On the Efficiency of Early Bird Sampling (EBS) an Error Detection-Correction Scheme for Data-Driven Voltage Over-Scaling

Roberto G. Rizzo[1], Valentino Peluso[1], Andrea Calimera[1(✉)], and Jun Zhou[2]

[1] Department of Control and Computer Engineering, Politecnico di Torino, 10129 Turin, Italy
andrea.calimera@polito.it
[2] University of Electronic Science and Technology of China, Chengdu, China

Abstract. An efficient implementation of voltage over-scaling policies for ultra-low power ICs passes through the design of on-chip *Error Detection and Correction* (EDC) mechanisms that can provide continuous feedback about the health of the circuit. The key components of a EDC architecture are embedded timing sensors that check the compliance of timing constraints at run-time and drives the computation to safely evolve toward the minimum energy point.

While most of the existing EDC solutions, e.g., *Razor* [1], have proved hardly applicable to circuits other than pipelined processors, our recent work [2] introduced a lightweight EDC alternative for general sequential circuits, what we called *Early Bird Sampling* (EBS). As a key strength, EBS reduces the design overhead by means of a *dynamic short path padding* that alleviates the overhead of timing sensors placement. Moreover, EBS implements an error correction mechanism based on *local logic-masking*, a technique that is well suited for digital IPs w/o an instruction-set. These features make EBS a viable solution to devise Data-Driven Voltage Over-Scaling (DD-VOS) for error-resilient applications.

Aim of this work is to recap the EBS strategy and quantify its figures of merit under different power management scenarios. We thereby provide accurate overhead assessment for different benchmarks and run under different DD-VOS policies. Comparison against a state-of-art EDC scheme, i.e., Razor, demonstrates EBS shows affordable area penalty (3.6% against 71.6% of Razor), still improving the efficiency of DD-VOS. Indeed, EBS leads circuits through lower energy-per-operation (savings w.r.t. Razor range from 36.2% to 40.2%) at negligible performance loss, from 2% to 5% (as much as Razor).

Keywords: Error Detection-Correction · Energy optimization · Error-resilient applications · Data-Driven Voltage-Over-Scaling

Published by Springer Nature Switzerland AG 2019
M. Maniatakos et al. (Eds.): VLSI-SoC 2017, IFIP AICT 500, pp. 153–177, 2019.
https://doi.org/10.1007/978-3-030-15663-3_8

1 Introduction

1.1 Context

The key to success for the Internet-of-Things (IoT) is the availability of always-on smart objects with embedded Integrated Circuits (ICs) that can process/transmit sensor data ceaseless. Due to the limited budget of energy made available by small batteries [3], such ICs must show ultra-low power consumption thus to guarantee reasonable throughput [4,5]. This poses stringent design constraints that can be hardly achieved with classical low-power techniques, such as Dynamic Voltage Frequency Scaling (DVFS) [6], Clock-Gating [7], Power-Gating [8–10]. Recent trends highlight the rise of adaptive power-management strategies, e.g., Adaptive Voltage Over-scaling (AVOS) [11], which leverage the error resilience of data-driven applications in order to bring computation closer to the point of minimum energy consumption.

The strength of adaptive strategies lies under the ability of tuning low-power knobs at a finer time scale granularity, depending on the actual workload (or context). In standard DVFS both voltage (Vdd) and frequency (fc) are jointly traded over a set of discrete points statically defined at design-time on the base of the worst-case timing path. By contrast, AVOS applies Vdd lowering on the base of the longest *synthesized* timing path, yet keeping the operating frequency untouched. This guarantees larger power savings, zero throughput degradation, and thus, higher energy efficiency. To notice that knobs other than Vdd can still work, e.g., adaptive body-biasing. With no lack of generality this work focuses on AVOS.

As any other adaptive power management strategy, AVOS makes use of *Error Detection and Correction* (EDC) mechanisms that give real-time feedback on the correctness of the circuit; such information is used by the power management unit to identify the most appropriate Vdd scaling. In its more general embodiment a EDC architecture consists of in-situ timing sensors that flag the occurrence of set-up time violations across the flip-flops. The flag count is used as metric to decide whether the circuit is getting too faulty, in which case the Vdd is raised up in order to alleviate the cost of the error correction, or there still enough margin for lowering the Vdd, and thus to save more power. To notice that different Vdd scaling policies may vary depending on the flag threshold(s) that triggers the Vdd scaling, i.e., the *error rate*, the period of observation of the flag count, i.e., the *monitoring period*, the height and number of Vdd steps, i.e., the *Vdd quantization*.

Leaving out the details related to the Vdd scaling policy, the key aspect concerns how timing errors are detected and corrected, namely, the circuit implementation of the EDC architecture. The design of a reliable, low-cost EDC represents the actual bottleneck, indeed.

1.2 Motivation

Among the many EDC solutions appeared in the recent literature, *Razor* [1,12] still represents the main reference. The error detection is implemented by replacing

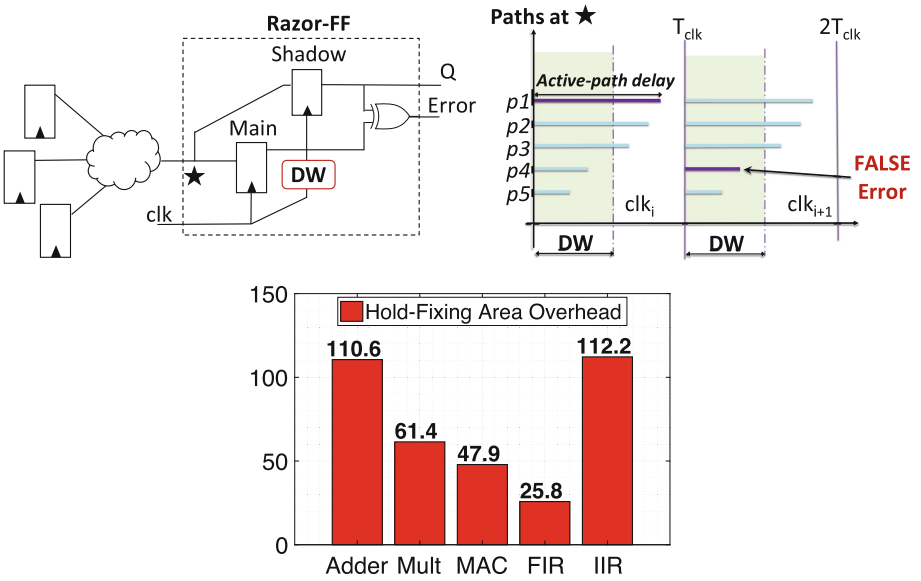


Fig. 1. Razor-FF implementation (left); short path race (right); area overhead due to short-path padding (center down), % w.r.t. baseline circuit [2].

standard flip-flops (FFs) with special FFs, a.k.a. *Razor-FFs* (Fig. 1), that sample logic signals at two different instants of time: first, at the rise edge of the clock, then, after a predefined timing window, the so called *Detection-Window* (DW). The two time-skewed samples are stored by two different FFs, i.e., the *main* flip-flop and the *shadow* flip-flop, and then compared through a XOR gate for parity check. A parity match implies the absence of errors and the availability of some timing slack, whereas a mismatch implies a faulty computation that is then recovered through some correction mechanism. To notice that Razor has been conceived for pipelined processors, hence, error recovery is accomplished through instruction replay.

Although Razor is considered a milestone in the scientific community, it shows intrinsic limitations that prevent its use on sequential circuits other than pipelined processors. The main reasons are two (described below in criticality order).

1. Short path race. While processors show a relative small number of end point FFs (the stage registers of the pipeline) most of which having a regular timing path distributions, generic sequential circuits have many FFs usually driven by logic cones with timing path distributions that seriously complicate the timing closure during logic synthesis. To better understand this critical aspect, one should consider Razor-FFs suffer the so called *short-path race*. As per their internal structure (Fig. 1), Razor-FFs cannot make distinction between the activation of a short-path within the DW and the activation of a long-path beyond the clock edge. This may cause "false" error detections. As depicted in

Fig. 1-center, the value sampled in the main FF at (T_{clk}) may differ from that sampled in the shadow FF at ($T_{clk} + DW$) due to a short path activation (p4); the error flag is then raised even if there is no timing violation.

In order to avoid overlaps between short- and long-paths, a common design practice is to apply a static short-path padding [1,12]. It is a constrained hold-time fixing procedure (*hold-fixing* hereafter) where buffers are selectively inserted in the logic cones such that the minimum arrival time of any logic path is shifted beyond DW (usually 50% of the clock-period); short paths delaying is done while keeping the longer timing paths untouched. The side effects are many. Firstly, long buffer chains induce huge area penalties. As a preliminary result, Fig. 1 shows the area overhead due to hold-fixing for the set of circuits we used as benchmarks: the worst case is 112%! Secondly, when the timing constraint on the long paths is tight, hold-fixing tries to reach timing closure turning on Boolean transformations that (*i*) further increase area (*ii*) reshape the path distribution with negative impact on Vdd scaling efficiency (more details in the experimental section). Finally, long buffer chains exacerbates the timing unpredictability due to PVT variations when the circuit works at ultra-low voltage, e.g., near-threshold [13].

