Enhancing the Efficiency of Energy-Constrained DVFS Designs

Andrew B. Kahng, *Fellow, IEEE*, Seokhyeong Kang, *Student Member, IEEE*, Rakesh Kumar, *Member, IEEE*, and John Sartori, *Student Member, IEEE*

***Abstract*— The proliferation of embedded systems and mobile devices has created an increasing demand for low-energy hardware. Dynamic voltage and frequency scaling (DVFS) is a popular energy reduction technique that allows a hardware design to reduce average power consumption while still enabling the design to meet a high-performance target when necessary. To conserve energy, many DVFS-based embedded and mobile devices often spend a large fraction of their lifetimes in a low-power mode. However, DVFS designs produced by conventional multimode CAD flows tend to have significant energy overheads when operating outside of the peak performance mode, even when they are operating in a low-power mode. A dedicated core can be added for low-energy operation, but has a high cost in terms of area and leakage. In this paper, we explore the DVFS design space to identify the factors that affect DVFS efficiency. Based on our insights, we propose two design-level techniques to enhance the energy efficiency of DVFS for energy constrained systems. First, we present a context-aware DVFS design flow that considers the intrinsic characteristics of the hardware design, as well as the operating scenario—including the relative amounts of time spent in different modes, the range of performance scalability, and the target efficiency metric—to optimize the design for maximum energy efficiency. We also present a selective replication-based DVFS design methodology that identifies hardware modules for which context-aware multimode design may be inefficient and creates dedicated module replicas for different operating modes for such modules. We show that context-aware design can reduce average power by up to 20% over a conventional multimode design flow. Selective replication can reduce average power by an additional 4%. We also use the generated insights to identify microarchitectural decisions that impact DVFS efficiency. We show that the benefits from the proposed design-level techniques increase when microarchitectural transformations are allowed.**

***Index Terms*— Context-aware design, dynamic voltage and frequency scaling, lifetime energy reduction, low-power design.**

I. Introduction

INCREASING thermal densities, reliability concerns, and the portability of emerging computing systems have made power and energy consumption primary concerns for microelectronic designers [1]. Dynamic voltage and frequency scaling (DVFS) [2] is a popular technique to reduce power and energy in situations where there is diversity in workloads (e.g., CPU-intensive versus memory-intensive, realtime versus nonrealtime, etc.), compute conditions (e.g., wall-powered versus battery-powered, low system utilization versus high system utilization, etc.), or objective functions (e.g., single-thread performance versus throughput, energy versus energy-delay product, etc.) [3]. Diversity in operating conditions is exploited by reducing the supply voltage when performance constraints are relaxed. The effectiveness of DVFS at reducing power consumption is due to the strong dependence of both static and dynamic power on supply voltage.

To conserve energy, many DVFS-based mobile and embedded devices typically spend the majority of their lifetimes operating in a low-power mode. For example, Windows switches a DVFS-capable processor into a low-power mode whenever utilization is low [4]. A conventional multimode design methodology spends resources to optimize the critical paths in all modes and is therefore over-optimized at any single operating mode. Conventional design flows also require the user to specify all constraints at design time and disregard factors that are critical to optimizing for energy efficiency, such as the amount of time spent in each mode.

Given the energy overheads of DVFS designs produced by conventional multimode CAD flows, we explore the DVFS design space to identify the sources of DVFS inefficiency and scenarios in which DVFS designs are typically inefficient. Based on our insights, we propose a context-aware multimode design flow to enhance DVFS efficiency. Our context-aware approach takes into consideration the intrinsic characteristics of a design, the desired range of scalability, the relative amounts of time spent in different operating modes, and the desired energy efficiency metric to select appropriate design constraints and optimize the design for maximum efficiency over multiple modes of operation.

We also propose a selective replication-based methodology that identifies modules for which context-aware multimode designs are inefficient as candidates for replication. Selective replication employs multiple replicas of such modules that have been optimized for different performance targets, such that the appropriate replica is active for a given operating scenario, while the other is turned off. Since replication adds an area overhead, we only suggest replication for modules that are particularly inefficient in terms of scalability and energy efficiency.

Finally, our study of the sources of DVFS inefficiency allows us to identify microarchitectural features that affect DVFS efficiency. Thus, we are able to suggest microarchitectural changes that can improve DVFS efficiency in general or for a particular scenario.

Manuscript received May 23, 2011; revised June 22, 2012; accepted August 31, 2012.

A. B. Kahng is with the Department of Computer Science and Engineering, and the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0404 USA (e-mail: abk@ucsd.edu).

S. Kang is with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0404 USA (e-mail: shkang@vlsicad.ucsd.edu).

R. Kumar and J. Sartori are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: rakeshk@illinois.edu; sartori2@illinois.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2012.2219084

The main contributions of our work are the following.

1) We quantify DVFS inefficiency for different operating scenarios for conventional multimode CAD flows and identify the sources of inefficiency. We observe that the average power of a conventional multimode design may be up to 28% higher than the ideal average power.
2) We propose a context-aware approach to multimode design that considers intrinsic characteristics of hardware and factors such as duty cycle and range of scalability to select energy efficient design constraints. Our context-aware multimode design flow reduces average power by up to 20% with respect to a conventional multimode design flow.
3) We propose a selective replication-based approach to improve the energy efficiency of DVFS in cases where even context-aware design is inefficient. Our selective replication-based methodology reduces average power by up to 25% with respect to a conventionally optimized DVFS design, achieving average power consumption that is within 1% of ideal, on average.
4) We show that optimizing the microarchitecture of a DVFS design has the potential to significantly improve DVFS energy efficiency. We observe that optimizing the microarchitecture reduces average power by up to 18% for a context-aware multimode design flow.

The rest of this paper is organized as follows. Section II presents related work. Section III quantifies and explains the inefficiencies of modern DVFS-based designs. Section IV presents the proposed approaches for maximizing DVFS efficiency. Section V explains the experimental methodology. Section VI presents results and analysis. Analysis includes a study of microarchitectural features that impact DVFS efficiency as well as the benefits from the proposed approaches. Section VII summarizes and concludes this paper.

II. Related Work

A. Multicorner Multimode Design

Recently, EDA manufacturers have included Multicorner Multimode (MCMM) capabilities [5], [6] in their implementation tools. MCMM capabilities allow analysis and optimization of a hardware design for multiple modes, where modes are defined by a set of clocks, supply voltages, timing constraints, and libraries. MCMM is typically geared toward achieving design closure across all modes (e.g., test mode and mission modes) in a single pass, significantly reducing design turnaround times.

In this paper, we show that variants of MCMM can be used for efficient DVFS designs, since MCMM can optimize a design for multiple voltage and frequency modes. However, capabilities must be added to identify the constraints that minimize energy for different duty cycles and ranges of scalability. We use MCMM sign-off in our context-aware multimode design flow as well as our baseline conventional design flow.

B. Heterogeneous CMP

DVFS is not the only technique to target multiple power and performance points. In a heterogeneous chip multiprocessor (CMP) [3], each core can be designed for a different power and performance target, and tasks are scheduled to cores depending on their performance or power requirements. Such designs incur significant overheads in terms of area, verification, complexity, and task scheduling. However, they may improve energy efficiency over DVFS designs by providing entire cores that are optimized for specific performance targets. The overheads associated with heterogeneous CMPs could potentially be reduced by replicating core functionalities at a finer granularity, rather than replicating the entire core (see Section IV-B).

C. Workload-Specific Datapath Customization

Work has also been done to improve energy efficiency by customizing hardware for a specific workload. Work on conservation cores (C-cores) [7] proposes to synthesize application-specific cores to improve energy efficiency at a specific frequency and voltage. Application-specific cores can significantly reduce energy consumption for their target applications. The limitations of application-specific cores include increased design and verification overheads, limited generality beyond applications and inputs for which the cores were synthesized, and significant area overhead for large applications (i.e., applications with multiple hot codes). Recent work suggests that the area overhead of conservation cores may be more acceptable if continued technology scaling results in a utilization wall, necessitating dark silicon [8].

D. Race-to-Halt

For some energy-based metrics and some specific (computationally intensive) workload conditions, an orthogonal approach to voltage scaling such as race-to-halt [9] may be useful for increasing energy efficiency. Race-to-halt proposes that it may be more energy efficient to execute a task as quickly as possible, then switch to a low-power sleep mode, than to operate at the lowest possible frequency and voltage that meets timing constraints. While race-to-halt can provide energy benefits over a naive DVFS approach in some scenarios, it suffers somewhat in generality (e.g., it does not benefit memory-intensive workloads). Also, while race-to-halt can potentially reduce energy, it cannot be used to reduce power (e.g., for thermal or reliability reasons).

Note that DVFS does not preclude the deployment of a race-to-halt strategy. Race-to-halt suggests a task scheduling policy-one that can be adopted by a DVFS processor that must, by its nature, be able to switch between different power and performance modes.

E. Energy-Efficient Processor Optimization

Recent work [10] aims to maximize processor energy efficiency by simultaneously optimizing processor microarchitecture, circuit design, and operating voltage. The proposed approach estimates marginal costs in terms of energy and performance when varying each parameter in the design space, and produces a model of the tradeoff space that is used to locate the energy-optimal (microarchitecture, design, voltage) point. [10] considers that an energy-optimal design depends on multiple factors. However, the work does not necessarily account for the fact that marginal energy and performance costs for one parameter may change depending on the values of the other parameters.

In this paper, we take an orthogonal approach. Demonstrating that the optimal design for a DVFS processor indeed depends on design and microarchitecture optimizations, we

propose techniques for finding the design and microarchitecture that maximize energy efficiency for a dynamically scalable processor. In other words, rather than locating the statically optimal (microarchitecture, design, voltage) point, we aim to find the (microarchitecture, design) combination that optimizes the power-performance tradeoff for voltage.

III. Understanding DVFS Inefficiency

Conventional single-mode CAD flows optimize a design for a single design constraint. Such a design may be inefficient at all operating points except the one for which the hardware was optimized. Consequently, conventional single-mode CAD flows may be inadequate for DVFS-based designs. Our present work points out that conventional multimode methodologies may also be inefficient for DVFS-based designs-especially energy-constrained designs. This is because the primary focus of a conventional multimode design methodology is to ensure timing closure over multiple, fully-constrained operating modes, not to optimize a design over multiple modes for a specific metric such as energy efficiency. As such, there exists no multimode design flow that considers the operating scenario [i.e., both the range of scalability (X) and the duty cycle (R)] during optimization. Range of scalability refers to the ratio of maximum and minimum operating frequencies for the design ($X = f_{\text{hi}}/f_{\text{lo}}$). Duty cycle refers to the fraction of time spent in an operating mode [R_k = (time at f_k)/(total time)]. Likewise, there exists no multimode design flow to optimize a design for minimum energy when the exact optimal constraints (all operating frequencies and voltages) are not known at design time.

