Design Margin Elimination in a Near-Threshold Timing Error Masking-Aware 32-bit ARM Cortex M0 in 40nm CMOS

Hans Reyserhove and Wim Dehaene
KU Leuven, ESAT-MICAS
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
Email: Hans.Reyserhove@esat.kuleuven.be, Wim.Dehaene@esat.kuleuven.be

***Abstract*—This paper presents a timing error masking-aware ARM Cortex M0 microcontroller system. Timing errors are detected through a timing error detection strategy, consisting of a soft edge flip-flop combined with a transition detector and an error latch. The time borrowing realized through soft edge flip-flops allows data to propagate after the clock edge (timing error masking). Thus operation at the point-of-first-failure is possible, effectively eliminating any timing margin. At the same time, time borrowing events are flagged which prevents corrupting the system state and allows dynamic voltage scaling. The error-aware microcontroller was implemented in a 40nm CMOS process and realizes ultra-low voltage operation down to 0.29V at 5MHz and 12.90pJ/cycle. 37% is energy overhead due to error detection. Minumum energy operation is achieved at 7.5MHz, 0.31V and 11.11pJ/cycle. A total of 75% energy is saved when comparing to a reference design without error detection running at slow-slow corner static timing analysis speed.**

***Index Terms*—CMOS digital integrated circuits, nearthreshold logic, better-than-worst-case design, transmission gate logic, variation resilience, timing margin elimination, soft edge flip-flop, time borrowing, transition detector, error detection, error masking, point-of-first-failure.**

I. Introduction

Advanced technologies suffer from ever increasing variations. Traditionally, timing margins are added at design time to guarantee functionality over a wide PVT range, leading to a significant energy overhead. While energy consumption is the key figure of merit for applications targeted by near-threshold designs, the exponential sensitivity to variations leads to even bigger timing margins combined with an increased energy overhead. Hence, a significant part of the energy gained through near-threshold operation is lost by introducing timing margins. On-die monitors based on canary circuits or critical path replicas track PVT conditions and can significantly decrease this margin. However, intra-die variations between the monitor and the actual critical path induce additional margins, especially at ULV. *In situ* timing error monitoring combined with dynamic voltage scaling (DVS) has been proposed to overcome all these timing margins [1]. By monitoring critical paths in real time, a digital system can work close to its point-of-first-failure (PoFF), irrespective of the PVT condition. Hence, all timing margin and energy overhead is removed. Additionally, a timing error correction strategy allows operation beyond the PoFF, possibly leading to even higher energy gains.

A wide range of timing error resilient techniques has been proposed in recent literature. As described in [2], a distinction can be made between either detecting, predicting or masking errors. Detecting errors ([1], [3]) means the timing error has occurred and should be corrected. Hence, detection occurs after the original clock edge. Correction usually means restoring the system state and comes at the cost of a throughput reduction. Predicting errors ([4], [5]) consists of preventing errors before they occur and corrupt the system. This prediction happens before the clock edge, taking some time out of the original clock period. Although such a system can outperform a margined design, some margin is needed on the error prediction to prevent the system from going corrupt. Hence, performance is sub-optimal. An error masking strategy ([2], [6]) combines aspects of both: it detects errors after the original clock edge, but propagates the correct values anyway, effectively borrowing time from the next pipeline stage. The system state remains intact because the next pipeline stage can cope with the reduced clock period. Error correction is unnecessary and throughput is guaranteed. In a DVS system, the error masking information can be used to operate the system at the PoFF: all timing margin is eliminated and there is no speed performance penalty when compared to a system without error masking. The energy overhead due to the error masking logic should be smaller than the overhead sustained by adding timing margin, resulting in an overall energy gain. While some strategies use flip-flop based pipelines ([1], [2], [4], [6]), other use (pulsed) latches ([2], [3], [5]). Although challenges caused by latch-based design can be overcome, flip-flop based design is generally considered more suited for timing closure. The soft edge flip-flops used in this work combine the time borrowing properties of latches with the easy timing closure found in flip-flop based designs.