2. Correction through functional redundancy. While pipelined processors offer an easy path to error correction, i.e., instruction reply, implementing the same mechanism on sequential circuits would require a too complex FSM rewind. Hence, alternative circuit strategies are needed [14]. Unfortunately, the design overhead of such correction circuitry might substantially affect the gain brought by adaptive power management.

These two issues make Razor implementation very hard, often impractical, to be adopted in low-power ICs. Also, most of the attempts made to generalize the Razor technique have turned out to be too costly. This work deals with the same design issue by describing a lightweight EDC strategy that orthogonally applies to a wider set of circuits.

1.3 Contribution

Our recent work [2] proposed a simple, yet effective EDC implementation that addresses the key limitations of Razor. We refer it as the *Early Bird Sampling* (EBS). The EBS approach improves over the Razor technique by means of two main components: *dynamic short path padding* and *local logic masking*. The former addresses the short-path race through the insertion of a *Tunable Delay Line* (TDL) shared among *all* the paths that flow onto the same end-point; the result is that of depleting the DW from short-paths thereby avoiding false error detections without any significant overhead[1]. The latter one consists of an error correction mechanism that is applied locally, i.e., on the faulty FFs, cycle-by-cycle, thereby avoiding complex flow re-execution. This solution is inspired by [14].

[1] To notice that the availability of tunable delays also enables post-silicon variability compensations (out of the scope of this work).

With this paper we elaborate more on the EBS implementation details and we quantify the figures of merit over different operating scenarios. After a brief overview of related works (Sect. 2), we give a detailed description of the EBS architecture (Sects. 3 and 4). We then demonstrate EBS improves the efficiency of adaptive power management. EBS is validated for *Data-Driven Voltage Over-Scaling* (DD-VOS) (Sect. 5), a VOS scheme where the Vdd lowering follows the actual workload; as shown later in the text, DD-VOS enhanced with EBS well fits the characteristics of error-resilient applications. A customized design framework integrated into a commercial design kit for a 28 nm FDSOI CMOS technology (Sect. 6) is used to validate EBS on a representative class of benchmarks. Those benchmarks are simulated under realistic workloads using different voltage scaling policies: *single-threshold*, *double-threshold* and *saturation-counter*. The collected results (Sect. 7) give a comparative analysis against Razor-based DD-VOS. The main achievements are as follows: (*i*) EBS reduces area overheads: 3.6% (EBS) vs. 71.6% (Razor) on average; (*ii*) EBS improves energy efficiency as it increases the voltage scaling margins DD-VOS can play with: average energy-per-operation savings (w.r.t. baseline) are 38.6% (EBS) vs. 0.7% (Razor); (*iii*) EBS induces a mere performance loss: operations-per-cycles ranges from 0.95 to 0.98 (as much as Razor).

2 Related Works

2.1 Razor-Based Overhead Reduction

In previous works, several solutions for limiting the area overhead of Razor-based monitoring schemes have been investigated. The works [12,15–17] proposed the use of a duty-cycled clock. Nevertheless, this solution was applied on specific designs and it is hard to generalize to random sequential circuit. Moreover, large detection windows may be necessary to cover delay variations over a wide voltage range operation. Authors in [18] presented a novel technique for preventing hold-fixing buffers while maintaining a traditional clock network design. Such method was demonstrated on a loop-accelerator for System-on-Chip designs. The short-path race is avoided through the insertion of negative-phase transparent latches at the middle of each timing path covered by a Razor-FF. In this way, the latches prevents signals propagation through the Razor-FFs during the high-phase of the clock. However, the arrival time to the latches may be affected by process variations, leading to an increase of the error rate. In this work, we opted for a simpler solution, with reduced area overhead and low implementation cost. In [19], the authors proposed an hold-fixing procedure based on a fine-grained load allocation that makes use of spare cells and dummy metals. The integration of this methodology in standard EDA tools might be a concern, whereas our implementation strategy is fully integrated into industrial design tools.

2.2 Existing Voltage Over-Scaling Approaches

Voltage Over-Scaling (VOS) leverages the quadratic relationship of dynamic power with supply voltage. VOS scales the Vdd below the minimum threshold

that satisfies the worst-case delay of the circuit. The implementation of this scheme requires error detection and/or correction mechanisms to properly tune the Vdd in case of set-up time violations. The basic idea underlying VOS is the empirical observation that the *sensitization* probability of long (hence, critical critical) paths is usually very low. If this is not the case, some optimization, e.g., those introduced in [20,21], can be applied in order to meet this requirement. In this way, the overhead of the correction mechanism is minimized.

As reported in [22], different implementations of the VOS strategy have been presented in the literature. The following is a taxonomy of the most representative works in the field.

Error Detection-Correction Schemes. Such techniques take corrective action according the signals coming from error monitors embedded in the logic. Razor [1,23] represents a milestone for error-detection systems and after more than ten years since its first appearance, is still the state-of-art. The Razor technique has been conceived for pipelined microprocessor architectures. In a Razor Flip-Flop, the signal is double-sampled by a *shadow* flip-flop triggered by a delayed clock (see Sect. 1). In case of detected errors, a recovery strategy is activated by a three-stage mechanism: first, the pipeline is stalled; second, the bits stored in the shadow flip-flops are loaded into the main flip-flops; finally, the last pipe-cycle is repeated. This system enable to remove excessive voltage margin, tuning the Vdd of the circuit according the error rate.

The main issue of Razor is the possible occurrence of metastability. Several works have addressed this problem. Razor II [12], an extension of Razor, proposed to use a transition detector. Another approach is to replace the Razor-FF with time borrowing latches [15,16].

In order to reduce the performance penalties introduced by error correction, authors in [11] proposed to remove any recovery circuitry. The error detection systems is used tune the Vdd according to the error rate. Such solution is suited for error resilient application, e.g. signal processing, where the degradation of the output marginally impact the quality of the final results.

Another solution is to reduce the number of activations of the recovery mechanisms by just changing the slack distribution. While standard tools optimize the longest timing path, several works [21,24–27] proposed to speed-up the most frequently exercised paths, thereby forcing timing errors on the most infrequent paths. Therefore, it is possible to further scale the voltage while maintaining the same error rate.

Prediction-Based Schemes. Differently from Razor, they are based on some prediction logic that prevents the occurrence of errors. Authors in [20] introduced a design methodology, called CRISTA, for voltage over-scaled circuits. The basic idea of CRISTA is to reduce the activation probability of the most critical paths through a customized optimization stage carried out during the logic-synthesis; then, at run-time, and provide those paths with an extra clock cycle when they are sensitized. The low activation rate of the long paths allows to minimize performance penalties.

Elastic-clock execution units [28,29] are another practical example of the CRISTA paradigm. A low overhead prediction logic check whether an input pattern exercise a critical path; then it dynamically allocates an additional clock cycle in order to meet the timing constraints. The remaining input patterns are executed in a single clock cycle.

Algorithmic Noise Tolerance (ANT). The basic principle underlying ANT is to accept errors if the output degradation is below a given threshold. Indeed, ANT [30,31] has been conceived for DSP arithmetic blocks, where the circuit output represents a quantity. Instead of using local timing monitors, the error detection is delegated to a lightweight replica of the main circuit, namely the *estimator*. The output computed by the estimator is checked against the main circuit one and a control unit flags an error if the difference overcomes an user-defined threshold. If this is the case, the estimator's output is forwarded towards the main output of the circuit. The main challenge in ANT-based systems is to limit the area and timing overhead of the estimator for complex arithmetic functions, while guaranteeing the desired output quality.

3 Early Bird Sampling

The objective we intend to pursue by proposing the Early Bird Sampling (EBS) technique is twofold: (*i*) reduce traditional design overhead imposed by Razor system, while (*ii*) maintaining those intrinsic characteristics of the circuit that enable an efficient implementation of VOS.

A schematic representation of the EBS circuit is given in Fig. 2a. Tunable Delay Lines (TDLs) are inserted just before the critical end-points of the circuit. Those end-point are equipped with a variant of the Razor-FF (more details provided later in the text).

The propagation delay of the TDLs can be changed at run-time such that the minimum arrival time (AT_{min}) is greater than the detection window (DW) of the Razor-FFs. Indeed, the delay of a TDL is given by:

$$TDL = DW - AT_{min}; \tag{1}$$

This prevents the activation of short-paths within the detection window, and so, races with long-path in setup time violation, i.e., "false" error detection. For the sake of clarity, we assumed that a TDL is tuned during post-fabrication stage, when also the nominal $Tclk$ can be properly set such that no paths can be delayed beyond the DW in nominal operating conditions. Each critical end-point comes with its dedicated TDL. To be also noticed that the tunable delays enable post-silicon compensation on the short-paths (out of the scope of this work).

