A Simple Flip-Flop Circuit for Typical-Case Designs for DFM

Sato, Toshinori
System LSI Research Center, Kyushu University

Kunitake, Yuji
Graguate School of ComputerScience and System Engineering, Kyushu Institute of Technology

出版情報：Proc. of 8th International Symposium on Quality Electronic Design, pp.539-545, 2007-03-27. International Symposium on Computer Quality Electronic Design
バージョン：
権利関係：

A Simple Flip-Flop Circuit for Typical-Case Designs for DFM

Toshinori Sato
System LSI Research Center
Kyushu University
toshinori.sato@computer.org

Yuji Kunitake
Graduate School of Computer Science
and System Engineering
Kyushu Institute of Technology
y-kunitake@mickey.ai.kyutech.ac.jp

Abstract

The deep submicron (DSM) 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. In this paper, canary *logic is proposed as a promising technique that enables the typical-case design. It is easier to design than the previously proposed* Razor *logic by eliminating delayed clock. Estimates based on gate-level simulations show that the canary logic achieves average power reduction of 30% by exploiting dynamic variations in circuit delay.*

1. Introduction

As the complexity of the semiconductor manufacturing process increases, it is likely that process variations will be more difficult to control [1, 2, 3]. The demand for low power leads supply voltage reduction and hence makes voltage variations a serious problem. Higher and higher clock frequency increases temperature variations in a chip. Under these situations, the deep submicron (DSM) semiconductor technologies will make the worst-case design impossible, since they can not provide design margins that it requires. In order to realize robust designs, designers have to be aware of design for manufacturing (DFM).

One of the promising solutions is typical-case design methodology, where LSIs should be designed with typical case considerations rather than with worst case considerations. Recently, several typical-case designs are investigated, such as Razor [4, 5], approximation circuits [6, 7], algorithmic noise tolerance (ANT) [8], TEAtime [9], and constructive timing violation (CTV) [10]. This paper focuses on the Razor logic and modifies it into canary logic in order to simplify clock design.

This paper is organized as follows. Section 2 introduces a typical-case design methodology. The Razor logic and the canary logic are described. Section 3 presents experimental results. Finally, Section 4 concludes.

2. Typical-Case Design Methodology

The DSM technologies increase variations, and hence design margins that the traditional worst-case design methodology requires, are reduced. The conservative approach will not work. Considering this situation, design methodology should be reconsidered for DFM. Typical-case design methodology is 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 methodology, 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.

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 Figure 1. One is called main part, and the other is called checker part. While two parts shares 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.

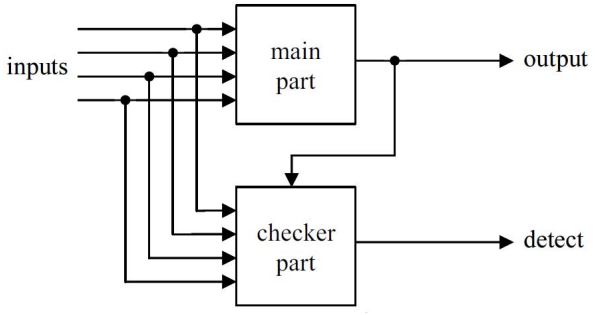

Figure 1: Typical-Case Design

Examples of the typical-case designs include Razor [4, 5], approximation circuits [6, 7], ANT [8], TEAtime [9], and CTV [10]. This paper focuses on Razor and the following section describes its details.

2.1. 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. In order to detect timing errors, Razor flip-flop (FF) shown in Figure 2 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.

Razor removes voltage margin for power reduction. The voltage control adapts the supply voltage based on timing error rates. Figure 3 shows the Razor's dynamic voltage scaling 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.

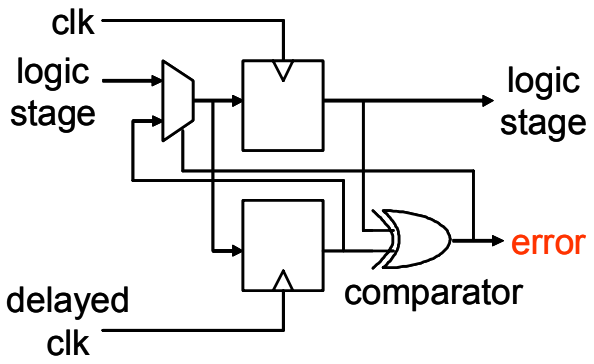

Figure 2: Razor FF

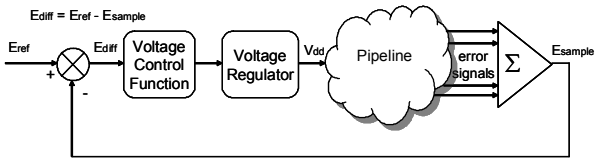

Figure 3: Razor's Voltage Scaling System

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].