All three mentioned strategies have been succesfully implemented in recent literature [1]–[6]. Their relative performance is application and implementation dependant. Allthough timing margins worsen drastically when decreasing supply voltage, relatively few work has been done to implement such strategies at ultra-low voltage. All strategies rely on timing windows related to the original clock, which is difficult to realize under variation sensitive conditions.

This work achieves timing error masking and detection in a 32-bit ARM Cortex M0 microcontroller. The system is realized using a differential transmission gate design flow [7] to allow ultra-low voltage variation resilient operation. Error masking and detection is guaranteed down to 290mV supply voltage by means of a robust soft edge flip-flop, transition detector and error latch. DVS can be used to operate at the PoFF for a specific target frequency. Section II elaborates on the circuit design. Section III discusses the implementation of the error masking-aware strategy in a 32-bit microcontroller. Section IV discusses the measurement results and compares with a reference implementation. Finally, section V draws a conclusion.

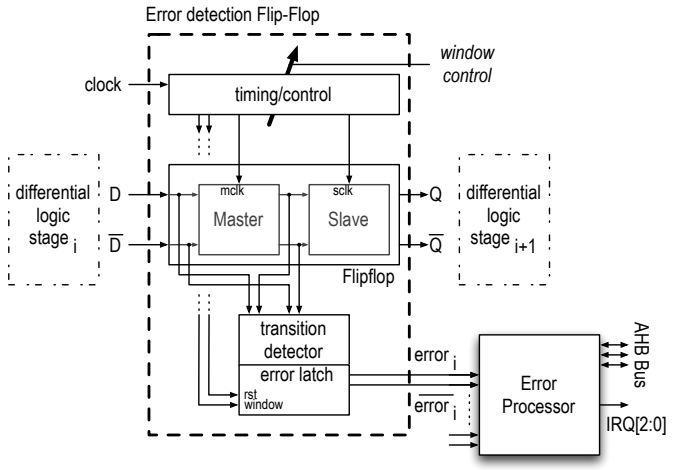


Fig. 1. Overview of the timing error-aware soft edge flip-flop, consisting of a timing/control block, a flip-flop, a transition detector and an error latch.

II. Circuit Design

Timing errors are masked and detected by means of an error detection flip-flop. A conceptual view of the flip-flop is shown in Fig. 1. It replaces normal master/slave flip-flops only on the most critical paths. The error detection flip-flop consists of a timing/control block, a master/slave flip-flop, a transition detector and an error latch. The timing/control block generates a separate master and slave clock which have some overlap. Hence, the latches form a soft edge flip-flop: during overlap, the flip-flop is fully transparant, allowing data arriving late to propagate through the flip-flop, even when it arrives after the original clock edge. The clock overlap defines a timing window that enables the error latch. In this window, the transition detector triggers the error latch when a data transition occurs. Since the data can pass through the flip-flop, a timing error is prevented and the pipeline state is kept intact. The error latch propagates an error signal which can be used to implement DVS at the system level. Because data is allowed to propagate after the clock edge, time is borrowed from the next pipeline stage. A circuit implementation of the error detection flip-flop is shown in Fig. 2. The timing window can be controlled through a biased delay line. Adding a timing/control block to each flip-flop omits the need for multiple clock trees with tight skew constraints. The differential data input of the flip-flop is used for transition detection. Key in this transition detection strategy is the use of the internal delay of the master latch to detect a rising edge on either of the differential signals. No additional logic is required in the data path. The window signal enables the transition detector to switch the error latch. At the end of every cycle the error latch is reset, which allows new error detection to take place.

The procedure becomes even more clear in the timing diagram in Fig. 3. In phase 1, the data arrives on time: data (D or D_{BAR}) propagates to the output (Q or Q_{BAR}) as soon as the slave latch becomes transparant. The transition detector detects the transition outside of the timing window. Hence, no error is flagged. The flip-flop does not borrow time and the next pipeline stage can equip the entire clock cycle. In phase 2, data (D or D_{BAR}) arrives late: after the slave latch becomes transparant, but before the master latch locks. Hence, D can ripple through the entire flip-flop at once, propagating the

