layouts that take chip floorplans into consideration [Kumar et al. 2005].

Unlike the modeling tools that focus on power, area, or timing, HotSpot [Huang et al. 2006] is an architecture-level modeling framework for accurate and fast thermal estimation. It uses outputs from a power/area/timing modeling tool to derive thermal implications at architecture level. HotSpot makes it possible to study thermal evolution over long periods of real and full-length applications.

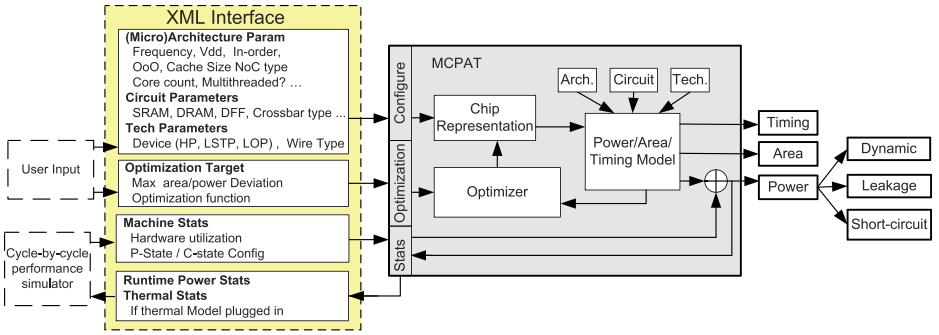


Fig. 1. Block diagram of the McPAT framework.

3. MCPAT: OVERVIEW AND OPERATION

McPAT is the first integrated power, area, and timing modeling framework for multithreaded and multicore/manycore processors. It is designed to work with a variety of performance simulators (and thermal simulators, etc.) over many technology generations. McPAT allows the user to specify low-level configuration details. It also provides default values when the user chooses to only specify high-level architectural parameters.

Figure 1 is a block diagram of the McPAT framework. Rather than being hardwired to a particular simulator, McPAT uses an XML-based interface with the performance simulator. This interface allows both the specification of the static microarchitecture configuration parameters and passing of dynamic activity statistics generated by the performance simulator. McPAT can also send runtime power dissipation back to the performance simulator through the XML-based interface, so that the performance simulator can react to power (or even temperature) data. This approach makes McPAT very flexible and easily ported to other performance simulators. Since McPAT provides complete hierarchical models from architecture to technology level, the XML interface also contains circuit implementation style and technology parameters that are specific to a particular target processor. Examples are array types, crossbar types, and the CMOS technology generation with associated voltage and device types.

The key components of McPAT are: (1) the hierarchical power, area, and timing models described in Section 4; (2) the optimizer for determining circuit-level implementations; and (3) the internal chip representation that drives the analysis of power, area, and timing. Most of the parameters in the internal chip representation, such as cache capacity and core issue width, are directly set by the input parameters.

McPAT's hierarchical structure allows it to model structures at a very low level, and yet still allows an architect to focus on the high-level configuration. The optimizer determines unspecified parameters in the internal chip representation, focusing on two major regular structures: interconnects and arrays. For example, the user can specify the frequency and bisection bandwidth of the network-on-chip, the capacity and the associativity of caches, and the number of cache banks, while letting the tool determine the implementation details such as the choice of metal planes, the effective signal wiring pitch for the interconnect, or the length of wordlines and bitlines of the cache bank. These optimizations lessen the burden on the architect to figure out every detail, and significantly lowers the learning curve to use the tool. Users always have the flexibility to turn off these features and set the circuit-level implementation parameters by themselves.

The main focus of McPAT is accurate power and area modeling, and a target clock rate is used as a design constraint. The detailed work flow of McPAT has two phases:

Table I. Feature and Execution Time of McPAT in Normal and Fast Mode on an Intel i7 3 GHz 8-Core Processor

Mode	Feature and Execution Time
`Normal`	Configuration meets timing constraints, or possible configurations are exhausted; in-order processor modeling time: $\sim$ 10 minutes; OOO processor modeling time: $\sim$ 20 minutes
`Fast`	Configuration balances timing, area, and power, but does not guarantee to meet timing constraints; in-order processor modeling time: $<$5 minutes; OOO processor modeling time: 5 $\sim$ 10 minutes

the chip representation building phase and the runtime power computation phase. In the chip representation building phase, the user of McPAT specifies the target clock frequency, the area and power deviation, the optimization function, and other architectural, circuit, and technology parameters. The optimization space that McPAT explores can be huge, especially when there are many unspecified parameters. McPAT performs an intelligent and extensive search of the design space. McPAT uses local greedy optimizations when searching the design space, except when there are obvious needs for considering the interplay of multiple components, which means McPAT finds the best solution for each component and assumes they will provide the best global configuration when individual components are put together. For each processor component, McPAT optimizes the circuit-level structure to satisfy the timing constraint. For the configurations satisfying the timing constraint, if the resulting power or area is not within the allowed deviation of the best value found so far, the configuration is discarded. Finally, among the configurations satisfying the power and area deviation, McPAT applies an optimization function to the internal chip representation that meets the target clock rate and reports the final peak power and chip-area values of the target processor design. Although local greedy optimizations are used for most components, there are exceptions when components need to be considered together. Currently, McPAT considers three global optimizations: (1) Bypass logic is assumed to be routed over functional units, register files, and reservation stations. (2) Global interconnects (links between routers) are assumed to be routed over last-level caches, if they are present, in horizontal, vertical, or even both directions. (3) Clock distribution networks are assumed to cover the entire chip using a global H-tree or entire domains such as a core using a semiglobal H-tree. Local clock distribution networks are assumed to cover each entire domain. In the runtime power computation phase, the peak power of individual units and the machine utilization statistics (activity factor) are used to calculate the final runtime power dissipation.

The chip representation building phase is the most time-consuming step, since it will repeat many times until valid configurations are found or the possible configurations are exhausted. In order to reduce its execution time, McPAT also provides a fast mode that can give a configuration with balanced power, area, and timing but does not guarantee to meet timing constraints. The fast mode enables expedited early-stage design space exploration. The execution time of McPAT in normal and fast mode on an Intel i7 3 GHz 8-core processor is shown in Table I. Since McPAT runs separately from a performance simulator and only reads performance statistics from it, its impact on the simulation speed of the native performance simulator is minimal. Although the chip representation building phase of McPAT may take some time to complete because of the huge search space, it will not affect the simulator speed significantly since it needs to be done only once at the beginning of the simulation. During the runtime power computation phase, simulator overhead may only result from added performance counters.

Another distinguishing feature of McPAT is its ability to model advanced power management techniques, such as P- and C-state [Naveh et al. 2006] power management of modern processors. After calling McPAT to finish initialization, a performance simulator can pass statistical information and invoke McPAT anytime during the

simulation. McPAT will then calculate the corresponding power dissipation for the particular period and send it back to the performance simulator when required. This allows the simulator to react to simulated power or thermal sensors (assuming a temperature model is attached to the backend), by changing voltage and frequency settings, or invoking one of multiple power-saving states on idle circuit blocks. When connected with a temperature model such as Huang et al. [2006], McPAT computes leakage power based on the temperature values passed from the thermal simulator. (This will require multiple iterations until the power and temperature reach steady state. An early experiment on the temperature-dependent leakage modeling can be found in Rodrigues et al. [2011].) This allows the architect to use the framework to model the full range of power management alternatives.

4. MODELING FRAMEWORK

In order to model the power, area, and timing of a multicore processor, McPAT takes an integrated and hierarchical approach. It is integrated in that McPAT models power, area, and timing simultaneously. Because of this, McPAT is able to ensure that the results are mutually consistent from an electrical standpoint. It is hierarchical in that it decomposes the models into three levels: architecture, circuit, and technology. This provides users with the flexibility to model a broad range of possible multicore configurations across several implementation technology generations. Taken together, this integrated and hierarchical approach enables the user to paint a comprehensive picture of a design space, exploring trade-offs between design and technology choices in terms of power, area, and timing.

4.1. Power, Area, and Timing Models

Power modeling. Power dissipation of CMOS circuits is modeled using Eq. (1) and has three main components: dynamic, short-circuit, and leakage power. Circuits dissipate dynamic power when they charge and discharge capacitive loads to switch states. As in Eq. (1), dynamic power is proportional to the total load capacitance (C), the supply voltage (V_{dd}), the voltage swing during switching (ΔV), the clock frequency (f_{clk}), and the activity factor (α). We calculate the load capacitance of a module by decomposing it into basic circuit blocks, and use analytic models for each block with appropriately sized devices. We calculate the activity factor using access statistics from architectural simulation together with circuit properties.

Switching circuits also dissipate short-circuit power due to a momentary short through the pull-up and pull-down devices. Short-circuit power is determined by the short-circuit energy per switch operation (E_S), the clock frequency (f_{clk}), and the activity factor (α) as in Eq. (1). The short-circuit energy per switch operation (E_S) is computed using the equations derived in the work by Nose and Sakurai [2000] that predicts trends for short-circuit power. If the ratio of the threshold voltage to the supply voltage shrinks, short-circuit power becomes more significant. It is typically about 10% of the total dynamic power, however, it can be as high as 25% of the dynamic power in some cases [Nose and Sakurai 2000].

$$P_{total} = \underbrace{\alpha C V_{dd} \Delta V f_{clk}}_{Dynamic} + \underbrace{\alpha E_S f_{clk}}_{Short\_circuit} + \underbrace{V_{dd} I_{leakage}}_{Leakage} \tag{1}$$

The third term in Eq. (1) is the *static* power consumed because of *leakage* current through the transistors, which in reality function as "imperfect" switches. Leakage power, or static power, depends on the leakage current ($I_{leakage}$) and the supply voltage (V_{dd}). Leakage current depends on the width of the transistors and the local state of the devices. There are two leakage mechanisms. Subthreshold leakage occurs because

a small current passes between the source and drain of off-state transistors. Gate leakage is the current leaking through the gate terminal, and varies greatly with the state of the device. We determine the unit leakage current using MASTAR [Semiconductor Industries Association 2007] and Intel's data [Auth et al. 2008] and compute the leakage current of a device using the unit leakage current and the size of the device. For each device type (e.g., inverter, NAND, NOR, etc.), we first compute the leakage currents of the device in different logical states and then compute weighted average leakage current that is used for further computation in McPAT.

Timing modeling. Like the power model, McPAT's timing model breaks the system down into components, circuit blocks, and eventually devices. While the power model requires only the capacitance to compute dynamic power, the timing model uses both resistance and capacitance to compute RC delays. McPAT computes the resistance and capacitance of individual devices and then uses the CMOS gate RC delay model by Horowitz [1984] to compute the delays of simple CMOS gates. For more complex RC network structures such as multidrop buses, McPAT uses the delay model [Horowitz 1984]) as shown in Eq. (2).

$$T = \tau \ln \frac{V_{end}}{V_{start}}, \tag{2}$$

where τ is the time constant of the complex RC network structure, computed with the Elmore delay method [Elmore 1948]; V_{start} and V_{end} are the beginning voltage and the voltage when the circuit is considered to have "switched", respectively. McPAT determines the achievable clock frequency of a processor from the delays of its components along the critical path.

