Total Energy Minimization of Real-Time Tasks in an On-Chip Multiprocessor Using Dynamic Voltage Scaling Efficiency Metric

Hyunjin Kim, Hyejeong Hong, Hong-Sik Kim, Jin-Ho Ahn, and Sungho Kang, *Member, IEEE*

***Abstract*—This paper proposes an algorithm that provides both dynamic voltage scaling and power shutdown to minimize the total energy consumption of an application executed on an on-chip multiprocessor. The proposed algorithm provides an extended schedule and stretch method, where task computations are iteratively stretched within the slack of a time-constrained dependent task set. In addition, the break-even threshold interval for amortizing the shutdown overhead is considered. By evaluating each set of stretched task computations, an energy-efficient set is obtained. The proposed dynamic voltage scaling efficiency metric is the ratio of the reduced energy to the increased cycle time when the supply voltage is scaled, which can be used to determine the task computation cycle to be stretched. Experimental results show that the proposed algorithm outperforms the traditional schedule and stretch method in the various evaluations of target real applications.**

***Index Terms*—Dynamic voltage scaling (DVS), energy-aware systems, load balancing and task assignment, on-chip multiprocessor, power management, processor scheduling.**

I. Introduction

Minimizing energy consumption has become a primary concern in the state-of-the-art on-chip multiprocessors to extend the battery life in embedded systems. Dynamic voltage scaling (DVS) and power shutdown (PS) are known as the main high-level task computation energy-saving techniques. The DVS technique scales the voltage swing and the operation frequency of voltage-variable processing elements (PEs) or cores on the fly; the PS technique shuts down the power of each PE or makes the PE dormant [1]. For on-chip multiprocessors, the DVS and PS techniques can be combined by adopting these techniques at the same time. For an example of the AMD's Opteron multiprocessor [2], a power scheduler can shutdown each core independently, and the supply voltage can be scaled by controlling an off-chip voltage regulator.

Several previous works related to energy-saving techniques using DVS on multiprocessor systems concentrated on the energy savings for independent tasks running on an on-chip multiprocessor (such as online scheduling) [3], [4] or dependent tasks running on an off-chip multiprocessor (such as offline scheduling) [5], [6]. Both the leakage-aware multiprocessor scheduling and the schedule and stretch (S&S) described in [7] and [8] applied DVS to dependent tasks on an on-chip multiprocessor by evenly distributing the remaining time before the deadline (slack). Moreover, the DVS efficiency of stretched task computations and the PS overhead were not considered. In addition, the approaches described in [7]–[9] concentrated on coarse-grained DVS techniques, where the optimal number of PEs was obtained to minimize total energy consumption. The energy model shown in [10] did not consider PS separately. The leakage-aware energy optimization proposed in [11] focused on the adaptive body biasing. Several previous works assumed that the supply voltage and operation frequency could be variable in each core [12]–[14] or the hardware area could be partitioned into several operation frequency islands [15].

This paper proposes an algorithm that uses both the DVS and PS techniques to minimize the total energy consumption of an application running on an on-chip multiprocessor with homogeneous voltage-variable PEs. The proposed algorithm provides an extended S&S, where the task computation cycles are iteratively stretched within the slack of a time-constrained dependent task set. In addition, the minimal break-even threshold interval of PS is proposed to amortize the energy and the time overhead of PS. The iterative stretches are evaluated to obtain the most energy-efficient set of stretched task computations. The DVS efficiency metric is proposed to evaluate the energy efficiency of the reduced task computation energy over the stretched task computation cycles, where the supply voltage and operation frequency for the task computation cycle with the highest DVS efficiency are lowered. We implement two versions of the extended S&S with and without adopting the DVS efficiency metric, so that the effectiveness of the extended S&S and the DVS efficiency metric can be analyzed. The extended S&S without adopting the DVS efficiency metric provides the iterative stretches and evaluations considering PS on the fly, but the slack is distributed evenly like the traditional S&S. Experimental results show that the proposed algorithm outperforms the traditional S&S in the various evaluations of real applications.

