Typical-case Design Methodology: Concept, Challenge, and Case Study

Kunitake, Yuji
Kyushu Institute of Technology

出版情報：SLRC 論文データベース. 13, 2008-01-23. Asia and South Pacific Design Automation Conference || 13
バージョン：
権利関係：

Typical-case Design Methodology: Concept, Challenge, and Case Study

Yuji Kunitake
Kyushu Institute of Technology
680-4, Kawazu, Iizuka, Fukuoka, 820-8502 Japan
y-kunitake@klab.ai.kyutech.ac.jp

1. The Concept: What is Typical-case Design?

This abstract presents my incremental study of the paper previously published at ISQED 2007 [1]. It is expected that I will graduate in March, 2008, under the advice from Toshinori Sato at Kyushu University. The aim of my study is developing a design methodology for the advanced semiconductor technologies, where transistors are unreliable and thus parameter variations are increased.

Until recently, circuit designs have been conducted under the considerations of worst-case design margins. However, deep submicron semiconductor technologies have caused the increase in parameter variations, which will make it difficult to estimate the design margins. In order to manage the unpredictable margins, we exploit the observation that worst cases rarely occur. Designers should focus on typical cases rather than worst cases so that the over estimated design margins can be eliminated. This is the concept, and we name it typical-case design methodology.

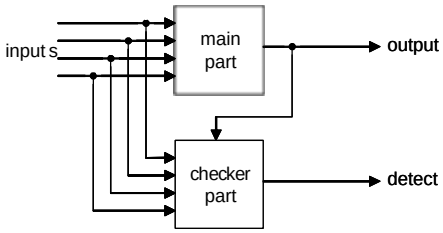

Figure 1: Constructive Timing Violation

Constructive Timing Violation (CTV) technique [1] is one of the typical-case designs and it aims to eliminate the overestimated timing margin. This is attained by utilizing the observation that input signals to a circuit, which activate the critical path, are limited to a few variations. The concept of the CTV is as follows. A circuit 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 share the single function, their roles and implementations are mutually different. The main part is designed in consideration of the performance, but might cause timing errors. The checker part is a safety net for the main part. It detects the timing errors in the main part, and thus it has to satisfy all timing constrains. However, designers do not have to optimize performance nor power but only have to guarantee the function. If a timing error is detected by the checker part, the circuit state to be recovered to a safe point by any means.

2. A Case Study: Carry Select Adder with Error History Buffer

We apply the CTV technique to a 32-bit carry select adder [1] as a case study. Unfortunately, however, performance loss due to timing errors is serious, even though the functionality is corrected. In order to mitigate the loss, we propose an enhancement of the CTV, which is called Error History Buffer (EHB) [1].

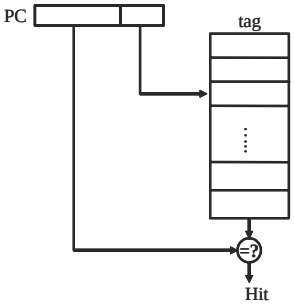

Figure 2: Error History Buffer

The EHB exploits the observation that just a small number of specific instructions are responsible for most timing errors. Every instruction that once causes a timing error might cause the same error over and over again. As shown in Figure 2, the EHB has a direct-mapped cache-like structure but only has a tag field. Instruction address is used to index it, and its most significant bits are kept in the tag field. The addresses of instructions that cause timing errors are registered in the EHB. Once the instruction is registered in the EHB, it is identified as an error-prone instruction regardless of its input operands. When there is a hit in the EHB, the result provided by the main part in Figure 1 is dropped but that provided by the checker part is used.

3. A Challenge: Co-simulations Environment

The previous study [1] presented evaluation results for only one organization of the EHB. This is because there were not any accurate environments for evaluating the typical-case design. This is a challenge. In order to evaluate the EHB, we developed a co-simulation environment. The co-simulator is an architectural-level simulator with the ability of considering circuit delay. It is built upon the orchestration of gate- and architectural-level simulations, as shown in Figure 3. The gate-level simulator simulates the CTV-based adders, and verifies whether timing errors occur or not. The architectural-level simulator simulates the entire processor.