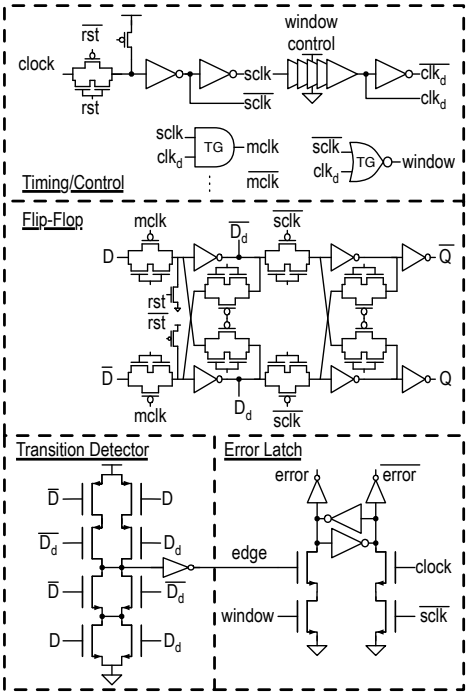


Fig. 2. Circuit implementation of timing error-aware soft edge flip-flop, consisting of a timing/control unit, flip-flop, transition detector and error latch.

correct data (error masking). In this timing window, transition detection triggers the error latch, which propagates an error signal to an error processing unit. Because of the differential data, a rising edge occurs each transition. Leveraging the delay of the master latch, 1-1 overlap on either D, D_{BAR}, D_d or $D_{d,BAR}$ is used to create a pulse on the *edge* signal, which toggles the error latch. With a normal flip-flop, data arriving at this moment would not be able to propagate, and a timing error would occur, corrupting the system state. The amount of time the data can arrive late is limited and is determined by the timing window created by the master and slave clock (*mclk* and *sclk*). The amount of time the data arrives late is borrowed from the next pipeline stage. This constrains the next pipeline stage, but is no problem as long as that stage is not critical. If it is critical, an error detection flip-flop is present there, similar to other critical paths. Time borrowing occurs over the pipeline stages, averaging the timing variation and mitigating a timing error. In phase 3, the error signal is reset and operation continues, either detecting a new timing error or not, depending on when the next data arrives.

Two trade-offs are key to implementing the proposed error detection strategy. First, choosing the window width has a severe impact on performance. A wide window makes error detection easy and enables coarse DVS. However, it severly constrains the pipeline stage following the error detection flip-flop. Additionally, more short path padding is required: as the flip-flop is transparant during the timing window, paths shorter than the timing window can propagate through the flip-flop within the same clock cycle. This can be considered as a hold time constraint and is resolved during hold time optimization, but introduces some overhead. Second, the amount of flip-flops that are enhanced with error detection capabilities has a severe influence on the overall performance of the system. It is clear that error detection introduces a significant overhead compared to a normal flip-flop. As such, that overhead should be

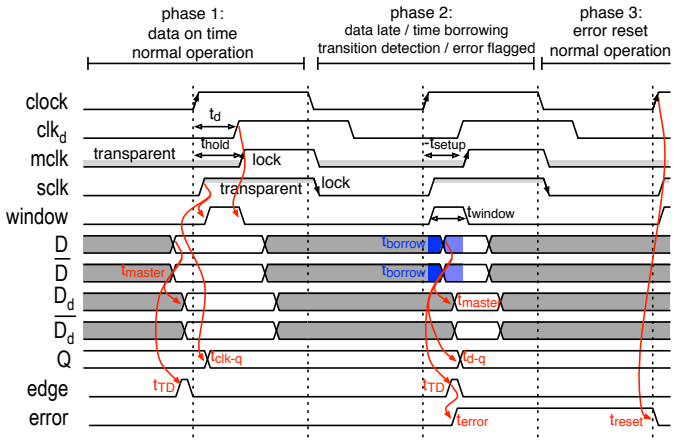


Fig. 3. Timing diagram of timing error masking and detection. Operation is divided in 3 phases: normal operation, error masking operation and error reset.