This paper is structured as follows. In Section II, several models and definitions are provided. In Section III, the proposed algorithm is discussed in more detail. In Section IV, experimental results are provided. Finally, our conclusions in this paper are summarized.

II. Models and Definitions

A. Task Model

The application running on a multiprocessor system is modeled as a directed acyclic *communication task graph* (CTG) [5], [6]. In a CTG, every task noted as a vertex has time constraints for the task-finishing deadline and task computation cycles. The communication noted as an edge represents the dependence between tasks. A task cannot be computed before finishing its predecessors. The task computation cycles of each task are fixed; the computation time can be changed according to the operation frequency.

B. Hardware Model

Considering the specifications of chip multiprocessors in [16], the majority are symmetric, so that it is assumed that N homogeneous PEs can be communicated with each other. Compared to the off-chip multiprocessor, the communication overhead can be reduced in the on-chip multiprocessor, as explained in [17]. Therefore, the communication energy and time overhead are not considered. A task should be computed on one PE. Due to the area and cost limitation of the off-chip regulator in embedded or mobile systems, all PEs are assumed to be powered from one off-chip regulator, so that supply voltage levels applied to PEs are equal at the same time. In addition, the same operation frequency is applied to every PE due to the sameness of supply voltage levels. Consequently, applying the same frequency level does not require independent clock distributions and complex synchronization circuits in an on-chip multiprocessor.

Manuscript received January 29, 2008; revised April 18, 2008 and July 1, 2008. Current version published October 22, 2008. This paper was recommended by Associate Editor D. Atienza.

H. Kim, H. Hong, H.-S. Kim, and S. Kang are with the Computer System and Reliable SOC Laboratory, Department of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, Korea (e-mail: hyunjin2.kim@gmail.com; hjhong@soc.yonsei.ac.kr; hongsik@yonsei.ac.kr; shkang@yonsei.ac.kr).

J.-H. Ahn is with the Department of Electronic Engineering, Hoseo University, Asan 336 795, Korea (e-mail: jhahn@hoseo.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/TCAD.2008.2006094

C. Power Model

This paper adopts the normalized power model shown in [7], which is based on the power model in [18]. Let $f_{\max}$ and $V_{\max}$ denote the maximum operation frequency and its corresponding supply voltage, respectively. In addition, δ and σ denote the ratio of the dynamic power consumption and the static power consumption to the total power consumption at $f_{\max}$. With the normalized frequency $f_{\text{norm}} = f/f_{\max}$ and the normalized voltage $V_{\text{norm}} = V_{\text{dd}}/V_{\max}$, the normalized voltage described in [7] and [18] is given by

$$V_{\text{norm}} = \beta_1 + \beta_2 f_{\text{norm}} \tag{1}$$

where $\beta_1 = V_{\text{th}}/V_{\max}$ and $\beta_2 = 1 - \beta_1$. The normalized energy consumption function, which means the normalized total energy consumption required in one cycle time, is given by

$$E(f_{\text{norm}}) = \delta(\beta_1 + \beta_2 f_{\text{norm}})^2 + \sigma(\beta_1/f_{\text{norm}} + \beta_2). \tag{2}$$

The normalized cycle time is defined as $C_{\text{norm}} = 1/f_{\text{norm}}$. In (2), δ, σ, β_1, and β_2 are predetermined based on the semiconductor technology constants. Even though V_{th} increases with V_{dd} as shown in [19], the increase is neglected due to its insignificant magnitude.

PS completely eliminates both the dynamic and the static power consumption because tasks cannot be computed and no power is supplied, whereas DVS reduces only the dynamic power consumption while computing tasks on the fly. The normalized operation frequency domain is discrete since both the number and the resolution of available supply voltage levels from an off-chip voltage regulator are limited. In addition, both the time and the energy overhead of PS are considered in this paper. The time and the energy overhead of DVS due to the voltage transitions, however, are not explicitly considered to obtain the total energy consumption because the operation frequency transitions between task computation cycles are minimized and the number of transitions can be much smaller than the number of computation cycles.