Area modeling. McPAT uses an analytical methodology to model basic logic gates and regular structures, including memory arrays (e.g., RAM, CAM, and DFFs (D flip-flops)), interconnects (e.g., router and link), and regular logic (e.g., decoder and dependency-checking unit). The areas of individual devices are computed together with the resistance and capacitance, since the resistance and capacitance of the devices are determined by the sizes of the devices as well as other physical device properties. The areas of regular circuit blocks are calculated by summing the area of all transistors and the wiring overhead. An algorithmic approach does not work well for complex structures that have custom layouts, such as ALUs. For these, currently McPAT takes an empirical modeling approach [Gupta et al. 2000; Rodrigues 2007] which uses curve fitting to build a parameterizable numerical model for area from published information on existing processor designs, and then scales the area for different target technologies.

4.2. Hierarchical Modeling Framework

McPAT's integrated power, area, and timing models are organized in a three-level hierarchy, as illustrated in Figure 2. This is the first modeling framework which completely models a multicore/manycore processor from architecture to technology level. On architectural level, a multicore processor is decomposed into major architectural components such as cores, NoCs, caches, memory controllers, and clocking. On circuit level, the architectural building blocks are mapped into four basic circuit structures: hierarchical wires, arrays, logic, and clocking networks. On technology level, data from the ITRS roadmap [Semiconductor Industries Association 2007] is used to calculate the physical parameters of devices and wires, such as unit resistance, capacitance, and current densities. It is worth noting that this hierarchical modeling framework gives McPAT another degree of freedom in that an architect can use the basic building blocks to compose new hardware structure for emerging microarchitecture designs.

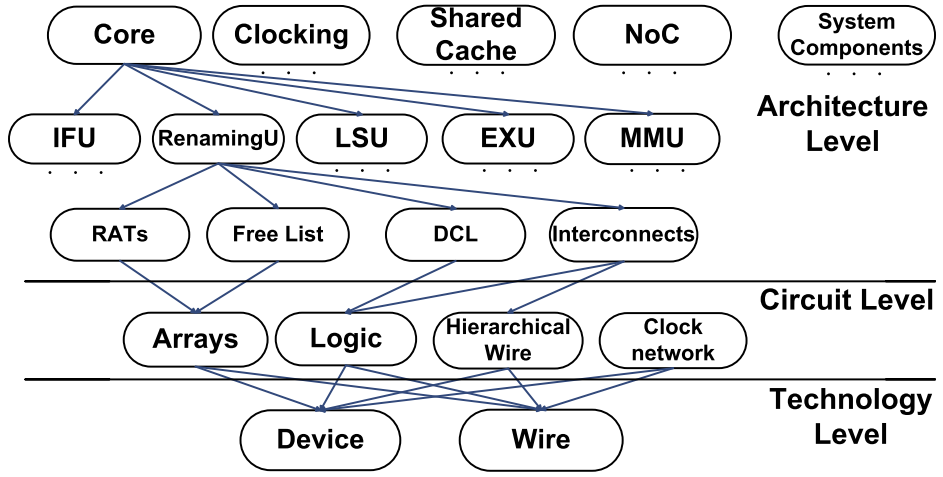


Fig. 2. Hierarchical modeling methodology of McPAT, with core and renaming logic showing microarchitecture breakdown.

4.2.1. Multicore Architecture-Level Modeling. The architecture level represents the building blocks of a multicore processor system. Next, we provide an overview of the models for these high-level blocks and how they are mapped to circuit level.

Core. A core can be divided into several main units: an Instruction Fetch Unit (IFU), an Execution Unit (EXU), a Load and Store Unit (LSU), and an Out-Of-Order (OOO) issue/dispatch unit for an OOO processor. Each of them can be further divided into hardware structures. For example, the EXU may contain ALUs, FPUs, bypass logic, and register files. In our hierarchical framework, the ALU and FPU are mapped to the complex logic model at circuit level. Bypass logic can be mapped to a combination of the wire and logic models, while register files can be mapped to the array model. McPAT supports detailed and realistic models that are based on modern industry processor designs including high-performance OOO processors, throughput-oriented multithreaded in-order processors, and lower-power embedded processors. For OOO processors, we greatly extend the basic analytical models in Palacharla et al. [1997] to support both reservation-station-based (data-capture scheduler) architectures such as the Intel P6 architectures, the Intel core architectures, and the Intel Nehalem architectures [Intel 1998] as well as the physical-register-file-based (nondata-capture scheduler) architectures such as the Intel Netburst architecture, the Intel Sandy Bridge architecture [Hinton et al. 2001; Yuffe et al. 2011], and the DEC Alpha architecture [Kessler 1999]. McPAT supports both RAM- and CAM-based renaming logic, which can be found in the Intel and Alpha architectures.

Besides their smaller caches and buffers (compared to those of the high-performance OOO processors), the low-power embedded processors have other major architectural differences. For example, the ARM Cortex A9 processor [ARM 2013] leverages OOO execution without traditional power/resource-hungry ROBs by sacrificing hardware speculation at the cost of a long branch misprediction penalty. McPAT captures these details and models the power, area, and timing of embedded processors based on the ARM cortex A9/A15 [ARM 2013] and Intel Atom [Intel 2013].

NoC. A NoC has two main components: links and routers. For links, we use hierarchical repeated wires, as described in Section 4.2.2. McPAT models traditional four-stage routers. The four stages of a traditional router are Route Computation

(RC), Virtual Channel Allocation (VA), Switch Allocation (SA), and Switch Traversal (ST). We use the same analytical approach used in modeling cores to model routers: breaking the routers into basic building blocks such as flit buffers, arbiters, crossbars, and inter-stage flip-flips; then building analytical models for each building block. Unlike Orion 2 [Kahng et al. 2009] that only models area and power, McPAT models power, area, and timing. McPAT is the first modeling tool supporting a double-pumped crossbar [Vangal et al. 2005], which reduces die area for on-chip interconnect intensive designs. A bus is a special case of NoCs, which are modeled as arbiters, buffers, and interconnects. Unlike routers that use crossbars, the physical interconnects of buses are modeled as hierarchical repeated wires with MUXes that are controlled by arbiters.

On-chip caches. McPAT supports both shared and private caches. It models private/-coherent caches by modeling the directory structure associated with the caches. Depending on the architecture, a directory can be mapped to CAM structures at circuit level as in Niagara processors [Kongetira et al. 2005; Nawathe et al. 2008] or normal cache structures as in the Alpha 21364 [Jain et al. 2001]. McPAT also models the recent distributed directory structure that merges directory information into cache structures and supports dynamic home nodes for future manycore processors [Ferdman et al. 2011; Marty and Hill 2007].

System controller. McPAT is the first modeling framework to model on-chip system controllers including memory controllers, network controllers, PCIe controllers, and SATA controllers. McPAT also models high-speed system interfaces such as Intel's QuickPath Interconnect (QPI) [Intel 2009] and AMD's HyperTransport [AMD 2002]. Most of these system controllers can be divided into three parts: (1) the frontend engine responsible for scheduling (or rescheduling) the requests, (2) the transaction processing engine that has the logic and sequencer to generate the command, address, and data signals, and (3) the physical interface (PHY) that serves as an actual channel of the controller for off-chip communications.

The frontend engine is partially modeled using CAM and RAM structures. To model the control logic in the frontend engine and the entire transaction processing engine in the system controllers we use Cadence's ChipEstimator [Cadence InCyte Chip Estimator 2013], which provides power, area, and timing information of different controllers taken from an extensive library of IP (Intellectual Property) blocks. The physical interfaces (PHYs) in these controllers are high-speed point-to-point SerDes links, for which we model both a high-performance and low-power types. The high-performance PHYs are modeled based on a design from Texas Instruments [Harwood et al. 2007] that supports a backplane crossing in typical blade-based server designs [HP 2009] (channel attenuation of -24 dB and a maximum bit-error rate of 10^{-17}). The low-power PHYs are modeled according to Fukuda et al. [2010] and Palmer et al. [2007] and are good for short distance communications (channel attenuation of -12 dB/-15 dB and a maximum bit-error rate of $10^{-12}/10^{-15}$). We then scale the area, bandwidth, and power to different technology generations based on SerDes scaling trends, as summarized in Palmer et al. [2007].

The 10Gb Ethernet controller is a special case because the controller usually contains extra TCP/IP offloading accelerators and packet filters since the processor overhead of supporting a NIC without accelerators is very high; a 10Gb NIC port can easily saturate a 2.33 GHz Intel Xeon E5345 core [Ram et al. 2009]. Thus, it is common for server class NICs to include TCP/IP accelerators, which can be seen in Niagara2 [Johnson and Nawathe 2007] and Niagara3 [Shin et al. 2010], as well as off-chip server-grade NICs such as the Broadcom 10Gb MAC controller [Broadcom 2008]. We follow this design choice and assume a TCP/IP-accelerated NIC in our models.

Clocking. Clocking circuitry has two main parts: the Phase-Locked Loop (PLL) with fractional dividers to generate the clock signals for multiple clock domains, and the clock distribution network to route the clock signals. McPAT uses an empirical model for the power of a PLL and fractional divider, based on scaling published results from Sun and Intel [Ahn and Allstot 2000; Rusu et al. 2006]. The clock distribution network can be directly mapped to the clock network model at circuit level.

4.2.2. Circuit-Level Modeling. Hierarchical repeated wires. Hierarchical repeated wires are used to model local, semiglobal, and global on-chip interconnect. The performance of wires is governed by two parameters: resistance and capacitance. Wires scale more slowly than gates with respect to RC delays, and nonrepeated wires cannot keep up with improved transistor delays. For long wires, we use a repeated wire model [Ho 2003; Horowitz et al. 1999], where we optimize the size of the repeaters and the distance between adjacent repeaters for either delay or energy-delay product. We assume multiple metal planes for each interconnect, each with different wire pitches and aspect ratios. The assignment of signals to wiring planes plays a key role in determining their power, area, and timing characteristics. McPAT's optimizer automatically assigns wires to different metal layers in the hierarchical repeated wire model at circuit level and computes the delays of the link at each metal layer. If the timing constraint (throughput or latency) cannot be satisfied at the current metal layer, McPAT will vary the signal wiring pitch, use double- or triple-wide metals, or move to higher levels of metal, trying to satisfy the timing requirement. For throughput-centric interconnects such as the heavily pipelined global interconnects which route across the entire chip, latches are also inserted and modeled when necessary to satisfy the target throughput.

Arrays. McPAT includes models for three basic array types at circuit level: RAM-, CAM-, and DFF-based arrays. RAM (cache) and CAM structures are modeled based on CACTI, with several important extensions. For example, McPAT reimplements the CAM model from CACTI to more accurately reflect its multibanked/ported (with search, read, and write ports) structure and adds priority arbitration logic and selection logic that are key to OOO cores. McPAT implements a detailed DFF array model to support the modeling of registers between pipeline stages in both processors and routers. McPAT also adds gate leakage and improved timing models for all arrays.

Logic. McPAT employs three different schemes for modeling logic blocks, depending on the complexity of the block. For highly regular blocks with predictable structures, such as memories or networks, McPAT uses the algorithmic approach of CACTI [Thoziyoor et al. 2008]. For structures that are less regular but can still be parameterized, such as thread selection or decoding logic, McPAT uses analytic models similar to those in Palacharla et al. [1997] but modeled after existing processors from Intel, AMD, and Sun. Finally, for highly customized blocks such as functional units, McPAT uses empirical models based on published data for existing designs scaled to different technologies. For example, the ALU and FPU models are based on actual designs by Intel [Mathew et al. 2005] and Sun [Leon et al. 2007].

