Exploiting Input Variations for Energy Reduction

Sato, Toshinori
System LSI Research Center, Kyushu University

Kunitake, Yuji
Graduate School of Computer Science and System Engineering, Kyushu Institute of Technology

Exploiting Input Variations for Energy Reduction

Toshinori Sato[1] and Yuji Kunitake[2]

[1] System LSI Research Center
Kyushu University
3-8-33-3F Momochihama, Sawara-ku, Fukuoka, 814-0001 Japan
toshinori.sato@computer.org
[2] Graduate School of Computer Science and System Engineering
Kyushu Institute of Technology
680-4 Kawazu, Iizuka, 820-8502 Japan
y-kunitake@klab.ai.kyutech.ac.jp

Abstract. The deep submicron semiconductor technologies will make the worst-case design impossible, since they can not provide design margins that it requires. Research directions should go to typical-case design methodologies, where designers are focusing on typical cases rather than worrying about very rare worst cases. They enable to eliminate design margins as well as to tolerate parameter variations. We are investigating canary logic, which we proposed as a promising technique that enables the typical-case design. Currently, we utilize the canary logic for power reduction by exploiting input variations, and its potential of 30% power reduction in adders has been estimated at gate-level simulations. In this paper, we evaluate how canary logic is effective for power reduction of the entire microprocessor and find 9% energy reduction.

Keywords: typical-case design, deep sub-micron, DVFS, dynamic retiming, reliable microarchitecture, robust microarchitecture.

1 Introduction

Parameter variations are predicted to present critical challenges for manufacturability in the future LSIs [2, 8, 17]. In deep submicron (DSM) semiconductor technologies, the traditional worst-case design will not work since process variations increase design margins it requires. The trend toward lower supply voltage and higher clock frequency makes voltage variations and temperature variations more serious. In order to realize robust designs under these situations, designers have to be aware of design for manufacturability (DFM). One of the keys to solve the serious problem is exploiting typical cases. Since worst cases rarely occur, it is better for designers to focus on typical cases. We call it typical-case design methodologies. Recently, several typical-case designs are investigated, such as Razor [4, 5], approximation circuits [11], constructive timing violation (CTV) [13], algorithmic noise tolerance (ANT) [15], and TEAtime [16]. This paper focuses on the Razor logic. We proposed canary logic, an improvement of the Razor logic. In our preliminary study [14], the potential in power reduction of 30% was found in the case of carry select adders. However, it is unclear that the canary logic could reduce energy consumption of the entire microprocessors. The aim of this paper is to evaluate this open issue.

This paper is organized as follows. Section 2 introduces a typical-case design methodology. Section 3 describes related works with an emphasis on the Razor logic. Section 4 proposes the canary logic. Section 5 explains our evaluation methodology and Section 6 presents experimental results. Finally, Section 7 concludes.

2 Typical-Case Design Methodologies

The DSM technologies increase variations, and hence design margins that the traditional worst-case design methodology requires, are increased. The conservative approach will not work. Considering this situation, design methodology should be reconsidered for DFM. Typical-case design methodologies are one of the promising ones. It exploits an observation that worst cases are rare. Designers should focus on typical cases rather than worst cases. Since they do not have to consider worst cases, design constraints are relieved, resulting in easy designs.

In the typical-case design methodologies, designers adopt two methods to a circuit design at a time. One is performance-oriented design, where only typical cases are under consideration. Since worst cases are not considered, design constraints are relaxed, resulting in easy designs. The other is function-guaranteed design. While worst cases are considered, designers don’t have to consider performance. They only have to guarantee functions, and thus design must be simple, resulting in easy verifications.