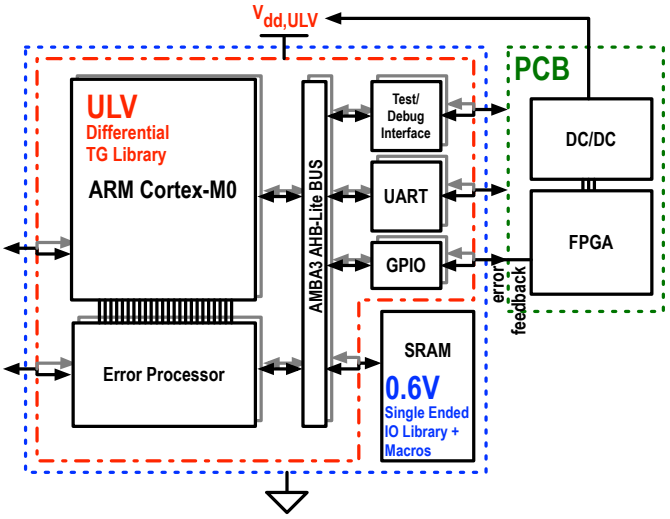


Fig. 4. Overview of the 32-bit ARM Cortex M0 microcontroller system, including error processor and dynamic voltage scaling loop.

minimized, keeping in mind that more error detection flip-flops also result in more timing error information. Additionally, the amount of error detection flip-flops required also depends on the slack distribution of the targeted system and the timing variation of the targeted paths. Overall, it should be made impossible for a timing error to occur without being detected. In its turn, the amount of overhead that can be tolerated depends on the overhead introduced by taking timing margins, something which is unnecessary in the proposed timing error detecting system. Section III discusses how these trade-offs were made for the presented microcontroller system.

III. Implementation

To demonstrate the error detection strategy discussed in section II, the timing error masking strategy was integrated in a 32-bit ARM Cortex M0 microcontroller system. An overview of the system can be seen in Fig. 4. The system is similar to the one presented in [7], as it equips a ARM Cortex M0 core, a 64KB SRAM and some peripherals. Error detection flip-flops as in section II were added to make it timing error-aware. The full system was built using an automated design flow using differential transmission gate logic of two different gate lengths to enable ultra-low voltage variation resilient operation and minimize energy consumption [7]. To select which paths require error detection, the timing slack histogram of the path endpoints is plotted in Fig. 5. 6% of total flip-flops were replaced by error detection flip-flops, a trade-off between overhead and slack distribution. The biased delay line controlling the timing window can scale from 3% to 25% of the clock cycle during test, but was fixed to 5% during timing analysis. The hold time optimization resulted in a 30% increase in hold time buffers compared to a design without error detection.

To analyze timing errors and enable DVS, an error processor was implemented in the microcontroller system. It joins all 224 errors signals using an OR-tree prioritized according to timing slack. Acting as a fully functional peripheral on the AHB bus, the error processor can average timing errors over multiple clock cycles and send interrupts to the M0 core at predefined error thresholds. As such, the M0 core becomes timing error-aware and can implement DVS with simple commands on the GPIO ports or UART tied to interrupt handlers.

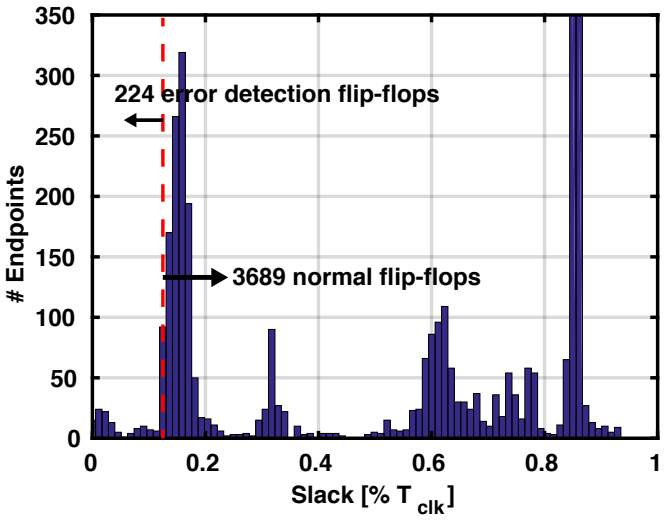
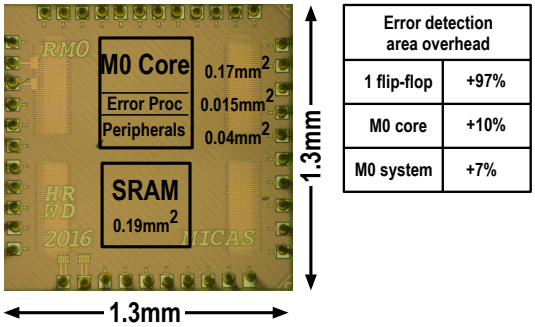


