Margin Elimination Through Timing Error Detection in a Near-Threshold Enabled 32-bit Microcontroller in 40-nm CMOS

Hans Reyserhove, *Student Member, IEEE*, and Wim Dehaene, *Senior Member, IEEE*

***Abstract*— This paper presents a near-threshold operating voltage timing error detecting 32-bit microcontroller system. The lightweight *in situ* error detection and correction technique uses a soft-edge flip-flop combined with in-latch transition detection and a set-dominant error latch to detect data path transitions after the clock edge. Inherent error correction is achieved through time borrowing in soft-edge flip-flops. The technique is implemented in an ARM Cortex M0 microcontroller system in 40-nm CMOS, rendering the microcontroller "timing error aware." Automatic critical path analysis results in an optimized timing error detection window and sparse flip-flop replacement. An autonomous dynamic voltage scaling (DVS) loop facilitates automatic operation at the point of first failure. The M0 system operates down to 290 mV and achieves 11–18 pJ/cycle core energy consumption in a 5–30 MHz frequency ranges. The architecture profits optimally from ULV operation at frequencies <10 MHz, where intra-die variations are significant.**

***Index Terms*— Adaptive circuits, better-than-worst case design, CMOS digital integrated circuits, energy-efficient digital design, error detection and correction (EDAC), microcontroller, near-threshold, point of first failure (PoFF), razor, soft edge flip-flop, time borrowing, timing error resilience, variation tolerance, voltage scaling.**

I. Introduction

REAL-TIME operating conditions of digital integrated circuits play a key role in state-of-the-art systems. The overhead resulting from margining for worst case conditions is now a major energy contributor. This results in speed, voltage, and energy operating points which are far from ideal. At the same time, there is a continued strive toward ultra-low energy operation: the demand for a ubiquitous sensatory environment was never higher. Margining for process, voltage and temperature variations or aging can compromise the low energy operation of such devices. Especially when they are being operated at ultra-low voltage, as is often suggested. While recent works like [1] succeed in achieving the necessary performance specifications to enable this kind of sensor processing, overhead due to margins is what is preventing a large-scale deployment of ultra-low voltage operated systems. Since these systems are increasingly susceptible to variations, the challenge to overcome margin induced overhead is vital.

Textbook approaches to overcome margin induced overhead have been discussed extensively in the literature. Post-fabrication sorting of ICs according to their (measured) performance (binning) is a typical low key solution, but comes with a high test cost scaling with production volume. A different solution is on-chip performance monitors, the simplest of which is a replica delay line. Since it shares most conditions with the actual system, it can be used to tune the entire system to the operating conditions. dynamic voltage frequency scaling enabled systems can benefit from such replica monitoring, tuning out a significant part of the margin induced overhead using well-engineered monitors. Ultra-low voltage systems, especially in deep submicron technologies, benefit marginally from this strategy due to high intra-die variation susceptibility. Reference [2] shows the correlation between estimated and real performance degrades at lower supply voltages. Activity-dependent aging and location-specific voltage droop are other effects difficulting the delay line-based performance monitoring.

In situ performance monitoring through error detection and correction (EDAC) techniques is a concept which has grown popular in the last two decades. By monitoring the actual logic paths in the system, information concerning the margined operation can be gathered. In the ideal case, this information is quite accurate, overcoming almost all overhead due to timing margins. By detecting the onset of errors or correcting them, the system can operate close to, at, or beyond its point of first failure (PoFF). The overall performance of such a system is a balance between margin reduction and EDAC overhead. This balance is skewed heavily by implementation-dependent effects such as hold time buffering or timing error detection window (DW) generation and pipeline restore/correction overhead. Enabling such techniques at ultra-low voltage poses severe challenges and often compromises low power or low overhead solutions. Ultra-low voltage systems operate around a flat optimal energy consumption [3] called the minimum energy point (MEP). Decreasing the supply voltage by means of (EDAC-enabled) dynamic voltage scaling (DVS) thus results in marginal energy/cycle reduction. The potential of *in situ* EDAC monitoring lies in its ability to operate the circuit with heavily reduced margins.

The key idea of this paper is to enable timing margin elimination through a lightweight EDAC system at near-threshold

Manuscript received November 23, 2017; revised February 2, 2018 and March 22, 2018; accepted March 26, 2018. This work was supported by the IWT–Flemish Fund for Innovation by Science and Technology. This paper was approved by Guest Editor Shidhartha Das. *(Corresponding author: Hans Reyserhove.)*

The authors are with the Department of Electrical Engineering, KU Leuven, B-3001 Heverlee, Belgium (e-mail: hans.reyserhove@esat.kuleuven.be; wim.dehaene@esat.kuleuven.be).

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/JSSC.2018.2821121

supply voltages. It expands the conclusions presented in [4]. Fast and robust transition detection (TD) is enabled by detecting data transitions across the master latch of the flip-flop. By delaying the master latch clock compared to the slave latch clock, a transparency window is created realizing a soft-edge flip-flop [5], [6]. The resulting operation is similar to a pulsed latch [7]–[12]: it allows late data to propagate during a transparency window after the rising edge of the clock, thus enabling error masking. Through error masking, the large overhead of error correction can be reduced while information regarding the margined operation of the circuit is retrieved. Local clock and timing window generation enable relatively precise monitoring, while the master/slave flip-flop can operate using an arbitrary small time borrow window. This limits the overhead due to increased hold time constraints. The concept is implemented in 40-nm CMOS in a 32-bit ARM Cortex M0 microcontroller system and shows up to 75% energy gains compared to a measured identical baseline system without EDAC operating under slow-slow process corner margined conditions. The technique presented in this paper mainly targets static or slow-varying but highly unpredictable variations such as intra-die variations, as discussed further in Section II-E. It overcomes this poor predictability and leverages the energy gains possible at ultra-low supply voltage. Although the reported gains are application specific, the concept is generic and lightweight. Thus, other systems will benefit similarly from the concept introduced in this paper.

The remainder of this paper is organized as follows. Section II discusses recent EDAC strategies and more specifically those targeting low voltage operation. Section III shows the proposed concept in more depth. Section IV discusses the augmented design flow for EDAC implementation, while Section V gives an overview of the implemented microcontroller system. Section VI shows the achieved measurement results and compares with other (margined) approaches. Finally, Section VII concludes this paper.

II. EDAC Review

Trying to operate circuits at the edge of their speed limit is a challenge inherent to clock edge-triggered sequential logic-based pipelines. While the concept of a clocked pipeline facilitates timing analysis and thus predictability, reserving a fixed clock period to complete operations results in a safety margin. While completion detection strategies [13] tackle the core of this problem, they are asynchronous in nature; and thus, bring with them all the problems of non-synchronous large-scale integration. The self-tuned system as described in [14] combines the best of both worlds by augmenting synchronous circuits with operation completion information. It can operate independent of absolute delay values, adjusting parameters until the "set point" is attained. This method of *in situ* performance monitoring and control is similar to EDAC strategies in many aspects, as they augment synchronous circuits with operation completion information. In accordance to [14], any EDAC strategy implies that the completion information can be used to control some circuit parameters, e.g., DV(F)S.

Extensive work has been proposed in the literature on EDAC strategies. They augment or convert the traditional flip-flop-based pipeline to enable EDAC in the sequential element. The sequential elements used are a flip-flop and/or latch combination, clocked by a traditional 50% duty cycle clock, a pulsed clock, or a two-phased non-overlapping clock. Most of the EDAC implementations can be classified in two major error detection strategies and three major correction strategies. Error detection usually exists of either transition detection (TD) or double sampling (DS), while error correction (or the lack thereof) occurs through prediction, correction, and/or masking.

A. Sequential Element

EDAC literature typically employs either a flip-flop-based pipeline [9], [15], [16], a pulsed latch-based pipeline [7]–[12], or a two-phased latch-based pipeline [17], [18] as their sequential element. A flip-flop-based EDAC element relates closest to the conventional flip-flop-based pipeline, but often increases clock load and area [15]. Pulsed latches behave similar to a flip-flop apart from the transparency window that they enable during the high phase of the pulsed clock. This window can be leveraged to enable time borrowing [7]. EDAC-enabled flip-flops and pulsed latches impose tight short path constraints, since they monitor or sample data arriving after the clock edge. Short paths, thus, require padding to extend their data transitions beyond the monitored window. Depending on the applied DW, this can have an extensive system area and energy impact. Two-phase latch-based pipelines [17], [18] overcome this requirement as they do not propagate short paths due to non-overlapping clocks. However, conversion of a flip-flop-based pipeline to a two-phase latch-based pipeline does introduce significant area and energy overhead. Such retiming can increase the number of sequential elements significantly, as well as more than double the clock load, resulting in $>10\%$ area and energy overhead [17], [19].

Reference [19] elaborates extensively on the comparison between flip-flop, pulsed latch, and two-phase latch in an EDAC multiplier implementation. Sequential area overhead is comparable when using flip-flops or pulsed latches, while two-phased latches increase sequential area significantly. Error detector insertion rate favors sparse insertion in a flip-flop or pulsed latch-based pipeline. Combinational area overhead is most impacted by short path padding and is, thus, influenced by the timing DW and the short paths adhering to the monitored critical path. When good practice short padding techniques like multi-V_T padding cells are used for similar timing DWs, no significant differences in combinational area overhead can be expected between flip-flops or pulsed latches.