2.2. Canary Logic

While Razor is a smart technique to eliminate design margins, its circuit implementation could be further improved. Each FF in the design is augmented with a delay buffer and a *canary FF*, as shown in Figure 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. Error signal triggers voltage or frequency control. Utilizing the canary FFs has the following two 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. Furthermore, the canary FF is variation resilient. The delay buffer always has a positive delay, even though parameter

variations affect it. Hence, the canary FF encounters a timing error before the main FF.

- **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 MUX 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 detected, the supply voltage stops falling or the clock frequency is felt down.

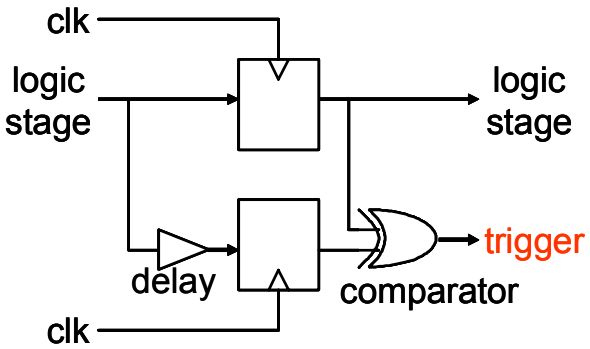

Figure 4: Canary FF

Figure 5 explains how Dynamic Voltage Frequency Scaling (DVFS) techniques utilize the canary FFs. The horizontal line presents time. The upper part of the figure presents supply voltage and the lower part presents clock frequency. There are two modes: initial mode and steady mode. In the initial mode (① in the figure), the delay of the delay buffer in the canary FF is considered and hence lower clock frequency is selected. However, the frequency can be increased by utilizing the fact that input signals 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 [11]. Timing errors rarely occur even if the timing constraints on the critical path are not satisfied. By exploiting the dynamic variations in circuit delay, the delay of the buffer can be ignored and the clock frequency can be increased (② in the figure). Now the mode is in the steady. In the steady mode, the dynamic variations can be further exploited to decrease the supply voltage (③, ④, and … in the figure). 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].

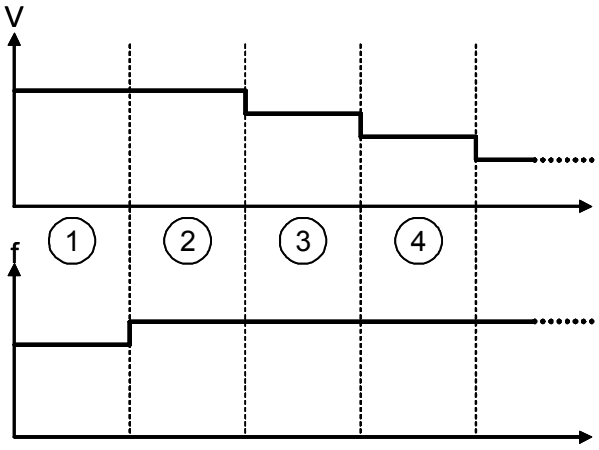

Figure 5: Dynamic Voltage Frequency Scaling

The initial mode changes into the steady mode after the defined number of clock cycles. If there are not any timing errors in the canary FFs all through the cycles, the mode is changed. Similarly, if there are not any timing errors through regular intervals in the steady mode, the supply voltage is decreased into the next lower value. In contrast, every time a timing error is detected in a canary FF, the supply voltage is immediately increased into the next upper value. If the voltage is highest, the clock frequency is reduced. In other words, the steady mode changes into the initial mode. As mentioned above, even though timing errors are detected, any recovery action is not required. This is because the canary FFs cause timing errors before the main parts of the circuit causes timing errors. In other words, the timing errors detected in the canary FFs are predictions of those in the main FFs.

On the other hand, utilizing the canary FF has the following demerits. First, as explained above, the delay of the delay buffer has to be considered to always satisfy timing constraints. Hence, in the initial state, possible power reduction can not be attained due to the conservative design choice. Second, power reduction is less aggressive in the canary logic than in the Razor logic. Since the main part never causes timing errors, there must be some margins in timing constraints, resulting in loss of the further reduce power consumption.