To quantify the inefficiency of conventional CAD flows, Table I compares the power consumption and characteristics of several designs, each with different timing constraints.[1] Each design operates at its minimum safe voltage. Table I demonstrates that the design for which power is minimized depends on the dominant operating frequency. At high frequency, dynamic power (CV^2f) dominates total power, and tightly constrained designs that allow more voltage scaling have lower power consumption. These designs do well in scenarios that favor high performance [high duty cycle (R), low range of scalability (X)]. However, these designs also have higher leakage and area due to their topology and cell composition [reduced fanout and increased buffering, higher drive strength cells, more low threshold voltage (LVT) cells, fewer high threshold voltage (HVT) cells]. These characteristics reduce efficiency for scenarios that favor low performance (low R, high X).

As the operating frequency is reduced, leakage power begins to dominate total power, and designs that are optimized to reduce voltage are inefficient due to large leakage overheads. In this regime, designs that favor low performance (low R, high X) do better. Thus, a conventional single-mode design may be inefficient for DVFS, since it suffers from either area and leakage overhead or higher voltage at the point for which it was not optimized. Table I also suggests that even a multimode design may be inefficient if it is duty cycle-agnostic, because energy efficiency requires that the design be constrained according to its dominant operating mode.

Fig. 1 shows the single-constraint netlist that consumes minimum energy for a given operating scenario, confirming that the minimum-energy netlist indeed depends on the amount

[1]In total, six netlists have been implemented at 1.0 V with different setup timing constraints: $\text{NET}_i = 1.00\text{ ns} + i * 0.05\text{ ns}$.

TABLE I
IMPLEMENTATION STATISTICS FOR THE OPENSPARC MULTIPLIER

Case	Area (μm^2)	Total Power @ 1 GHz	Total Power @ 100 MHz	% of LVT Cells	Avg. Fanout	Avg. Cell Drive Strength
NET_0	60509	37.4 mW	0.923 mW	28%	2.26	1.30
NET_1	59277	38.1 mW	0.882 mW	19%	2.28	1.29
NET_2	55412	38.2 mW	0.872 mW	16%	2.31	1.18
NET_3	52383	38.3 mW	0.877 mW	14%	2.32	1.08
NET_4	49594	38.5 mW	0.858 mW	14%	2.35	1.00
NET_5	48944	38.9 mW	0.853 mW	11%	2.34	0.97

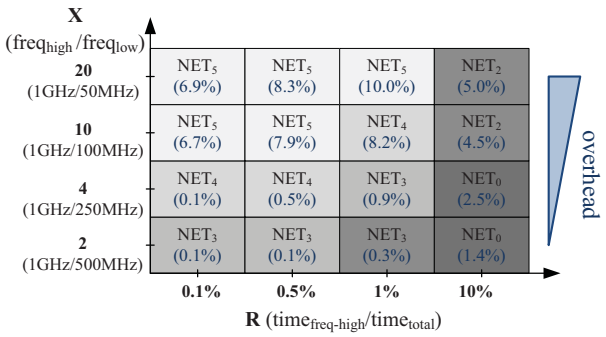


Fig. 1. Single constraint netlist with minimum average power in each (R, X) scenario and average power overhead compared to ideal average power (OpenSPARC multiplier).

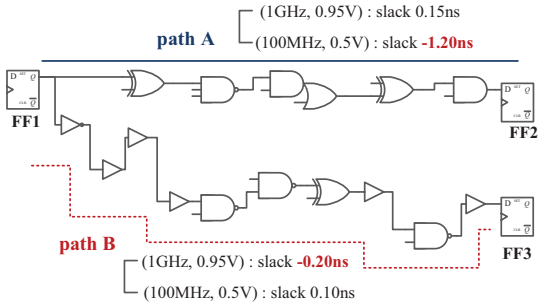


Fig. 2. As voltage scales, the set of critical paths for a design can change. If paths are not properly optimized, this phenomenon can result in limited voltage scaling for a conventional DVFS design as frequency is scaled down. On the other hand, optimizing the full set of critical paths for a multimode design contributes to leakage overhead in each operating mode.

of time spent in each operating mode (R), the range of scalability between high and low performance (X), and the tightness of the timing constraint. We quantify energy in terms of average power, which accounts for the power consumption and fraction of time spent in each mode: $E = R \cdot Pwr(f_{\text{hi}}) + (1 - R) \cdot Pwr(f_{\text{lo}})$.

Fig. 1 also shows the average power overhead of the netlists compared to the ideal average power. Ideal average power is calculated using the power consumption of the minimum power implementation for each mode. In general, energy inefficiency is greater when duty cycle is low and range of scalability is high. This can result in significant energy overheads for energy-constrained designs that attempt to reduce energy by aggressively scaling down voltage and frequency and spending more time in a low-power mode.

One reason for inefficiency compared to the ideal case is that the delays of different paths scale differently with voltage, and thus, the set of critical paths changes as voltage scales. For example, Fig. 2 shows two paths from the integer

multiplier of the OpenSPARC core, along with their timing slack at different minimum-energy operating points. At high performance (1 GHz, 0.95 V), path B is a critical path, while path A is not. However, at low performance (100 MHz, 0.5 V), path A is a critical path, while path B is not. The reason for this reversal is that different paths have different delay sensitivities to voltage scaling. In high performance mode, paths with large logic depths tend to be critical, but such paths are also optimized with cells that tend to be less sensitive to voltage scaling [e.g., cells with lower threshold voltage (V_t), shorter interconnect, etc.]. Paths that have high V_t cells or long interconnects, e.g., are more sensitive to voltage scaling. While these conditions do not significantly affect delay near nominal voltage, they amplify the increase of path delay when voltage is scaled down. In the example, voltage scaling causes delay to increase faster for path A so that the delay of path A overtakes the delay of path B. This means that path A, which was not critical at high performance, limits voltage scaling at low performance if not properly optimized. Random logic structures may be naturally more susceptible to reversal of critical paths under voltage scaling than regular memory structures, since path characteristics in random logic vary more.

A conventional multimode design methodology spends resources to optimize the critical paths in all modes and is therefore over-optimized at any single operating mode. A context-aware multimode design methodology increases DVFS efficiency over conventional multimode design approaches by accounting for the operating scenario during constraint selection. However, in some scenarios, there can still be significant overhead, especially when the range of scalability is large. For such scenarios, microarchitectural techniques may be useful for increasing DVFS efficiency, as described in Sections IV-B and VI.

Another factor that can influence DVFS efficiency is the amount of combinational logic in a design. As we show in Section III (Table I), combinational logic area can vary substantially as the timing constraint changes, because the entire topology of the optimal implementation can change (re-clustering, buffering, cell sizes and types, etc.). For sequential logic, changes are limited to cell threshold voltages and drive strengths, so there is less variation between tightly and loosely constrained designs. Designs heavy on combinational logic (e.g., ALUs) tend to have worse DVFS efficiency, especially when the range of frequency scaling is large.

Algorithm 1 Context-Aware Multimode Design

Procedure MultiModeDesign($v_{\text{hi}}(0)$, f_{hi}, R, X)
1. $f_{\text{lo}} \leftarrow f_{\text{hi}}/X$; $v_{\text{hi}} \leftarrow v_{\text{hi}}(0)$;
2. **while true do**
3. $E_{\min} \leftarrow \infty$;
4. **for** $i = 0$; $i \leq 2\delta$; $i \leftarrow i + 1$ **do**
5. $v_i \leftarrow v_{\text{hi}} + (i - \delta)v_{\text{step}}$;
6. $N_i \leftarrow SM(v_i, f_{\text{hi}})$;
7. $E_i \leftarrow R \cdot Pwr(N_i, f_{\text{hi}}) + (1 - R) \cdot Pwr(N_i, f_{\text{lo}})$;
8. **if** $E_i < E_{\min}$ **then**
9. $v_{\min} \leftarrow v_i$; $E_{\min} \leftarrow E_i$; $N_{\min} \leftarrow N_i$;
10. **end if**
11. **end for**
12. **if** $v_{\min} = v_{\text{hi}}$ **then break;**
13. **else** $v_{\text{hi}} \leftarrow v_{\min}$;
14. **end while**
15. $v_{\text{lo}} \leftarrow$ min. safe operating voltage of netlist $N_{\min}$ at frequency f_{lo};
16. **while true do**
17. $N \leftarrow MM(N_{\min}, f_{\text{hi}}, v_{\text{hi}}, f_{\text{lo}}, v_{\text{lo}})$;
18. $E \leftarrow R \cdot Pwr(N, f_{\text{hi}}) + (1 - R) \cdot Pwr(N, f_{\text{lo}})$;
19. **if** $E < E_{\min}$ **then**
20. $E_{\min} \leftarrow E$; $N_{\min} \leftarrow N$; $v_{\text{lo}} \leftarrow v_{\text{lo}} - v_{\text{step}}$;
21. **else**
22. $v_{\text{lo}} \leftarrow v_{\text{lo}} + v_{\text{step}}$; **break**;
23. **end if**
24. **end while**
25. **return** $N_{\min}, v_{\text{hi}}, v_{\text{lo}}$;

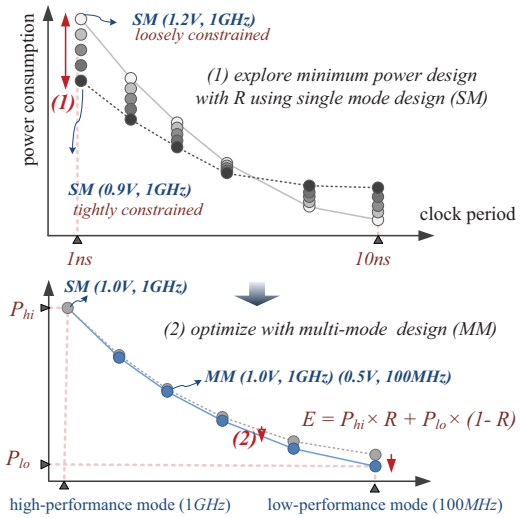


Fig. 3. Context-aware multimode design flow. (1) Algorithm 1: Lines 1 ∼ 14. (2) Algorithm 1: Lines 15 ∼ 25.

IV. Maximizing DVFS Efficiency

In Section III, we have discussed several reasons why modern DVFS designs may be inefficient. We now propose a context-aware multimode design approach that considers operating conditions, performance metric, and constraints to maximize energy efficiency over multiple modes of operation.

A. Context-Aware Multimode Design

Before implementing a multimode design, conventional EDA tools require all operating modes to be completely constrained (frequency and voltage). However, as we have shown in Section III, multimode designs that are oblivious to the operating scenario can be energy-inefficient. For this reason, we propose a design flow that postpones constraint finalization and uses information about the design and operating scenario to select the most appropriate set of constraints.

Fig. 3 and Algorithm 1 describe our context-aware multimode design flow. The figure shows the total power consumption (y-axis) of a design for each clock period (x-axis). Circles with the same color indicate the same netlist.