Fig. 5. Timing slack histogram of path endpoints of the microcontroller system before error detecting flip-flops are added. 6% of flip-flops are replaced by error detection flip-flops.

IV. Measurement Results & Analysis

The microcontroller discussed in section III was fabricated in a General Purpose 40nm CMOS process. The chip micrograp is shown in Fig. 6. Replacing a normal flip-flop with an error detection flip-flop increases flip-flop area by about 93%. The entire timing error detection strategy increased active area by about 7% compared when compared to [7]. M0 core area increased by about 10% due to larger flip-flops and short path padding. While the error processor can be seen as pure overhead, it enables easy DVS and can be tuned to the application, severely reducing its area and energy overhead. Fig. 7 shows measured points-of-first-failure for a wide frequency range and their respective core energy consumption per clockcycle. Each point is the result of running the DVS loop for the targeted frequency, decreasing the voltage until timing errors are detected, all while running the Dhrystone benchmark C code and averaged over 5 dies. Because errors are masked rather than corrected, operating far beyond the PoFF can not guarantee correct operation. The system achieves minimum energy operation at 7.5MHz, corresponding to a supply voltage



Error detection area overhead	
1 flip-flop	+97%
M0 core	+10%
M0 system	+7%

Fig. 6. Chip micrograph of the microcontroller system. Active area is 0.42mm^2.

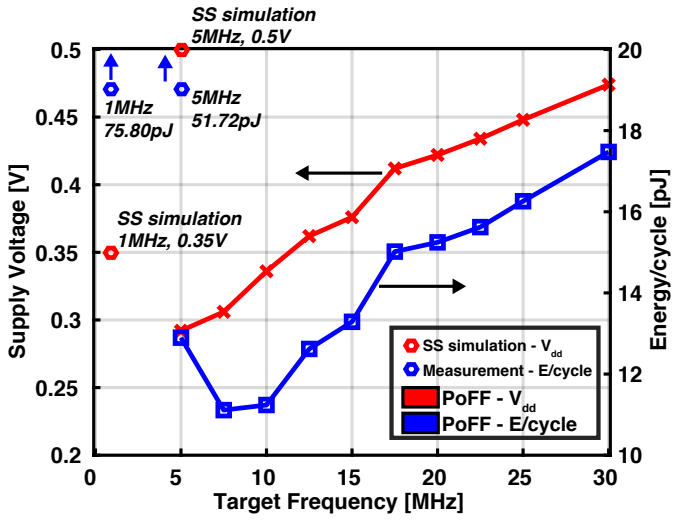


Fig. 7. Measurement of the PoFF curve for a wide frequency range, showing achieved minimum supply voltage and required energy consumption. Slow-slow corner static timing analysis and measured energy/cycle are also indicated.

of 306mV and 11.11pJ/cycle. As a reference, the slow-slow simulation corners resulting from static timing analysis for 350mV and 500mV are also indicated. Energy measurements on a reference design [7] without error detection are indicated as well. This work achieves 5MHz operation at a supply voltage of 290mV compared to the 500mV required when taking timing margins. Core energy consumption at this point is 12.90pJ/cycle, a 75% energy reduction when compared to the margined design. 20MHz operation is achieved at 422mV and 15.25pJ/cycle.

