Design and Evaluation of Confidence-Driven Error-Resilient Systems

Chia-Hsiang Chen, *Student Member, IEEE*, David Blaauw, *Fellow, IEEE*, Dennis Sylvester, *Fellow, IEEE*, and Zhengya Zhang, *Member, IEEE*

***Abstract*— Deeply scaled CMOS circuits are increasingly susceptible to transient faults and soft errors; emerging post-CMOS devices can be more vulnerable, sometimes exhibiting erratic errors of arbitrary duration. Applying timing and supply voltage margin is wasteful and becoming ineffective, and conventional checking and sparing techniques provide only a limited error coverage against widely varying errors. We propose a confidence-driven computing (CDC) model for an adaptive protection against nondeterministic errors. The CDC model employs fine-grained temporal redundancy and confidence checking for a faster adaptation and tunable reliability. The CDC model can be extended to deeply scaled CMOS circuits that are mainly affected by transient faults and soft errors, where an early checking (EC) technique can be used to perform independent error checking for more flexibility and better performance. To evaluate the CDC model, we apply a sample-based field-programmable gate array emulation along with real-time error injection. The CDC model is shown to adapt to fluctuating error rates and enhance the system reliability by effectively trading off performance. To evaluate the EC technique at a finer time scale, we create a new event-based simulation to capture path delay distribution, error model, and their interactions. The EC technique improves the system reliability by more than four orders of magnitude when errors are of short duration. Both the CDC model and the EC technique are synthesized in a 45-nm CMOS technology for cost estimates: 1) the area overhead is as low as 12% and 2) energy overhead can be limited to 19%.**

***Index Terms*— Error detection, error simulation, field-programmable gate array (FPGA) emulation, reliability, resilient design.**

I. INTRODUCTION

THE scaling of CMOS devices continues to improve the performance of digital integrated circuits but with significant degradation of device reliability [1]–[3]. The reduced critical charge with each generation of new devices makes them more susceptible to internal and external noise sources, admitting transient faults, and soft errors. Low power designs that rely on reduced supply voltage have increased variations and sensitivity, further exacerbating the reliability problem [4].

Transient faults and soft errors often last for a short duration. The impact on the system varies depending on when and where the error occurs, known as timing masking and logic masking, respectively. Sometimes these errors do not harm to a system [5], while at other times they can propagate and accumulate, causing system failures. Conservative designs with margining or error detection and correction, such as Razor [6] and built-in soft error resilience (BISER) [7], can be applied to protect systems against transient faults and soft errors. Continued technology scaling, however, poses additional challenges in resilient system designs [8], as witnessed in the increasing spread of datapath delay distribution and error delay distribution [9]. The addition of the two distributions, complicated by the random arrivals of errors, requires a longer and tunable error detection window, rendering existing approaches wasteful and less effective.

In parallel with CMOS device scaling, a variety of nanodevices, such as carbon nanotube, graphene, spin, nanoelectromechanical relay, and memristor, have been proposed to sustain Moore's law of scaling for years to come [10], [11]. Although these post-CMOS devices boast potentially much higher integration density and substantially lower energy consumption compared with CMOS, some of them exhibit nondeterministic behavior. For example, memristor switching is a stochastic process that depends on the probabilistic filament formation [12], [13]. Such stochastic devices yield unpredictable operations that are manifested as erratic errors of arbitrary duration. Containing nondeterministic errors of unpredictable duration is even more challenging than transient faults and soft errors alone that are seen in conventional CMOS devices. Common approaches, including spatial duplication and checkpoint and rollback in software are very costly [14], providing only a limited coverage against widely varying errors.

In this paper, we propose a confidence-driven computing (CDC) model [29] for protection against nondeterministic errors over a wide range of rate and duration. The key concept of the proposed computing model is to employ fine-grained temporal redundancy with tunable threshold for a faster adaptation and an adjustable reliability. The CDC model is suitable for designs using nondeterministic post-CMOS devices. It allows systems to adapt to large runtime variations and reduces excessive design margins for an efficient computing system.

The CDC model can be extended to deeply scaled CMOS devices that are mainly affected by soft errors and noise-induced transient faults that often last for a short duration [15]. To be competitive with existing approaches, we propose an extension of the CDC model called early checking (EC) for added flexibility and better efficiency. Using the EC technique,

Manuscript received November 19, 2012; revised June 10, 2013; accepted July 25, 2013. Date of publication August 23, 2013; date of current version July 22, 2014. This work was supported by the Intel Corporation.