The procedure takes as input the high-performance frequency target (f_{hi}), the operational duty cycle (R), the range of scalability (X), and an initial high-performance voltage target from which to start the optimization ($v_{\text{hi}}(0)$). This flow implements energy-efficient netlist $N_{\min}$ according to the scenario (R, X) and determines the voltage constraints that minimize average power across high and low-performance modes (v_{hi}, v_{lo}). The first while loop in Algorithm 1 [corresponding to Fig. 3(a)] selects a design with minimum average power among single-mode implementations with different timing

Algorithm 2 Constraint Selection Pre-Processing Stage

```
Procedure RangeSelection(v_hi, f_hi, R, X)
1. f_lo ← f_hi/X;   E_min ← ∞;
2. for i = 0 ; i ≤ 2γ ; i ← i + 1 do
3.    v_i ← v_hi + (i − γ)v_step;
4.    N_i ← SM(v_i, f_hi);
5.    E_i ← R · Pwr(N_i, f_hi) + (1 − R) · Pwr(N_sm, f_lo);
6.    if E_i < E_min then
7.       v_hi(0) ← v_i;   E_min ← E_i;
8.    end if
9. end for
10. return v_hi(0);
```

Algorithm 3 Context-Aware Multimode Design

```
Procedure K-ModeDesign(v(0), f_1, f_2, ..., f_K, R_1, R_2, ..., R_K)
1. v_1 ← v(0);
2. while true do
3.    E_min ← ∞;
4.    for i = 0 ; i ≤ 2δ ; i ← i + 1 do
5.       v_i ← v_1 + (i − δ)v_step;
6.       N_i ← SM(v_i, f_1);
7.       E_i ← R_1·Pwr(N_i, f_1)+R_2·Pwr(N_i, f_2)+···+R_K·Pwr(N_i, f_K);
8.       if E_i < E_min then
9.          v_min ← v_i;   E_min ← E_i;   N_min ← N_i;
10.      end if
11.   end for
12.   if v_min = v_1 then break;
13.   else v_1 ← v_min;
14. end while
15. for j = 2 ; j ≤ K ; j ← j + 1 do
16.   v_j ← min. safe operating voltage of netlist N_min at frequency f_j;
17.   while true do
18.      N ← MM(N_min, f_1, v_1, ..., f_j, v_j, ..., f_K, v_K);
19.      E_i ← R_1·Pwr(N_i, f_1)+R_2·Pwr(N_i, f_2)+···+R_K·Pwr(N_i, f_K);
20.      if E < E_min then
21.         E_min ← E;   N_min ← N;   v_j ← v_j − v_step;
22.      else
23.         v_j ← v_j + v_step;
24.         break;
25.      end if
26.   end while
27. end for
28. return N_min, v_1, ..., v_K;
```

constraints. The constraint is varied by changing the target voltage (using different timing libraries characterized for each voltage). $SM(v_i,\ f_{\text{hi}})$ (Line 6) represents a physical design implementation that has been optimized for a single design point-voltage v_i and frequency f_{hi}. E is average power consumption, and $Pwr(N,\ f)$ is the power consumption of netlist N at frequency f and the minimum safe voltage available through dynamic voltage scaling. δ defines the radius of a local comparison window to avoid selecting a local optimum because of EDA tool noise.

While optimizing constraint specification encompasses a 2-D space (v_{hi}, v_{lo}), performing a full evaluation of the constraint space would require too much computational effort. We rely on the following observations to accelerate constraint selection. First, the tightest single constraint dominates the others in determining the amount of area spent to reduce voltage. By considering the energy efficiency as v_{hi} is varied, we can effectively constrain v_{hi} to balance leakage/area reduction and voltage scaling in an operating scenario (R, X), independent of v_{lo}. To determine the minimum-energy constraints, we vary v_{hi} first, because delay sensitivity to voltage is greater at low voltages. Small voltage changes around v_{lo} cause large timing changes and result in large netlist changes. Thus, initially tuning the constraint with the same precision at v_{lo} would require approximately an order of magnitude finer characterization of voltage libraries.

The second while loop of Algorithm 1 [corresponding to Fig. 3(b)] optimizes the selected design for multiple operating modes. $MM(N,\ f_{\text{hi}}, v_{\text{hi}},\ f_{\text{lo}}, v_{\text{lo}})$ performs a multimode (incremental) optimization for high-performance mode ($f_{\text{hi}},\ v_{\text{hi}}$) and low-performance mode ($f_{\text{lo}},\ v_{\text{lo}}$). We continue to reduce v_{lo} (tighten the constraint on low-performance mode) until average power E is minimized. Since multimode optimization considers high and low-performance constraints, the second optimization stage optimizes critical paths in all modes and enables further voltage scaling in low-performance mode. To further accelerate constraint selection for v_{hi}, we evaluate the effectiveness and runtime efficiency of using adaptive step sizing for v_{step}. We add an optional pre-processing loop (Algorithm 2) and initially use a large value of v_{step} to select an appropriate range for fine tuning of the constraint. The search space for the pre-processing stage spans a radius of $\gamma \cdot v_{\text{step}}$, centered around an initial estimate of the high-performance constraint (v_{hi}). For example, if $\gamma = 3$ and $v_{\text{step}} = 0.05V$, the pre-processing stage will locate an efficient starting point for Algorithm 1 ($v_{\text{hi}}(0)$) in the range of $v_{\text{hi}} \pm 0.15V$. We observe the shape of the average power versus v_i curve to locate the voltage range that contains the minimum energy design point. Algorithm 2 describes the range selection algorithm that feeds the selected initial constraint value ($v_{\text{hi}}(0)$) to Algorithm 1. With the coarse-grained search, our heuristic reduces the number of implementation steps required to fine-tune the constraint, thus reducing runtime. We compare the runtime and average power reduction of our context-aware multimode design heuristic with and without adaptive step sizing in Section VI.

Although we focus on enhancing DVFS efficiency over two modes that define the boundaries of the range of scalability, the concepts presented here can be generalized for an arbitrary number of modes. In Algorithm 3, we present a generalized, K-mode version of our context-aware multimode design heuristic. Rather than a single (R, X) pair, we specify K frequency constraints $(f_1, \ldots, f_K)$ and the fraction of time spent in each mode $(R_1,_2, \ldots, R_K)$. The K-mode design flow is similar to the previous two-mode flow (Algorithm 1). The formulation of average power (Lines 7, 19) accounts for the contributions of each mode, and the second while loop (Lines 17–26) is repeated $K - 1$ times to successively refine the multimode design and find the voltage constraints for each mode that minimize average power. We demonstrate the generalized context-aware multimode design for the case of three modes in Section VI.

B. Replication-Based DVFS Design

As noted in Section III, there are scenarios in which multimode designs exhibit significant inefficiency with respect to the ideal, due to the area and power overheads of operating in modes for which the design was not specifically optimized. In cases when the overhead is substantial, replication-based design can be used to target each mode individually. We propose a selective replication technique that identifies the modules that cause the most inefficiency and suggests replication for only those modules. Other modules are optimized with context-aware multimode design.

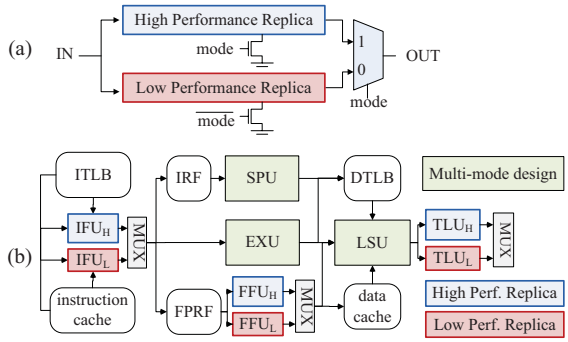


Fig. 4. (a) Overview of replication-based design. (b) Example selective replication-based processor design for the OpenSPARC T1.

Fig. 4(a) offers a high-level representation of replication and power gating circuitry. A circuit module is replicated, and each replica is optimized for a different performance mode. The control signal, mode, selects the active operating mode by power gating the appropriate replica and selecting the correct MUX input. Fig. 4(b) shows an example of selective replication-based design.

The benefits of replication come at the cost of substantial area overhead due to replicated circuitry, especially when replication is performed at a coarse granularity (i.e., large replicated blocks). Replication overheads due to power gating and MUX circuitry can also be high when replicating at a fine granularity. We evaluate replication decisions at the granularity of RTL modules, and we analyze replication at different granularities in this section. Partitioning the design for optimal-granularity replication requires rewriting the RTL and is beyond the scope of this paper. We focus on finding the best way to optimize a given DVFS design.

Though replication area overheads can be substantial, most modules do not warrant replication. Modules that are loosely constrained or that have a significant fraction of sequential cells do not require many high-leakage cells or significant topology restructuring to meet performance constraints. Consequently, they do not impose significant power overheads at scaled frequencies and voltages, and do not necessitate replication.

In addition, implementations of large structures, such as caches, that are optimized for the lowest safe operating voltage or on a separate voltage rail are not significantly affected by scaling. These structures do not necessitate replication. This is beneficial for consistency (no state copying on mode switch), rapid switching between power-gated replicas, and reduction of the area overhead of replication. If replicas share access to a memory structure, the interface and interconnect might require modification to accommodate the multiple replicas, possibly affecting the access time for the structure. If this is the case, access time for the high-performance replica can be minimized at the expense of the low-performance replica. This strategy avoids performance degradation, since timing constraints in the low-performance mode are considerably relaxed with respect to the high-performance mode.

To accommodate aggressive voltage scaling, memory structures are typically optimized for the lowest safe voltage or placed on a separate voltage rail. Low-voltage SRAMs [11], [12] can safely operate at voltages below 400 mV, which is the minimum voltage we consider in our study. At 45 nm and below, split-rail power distribution [13] is common. Except where noted otherwise, our results assume

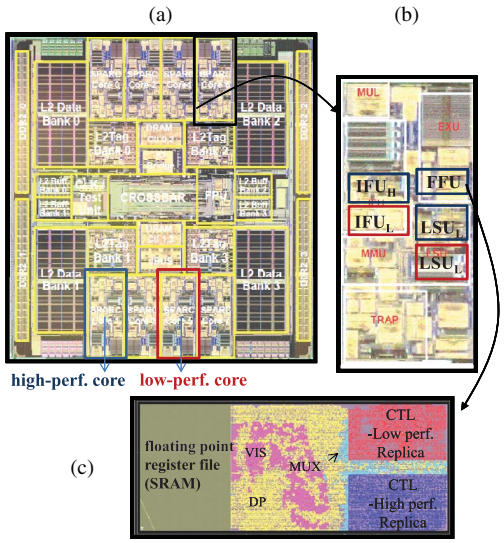


Fig. 5. Replication-based design at different levels of abstraction can lead to (a) heterogeneous CMP, (b) coarse-grained, and (c) fine-grained selective replication designs.

split-rail power distribution with SRAMs on a separate voltage rail.