Clock distribution network. A clock distribution network is responsible for routing clock signals of different frequencies to clock domains, with drops to individual circuit blocks. This network is a special case of a hierarchical repeated wire network, but has strict timing requirements with large fanout loads spanning the entire chip. It is wave pipelined [Rabaey et al. 2003], which is very different from other global wires. It consumes a significant fraction of total chip power [Gowan et al. 1998]. We represent a clock distribution network using a separate circuit model that has three distinct levels: global, domain, and local. We assume an H-tree topology for global-level and domain-level networks and a grid topology for the local networks as shown in both the

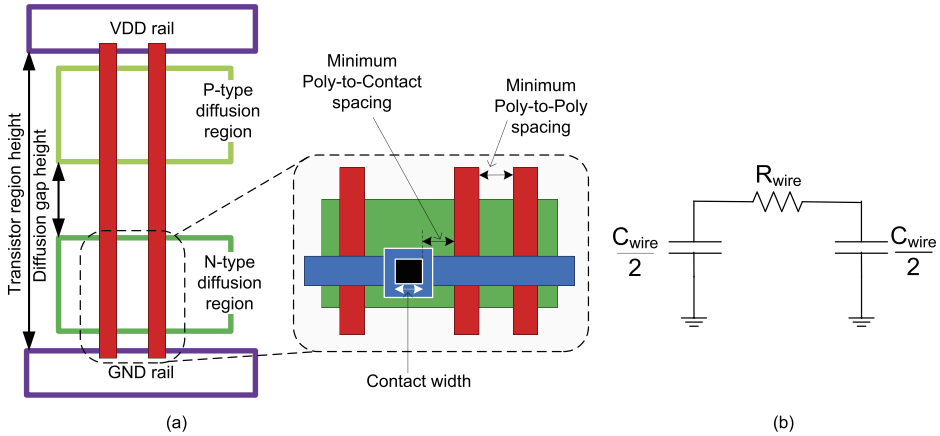


Fig. 3. Technology-level models. (a) generic area model of transistors and gates based on Yoshida et al. [2004]; (b) π-RC model for wires.

Niagara processors [Kongetira et al. 2005; Nawathe et al. 2008] and the Intel Itanium processors [Mahoney et al. 2005]. NAND gates are used at all final clock heads to enable clock gating.

4.2.3. Technology-Level Modeling. There are two categories of models at technology level: device models and wire models. McPAT uses MASTAR [Semiconductor Industries Association 2007] to derive device parameters from the ITRS [Semiconductor Industries Association 2007]. The current implementation of device models in McPAT includes data for the 90nm, 65nm, 45nm, 32nm, 22nm, and 16nm technology nodes, which covers the ITRS roadmap through 2019. ITRS assumes that planar bulk CMOS devices will reach practical scaling limits at 36nm, at which point the technology will switch to Silicon-On-Insulator (SOI). Below 25nm, the ITRS predicts that SOI will reach its limits and that double-gate and FinFET devices will be the only option. McPAT captures each of these options, making it scalable with the ITRS roadmap. McPAT also models gate leakage based on MASTAR and assuming hi-k metal gate devices are used for 45nm technology and beyond. Using the fundamental device parameters from ITRS and computed device sizes, McPAT calculates the resistance and capacitance of individual devices, including full-on resistance, switch resistance, gate capacitance, and drain/source capacitance. The same methodology as in CACTI [Li et al. 2011; Thoziyoor et al. 2008; Wilton and Jouppi 1994] is used for device modeling, including stacked transistors and folded transistors.[1] While computing device resistance and capacitance, McPAT calculates the area of individual devices and gates simultaneously. Figure 3(a) shows the generic area model of transistors and gates used in McPAT, which is based on a layout model in Yoshida et al. [2004]. The area of a circuit component is calculated by summing the area of all transistors/gates and wiring overhead.

We model nonrepeated metal wires using a one-section π-RC model [Thoziyoor et al. 2008] as shown in Figure 3(b) and use Ho [2003] and Horowitz et al. [1999] to derive wire parameters for different technology nodes from 90nm to 16nm. We also consider

[1]When the width of a transistor, such as a large sleep transistor or a repeater in the global interconnects, exceeds a certain maximum value, the transistor is assumed to be folded. This maximum value can either be process specific or context specific. An example of when a context-specific width would be used is in the case of repeaters which typically have to be laid out at a certain pitch.

Table II. Sharing and Duplication Assumptions for Hardware Resources within a Core in McPAT for Multithreaded Processors

Architecture	Private and Duplicated Units	Partitioned and Tagged Units	Shared Units
SMT	Instruction buffers, RAS, Architecture RF, RAT (FRAT and RRAT), inter-stage buffers	ITLB, DTLB, BTB, BPT, ROB, instruction decoders, load buffers, store buffers	Functional units, Icache, Dcache, L2cache, Reservation Station, instruction issue queue, physical register files
CMT	Instruction buffers, RAS, architecture RF, inter-stage buffers	ITLB, DTLB, BTB, BPT, instruction decoders, load buffers, store buffers	Functional units, Icache, Dcache, L2cache

double/triple/quad-width wires for each wire type in the interconnect technology model.

4.3. Modeling Multithreaded Architectures

McPAT models the power, area, and timing of multithreaded processors, whether in-order CMT (Chip-MultiThreading) (e.g., Sun Niagara) or out-of-order SMT (Simultaneous MultiThreading) (e.g., Intel Nehalem). Since McPAT already contains models for each of the base processors, multithreading support is included by modeling the sharing and duplication of hardware resources, as well as the extra hardware overhead. McPAT models multithreaded architectures based on designs of the Niagara processors [Kongetira et al. 2005; Nawathe et al. 2008], Intel hyperthreading technology [Koufaty and Marr 2003], and early research in SMT architecture [Tullsen et al. 1996].

Table II shows the sharing and duplication assumptions for hardware resources within a multithreaded processor core in McPAT. All private units are modeled in a duplicated manner, with the number of copies being equal to the number of hardware threads. Partitioned units are modeled with extra thread IDs in hardware. Each entry of the partitioned unit is assumed to include a CAM portion to store thread IDs as tags. Therefore some of these units, such as the branch predictor, are changed from RAM structures in single-threaded processors to cache-like structures with both tag and data arrays in multithreaded processors. Models of shared units in multithreaded processors are the same as those of single-threaded processor models. Since McPAT computes energy per access to calculate final dynamic power, the duplicated units for nonactive threads will not affect the dynamic power. However, these units will affect the leakage power and area of a multithreaded target design compared to a single-threaded version. Besides the CMT and SMT cores, McPAT also models the conjoined-core architecture [Kumar et al. 2004]. One example of the conjoined-core architecture is the latest module-based multithreaded architecture as in the AMD Bulldozer [Butler 2010], where a module is the middle ground between dual single-threaded cores and a single core with two SMT threads.

5. MODELING POWER MANAGEMENT TECHNIQUES

McPAT models two major power-saving techniques: clock gating to reduce dynamic power and power gating to reduce static power. Combined with the flexible XML interface, these models enable McPAT to support advanced power management

techniques, such as P- and C-states [Naveh et al. 2006]. A P-state is an operational state of a core/processor defined by a combination of clock frequency and supply voltage. By changing the clock frequency and supply voltage (and the associated device parameters), an architect can use McPAT to model P-state management techniques. A C-state is an idle state that is characterized by the amount of power consumed during the state and the latency and power consumption to enter and exit the state [Naveh et al. 2006]. C-state management starts from clock-gating processor components at C1 state and then applying both clock gating and power gating when going to deeper C-states. McPAT models C-states as the combination of clock gating and power gating according to Naveh et al. [2006].

McPAT models the actual clock gating buffer circuitry (NAND gates and inverters) at the clock distribution network heads, rather than using a heuristic approach as in Wattch [Brooks et al. 2000]. Having different power-gating modes allows trade-offs between the power saving and the wakeup overhead with respect to wakeup power and delay. Thus, McPAT models sleep transistor designs to support multiple power-gating states including active, sleep, and shut-off, as in the latest Intel processor designs [George et al. 2007; Kumar and Hinton 2009; Rusu et al. 2006; Wang et al. 2010; Yuffe et al. 2011]. The active mode is the state when the circuit block is fully active.[2] The shut-off state is when the circuit block is fully idle and the virtual power supply reduces to almost zero[3]. The sleep mode is state retentive, while the shut-off mode results in losing circuit information.

McPAT incorporates power-gating models for state-of-the-art design techniques based on the power gating modeling framework in CACTI-P [Li et al. 2011] including: (1) the types and sizes of sleep transistors, (2) power-gating granularity, (3) the placement and topology of sleep transistors, and (4) the supply voltage of idle circuit blocks. Both header switch (PMOS) and footer switch (NMOS) are modeled as sleep transistors in McPAT to support power gating. While PMOS sleep transistors leak less than NMOS transistors of the same size and thus save more static power, NMOS sleep transistors drive more current so that these are more area efficient. McPAT chooses the sleep transistor type for individual circuit blocks according to power, area, and timing during optimization. We assume that sleep transistors are applied for individual components such as caches and functional units, although different components can be combined as large power islands to relax the complexity of on-chip power supply and delivery. McPAT models Distributed Sleep Transistor Networks (DSTNs) [Long and He 2004] for all components including both analytically modeled regular circuit structures and empirically modeled irregular circuit structures. DSTN achieves smaller overhead than traditional module-based [Kao et al. 1998] and cluster-based [Anis et al. 2002] sleep transistor designs, with the same performance loss. The supply voltages of idle circuit blocks are obtained from the latest Intel Designs [Rusu et al. 2006; George et al. 2007; Wang et al. 2010] and SPICE simulations, considering the state retention requirements of memory arrays and sequential logic in the sleep state.

6. RENAMING UNIT MODELING—AN EXAMPLE OF THE HIERARCHICAL MODELING FRAMEWORK

The renaming unit in OOO processors is the critical component for eliminating the false data dependencies, including Write-After-Write (WAW) and Write-After-Read (WAR), and maintaining true Read-After-Write (RAW) data dependencies. It is also

[2]PMOS sleep transistors associated with the circuit block work in the linear region. Because of the large width of the sleep transistors, the virtual power supply V_{VDD} is virtually identical to the real power supply when active.

[3]PMOS sleep transistors are completely off when the circuit block is in shut-off state.