D. Critical Speed and Shutdown Overhead

The critical speed described in [7] is the available normalized frequency that minimizes the total energy consumption needed in a clock cycle time. The critical speed f_{critical} can be obtained using the strict convexity of $E(f_{\text{norm}})$. First, the optimal normalized frequency f_{optimal} is obtained by solving $\nabla E(f_{\text{norm}}) = 0$. Then, the minimal value of the available normalized operation frequencies can be f_{critical}, which can be equal to or higher than f_{optimal}.

The total energy consumption needed in a cycle time can increase below the critical speed due to the increased static energy consumption. In this case, the choice whether the redundant slack is used to shutdown all PEs of the on-chip multiprocessor or maintain the supply voltage level of the critical speed is dependent on the energy efficiency of PS considering the PS overhead. In addition, nonactivated intervals of each PE can be used to shutdown the PE (each PE can be dormant) because each PE is controlled independently.

To determine whether making all PEs dormant is energy efficient or not, the break-even threshold interval for PS $T_{\text{threshold}}(\text{PS})$ is provided by

$$T_{\text{threshold}}(\text{PS}) = \max\left(\frac{E_{\text{overhead}}(\text{PS})}{P_{\text{dc}}(\text{critical\_speed})}, T_{\text{overhead}}(\text{PS})\right) \tag{3}$$

where $E_{\text{overhead}}(\text{PS})$, $P_{\text{dc}}(\text{critical\_speed})$, and $T_{\text{overhead}}(\text{PS})$ denote the normalized energy overhead of PS, the normalized static power consumption at the critical speed, and the normalized time overhead of PS, respectively. Both the time and the energy overhead are normalized with a cycle time at the maximum operation frequency and the maximum total energy consumption needed in the cycle time.

To evaluate the energy-saving efficiency in the dormant mode of each PE, the static energy consumption needed in the chained cycle time of the nonactivated PE is compared with $E_{\text{overhead}}(\text{PS})$, where making all PEs dormant is impossible due to the activation of other PEs in the middle of the chained cycles. The length of time required for the chained nonactivated cycles should be longer than the time overhead of PS.

E. DVS Efficiency Metric

The DVS efficiency metric is proposed to determine the cycle to be stretched. The DVS efficiency metric represents the ratio of the total energy savings to the increased cycle time when DVS is applied to all activated PEs and the operation frequency is lowered. By using (2), the formulation and concept of the DVS efficiency metric are introduced. Starting from an operation frequency, DVS increases the cycle time, along with decreasing the energy consumption needed in one cycle and the slack. Let the number of activated PEs at the cth cycle be denoted by $N_A(c)$. With the normalized operation frequencies f_1, f_2 and their cycle times C_1, C_2, the DVS efficiency metric obtained from the noninfinitesimal differentiation is formulated by

$$N_A(c) \cdot \left(-\frac{E(f_1) - E(f_2)}{C_1 - C_2}\right), \qquad \text{where} \quad f_1 > f_2. \tag{4}$$

Considering the strict convexity of the DVS efficiency metric in (4), DVS starting at the maximum normalized operation frequency is the most energy efficient. For task computation cycles with a different number of activated PEs, the number of activated PEs should be multiplied.

III. Proposed Algorithm

The outline of the proposed algorithm is as follows: After task scheduling, DVS is applied to the task computation cycle with the highest DVS efficiency. In the extended S&S, DVS is repeated until there is no available slack. In each stretch, the stretched task computations are evaluated and compared with the previous best solution. When the loop is finished, the best solution with the lowest total energy consumption is obtained. In the following, the proposed algorithm based on the outline will be explained in detail.