In order to choose the most energy-efficient partitioning of modules between replication and context-aware multimode design, we solve a disjunctively-constrained 0-1 knapsack problem [14] in which the knapsack items are the replication-based and multimode module implementations. For each module, one implementation is selected. The profit for an item is the average power savings afforded by selecting a certain optimization strategy for the module, and the weight is the area of the implementation. For replicated modules, average power savings must account for the energy consumed by the active replica, the power-gated replica, MUX logic, and power gating cells. Area must account for both replicas, power gating cells, and MUXes. The capacity of the knapsack is the area budget for the core. Thus, solving the knapsack problem corresponds to choosing the partitioning of replicated and multimode modules that maximizes energy savings while fitting within the core's area budget.

Fig. 5 shows a replication-based DVFS architecture at different levels of abstraction. Subfigure (a) shows that replication for different performance targets can be performed at the core level to create a heterogeneous multicore architecture in which tasks with different power and performance requirements are scheduled to different cores. Subfigure (b) zooms in on an individual core, showing coarse-grained replication at the module level. The instruction fetch unit (IFU) and load store unit (LSU) have been replicated. Subfigure (c) zooms further into the floating point frontend unit (FFU) module to show fine-grained replication of the FFU control unit.

Our replication strategy uses power gating to turn off the inactive replica. The overhead of power gating depends on the number of gating cells required. Equation (1) calculates the required number of gating cells for a circuit module with maximum current I_{total}

$$\delta V_{\text{dd}} = I_{\text{total}} * (R_{\text{sw}}/N_g) < \text{margin} * V_{\text{dd}}. \quad (1)$$

In this equation, R_{sw} is the resistance of a power gating cell, N_g is the required number of power gating cells, and $margin$ is chosen to be 1% V_{dd}. We use SPICE simulation

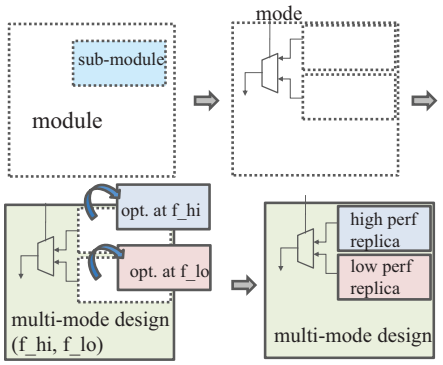


Fig. 6. Implementation flow for selective replication.

TABLE II
TARGET MODULES FROM OPENSPARC T1

Module	Stage	Description	%seq	Area (μm^2)
EXU	EX	Integer Execution	20%	81 902
FFU	EX	Floating Point Front-end	30%	28 584
IFU	F/D	Instruction Fetch & Decode	40%	121 458
LSU	MEM/WB	Load Store	45%	125 725
MUL	EX	Integer Multiplier	15%	61 978
SPU	EX	Stream Processing	45%	33 580
TLU	MEM/WB	Trap Logic	47%	111 902

to obtain R_{sw} and gate leakage power at each voltage. We also account for the delay overhead from IR drop. In addition to the overhead of power gates, replication requires that we place MUXes at the ports of a replicated module. This incurs an additional power and area cost, and also adds some latency to the timing paths of the module.

Fig. 6 shows the implementation flow of our selective replication procedure. The proposed design approach replicates only the modules that maximize energy savings per area, and performs multimode design for the rest. We first implement a target module and select sub-modules to be replicated, using the knapsack optimization. We then change the implemented netlist and make a floorplan for the top module. In the netlist modification, the selected module is re-synthesized for high- and low-performance modes, and MUXes are connected to the output ports of the replicas. Third, we partition each replica from the top module and implement (place and route) them separately. The high-performance replica is optimized at high frequency, and the low-performance replica at low frequency. Finally, we merge each partition and report timing and power for the entire module.

V. METHODOLOGY

For our experiments, we use all modules (Table II) that comprise the OpenSPARC T1 processor [15]. In Table II, %seq is the percentage of sequential cell area in the module. As we will show, %seq impacts DVFS efficiency. We also evaluate the issue and load store unit (LSU) stages of the FabScalar [16] processor in different microarchitectural configurations to understand the microarchitectural dependence of DVFS efficiency. We evaluate pipelining and superscalar width in the Issue stage, from 1-wide, 1-deep to 4-wide, 3-deep, and we evaluate the impact of changing the queue sizes and superscalar widths in both the Issue and LSU stages. Since the maximum operating frequency of FabScalar is less than that of OpenSPARC, we consider a smaller range of scalability

TABLE III
ANALYSIS OF DESIGN CHOICES FOR MULTIMODE DESIGN HEURISTIC IMPLEMENTATION (FOR A SUBMODULE OF LSU)

	Without Pre-Processing		With Pre-Processing	
Step	0.01 V	0.02 V	0.01 V	0.02 V
(a) No. of Implementation	23	18	12	11
(b) No. of Optimization	7	4	7	4
(c) Runtime (sec)	4199	3196	2395	2048
(d) Average Power (W)	3.26E-05	3.38E-05	3.26E-05	3.46E-05

(X) for FabScalar experiments. Consequently, we increase the range of R for these experiments to ensure adequate coverage of the interesting points in the (R, X) space. Designs are implemented with TSMC 65GP (65nm) libraries that have been characterized by Cadence Library Characterizer v9.1 [17] for low, nominal, and high threshold voltages over a range of operating voltages. The initial netlists are synthesized with Synopsys Design Compiler C-2009.06 [18], and layout is performed in Cadence SoC Encounter v8.1 [6]. As described in Section IV-A, we perform implementation at various voltages to find a minimum-energy solution. To mitigate "inherent noise" in EDA tools [19], [20], we implement each design three times with a small variation in the timing constraint ($+0.5$ ps, -0.5 ps and no variation) and choose the design with minimum average power. When evaluating conventional multimode design, we choose the voltage constraint for each mode as the voltage that minimizes power for a single-mode design at that mode's operating frequency.

To estimate power consumption, we perform gate-level simulations for one million clock cycles using real workloads. Test vectors for gate-level simulations are gathered from full-system RTL simulations of SPEC benchmarks (art, bzip2, equake, gzip, mcf, mesa, twolf) on the OpenSPARC and FabScalar processors. Leakage and dynamic power consumption are reported by Synopsys PrimeTime-PX [21]. Note that our results assume the same workloads for high and low-performance modes, whereas the activity characteristics may vary significantly between a workload that runs at high frequency and one that runs at low frequency. Though this may be the case, the duty cycle (R) can act as a catch-all to adjust the weighting between the two modes during optimization, not only in terms of time spent in each mode, as originally described, but also to account for factors such as disparity in average circuit activity.

VI. RESULTS AND ANALYSIS

In this section, we analyze our heuristic design choices, quantitatively demonstrate the benefits of context-aware multimode and selective replication-based designs, and explore the implications of microarchitecture on the energy efficiency of DVFS design.

A. Heuristic Design

We begin with an evaluation of the design choices that led to our chosen heuristic implementation for multimode design. Table III compares the average power and runtime efficiency of heuristic implementations that vary in voltage step size (v_{step}) and whether or not the pre-processing stage (Algorithm 2 in Section IV-A) is used. The pre-processing stage reduces runtime significantly by reducing the number of single-mode implementations [row (a) in Table III] required for minimum-energy constraint selection. For a step size of 0.01 V, pre-processing reduces the constraint search space by 48% while

TABLE IV
AVERAGE POWER REDUCTION FOR CONTEXT-AWARE MULTIMODE AND REPLICATION-BASED DESIGNS AGAINST CONVENTIONAL MULTIMODE DESIGN (HIGHER IS BETTER)

		Context-Aware Multimode			Replication-Based Design		
Test Case	X	$R = 0.5\%$	$R = 1\%$	$R = 5\%$	$R = 0.5\%$	$R = 1\%$	$R = 5\%$
EXU	5	1.5%	1.4%	2.3%	8.7%	8.3%	6.6%
	10	4.1%	4.9%	4.4%	9.6%	9.1%	7.4%
	20	13.0%	9.4%	2.6%	15.6%	12.1%	4.1%
FFU	5	3.8%	4.3%	6.9%	5.8%	6.2%	8.1%
	10	4.6%	4.9%	3.3%	6.1%	5.5%	3.4%
	20	4.0%	2.1%	1.8%	3.6%	2.1%	-1.7%
IFU	5	2.7%	2.7%	9.0%	0.4%	1.1%	4.1%
	10	2.7%	3.3%	7.1%	3.6%	3.5%	3.4%
	20	1.5%	2.4%	2.6%	0.3%	0.8%	1.8%
LSU	5	4.4%	5.4%	9.8%	6.2%	7.3%	12.1%
	10	1.6%	1.8%	9.7%	4.8%	6.2%	10.2%
	20	3.6%	2.3%	5.4%	5.5%	6.1%	7.2%
MUL	5	19.5%	12.4%	8.7%	25.4%	24.0%	16.6%
	10	4.0%	2.2%	5.4%	15.1%	14.2%	11.5%
	20	16.2%	6.5%	5.7%	23.1%	19.7%	11.5%
SPU	5	4.0%	3.9%	4.7%	1.4%	1.7%	3.0%
	10	2.6%	3.0%	4.4%	4.2%	3.9%	2.9%
	20	5.9%	4.6%	1.8%	8.0%	5.9%	1.0%
TLU	5	5.3%	5.6%	7.0%	3.4%	3.9%	6.3%
	10	7.0%	7.8%	3.6%	3.9%	4.7%	7.1%
	20	6.5%	5.1%	3.6%	8.8%	8.6%	8.2%
Total	5	6.0%	5.1%	7.6%	8.2%	8.3%	9.1%
	10	2.0%	3.1%	6.8%	7.0%	7.1%	7.4%
	20	7.1%	4.5%	3.7%	9.1%	7.9%	5.1%

TABLE V
AVERAGE POWER OVERHEAD FOR CONTEXT-AWARE MULTIMODE AND REPLICATION-BASED DESIGNS WITH RESPECT TO THE IDEAL CASE (LOWER IS BETTER)

		Context-Aware Multimode			Replication-Based Design		
Test Case	X	$R = 0.5\%$	$R = 1\%$	$R = 5\%$	$R = 0.5\%$	$R = 1\%$	$R = 5\%$
EXU	5	8.1%	8.0%	5.8%	0.2%	0.3%	1.0%
	10	6.5%	5.3%	4.7%	0.4%	0.6%	1.5%
	20	3.8%	4.2%	3.3%	0.7%	1.0%	1.7%
FFU	5	2.4%	2.5%	2.6%	0.2%	0.4%	1.3%
	10	2.1%	1.4%	2.0%	0.4%	0.8%	1.9%
	20	3.9%	3.9%	3.4%	4.3%	4.0%	3.3%
IFU	5	1.5%	2.1%	1.7%	3.9%	3.9%	3.6%
	10	1.4%	1.2%	1.8%	0.6%	1.0%	2.2%
	20	0.2%	0.1%	1.7%	1.0%	1.5%	2.6%
LSU	5	2.1%	2.4%	3.5%	0.2%	0.3%	0.9%
	10	7.1%	5.3%	1.9%	0.4%	0.6%	1.3%
	20	2.7%	5.0%	3.6%	0.6%	0.9%	1.6%
MUL	5	8.1%	15.7%	10.7%	0.2%	0.4%	1.1%
	10	13.5%	14.8%	8.6%	0.4%	0.7%	1.6%
	20	9.8%	17.7%	8.6%	0.8%	1.1%	2.0%
SPU	5	1.8%	2.2%	3.1%	4.6%	4.6%	4.9%
	10	2.5%	2.4%	2.0%	0.9%	1.5%	3.6%
	20	7.7%	6.8%	4.6%	5.3%	5.3%	5.4%
TLU	5	2.0%	1.8%	0.8%	1.6%	1.8%	2.0%
	10	3.2%	3.3%	3.8%	2.1%	2.4%	2.7%
	20	2.5%	3.8%	5.0%	1.8%	1.1%	1.4%
Total	5	3.8%	5.1%	3.5%	1.4%	1.5%	2.0%
	10	5.9%	5.2%	2.4%	0.5%	0.8%	1.8%
	20	3.5%	5.3%	3.8%	1.3%	1.6%	2.3%

achieving the same energy efficiency. This reduces runtime by 43%.