Pulsed latches benefit from smaller sequential size, since they eliminate the master latch of a flip-flop. A clock pulse opens and locks the feedback loop of the latch. Reliable distribution of such a clock pulse can put major constraints on the clock tree, especially in variation-prone ultra-low voltage conditions. Local pulse generation in the latch [5] or at the lower level nodes of the clock tree [12] may offer a solution. However, the latch propagation delay puts a lower bound on the clock pulsewidth and slew rate. Since the clock pulse

almost always acts as the DW, this prevents timing DWs of an arbitrary small size and increases the necessary short path padding.

As given in Table I, a few recent works enable near-V_T supply voltage operation. Reference [18] is the only work which realizes near-V_T supply voltage operation and uses two-phased latch-based operation to do so. This paper employs robust flip-flops with operation similar to pulsed latches and does enable the near-V_T operation. Overhead due to the choice of sequential element is limited, and system overhead is heavily impacted by other factors such as detection/correction strategy, DW size, sparse replacement, and the implementation timing histogram.

B. Error Detection

Detecting errors implies information redundancy. Such redundancy can be achieved either spatially, timing-wise, or through a combination of both [20]. Spatial redundancy typically leads to large hardware overhead, which is why most EDAC implementations choose for timing redundancy [7]–[9], [11], [12], [15]–[18], [21]–[25]. The easiest way to provide timing redundancy in a clock edge-triggered sequential logic-based pipeline is by sampling the data at two distinct moments in time: double sampling (DS). It relies on the fact that the logic takes a finite amount of time to compute, making the sample taken at t_2 more likely to be correct than the one taken at t_1. Reference [15] does this by adding a (shadow) latch to the traditional flip-flop to sample the data path for the second time, relying on the high clock phase to provide the time shift. Reference [8] also switches the flip-flop and latch, enabling time borrowing in the main sequential element, known as double sampling time borrowing (DSTB). Reference [9] succeeds in disconnecting master and slave latch of the traditional flip-flop, each sampling the data path at distinct times (TIMBER). Reference [17] employs latches only, thus detecting time borrowing events. Reference [25] also uses two latches, but compares the shadow latch value with incoming data, while enabling time borrowing for the main latch.

Another approach is TD: after initially sampling the data path, any subsequent data path transitions can be considered as correct data arriving late, making the initial captured data faulty. The lack of additional sequential logic relaxes the clock network constraint. While also providing redundant data, TD flags any data transition, thus also single-event upsets (SEUs) and glitches which otherwise might not be visible using DS. Since the TD does not store any information, [7], [8], [11], and [23] augment it with a set-dominant latch (SDL). Reference [21] detects transitions halfway the data path on the negative clock edge, thus, taking half path delay information as an estimate for full path delay. Virtual supply node monitoring (virtual V_{dd} TD) is a convenient way of providing lightweight TD as in [12], [18], and [22] by checking the conditional charging of the internal nodes of the tri-state inverter present in the latch.

C. Detection Window

To distinguish between correct and incorrect data (DS) or in-time and late transitions (TD), all architectures rely on a predefined timing DW. Hence, the DW choice plays a crucial role in the error decision taking. References [8], [11], [15], [17], [18], [21], [23], and [25] reuse a single clock phase to determine the DW. A reduced duty cycle is a convenient way to modulate the DW in that case. Reference [22] tunes the DW to the edge of short path delays. References [7] and [9] provide a tunable DW unrelated to the clock. Reference [24] provides an optimal DW by closing the DW only when data arrives using a data arrival detector. While using a single clock phase as DW is a lower overhead solution compared to dedicated DW generation, such a large DW often results in severe short path constraints. Fixing these hold time constraints can lead to even higher overheads, depending on the implementation. Pulsed latches also constrain the pulsed clock, inferring their constraints on the DW.

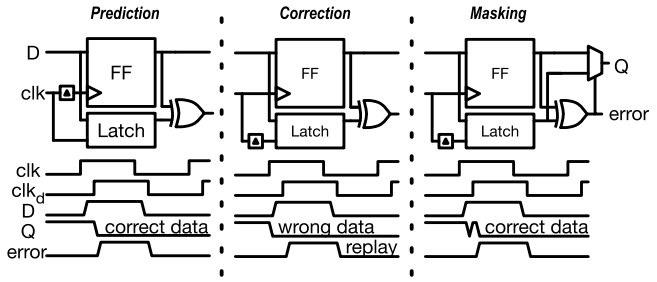


Fig. 1. Overview of different EDAC correction strategies using DS: prediction (left), correction (middle), and masking (right).

D. Error Correction

To guarantee functional correctness under all conditions, EDAC strategies employ error correction. While linked to the error detection strategy, correction strategies are often interchangable. They can be categorized in three categories: prediction, correction, and masking. Fig. 1 illustrates all three strategies using DS, but they can be employed equally with TD. Prediction detects errors before the clock edge, thus, providing completion information before actual failure. This strategy can prevent the system from introducing actual errors. While the margin necessary to prevent errors is similar to canary circuits, *in situ* prediction can overcome margins due to intra-die variation. Reference [21] detects transitions occurring in the second half of the clock cycle halfway the data path. As such, it predicts the onset of an error, gating the next clock cycle to prevent the error from occurring. Reference [23] prevents errors from occurring by elongating the clock phase when time borrowing events are detected. Because of the time borrowing enabled in this design, actual system errors would only occur after multiple cascaded time borrowing events or voltage/frequency scaling despite error detection. Although *in situ* error prediction can allow near failure operation, it requires enough margin to guarantee error free operation in all conditions. In such a context, *in situ* error prediction introduces a large overhead while only marginally improving performance. Error correction or masking is, therefore, preferred for removing the additional margin and benefiting optimally from *in situ* detection.

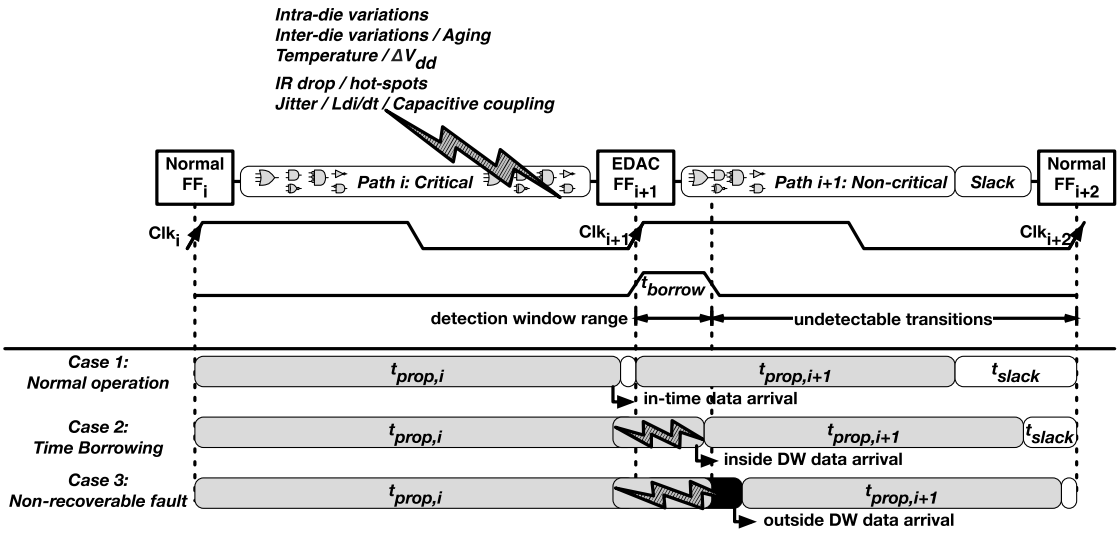


Fig. 2. Incidence of variations on timing critical paths. Case 1: data arrives in time; case 2: data arrives in DW, time borrowing resolves timing error; case 3: data arrives beyond DW, resulting in misdetection, and non-recoverable fault.

Error correction detects miscaptured data after the clock edge. Without intervention, this data would ripple through the pipeline resulting in system corruption. Reference [15] implements a flip-flop level replay of the data. This nullifies data in subsequent pipeline stages, but guarantees forward progress even at critical voltage/frequency conditions with single cycle throughput delay. Reference [8] replays instructions in the event of timing errors. Reference [17] propagates a bubble through the latch-based pipeline which gates the instruction flow, resulting in a single cycle delay. Reference [11] tracks time borrowing and uses a logic estimator to correct data in a high-speed DSP data path, but cannot do the same with critical control signal paths.

Error masking correctly propagates data arriving after the clock edge, thus, not corrupting the pipeline in cases where it would have been corrupted using normal edge-triggered sampling. It is employed frequently in combination with latches, since they allow data to ripple through in the transparent clock phase. Such time borrow events cannot be cascaded indefinitely without resulting in actual errors, since the borrowed calculation time adds up in subsequent pipeline stages. Reference [9] provides a hybrid approach which allows masking through latch-based time borrowing in predetermined intervals while reducing the clock frequency when multiple time borrow events take place. Reference [18] boosts the supply voltage of the subsequent pipeline stage in the case of a time borrow event, providing the necessary speed-up to overcome a timing error. Reference [25] provides the same speed-up by swapping the voltages applied to the device wells, thus employing a simple body biasing scheme. Reference [12] restores the correct data in the latch after the timing window. As such, it maximizes the borrowed time and relies on clock gating to insert a single cycle stall to provide enough calculation time for the new data to compute correctly in the next pipeline stage. While latch-based pipelines seem like a low overhead method to implement error masking, they often suffer from (implementation dependent) problems like higher clock tree loading, more sequential logic, and more stringent short path constraints, thus introducing overhead elsewhere.