3. Evaluations

The usefulness of the canary logic is evaluated on a 32b carry select adder (CSLA). Evaluating an entire processor is remained for the future study. A 32b CSLA shown in Figure 6 is designed using Verilog-

HDL. It consists of 13 ripple carry adders (RCAs) with four different bit widths.

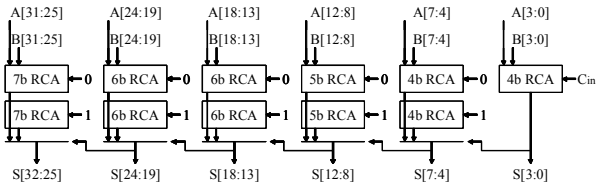

Figure 6: Carry Select Adder

The CSLA is logic-synthesized using SYNOPSIS DesignCompiler with Hitachi 0.18um standard cell libraries. DesignCompiler reports the critical path delay, and the clock frequency higher than that determined by the critical path delay is used. The clock frequency is increased by 20-200%. In order to detect timing errors, we simulate two CSLAs simultaneously using Cadence Verilog-XL simulator, as shown in Figure 7. One is the upper CSLA with delay considerations, and is a gate-level circuit synthesized by DesignCompiler. The other is the lower CSLA without delay considerations, and is an RTL description that is implemented using Verilog-HDL. As the clock frequency delivered to the registers is increased, the comparator signals there is a difference between two results.

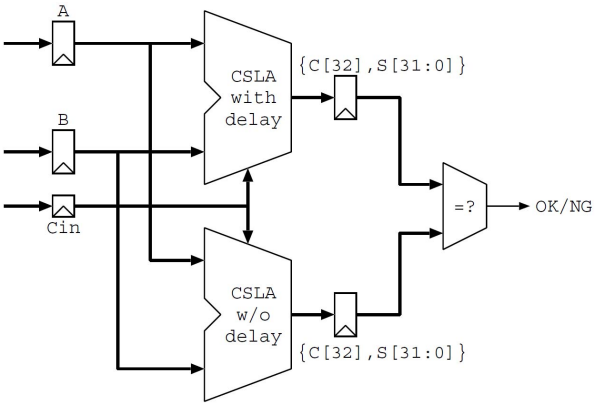

Figure 7: Evaluation Circuit

Six programs, 164.gzip, 175.vpr, 176.gcc, 197.parser, 255.vortex, and 256.bzip, from SPEC2000 are used as workloads and generate test vectors for the gate-level simulations by using SimpleScalar's functional-level simulator [12].

The gate-level simulation results are shown in Figure 8. It can be found that the error rate is less than 40% even if the clock frequency is two times boosted (100% overclocking rate). This confirms Usami's observations [11] explained in Section 2.1.

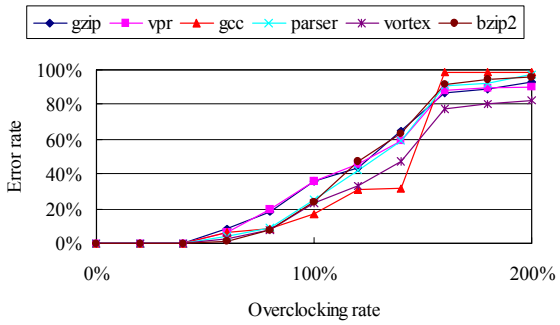

Figure 8: Error vs. Overclocking rate

Instead of boosting the clock frequency, the supply voltage is decreased in order to reduce power consumption. In other words, design margin is dynamically decreased for power reduction. Since the canary in a coal mine is provided, correct operations are always guaranteed. In order to evaluate how power consumption can be reduced, the combinations of the clock frequency and the supply voltage of Intel Pentium M [13], which is shown in Table 1, is used. Based on the frequency and voltage specifications, the transformation from the overclocking rate shown in Figure 8 into the expected supply voltage is calculated as shown in Table 2.

Table 1: Frequency - Voltage Specifications

GHz	Vdd
2.1	1.340
1.8	1.276
1.6	1.228
1.4	1.180
1.2	1.132
1.0	1.084
0.8	1.036
0.6	0.988

Using Figure 8 and Table 2, power reduction can be estimated as shown in Figure 9. Only active power is considered in this estimate. For all programs selected in this study, almost 30% of power reduction is expected.