A. Task Scheduling

The proposed algorithm adopts the priority-based task scheduling described in [5], which attempted not to make unnecessary long paths (dependent sets of ordered tasks). Every task is scheduled by repetitions of the priority-based task selection, the task mapping, and the task ordering for every task. Except for the task scheduling adopted earlier, other task scheduling methods can be acceptable because applying DVS for an on-chip multiprocessor does not violate the dependences between tasks. In addition, the same operation frequency is shared by every PE in the on-chip multiprocessor, so that the voltage scaling can be performed after the task scheduling.

B. Voltage Scaling With DVS Efficiency Metric

By using a greedy approach, the task computation cycle with the highest DVS efficiency is determined and stretched. In Fig. 1, the scheduled tasks of a CTG are given and three homogeneous PEs

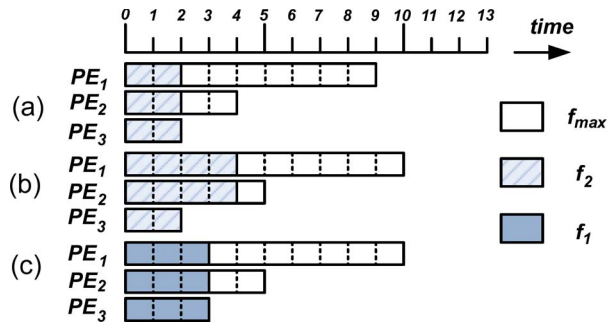

Fig. 1. Example of applying DVS to task computation cycles.

are assumed. For a set of nonincreasingly ordered three operation frequencies in Fig. 1, $F = \{f_{\max}, f_2, f_1\}$, the cycle times are assigned as one, two, and three time units, respectively. In the partially stretched task computation cycle shown in Fig. 1(a), the first task computation cycle with the highest DVS efficiency has been stretched, where three PEs are all activated. After obtaining the partial solution shown in Fig. 1(a), the next task computation cycle to be stretched is adopted after calculating the DVS efficiency for each task computation cycle. If $2 \cdot (E(f_{\max}) - E(f_2)) \geq 3 \cdot (E(f_2) - E(f_1))$, as shown in Fig. 1(b), the second cycle is preferred because the DVS efficiency of the second cycle is greater than that of the stretched first cycle; otherwise, DVS is applied to the first stretched cycle, as shown in Fig. 1(c). To minimize the transitions of the operation frequency, the task computation cycles with the same operation frequency should be chained. Therefore, even though the second and the third cycles in Fig. 1(b) can be stretched with the same operation frequency, the second task computation cycle is selected because applying DVS to the second cycle reduces the voltage transition.

C. Iterative Stretches Considering PS Overhead in Extended S&S

Even though task computation cycles are stretched by lowering the supply voltage and operation frequency, the total energy consumption, however, might not decrease due to the PS overhead or the static energy consumption. For example, if the break-even threshold interval of PS is assumed to be four time units in Fig. 1, the partially stretched task computations of PE_1 in Fig. 1(b) or Fig. 1(c) might not make PE_1 dormant at an interval (10–13) due to the time overhead of PS and might not be energy efficient due to the static energy consumption. Moreover, even though the dormant mode can be applied to the interval when the break-even threshold interval of PS is shorter than or equal to three time units, the partially stretched task computations might not be energy efficient due to the energy overhead of PS, which means that the stretched task computations cannot always minimize the total energy consumption when the PS overhead is considered. In the proposed extended S&S, therefore, the solution with the partially stretched task computation cycles is evaluated and then compared with the previous best solution. The comparison is also repeated until there is no available slack required to stretch task computations.

In spite of the repeated evaluation and comparison, there is no need to calculate the normalized energy consumption of all task computation cycles. When the DVS efficiency metric is adopted, the order of stretches can be determined before the voltage scaling. Due to the unidirectional reduction of the available slack, only the addition or the reduction of the energy consumption by the stretched cycle is required. Therefore, in terms of the number of evaluations, the proposed algorithm shows reasonable time complexity $O(C)$, where C denotes the total number of cycles of tasks in an application.