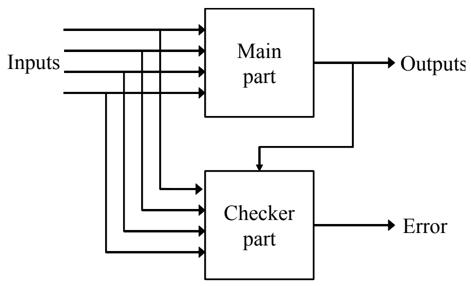

Fig. 1. Typical-Case Design

The concept of a typical-case design methodology is as follows. Every critical function in an LSI chip is designed by two methods. The design consists of two components as shown in Fig.1. One is called main part, and the other is called checker part. While two parts share the single function, their roles and implementations are mutually different. On designing the main part, performance is optimized to increase, but correct function is ignored to guarantee. The main part might cause errors. That is, it is implemented by the performance-oriented design. The checker part is provided as a safety net for the unreliable main part. It detects errors that occur in the main part, and thus it has to satisfy all design constrains in the chip. However, on the checker part design, while designers have to guarantee the function, they do not have to optimize neither of performance and power. That is, it is implemented by the function-guaranteed design. If an error is detected by the checker part, the circuit state has to be recovered to a safe point where the error is detected by any means.

3 Related Works

Examples of the typical-case designs include Razor [4, 5], approximation circuits [11], CTV [13], ANT [15], and TEAtime [16].

In the approximation circuits [11], instead of implementing the complete circuit necessary to realize a desired functionality, a simplified circuit is implemented to approximate it. The approximation circuit works at higher frequency than the complete circuit does, and usually produces correct results. If it fails, the system utilizing the approximation circuit has to recover to a safe point.

CTV [13] exploits input value variations. Considering that the critical path in the system is not always active, clock frequency and supply voltage, which violate critical path delay, are selected in use. In order to guarantee correct operations, the system utilizing CTV has a conservative circuit that realizes a desired functionality to find timing violation.

In ANT [15], information theoretic technique is employed to determine the lower bounds on energy and performance. In order to approach these bounds, circuit- and algorithmic-level techniques are evolved.

TEAtime [14] uses a tracking circuit to mimic the worst-case delay. As long as the tracking circuit works correctly, clock frequency can be increased and supply voltage can be decreased. Usually, a 1-bit-wise critical path is used for the tracking circuit.

3.1 Razor Logic

Razor [4, 5] permits to violate timing constraints to improve energy efficiency. Razor works at higher clock frequency than that determined by the critical path delay, and removes voltage margin for power reduction. The voltage control adapts the supply voltage based on timing error rates. Figure 2 shows the Razor's dynamic voltage scaling (DVS) system. If the error rate is low, it indicates that the supply voltage could be decreased. On the other hand, if the rate is high, it indicates that the supply voltage should be increased. The control system works to maintain a predefined error rate, E_{ref}. At regular intervals the error rate, E_{sample}, is computed and the rate differential, $E_{diff} = E_{ref} – E_{sample}$, is calculated. If the differential is positive, it indicates that supply voltage could be decreased. The otherwise indicates that the supply voltage should be increased.

In order to detect timing errors, Razor flip-flop (FF) shown in Fig.3 is proposed. Each timing-critical FF (main FF) has its shadow FF, where a delayed clock is delivered to meet timing constrains. In other words, the shadow FFs are expected to always hold correct values. If the values latched in the main and shadow FFs do not match, a timing error is detected. When the timing error is detected in microprocessor pipelines, the processor state is recovered to a safe point with the help of a mechanism based on counterflow pipelining. One of the difficulties on Razor is how it is guaranteed that the shadow FF could always latch correct values. The delayed clock has to be carefully designed considering so-called short path problem [5].

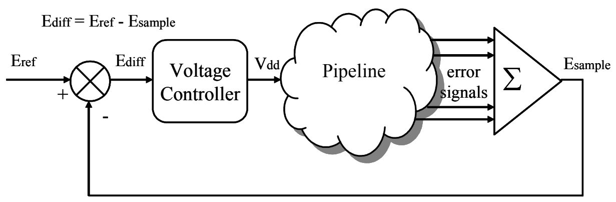

Fig. 2. Razor's DVS System

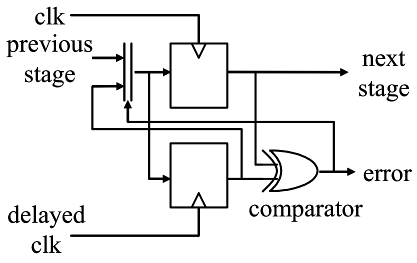

Fig. 3. Razor Flip-Flop

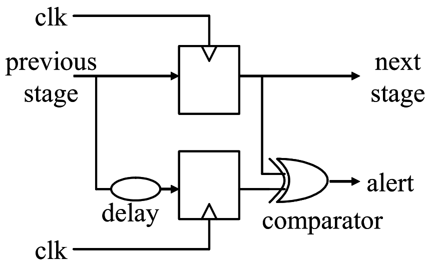

Fig. 4. Canary Flip-Flop

4 Canary Logic

While Razor is a smart technique to eliminate design margins, its circuit implementation could be further improved.