Increasing the size of v_{step} also reduces runtime, but with a potential cost in energy efficiency, due to the coarser granularity of constraint tuning. Without pre-processing, doubling v_{step} reduces runtime by 24% and increases average power by 3.7%. When the pre-processing stage is also used, doubling v_{step} reduces runtime by 15% and increases average power by 6.1%. Thus, the marginal benefit of increasing the step size is higher when pre-processing is not used. Based on our analysis, we apply pre-processing with $\gamma = 2$, $v_{\text{hi}} = 0.80V$, and $v_{\text{step}} = 0.05V$. A step size of $v_{\text{step}} = 0.01V$ is used for the main heuristic procedure.

B. Context-Aware Multimode Design

To gauge the effectiveness of context-aware DVFS design, we first present average power results for the modules of the OpenSPARC processor. Table IV shows the average power reduction achieved by context-aware multimode and replication-based designs,[2] compared to conventional multimode design (high-performance clock frequency $f_{\text{hi}} =$ 1 GHz). Table V shows average power overhead for context-aware DVFS with respect to the ideal. The "total" rows represent processor-wide results.

From Table IV, we observe that context-aware multimode design improves DVFS efficiency by up to 19.5% with respect to conventional multimode design. In general, benefits are higher for low %seq (e.g., EXU, MUL) and low R, as is typical in energy-constrained designs. Processors that have a higher fraction of area devoted to execution units (e.g., DSP processors) or spend more time in a low-power mode (e.g., mobile devices), should see more benefits from context-aware design. For combinational logic, area and leakage can change significantly with different constraints, due to topological adaptations. Therefore, targeted designs for different performance constraints can differ substantially. As such, any multimode design is necessarily inefficient at one or more performance targets, especially when the range of scalability (X) is high.

To analyze context-aware multimode design for more than two modes, we have evaluated designs with more modes. Table VI compares three-mode context-aware designs (f_1, f_2, f_3 = 1 GHz, 500 MHz, 100 MHz) against high performance targeted designs ($f = 1$ GHz) in terms of power consumption in each operating mode. We have chosen the duty cycles (R_1, R_2, and R_3) such that each mode contributes roughly an equivalent fraction of average power. We also evaluate the impact of split-rail design, where SRAMs are on a separate voltage rail, versus single-rail design, where the minimum operating voltage of the chip is limited to 0.6 V. Although the minimum operating voltage of typical SRAMs is around 0.7 V, some special-purpose SRAMs can operate well at lower voltages with design overheads; 0.6 V represents a voltage within the operating range of many low-power SRAMs (e.g., [11], [12], [22]). On average, context-aware multimode design reduces average power by 12.4% for a design with low %seq (MUL) and 10.3% for a design with high %seq. For the single-rail case, benefits for MUL decrease only slightly, since limiting the minimum voltage reduces the disparity between optimal high-performance and low-performance designs, such that the optimal design point shifts more toward high-performance mode. So, although power reduction decreases for low-performance mode (mode3), it increases for high-performance mode (mode1). Average power

[2]In the replication-based results, power, area, and delay overheads from power gating cells, MUXes, and IR drop have been included. We do not consider wakeup energy.

TABLE VI
COMPARISON OF AVERAGE POWER CONSUMPTION BETWEEN SINGLE-MODE DESIGN AND CONTEXT-AWARE MULTIMODE DESIGN FOR THREE MODES

Module	Mode	Frequency	R	Single-Mode (Mode-1)		Context-Aware Multimode			Context-Aware Multimode (Single-Rail)		
				Voltage (V)	Total Power (W)	Voltage (V)	Total Power (W)	Reduction (%)	Voltage (V)	Total Power (W)	Reduction (%)
MUL	mode1	1 GHz	0.02	0.92	2.950E-2	0.99	3.181E-2	−7.83%	0.98	3.077E-2	−4.31%
	mode2	500 MHz	0.28	0.69	8.892E-3	0.69	7.755E-3	12.78%	0.68	7.873E-3	11.46%
	mode3	100 MHz	0.70	0.48 (0.60)	1.140E-3 (1.82E-3)	0.46	8.405E-4	24.55%	0.60	1.565E-3	14.13%
	Average				3.877E-3		3.396E-3	12.41%		3.915E-3	10.11%
SPU	mode1	1 GHz	0.02	0.86	1.050E-2	0.87	1.108E-2	−5.47%	0.88	1.108E-2	−5.56%
	mode2	500 MHz	0.28	0.67	3.231E-3	0.63	2.875E-4	11.03%	0.64	2.942E-3	8.96%
	mode3	100 MHz	0.70	0.47 (0.60)	3.731E-4 (6.30E-4)	0.41	2.968E-4	20.44%	0.60	6.064E-4	3.66%
	Average				1.376E-3		1.234E-4	10.30%		1.470E-3	5.50%

TABLE VII
DIFFERENT SCENARIOS FOR THREE-MODE IMPLEMENTATION

Scenario	Energy Consumption			Duty Cycle (R)		
	mode1	mode2	mode3	mode1	mode2	mode3
S_1	100%	0%	0%	1.000	0.000	0.000
S_2	65%	30%	5%	0.100	0.660	0.240
S_3	65%	5%	30%	0.060	0.070	0.870
S_4	30%	65%	5%	0.020	0.830	0.150
S_5	30%	5%	65%	0.020	0.030	0.950
S_6	5%	65%	30%	0.002	0.498	0.500
S_7	5%	30%	65%	0.002	0.172	0.826

TABLE VIII
AVERAGE POWER CONSUMPTION IN EACH SCENARIO (MUL CASE)

Scenario	net_1	net_2	net_3	net_4	net_5	net_6	net_7
S_1	1.000	1.043	1.043	1.092	1.084	1.126	1.136
S_2	1.088	1.000	1.000	1.009	1.036	1.021	1.048
S_3	1.067	1.000	1.000	1.033	1.005	1.049	1.026
S_4	1.154	1.009	1.009	1.000	1.046	1.003	1.042
S_5	1.171	1.031	1.031	1.064	1.000	1.070	1.003
S_6	1.178	1.010	1.010	1.001	1.035	1.000	1.026
S_7	1.215	1.023	1.023	1.027	1.013	1.024	1.000

reduction decreases more significantly for SPU. Since SPU has high %seq, the difference between high-performance and low-performance designs is not as significant. Thus, the primary source of power savings is power reduction in low-performance mode (mode3), and limiting the minimum voltage significantly cuts into those savings; in the single-rail case, SPU shows only 3.66% average power reduction in mode3, due to the minimum voltage limitation.

To demonstrate the potential benefit of context-aware design in the three-mode case, we have implemented context-aware multimode designs targeting seven different scenarios. Each scenario has different duty cycles (R), as described in Table VII. The optimized netlist for scenario S_i is net_i. Table VIII shows the average power consumption (normalized to the power of net_i for S_i) of each netlist in each scenario. The context-aware design targeting a specific scenario minimizes average power for that scenario. Since our heuristic pinpoints the minimum energy constraints for a design by performing small explorations in the constraint space, the optimization result can be improved by searching more design points or searching at a finer granularity (smaller step size, V_{step}). Our experiments use $V_{step} = 0.01$ V, and some netlists show only small benefits over the netlist optimized for a different scenario. E.g., net_2 and net_3 are nearly identical, since the two scenarios are similar (mode 1 is dominant for S_2 and S_3). Similarly, net_4 and net_6 have similar average power in all scenarios, because mode 2 is dominant for S_4 and S_6.

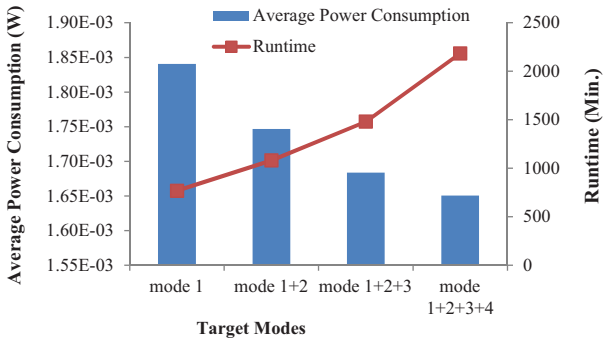


Fig. 7. Results for context-aware implementations with different numbers of modes (MUL case). Frequencies for each mode are 1 GHz, 500 MHz, 250 MHz, and 100 MHz, and duty cycle values (R_k) are selected so that each mode consumes roughly an equal share of the total power.

Fig. 7 quantifies runtime and average power savings for context-aware designs as the number of modes increases from one to four. Increasing the number of modes causes runtime to increase approximately linearly, since the complexity of Algorithm 3 is linear with respect to the number of modes. As the number of modes increases, average power decreases, since context-aware design: 1) enables a lower voltage for a given mode by optimizing the critical paths in each mode and 2) reduces area and leakage for the design by accounting for the duty cycles and range of scalability.