TABLE I
PROPERTIES OF CTGS FOR REAL APPLICATIONS

Name	#Tasks	#Edges	Deadline (NT)		
			tight	normal	loose
fpppp	334	1196	1593	2124	2655
robot	88	130	854	1138	1423
sparse	96	128	183	244	305

IV. EXPERIMENTAL RESULTS

The proposed algorithm was implemented by the C++ language, the boost graph library [20], and the standard template library [21]. For apple-to-apple comparisons in terms of the total energy consumption, two versions of the extended S&S with and without adopting the DVS efficiency metric and the traditional S&S were implemented.

A. Experimental Environments

In experiments, standard task graphs (STGs) [22] extracted from SPEC fpppp benchmark, robot control application, and sparse matrix controller were adopted as targeted CTGs. The sparse STG could provide high parallelism, so that the evaluation of sparse was performed on the large number of PEs. In robot, the maximum number of predecessors for any task was only three. On the other hand, the maximum number of predecessors was 81 in fpppp. Moreover, fpppp provided a large number of tasks more than 300.

All CTGs had three types of maximum deadlines: tight, normal, and loose. The deadlines were determined based on the critical path of each CTG. The number of tasks and edges, and the deadlines of the CTGs are listed in Table I, where dummy tasks and edges are not included.

The normalized time and the maximum energy consumption required for one activated PE in one cycle time at maximum operation frequency are denoted as NT and NE, respectively. NT and NE were used as the time and energy units. The technology-dependent parameters (δ, σ, β_1, and β_2) of the normalized energy consumption function were predetermined. Ten percent of the power consumption at maximum operation frequency was assumed to be the static power consumption ($\delta = 0.9$ and $\sigma = 0.1$). In addition, the threshold voltage was determined to be 0.3 times the maximum supply voltage ($\beta_1 = 0.3$ and $\beta_2 = 0.7$) based on the process data in [18]. The optimal normalized frequency is located between 0.3 and 0.2 (1/NT). In addition, the DVS efficiency when the normalized operation frequency is 0.3 (1/NT) is approximately zero. Therefore, a set of normalized operation frequencies $F_{\text{norm}} = \{1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3\}$ was adopted, where the critical speed was set as 0.3 (1/NT). The energy overhead of PS and the time overhead of PS were set as 10 (NE) and 10 (NT), respectively. Therefore, the break-even threshold interval of PS was 196.08 (NT). The values of the technology-dependent parameters, the set of the normalized operation frequencies, and the PS overhead mentioned earlier were used, unless noted otherwise.

B. Performance Evaluations

The number of PEs adopted should be determined according to the complexity of each application. Therefore, the number of PEs was determined considering the experimental results described in [7]. Table II shows the evaluation results for total energy consumption when the normal deadline was adopted. The number of PEs was set according to each real application; nine, seven, and nineteen PEs were adopted for fpppp, robot, and sparse, respectively.

In Table II, the total energy consumption is reduced by 51.01% on average, compared with the total energy consumptions when DVS and PS are not applied. On average, the energy saving was improved

TABLE II
TOTAL ENERGY CONSUMPTION WITH NORMAL DEADLINE

name	No DVS[1] (NE)	SS[2] (NE)	ESS[3] (NE)	ESS+ DVS[4] (NE)	Over SS[5] (%)	Over ESS[6] (%)
fpppp	8184.3	4853.3	4678.6	4477.4	7.75	4.30
robot	3310.1	1462.0	1373.1	1311.5	10.29	4.49
sparse	2368.2	1037.5	1037.5	1001.8	3.44	3.44
Avg.	4620.8	2450.93	2363.07	2263.57	7.16	4.08

[1]No DVS denotes the energy consumption when DVS and PS are not applied. [2]SS, [3]ESS, and [4]ESS+DVS denote the traditional S&S and the two versions of the extended S&S method with and without adopting DVS efficiency metric, respectively. In addition, [5]Over SS and [6]Over ESS represent the energy-saving improvements of ESS+DVS over SS and ESS, respectively.