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: uchchen@umich.edu; blaauw@umich.edu; dmcs@umich.edu; zhengya@eecs.umich.edu).

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/TVLSI.2013.2277351

1063-8210 © 2013 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

error checking is performed independently of the main clock. The independent checking allows the confidence to be accumulated quickly because the vast majority of the datapaths have significant delay slacks [16] to permit an early start on checking.

Another contribution of this paper is two new field-programmable gate array (FPGA)-based methods for the design and evaluation of resilient systems, a sample-based emulation method, and an event-based simulation method. The sample-based emulation method incorporates error injection in realtime to accelerate error simulation by up to six orders of magnitude compared with software solutions. The method is faster than existing FPGA-based error simulations that rely on prestored error vectors or supplied through scan chains. The sample-based emulation method incorporates synchronous error injection and is suitable for systems based on approaches, including CDC and N-modular redundancy (NMR).

To accurately account for transient faults and soft errors and timing masking effects that are important in systems protected by EC, as well as other well-known approaches, such as Razor and BISER, we propose an event-based simulation method that considers the path delay distribution and error occurrence at a finer time scale. The event-based simulator keeps track of complex runtime scenarios using events, e.g., path completion and error occurrence, for the quantitative evaluation of a resilient design. The fast sample-based emulation and the versatile event-based simulation improve upon the previous works in error simulation [17], [18].

The proposed CDC model is evaluated using the sample-based emulation method and EC technique using the event-based simulation method. The results demonstrate several orders of magnitude improvement in system reliability with a moderate to negligible throughput penalty. Synthesized designs using a 45-nm CMOS technology show that the area and energy overhead for CDC and EC are as low as 10% to 20%.

II. Related Work

Error tolerance and error correction are the primary solutions to designing resilient systems. Error tolerance can be provided by the algorithm itself. For instance, digital signal processing algorithms such as image and video processing naturally accommodate imprecision in the computation [19]. Low-power circuits have been designed to take advantage of the algorithmic noise tolerance (ANT) [20], where errors incurred because of supply voltage over scaling can be tolerated by the algorithm. Similarly, the scalable stochastic processor [21] applies error-scaling friendly circuits to make full use of algorithmic error tolerance. Error tolerance can also be provided at the architecture level, as done in error resilient system architecture (ERSA) [22] that employs multiple cores of lower reliability to execute probabilistic applications. These recent researches improve the robustness, and thus the supply voltage and clock frequency margins can be reduced to lower power consumption and improve performance. Error tolerance techniques are, however, often limited in their applicability. ANT, stochastic processor, and ERSA all rely on the

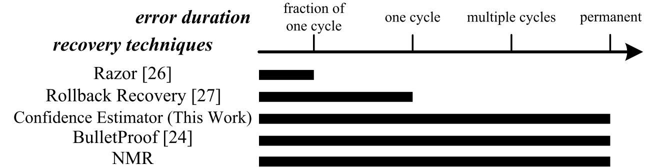

Fig. 1. Error detection and recovery techniques and the applicable error duration.

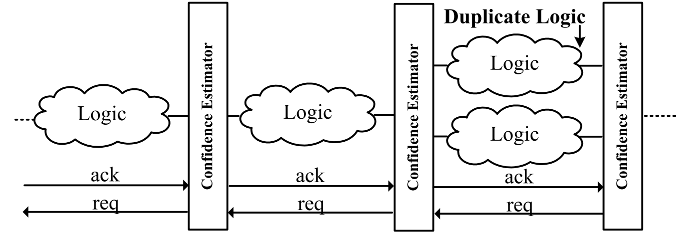

Fig. 2. Proposed CDC model incorporating confidence estimators.

characteristics of the target algorithms and they need to be tailored to each.

Error detection and correction is general purpose and it is usually aided by spatial duplication or temporal redundancy. For example, feed-forward recovery uses spatial duplication to detect and correct errors with no interruption to the computation. The approach is commonly known as NMR. Its cost, however, increases to support a wide range of device error rates [23]. The BulletProof design [24] incorporates adaptive sparing using routers in a defect-tolerant architecture, but the area and energy overhead are still significant.