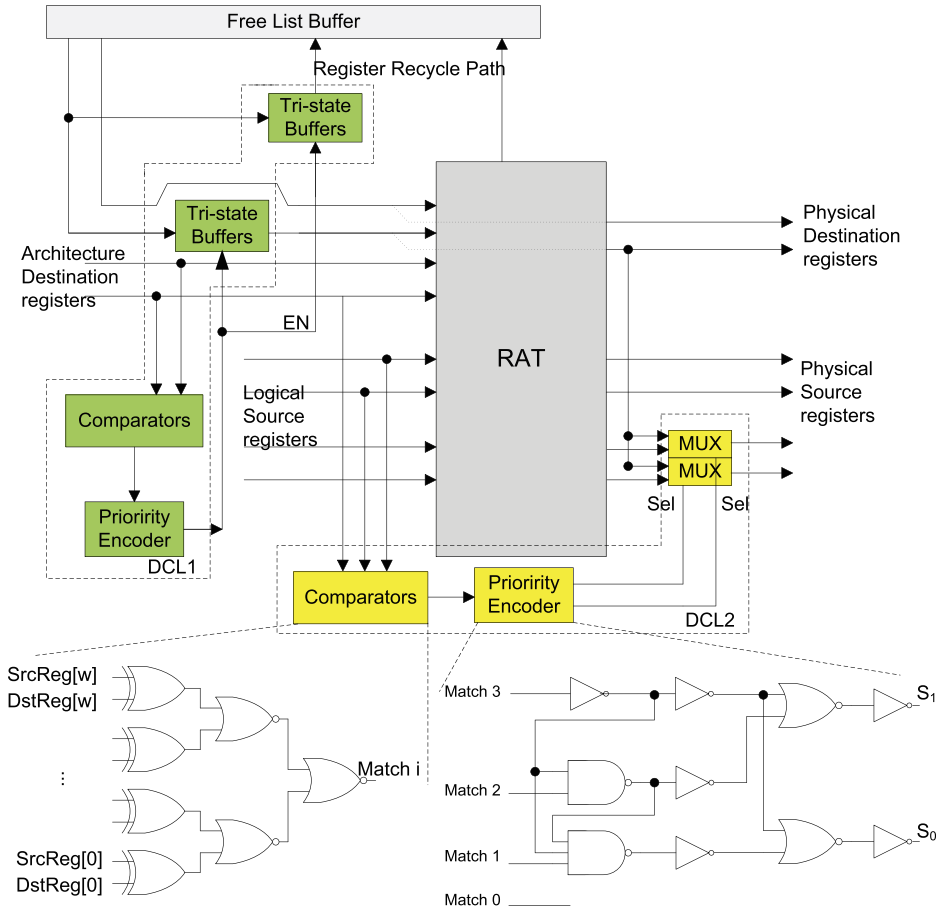


Fig. 4. Architectural- and circuit-level models of renaming logic in McPAT, with three major parts: the register alias table, the free-list buffer, and the dependence check logic.

very power/resource hungry [Gowan et al. 1998]. In this section, we use it to illustrate how the architecture components are decomposed into circuit- and technology-level models in McPAT. Figure 4 shows the renaming unit modeled in McPAT with three major parts: the Register Alias Table (RAT), the free-list buffer, and the Dependence Check Logic (DCL).

Free-list buffer. The free-list buffer provides the designators of available physical registers (or reorder buffer entries) during the renaming process. It is modeled as a multiported FIFO mapped to an SRAM array with R read ports, C write ports, and P entries at circuit level, where R, C, and P are the renaming width, commit width, and the number of physical registers (or reorder buffer entries) of the core, respectively.

Register alias table (RAT). The RAT stores the active mapping between architectural registers and physical registers (or reorder buffer entries) and maintains RAW dependencies for in-flight instructions. McPAT supports both a RAM-based RAT as in the Intel P6 [Intel 1998] and Intel NetBurst architectures [Hinton et al. 2001] as well as a CAM-based RAT as in Alpha processors [Kessler 1999]. The RAM-based RAT is modeled as an SRAM array with the number of entries being equal to the number of

architectural registers. The entries are indexed by the designator of the architectural registers. The physical register designators for the corresponding renamed architectural registers are stored in the entries. For the CAM scheme, the RAT is modeled and mapped to CAM array models at circuit level with the number of entries being equal to the number of physical registers (or reorder buffer entries). Each entry has two data fields: one data field contains the designator of the architectural register that is mapped to this physical register, and the other data field contains a valid bit to indicate that the current mapping is valid. McPAT assumes that the RAT needs $2 * W$ read ports and W write ports [Sima 2000], where W is the issue width.

For physical-register-based cores, the physical register file holds both speculative and nonspeculative data. However, the nonspeculative subset of the data must be explicitly denoted when the processor state must be saved in case of interrupts or context switches. This requirement leads to the dual-RAT technique [Hinton et al. 2001]. McPAT models the dual-RAT scheme in its physical-register-based core models following the design of the Intel Netburst and Sandy Bridge architectures, where a Frontend RAT (FRAT) is used for register renaming as usual and a Retire RAT (RRAT) is used to store the current mapping for architectural registers. The RRAT is only accessed during the instruction commit stage and does not affect the FRAT. While the FRAT is modeled using both RAM and CAM schemes in McPAT, the RRAT is always modeled as a RAM structure with its number of write ports equal to the commit width of the modeled core. McPAT also models checkpointing hardware in the renaming unit that can reduce branch misprediction penalties based on the Alpha 21264 processor [Gowan et al. 1998].

Dependency check logic (DCL). As shown in Figure 4, there are two sections of Dependency Check Logic (DCL). The first section (DCL1) checks the WAW dependency among destination registers of instructions in the current renaming pool. DCL1 has three major parts: the comparators, the priority encoders, and the tristate buffers. When a dependency is detected among destination registers, the outputs of the priority encoders act as the enable signals, disabling the write path to the RAT and enabling the recycle path to the free-list buffer. The comparators are modeled based on Bishop et al. [1999]. Since McPAT supports processors with issue widths up to eight it is important to have the DCL logic scalable so that it will not limit the clock rate. We model a scalable priority encoder as shown in Figure 4. The total number of comparator sets of this section is $W * (W - 1)$, where W is the decode width.

The second section of DCL (DCL2) is shown in Figure 4. DCL2 is necessary to maintain RAW dependencies by detecting cases where the registers being renamed are destination registers of an earlier instruction in the same renaming pool. It detects RAW dependencies, chooses the correct mapping of the source registers, and sends them to later pipeline stages. DCL2 is modeled as multiple sets of comparators, priority encoders, and multiplexors according to Bishop et al. [1999]. The number of comparator sets used for DCL2 is twice the number used in for DCL1.

Power, area, and timing of the renaming unit. The total energy, delay, and area of the renaming unit is shown in Eqs. (3)–(5),

$$T = max(T_{DCL1}, T_{FreeList}) + max(T_{RATread}, T_{RATwrite}), T_{DCL2}), \tag{3}$$

$$Energy_{RenamingLogic} = E_{DCL} + E_{RAT} + E_{FreeList}, \tag{4}$$

$$Area_{RenamingLogic} = (A_{DCL} + A_{RAT} + A_{FreeList}), \tag{5}$$

where $T_{FreeList}$ is the access time of the free-list buffer, T_{DCL1} is the delay of DCL1, T_{DCL2} is the delay of DCL2, and $T_{RATread}/T_{RATwrite}$ is the read/write latency of the RAT (in a dual-RAT scheme, FRAT determines the timing of the critical path). The

total delay must be less than the target clock period in order to satisfy the timing constraint. Energy consumed by each access is computed as shown in Eq. (4), where E_{DCL} is the energy per access of the dependency check logic, E_{RAT} is the energy per access for the RAT which depends on the operation type (in a dual-RAT scheme, E_{RAT} is the total energy of FRAT and RRAT for the same instruction), and $E_{FreeList}$ is the energy per access of the free-list buffer. The area of the renaming unit is shown in Eq. 5, where A_{DCL} is the total area of DCL1 and DCL2, A_{RAT} is the area of the RAT (in a dual-RAT scheme, A_{RAT} is the total area of the FRAT and RRAT), and $E_{FreeList}$ is the area of the free-list buffer.

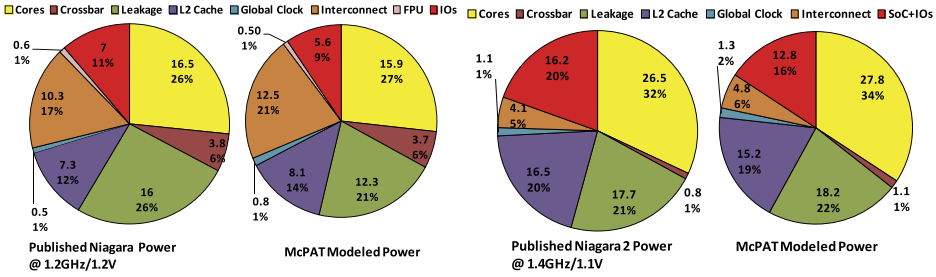
7. VALIDATION OF MCPAT

The primary focus of McPAT is accurate power and area modeling at architectural level when timing is given as the main design constraint. Both *relative* and *absolute* accuracy are important for architectural-level power modeling. Relative accuracy means that changes in modeled power as a result of architecture modifications should reflect the changes one would see in a real design on a relative scale. While relative accuracy is critical, absolute accuracy is also important if one wants to compare results against Thermal Design Power (TDP) limits, or to put power savings in the core in the context of the whole processor or whole system. The relative accuracy of McPAT ensures that the relative power weights of different components of a chip have been correctly modeled. The absolute accuracy of McPAT means that the power numbers for individual components and total power are evaluated correctly.

We compare the output of McPAT against published data for the 90nm Niagara processor [Leon et al. 2007] running at 1.2 GHz with a 1.2V power supply, the 65nm Niagara2 processor [Nawathe et al. 2008] running at 1.4 GHz with a 1.1V power supply, the 65nm Xeon processor [Rusu et al. 2006] running at 3.4 GHz with a 1.25V power supply, the 180nm Alpha 21364 processor [Jain et al. 2001] running at 1.2 GHz with a 1.5V power supply, the 45nm dual-core Diamondville [Intel 2013] Atom processor running at 1.6 GHz with a 1.0V power supply, and the 40nm Cortex A9 dual-core hard IP implementation running at 2.0 GHz [ARM 2013]. These include in-order and out-of-order processors, single-threaded and multithreaded processors, as well as high-performance and embedded processors in the validation targets. Thus, the validations stress McPAT in a comprehensive and detailed way. The comparisons against Niagara and Niagara2 processors also test our ability to cross technology generations and retain accuracy. The configurations for the validations are based on published data on the target processors in Sun Microsystems [2013], Leon et al. [2007], Nawathe et al. [2008], Jain et al. [2001], Rusu et al. [2006], Intel [2013], and ARM [2013], including target clock rate, working temperature, and architectural parameters. The target clock cycle time is used as the upper bound for the timing constraint that is used in McPAT to determine the basic circuit properties and must be satisfied before other optimizations and trade-offs can be applied. Therefore, the generated results must match the clock rate of the actual target processor. Because timing (target clock rate) is already considered when computing and optimizing power and area, only power and area validation results are shown in this section.

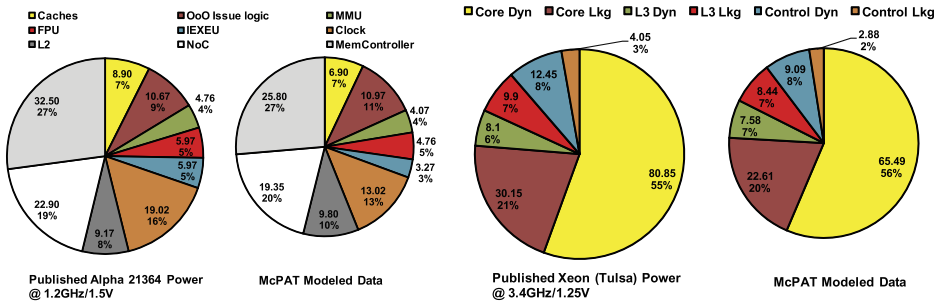
Figures 5 and 6 show the detailed validation results on power[4]. Unfortunately, the best power numbers we have for these processors are for peak power rather than average power. Fortunately, McPAT can also output peak power numbers based on maximum activity factors and maximum switching activity. These results show that

[4]Since The Intel Atom processor and ARM Cortext A9 processor do not have publicly available component-level power breakdowns, we can only validate at whole processor level for the two processors, with results listed in Table III.