E. EDAC-Based Variation Resilient Operation

In accordance to [16], variations can be classified according to their spatial and temporal properties. Intra-die variations influence system performance very locally, but are static or very slow. Inter-die variations and aging effects are equally static but have more of a global performance impact. Ambient temperature variations or supply voltage fluctuations are more dynamic, as are local IR drop and temperature hot-spots. Jitter, Ldi/dt effects, and capacitive coupling are fast-changing with capacitive coupling and clock jitter having the most local impact.

In situ error detection simply detects timing errors whatever their cause, whether it is local or global, slow or fast. As such, EDAC strategies have been proposed to overcome most of the margins induced by these effects. The *in situ* monitoring that they provide is inherently good for local variations and can, thus, outperform global monitoring and compensation techniques such as replica biasing. Ultra-low voltage operated systems exhibit more variation-induced performance shifting. Enabling EDAC at ultra-low voltage can, thus, improve these systems significantly, benefiting optimally from the energy reduction realized through ultra-low voltage operation. Both in DS and TD, almost all EDAC techniques do apply a predetermined DW. This implicitly assumes none of the variations overstretch data arrival outside of the DW. Data arrival outside the DW is not detected resulting in corrupt data, whatever the correction strategy may be. This is illustrated in Fig. 2. Any effect causing the propagation delay of path i to be larger that T_{clk} can be overcome with time borrowing, assuming the effect completes within the DW range.

Correction strategies can differ in their ability to cope with fast-changing variations. Error masking through time

borrowing relies on slack in subsequent data paths to resolve the error, as shown in Fig. 2. However, first and foremost, the affected data propagation should complete in the current cycle with time borrowing. Clock gating strategies [12] often gate the next clock cycle (Clk_{i+2} as shown in Fig. 2), thus, making sure the subsequent data paths have enough time to complete despite time borrowed. This strategy does not improve data completion of the affected path, and still relies on time borrowing for path i to complete: $t_{\text{prop,path}_i}$ remains the same, as does Clk_{i+1}. Only by gating Clk_{i+1}, $t_{\text{prop,path}_i}$ can be allowed to extend further. Gating Clk_{i+1} implies propagating the detected error to the clock tree root with enough margin for the insertion delay of the clock tree. Such margin can have a significant performance impact which should be outweighed against targeted improvements. Extensive time borrowing can provide the affected path with enough slack to overcome the fast-changing-induced delay penalty, but directly impacts short path padding. Canceling the affected data path and completely replaying the instruction pipeline does provide the correct data, but results in an extensive energy/throughput overhead.

The strategy proposed in this paper does not intend to overcome fast-changing variations that stretch $t_{\text{prop,path}_i}$ beyond $T_{\text{clk}} + t_{\text{borrow}}$. Its primary aim is to overcome static or slow-changing variations. The *in situ* monitoring applied in this paper targets ultra-low voltage application and was designed accordingly. In ultra-low voltage system sign-off, local variations can induce major design margins. While designing for fast-changing variations like local voltage droop may shift the design decisions proposed in this paper, it often requires more extensive techniques, e.g., as presented in [26] and [27]. The cost of overcoming fast-changing variations using traditional techniques should be considered. The system targeted in this paper, operated at near-threshold supply voltage, has a fairly high leakage versus active current ratio (10% or more). Thus, aggressive current changes triggering large local voltage droop are not a major design concern. That being said, the proposed work does not prevent techniques such as in [26] to be used.

III. Proposed Concept and Analysis

A. Key Concepts

The proposed work overcomes timing errors in the circuit pipeline caused by the reduced guard band operation. Late data transitions which would result in faulty operation in a normal flip-flop-based pipeline are overcome using timing error masking flip-flops. Such late data transition events are flagged using a TD and propagated to the system level to provide information regarding the margined operation of the circuit. This enables closed-loop DVS operation.

Fig. 3 shows a block diagram of the timing error masking flip-flop. It combines a timing control block, a master and a slave latch, a transition detector, and an error latch to detect and inherently correct data which arrives late. The timing control block splits up the master and the slave clock. By delaying the master clock relative to the slave clock, a transparency window is created: data arriving after the slave clock rising edge but before the master clock rising edge can instantly propagate through the flip-flop, while still being locked on

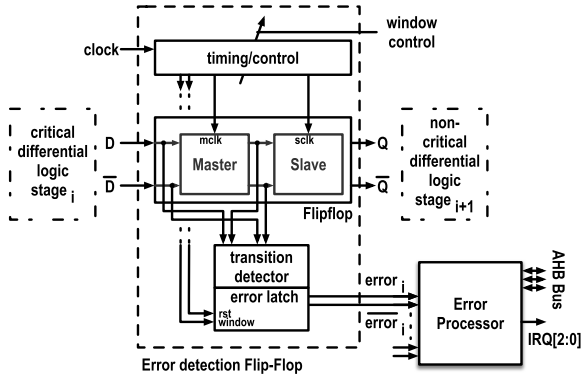


Fig. 3. Overview of timing error masking flip-flop.

the rising edge of the master clock. Using such a soft-edge flip-flop [5], [6] enables error masking in a flip-flop-based pipeline. Thus, late arriving data is masked and tunneled through the flip-flop, despite arriving after the clock edge. In this case, late arriving data is flagged by means of a TD. The TD compares data before and after the master latch. Since the master latch is transparent at this moment incoming data transitions result in a detectable delay. Since the correction is instantaneous and old data is not latched, the occurrence of such an event needs to be stored. This is done by triggering a set dominant error latch. As the soft-edge flip-flop allows data transitions after the clock edge, it effectively borrows time from the subsequent pipeline stage, similar to a time borrowing latch. This results in a more stringent constraint on the next pipeline stage, as shown in Section III-C. The resulting operation of the soft-edge flip-flop is identical to a pulsed latch [7]–[12] but has the benefit of being operated by 50% duty cycle clock while still employing an arbitrary small amount of time borrowing. In addition, pulsed latch time borrowing is constrained by the minimum clock pulsewidth to toggle the (ultra-low voltage) latch.

B. Transistor Level Design

While the proposed strategy is generic, the transistor level implementation has been realized for the ultra-low voltage-enabled differential transmission gate circuits as presented in [1]. As shown in Fig. 4, it thus equips a differential input–differential output latch with a single clock. The skewed master and slave clock are generated in the timing control block using a delay line. The delay line is externally biased for silicon debug, but operated at the same ultra-low voltage as the logic during measurements. The same block also deduces a DW signal used in the set-dominant error latch.

The transition detector XNORs the input of the master and the slave latch. As differential signals are readily available, this can easily be implemented in a complementary structure and can equally detect rising and falling transitions. The transition detector can set the error latch during the DW. At the end of every clock cycle, the error latch is reset. Reset needs to occur before the next DW to facilitate new error detection, but after the error data is captured. Using the flip-flop clock root signal and the slave clock, the error latch can reliably be reset before the next DW.

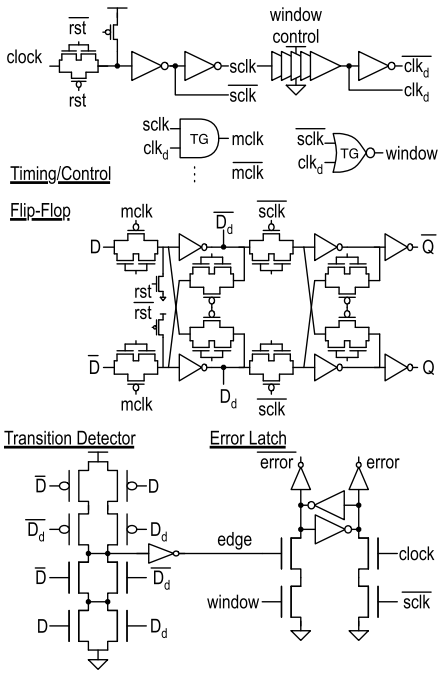


Fig. 4. Transistor level implementation of the timing error masking flip-flop.

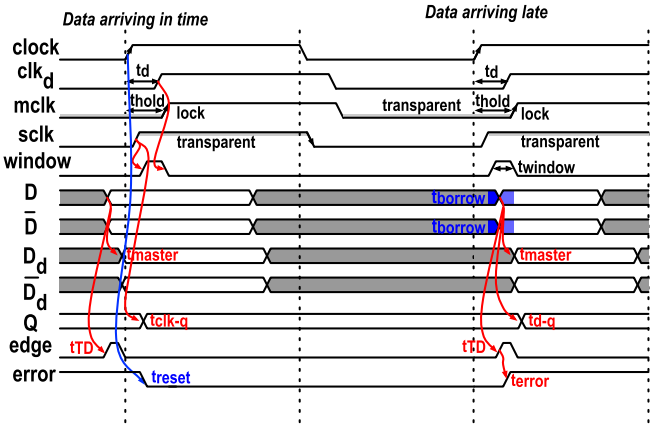


Fig. 5. Timing diagram of the timing error masking flip-flop operation.