C. Selective Replication-Based Design

Even though context-aware multimode design has significant benefits over conventional multimode design, when %seq is low, it can still exhibit considerable inefficiency (up to 17.7%) with respect to the ideal. This is because the ideal area and power consumption of combinational logic change considerably with the timing constraint. Fortunately, selective replication minimizes average power in these cases, coming within 1% of the ideal average power, on average, at the expense of area overhead. This suggests that an appropriate combination of context-aware and replication-based approaches may be nearly ideal for maximizing DVFS efficiency. Note that replication does not achieve significant benefits over multimode design for modules like SPU and IFU. Most of the benefits of replication come from reducing leakage and area overheads from the combinational logic. However, these modules have high %seq, so that the extent of achievable benefits is considerably reduced. In fact, in a few scenarios, context-aware multimode design even does better than replication, due to the timing and power overheads of replication. In many cases, the significant area overheads of module replication (94% on

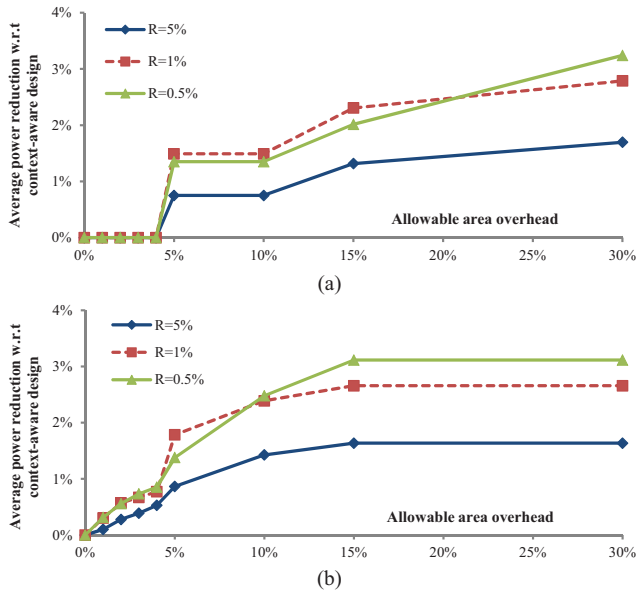


Fig. 8. Selective replication achieves additional average power reduction over context-aware multimode design for the OpenSPARC processor, coming within 1% of the ideal average power, on average, for only 5% area overhead. (a) Coarse-grained selective replication. (b) Fine-grained selective replication.

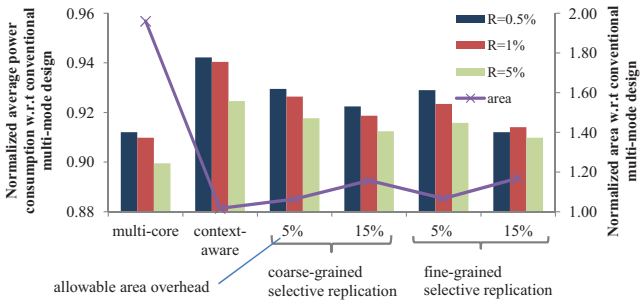


Fig. 9. Average power consumption and area comparison among multicore, context-aware, and selective replication-based design.

average) are not justified. For a given area budget, we use our knapsack-based selective replication technique to identify the processor modules to replicate, such that the average power of the processor is minimized.

Fig. 8 shows average power reduction achieved by selective replication for the OpenSPARC processor with a given area budget. Note that the benefits shown in Fig. 8 are in addition to those achieved by the best context-aware multimode design for the particular scenario. The results demonstrate that in most cases, context-aware multimode design achieves close to ideal average power, and that replicating only a small number of modules closes the gap between context-aware design and ideal average power. The final result with context-aware and selective replication-based design is a DVFS processor with average power that is within 1% of ideal, on average, with only 5% area overhead. Note that increasing the area budget beyond 5% yields only minimal incremental benefits, confirming that only a few modules need to be replicated.

Fig. 8 also compares replication at the module granularity (Table II modules) to replication at the fine granularity of leaf modules. Typically, replication at the finer granularity achieves larger average power reduction for a given area budget, as inefficient sub-modules can be targeted more precisely without

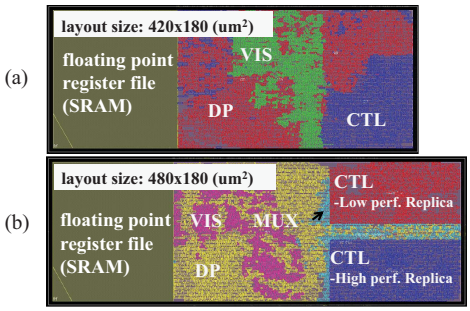


Fig. 10. Layout of floating point front-end unit (FFU) module (a) without replication and (b) with replication.

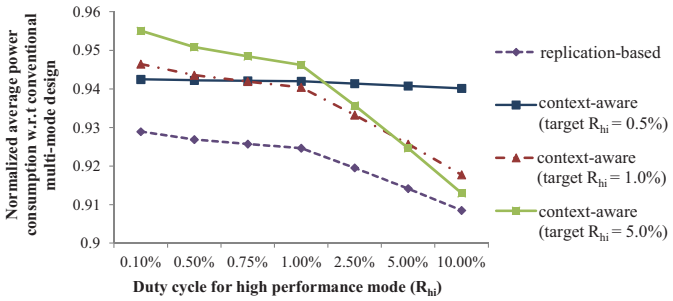


Fig. 11. Normalized average power consumption when the duty cycle of high-performance mode (R_{hi}) varies from the value targeted at design time (target R_{hi}), $X = 5$.

replicating the entire encompassing module. Also, fine-grain replication can provide average power reduction even for small allowable area overheads, whereas coarse-grain replication must overcome an initial area hurdle before any module can be replicated. One disadvantage of fine-grain replication is increased overhead for power gating and MUX logic, which can prevent several small leaf modules from achieving replication benefits.

As discussed in Section II-B, a heterogeneous multicore processor can also target multiple power and performance points. In Fig. 9, we compare power reduction and area overhead among heterogeneous multicore, context-aware, and selective replication-based designs. All designs target high-performance and low-power modes. Selective-replication-based design achieves energy efficiency benefits within 2% of the heterogeneous multicore with a significantly smaller area overhead.

While area, power, and timing overheads of replication are carefully modeled in the above studies, replication also affects the physical layout of the design, particularly in the neighborhood of the replicated module. Fig. 10 shows layouts for conventional multimode design and selective replication-based design for the FFU module. In the selective replication-based implementation, the CTL module has been replicated, affording 12% average power savings with 14% area overhead. This result demonstrates the feasibility of the proposed replication-based approach described in Fig. 6, in spite of the potential layout complications.

D. Variation Analysis

In general, it may not be possible to estimate duty cycle (R) precisely for all users of a particular DVFS product. Fig. 11 shows how average power savings may be affected when R_{hi} differs from the average value of R_{hi} targeted at design time.

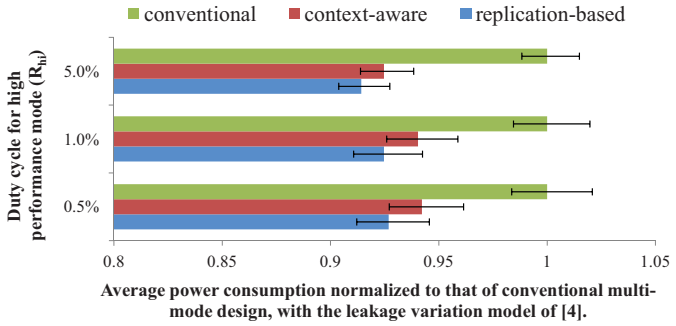


Fig. 12. Normalized average power consumption with leakage variations ($R_{\text{hi}} = \{0.5\%, 1\%, 5\%\}$, $X = 5$).

In these experiments, average power savings are reduced by at most 3% in the case where target R_{hi} is maximum (5%) and duty cycle is 50x lower. This is because when actual duty cycle is lower than the target, the resulting design is over-optimized for the user scenario and exhibits area and leakage overhead. Note that duty cycle variation does not affect the efficiency of a selective replication-based design.

We also evaluate the impact of variations, including systematic and random within-die (WID) variations on our DVFS designs. Since our DVFS designs scale between fixed operating points, and these operating points must be defined based on worst case corners, variations do not affect the voltage or frequency at which our designs operate. Variations can, however, affect power consumption, primarily in terms of leakage power (e.g., due to changes in threshold voltage and gate length). While we present results for 65-nm process technology, it is well known that in recent and future technology nodes (e.g, 32-nm, 22-nm), static power constitutes a larger fraction of total power, and can vary more substantially (e.g., in response to temperature changes). Analyzing the impact of leakage variations may be particularly relevant for context-aware design, where design optimization accounts for the relative contribution of leakage in different operating modes. The impact of variations may be felt most prominently for low R_{hi}, when leakage accounts for a larger fraction of total power. Following the methodology of [23], we model WID variations, including lithography-induced systematic WID variations. We use standard deviation (σ) of leakage for each standard cell following [23] and repeat power analysis for 1000 different variation maps, recording the average power in each trial. Note that to model lithography-induced, pattern-dependent WID variations, instances of the same standard cell have the same systematic WID variations for a given Monte Carlo trial. Fig. 12 compares average power consumption for the processor observed during variation analysis for different multimode design styles. Error bars show the min and max average power observed during Monte Carlo analysis. For designs with low R_{hi}, where leakage power variations impact total power more significantly, we see that variations impact average power savings by less than 2%.

E. Impact of Microarchitecture on DVFS Efficiency

In Section III, and in the results above (Table IV), we observe that the benefits of context-aware multimode design depend on factors such as the relative amount of sequential logic in a design and the tightness of the timing constraints. Since microarchitecture can influence these factors, DVFS energy efficiency can potentially be enhanced through microarchitectural adaptations. To gauge the potential impact of microarchitecture on DVFS efficiency, we evaluate the effectiveness of context-aware multimode design for different microarchitectural adaptations; namely, pipeline depth, superscalar width, and queue sizes in the Issue and LSU stages.

TABLE IX
PERCENT REDUCTION IN AVG. POWER FROM INCREASING PIPELINE DEPTH FROM 1 TO 3 FOR DIFFERENT SCENARIOS AND SUPERSCALAR WIDTHS

Width	X	$R = 0.5\%$	$R = 1\%$	$R = 10\%$	$R = 50\%$
1	2	1.6	1.5	3.3	8.1
	4	0.8	0.4	2.8	10.1
	10	5.8	4.0	4.6	11.3
2	2	6.9	6.8	6.3	5.1
	4	0.4	1.1	7.1	6.1
	10	1.0	1.6	7.0	5.5
4	2	4.2	4.3	5.1	11.8
	4	5.3	5.1	4.6	13.0
	10	7.7	7.4	8.4	14.0

1) Pipeline Depth and Logic Complexity: To evaluate the effects of changing the pipeline depth and superscalar width, we use FabScalar to generate six versions of the issue stage of a superscalar processor, with pipeline depths $\in \{1, 3\}$ and superscalar widths $\in \{1, 2, 4\}$. Table IX shows how increasing the pipeline depth from 1 to 3 affects the average power of various context-aware multimode designs with different superscalar widths in different scenarios.

Table IX shows that deeper pipelining in a context-aware multimode design improves energy efficiency more for designs with higher R, X, and logic complexity (e.g., superscalar width). Higher R means that high-performance mode is weighted more in the energy metric. Deeper pipelining allows context-aware multimode design to reduce voltage significantly (10%) at high frequency, where the impact of voltage (dynamic power) reduction is more pronounced, providing benefits for designs with high R. At low frequency and voltage, delay sensitivity to voltage is much higher, and pipelining does not allow our DVFS design flow to achieve significant voltage reduction. Thus, the energy efficiency of context-aware multimode design does not improve much with pipelining in scenarios with low R.