4.1 Canary Flip-Flop

Each FF in the design is augmented with a delay buffer and a canary FF, as shown in Fig.4. The canary FF is used as a canary in a coal mine to help detect whether a timing error is about to occur. Timing errors are predicted by comparing the main FF value with that of the canary FF, which runs into the timing error a little bit before the main FF. Alert signal triggers voltage or frequency control. Utilizing the canary FFs has the following three advantages.

- Elimination of the delayed clock: Using single phase clock significantly simplifies clock tree design. It also eliminates the short path problem [5] in the Razor FF, and hence its minimum-path length constraint should not be considered.
- Protection offered against timing errors: As explained above, the canary FF protects the main FF against timing errors. This freedom from timing errors eliminates any complex recovery mechanism. The selector placed in front of the main FF is removed, leading that some timing pressure is relaxed. Instead, the signal generated

by the comparator triggers voltage or frequency control. If the timing error is alerted, the supply voltage stops falling or the clock frequency is felt down.

- Robustness for variations: The canary FF is variation resilient. The delay buffer always has a positive delay, even though parameter variations affect it. Hence, the canary FF always encounters a timing error before the main FF.

4.2 Power Reduction with Canary FFs

Figure 5 explains how DVS techniques utilize the canary FFs. The horizontal and vertical lines present time and supply voltage, respectively. At regular intervals, the supply voltage is decreased if a timing error is not predicted during the last interval. This is possible since input values activating the critical path are limited to a few variations. For example, it has been reported that nearly 80% of paths have delays of half the critical time [18]. Timing errors rarely occur even if the timing constraints on the critical path are not satisfied. The input value variations can be exploited to decrease the supply voltage. Because the supply voltage is lower than that determined by the critical path delay, significant power reduction is achieved in the canary logic as in the Razor logic [4, 5]. When a timing error is predicted to occur, the supply voltage is increased.

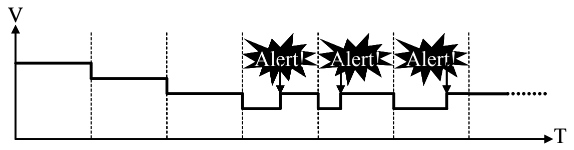

Fig. 5. Canary's DVS

After the first timing error is predicted, timing errors will be repeatedly alerted if the supply voltage is decreased again in the end of the current interval, as shown in Fig.5. Since every supply voltage switching makes processor unavailable during the transition, this oscillation has a serious impact on performance and on power efficiency. In order to prevent the oscillation, we use a counter that counts the number of alerts and stop decreasing the supply voltage when the number exceeds a predefined threshold value. We consider the overhead due to the switches to be small in order to determine the clock cycles of the period where supply voltage reduction is stopped. After the period passes by, the DVS system begins to work again.

4.3 Canary FF Implementation via Scan Reuse

Scan resources, which is implemented for production testing, can be reused to realize the canary FF. Figure 6 shows a scan FF design [12] that consists of a system FF (the lower part in the figure) and a scan portion (the upper part in the figure). The SI input is connected to the SO output of the next scan FF to be a shift register. In the test mode, clocks SCA and SCB are applied to shift a test pattern into latches LA and LB. Next, the UPDATE clock is applied to write the test pattern in LB into the system latch, PH1. Then, the CLK clock is applied to capture the system response to the test pattern. After that, the CAPTURE signal is applied to move the contents of PH1 to LA. And last, clocks SCA and SCB are applied to shift the system response out.