C. Timing Constraints

While the timing error masking flip-flop operates in conjuction with a normal flip-flop-based pipeline and mostly behaves similar, it does have some altered timing constraints. In normal in-time operation, the timing error masking flip-flop behaves like a flip-flop. Detailed operation of the timing error masking flip-flop is shown in the timing diagram in Fig. 5. The delayed master clock relaxes the setup time t_{setup} and tightens the hold time t_{hold} by allowing time borrowing. Suppose a sequence of pipeline stages where stage i is replaced by a timing error masking flip-flop, the constraints on the period T_i and T_{i+1} can then be described as follows:

$$T_i \geq t_{\text{clk}-q,i-1} + \max(t_{p,\text{logic},i}) + t_{\text{setup},i} - t_{\text{borrow},i} \quad (1)$$

$$T_{i+1} \geq t_{d-q,i} + \max(t_{p,\text{logic},i+1}) + t_{\text{setup},i+1} + t_{\text{borrow},i} \quad (2)$$

$$\text{with } t_{\text{borrow},i} \leq t_{\text{window}}. \quad (3)$$

The hold time constraint for path i is tightened and can be described as follows:

$$t_{\text{hold},i} \leq t_{c,\text{clk}-q,i-1} + \min(t_{p,\text{logic},i}) - t_{\text{window}}. \quad (4)$$

Note the difference between (1) and (4). While the enabled time borrowing window is not necessarily equipped fully, it does reflect in full in the tightened hold time constraint. Path $i+1$ is limited by the $data-q$ delay rather than the $clock-q$ delay and receives an additional penalty because of time borrowing in path i. The setup time origins from the difference in delay between the clock and the data path for the master latch. Since the master clock edge is deliberately postponed, relating the setup time to the *clock* net in Fig. 4 typically results in a negative setup time. This corresponds to the intuitive analysis in Section III-A: the data can arrive after the clock edge without corrupting the system.

D. Inherent Error Correction

While the additional logic is necessary to flag a timing error event, i.e., timing error detection, the timing error correction is inherent to the system. Because the flip-flop allows data to propagate during the transparency window, *normally wrong data* is propagated correctly because of time borrowing. This allows operation at or close to the PoFF. In such error correction system, it is imperative to not scale the supply voltage beyond the PoFF in order to avoid timing errors which cannot be compensated through time borrowing. In addition, the subsequent pipeline stage should have enough margin to compensate for the borrowed time. Fortunately, this is often the case in microprocessors: [9] demonstrates few critical path endpoint flip-flops have critical paths originating from them. The design flow demonstrated in Section IV is generic in the sense that these cases do not require special attention: if a critical path endpoint is the start of an other critical path, the endpoint of the new critical path is equally equipped with timing error detection, hence allowing multi-stage time borrowing. In designs where time borrowing overconstrains subsequent paths, additional error correction mechanisms may be required. Gating the next clock cycle is a low key solution to provide enough timing slack for paths to complete. However, the throughput decrease and energy per operation increase should be taken into account.

E. Ultra-Low Voltage Operation

A critical part of the timing error flip-flop for ultra-low voltage operation is the TD. A fast transition at the input of the master latch is elongated due to the input transmission gate. The pull-up network in the transition detector is weakened to enable fast pull-down. In addition, the short in the pull-down network allows mismatch between differential signals to trigger the error latch as well. Aided by the relatively weak PMOS devices at ULV in this technology, this results in a small area transition detector. The set/reset pull-down network in the error latch is scaled up according to the same logic. The flip-flop operation was verified under intra- and inter-die variations down to 350 mV. The silicon measurements presented in Section VI revealed the most critical part to be the

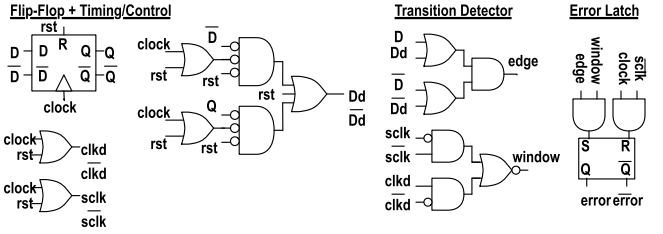


Fig. 6. Functional description used for characterization of the timing error flip-flop.

error latch reset. At the lowest supply voltages, the $clock-\overline{sclk}$ overlap was insufficient to reset the error latch.

F. Overhead

As can be expected, the timing error flip-flop introduces some overhead. The timing/control block is equipped in every timing error flip-flop and results in the biggest area overhead. The transition detector and error latch can be implemented with relatively low area. The net flip-flop area overhead due to timing error detection is 76–92% depending on the drive strength. $Clk-q$ delay is increased by 12–19%. Clock energy is 2.2–2.8× larger, and cell leakage power is increased by 52–89%. Sharing the timing/control block across multiple flip-flops similar to [12] can significantly reduce this overhead. However, such a strategy can compromise ULV operation since the constraints on the slew rate and relative skew of the clock and window signals are strict. As shown in Section V, the total area overhead attributed to timing error detection and timing error processing is limited to 7% due to sparse flip-flop replacement.

IV. Modeling and Automated Design Flow

It is crucial to enable an automated design flow to equip timing error detection. Otherwise, large scale integration on the same scale as current digital designs is impossible, rendering the technique unusable. Such an automated design flow equips standard cell libraries with logic/timing/power information to facilitate synthesis, timing analysis, and physical implementation.

A. Standard Cell Description

The complex behavior of the timing error flip-flop is modeled to fit a standard cell description and is characterized across multiple voltages and corners. The functional descriptions used for the timing error flip-flop are shown in Fig. 6. A functional subdivision is chosen so as to avoid output-to-output relationships and temporal dependencies, e.g., the $window$ signal is modeled as a logic level resulting from $sclk$ and $clkd$ rather than a pulse originating from the $clock$ signal. Intermediate signals (e.g., D_d, $edge$, $window$, ...) are characterized in a small range of slew/load conditions to accurately model the limited interconnect, slew and load at those nodes. To accurately represent the time borrowing functionality of the system, time borrowing is modeled as a negative setup time, while significantly increasing hold time by an amount equal to the transparency window.

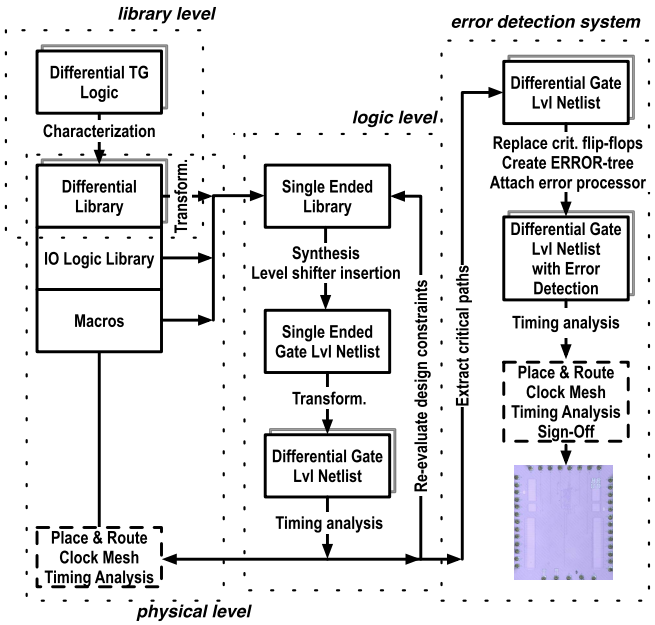


Fig. 7. Flowchart of the differential standard cell design flow from [1] augmented with timing error detection.

B. Augmented Design Flow

To fit the timing error flip-flop strategy in a generic design, the design flow presented in [1] is extended to incorporate timing error detection (see Fig. 7). Apart from the original flow, the final gate level netlist is evaluated for critical path endpoint flip-flops. A subset of these flip-flops is replaced by the timing error flip-flop according to the strategy presented in Section V-B. $Error$ signals are gathered at every hierarchy and routed to a single system level *Error Processor* capable of making intelligent decisions based on the arriving timing errors (see Section V). As mentioned in Section III-C, the timing error flip-flop relaxes the critical path constraint due to time borrowing. While this is the main goal of this architecture, the timing optimizer should not equip this margin at design time, since it would nullify the predetermined functionality. Hence, an additional constraint based on the timing analysis from the original design is imposed on the timing error enabled path.