(a) Validation against the Niagara processor. (b) Validation against the Niagara2 processor.

Fig. 5. McPAT validation against Niagara processors. The numbers in all charts are the reported and modeled power numbers of the components. The percentages denote the ratios of the component power to the total power. There are miscellaneous components such as SoC logic and I/O that McPAT does not model in detail because their structures are unknown. Therefore, 61.4W out of the total 63W are modeled for Niagara, and 77.9W out of the total 84W are modeled for Niagara2.



(a) Validation against the Alpha 21364 processor. (b) Validation against the Xeon Tulsa processor.

Fig. 6. McPAT validation against OOO processors. 119.8W out of the total 125W are modeled for the Alpha 21364, and 145.5W out of the total 150W are modeled for the Xeon Tulsa.

modeled power numbers track the published numbers well. For the Niagara processor, the absolute power numbers for cores and crossbars generated by McPAT match very well with the published data. Over all eight components, the average difference in absolute power between the modeled power numbers and published Niagara data is just 1.15 W, for an average error per component of 19%. That number seems high, but the two big contributors are clock power (60% error, but a small contributor to total power) and leakage power (23% error). Both are significantly more accurate for the Niagara2. For Niagara2, the average error is 1.1W (16%). If we were measuring average power instead of peak power, this difference would shrink given the expected activity factors of those components. The modeled power of the OOO issue logic, a key component of the OOO core in the Alpha 21364 processor, is very close to the reported power with only a 2.78% difference. Although there are no detailed power breakdowns of both the core and uncore parts of Xeon Tulsa, the modeled bulk power of core and uncore comes close to the reported data, with a -20.63% and -11% error, respectively. The validating process against the ARM Cortex A9 and Intel ATOM processors demonstrates the accuracy of our models for embedded-class cores. The differences between the total peak power generated by McPAT and reported data are under 5%, and the area differences between the die area generated and reported data from industry are all under 5% as well.

Table III shows the comparison of total power and area for validations against the target processors. Differences between the total peak power generated by McPAT and

Table III. Validation Results of McPAT with regard to Total Power and Area of Target Processors

Processor	Published total Power and Area	McPAT Results	% McPAT error
Niagara	63 W / 378 mm^2	59.4 W / 298 mm^2	−5.7 / −20.8
Niagara2	84 W / 342 mm^2	81.2 W / 277 mm^2	−3.3 / −19.0
Alpha 21364	125 W / 396 mm^2	97.9 W / 324 mm^2	−21.68 / −18.2
Xeon Tulsa	150 W / 435 mm^2	116.08 W / 362 mm^2	−22.61 / −16.7
Atom Diamondville	8 W / 51.92 mm^2	7.74 W / 49.8 mm^2	−3.2 / −4.1
Cortex A9 Hard IP	1.9 W / 6.7 mm^2	1.86 W / 6.52 mm^2	−2.3 / −2.7

reported data are 5.7%, 5.3%, 21.68%, 22.61%, 3.2%, and 2.3% for the Niagara, Niagara2, Alpha 21364, Xeon Tulsa, Cortex A9, and ATOM Diamondville, respectively. It is worth noting that chip-to-chip power variation in recent microprocessor designs [Borkar et al. 2003] is also comparable to the magnitude of the power validation errors reported in Table III. The modeled area numbers also track the published numbers well, as shown in Table III. Given the generic nature of McPAT, we consider these errors on both power and area acceptable.

8. SCALING AND CLUSTERING TRADE-OFFS IN MANYCORE PROCESSORS

We illustrate the utility of McPAT by applying it to scaling and clustering trade-offs in a manycore architecture. We evaluate the following aspects of this architecture: (1) the scaling trends of power, area, and timing of the proposed manycore architecture, and (2) the benefits of organizing cores into clusters with local interconnect. We evaluate this architecture across five technologies—90, 65, 45, 32, and 22nm—which covers years 2004 to 2016 of the ITRS roadmap.

8.1. Experimental Setup

Figure 7 shows the manycore architecture we assume targeting future high-throughput computing. It consists of multiple clusters connected by a 2D-mesh on-chip network. A cluster has one or more multithreaded Niagara-like [Kongetira et al. 2005] cores and a multibanked L2 cache. Each core has up to 4 active threads and 32KB 4-way set-associative L1 instruction and data caches. All cores in a cluster share a multibanked 16-way set-associative L2 cache. The number of L2 banks equals the number of cores per cluster. The size of an L2 bank is 256KB. All caches have 64B cache lines. A crossbar is used to connect cores and L2 cache banks for intra-cluster communications. A two-level hierarchical directory-based MESI protocol is used for cache coherency. Within a cluster, the L2 cache is inclusive and filters coherency traffic between L1 caches and directories. Between clusters, a cache directory is implemented by using directory caches that are associated with the on-chip memory controllers, similar to the implementation in the Alpha 21364 processor. The 2D-mesh networks have a data width of 256 bits. We use minimal dimension-order routing and two virtual channels per physical port. Each virtual channel has a 32-deep flit input buffer. Double-pumped crossbars [Vangal et al. 2005] are used in the routers to reduce the die area. Routers in the networks have a local port that connects the hub of a cluster as well as ports that connect the neighboring routers.

Table IV shows the parameters of the manycore architecture at each technology generation. We start from a conservative 2.0 GHz clock rate at 90nm technology, which is about the average of the Niagara processor and Intel processors fabricated in a 90nm process. The intrinsic speed of high-performance transistors increases by 17% per year according to the ITRS. However, increasing clock frequency at this pace will lead to unmanageable chip power density. Moreover, unlike Intel's approach of

Table IV. Parameters of the Manycore Architecture across Technology Generations

Parameters	90nm	65nm	45nm	32nm	22nm
Clock rate (GHz)	2.0	2.3	2.7	3.0	3.5
The number of cores	4	8	16	32	64
The number of memory controllers	2	3	4	6	8
Memory capacity per channel (GB)	2	4	4	8	8
Main memory type	DDR2-667	DDR3-800	DDR3-1066	DDR3-1333	DDR3-1600

Each memory controller has one memory channel.

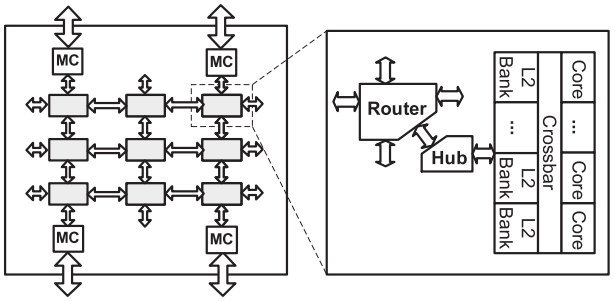


Fig. 7. The manycore system architecture. MCs refer to memory controllers.

changing microarchitectures of their processors during technology scaling, we increase the number of cores and memory controllers aggressively, while keeping the same microarchitecture for new generations. Therefore, we increase the clock frequency conservatively by around 15% every generation. We also start from a conservative die size of around 200mm^2 at 90nm technology and use McPAT to optimize power, area, and timing. The results show that four cores can be placed within the specified area at 90nm. Then, we double the number of cores for each generation. It is difficult to increase the number of memory controllers and channels linearly with the increased core count because of the limited pin count of the chip [Semiconductor Industries Association 2007]. We assume that the number of memory controllers grows proportionally to the square root of the cluster count since the bandwidth of each controller also increases over time. The memory channels are shared by all clusters through the on-chip network and placed at the edge of the chip to minimize the routing overhead as shown in Figure 7. As shown in Table IV, we also scale the bandwidth of main memory based on the expected availability of major DIMM products at each technology node.

We develop a manycore simulation infrastructure where a timing simulator and a functional simulator are decoupled, as in GEMS [Martin et al. 2005]. We modify a user-level thread library [Pan et al. 2005] in the Pin [Luk et al. 2005] binary instrumentation tool to support more pthread APIs, and used it as a functional simulator to run applications. In-order cores, caches, directories, on-chip networks, and memory channels are modeled in an event-driven timing simulator, which controls the execution flow of a program running in the functional simulator and effectively operates as a thread scheduler.

SPLASH-2 [Woo et al. 1995], PARSEC [Bienia et al. 2008], and SPEC CPU2006 [Henning 2007] benchmark suites are used for experiments. The number of threads spawned in a multithreaded workload is the same as the number of hardware threads so each thread is statically mapped to a hardware thread. We use all SPLASH-2 applications and five of the PARSEC applications. The simlarge dataset is used for PARSEC applications while the datasets used for SPLASH-2 applications are summarized in Table V. For each application, the same dataset is used throughout

Table V. SPLASH-2 and SPECCPU 2006 Benchmarks

(a) SPLASH-2 datasets

SPLASH-2

Application	Dataset	Application	Dataset
Barnes	16K particles	Cholesky	tk17.O
FFT	1024K points	Radiosity	room
FMM	16K particles	Raytrace	car
LU	512×512 matrix	Volrend	head
Ocean	258×258 grids		
Radix	8M integers		
Water-Sp	4K molecules		

(b) SPECCPU 2006 application mixes for high, med, and low memory bandwidth

SPEC CPU2006

Set	Applications
CINT	
high	429.mcf, 462.libquantum, 471.omnetpp, 473.astar
med	403.gcc, 445.gobmk, 464.h264ref, 483.xalancbmk
low	400.perlbench, 401.bzip2, 456.hmmer, 458.sjeng
CFP	
high	433.milc, 450.soplex, 459.GemsFDTD, 470.lbm
med	410.bwaves, 434.zeusmp, 437.leslie3d, 481.wrf
low	436.cactusADM, 447.dealII, 454.calculix, 482.sphinx3

Table VI. Area and Maximum Power of Configurations with 4 Cores per Cluster across Technology Generations

	90nm	65nm	45nm	32nm	22nm
Core area (mm^2)	81.9	96.4	113.4	133.5	157.1
Uncore area (mm^2)	104.3	111.3	102.7	101.6	93.5
Die area (mm^2)	186.3	207.7	216.2	235.1	250.6
Max core dynamic power (W)	24.1	30.7	41.7	48.3	56.4
Max uncore dynamic power (W)	20.6	36.1	45.9	54.5	61.8
Total subthreshold leakage (W)	6.5	11.2	17.6	21.5	25.8
Total gate leakage (W)	2.6	6.7	0.7	1.6	2.5
Chip max power (W)	53.8	84.8	106.0	125.9	146.7

all process generations. We use the SPEC CPU2006 benchmark suite to measure the system performance on consolidated workloads. The benchmark suite consists of integer (CINT) and floating-point (CFP) benchmarks, all of which are single threaded. We pick 12 applications from both CINT and CFP, and make three groups each, four applications per group, based on their main-memory bandwidth demand [Henning 2007] as shown in Table V. We find the representative simulation phases of each application and their weights using Simpoint 3.0 [Sherwood et al. 2002]. Each hardware thread runs a simulation phase and the number of instances per phase is proportional to its weight. We skip the initialization phases of each workload and simulate 2 billion instructions unless it finishes earlier.