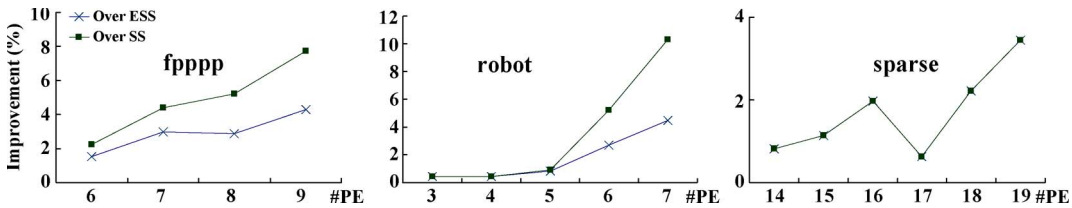

Fig. 2. Summary of energy-saving improvements by varying the number of PEs with normal deadline.

by 7.16% over the traditional S&S and 4.08% over the extended S&S without adopting the DVS efficiency metric. The energy-saving improvement of robot, which had the smallest number of PEs, was greatest.

As shown in Fig. 2, experiments were performed with the normal deadline by varying the number of PEs, where the number of PEs adopted for each application was also determined based on the experimental results in [7]. In general, the energy-saving improvements increased with growing the number of adopted PEs. The energy-saving improvements over the traditional S&S were always greater than or equal to those over the extended S&S without adopting the DVS efficiency metric. In particular, for sparse, the improvements over the traditional S&S were equal to those over the extended S&S without adopting the DVS efficiency metric, which means that the static energy savings by PS were equal.

When the same number of PEs was adopted, the energy savings for the tight deadline could be more improved than those for the normal or the loose deadline. For example, the energy saving for robot with the tight deadline was improved by 6.44%, whereas the energy savings for robot with the normal and loose deadlines were improved by 4.49% and 2.36%, respectively, compared to the extended S&S without adopting the DVS efficiency metric.

The experiments were performed by changing the ratio of the static power consumption at the maximum operation frequency, where we adopted three pairs with δ and σ: (0.9, 0.1), (0.7, 0.3), and (0.5, 0.5). Because the DVS efficiency and the critical speed are changed according to δ and σ, the set of normalized operation frequencies was changed. For side-by-side comparisons, both the operation frequency resolution and the PS overhead were maintained, so that the set of normalized operation frequencies for $\delta = 0.7$ and $\sigma = 0.3$ was $F_{\text{norm}} = \{1, 0.9, 0.8, 0.7, 0.6, 0.5\}$, where 0.5 was set as the critical speed. In addition, the set of normalized operation frequencies for $\delta = 0.5$ and $\sigma = 0.5$ was $F_{\text{norm}} = \{1, 0.9, 0.8, 0.7\}$, where 0.7 was set as the critical speed. The evaluations were performed with the normal deadline, where nine, seven, and nineteen PEs were adopted for fpppp, robot, and sparse, respectively. In Table III, the energy-saving improvements over the traditional S&S were increased with σ, whereas the energy-saving improvements over the extended S&S without adopting the DVS efficiency metric were decreased with σ.

TABLE III
AVERAGED TOTAL ENERGY CONSUMPTION WITH NORMAL DEADLINE ACCORDING TO (δ, σ)

(δ, σ)	No DVS (NE)	SS (NE)	ESS (NE)	ESS+ DVS (NE)	Over SS (%)	Over ESS (%)
(0.9,0.1)	4620.9	2450.9	2363.1	2263.5	7.2	4.1
(0.7,0.3)	5720.8	3327.3	2965.6	2927.2	10.3	0.7
(0.5,0.5)	6651.9	4748.2	3669.4	3668.7	23.7	0.0

These results indicate that the improvements over the traditional S&S can be obtained by reducing the static energy consumption using PS. On the other hand, the improvements over the extended S&S without adopting the DVS efficiency metric could not be enhanced due to the reduced DVS efficiency in lower operation frequencies.