V. Design of the 32-bit Microcontroller

To demonstrate the operation of the timing error flip-flop presented in Section III and evaluate the system level implications of such a timing error detection system, the principle was integrated in a 32-bit microcontroller system. An overview of the system is shown in Fig. 8. It equips an ARM Cortex M0 core with AHB enabled peripherals (UART, GP-IO, and TEST/DEBUG) and a 64-KB SRAM memory, identical to the system described in [1]. Using the design flow shown in Section IV, it is transformed to enable timing error detection. This implies a sparse replacement of normal flip-flops by timing error detection flip-flops, and an AHB enabled error processor rendering the system timing error aware. The error processor, as shown in Fig. 9, is a distributed error capture and

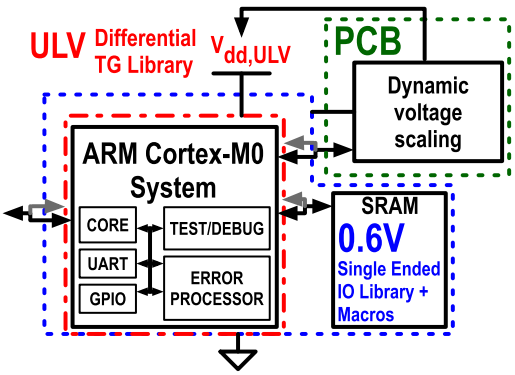


Fig. 8. System overview of the microcontroller system equipped with timing error detection.

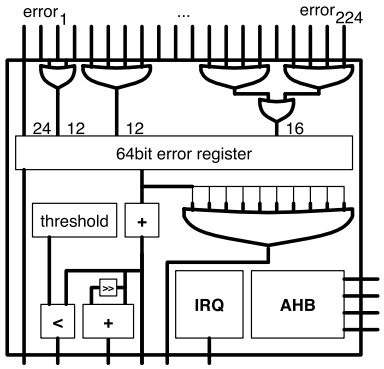


Fig. 9. Error processor enabled as a peripheral in the microcontroller system.

decision block. It gathers all error signals using a prioritized logic OR tree and maps them to a 64-bit register according to path criticalness. From here, it generates interrupts based on a programmable threshold, a running average, or single error events. The OR tree uses a single cycle, and hence, errors can be flagged using interrupts within a two-cycle delay. In its current form, the error processors add substantially to the sequential overhead. Fixing the threshold value at design time or eliminating some of the debug functionality can significantly reduce this overhead. Due to the integration of the error processor within the architecture, the microcontroller is capable of using dedicated subroutines according to error occurrence and control DVS. As such, the system can work autonomously at or near the PoFF from startup, without calibration or offline testing, resulting in the measurements shown in Section VI.

A. Detection Window Selection

Applying an efficient DW is imperative to balance energy and area overhead due to timing error detection versus margins. The choice of DW size is based on four major design considerations.

1) The DVS step size: during DVS the system should evolve from zero errors (high V_{dd}, before PoFF) to some errors (lower V_{dd}, at PoFF) before corrupting the pipeline (lowest V_{dd}, beyond PoFF). A small DW results in very fine grained DVS susceptible to noise and difficults correct operation close to the PoFF.
2) The DW directly determines the allowed amount of time borrowing. More time borrowing can overcome more timing errors, but also tightens the constraint on the subsequent pipeline stage. This makes inherent error correction less accessible, or requires more timing error detection flip-flops to be equipped at paths with more slack.
3) The DW is created in every flip-flop. While this is beneficial for skew between timing related signals such as $mclk$, $sclk$, and $window$, it introduces overhead in every flip-flop. A larger DW requires more hardware overhead to create. As shown in Fig. 4, the delay line employed in this paper is biased externally. This allows modulation of DW modulation for silicon debug but was not required during measurements.
4) As shown in (4), the DW directly impacts the hold time constraint. While timing error detection flip-flops are only equipped on critical path endpoints, an arbitrary number of short paths can have this flip-flop as endpoint. This results in a significant short path padding overhead.

The 32-bit microcontroller system was equipped with a 5% T_{clk} DW. This results in a significant hold time buffer overhead. 30% more buffers were necessarily compared to the baseline design, impacting both area and energy consumption. Hold time optimization was applied conservatively, considering fast launch paths and slow capture paths, but benefits from the use of long gate-length buffer cells. To limit the overhead of the DW generation, long device length buffers were used in the delay line shown in Fig. 4. An external bias voltage enables window size modulation from 3% to 25% of T_{clk}. However, the measurements presented in Section VI, all apply the same $V_{dd,\text{ULV}}$ as the bias voltage. Section V-B shows a 15% monitoring range, meaning a path following a timing error flip-flop either has more than 15% slack, or is equipped with a timing error flip-flop as well. Hence, in the nominal case, subsequent pipeline stages can easily make up for the borrowed time because of timing error correction. The maximum voltage step for a 5% delay penalty through DVS across corners and in a 250–500 mV, V_{dd} range is 8 mV, which is a reasonable step size for the DVS loop used in Section VI.

B. Critical Path Analysis

The energy gains realized by timing error detection systems as in [7]–[9], [11], [12], [15], [17], [18], and [21]–[25] are possible largely due to an imbalanced timing histogram: while critical paths determine the maximum clock frequency, they are outnumbered by many non-critical paths. While efforts to balance the timing histogram are necessary, pipeline imbalance remains a reality. Timing error detection systems benefit from this property through sparse flip-flop replacement: only the most critical path endpoint flip-flops are replaced. This allows timing error detection with a limited overhead.

Fig. 10 shows a histogram of all the endpoint flip-flops ordered according to the smallest timing slack path they serve. The amount of replaced path endpoint flip-flops maps to the amount of timing slack "covered" by timing error detection. Intra-die variations can increase path delay to >100% T_{clk}, resulting in a timing error. Rather than monitoring all the

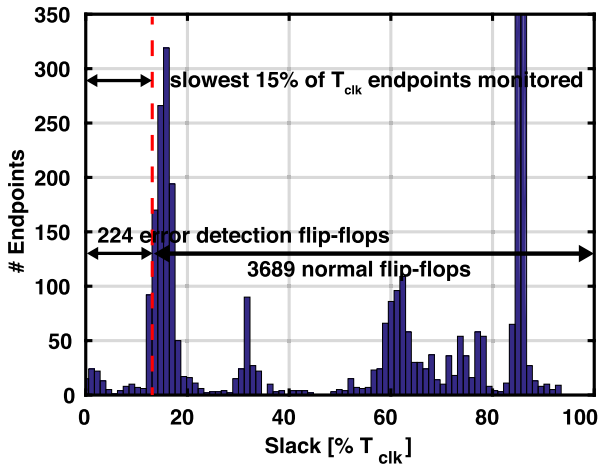


Fig. 10. Histogram of the path with the smallest timing slack at each endpoint flip-flop.

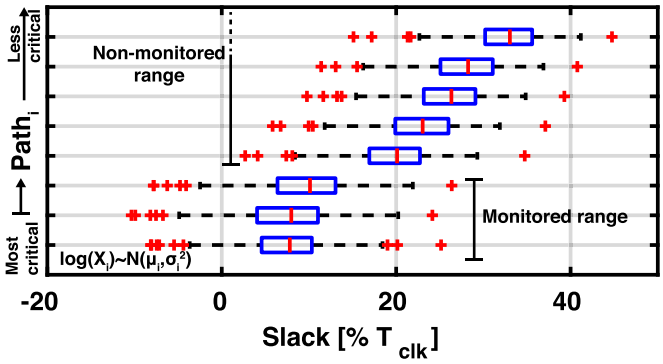
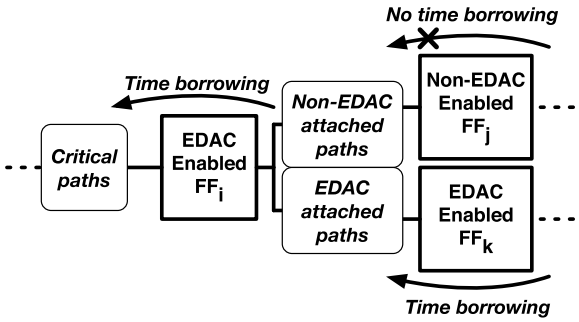


Fig. 11. Boxplot of the slack distribution of a subset of timing paths, acquired through 300 MC simulations at 300 mV.



	SS corner	TT corner	FF corner
	Non-EDAC attached paths	***Non-EDAC attached paths***	***Non-EDAC attached paths***
Min. slack	20% T_{clk}	15% T_{clk}	17% T_{clk}
Avg. slack	65% T_{clk}	42% T_{clk}	49% T_{clk}
Slack Statistics	1 path < 25% T_{clk} 19 paths < 43% T_{clk}	2 paths < 20% T_{clk} 19 paths < 30% T_{clk}	2 path < 20% T_{clk} 27 paths < 30% T_{clk}
	EDAC attached paths	***EDAC attached paths***	***EDAC attached paths***
Min. slack	6% T_{clk}	3% T_{clk}	4% T_{clk}
Avg. slack	57% T_{clk}	44% T_{clk}	50% T_{clk}
Slack Statistics	2 paths < 10% T_{clk} 9 paths < 20% T_{clk} 21 paths < 30% T_{clk}	3 paths < 10% T_{clk} 30 paths < 20% T_{clk} 51 paths < 30% T_{clk}	2 path < 10% T_{clk} 15 paths < 20% T_{clk} 47 paths < 30% T_{clk}

Fig. 12. Slack analysis of EDAC equipped paths versus non-EDAC equipped paths.

endpoints with a non-zero chance of exhibiting $>100\%$ T_{clk}, this paper approaches the sparse flip-flop insertion problem statistically. We determine the chance of false positive monitoring of the system, i.e., the chance that a non-monitored path propagates slower than all the monitored paths due to variation. In doing this, we combine the chance of all the monitored paths not failing, while a non-monitored path does. We keep in mind that the DVS loop is able to re-scale overall circuit performance to be $<100\%$ T_{clk}. This restriction prevents the system from going corrupt without flagging a timing error. To avoid this occurrence, enough timing slack should be "covered." In total, 224 out of 3913 flip-flops (6%) are replaced by timing error flip-flops. 15% of the clock period is covered. Any additional coverage would result in a significant overhead due to the large amount of endpoints in the 15–20% slack region. The chance of false positive monitoring can be described as follows:

$$P_{\text{falsepos}} = P(\exists\ p_i : t_{\text{prop},p_i} > \max(t_{\text{prop},q_j})) \qquad (5)$$

with p representing all non-monitored paths and q all monitored paths. The probability of such an event was determined using the delay distribution of a subset of paths determined by 300 Monte Carlo (MC) simulations at 300 mV (see Fig. 11). Each path is log-normally distributed. The resulting chance of false positive monitoring was determined to be less than 1 e^{-16}, decreasing with increased supply voltage. For an even deeper analysis, the activation probability of each path can be taken into account, as well as path redistribution effects at different V_{dd}.