8.2. Overview of Area and Power

Table VI shows the area and maximum power of the proposed architecture with four cores per cluster across five technology generations. Core area varies from 43% to 62% of total die size across technologies. Core area scales worse than uncore area since uncore components, especially L2 caches, crossbars, and routers, are more regular and easier to scale across generations than cores. According to McPAT's area modeling results, the double-pumped crossbars reduce the area of intra-cluster crossbars and 2D-mesh routers by 54.1% and 35.6% respectively, compared to the single-pumped crossbar implementations. Although we save significant area on interconnects within and between clusters, uncore components still occupy a large portion of total die area.

Short-circuit power is around 10% of the total dynamic power, with fluctuations within 3.1% across all the technology generations. The main reason for the stable short-circuit power is that we use ITRS technology models that have stable V_{th} to

Table VII. NoC Configuration and Die Size of Different Clustering Configurations of the Manycore Architecture at 22nm

Chip Config	NoC Config	Die size (mm^2)	Chip Config	NoC Config	Die size (mm^2)
1 core per cluster	8×8	239.1	**2 cores per cluster**	4×8	246.3
4 cores per cluster	4×4	250.6	**8 cores per cluster**	2×4	278.6

The NoC bisection bandwidth is kept constant across configurations.

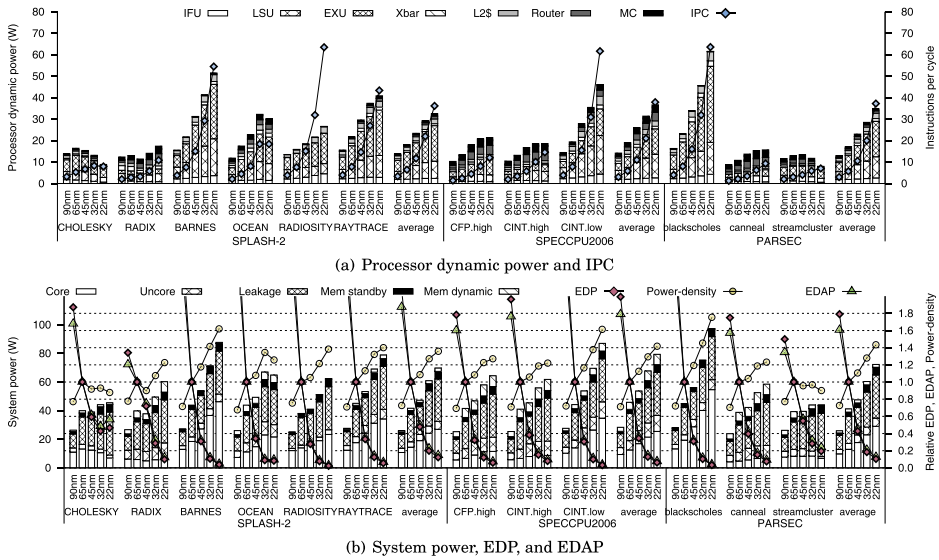
V_{dd} ratios. Cores burn about half of the total maximum dynamic power across generations. Gate leakage is an important component in 90nm and 65nm technologies, being 37.6% of the total leakage power in the 65nm technology. Hi-k metal gate transistors [Auth et al. 2008] are introduced at 45nm, which reduces the gate leakage by more than 90%. SOI technology and Double-Gate (DG) devices that are used at 32nm and 22nm technologies also help to keep the subthreshold leakage under control.

Table VII shows the die size and NoC configurations of various manycore architectures when the number of cores per cluster is varied from 1 to 8 at the 22nm technology node. Since the total number of cores is fixed at 64, the NoC size decreases as the number of cores per cluster increases. We keep the same NoC bisection bandwidth on all configurations. The 1 core per cluster design is the smallest in die size because it does not need crossbars between cores and L2 caches. Even though it needs more routers to connect clusters, the silicon area of each router is smaller since we keep the same bisection bandwidth. When the mesh network is not square, its bisection bandwidth is limited by a cut through its smaller dimension. So we need the same link width for 8×4 and 4×4 networks. That is why the die size of the 2 core per cluster design is very close to the die size of the 4 core per cluster design, even though the latter has much bigger crossbars in the clusters. The 8 core per cluster design shows the largest die size because it needs both the intra- and inter-cluster interconnects. Moreover, keeping the bisection bandwidth of the 2×4 NoC the same as other configurations requires the use of larger routers and links, which further increases the whole chip die size.

8.3. Performance and Efficiency Trade-Offs in Technology Scaling and Clustering

Because McPAT provides an integrated power, area, and timing model, when combined with performance data, chip multiprocessors can be analyzed using several metrics previously unavailable in architecture studies. We think Energy-Delay-Area^2 Product (EDA^2P) and Energy-Delay-Area Product (EDAP) are particularly interesting metrics. These metrics include both an operational cost component (energy) as well as a capital cost component (area). Although the die yield is proportional to the fourth power of the area [Rabaey et al. 2003], in practice due to good die yield and when combined with die per wafer, die cost is roughly proportional to the square of the area [Hennessy and Patterson 2011]. So when designing and manufacturing a chip multiprocessor, we believe EDA^2P is a good way of including chip cost in the optimization process. However, other fixed system costs such as memory and I/O reduce the overall system cost dependence on chip multiprocessor cost. Thus we believe EDAP could be a more useful metric for chip multiprocessors at system level than EDA^2P. Hence, a chip vendor may favor EDA^2P, while a system vendor could prefer EDAP. Finally, given McPAT's area models, another interesting metric is power density: chip power divided by area. Cooling a microprocessor becomes substantially more difficult as power density increases, and this can add significant capital cost for more advanced packaging.

Figure 8 shows power, Instructions Per Cycle (IPC), system Energy-Delay Product (EDP), and EDAP of the five system configurations where the technology nodes are changed from 90nm to 22nm. These are all 4-core/cluster configurations. Applications are grouped by three benchmark suites, and in each group there are applications which



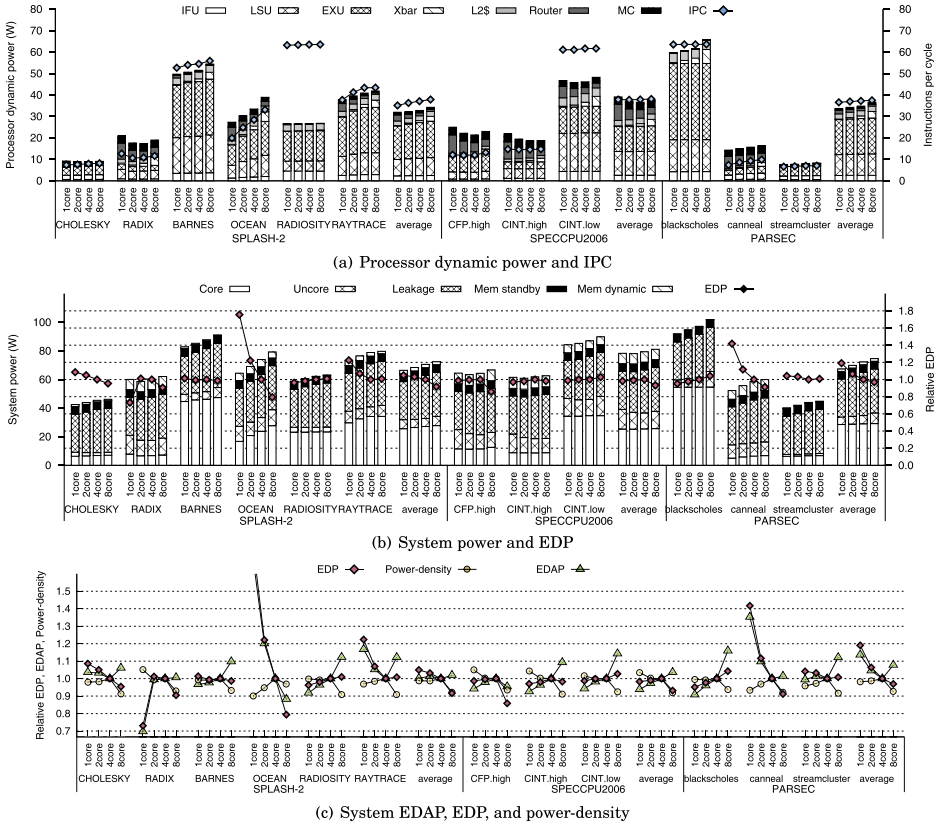
(a) Processor dynamic power and IPC

(b) System power, EDP, and EDAP

Fig. 8. Power, power density, IPC, EDP, and EDAP of the manycore (with Niagara-like cores) systems while the technology nodes are changed from 90nm to 22nm. For each suite, there are applications which are not listed due to space limitations, but they are included when average values are computed.

are not listed due to space limitations, but they are included when average values are computed. Figure 9(a) shows the IPC and the dynamic power of five configurations. Applications such as RADIOSITY, CINT.low, and blackscholes are not limited by main-memory bandwidth and scale close to linearly with the number of cores in the system. The IPCs of RADIX, CFP.high, CINT.high, and canneal improve relatively slowly since they are limited by main-memory bandwidth, which scales worse than computation power. CHOLESKY and OCEAN are the applications with limited IPC scaling because of the insufficient parallelism within applications or small datasets, which in turn leads to the decrease of power density over technology generations. Figure 8(b) shows the system power breakdown, the power density, the EDP, and the EDAP of five configurations. The power density, the EDP, and the EDAP values are normalized to the values of the 65nm configuration. Many of the applications have inflection points in power density at 65nm because gate leakage is reduced by more than 90% when moving from 65nm to 45nm. On many applications, the EDP values improve rapidly as process technology improves. But on CHOLESKY and OCEAN the EDP stops improving after the 32nm process due to limited IPC scaling. The EDAP values scale in a similar way to the EDP values, but change less since the die area increases as shown in Table VI.

In Figure 9, the number of cores per cluster is varied from 1 to 8 on the 22nm technology node. The layout of Figure 9 is the same as that of Figure 8, but here the power density, the EDP, and the EDAP values are normalized to the 4-core per cluster configuration. Figure 9(a) shows that throughout the applications, the intra-cluster crossbar power (Xbar) increases and inter-cluster power decreases as the number of cores per cluster increases. This is expected because the size and power of a crossbar scales superlinearly with the number of cores per cluster. The effect of this clustering on IPC depends heavily on the applications. As more cores are grouped into a cluster, the size of the L2 cache that a core can access increases even though more cores share the multibanked cache, so if multiple cores in a cluster share data (cores use a shared L2 cache synergistically), then an L2 cache can retain more of the combined working



(a) Processor dynamic power and IPC

(b) System power and EDP

(c) System EDAP, EDP, and power-density

Fig. 9. Power, power density, IPC, EDP, and EDAP of the manycore (with Niagara-like cores) systems while the number of cores per cluster is changed from 1 to 8 on the 22nm technology node. For each suite, there are applications which are not listed due to space limitations, but they are included when average values are computed.

sets, hence lowering L2 misses. On OCEAN, RAYTRACE, and canneal, the IPC increases noticeably because of this synergistic cache sharing effect. As a result, the EDP of these applications improves as the number of cores per cluster increases, as shown in Figure 9(b). In contrast, there are applications like RADIOSITY, CINT.low, and blackscholes whose performance is not affected by cache sharing. On these applications, the EDP gets worse as more cores are grouped since the intra-cluster crossbar power increases.