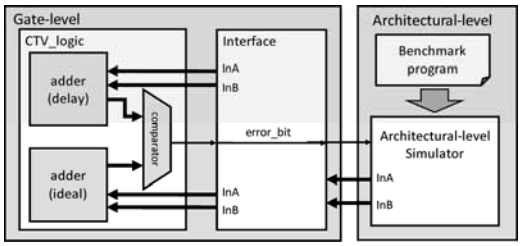

Figure 3: Diagram of Co-simulation Procedure

The processor configuration and benchmark programs are equivalent with those used in [1]. We skip the first 1 billion instructions to avoid unrepresentative behavior at the beginning of the program's execution. Results are then reported for simulating each program for 100 million instructions.

We provide the clock two times faster than that meets the critical path delay of the main part. It is assumed the checker part is timing-error free. We have already designed the circuit that satisfies the assumption but omit its details due to lack of space.

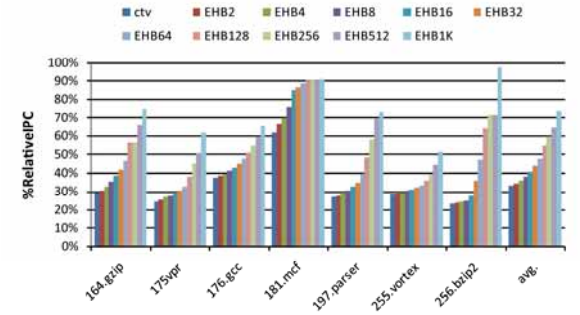

Figure 4: Effect of EHB on IPC

Figure 4 shows relative performance of the CTV-based processor that uses the EHB. There are 11 bars. The most left one is for the case where only CTV is utilized. The remaining bars are for the cases where the number of EHB entry is varied from 2 to 1K. The 1K-entry EHB increases IPC to 74% of the baseline IPC on average. Considering that clock frequency is two times increased, the processor performance is improved by 48%. In particular, the achieved IPC in the cases of 181.mcf and 256.bzip2 is over 90% of the baseline IPC, which turns 80% performance improvement. If we do not prefer performance improvement, clock frequency is reduced appropriately, resulting in large safety margin. Hence, we can expect the EHB is a beneficial technique.

Now, we should revisit the original CTV. It is important to know how frequently the main part is used. When an instruction is hit in the EHB, the instruction is executed on the checker part. If all the instructions become a hit in the EHB, they use the checker part. In other words, the main part is not necessary. Figure 5 shows the EHB hit rate. The hit rate is as much as 61%. At least 39% of instructions require the main part. From the observations, we can conclude that the processor benefit from the original CTV itself.

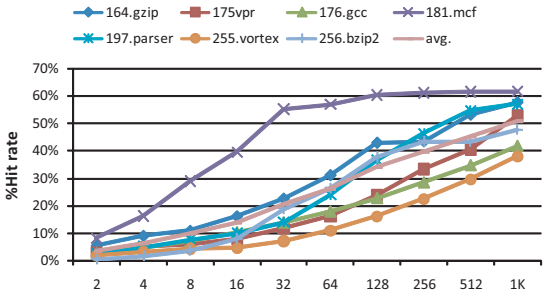

Figure 5: EHB Hit Rate

4. Conclusions

The CTV-based design methodology can eliminate timing margins by exploiting input value variations. However, unfortunately, the penalty due to the recovery from timing errors has serious impact on performance. This paper investigated an enhancement of the CTV-based design. From the detailed co-simulations, we found that the EHB recovers the performance loss.

Acknowledgements

This work is partially supported by Grant-in-Aid for Scientific Research (KAKENHI) (A) # 19200004 from JSPS, and by the CREST programs of JST.

References

[1] Y. Kunitake, et al.: Challenges in Evaluations for a Typical-case Design Methodology, 8th International Symposium on Quality Electronic Design, pp.374-379, 2007.