In Fig. 1, we classify different error recovery techniques based on their target error duration from short transient to permanent. NMR is capable of correcting both permanent and transient errors. If an error is known to last for a short duration, a more area- and energy-efficient alternative is through temporal redundancy [25] that can be implemented in a checkpoint and rollback scheme. The Razor technique [26] performs rollback at the circuit level by a parallel shadow latch to detect and correct errors. Razor only incurs a small performance overhead, but the error detection window is short and fixed, limiting its effectiveness for errors that last for a larger fraction of a cycle. An alternative circuit-level rollback and recovery technique [27] requires a duplicate datapath for error detection and rollback buffers for correction. The rollback recovery technique has a smaller area overhead compared with NMR, but the protection level cannot be adjusted either.

The CDC model proposed in this paper can be applied with temporal redundancy to extend the error detection window to multiple clock cycles. It also allows temporal redundancy to be applied in conjunction with spatial redundancy to improve throughput and latency. Its key feature is that it enforces a confidence threshold to be met by looking for agreements, either through repeated computation over the same datapath (temporal redundancy) or duplicate datapaths (spatial redundancy), as shown in Fig. 2. The confidence threshold can be adjusted based on the device error rate and the application requirement. For example, when the application-required reliability is high and the device error rate is high, the confidence

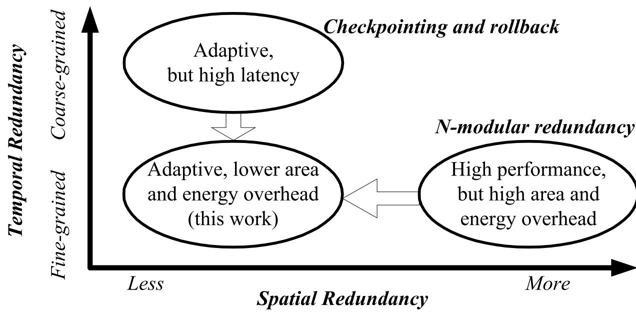

Fig. 3. Combination of spatial and temporal redundancy for an efficient error-resilient computing system.

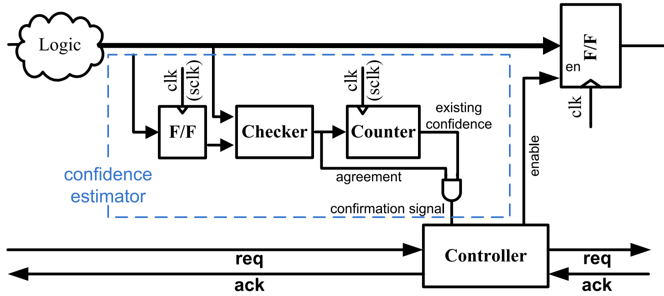

Fig. 4. Block diagram of confidence estimator.

threshold is raised to allow more repeated computations. The combination of spatial and temporal redundancy produces an adaptive resilient computing shown in Fig. 3.

III. Confidence-Driven Computing

In the CDC model, a datapath is partitioned into segments where the probability of error of each segment is bounded. A confidence estimator is placed at the end of each segment to harden the output for forward propagation to the next stage. A confidence estimator consists of four components: 1) a flip-flop; 2) a checker; 3) a counter; and 4) a small controller as shown in Fig. 4. The confidence estimator samples the output of the datapath in every clock cycle and compares it with the previous sample stored in the flip-flop (through temporal redundancy). The checker performs the comparison and looks for either an agreement or a disagreement. The counter keeps track of the confidence level: the confidence level is raised upon an agreement and reset upon a disagreement. The controller ensures that the confidence level reaches a required threshold before allowing the output to be propagated through the main flip-flop by enabling the clock.

An n-bit counter is capable of tracking 2^n distinct confidence levels. The lowest confidence level indicates bypass. The controller generates handshaking signals to synchronize with its neighboring stages. Upon confirming the confidence level meeting the required confidence threshold, the controller sends a request req to the following stage. Once the following stage becomes ready to accept a new input, an acknowledgment ack is sent back and the controller then enables the main flip-flop to pass the current output to the next stage. We develop two synchronization schemes: 1) lockstep synchronization and 2) speculative synchronization, and a method to

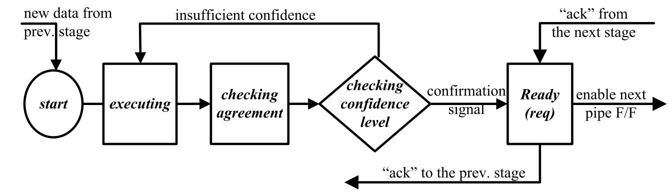

Fig. 5. Flow chart of lockstep synchronization.