On average, clustering more cores together improves the system energy-delay product. However, if we take the area of the processors into account, the benefits of cache sharing are negated, especially when the number of cores per cluster is 8. In that configuration, the system energy-delay-area product is worse than the configuration with 4 cores per cluster on all benchmark suites on average (Figure 9(c)). RADIX and CFP.high are two applications showing interesting performance characteristics, which in turn are reflected in the EDP. On RADIX, when the number of cores per cluster changes from 1 to 2, the L2 miss percentage increases noticeably because cache lines in the L2 caches are evicted more frequently, meaning that cores in a cluster interfere with each other. On CFP.high, cache sharing does not affect performance when the number of cores per cluster is changed from 1 to 4, but the L2 miss rate drops considerably as 8 cores are grouped into a cluster. This is because with fewer cores

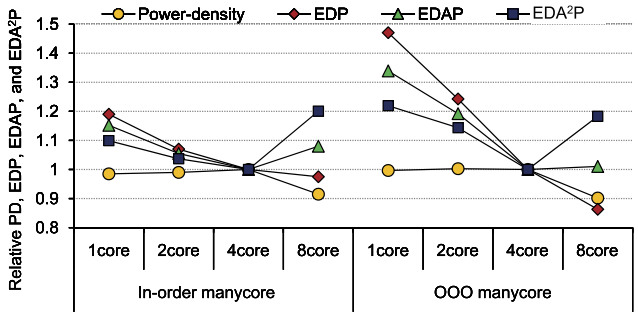


Fig. 10. Averaged power density, EDP, EDAP, and EDA^{2}P of both in-order and OOO manycore architectures at the 22nm technology node running PARSEC benchmarks. The total numbers of cores/threads are 64/256 and 16/16 for in-order and OOO processors, respectively. The number of cores per cluster is changed from 1 to 8 for both in-order and OOO processors.

per cluster, there are noticeable miss rate fluctuations between L2 caches, which are mostly attenuated when 8 cores share an L2 cache.

By using McPAT together with M5 [Binkert et al. 2006], another cycle-accurate performance simulator, we also studied clustering trade-offs of manycore processors with 16 OOO cores at 22nm running at 3.5 GHz. Each OOO core is similar to the single-threaded four-issue Alpha 21264 processor but with 32KB 4-way set-associative instruction and data caches. The total capacity of on-chip L2 caches of the OOO manycore is the same as the in-order manycore. The L2 caches are shared within a cluster and are coherent among different clusters. The OOO manycore architecture (with fewer, larger cores) can provide similar peak performance and occupy similar die area to the previously described in-order manycore architecture. Specifically, both architectures' ideal IPCs are 64, and the area of the OOO manycore is about 3.5% larger than the in-order manycore when using the same number of cores per cluster. The same subset of the PARSEC suite benchmarks is used for the OOO simulations.

Figure 10 shows that clustering trade-offs with OOO cores have similar, but magnified, trends as when using in-order cores. For both the in-order and OOO cores running PARSEC benchmarks, if the manycore die cost is not taken into account 8-core clusters provide the best EDP. However, 4-core clusters are best for both in-order and OOO cores when the manycore die cost is taken into account by using EDA^{2}P and EDAP.

9. CONCLUSIONS AND FUTURE WORK

McPAT is the first tool to integrate power, area, and timing models for a complete chip multiprocessor, including cores and uncore components. McPAT's power models account for dynamic, subthreshold leakage, gate leakage, and short-circuit power. It is also the first processor power modeling environment to support clock gating and power gating. McPAT uses MASTAR [Semiconductor Industries Association 2007] to calculate device models based on the ITRS roadmap [Semiconductor Industries Association 2007], giving it the scalability to evolve together with technology advancement. McPAT's internal optimizer can find power- and/or area-optimal configurations for array structures and interconnects, given specified timing constraints and optimization targets. Rigorous validation shows good agreement between McPAT's predictions and published data for a variety of processors, ranging from high-performance out-of-order multithreaded chip multiprocessors to embedded in-order single-threaded processors. Unlike prior tools that were tightly integrated with specific performance simulators, McPAT uses an XML interface to decouple the architectural simulator from the power, area, and timing analysis, so that it can be readily used with a variety of

simulators. Thanks to the contributions from the McPAT user community in interfacing McPAT with performance simulators, McPAT currently works with almost all major performance simulators including M5 [Binkert et al. 2006], GEMS [Martin et al. 2005], GEM5 [Binkert et al. 2011], MacSim [Gerard and of Engineering Physics 1997], Graphite [Miller et al. 2010], SST [Rodrigues et al. 2011], and Multi2Sim [Ubal et al. 2007]. McPAT is an active ongoing project. Future work includes modeling emerging new technologies such as 3D stacking, nonvolatile memory technologies, and GPGPU architecture modeling. New updates together with related papers and documents will be available at its Web site: `http://www.hpl.hp.com/research/mcpat/`.

By providing these capabilities, McPAT supports architects in exploring a broad design space for future multicore and manycore systems. Furthermore, metrics such as Energy-Delay-Area^2 Product (EDA^{2}P) and Energy-Delay-Area Product (EDAP) that include die cost can now be used. By combining the power, area, and timing of McPAT capabilities with performance simulation, we explore the interconnect options of future manycore processors by varying the degree of core clustering over several generations of process technologies. At the 22nm technology node when running PARSEC benchmarks for manycores built from both in-order and out-of-order cores, we found that when area cost is not taken into account, clusters of 8 cores provide the best EDP, but when area cost is included clusters of 4 cores provide the best EDA^{2}P and EDAP. We believe McPAT will be essential for manycore architectural insights in the future.

REFERENCES

Ahn, H.-T. and Allstot, D. 2000. A low-jitter 1.9-v cmos pll for ultrasparc microprocessor applications. *J. Solid State Circ. 35,* 3, 450–454.

AMD. 2002. HyperTransport technology: Simplifying system design. http://www.hypertransport.org/docs/wp/26635A_HT_System_Design.pdf.

Anis, M., Areibi, S., Mahmoud, M., and Elmasry, M. 2002. Dynamic and leakage power reduction in mtcmos circuits using an automated efficient gate clustering technique. In *Proceedings of the 39th Design Automation Conference (DAC)*. 480–485.

ARM. 2013. http://www.arm.com/products/processors/cortex-a/cortex-a9.php.

Auth, C., Buehler, M., Cappellani, A., Hing Choi, C., Ding, G., Han, W., Joshi, S., Mcintyre, B., Prince, M., Ranade, P., Sandford, J., and Thomas, C. 2008. 45nm high-k+metal gate strain-ehanced transistors. *Intel Technol. J. 12*.

Bienia, C., Kumar, S., Singh, J. P., and Li, K. 2008. The parsec benchmark suite: Characterization and architectural implications. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT).*

Binkert, N., Beckmann, B., Black, G., Reinhardt, S. K., Saidi, A., Basu, A., Hestness, J., Hower, D. R., Krishna, T., Sardashti, S., Sen, R., Sewell, K., Shoaib, M., Vaish, N., Hill, M. D., and Wood, D. A. 2011. The gem5 simulator. *SIGARCH Comput. Archit. News 39,* 2.

Binkert, N. L., Dreslinski, R. G., Hsu, L. R., Lim, K. T., Saidi, A. G., and Reinhardt, S. K. 2006. The m5 simulator: Modeling networked systems. *IEEE Micro 26,* 4, 52–60.

Bishop, B., Kelliher, T. P., and Irwin, M. J. 1999. The Design of a register renaming unit. In *Proceedings of the 9th Great Lakes Symposium on VLSI.*

Borkar, S., Karnik, T., Narendra, S., Tschanz, J., Keshavarzi, A., and De, V. 2003. Parameter variations and impact on circuits and microarchitecture. In *Proceedings of the Design Automation Conference (DAC).*

Broadcom. 2008. BCM57710 - dual-port 10G/2500/1000base-x toe, rdma, isci pci express ether-net controller. http://www.broadcom.com/products/Ethernet-Controllers-and-Adapters/Enterprise-Server-Controllers/BCM57710.

Brooks, D., Tiwari, V., and Martonosi, M. 2000. Wattch: A framework for architectural-level power analysis and optimizations. In *Proceedings of the 27th Annual International Symposium on Computer Architecture (ISCA).*

Brooks, D., Bose, P., Srinivasan, V., Gschwind, M., Emma, P., and Rosenfield, M. 2003. New methodology for early-stage, microarchitecture-level power-performance analysis of microprocessors. *IBM J. Res. Devel. 47*.

Burger, D. and Austin, T. M. 1997. The simplescalar tool set. version 2.0. *SIGARCH Comput. Archit. News 25,* 3.

Butler, M. 2010. Bulldozer: A new approach to multithreaded compute performance. In *Proceedings of Hot Chips: A Symposium on High Performance Chips*.

Cadence Incyte Chip Estimator. 2013. http://www.chipestimate.com/.

Cai, G. Z. N. and Lim, C. H. 2012. Microarchitectural power analysis for cpu power/performance optimization. http://web.eecs.umich.edu/~panalyzer/pdfs/Microacrhitectural_Power_Analysis_for_CPU_Power_Performance_Optimization.pdf.

Elmore, W. C. 1948. The transient response of damped linear netwoks with particular regard to wideband amplifiers. *J. Appl. Phys. 19*, 55–63.

Ferdman, M., Lotfi-Kamran, P., Balet, K., and Falsafi, B. 2011. Cuckoo directory: A scalable directory for many-core systems. In *Proceedings of the IEEE 17th International Symposium on High Performance Computer Architecture (HPCA)*. 169–180.

Fukuda, K., Yamashita, H., Ono, G., Nemoto, R., Suzuki, E., Takemoto, T., Yuki, F., and Saito, T. 2010. A 12.3mW 12.5Gb/s complete transceiver in 65nm cmos. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC'10)*. 368–369.

George, V., Jahagirdar, S., Tong, C., Smits, K., Damaraju, S., Siers, S., Naydenov, V., Khond-Ker, T., Sarkar, S., and Singh, P. 2007. Penryn: 45-nm next generation intel core 2 processor. In *Proceedings of the IEEE Asian Solid-State Circuits Conference (ASSCC'07)*.

Gerard, P. and of Engineering Physics, M. U. D. 1997. *Macsim: Simulating the Mcmaster Nuclear Reactor Using a Distributed Approach*. McMaster University.

Gowan, M. K., Biro, L. L., and Jackson, D. B. 1998. Power considerations in the design of the alpha 21264 microprocessor. In *Proceedings of the Design Automation Conference (DAC)*.

Gunther, S. H., Binns, F., Carmean, M., and Hall, J. C. 2001. Managing the impact of increasing microprocessor power consumption. *Intel Technol. J. 1.*

Gupta, S., Keckler, S., and Burger, D. 2000. Technology independent area and delay estimates for microprocessor building blocks. Tech. rep., Department of Computer Science, University of Texas at Austin.