The microcontroller system published in [7] acts as a reference design and allows to evaluate energy overhead caused by timing error detection. Fig. 8 shows an energy consumption comparison for the targeted frequencies between the two designs measured performance: one with timing error detection, one without timing error detection, both operating at the same supply voltage and frequency. Energy overhead ranges from 35% to 60% at frequencies from 5MHz to 30MHz. As mentioned in section III, this overhead is due to both error detection circuitry, short path padding and the error processor. Thanks to this overhead the presented work can operate at the PoFF. This would be impossible without the enabled timing error detection. A 75% energy decrease is realized when compared to the measured energy of the reference design operating at the slow-slow corner operating

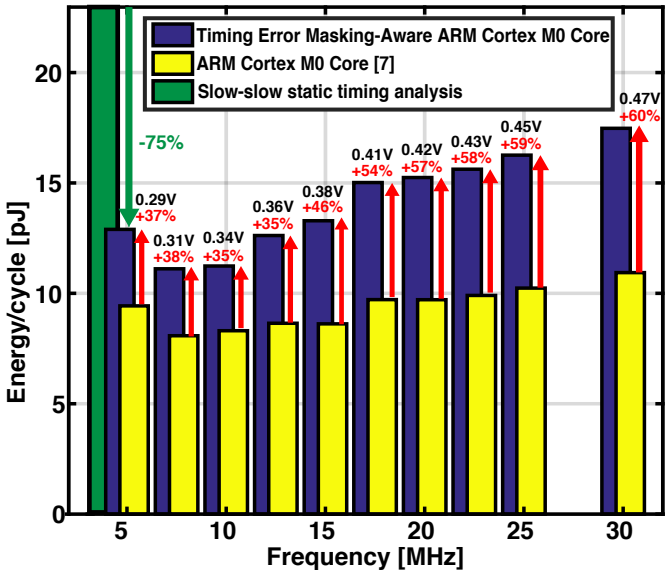


Fig. 8. Measured core energy overhead due to error detection system when compared to [7]. Overheads between 35% and 60% are measured for frequencies between 5 and 30MHz.

speed acquired through static timing analysis.

V. Conclusion

This work presents a timing-error aware ARM Cortex M0 microcontroller system. The system succeeds in eliminating traditional timing margins by detecting timing errors through a soft edge flip-flop augmented with a transition detector and an error latch. The microcontroller cooperates with an on-chip error processor to implement DVS for a target frequency range up to 30MHz. Measurements of both the proposed design and a reference design show an energy overhead down to 35% due to the error detection. The PoFF curve shows minimum energy per operation at 7.5MHz and 11.11pJ/cycle. Thanks to the error detection this work overcomes the large energy overhead due to margins typically incurred when employing near-threshold circuits, resulting in 75% energy savings.

References

[1] S. Das, *et al.*, "A self-tuning DVS processor using delay-error detection and correction," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 792–804, April 2006.

[2] M. Choudhury, V. Chandra, K. Mohanram, and R. Aitken, "TIMBER: Time borrowing and error relaying for online timing error resilience," in *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, March 2010, pp. 1554–1559.

[3] M. Fojtik, *et al.*, "Bubble Razor: Eliminating Timing Margins in an ARM Cortex-M3 Processor in 45 nm CMOS Using Architecturally Independent Error Detection and Correction," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 66–81, Jan 2013.

[4] K. Bowman, *et al.*, "Circuit techniques for dynamic variation tolerance," in *2009 46th ACM/IEEE Design Automation Conference*, July 2009, pp. 4–7.

[5] M. Hiienkari, J. Teittinen, L. Koskinen, M. Turnquist, and M. Kaltiokallio, "A 3.15pJ/cyc 32-bit RISC CPU with timing-error prevention and adaptive clocking in 28nm CMOS," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, Sept 2014, pp. 1–4.

[6] M. Wieckowski, *et al.*, "Timing yield enhancement through soft edge flip-flop based design," in *2008 IEEE Custom Integrated Circuits Conference*, Sept 2008, pp. 543–546.

[7] H. Reyserhove and W. Dehaene, "A Differential Transmission Gate Design Flow for Minimum Energy Sub-10pJ/cycle ARM Cortex-M0 MCUs," *accepted for publication in IEEE Journal of Solid-State Circuits*, July 2017.