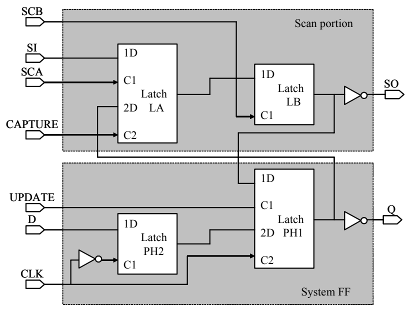

Fig. 6. Scan Cell [12]

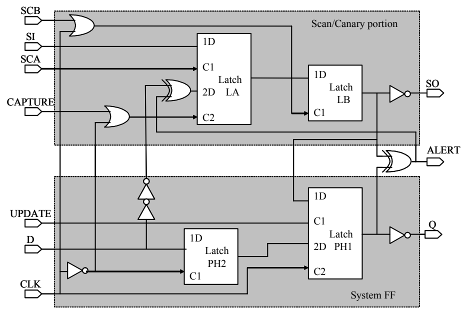

Fig. 7. Canary's Scan Cell

In system operation mode, latches LA and LB are not utilized. The canary FF can be implemented with a little hardware cost by reusing the latches in the scan portion. Figure 7 shows how reusing scan FF design realizes the canary FF. The FF design's test mode operation is identical to the design in Fig.6. In system operation mode, latches LA and LB hold the replicas of PH2 and PH1, respectively. If any timing error does not occur, the ALERT signal is low and thus the delayed signal of D is written into LA. The reuse is possible since the canary logic does not require delayed clock. This is not adapted to the Razor logic.

5 Evaluation Methodology

First, we show how timing error rate is determined and then describe architectural-level simulation environment.

5.1 Timing Error Rates

We estimate timing error rates of the entire microprocessor using a 32b carry select adder (CSLA), since the yield of pipeline is mainly determined by the timing error in execution stage [10]. SYNOPSIS DesignCompiler logic-synthesizes the CSLA with Hitachi 0.18um standard cell libraries. The combinations of the clock frequency and the supply voltage of Intel Pentium M [7], which is shown in Table 1, are used. We project the highest clock frequency, which is determined by CSLA's critical path delay reported by DesignCompiler, onto Pentium's highest clock frequency. In order

Table 1. Frequency - Voltage Specifications

F(GHz)	2.1	1.8	1.6	1.4	1.2	1.0	0.8	0.6
V_{dd}(V)	1.340	1.276	1.228	1.180	1.132	1.084	1.036	0.988

to estimate how timing error occurs, we simulate the CSLA using Cadence Verilog-XL simulator. Gate-level simulation results are shown in Figure 8. It is observed that supply voltage reduction down to 1.18V suffers little timing errors. We use the timing error rates in architectural-level simulations explained in the next section.

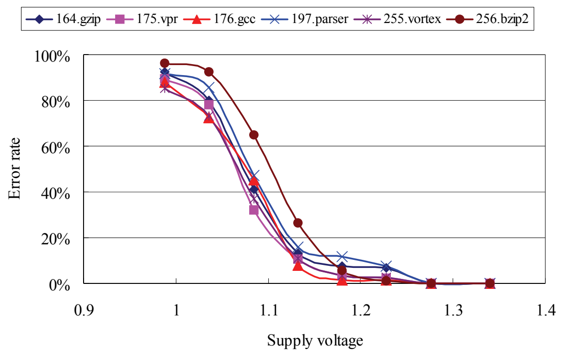

Fig. 8. Error Rate - V_{dd}

5.2 Architectural-level Simulation Environment

SimpleScalar/PISA tool set [1, 3] is used for architectural-level simulation. Table 2 summarizes processor configurations. Six integer programs from SPEC2000 CINT benchmark are used. For each program, 1 billion instructions are skipped before

Table 2. Processor Configurations

Clock frequency	2 GHz
Fetch width	8 instructions
Li instruction cache	16K, 2 way, 1 cycle
Branch predictor	gshare + bimode
gshare predictor	4K entries, 12 histories
Bimodal predictor	4K entries
Branch target buffer	1K sets, 4 way
Dispatch width	4 instructions
Instruction window size	128 entries
Issue width	4 instructions
Integer ALUs	4 units
Integer multiplires	2 units
Floating ALUs	1 unit
Floating multiplires	1 unit
L1 data cache ports	2 ports
L1 data cache	16K, 4 way, 2 cycles
Unified L2 cache	8M, 8 way, 10 cycles
Memory	Infinite, 100 cycles
Commit width	8 instructions

actual simulation begins. After that each program is executed for 2 billion instructions.