Harwood, M., Warke, N., Simpson, R., Leslie, T., Amerasekera, A., Batty, S., Colman, D., Carr, E., Gopinathan, V., Hubbins, S., et al. 2007. A 12.5Gb/s SerDes in 65nm cmos using a baud-rate adc with digital receiver equalization and clock recovery. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC'07)*. 436–591.

Hennessy, J. L. and Patterson, D. A. 2011. *Computer Architecture: A Quantitative Approach* 5th Ed. Morgan Kaufmann, San Fransisco, CA.

Henning, J. L. 2007. Performance counters and development of spec cpu2006. *Comput. Archit. News 35,* 1.

Hinton, G., Sager, D., Upton, M., Boggs, D., Group, D. P., and Corp, I. 2001. The microarchitecture of the pentium 4 processor. *Intel Technol. J. 1.*

Ho, R. 2003. On-Chip wires: Scaling and efficiency. Ph.D. thesis, Stanford University.

Horowitz, M. A. 1984. Timing models for mos circuits. Tech. rep., Stanford University.

Horowitz, M., Ho, R., and Mai, K. 1999. The future of wires. http://velox.stanford.edu/.

HP. 2009. HP BladeSystem c-class SAN connectivity technology brief. http://h20000.www2.hp.com/bc/docs/support/SupportManual/c00810839/c00810839.pdf.

Huang, W., Member, S., Ghosh, S., Velusamy, S., Sankaranarayanan, K., Skadron, K., Stan, M. R., Member, S., and Member, S. 2006. Hotspot: A compact thermal modeling method for cmos vlsi systems. *IEEE Trans. VLSI 14*, 501–513.

Intel. 1998. P6 family of processors hardware developer's manual. Intel white paper.

Intel. 2009. An introduction to the intel quickpath interconnect. http://www.intel.com/content/dam/doc/white-paper/quick-path-interconnect-introduction-paper.pdf.

Intel. 2013. http://www.intel.com/products/processor/atom/techdocs.htms.

Jain, A., Anderson, W., Benninghoff, T., Bertucci, D., Braganza, M., Burnette, J., Chang, T., Eble, J., Faber, R., Gowda, D., Grodstein, J., Hess, G., Kowaleski, J., Kumar, A., Miller, B., Mueller, R., Paul, P., Pickholtz, J., Russell, S., Shen, M., Truex, T., Vardharajan, A., Xanthopoulos, D., and Zou, T. 2001. A 1.2 ghz alpha microprocessor with 44.8 GB/s chip pin bandwidth. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC)*.

Johnson, T. and Nawathe, U. 2007. An 8-core, 64-thread, 64-bit power efficient sparc soc (niagara2). In *Proceedings of the International Symposium on Physical Design (ISPD)*.

Kahng, A., Li, B., Peh, L.-S., and Samadi, K. 2009. ORION 2.0: A fast and accurate noc power and area model for early-stage design space exploration. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*.

Kao, J., Narendra, S., and Chandrakasan, A. 1998. MTCMOS hierarchical sizing based on mutual exclusive discharge patterns. In *Proceedings of the Design Automation Conference (DAC)*. 495–500.

Kessler, R. E. 1999. The alpha 21264 microprocessor. *IEEE Micro 19,* 2.

Kongetira, P., Aingaran, K., and Olukotun, K. 2005. Niagara: A 32-way multithreaded sparc processor. *IEEE Micro 25,* 2.

Koufaty, D. and Marr, D. T. 2003. Hyperthreading technology in the netburst microarchitecture. *IEEE Micro 23,* 2.

Kumar, R. and Hinton, G. 2009. A family of 45nm ia processors. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC).* 58–59.

Kumar, R., Jouppi, N. P., and Tullsen, D. M. 2004. Conjoined-Core chip multiprocessing. In *Proceedings of the Annual IEEE/ACM International Symposium on Microarchitecture (MICRO).* 195–206.

Kumar, R., Zyuban, V., and Tullsen, D. M. 2005. Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Leon, A. S., Tam, K. W., Shin, J. L., Weisner, D., and Schumacher, F. 2007. A power-efficient high-throughput 32-thread sparc processor. *J. Solid State Circ. 42*.

Li, S., Ahn, J. H., Strong, R. D., Brockman, J. B., Tullsen, D. M., and Jouppi, N. P. 2009. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proceedings of the Annual IEEE/ACM International Symposium on Microarchitecture (MICRO).* 469–480.

Li, S., Chen, K., Ahn, J. H., Brockman, J. B., and Jouppi, N. P. 2011. CACTI-P: Architecture-Level modeling for sram-based structures with advanced leakage reduction techniques. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD).*

Long, C. and He, L. 2004. Distributed sleep transistor network for power reduction. *IEEE Trans. Very Large Scale Integr. Syst. 12*, 937–946.

Luk, C.-K., Cohn, R., Muth, R., Patil, H., Klauser, A., Lowney, G., Wallace, S., Reddi, V. J., and Hazelwood, K. 2005. Pin: Building customized program analysis tools with dynamic instrumentation. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI).*

Mahoney, P., Fetzer, E., Doyle, B., and Naffziger, S. 2005. Clock distribution on a dual-core multi-threaded itanium family processor. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC).*

Martin, M. M. K., Sorin, D. J., Beckmann, B. M., Marty, M. R., Xu, M., Alameldeen, A. R., Moore, K. E., Hill, M. D., and Wood, D. A. 2005. Multifacet's general execution-driven multiprocessor simulator (gems) toolset. *SIGARCH Comput. Archit. News 33,* 4.

Marty, M. R. and Hill, M. D. 2007. Virtual hierarchies to support server consolidation. *SIGARCH Comput. Archit. News 35,* 2, 46–56.

Mathew, S., Anders, M., Bloechel, B., Nguyen, T., Krishnamurthy, R., and Borkar, S. 2005. A 4-GHz 300-mW 64-bit integer execution alu with dual supply voltages in 90-nm cmos. *J. Solid State Circ. 40,* 1.

Miller, J. E., Kasture, H., Kurian, G., Iii, C. G., Beckmann, N., Celio, C., Eastep, J., and Agarwal, A. 2010. Graphite: A distributed parallel simulator for multicores. In *Proceedings of the International Symposium on High-Performance Computer Architecture (HPCA)*. 1–12.

Naveh, A., Rotem, E., Mendelson, A., Gochman, S., Chabukswar, R., Krishnan, K., and Kumar, A. 2006. Power and thermal management in the intel core duo processor. *Intel Technol. J. 10*, 109–122.

Nawathe, U., Hassan, M., Yen, K., Kumar, A., Ramachandran, A., and Greenhill, D. 2008. Implementation of an 8-core, 64-thread, power-efficient sparc server on a chip. *J. Solid State Circ. 43,* 1.

Nose, K. and Sakurai, T. 2000. Analysis and future trend of short-circuit power. *IEEE Trans. Comput.-Aided Des. 19*, 9.

Palacharla, S., Jouppi, N. P., and Smith, J. E. 1997. Complexity-Effective superscalar processors. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Palmer, R., Poulton, J., Dally, W., Eyles, J., Fuller, A., Greer, T., Horowitz, M., Kellam, M., Quan, F., and Zarkeshvari, F. 2007. A 14mW 6.25Gb/s transceiver in 90nm cmos for serial chip-to-chip communications. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC'07)*. 440–614.

Pan, H., Asanovi'c, K., Cohn, R., and Luk, C.-K. 2005. Controlling program execution through binary instrumentation. *Comput. Archit. News 33*, 5.

Rabaey, J., Chandrakasan, A., and Nikolic, B. 2003. *Digital Integrated Circuits: A Design Perspective* 2nd Ed. Prentice-Hall, Englewood Cliffs, NJ.

Ram, K. K., Santos, J. R., Turner, Y., Cox, A. L., and Rixner, S. 2009. Achieving 10 Gb/s using safe and transparent network interface virtualization. In *Proceedings of the ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE).* 61–70.

Rodrigues, A. F. 2007. Parametric sizing for processors. Tech. rep., Sandia National Laboratories.

Rodrigues, A. F., Hemmert, K. S., Barrett, B. W., Kersey, C., Oldfield, R., Weston, M., Risen, R., Cook, J., Rosenfeld, P., Cooperballs, E., and Jacob, B. 2011. The structural simulation toolkit. *SIGMETRICS Perform. Eval. Rev. 38,* 4.

Rusu, S., Tam, S., Muljono, H., Ayers, D., and Chang, J. 2006. A dual-core multi-threaded xeon processor with 16mb l3 cache. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC).*

Semiconductor Industries Association. 2007. Model for assessment of cmos technologies and roadmaps (mastar). http://www.itrs.net/models.html.

Sherwood, T., Perelman, E., Hamerly, G., and Calder, B. 2002. Automatically characterizing large scale program behavior. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS).*

Shin, J., Tam, K., Huang, D., Petrick, B., Pham, H., Hwang, C., Li, H., Smith, A., Johnson, T., Schumacher, F., Greenhill, D., Leon, A., and Strong, A. 2010. A 40nm 16-core 128-thread cmt sparc soc processor. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC).* 98–99.

Sima, D. 2000. The design space of register renaming techniques. *IEEE Micro 20*, 5, 70–83.

Sun Microsystems. 2013. OpenSPARC. http://www.opensparc.net.

Thoziyoor, S., Ahn, J., Monchiero, M., Brockman, J., and Jouppi, N. 2008. A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Tullsen, D. M., Eggers, S. J., Emer, J. S., Levy, H. M., Lo, J. L., and Stamm, R. L. 1996. Exploiting choice: Instruction fetch and issue on an implementable simultaneous multithreading processor. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Ubal, R., Sahuquillo, J., Petit, S., and Lpez, P. 2007. Multi2Sim: A simulation framework to evaluate multicore-multithreaded processors. In *Proceedings of the International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD).* 62–68.

Vangal, S., Borkar, N., and Alvandpour, A. 2005. A six-port 57gb/s double-pumped nonblocking router core. In *Proceedings of the Symposium on VLSI Circuits (VLSI).*

Vijaykrishnan, N., Kandemir, M., Irwin, M. J., Kim, H. S., and Ye, W. 2000. Energy-Driven integrated hardware-software optimizations using simplepower. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Wang, Y., Bhattacharya, U., Hamzaoglu, F., Kolar, P., Ng, Y.-G., Wei, L., Zhang, Y., Zhang, K., and Bohr, M. 2010. A 4.0 ghz 291 mb voltage-scalable sram design in a 32 nm high-k + metal-gate cmos technology with integrated power management. *IEEE J. Solid-State Circ. 45*, 103–110.

Wilton, S. and Jouppi, N. P. 1994. An enhanced access and cycle time model for on-chip caches. Tech. rep. 93/5, DEC WRL.

Woo, S. C., Ohara, M., Torrie, E., Singh, J. P., and Gupta, A. 1995. The splash-2 programs: Characterization and methodological considerations. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA).*

Yoshida, H., De, K., and Boppana, V. 2004. Accurate pre-layout estimation of standard cell characteristics. In *Proceedings of the Annual Conference on Design Automation (DAC).* ACM Press, NewYork, 208–211.

Yuffe, M., Knoll, E., Mehalel, M., Shor, J., and Kurts, T. 2011. A fully integrated multi-cpu, gpu and memory controller 32nm processor. In *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC).* 264–266.

Received May 2012; revised November 2012; accepted November 2012