Since the timing error detection relies on time borrowing, it constrains [see (6)] logic paths originating from the replaced timing error masking flip-flop. To this end, those attached paths should either have enough slack to complete despite the borrowed time, or should be able to borrow time in their turn. Fig. 12 shows timing statistics after replacement of the mentioned 224 flip-flops across process corners. Non-EDAC attached paths have at least 15% slack available, but on average often have close to 50% slack remaining. Attached paths with EDAC enabled flip-flops have much less slack (at least 3%), as can be expected, but can borrow time from their adjacent paths. These path endpoints return in the original 224 flip-flops, making the analysis recursive. Non-EDAC attached paths thus provide sufficient slack for time borrowing to occur and still complete correctly.

C. SRAM Interface

Since the SRAM used in this paper is a commercial macro block, the memory input interface is not capable of detecting timing errors similar to the timing error detection flip-flops. To avoid critical paths at the input of the SRAM, an additional pipeline stage is inserted at the SRAM interface. This results in a single-cycle delay penalty during SRAM access.

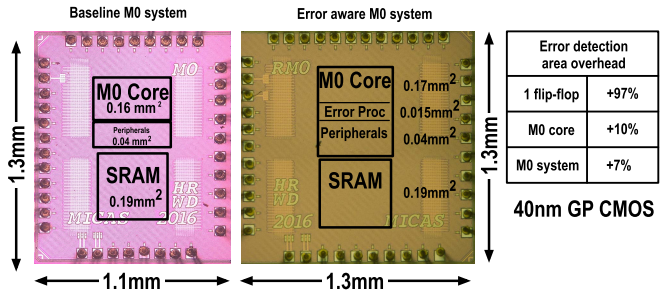


Fig. 13. Chip micrograph of the baseline design without error detection [1] (left) and with error detection system (right).

VI. Silicon Measurement Results

The timing error aware M0 system was implemented in a general purpose 40-nm CMOS process. The micrograph is shown in Fig. 13. The baseline design presented earlier in [1] allows a careful performance comparison since the RTL code is identical, except for the error detection system. Timing error detection results in a total system area overhead of 7%. The additional chip size increase compared to that in [1] is due to an increase in bond pads used for silicon debug.

A. EDAC-Based DVS and PoFF Performance

The microcontroller system with DVS control loop is set to target a frequency range from 5 to 30 MHz while running the Dhrystone benchmarking C-code. The control loop decreases the core supply voltage while continuously monitoring errors down to $V_{\mathrm{dd,PoFF}}$. At the PoFF, correct operation is still possible due to time borrowing. $V_{\mathrm{dd,PoFF}}$ allows near critical operation specific to every die, hence overcoming inter-die variation margins. The *in situ* detection approach overcomes monitor mismatch and conservatively detects the first error occurrence, since 15% of the most critical path endpoints are monitored in real time. A small additional voltage margin to compensate for fast varying conditions can be applied. The results of running the DVS loop until the first error occurrence for the applied target frequency obtained from 14 dies are shown in Fig. 14. The MEP is achieved at 7.5 MHz at 11.12 pJ/cycle and 310 mV. Correct timing error detection is realized down to 290 mV and 5 MHz. As mentioned in Section III-E, ULV operation is limited by the reset operation of the error latch. Slow-slow corner static timing analysis sign-off points were 1 MHz at 350 mV and 5 MHz at 500 mV. Operating the baseline design without error detection at these sign-off points results in a measured energy increase of more than 300% for a near-typical produced die.

Fig. 15 shows the averaged error rate per critical path group for six samples. It is the result of averaging the errors extracted from the 64-bit register in the error processor while running the Dhrystone benchmark in consecutive burst of 37 clock cycles of 10-MHz clock frequency. Errors were extracted at 3600 different times to average out influence of path activation rate and supply noise. Most timing errors occur in the 27 most critical path groups, corresponding to the 30 most critical endpoint flip-flops. Error prone paths vary significantly between individual samples. As expected, intra-die variation

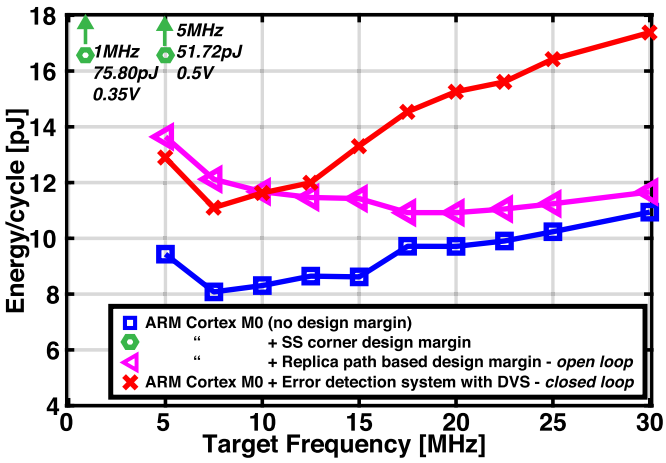


Fig. 14. Measurement of the PoFF curve for a wide frequency range, showing required energy consumption for the achieved target frequency.

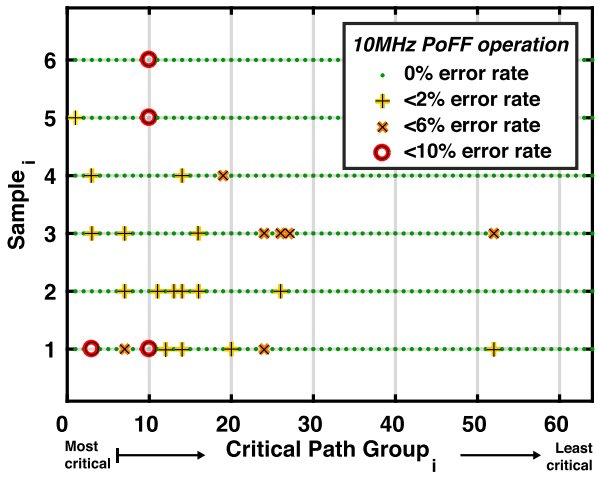


Fig. 15. Average error rate of monitored path groups for six different samples.

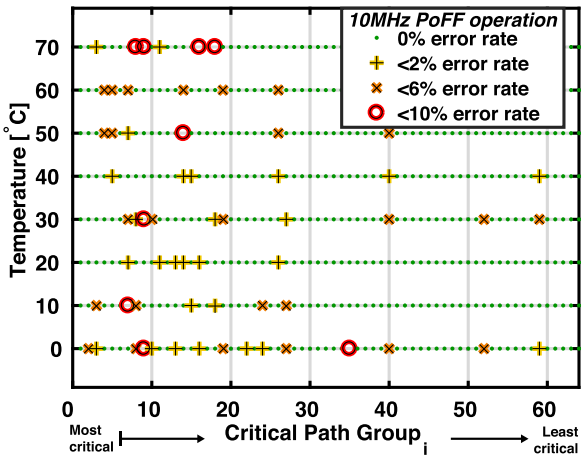


Fig. 16. Average error rate of monitored path groups for a 0 °C–70 °C temperature range.

results in a mismatch between critical path prediction and actual critical paths of static timing analysis. Fig. 16 shows a similar analysis for a single die across a 0 °C–70 °C

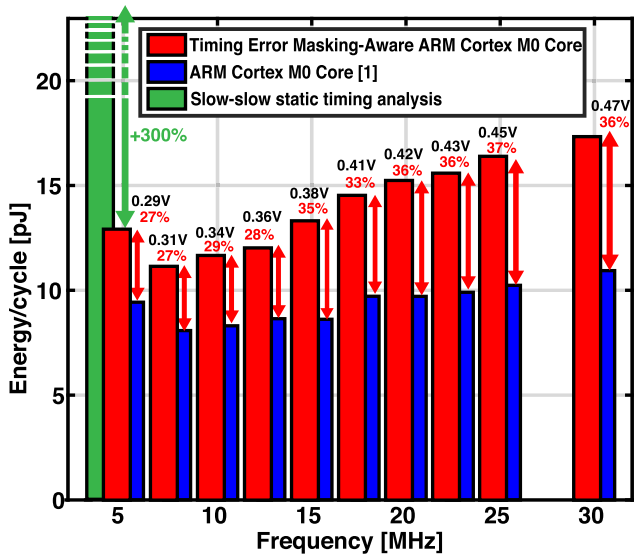


Fig. 17. Measured core energy overhead due to timing EDAC when compared to a baseline design without timing error detection.

temperature range. The 10-MHz target frequency results in a wide $V_{\mathrm{dd,PoFF}}$ range, which skews the critical path analysis from Section V-B. As a result, more path groups which were considered less critical during static timing analysis now become the most critical paths. These measurements show the importance of intra-die variations influencing the sign-off strategy. The applied critical path analysis and sparse flip-flop replacement overcome these effects and demonstrate error detection capability using time borrowing to overcome said errors under the demonstrated conditions.