We evaluate three intervals between supply voltage scaling, which are 100K, 1M, and 10M clock cycles. It is assumed every supply voltage switching requires 10μs [6]. Two threshold values of 2 and 8 in the canary alert sequence are selected for stopping the supply voltage switching. The cycles where supply voltage reduction is prohibited is determined so as to the impact of switching on performance is less than 0.1%. For example, if the interval is 100K cycles and the threshold is 2, switching is prohibited until 400 intervals pass by.

As explained above, we model the timing error rate of the entire processor based on that of the CSLA. Since it is difficult to consider input value variations in architectural-level simulations, we assume that timing errors occur at random at the error rate shown in Figure 8 in every supply voltage. For example, in the case of 164.gzip, 98.8% of dynamic ALU instructions meet timing error at 0.988V. Since it was found that performance results with and without considerations of actual input variations did not show any significant difference in our previous study on the CTV [9], we expect that the evaluation methodology in the present paper is enough accurate for the preliminary evaluation. We are currently building the simulation environment proposed in [9] in order to improve the accuracy.

6 Results

Figure 9 presents which supply voltages are selected during each program's execution. For each program, the left three bars are for the case where the threshold is two, and the right ones are for the case where that is eight. For each group of three bars, the left, the middle, and the right bars indicate results for the cases where the intervals are 100K, 1M, and 10M cycles, respectively. First, it is observed that only three voltages are selected while eight combinations of voltage and frequency are provided. In addition, almost all the time supply voltage of 1.276V is selected. This means that the DVS system loses the chance to exploit lower supply voltages. Second, any considerable differences can not be observed among three intervals between voltage scaling. Since we determined the cycles where supply voltage reduction is prohibited so as to the impact of switching on performance is identical among different intervals, the configurations with larger intervals perform voltage switching less frequently. Hence, the interval has little impact on supply voltage selection. Last, similarly with the second observation, any significant differences can not be found between two configurations with different thresholds.

The impact on performance is very small. The increase in the execution cycles is at most 0.23%. We find that the oscillation of supply voltage switching is prevented. However, it is two times larger than that we expected. On the other hand, the configurations with the threshold value of 2 suffer slightly less performance penalty than those with the one of 8 do. Based on these observations above, we expect that every configuration achieves similar energy reduction.

Figure 10 presents how energy consumption is reduced. The layout of the graph is the same with Fig.9 and the right six bars indicate the averages. As can be expected, there are not any significant differences among the configurations evaluated in this study. Every configuration achieves approximately 9% of energy reduction.

These results imply that the energy reduction was attained just by selecting 1.276V for the supply voltage instead of 1.340V. In order to exploit input value variations more efficiently, the adaptable DVS system should be improved.

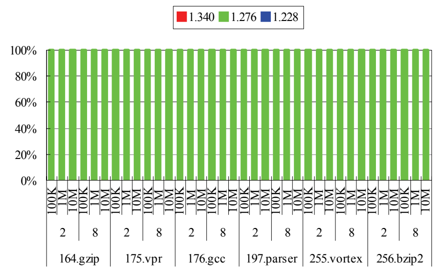

Fig. 9. Breakdown of Supply Voltage

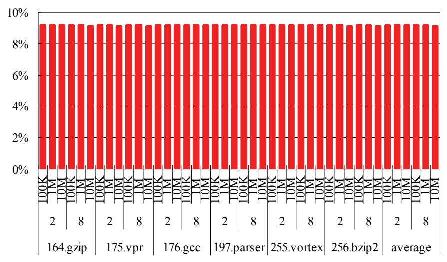

Fig. 10. Energy Reduction

7 Conclusions

As the complexity of the semiconductor manufacturing process increases, it is likely that process variations will be more difficult to control. Under the situations, the DSM semiconductor technologies will make the worst-case design impossible, since they can not provide design margins that it requires. In order to attack the problem, we proposed the canary logic as an alternative of the Razor logic, which is a smart technique to eliminate design margins. The canary logic eliminates the delayed clock required by the Razor logic, resulting in easy design. The canary FF relies on the delay buffer, which always has a positive delay, and hence they are variation resilient.