The EBS strategy can be seen as a "weak" hold-fixing optimization procedure where the set-up constraints are not taken into account. Indeed, a TDL does not delay short-paths only, actually, it evenly affects all the paths in its fan-in cone. The longest paths may thereby suffer early sampling, which is why we called this technique *Early Bird Sampling*. The relaxation of the timing constraints is the key for a lightweight implementation of the error-detection mechanism.

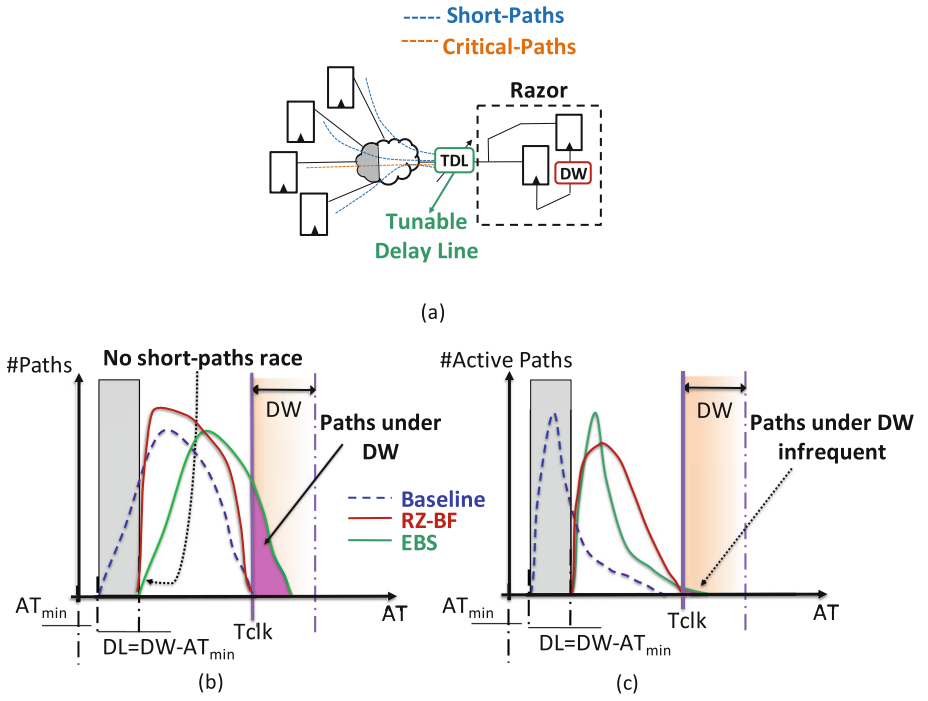


Fig. 2. Early Bird Sampling at a timing critical end-point: circuit implementation (a); static (b) and dynamic (c) timing paths analysis. Plots are illustrative and do not refer to a specific case, rather, they show typical distributions observed on generic circuits. (Color figure online)

To better understand the working principle of EBS, Fig. 2b provides a comparison among the static path distributions at a critical end-point for three different circuits implementations: (*i*) a generic circuit after synthesis, the *baseline* (dashed line), (*ii*) the circuit after standard hold-fixing optimization, *RZ-BF* (red-line) (*iii*) *EBS* (green line). Hold-fixing reshapes the path distributions guaranteeing that all paths are beyond the detection window, namely, outside the gray area in figure, while maintaining the longest path delay unchanged. By contrast, the effect of EBS is to shift the whole timing distribution, hence, some paths move beyond $Tclk$ (purple area).

In principle, this issue may be seen as a potential impediment. However, a more accurate analysis reveals that the problem is less relevant from a practical viewpoint. EBS exploits the fact that for real-life workloads the activation probability of long paths is usually pretty low. This feature, shown by the majority of digital circuits, suggests that latent faults on long paths are rarely excited. Experimental results give evidence of such empirical rule of thumb, which can be inferred by probing the arrival time of timing end-points during workload execution, i.e., through a dynamic timing analysis. Figure 2c plots the dynamic

path distribution for a typical workload run on three different implementations of the circuits: baseline, RZ-BF, EBS. As a matter of fact, the number of violating paths is much lower that those estimated using a worst-case static timing analysis (purple area in Fig. 2b).

The most interesting aspect, is that EBS does not alter the shape of the distribution (both static and dynamic); referring to the plots in Fig. 2, the green line is a copy of the dashed line, just shifted on the right. This allows to preserve the intrinsic characteristics of the original circuit, thus enabling a more efficient voltage scaling. The same is not for RZ-BF, where path compression resulting from hold-fixing optimization substantially increases the number of "quasi-critical" paths (i.e., paths close to $Tclk$) as shown in Fig. 2c. As a side effect, even small voltage variations would bring a large number of paths beyond $Tclk$, therefore triggering more timing errors. As a result, power-management techniques, and VOS in particular, would have less margins to operate.

4 Implementation Details

4.1 Tunable Delay Line (TDL)

Different implementations of TDLs have been proposed in literature; as the modeling of a TDL is out of the scope of this work, we opted for the solution presented in [32]. It consists of a pair of inverters with a voltage-controlled variable load between them; the load is a transmission gate whose ON-resistance is controlled by V_{delay}, as shown in Fig. 3. Such solution allows to cover a wide range of delays with a limited area overhead. The main drawback is that an extra power grid is needed for the distribution of V_{delay}. An alternative solution is to use tunable buffers adopted for on-line clock-skew compensation [33].

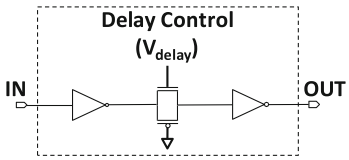


Fig. 3. Tunable Delay Line (TDL) implementation [32].

4.2 Error Detection and Correction

The EBS detection and correction mechanism is implemented using standard Razor-FFs [23] augmented with a logic masking circuitry [14], Fig. 4. Hereafter, we refer to this architecture as *Razor-Logic-Masking* (Razor-LM). A polarity change at the input of the main flip-flop after the rise edge of the clock implies some long-path is violating the timing constraint, i.e., a timing error. This event is flagged through the XOR gate that runs a parity check between the signals at pins D_{FF} and Q_{FF}. The error flag is sampled in a shadow latch triggered on the fall edge of the clock.

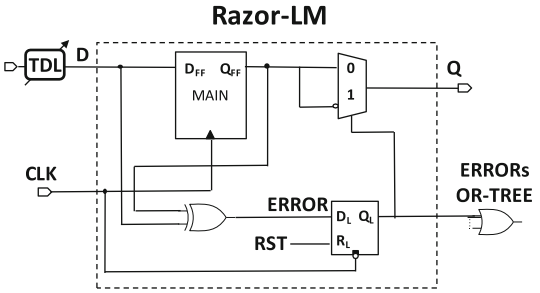


Fig. 4. Error detection and logic masking circuitry in EBS.

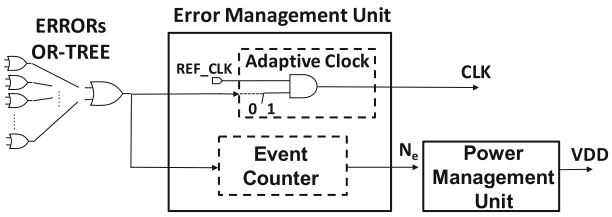


Fig. 5. Error Management Unit in EBS.

This smart solution allows large detection windows (i.e., $DW = 50\%Tclk$) w/o any modification of the clock distribution network. Once detected, the error is locally corrected through logic masking, that is, a MUX switches the output with the complement of the wrong signal stored in the main FF.

In order to let the corrected value propagate toward the fanout logic, the whole circuit has to be stopped for at least one clock cycle. Such an error-driven clock-gating is managed by the *Error Management Unit* (EMU), Fig. 5, that uses a superset of the error flags (OR among all the Razor-LM in the circuit) as clock enable. The EMU is also in charge of collecting the error statistics, i.e., the number of error occurrences N_e within a predefined monitoring period of N clock cycles. The *Power Management Unit* (PMU) uses this feedback to implement the dynamic voltage scaling.

4.3 Design Flow

The EBS design flow encompasses three different stages we integrated into a commercial design platform (the *Synopsys*® Galaxy) using wrappers written in `TCL`:

1. **Logic Synthesis:** a classical timing-driven, low-power logic synthesis run using 28 nm industrial technology libraries characterized at the nominal Vdd = 1.10 V.