B. Baseline Comparison

To evaluate the energy overhead due to the timing error detection system, energy consumption is compared to the baseline system from [1]. Both designs are operated at the same target frequency at their $V_{\mathrm{dd,PoFF}}$. Note that $V_{\mathrm{dd,PoFF}}$ for the baseline system is achieved with extensive lab tests and calibration operating in open loop, while $V_{\mathrm{dd,PoFF}}$ for the timing error detection system is achieved autonomously through the closed loop DVS system. An average energy overhead of 27–37% is observed. This energy overhead results from the increased short path padding, the additional circuits in the timing error flip-flop, and the error processor.

C. Canary Comparison

The error aware microcontroller system is compared to the baseline system equipped with a ring oscillator-based performance monitor. The speed of a 15-stage ring oscillator is mapped to the critical path speed of the baseline system in simulation, taking into account a 99.85% lognormal distribution-based confidence interval using MC simulation. An additional safety margin of 10% is taken on the mapped critical path speed. We assume a perfect process corner match between the monitor and the critical path, since they are on the same die. Lab measurements of the ring oscillator speed across the entire V_{dd} range are used to predict the process corner in a continuous operation spectrum. The resulting process corner is used to calculate the margined V_{dd} necessary to achieve the same target frequencies as in Section VI-A. The effort to predict circuit performance in such a way is non-trivial and die-specific. This effort should be outweighed against the autonomous operation of the EDAC-based system.

Energy measurements of the baseline design without error detection using these operating conditions are shown in Fig. 14. The replica path-based design achieves energy performance close to the baseline design without margin at high target frequencies, hence high V_{dd}. At the lowest target frequencies (lowest V_{dd}), intra-die variation results in a high margin, even when the replica path-based design is calibrated extensively as described. Note the replica path-based design has virtually no hardware overhead (no additional short path padding or error processor) and does require extensive post-fabrication measurement to achieve the reported performance in an open-loop system. In addition, the replica path does not take into account mismatch in process corner, has scaling mismatch under global variation conditions and cannot detect local effects such as aging or single event upsets.

D. State-of-the-Art Comparison

Table I summarizes the achieved performance and compares with eight recent similar works [7], [8], [11], [12], [16]–[18], [22]. Since implementations may differ significantly, a thorough comparison of the EDAC strategy is presented. The upper part of Table I compares the detection and correction strategies and their impact on clock generation, DW, hold time constraints, area, and ultra-low voltage operation. Clock generation, DW generation, and the impact of both at system level are not always clear and can skew overall system performance heavily. All works rely on a limited DW to capture erroneous data, thus assuming data completion in $T_{\mathrm{clk}} + t_{DW}$, whatever the cause of the timing error. The variation mitigation capability of these works, thus, primarily depends on the detection capability and if a variation source generates timing errors within the DW, rather than the correction strategy the referenced work equips. Except for this paper, [18] is the only one that enables near-threshold supply voltage operation.

The lower part of Table I focuses on the implemented system and its EDAC overhead, the technology, and how a (baseline) energy comparison is made. Different architectures can heavily influence EDAC overhead, especially when considering the short path padding overhead and sparse EDAC insertion rate. A few hundred up to 10 000+ flip-flops are reported. As with the most EDAC implementations, this paper keeps overall area overhead low due to sparse flip-flop replacement and large parts of non-sequential logic or macros in the system, e.g., SRAM. Different works report different energy reductions according to the used energy reference. This paper reports measured net energy increase compared to a separate baseline silicon implementation, optimized for operation without EDAC implementation. Few works report energy compared to a separate silicon implementation; most

TABLE I
PERFORMANCE SUMMARY AND STATE-OF-THE-ART COMPARISON

	This work	**Razor II [7] JSSC'09**	**TDTB / DSTB [8] JSSC'09**	**DSTB [16] JSSC'11**	**Bubble Razor [17] JSSC'13**	**RZL [11] JSSC'14**	**Razor-Lite [22] JSSC'14**	**Rproc [18] JSSC'15**	**iRazor [12] JSSC'17**
Sequential element	**Soft-edge flip-flop**	Latch	Latch	Flip-flop	Two-phase latch	Latch	Flip-flop	Two-phase latch	Latch
Extra transistors	**46**	31	15 / 26	28+delay chain	20+dynOR+cluster	29	8	24	3+6.5∗
Detection	**In-latch TD**	TD	TD / DS	TD	DS	TD	Virtual V_{dd} TD	Virtual V_{dd} TD	Virtual V_{dd} TD
Clock (duty cycle)	**50%**	13% and 40%	Controllable DC	50%	2-phase non-overlap.	Controllable DC	Controllable DC	2-phase non-overlap.	*not given*
Detection window DW	**Local gen. 5% T_{clk}**	Local 25FO4	High phase T_{clk}	Local	High phase T_{clk}	Global Low phase T_{clk}	High phase T_{clk} post-fabr. calibr., 16%	High phase T_{clk}	±High phase T_{clk}
t_{hold} constraint	t_{DW}	t_{DW}	t_{DW}	$\pm t_{DW}$	None	High phase T_{clk}	High phase T_{clk}	None	t_{DW}
Correction	**Time borrow**	Time borrow Instr. replay	T_{clk}/2 Instr. replay	T_{clk}/2 Instr. replay	Stall (Bubble)	Time borrow Interpolation	T_{clk}/2 Instr. replay	V_{dd} boost	1-cycle stall
FF_{area} overhead	**+76..92%**	/	/	/	/	/	+33%	+268%	+4.3%
FF_{clock} overhead	**+120..180%**	+25..70%	+38..64%/+81..143%	/	+88%	+16.9%	/	/	/
Near-V_T enabled	**Yes 0.29V**	No	No 0.7V	No 0.9V	No 0.68V	No 0.85V	No 0.83V	Yes 0.29V	No 0.6V
Architecture	**32-bit Cortex M0**	64-bit Alpha	3-stage test circuit	6-stage ARM proc.	32-bit Cortex M3	16-bit FIR	64-bit Alpha	16-bit R proc.	32-bit Cortex R4
Technology	**40nm CMOS**	130nm CMOS	65nm CMOS	65nm CMOS	45nm SOI	65nm CMOS	45nm SOI	65nm CMOS	40nm CMOS
$System_{area}$ overhead	**7%**	/	/	6.9%	87%	/	4.42%	8.3%	13.6%
Sparse insertion rate % #FFs	**224/3913 5.7%**	121/826 14.6%	/	503/2976 17%	- 100%	118/393 30%	492/2482 20%	57/445 13%	1115/12875 8.7%
Energy comparison	**unmargined baseline +26..37%**	margined EDAC -30..36%	margined baseline -31..37%	sim. basel. / marg. EDAC +9.4% / -24%	margined EDAC -54..62%	margined EDAC +25..37% efficiency	margined EDAC -45.4%	margined baseline -33..51%	margined baseline -33..41%

∗ due to shared local clock generation

compare to margined operation of the EDAC silicon, which then includes clock, DW, and short path padding overhead due to EDAC. Two-phase latch-based implementations benefit from having no short path padding requirements, but result in significant overhead when starting from a flip-flop-based design. Latch (pulsed)-based designs have less overhead, but often rely on a carefully controlled clock tree and duty cycle. A baseline pulsed latch implementation without EDAC often has the same short path padding constraints as its EDAC implementation. To the best of our knowledge, this paper presents one of the first flip-flop EDAC implementations fully functional at near-threshold supply voltage. It quantifies EDAC impact and overhead and was implemented on an industrially relevant 32-bit microcontroller.