Designs that have more complexity (e.g., higher superscalar width) or shallower pipelines have deeper logic depth and higher fanout. For such designs, the difference between high and low performance targeted netlists is considerable, especially for larger X, since paths with deep logic, high fanouts, and tight timing constraints require larger, more leaky cells, more buffering, etc. to meet tight timing constraints. Deeper pipelining provides more benefits in such scenarios, as it relaxes tight constraints, allowing a multimode design to meet timing in multiple modes with considerably less overhead, especially when the range of scalability (X) is large.

In typical circuits, like the FabScalar test cases discussed above, perfect pipelining is not possible due to logic complexity. To investigate the potential influence of pipelining on DVFS efficiency, we have created a generic test circuit that can be subdivided cleanly. Fig. 13 shows the different implementations of the pipelining test circuit. The basic, single-stage design (a) is created by cascading four 8-bit multipliers end-to-end between a pair of latches. Two-stage (b) and four-stage (c) versions of the circuit are created by

TABLE X
POWER COMPARISON FOR DVFS IN EACH PIPELINED MULTIPLIER IMPLEMENTATION

Pipeline	1-stage			2-stage			4-stage		
Operating Frequency (MHz)	Operating Voltage (V)	Total Power (W)	Leakage Power (%)	Operating Voltage (V)	Total Power (W)	Leakage Power (%)	Operating Voltage (V)	Total Power (W)	Leakage Power (%)
1000	N/A	N/A	N/A	1.16	5.15E-04	3.4	0.84	3.52E-04	3.0
500	N/A	N/A	N/A	0.76	1.09E-04	4.7	0.65	1.07E-04	5.2
200	0.96	4.53E-05	4.9	0.58	2.40E-05	9.3	0.53	2.99E-05	11.3
100	0.74	1.37E-05	7.7	0.51	1.06E-05	15.3	0.47	1.29E-05	19.9
50	0.62	5.04E-06	13.5	0.45	5.46E-06	21.8	0.41	6.10E-06	31.0

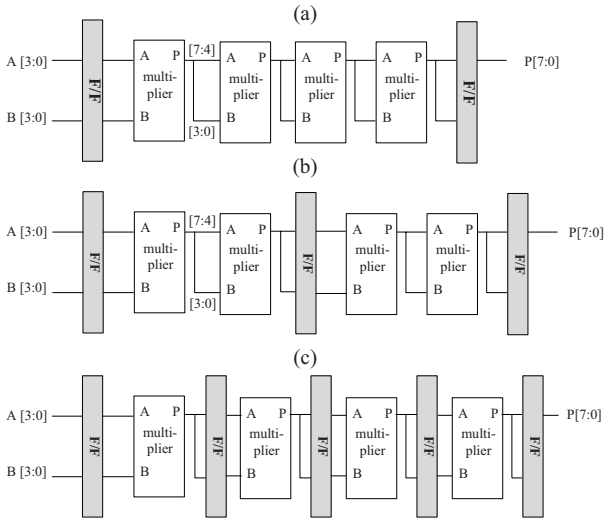


Fig. 13. Pipeline test circuit composed of cascaded multiplier blocks for pipeline depths of (a) 1, (b) 2, and (c) 4.

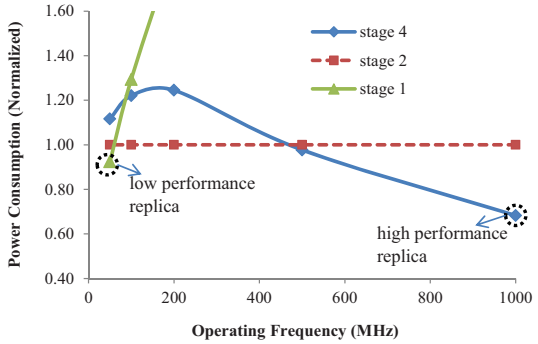


Fig. 14. Power consumption of each pipelined multiplier (normalized to the 2-stage implementation). The minimum-energy implementation depends on the dominant performance mode.

placing pipeline latches at the appropriate junctions between multipliers.

Table X shows the DVFS power versus performance trade-off for the pipelined multiplier test case. Interestingly, each pipeline implementation is the minimum energy design over some portion of the frequency scaling range. At high frequency, the deepest pipeline (4-stage) consumes the least average power, since a large fraction of total power is dynamic power. While the 1-stage design cannot operate at 1 GHz for any available voltage, the deeper pipelining of the 4-stage design enables a voltage reduction of 28% with respect to the 2-stage design, significantly reducing dynamic power consumption. The 4-stage design remains the minimum power design as frequency is scaled down to 500 MHz. All the while, the voltage differential between the 4-stage and 2-stage designs shrinks, due to increasing delay sensitivity to voltage. Also, as frequency is scaled down, leakage power accounts for a larger fraction of total power. Leakage increases faster for deeper pipelines due to increased latch area. By 200 MHz, the voltage advantage of the 4-stage design is only 9%, and reduced area and leakage due to fewer pipeline latches make the 2-stage design more efficient. Scaling down to 50 MHz, the 1-stage design takes over as most energy efficient.

As demonstrated in Table IX and X, DVFS efficiency depends on microarchitectural parameters, including pipeline depth. Thus, pipeline depths for the modules in a selective replication-based DVFS design should be chosen appropriately to minimize average power. For a design that weights high-performance mode more heavily (high R), a deeper pipeline is beneficial, since it allows more voltage scaling for reduced dynamic power. For a design that weights low performance more heavily (low R), a shallower pipeline is beneficial for area and leakage reduction. Consequently, duty cycle (R) plays a strong role in determining the optimal pipeline depth of a multimode design. As R increases, optimal pipeline depth also increases. For a replicated module in a selective replication-based design, high and low-performance replicas may have different energy-optimal pipeline depths. Thus, module replicas, though identical in functionality, may not have identical implementations. In addition to optimizing the design level implementation of each replica for the performance target, the microarchitecture-including pipeline depth-should also be optimized. For the multiplier at $R = 1\%$, $X = 10$, a replication-based design that optimizes the pipeline depths of each replica has 14% lower average power than a replication-based design without microarchitectural optimization. Average power reduction is 9.5% with respect to a context-aware multimode design. Fig. 14 shows normalized average power consumption for each pipelined multiplier implementation, demonstrating that the optimal pipeline depth for a DVFS design depends on the dominant performance mode.

2) Microarchitectural Structure Sizing: Besides pipeline depth, we also investigate the effect on DVFS efficiency of changing the sizes of microarchitectural structures in the processor. We evaluate the Issue stage of the FabScalar processor for issue queue (IQ) lengths $\in \{16,\ 32\}$. We also evaluate the LSU of the processor for load and store queue lengths $\in \{8,\ 16\}$. Note that the total length of the load store queue (LSQ) is the length of the load queue plus the length of the store queue $\{16,\ 32\}$. In both cases, we observe the effect of changing the superscalar width for widths $\in \{1,\ 2,\ 4\}$. Superscalar width and IQ and LSQ lengths have a significant impact on processor complexity [24], and consequently, affect several design characteristics that influence DVFS efficiency, as outlined in Section III.

In order to isolate the effect of these microarchitectural optimizations on DVFS efficiency, we compare the average

TABLE XI
PERCENT AVERAGE POWER OVERHEAD WITH RESPECT TO IDEAL FOR THE LSU STAGE FOR DIFFERENT SCENARIOS, LSQ LENGTHS, AND SUPERSCALAR WIDTHS

		$LSQ\ LENGTH = 16$				$LSQ\ LENGTH = 32$			
W	X	$R =$ 0.5%	$R =$ 1%	$R =$ 10%	$R =$ 50%	$R =$ 0.5%	$R =$ 1%	$R =$ 10%	$R =$ 50%
1	2	0.3	0.7	4.0	7.9	6.0	6.0	6.9	0.7
	4	2.8	3.7	11.9	9.0	6.0	6.1	7.4	2.6
	10	7.1	9.7	14.8	11.1	8.8	6.8	4.9	2.4
2	2	3.7	4.4	8.7	13.7	7.5	7.4	5.6	2.6
	4	5.4	6.8	20.2	31.7	7.3	7.2	2.6	0.7
	10	10.0	13.1	34.9	39.4	10.2	10.0	1.4	0.3
4	2	6.0	6.5	13.7	1.9	1.0	1.0	2.5	0.7
	4	8.4	10.4	19.5	1.3	8.0	8.1	2.5	1.1
	10	12.2	16.7	28.0	0.7	10.1	10.5	3.8	0.5

TABLE XII
PERCENT AVERAGE POWER OVERHEAD WITH RESPECT TO IDEAL FOR THE ISSUE STAGE FOR DIFFERENT SCENARIOS, IQ LENGTHS, AND SUPERSCALAR WIDTHS

		$IQ\ LENGTH = 16$				$IQ\ LENGTH = 32$			
W	X	$R =$ 0.5%	$R =$ 1%	$R =$ 10%	$R =$ 50%	$R =$ 0.5%	$R =$ 1%	$R =$ 10%	$R =$ 50%
1	2	1.0	1.2	4.0	9.5	1.4	1.7	4.8	3.7
	4	8.4	8.8	7.4	2.2	4.7	5.2	10.2	2.3
	10	6.1	7.3	11.7	3.9	13.2	13.6	13.5	2.1
2	2	1.3	1.5	3.7	3.8	0.2	0.4	3.2	0.3
	4	6.3	6.8	10.6	2.4	11.0	11.2	4.9	1.3
	10	6.7	7.6	6.7	1.1	16.1	18.3	7.0	1.2
4	2	0.1	0.2	2.4	0.1	0.1	0.2	1.6	0.2
	4	11.3	10.4	8.5	1.9	5.8	5.9	7.3	1.7
	10	14.5	14.0	6.2	1.5	11.4	10.9	6.0	1.0

power consumption of the context-aware multimode design for each scenario against the average power of an ideal design. Ideal average power in each scenario is computed using the power of the minimum power targeted design in each mode.

Table XI shows the average power overhead with respect to ideal average power in different scenarios for the LSU stage with different LSQ lengths and superscalar widths. Averaged over all scenarios and superscalar widths, the larger LSQ has 5% overhead with respect to ideal, compared to 11% for the smaller LSQ. This is partially because the LSU has a large %seq (40%–50%), and increasing the size of the LSQ further increases %seq. Due to this and the increased logic complexity incurred by the larger LSQ, the difference between high and low-performance designs is less for the larger LSQ. Thus, DVFS energy efficiency does not suffer considerably when the LSQ size increases. With the smaller LSQ, there is more variation between targeted designs for each performance mode, resulting in more DVFS inefficiency, especially for larger X, where the difference is stressed. Fig. 15 demonstrates that the inefficiency of the designs with the smaller LSQ increases significantly with X, while the inefficiency of the designs with the larger LSQ stays fairly constant, around 5%.