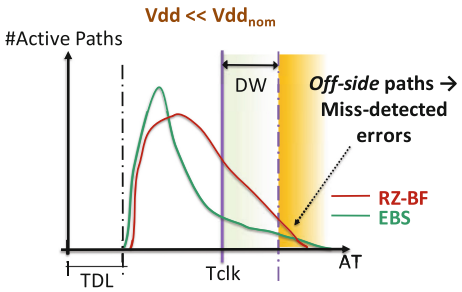


Fig. 6. Miss-detected errors representation.

2. **Identification of critical end-points:** after the clock-tree synthesis, the end-points whose worst-case arrival time at minimum voltage Vdd = 0.60 V (lower bound of the voltage scaling range) miss the clock-period $Tclk$ are labeled as "critical".
3. **Razor-LM re-placement and TDL insertion:** for each critical end-point, the standard FF is replaced with a Razor-LM and the TDL properly inserted; the error OR-tree is also synthesized.

5 Data-Driven Voltage Over-Scaling with EBS

Data-Driven Voltage Over-scaling (DD-VOS) belongs to the class of adaptive voltage scaling [11] techniques. It implements a context-driven voltage lowering, that is, voltage gets regulated by the occurrence of timing errors on the actual *sensitized* critical paths, i.e., those activated by the actual input pattern.

As already discussed, this may lead some of the longest paths beyond the clock period. Those which fall within the detection window (DW) are detected and eventually corrected; that's the basic principle of EBS. However, there might be specific sequences of input patterns that push the supply voltage so down that some of the longest paths could even exceed the DW; such *off-side* paths represent the main source of error *miss-prediction*. The latter case is graphically depicted in Fig. 6. Paths in *off-side* run out of control, and their activation is the main source of error propagation. Here's why DD-VOS is particularly suited for error-resilient applications.

It is worth to emphasize that miss-detections mainly raise depending on the voltage scaling policy adopted. We therefore provide a parametric analysis among different DD-VOS parameters and different management policies (the latter being described in the next subsection).

5.1 DD-VOS Policies

The main feedback provided by the error management unit (EMU) is the number of errors N_e within a predefined number of clock-cycles N, the monitoring period

(please refer to Sect. 4). The power management unit (PMU) makes use of such error-rate ER in order to implement some voltage scaling policy. More specifically, the ER is compared against a given ER_{th} (or multiple error-thresholds) in order to trigger the voltage scaling. In this work we implemented three different policies as follows.

1. Single-threshold (STh): as shown in Fig. 7a, given ER_{Th} as a user-defined error threshold, the policy works as follow:

- as soon as N_e gets larger than ER_{Th}, the supply voltage is increased w/o waiting for the end of monitoring period.
- if $N_e \leq ER_{Th}$ at the end of the monitoring period, i.e., after N cycles, the supply voltage is reduced for power minimization.

To notice that STh enables the control over the minimum Operation per Clock-cycle (OPC), a measure of performance overhead due to error correction; indeed, ER_{Th} represents the maximum OPC loss.

2. Double-threshold (DTh): conceived to be more conservative, the DTh policy exploits a "neutral" region defined by two thresholds $ER_{Th_{min}}$ and $ER_{Th_{max}}$, Fig. 7b; within this region, the supply voltage is kept untouched. This avoids excessive Vdd ripples thus making the voltage scaling smoother. The policy works as follows:

- as soon as $N_e \geq ER_{Th_{max}}$, Vdd is scaled up w/o waiting for the end of monitoring period;
- if $N_e \leq ER_{Th_{min}}$ at the end of the monitoring period, Vdd is scaled down for power minimization;
- if $ER_{Th_{min}} < N_e < ER_{Th_{max}}$ at the end of the monitoring period, Vdd is kept unchanged in order to avoid excessive Vdd ripple.

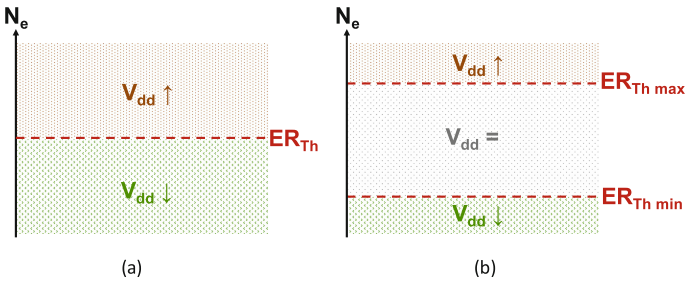


Fig. 7. STh (a) vs. DTh (b) Vdd scaling policies.

3. Threshold-exceeding Saturation Counter (SC): this policy is more elaborated as it takes into account how the supply voltage evolves over time. The working mechanism, depicted in Fig. 8 as a Mealy Finite State Machine (FSM),

makes use of a 4-bit saturation counter to decide whether the voltage has to be scaled up/down. It works as follows:

- the policy starts reducing Vdd (*Safe* state);
- if N_e exceeds the ER_{Th} a warning signal is raised ($E=1$) at the end of the monitoring period and the FSM state evolves to *Saturation Count*. Vdd is not increased;
- the Vdd is scaled up if the number of consecutive warning signal c is equal to c_{max} (2^5-1). In this case, the FSM moves to the *Unsafe* state;
- If the current state is *Unsafe* and no warning signal is raised ($E=0$), the FSM evolves in *Safe* state and Vdd is scaled down;
- anytime FSM reaches the *Saturation Count* state, the warning count c is set to zero.

To notice that the SC policy has been thought to be more aggressive than STh; indeed it allows to increase the time spent at lower Vdd, even when the N_e exceeds ER_{Th}. This enables larger energy savings at the cost of some performance and quality-of-results loss.

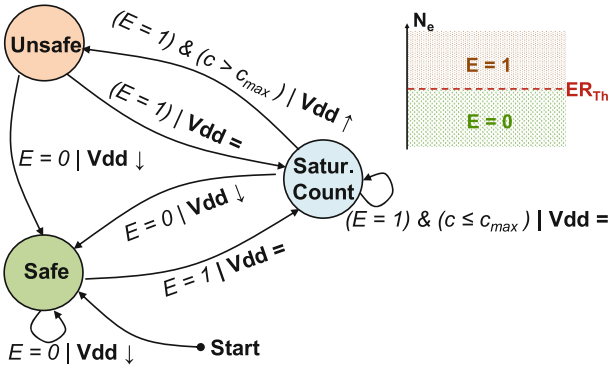


Fig. 8. Saturation counter Vdd scaling policy.

6 Experimental Framework

6.1 Benchmarks and Testbenches

The proposed EBS technique has been tested on a set of open source benchmarks over which we applied a DD-VOS scheme. The five circuits under analysis are:

- *Adder:* 32 × 32-bit + Carry-In Adder; $\mathrm{f}_{clk} = 750$ MHz.
- *Multiplier:* 32 × 32-bit Multiplier; $\mathrm{f}_{clk} = 500$ MHz.
- *MAC:* 16 × 16-bit Multiply Accum. Unit; $\mathrm{f}_{clk} = 650$ MHz.
- *FIR Filter:* Pipelined 16th-order low-pass FIR filter in direct form (12-bit in, 24-bit out); $\mathrm{f}_{clk} = 650$ MHz.

– *IIR Filter:* Pipelined 8th-order low-pass IIR filter in direct form I, modeled after a Bessel analog filter (16-bit in, 32-bit out); $f_{clk} = 650$ MHz.

For each benchmark we designed both the EBS and the RZ-BF versions. The difference between them is the method adopted to solve the short-path races, i.e., TDLs for EBS and standard post-synthesis hold-fixing for RZ-BF; in both cases the number of monitored end points is the same. The hold-fixing procedure implemented for the RZ-BF circuits uses multi-V_{th} clock buffers that minimize the area overheads.

The DD-VOS is emulated using a in-house tool (Fig. 9) which runs functional simulations (*Mentor QuestaSim*) with back-annotated `sdf` delay information. Propagation delays are extracted using a Static Timing Analysis engine (*Synopsys PrimeTime*) loaded with technology libraries characterized at different supply voltages; for those supply voltages not available in the library set we used derating factors embedded into the STA. The power dissipation is calculated using probabilistic models (*Synopsys PrimePower*) with back-annotated signal statistics from `saif` format files. The energy consumption is estimated considering the supply voltage profiles collected from simulations.

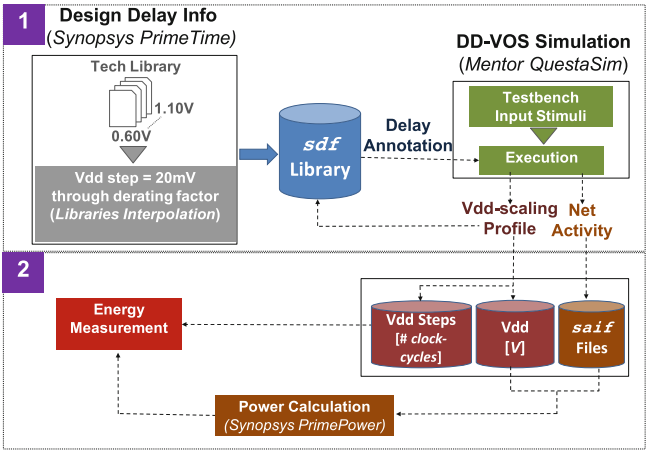


Fig. 9. In-house DD-VOS emulation tool flow diagram.