VII. CONCLUSION

This paper discusses a lightweight timing EDAC strategy operated at ultra-low voltage. The strategy uses soft-edge flip-flops with in-latch TD to detect data arriving after the clock edge and flag it to a set-dominant error latch. Time borrowing through the soft-edge flip-flops is used to provide inherent error correction. The EDAC strategy is implemented in a 32-bit ARM Cortex M0 microcontroller system using an augmented standard cell design flow. The timing error flip-flop is fully characterized and used as any other cell in the design flow. The different design considerations needed to determine a sufficiently large timing DW are discussed extensively. Automatic sparse flip-flop replacement is applied and analyzed to limit hardware overhead and benefits from statistical path variation analysis. A 5% T_{clk} DW is used while 224 endpoint flip-flops are equipped with timing error detection, corresponding to a monitoring range of 15% T_{clk}. The silicon implementation of the M0 system operates autonomously at the PoFF through the on-chip error processor and PCB enabled DVS loop. Ultra-low measured core energy consumption (11–18 pJ/cycle) is achieved for a frequency range of 5–30 MHz. Error detection rates for different dies under different temperatures show the influence of intra-die variations and critical path redistribution due to temperature variations. Finally, the system is compared extensively with measurements from an identical baseline system without error detection, both operating at $V_{\text{dd,PoFF}}$ and at $V_{\text{dd,margined}}$ using a ring oscillator-based performance monitor. The error-aware M0 system reduces energy consumption by more than 75% compared to slow-slow corner STA operating conditions. At the lowest supply voltage, the proposed M0 system benefits optimally from *in situ* detection, since intra-die variations are dominant.

REFERENCES

[1] H. Reyserhove and W. Dehaene, "A differential transmission gate design flow for minimum energy sub-10-pJ/cycle ARM Cortex-M0 MCUs," *IEEE J. Solid-State Circuits*, vol. 52, no. 7, pp. 1904–1914, Jul. 2017.

[2] E. Beigné, "A 460 MHz at 397 mV, 2.6 GHz at 1.3 V, 32 bits VLIW DSP embedding F_{MAX} tracking," *IEEE J. Solid-State Circuits*, vol. 50, no. 1, pp. 125–136, Jan. 2015.

[3] A. Wang and A. Chandrakasan, "A 180-mV subthreshold FFT processor using a minimum energy design methodology," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 310–319, Jan. 2005.

[4] H. Reyserhove and W. Dehaene, "Design margin elimination in a near-threshold timing error masking-aware 32-bit ARM Cortex M0 in 40 nm CMOS," in *Proc. 43rd IEEE Eur. Solid State Circuits Conf. (ESSCIRC)*, Sep. 2017, pp. 155–158.

[5] H. Partovi, R. Burd, U. Salim, F. Weber, L. DiGregorio, and D. Draper, "Flow-through latch and edge-triggered flip-flop hybrid elements," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 1996, pp. 138–139.

[6] V. Joshi, D. Blaauw, and D. Sylvester, "Soft-edge flip-flops for improved timing yield: Design and optimization," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2007, pp. 667–673.

[7] S. Das *et al.*, "RazorII: *In situ* error detection and correction for PVT and SER tolerance," *IEEE J. Solid-State Circuits*, vol. 44, no. 1, pp. 32–48, Jan. 2009.

[8] K. A. Bowman *et al.*, "Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance," *IEEE J. Solid-State Circuits*, vol. 44, no. 1, pp. 49–63, Jan. 2009.

[9] M. Choudhury, V. Chandra, K. Mohanram, and R. Aitken, "TIMBER: Time borrowing and error relaying for online timing error resilience," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2010, pp. 1554–1559.

[10] K. A. Bowman *et al.*, "A 45 nm resilient microprocessor core for dynamic variation tolerance," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 194–208, Jan. 2011.

[11] P. N. Whatmough, S. Das, and D. M. Bull, "A low-power 1-GHz razor FIR accelerator with time-borrow tracking pipeline and approximate error correction in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 49, no. 1, pp. 84–94, Jan. 2014.

[12] Y. Zhang *et al.*, "iRazor: Current-based error detection and correction scheme for PVT variation in 40-nm ARM Cortex-R4 PROCESSOR," *IEEE J. Solid-State Circuits*, vol. 53, no. 2, pp. 619–631, Feb. 2018.
[13] F.-C. Cheng, "Practical design and performance evaluation of completion detection circuits," in *Proc. Int. Conf. Comput. Design, VLSI Comput. Process. (ICCD)*, Oct. 1998, pp. 354–359.
[14] T. Kehl, "Hardware self-tuning and circuit performance monitoring," in *Proc. Int. Conf. Comput. Design, VLSI Comput. Process. (ICCD)*, Oct. 1993, pp. 188–192.
[15] S. Das *et al.*, "A self-tuning DVS processor using delay-error detection and correction," *IEEE J. Solid-State Circuits*, vol. 41, no. 4, pp. 792–804, Apr. 2006.
[16] D. Bull, S. Das, K. Shivashankar, G. S. Dasika, K. Flautner, and D. Blaauw, "A power-efficient 32 bit ARM processor using timing-error detection and correction for transient-error tolerance and adaptation to PVT variation," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 18–31, Jan. 2011.
[17] 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 J. Solid-State Circuits*, vol. 48, no. 1, pp. 66–81, Jan. 2013.
[18] S. Kim and M. Seok, "Variation-tolerant, ultra-low-voltage microprocessor with a low-overhead, within-a-cycle *in-situ* timing-error detection and correction technique," *IEEE J. Solid-State Circuits*, vol. 50, no. 6, pp. 1478–1490, Jun. 2015.
[19] W. Jin, S. Kim, W. He, Z. Mao, and M. Seok, "*In situ* error detection techniques in ultralow voltage pipelines: Analysis and optimizations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1032–1043, Mar. 2017.
[20] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *Proc. 17th IEEE VLSI Test Symp.*, Apr. 1999, pp. 86–94.
[21] J. Zhou *et al.*, "Hepp: A new *in-situ* timing-error prediction and prevention technique for variation-tolerant ultra-low-voltage designs," in *Proc. IEEE Asian Solid-State Circuits Conf. (A-SSCC)*, Nov. 2013, pp. 129–132.
[22] I. Kwon, S. Kim, D. Fick, M. Kim, Y.-P. Chen, and D. Sylvester, "Razor-lite: A light-weight register for error detection by observing virtual supply rails," *IEEE J. Solid-State Circuits*, vol. 49, no. 9, pp. 2054–2066, Sep. 2014.
[23] M. Hiienkari, J. Teittinen, L. Koskinen, M. Turnquist, and M. Kaltiokallio, "A 3.15 pJ/cyc 32-bit RISC CPU with timing-error prevention and adaptive clocking in 28 nm CMOS," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Sep. 2014, pp. 1–4.
[24] M. Nejat, B. Alizadeh, and A. Afzali-Kusha, "Dynamic flip-flop conversion: A time-borrowing method for performance improvement of low-power digital circuits prone to variations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 11, pp. 2724–2727, Nov. 2015.
[25] S. Kim, J. P. Cerqueira, and M. Seok, "A 450 mv timing-margin-free waveform sorter based on body swapping error correction," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2016, pp. 1–2.
[26] K. A. Bowman, C. Tokunaga, T. Karnik, V. K. De, and J. W. Tschanz, "A 22 nm dynamically adaptive clock distribution for voltage droop tolerance," in *Proc. Symp. VLSI Circuits (VLSIC)*, Jun. 2012, pp. 94–95.
[27] P. I.-J. Chuang *et al.*, "Power supply noise in a 22 nm z13^{TM} microprocessor," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, Feb. 2017, pp. 438–439.

Hans Reyserhove (S'10) was born in Turnhout, Belgium, in 1989. He received the M.S. degree in electrical engineering from the University of Leuven (KU Leuven), Leuven, Belgium, in 2012, where he is currently pursuing the Ph.D. degree in variation-resilient near-threshold digital circuit design with a focus on automating design flows, microprocessors, and better-than-worst case design. His thesis was entitled pixel level A/D converter for extreme parallelism, high frame rate, and high dynamic range image sensors.

He is currently a Research Assistant with the ESAT-MICAS laboratories, KU Leuven. For this research, he is being sponsored by the IWT (the Flemish Institute, Flanders, Belgium, for Scientific Research in the Industry).

Wim Dehaene (SM'04) was born in Nijmegen, The Netherlands, in 1967. He received the M.Sc. degree in electrical and mechanical engineering and the Ph.D. degree from the Katholieke Universiteit Leuven (KU Leuven), Leuven, Belgium, in 1991 and 1996, respectively. His Ph.D. thesis was entitled CMOS integrated circuits for analog signal processing in hard disk systems.

In 1996, he joined Alcatel Microelectronics, Belgium, where he was a Senior Project Leader for the feasibility, design, and development of mixed mode systems on chip. The application domains were telephony, xDSL, and high speed wireless LAN. In 2002, he joined the Staff of the ESAT-MICAS Laboratory, KU Leuven. He was a Research Assistant with the ESAT-MICAS Laboratory, KU Leuven. He is currently a Full Professor and the Head of the MICAS Division, KU Leuven. He is teaching several classes on electrical engineering and digital circuit and system design. He is also very interested in the didactics of engineering. As such, he is guiding several projects aiming to bring engineering to youngsters in secondary education and he is a teacher in the teacher education program of the KU Leuven. His current research interest includes the design of novel CMOS building blocks for hard disk systems. The research was first sponsored by the IWONL (Belgian Institute for Science and Research in Industry and agriculture) and later by the IWT (the Flemish Institute for Scientific Research in the Industry). His research domain is circuit level design of digital circuits, with a focus on ultralow power signal processing and memories in advanced CMOS technologies. Part of this research is performed in cooperation with IMEC, Belgium, where he is also a Part Time Principal Scientist.

Dr. Dehaene was the Technical Program-Chair for ESSCIRC 2017. He is a member of the ISSCC program committee.