Figure 10 presents which power supply voltages are selected during each program's execution. It is observed that the supply voltage of less than 1.081V is selected for more than 50% of time. That of more than 1.223V is required only for less than 10% of time. These characteristics enable power reduction presented in Figure 9.

Table 2: Overclocking - Voltage Specifications

%Overclocking	Vdd
0	1.340
20	1.283
40	1.233
60	1.188
80	1.148
100	1.112
120	1.081
140	1.055
160	1.033
180	1.016
200	1.003

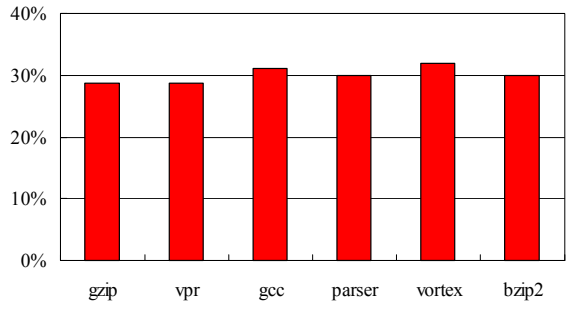

Figure 9: Power Reduction

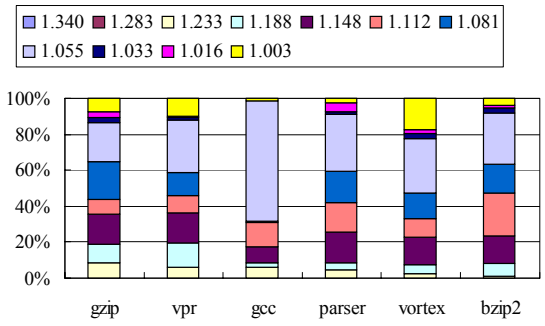

Figure 10: Breakdown of Supply Voltage

4. Conclusions

This paper 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. The canary logic is adopted to power reduction in a CSLA design, and it is found that approximately 30% of power reduction is achieved.

Future directions of this study include evaluating how the canary logic reduces power consumption of an entire processor. Comparisons between the Razor logic and the canary logic are also interesting.

Acknowledgements

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] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De, “Parameter Variations and Impact on Circuits and Microarchitecture”, 40th Design Automation Conference, 2003.

[2] T. Karnik, S. Borkar, and V. De, “Sub-90nm Technologies: Challenges and Opportunities for CAD”, International Conference on Computer Aided Design, 2002.

[3] X. Tang, V. K. De, and J. D. Meindl, “Intrinsic MOSFET Parameter Fluctuations Due to Random Dopant Placement”, IEEE Transactions on VLSI Systems, Vol.5, No.4, 1997.

[4] S. Das, P. Sanjay, D. Roberts, L. Seokwoo Lee, D. Blaauw, T. Austin, T. Mudge, and K. Flautner, “A Self-Tuning DVS Processor Using Delay-Error Detection and Correction”, Symposium on VLSI Circuits, 2005.

[5] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, “Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation”, 36th International Symposium on Microarchitecture, 2003.

[6] T. Liu and S-L. Lu, “Performance Improvement with Circuit-level Speculation”, 33rd International Symposium on Microarchitecture, 2000.

[7] S-L. Lu, “Speeding up Processing with Approximation Circuits”, IEEE Computer, Vol.37, No.3, 2004.

[8] N. R. Shanbhag, “Reliable and Efficient System-on-chip Design”, IEEE Computer, Vol.37, No.3, 2004.

[9] A. K. Uht, “Going beyond Worst-case Specs with TEAtime”, IEEE Computer, Vol.37, No.3, 2004.

[10] T. Sato and I. Arita, “Constructive Timing Violation for Improving Energy Efficiency”, in L. Benini, M. Kandemir, and J. Ramanujam, “Compilers and Operating Systems for Low Power”, Kluwer Academic Publishers, 2003.

[11] K. Usami, M. Igarashi, F. Minami, T. Ishikawa, M. Kanazawa, M. Ichida, and K. Nogami, “Automated Low-Power Technique Exploiting Multiple Supply Voltages Applied to a Media Processor”, IEEE Journal of Solid-State Circuits, Vol. 33, No. 3, 1998.

[12] D. Burger and T. M. Austin, “The SimpleScalar Tool Set, Version 2.0”, ACM SIGARCH Computer Architecture News, Vol. 25, No. 3, 1997.

[13] Intel Corporation, “Intel Pentium M Processor on 90nm Process with 2-MB L2 Cache”, Datasheet, 2006.