The emulated workload consists of realistic input stimuli made up of 5×10^6 patterns customized for each benchmark. For arithmetic circuits (*Adder*, *Multiplier* and *MAC*) we organized the patterns as sequence of Gaussian distributions each of them having a variable mean; for *Adder* and *Multiplier*: $\mu_1 = 2^8$, $\mu_2 = 2^{16}$ $\mu_3 = 2^{28}$ with standard deviation $\sigma = 2^8$; for *MAC*: $\mu_1 = 2^4$, $\mu_2 = 2^8$ $\mu_3 = 2^{12}$ with standard deviation $\sigma = 2^4$. For *FIR* and *IIR filters*, stimulus consists of a set of baseband audio samples.

6.2 Quality Metrics

1. *Average Vdd*: average of the Vdd measured over the testbench trace.
2. *Energy per Operation* (EPO): ratio between energy consumed and number of operations.
3. *Operation per Clock Cycle* (OPC): ratio between the number of operation run and total number of clock cycles.
4. *Miss-detected Errors* (MDE): the count of logic errors due to miss-detected timing faults occurred during simulation, measured in *ppm* (parts per million).
5. *Normalized Root Mean Squared Error* (NRMSE):

$$NRMSE = \sqrt{\frac{\sum_{i=0}^{n}(y[i] - y_o[i])^2}{n}} \cdot \frac{1}{y_{max} - y_{min}} \tag{2}$$

with y the value sampled at the output of the circuit, y_o the right output value, n is the total number of operations; y_{max} and y_{min} are the max and the min value of y_o, they define output dynamic. $NRMSE$ quantifies the QoR.

To be noticed that our simulations do not consider process variations as they do not affect the functionality of the proposed technique.

7 Results

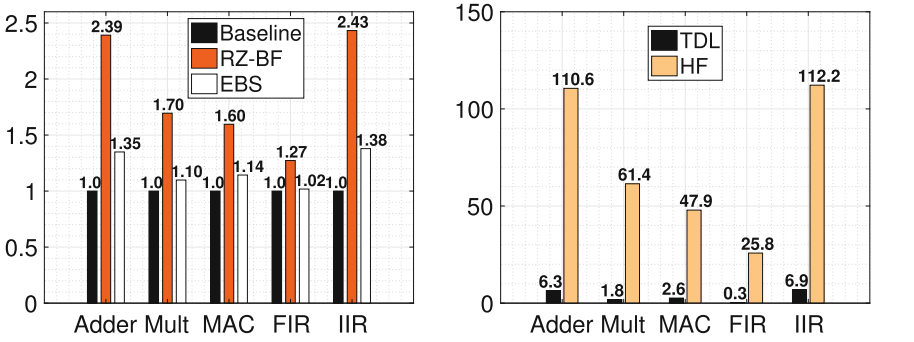
7.1 Area Overhead

Table 1 collects the statistics of the five benchmarks; column **#FFs** reports the total number of flip-flops (FFs), while column **#Critical-FFs** the percentage of FFs replaced with timing monitors, the Razor-LM.

Table 1. Benchmarks designed for EBS.

Benchmark	Area [μm^2]	#FFs	#Critical-FFs	DW [ps]	TDL [ps]
Adder	339.45	98	22.4%	665	616
Mult	2954.01	128	42.2%	1000	898
MAC	1241.12	72	45.8%	750	656
FIR	1946.32	228	8.3%	750	634
IIR	3296.80	296	78.4%	750	692

The DW is set to 50%$\cdot Tclk$, while TDL is sized according to Eq. 1. The analysis reported in [32] ensures that the circuit adopted to implement the delay lines (Sect. 4.1) allows to achieve the values reported in the Table 1.



(a) Total Area Overhead (b) Hold-fixing vs. TDL area overhead (%)

Fig. 10. Area overhead comparison.

The bar-charts in Fig. 10a and b provide a more detailed area comparison between EBS and RZ-BF; both the implementations show the same number of timing monitors (Table 1). Figure 10a shows the overall area overhead normalized w.r.t. the baseline circuit (i.e., w/o any EDC scheme). EBS is by far a more compact architecture; average area overhead is 19.8% against 87.8% of RZ-BF. That's due to the efficiency of the proposed dynamic short path padding. As shown in Fig. 10b, TDLs requires much less area (3.6% on average) than buffers insertion using hold-fixing procedures (71.6% on average). For instance, the *IIR*, which shows a large number of short-paths in the feedback network, area penalty of RZ-BF is 112.2%, while it drastically reduces to 6.9% with EBS.

7.2 DD-VOS Improvement with EBS

In order to quantify the improvements brought by EBS, Tables 2 and 3 summarize the results achieved during DD-VOS emulation on the five benchmarks under analysis. The two tables report a collection of the quality metrics presented in Sect. 6.2. Collected results refer to the *single-threshold* policy (STh) described in Sect. 5 assuming (*i*) a monitoring period $N = 10^3$ clock cycles, (*ii*) a Vdd step 20 mV, (*iii*) two different values for the error-rate ER_{Th}: 2%, i.e., 20 errors in 10^3 cycles - Table 2), and 5%, i.e., 50 errors in 10^3 cycles - Table 3.

A large ER_{Th} accelerates the voltage scaling, hence, it may induce some performance penalty (due to more errors to be corrected) and some QoR degradation (due to a possible increase of miss-detected errors).

Except for the *Mult* benchmark, which we discuss later as a special testcase, the results clearly show EBS outperforms RZ-BF. The savings achieved with EBS are quantified by (*i*) the average Vdd recorded during testbench simulations (column Vdd_{avg}), and (*ii*) the energy-per-operation savings w.r.t. the baseline circuit (column $\text{EPO}_{savings}$). The EBS implementation reaches lower Vdd_{avg} (and also minimum Vdd - column Vdd_{min}) for both the ER_{Th} thresholds. This translates into larger EPO savings w.r.t. RZ-BF. Best cases have been measured

Table 2. Results summary for DD-VOS set as $N = 10^3$ (clock cycles) and $ER_{Th} = 2\%$. Notes: *EPO* savings w.r.t. Baseline.

Benchmarks	EBS					
	Vdd_{min} [V]	Vdd_{avg} [V]	EPO savings [%]	OPC	MDE [ppm]	NRMSE [%]
MAC	0.74	0.87	43.6	0.98	0	0
Adder	0.60	0.83	41.9	0.98	6	0.001
IIR	0.92	0.95	30.6	0.98	0	0
FIR	0.84	0.96	28.2	0.98	0	0
Mult	1.10	1.10	−5.5	0.98	0	0
Benchmarks	RZ-BF					
	Vdd_{min} [V]	Vdd_{avg} [V]	EPO savings [%]	OPC	MDE [ppm]	NRMSE [%]
MAC	0.94	1.00	9.0	0.98	0	0.0
Adder	0.66	0.87	18.6	0.99	35	0.001
IIR	1.00	1.02	−47.7	0.97	0	0
FIR	0.86	0.98	12.8	0.98	0	0
Mult	1.00	1.04	−23.7	0.98	0	0

Table 3. Results summary for DD-VOS set as $N = 10^3$ (clock cycles) and $ER_{Th} = 5\%$. Notes: *EPO* savings w.r.t. Baseline.

Benchmarks	EBS					
	Vdd_{min} [V]	Vdd_{avg} [V]	EPO savings [%]	OPC	MDE [ppm]	NRMSE [%]
MAC	0.72	0.83	48.6	0.95	6	0.128
Adder	0.60	0.81	44.4	0.97	15	0.001
IIR	0.88	0.90	37.6	0.95	0	0
FIR	0.80	0.93	34.0	0.95	0	0
Mult	1.10	1.10	−8.5	0.95	0	0
Benchmarks	RZ-BF					
	Vdd_{min} [V]	Vdd_{avg} [V]	EPO savings [%]	OPC	MDE [ppm]	NRMSE [%]
MAC	0.94	0.99	9.6	0.95	0	0
Adder	0.66	0.86	20.7	0.97	91	0.002
IIR	0.98	1.01	−45.6	0.96	0	0
FIR	0.86	0.96	21.5	0.96	0	0
Mult	1.00	1.02	−19.5	0.96	0	0

for *MAC* (48.6% for EBS vs. 9.6% for RZ-BF at $ER_{Th} = 5\%$) and *Adder* (44.4% for EBS vs 20.7% for RZ-BF at $ER_{Th} = 5\%$). It is worth to emphasize that in the worst-case (*FIR*), *EPO* savings achieved with EBS are 2.2× larger than those obtained by RZ-BF: 28.2% vs. 12.8% for $ER_{Th} = 2\%$; 34.0% vs. 21.5% for $ER_{Th} = 5\%$.