In this paper, we utilize the canary logic for power reduction by exploiting input value variations. Since timing errors are expected to rarely occur even if the timing constraints on the critical path are not satisfied, input variations can be exploited to decrease the supply voltage. Because the supply voltage is lower than that determined by the critical path delay, significant power reduction is achieved. From the detailed simulation results, we found that the potential energy reduction of 9% without serious impact on performance.

One of the future directions of this study is to build the entire processor at RT- or gate-level to evaluate the canary's usefulness more accurately. Especially, we are interested in how eliminating design margins for parameter, voltage, and temperature variations improves energy efficiency. We are also investigating to improve robustness of microprocessors by utilizing the canary FF.

Acknowledgments. Hitachi 0.18um standard cell libraries are provided by VDEC (VLSI Design and Education Center) in the University of Tokyo. This work is partially supported by the CREST (Core Research for Evolutional Science and Technology) program of Japan Science and Technology Agency.

References

1. Austin T, Larson E, Ernst D: SimpleScalar: an Infrastructure for Computer System Modeling. IEEE Computer, 35(2) (2002)
2. Borkar S, Karnik T, Narendra S, Tschanz J, Keshavarzi A, De V: Parameter Variations and Impact on Circuits and Microarchitecture. 40th Design Automation Conference (2003)
3. Burger D, Austin T M: The SimpleScalar Tool Set, Version 2.0. ACM SIGARCH Computer Architecture News, 25(3), (1997)
4. Das S, Sanjay P, Roberts D, Lee L S, Blaauw D, Austin T, Mudge T, Flautner K: A Self-Tuning DVS Processor Using Delay-Error Detection and Correction. Symposium on VLSI Circuits (2005)
5. Ernst D, Kim N S, Das S, Pant S, Rao R, Pham T, Ziesler C, Blaauw D, Austin T, Flautner K, Mudge T: Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation. 36th International Symposium on Microarchitecture (2003)
6. Gochman S, Ronen R, Anati I, Berkovits A, Kurts T, Naveh A, Saeed A, Sperber Z, Valentine R C: The Intel Pentium M Processor: Microarchitecture and Performance. Intel Technology Journal, 7(2), (2003)
7. Intel Corporation: Intel Pentium M Processor on 90nm Process with 2-MB L2 Cache. Datasheet (2006)
8. Karnik T, Borkar S, De V: Sub-90nm Technologies: Challenges and Opportunities for CAD. International Conference on Computer Aided Design (2002)
9. Kunitake Y, Chiyonobu A, Tanaka K, Sato T: Challenges in Evaluations for a Typical-Case Design Methodology. 8th International Symposium on Quality Electronic Design (2007)
10. Li H, Chen Y, Roy K, Koh C-K: SAVS: A Self-Adaptive Variable Supply-Voltage Technique for Process-Tolerant and Power-Efficient Multi-Issue Superscalar Processor Design, 11th Asia and South Pacific Design Automation Conference (2006)
11. Lu S-L: Speeding up Processing with Approximation Circuits. IEEE Computer, 37(3), (2004)
12. Mitra S, Seifert N, Zhang M, Shi Q, Kim K S: Robust System Design with Built-In Soft-Error Resilience. IEEE Computer, 38(2), (2005)
13. Sato T, Arita I: Constructive Timing Violation for Improving Energy Efficiency. in Benini L, Kandemir M, Ramanujam J: Compilers and Operating Systems for Low Power. Kluwer Academic Publishers (2003)
14. Sato T, Kunitake Y: A Simple Flip-Flop Circuit for Typical-Case Designs for DFM. 8th International Symposium on Quality Electronic Design (2007)
15. Shanbhag N R: Reliable and Efficient System-on-chip Design. IEEE Computer, 37(3), (2004)
16. Uht A K: Going beyond Worst-case Specs with TEAtime. IEEE Computer, 37(3), (2004)
17. Unsal O S, Tschanz J W, Bowman K, De V, Vera X, Gonzalez A, Ergin O: Impact of Parameter Variations on Circuits and Microarchitecture, IEEE Micro, 26(6), (2006)
18. Usami K, Igarashi M, Minami F, Ishikawa T, Kanazawa M, Ichida M, Nogami K: Automated Low-Power Technique Exploiting Multiple Supply Voltages Applied to a Media Processor. IEEE Journal of Solid-State Circuits, 33(3), (1998)