V. CONCLUSION

This paper proposes an algorithm to minimize the total energy consumption of an application being executed on an on-chip multiprocessor. The energy efficiency of DVS is maximized using the proposed DVS efficiency metric. In the extended S&S method, using the iterative stretches of task computations and energy evaluations including PS, the best solution with the lowest total energy consumption can be obtained. In addition, the break-even threshold interval for amortizing the shutdown overhead is proposed. The experiments were performed for targeted real applications, where the proposed algorithm outperformed the traditional S&S in various evaluations. Considering the normalized energy consumption function, it is concluded that the extended S&S using the DVS efficiency metric must be useful for improving the energy-saving efficiency.

REFERENCES

[1] M. Keating *et al.*, *Low Power Methodology Manual: For System-on-Chip Design*. New York: Springer-Verlag, 2007.
[2] J. Dorsey *et al.*, "An integrated quad-core Opteron processor," in *Proc. Int. Solid State Circuits Conf.*, 2007, pp. 102–103.
[3] J. H. Anderson and S. K. Baruah, "Energy-efficient synthesis of periodic task systems upon identical multiprocessor platforms," in *Proc. Int. Conf. Distrib. Comput. Syst.*, 2004, pp. 428–435.

[4] C. Yang, J. Chen, and T. Kuo, "An approximation algorithm for energy-efficient scheduling on a chip multiprocessor," in *Proc. Des. Autom. Test Eur.*, 2005, pp. 468–473.
[5] Y. Zhang, X. S. Hu, and D. Z. Chen, "Task scheduling and voltage selection for energy minimization," in *Proc. Des. Autom. Conf.*, 2002, pp. 183–188.
[6] G. Varatkar and R. Marculescu, "Communication-aware task scheduling and voltage selection for total systems energy minimization," in *Proc. Int. Conf. Comput.-Aided Des.*, 2003, pp. 510–517.
[7] P. de Langen and B. Juurlink, "Leakage-aware multiprocessor scheduling for low power," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2006, pp. 8–15.
[8] P. de Langen, and B. Juurlink,"Trade-offs between voltage scaling and processor shutdown for low-energy embedded multiprocessors," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, ser. Lecture Notes in Computer Science, vol. 4599. New York: Springer-Verlag, 2007, pp. 75–85.
[9] V. W. Freeh, T. K. Bletsch, and F. L. Rawson, III, "Scaling and packing on a chip multiprocessor," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2007, pp. 1–8.
[10] J. Li and J. F. Martínez, "Dynamic power-performance adaptation of parallel computation on chip multiprocessors," in *Proc. Int. Symp. High-Performance Comput. Archit.*, 2006, pp. 77–87.
[11] A. Andrei, P. Eles, Z. Peng, M. T. Schmitz, and B. M. Al Hashimi, "Energy optimization of multiprocessor systems on chip by voltage selection," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 3, pp. 262–275, Mar. 2007.
[12] I. Kadayif, M. Kandemir, and I. Kolcu, "Exploiting processor workload heterogeneity for reducing energy consumption in chip multiprocessors," in *Proc. Des. Autom. Test Eur.*, 2004, pp. 1158–1163.
[13] C. Isci, A. Buyuktosunoglu, C. Cher, P. Bose, and M. Martonosi, "An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget," in *Proc. Int. Symp. Microarchitecture*, 2006, pp. 347–358.
[14] K. Srinivasan and K. S. Chatha, "Integer linear programming and heuristic techniques for system-level low power scheduling on multiprocessor architectures under throughput constraints," *Integr. VLSI J.*, vol. 40, no. 3, pp. 326–354, 2007.
[15] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. Low Power Electron. Des.*, 2007, pp. 38–43.
[16] *Chip Multiprocessor Watch*, 2008. [Online]. Available: http://view.eecs.berkeley.edu/wiki/Chip_Multi_Processor_Watch
[17] K. Olukotun and L. Hammond, "The future of microprocessors," *ACM Queue*, vol. 3, no. 7, pp. 26–29, Sep. 2005.
[18] N. Kim *et al.*, "Leakage current: Moore's law meets static power," *Computer*, vol. 36, no. 12, pp. 68–75, Dec. 2003.
[19] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proc. Des. Autom. Conf.*, 2004, pp. 275–280.
[20] J. G. Siek, L. Lee, and A. Lumsdaine, *The Boost Graph Library: User Guide and Reference Manual*. Reading, MA: Addison-Wesley, 2002.
[21] N. M. Josuttis, *C++ Standard Library: A Tutorial and Reference*. Boston, MA: Addison-Wesley Professional, 1999.
[22] *Standard Task Graph Set*. [Online]. Available: http://www.kasahara.elec.waseda.ac.jp/schedule