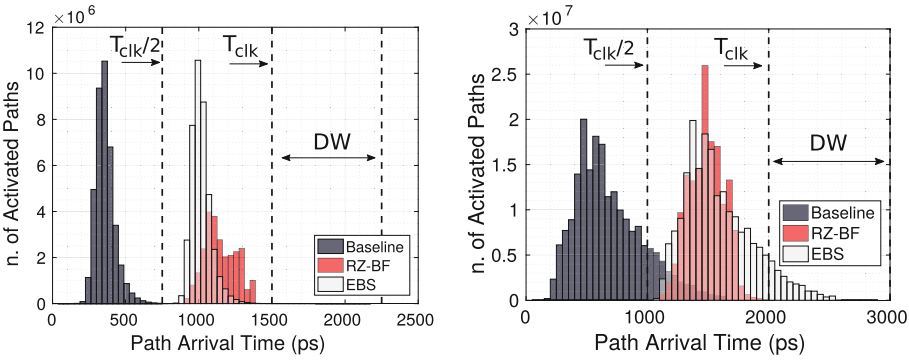
The *IIR filter* is a kind of circuit for which RZ-BF results quite inefficient; the *EPO* increases w.r.t. the baseline circuit leading to negative savings: −47.7%

at $ER_{Th} = 2\%$ and -45.6% at $ER_{Th} = 5\%$. Such huge design overhead is due to the fact that hold-fixing overwhelms the power savings of voltage scaling. By contrast, EBS still gets remarkable EPO savings: 30.6% at $ER_{Th} = 2\%$ and 37.6% at $ER_{Th} = 5\%$.

For what concerns performance degradations due to errors correction, Tables 2 and 3 clearly shows EBS guarantees a OPC close to that of the RZ-BF strategy: $OPC \geq \{0.98, 0.95\}$ for both the thresholds $ER_{Th} = \{2\%, 5\%\}$. This confirms once again TDLs insertion has marginal effect on the error-rate.

Remarkable results have been also observed in terms of reliability. Although EBS pushes Vdd to values below those achieved with RZ-BF, the number of miss-detections MDE is zero for all the benchmarks. The two exceptions are *Adder* ($MDE = 6$ ppm and $MDE = 15$ ppm, with ER_{Th} equals to 2% and 5% respectively) and *MAC* ($MDE = 6$ ppm at $ER_{Th} = 5\%$). Nonetheless only marginal QoR degradation has been observed: $NRMSE$ is a mere 0.128% at worst case. Such a low QoR degradation is achieved thanks to the internal logic topology of the circuits which, in turn, reflects into a low activation of the most critical paths.

As a counterexample, the *Mult* benchmark belongs to that class of circuits whose internal characteristics are not particularly suited for aggressive voltage over-scaling. Both EBS and RZ-BF fail, suggesting DD-VOS might not be a valuable low-power option. To better understand the reasons behind such behavior, we resort to a comparison between two benchmarks, the *Mult* (for which DD-VOS does not work) and the *MAC* (for which DD-VOS gets substantial savings). Figure 11 recalls the qualitative analysis discussed in Sect. 3. More specifically, it shows the dynamic path distribution of three different implementations: baseline, EBS and RZ-BF. The bars represent the cumulative number of timing path activations vs. their arrival time.



(a) 16×16 bit MAC (b) 32×32 bit Multilplier

Fig. 11. Dynamic path distribution analysis.

Some key comments are as follows. First. For both EBS and RZ-BF the path distribution is skewed such that none of the short-paths falls behind $T_{clk}/2$ (the

width of the detection window DW). This avoids short-path races thus ensuring the right functionality of the error detection mechanism. Second. EBS keeps the path distribution unchanged (just a right shift of the baseline distribution) avoiding the growth of those "quasi-critical" paths that, just standing behind T_{clk}, may prevent voltage lowering. Usually, RZ-BF works on the opposite direction instead, as the number of "quasi-critical" increases due to timing-constrained buffer insertion. This behavior is quite evident for *MAC* (Fig. 11a), for which the red bars (RZ-BF implementation) stand over the white ones (EBS implementation). Since a larger number of activated "quasi-critical" paths reduces the chance of Vdd lowering, EBS results to be more efficient. That's what makes EBS outperforming RZ-BF. However, there might be particular circuits for which this feature does not hold. Such circuits are those for which the basic principle under which EBS is built, namely, *the longer the path, the lower its activation*, gets weaker. That's the *Mult*. As reported in Fig. 11b, the original dynamic path distribution (baseline implementation) is pretty large, with very active paths that take the whole clock-period. This negatively affects EBS, where the TDLs push many paths into the DW; as a result, the supply voltage is stuck at high values and the EPO gets larger than the original circuit due to error corrections: 1.06× and 1.09× for $ER_{Th} = 2\%$ and $ER_{Th} = 5\%$ respectively. Also RZ-BF suffers from the same problem, as the number of active paths across T_{clk} is huge; EPO increases w.r.t. the baseline circuit: 1.24× and 1.20× for $ER_{Th} = 2\%$ and $ER_{Th} = 5\%$ respectively. However, the overhead of RZ-BF gets larger than that of EBS.

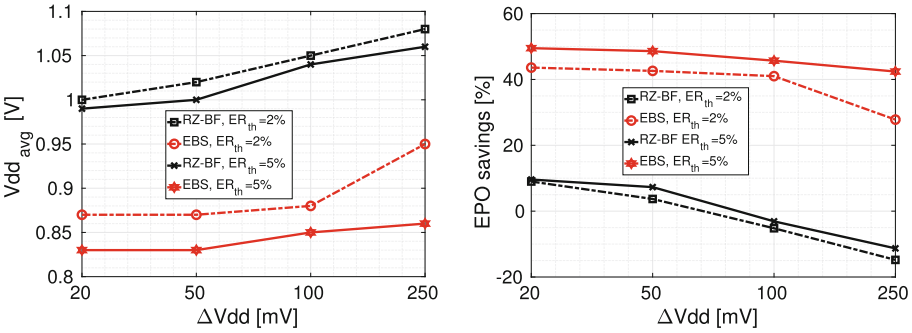
As a final comment, one should consider that circuits on which DD-VOS does not work properly, may radically change their behavior when integrated into more complex architectures. That's the case of *Mult* integrated into *MAC*.

7.3 EBS Characterization Under Different DD-VOS Implementations

The main goal of this section is to quantify the figures of merit of EBS under different DD-VOS settings, and thus, to demonstrate EBS performs well under several power management scenarios. We therefore characterize the quality metrics according to: *(i)* the Vdd step, namely, the ΔVdd used for voltage scaling; *(ii)* the monitoring period, that is, the clock cycles N used to measure the error-rate; *(iii)* the Vdd scaling policies presented in Sect. 5. For the sake of space, we just report the analysis for *MAC*. Similar results hold for the other benchmarks.

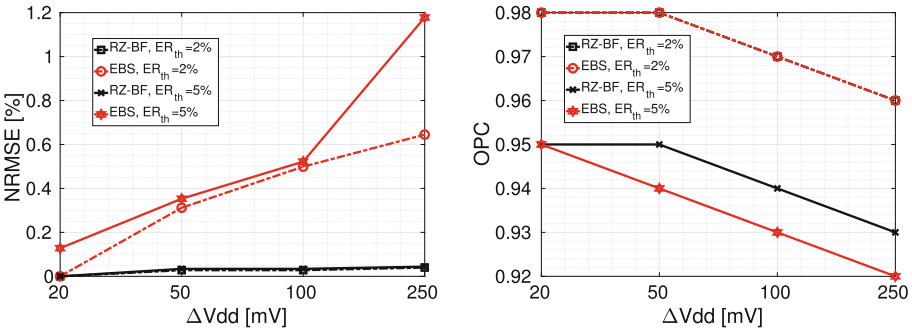
Vdd Step. The collected results refer to three different values of ΔVdd: 20 mV, 50 mV, 100 mV, 250 mV. In order to make the analysis more realistic, we also take into consideration different voltage steps may require different clock-cycles to be properly delivered; we therefore assume a latency of $\{1, 2, 5, 12\}$ clock cycles for $\{20\,\text{mV}, 50\,\text{mV}, 100\,\text{mV}, 250\,\text{mV}\}$ respectively.

Simulations are conducted on EBS and RZ-BF using the *Single Threshold* Vdd scaling policy (STh) under two different values of error-threshold, $ER_{Th} = 2\%$ and 5%.



(a) Vdd_{avg} (b) *EPO* savings (w.r.t. Baseline)

Fig. 12. Energy efficiency vs. Vdd step width (ΔVdd).



(a) *NRMSE* (%) (b) Operation/Clock-cycle

Fig. 13. QoR and performance vs. Vdd step width (ΔVdd).