Table XII shows the average power overhead with respect to ideal in different scenarios for the Issue stage with different IQ lengths and superscalar widths. The Issue stage has a significantly larger fraction of combinational logic (lower %seq) than the LSU. Thus, increasing the size of the IQ can result in a significant difference between the optimal designs for high and low performance, especially when superscalar width is low (since this exaggerates the impact of IQ length on design complexity). This also means that the effect of increasing X is amplified when the IQ is larger, as shown in Fig. 16. Based on the above, we observe that reducing the sizes of microarchitectural structures that have low %seq (e.g., IQ) improves DVFS energy efficiency when range of scalability is large.

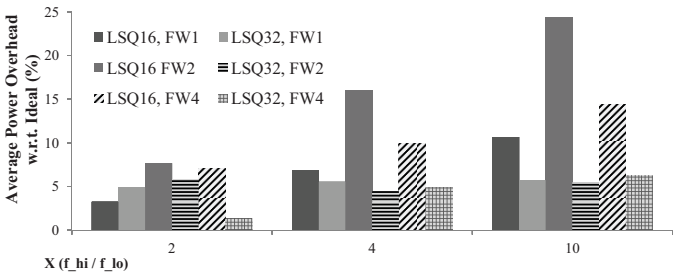


Fig. 15. DVFS inefficiency increases more significantly with X for the LSU with a smaller LSQ.

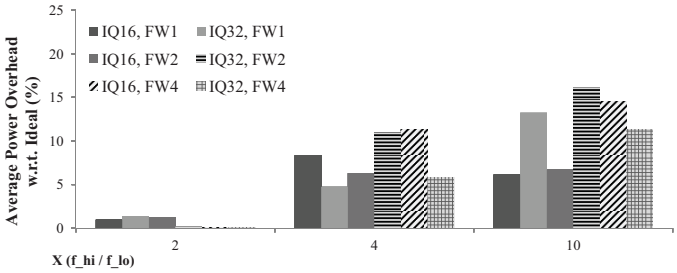


Fig. 16. DVFS inefficiency increases more significantly with X for the Issue stage with a larger IQ, especially when superscalar width is low.

The results above demonstrate that adapting the microarchitecture of a DVFS design can potentially improve DVFS efficiency. Note that we explore only a few microarchitectural features and primarily consider the design-level efficiency implications of microarchitectural decisions; other considerations are beyond the scope of this paper. For example, our results do not reflect the potentially increased cost of hazard recovery due to increased pipeline depth. A more thorough investigation of the effects of microarchitecture on DVFS efficiency at the system level is the subject of ongoing work.

VII. CONCLUSION

DVFS is a popular technique for reducing power and energy consumption under dynamic operating conditions by targeting multiple power and performance modes in a single design. In this paper, we demonstrated that DVFS-based designs obtained with conventional CAD methodologies can be energy inefficient. This may be especially true for energy-constrained designs that spend a large fraction of time in a low-power mode. We identified the different factors that impact DVFS efficiency. Based on our insights, we proposed a new approach to optimize for DVFS—context-aware multi-mode design—that considers the operating scenario to constrain and optimize a multimode design for improved energy efficiency. We also identified operating scenarios in which even an efficient multimode design exhibits substantial energy overhead with respect to the ideal energy consumption. For such operating scenarios, we proposed a selective replication-based approach that maximizes energy efficiency while minimizing area overhead. We demonstrated that context-aware and selective replication-based design can provide up to 25% average power reduction with respect to conventional multimode design. Average power for the optimized design is within 1% of ideal, on average.

Finally, we showed that microarchitectural optimizations influence DVFS efficiency and demonstrate up to 18% average power reduction by optimizing processor microarchitecture for DVFS efficiency. Ongoing work includes continuing to investigate the connection between microarchitecture and DVFS efficiency. Future work includes exploring the implications of our techniques in design for yield – e.g., by improving the energy efficiency of processor designs that are binned according to their post-manufacturing characteristics and operate at a voltage or frequency that is only determined after manufacturing.

REFERENCES

[1] *International Technology Roadmap for Semiconductors*. (2009) [Online]. Available: http://www.itrs.net/

[2] L. Benini, A. Bogliolo, and G. De Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 8, no. 3, pp. 299–316, Jun. 2000.

[3] R. Kumar, K. I. Farkas, N. P. Jouppi, P. Ranganathan, and D. M. Tullsen, "Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction," in *Proc. Int. Symp. Microarch.*, 2003, pp. 81–92.

[4] A. Shye, B. Ozisikyilmaz, A. Mallik, G. Memik, P. Dinda, R. Dick, and A. Choudhary, "Learning and leveraging the relationship between architecture-level measurements and individual user satisfaction," in *Proc. Int. Symp. Comput. Arch.*, 2008, pp. 427–438.

[5] *Synopsys Power Compiler User's Manual* [Online]. Available: http://www.synopsys.com/

[6] *Cadence SoC Encounter User's Manual* [Online]. Available: http://www.cadence.com/

[7] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor, "Conservation cores: Reducing the energy of mature computations," in *Proc. Int. Conf. Arch. Support Program. Lang. Operat. Syst.*, Mar. 2010, pp. 205–218.

[8] N. Goulding-Hotta, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, P. Huang, M. Arora, S. Nath, V. Bhatt, J. Babb, S. Swanson, and M. Taylor, "The GreenDroid mobile application processor: An architecture for silicon's dark future," *IEEE Micro*, vol. 31, no. 2, pp. 86–95, Mar.–Apr. 2011.

[9] G. Dhiman, K. Pusukuri, and T. Rosing, "Analysis of dynamic voltage scaling for system level energy management," in *Proc. USENIX Workshop Power Aware Comput. Syst.*, 2008, p. 9.

[10] O. Azizi, A. Mahesri, B. Lee, S. Patel, and M. Horowitz, "Energy-performance tradeoffs in processor architecture and circuit design: A marginal cost analysis," in *Proc. Int. Symp. Comput. Arch.*, 2010, pp. 26–36.

[11] B. H. Calhoun and A. P. Chandrakasan, "A 256-kb 65-nm sub-threshold SRAM design for ultralow-voltage operation," *IEEE J. Solid-State Circuits*, vol. 42, no. 3, pp. 680–688, Mar. 2007.

[12] K. Fujita, Y. Torii, M. Hori, J. Oh, L. Shifren, P. Ranade, M. Nakagawa, K. Okabe, T. Miyake, K. Ohkoshi, M. Kuramae, T. Mori, T. Tsuruta, S. Thompson, and T. Ema, "Advanced channel engineering achieving aggressive reduction of V_T variation for ultralow-power applications," in *Proc. IEEE Int. Electron. Devices Meeting*, Mar. 2011, pp. 32.3.1–32.3.4.

[13] Intel Corporation. (2008). *Intel Atom Processor Z5xx Series*, Santa Clara, CA [Online]. Available: http://versalogic.com/support/Downloads/PDF/Intel_Atom_Datasheet.pdf

[14] D. Pisinger, "A minimal algorithm for the multiple-choice knapsack problem," *Eur. J. Operat. Res.*, vol. 83, no. 2, pp. 394–410, Jun. 1995.

[15] *Sun OpenSPARC Project* [Online]. Available: http://www.sun.com/processors/opensparc/

[16] N. Choudhary, S. Wadhavkar, T. Shah, S. Navada, H. Najaf-Abadi, and E. Rotenberg, "FabScalar," in *Proc. Workshop Arch. Res. Prototyp.*, 2009, pp. 1–12.

[17] *Cadence LC User's Manual* [Online]. Available: http://www.cadence.com/

[18] *Synopsys Design Compiler User's Manual* [Online]. Available: http://www.synopsys.com/

[19] A. B. Kahng and S. Mantik, "Measurement of inherent noise in EDA tools," in *Proc. Int. Symp. Quality Electron. Design*, 2002, pp. 206–211.

[20] K. Jeong and A. B. Kahng, "Methodology from chaos in IC implementation," in *Proc. Int. Symp. Quality Electron. Design*, 2010, pp. 885–892.

[21] *Synopsys Prime Time User's Manual* [Online]. Available: http://www.synopsys.com/

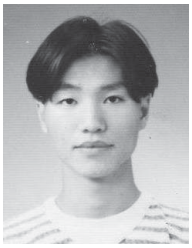
[22] M. Qazi, M. Sinangil, and A. Chandrakasan, "Challenges and directions for low-voltage SRAM," *IEEE Trans. Des. Test Comput.*, vol. 28, no. 1, pp. 32–43, Jan.–Feb. 2011.

[23] K. Cao, S. Dobre, and J. Hu, "Standard cell characterization considering lithography induced variations," in *Proc. Design Autom. Conf.*, 2006, pp. 801–804.

[24] S. Palacharla, N. Jouppi, and J. Smith, "Complexity-effective superscalar processors," in *Proc. Int. Symp. Comput. Arch.*, 1997, pp. 206–218.

Andrew B. Kahng (F'10) received the Ph.D. in computer science from the University of California at San Diego (UCSD), La Jolla, in 1989.

He was with the Computer Science Department, University of California at Los Angeles, Los Angeles, from 1989 to 2000. Since 2001, he has been with the Department of Computer Science and Engineering and the Department of Electrical and Computer Engineering, UCSD, where he holds the endowed chair in high-performance computing. He has authored or co-authored more than 350 journal and conference papers, and three books. He holds 22 U.S. patents. His current research interests include IC physical design, the design-manufacturing interface, and combinatorial algorithms and optimization.



Seokhyeong Kang (S'11) received the B.S. and M.S. degrees in electrical engineering from the Pohang University of Science and Technology, Pohang, Korea, in 1999 and 2001, respectively. He is currently pursuing the Ph.D. degree with the VLSI CAD Laboratory, University of California at San Diego, San Diego.

He was with the System-on-Chip (SoC) Development Team, Samsung Electronics, Suwon, Korea, from 2001 to 2008, where he was involved in development and commercialization of optical disk drive SoC. His current research interests include low-power design optimization and cost-driven methodology for chip implementation.

Rakesh Kumar (M'07) received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology Kharagpur, Kharagpur, India, and the Ph.D. degree in computer engineering from the University of California at San Diego (UCSD), San Diego, in 2001 and 2006, respectively.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana. In 2007, he was a Visiting Researcher with Microsoft Research, Redmond, WA. His current research interests include reliable and low-power computing.

Dr. Kumar was the recipient of several awards, including the IBM Ph.D. Fellowship in 2005, the UCSD CSE Best Dissertation Award in 2007, the FAA Creative Research Award in 2008, the Arnold O. Beckman Research Award in 2009, the ARO Young Investigator Award in 2012, and two Best Paper Awards. Other recognitions include four keynotes and over twenty invited lectures at conferences and workshops. He was an invited Guest Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and the IEEE EMBEDDED SYSTEMS LETTERS.

John Sartori (S'03) received the B.S. degree in electrical engineering, computer science, and mathematics from the University of North Dakota, Grand Forks, in 2006, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Urbana, in 2010 and 2012, respectively.

Dr. Sartori recently joined the Faculty at the University of Minnesota, Twin Cities. His research has garnered recognition, including a Best Paper Award and an Intel Computer Engineering Fellowship in 2012.