Automated Scan Chain Division for Reducing Shift and Capture Power During Broadside At-Speed Test

Ho Fai Ko, *Student Member, IEEE*, and
Nicola Nicolici, *Member, IEEE*

***Abstract*—Scan chain division has been successfully used to control shift power by enabling mutually exclusive flip-flops at different times during the scan cycle. However, to control capture power without losing transition fault coverage during at-speed scan test, the existing automatic test pattern generation (ATPG) flows need to be modified. In this paper, we present a novel scan chain division algorithm that analyzes the signal dependencies and creates the circuit partitions such that both shift and capture power can be reduced when using the existing ATPG flows. This novel algorithm has been designed for the broadside test application strategy, and a technique for employing partial scan when dividing the scan chains is also proposed.**

***Index Terms*—Broadside (BS), low-power at-speed test, partial scan.**

I. Introduction

As the feature size of technology nodes decreases, the number of physical defects that affect the dynamic behavior (i.e., timing failures) of digital integrated circuits is on the rise. However, it is increasingly difficult to detect such defects using the existing current-based testing methods (e.g., IDDQ) since the quiescent current of a faulty circuit is hard to be distinguished from a fault free one [2]. As past research has shown that, applying stuck-at vectors at operating frequency can help detect the timing-related defects [3], at-speed testing has established itself as an essential step in manufacturing test.

At-speed delay fault tests are graded in terms of fault coverage with respect to a particular delay fault model [4]. In this paper, we focus on the *transition fault model*. For testing a delay fault, a pair of test patterns V_1 and V_2 is applied in successive clock cycles at the operating frequency. The first pattern V_1 initializes the circuit under test (CUT) into a known state, while the second pattern V_2 is used to trigger the transitions that will detect the targeted fault(s). When scan-based design-for-test (DFT) method is employed, V_1 is scanned into the internal flip-flops (FFs). To obtain V_2, there are three test application strategies: enhanced scan, skewed load (launch off shift), and broadside (BS) (launch off capture).

In enhanced scan, V_2 is buffered in shadow latches, which may incur an unacceptable area penalty for most applications. In skewed load, V_2 is generated by shifting the V_1 by one position, which requires the scan enable (SE) signal to switch at speed. The coverage for BS, where V_2 is functionally justified through the combinational logic, is typically lower. However, this is mainly because both enhanced scan and skewed load cover more faults that cannot be excited in the native mode. Because both skewed load and BS are employed in practice [2], in our prior work, we focused on skewed load [5]; in this paper, the focus is on BS.

While applying at-speed patterns can detect timing-related defects, the use of scan poses a problem on test power consumption. This

Manuscript received February 21, 2008; revised June 7, 2008. Current version published October 22, 2008. An earlier version of this work was presented at the International Symposium on Quality Electronic Design in 2008 [1]. This paper was recommended by Associate Editor N. K. Jha.

The authors are with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4K1, Canada (e-mail: henryko@grads.ece.mcmaster.ca; nicola@ece.mcmaster.ca).

Digital Object Identifier 10.1109/TCAD.2008.2006091