EBS outperforms RZ-BF for any of the vdd-steps values under analysis. As shown in Fig. 12b, EBS brings the circuit to a lower average Vdd; along the whole ΔVdd range, the average improvement w.r.t. RZ-BF reaches 140 mV at $ER_{th} = 2\%$ and 180 mV at $ER_{th} = 5\%$. The same results hold for energy efficiency; average EPO savings are: 46.6% and 38.8% at $ER_{th} = 5\%$ and $ER_{th} = 2\%$ for EBS vs. a mere 0.6% and 0.01% for RZ-BF. More in details, Fig. 12 shows how savings drift with ΔVdd. Vdd_{avg} reaches lower values using a finer voltage resolution. For instance, considering the EBS at $ER_{Th} = 2\%$, it reduces from 0.95 V at $\Delta\text{Vdd} = 250$ mV to 0.87 mV at $\Delta\text{Vdd} = 20$ mV. As a result, the energy savings reported in Fig. 12b show substantial improvements, from 27.8% at $\Delta\text{Vdd} = 250$ mV to 43.6% at $\Delta\text{Vdd} = 20$ mV.

The voltage resolution does also impact the QoR. As shown in Fig. 13a, the lower the Vdd step, the better the QoR. Indeed, a larger ΔVdd makes harder to control the occurrence of miss-detected errors. For $ER_{Th} = 5\%$, the *NRMSE* measured for EBS reduces from 1.2% at $\Delta\text{Vdd} = 250$ mV to 0.1% at $\Delta\text{Vdd} = 20$ mV. By contrast, the *NRMSE* of the RZ-BF implementation is less

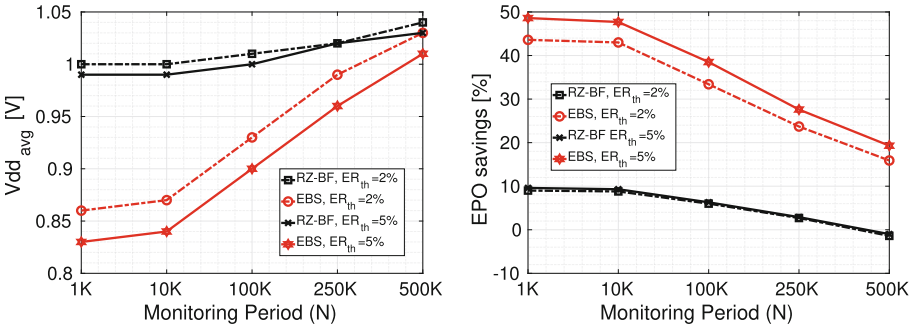
sensible (variation in the range [0.0%–0.1%]); that's mainly due to the fact that the Vdd scaling is slower than in EBS, therefore, less miss-detections do occur.

Concerning the performance, the *STh* Vdd-scaling policy is conceived as a mechanism to control the minimum *OPC* value; ideally, as introduced in Sect. 5, ER_{Th} represents the max. *OPC* loss, thus the minimum OPC equals $1 - ER_{Th}$. However, since larger Vdd steps come with larger latencies, the performance achieved by EBS and RZ-BF are substantially affected and *OPC* may drop below that ideal minimum boundary if OPC loss $> ER_{Th}$. Figure 13b shows this drawback through *OPC* vs. ΔVdd plot; both EBS and RZ-BF *OPC* losses are still kept lower than ER_{Th} only for $\Delta\text{Vdd} = 20\,\text{mV}$ (both the ER_{Th}s). In the worst case, i.e., $ER_{Th} = 5\%$, *OPC* loss raises from 5% (i.e., the ideal max. loss value) to 8% for EBS and from 5% to 7% for RZ-BF in the interval $\Delta\text{Vdd} = [20\,\text{mV} - 250\,\text{mV}]$.

Monitoring Period. The plots reported in Fig. 14 show Vdd_{avg} and *EPO* using different monitoring period N. Simulations are conducted on EBS and RZ-BF using the Single Threshold Vdd (STh) scaling policy $\Delta\text{Vdd} = 20\,\text{mV}$ and two different values of error-threshold, $ER_{Th} = 2\%$ and 5%.

EBS performs more efficiently than RZ-BF for all the operating conditions. Considering the case $ER_{Th} = 5\%$ (the best case), EBS reaches lower Vdd_{avg}, 0.91 V vs 1.02 V of RZ-BF, and larger *EPO* savings, 36.4% vs. 5.4% of RZ-BF (average values on N interval).

As a general rule, the larger the N the slower the Vdd scaling. While this trend is less evident in RZ-BF (Vdd_{avg} increases by just 40 mV), EBS amplifies the effect showing an overall spread of 180 mV (from 0.83 mV to 1.01 mV). The same consideration can be inferred for *EPO*, where savings gets smaller with N, from 48.6% ($N = 10^3$) to 19.3% ($N = 5 \cdot 10^5$).



(a) Vdd_{avg} (b) *EPO* savings (w.r.t. Baseline)

Fig. 14. Energy efficiency vs. monitoring period (N).

Finally, the analysis reported in Fig. 15 shows that a more aggressive voltage scaling strategy, i.e., smaller N, affects output quality and performance due to

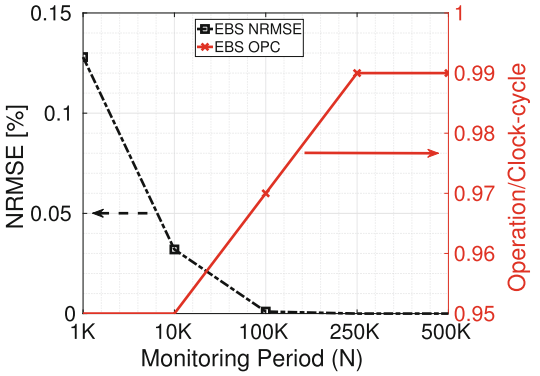


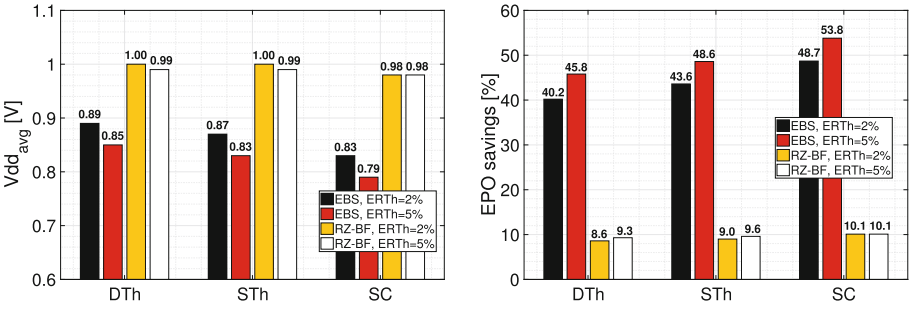
Fig. 15. EBS with $\mathrm{ER}_{th} = 5\%$: QoR and performance vs. monitoring period (N).

an increasing number of error corrections. Results for RZ-BF are omitted as the $NRMSE$ gets always zero whatever the value of N. The $NRMSE$ of EBS increases, yet, only marginally: from zero to 0.128%. Also the OPC drops: from 0.99 to 0.95 when N reduces from $= 5 \cdot 10^5$ to 10^3 for both EBS and RZ-BF. That's the cost to be payed for a more energy efficient DD-VOS.

Vdd Scaling Policies. Simulations of the three Vdd scaling policies described in Sect. 5 have been run fixing the DD-VOS parameters as follows:

- Vdd step, $\Delta\mathrm{Vdd} = 20\,\mathrm{mV}$;
- monitoring period, $N = 10^3$.
- error-threshold: $\mathrm{ER}_{Th} = \{2\%, 5\%\}$ for STh and SC, $\mathrm{ER}_{Th_{max}} = \mathrm{ER}_{Th} = \{2\%, 5\%\}$ and $\mathrm{ER}_{Th_{min}} = 0.2 \cdot \mathrm{ER}_{Th_{max}}$ for DTh.

Figure 16 plots the collected results, which show once again EBS improves the figures of merit of DD-VOS, whatever the adopted policy.



(a) Vdd_{avg} (b) EPO savings (w.r.t. Baseline)

Fig. 16. DD-VOS policies energy efficiency comparison.

Table 4. Results summary for DD-VOS policies OPC and $NRMSE$.

Benchmarks	RZ-BF				EBS			
	OPC		NRMSE		OPC		NRMSE [%]	
	ER_{Th}= 2%	ER_{Th}= 5%	ER_{Th}= 2%	ER_{Th}= 5%	ER_{Th}= 2%	ER_{Th}= 5%	ER_{Th}= 2%	ER_{Th}= 5%
Double Th	0.99	0.97	0.000	0.000	0.99	0.97	0.000	0.000
Single Th	0.98	0.95	0.000	0.000	0.98	0.95	0.000	0.128
Saturation Counter	0.87	0.86	0.000	0.000	0.95	0.90	0.158	0.185

The DTh is more conservative. In this case EBS reaches a lower Vdd_{avg} than that of RZ-BF: at $\text{ER}_{Th} = 5\%$ (best case), 0.85 V vs. 0.99 V. With a lower Vdd, also EPO savings improve: 45.8% vs 9.3%. At the opposite corner, the SC approach pushes a more aggressive Vdd scaling. EBS reaches the lowest Vdd_{avg}, hence, the largest EPO savings: 0.79 V with 53.8% energy savings, against 0.98 V and 10.1% of RZ-BF. This comes at the cost of some miss-detection. As shown in Table 4, the adoption of the SC policy induces a $NRMSE$ degradation, mainly due to miss-detected errors: 0.185% in the worst case ($\text{ER}_{Th} = 5\%$). By contrast DTh ensures zero miss-detected errors, both for EBS and RZ-BF.

For what concerns performance, DTh affects OPC only marginally: 3% loss for both EBS and RZ-BF in the worst case ($\text{ER}_{Th} = 5\%$); DTh reduces Vdd ripples thus bringing to a lower number of error corrections. On the contrary, SC heavily impacts performance with OPC loss in the order of 10% for EBS and 14% for RZ-BF (at $\text{ER}_{Th} = 5\%$).

8 Conclusions

Early Bird Sampling (EBS) is a Razor variant strategy that applies to generic low-power sequential designs. The EBS allows to solve the problem of short-path races bypassing tedious hold-time fixing design stages, and enables aggressive Data-Driven Voltage Over-Scaling (DD-VOS), suited for ultra-low power error-resilient applications. Simulation runs on a representative set of circuits under realistic workloads using different voltage scaling policies provide a fair comparison with a standard Razor strategy. Collected results show EBS reduces area overheads (3.6% against 71.6% for Razor) and improves the voltage scaling thereby achieving higher energy efficiency (savings w.r.t. Razor range from 36.2% to 40.2%).

References

1. Ernst, D., Kim, N., et al.: Razor: a low-power pipeline based on circuit-level timing speculation. In: 36th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO-36, Proceedings, pp. 7–18. IEEE (2003)
2. Rizzo, R.G., Peluso, V., Calimera, A., Zhou, J., Liu, X.: Early bird sampling: a short-paths free error detection-correction strategy for data-driven VOS. In: 2017 IEEE 25th International Conference on Very Large Scale Integration (VLSI-SoC). IEEE (2017)
3. Benini, L., Castelli, G., Macii, A., Macii, B., Scarai, R.: Battery-driven dynamic power management of portable systems. In: Proceedings 13th International Symposium on System Synthesis, pp. 25–30 (2000)
4. Alioto, M.: Ultra low power design approaches for IoT. Singapore-Hotchips (2014)
5. Bortolotti, D., Rossi, D., Bartolini, A., Benini, L.: A variation tolerant architecture for ultra low power multi-processor cluster. In: 2013 23rd International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS), pp. 32–38. IEEE (2013)
6. Peluso, V., Rizzo, R.G., Calimera, A., Macii, E., Alioto, M.: Beyond ideal DVFS through ultra-fine grain vdd-hopping. In: Hollstein, T., Raik, J., Kostin, S., Tšertov, A., O'Connor, I., Reis, R. (eds.) VLSI-SoC 2016. IAICT, vol. 508, pp. 152–172. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67104-8_8
7. Benini, L., De Micheli, G., Macii, E., Poncino, M., Scarsi, R.: Symbolic synthesis of clock-gating logic for power optimization of control-oriented synchronous networks. In: Proceedings of the 1997 European Conference on Design and Test, EDTC 1997, p. 514. IEEE Computer Society, Washington, DC (1997)
8. Babighian, P., Benini, L., Macii, A., Macii, E.: Post-layout leakage power minimization based on distributed sleep transistor insertion. In: Proceedings of the 2004 International Symposium on Low Power Electronics and Design, ISLPED 2004, pp. 138–143. ACM (2004)
9. Calimera, A., Bahar, R.I., Macii, E., Poncino, M.: Temperature-insensitive dual-Vth synthesis for nanometer CMOS technologies under inverse temperature dependence. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **18**(11), 1608–1620 (2010)
10. Calimera, A., et al.: Design of a family of sleep transistor cells for a clustered power-gating flow in 65 nm technology. In: Proceedings of the 17th ACM Great Lakes symposium on VLSI, pp. 501–504. ACM (2007)
11. Krause, P.K., et al.: Adaptive voltage over-scaling for resilient applications. In: 2011 Design, Automation Test in Europe, pp. 1–6, March 2011
12. Das, S., et al.: RazorII: in situ error detection and correction for PVT and SER tolerance. IEEE J. Solid-State Circ. **44**(1), 32–48 (2009)
13. Kim, S., et al.: 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 Circ. **50**(6), 1478–1490 (2015)
14. Valadimas, S., et al.: Timing error tolerance in nanometer ICs. In: 2010 IEEE 16th International On-Line Testing Symposium (IOLTS), pp. 283–288. IEEE (2010)
15. Bowman, K.A., et al.: Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. IEEE J. Solid-State Circ. **44**(1), 49–63 (2009)
16. Bowman, K., et al.: A 45 nm resilient microprocessor core for dynamic variation tolerance. IEEE J. Solid-State Circ. **46**(1), 194–208 (2011)
17. Kwon, I., et al.: Razor-lite: a light-weight register for error detection by observing virtual supply rails. IEEE J. Solid-State Circ. **49**(9), 2054–2066 (2014)

18. Das, S., et al.: A 1 GHz hardware loop-accelerator with razor-based dynamic adaptation for energy-efficient operation. IEEE Trans. Circ. Syst. I: Regul. Pap. **61**(8), 2290–2298 (2014)
19. Yang, Y.-M., et al.: PushPull: short-path padding for timing error resilient circuits. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **33**(4), 558–570 (2014)
20. Ghosh, S., Bhunia, S., Roy, K.: CRISTA: a new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **26**(11), 1947–1956 (2007)
21. Kahng, A.B., et al.: Slack redistribution for graceful degradation under voltage overscaling. In: Proceedings of the 2010 Asia and South Pacific Design Automation Conference, pp. 825–831. IEEE Press (2010)
22. Karakonstantis, G., Roy, K.: Voltage over-scaling: a cross-layer design perspective for energy efficient systems. In: 2011 20th European Conference on Circuit Theory and Design (ECCTD), pp. 548–551. IEEE (2011)
23. Ernst, D., et al.: Razor: circuit-level correction of timing errors for low-power operation. IEEE Micro **24**(6), 10–20 (2004)
24. Ramasubramanian, S.G., Venkataramani, S., Parandhaman, A., Raghunathan, A.: Relax-and-retime: a methodology for energy-efficient recovery based design. In: Proceedings of the 50th Annual Design Automation Conference, p. 111. ACM (2013)
25. Wan, L., Chen, D.: DynaTune: circuit-level optimization for timing speculation considering dynamic path behavior. In: Proceedings of the 2009 International Conference on Computer-Aided Design, pp. 172–179. ACM (2009)
26. Greskamp, B., et al.: Blueshift: designing processors for timing speculation from the ground up. In: IEEE 15th International Symposium on High Performance Computer Architecture, HPCA 2009, pp. 213–224. IEEE (2009)
27. Wan, L., Chen, D.: CCP: common case promotion for improved timing error resilience with energy efficiency. In: Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design, pp. 135–140. ACM (2012)
28. Mohapatra, D., Karakonstantis, G., Roy, K.: Low-power process-variation tolerant arithmetic units using input-based elastic clocking. In: Proceedings of the 2007 International Symposium on Low Power Electronics and Design, pp. 74–79. ACM (2007)
29. Carmona, J., Cortadella, J., Kishinevsky, M., Taubin, A.: Elastic circuits. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **28**(10), 1437–1455 (2009)
30. Shim, B., Sridhara, S.R., Shanbhag, N.R.: Reliable low-power digital signal processing via reduced precision redundancy. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **12**(5), 497–510 (2004)
31. Pagliari, D.J., Calimera, A., Macii, E., Poncino, M.: An automated design flow for approximate circuits based on reduced precision redundancy. In: 2015 33rd IEEE International Conference on Computer Design (ICCD), pp. 86–93. IEEE (2015)
32. Zhou, J., et al.: HEPP: a new in-situ timing-error prediction and prevention technique for variation-tolerant ultra-low-voltage designs. In: 2013 IEEE Asian Solid-State Circuits Conference (A-SSCC), pp. 129–132. IEEE (2013)
33. Chakraborty, A., et al.: Dynamic thermal clock skew compensation using tunable delay buffers. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **16**(6), 639–649 (2008)