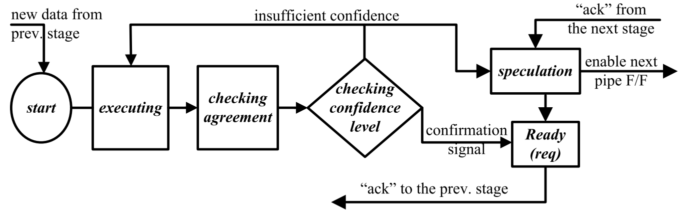

Fig. 6. Flow chart of speculative synchronization.

incorporate spatial redundancy. The operations are elaborated in the following sections.

A. Lockstep Synchronization

Lockstep synchronization guarantees the output of one stage to be hardened before it is propagated to the next stage. A slow stage, because of errors, holds back the output and leaves the neighboring stages waiting. The flow chart of lockstep synchronization is shown in Fig. 5. The output is sampled and checked for agreement. The counter accumulates the confidence level until it reaches the confidence threshold, at which point the current stage enters the ready state and sends a req to the following stage. It waits until the following stage signals an ack and then enables the output to be propagated to the following stage. After the output exits the current stage, an ack is also passed to the previous stage.

The confidence threshold is the primary knob to tune the output reliability level. A confidence threshold of 2 requires two agreements and a threshold 3 requires three agreements. Note that the confidence estimator looks for consecutive agreements in time, which is different from the majority voting scheme used in NMR. A disagreement resets the confidence level and restarts the confidence accumulation process. Through setting higher threshold in the less reliable stages, we can avoid having less reliable stages dominate the overall system error rate and therefore improve the system reliability more effectively.

B. Speculative Synchronization

Speculative synchronization allows an output to proceed to the next stage even if the confidence level has not reached the confidence threshold. Compared with the lockstep scheme, speculative execution shortens the latency and permits a higher throughput. The flow chart of speculative synchronization is shown in Fig. 6. Under this scheme, the confidence estimators act as transparent gate keepers: tentative output is passed to the next stage when the next stage is ready, while the tentative output is still being hardened by the current stage. When the

current stage finally reaches the confidence threshold, an ack is sent to the previous stage, indicating that it is ready to accept a new input. Note that to ensure data being hardened in the correct sequential order using the speculative synchronization, the confidence threshold in each stage needs to be set to no lower than the one in the previous stage.

Speculative synchronization cuts the idle cycles when one stage is complete and waiting for the previous stage to finish accumulating confidence. The scheme assigns tentative work to otherwise idle stages. Therefore, it improves the throughput and latency compared with the lockstep scheme. The speculative synchronization provides almost identical protection as the lockstep scheme, but the energy consumption is higher because of higher switching activity in speculative execution.

C. Spatial Redundancy

To speed up the accumulation of the confidence level for a higher performance, spatial redundancy can be incorporated in conjunction with temporal redundancy. The simplest way to include spatial redundancy is by providing a duplicate datapath in the form of dual modular redundancy (DMR). A duplicate datapath allows the confidence to be accumulated quickly, thus increasing throughput and minimizing latency. DMR is less expensive than triple modular redundancy. Errors detected in DMR trigger recomputations for error correction in an iterative DMR method. The interstage synchronization is performed using either the lockstep or the speculative scheme that is described.

IV. Reliability and Performance Evaluation Using Sample-Based Emulation

To evaluate the reliability and performance of CDC, we propose an FPGA-based system emulation, which has been demonstrated to accelerate error simulation by up to six orders of magnitude [17]. Fast emulation permits reliability measurement down to low error rates that are relevant in practical systems, and bit- and cycle-accurate emulation also provides good performance measurements. Compared with other hardware emulators proposed in the past, our emulator uses realtime, on-FPGA error generation, instead of prestored error vectors or scan chains, hence a much higher emulation throughput is possible.

We use a coordinate rotation digital computer (CORDIC) processor as the test vehicle and the emulation platform is shown in Fig. 7. The platform consists of an error-free 16-bit, 12-stage CORDIC processor as the reference and an identical but fault-injected CORDIC processor protected by CDC. The entire system is mapped to a Xilinx Virtex-5 FPGA on a BEE3 platform [28]. The platform offers ample resources for evaluating complex designs.

An error injection block is added to the end of each CORDIC stage to simulate runtime errors as shown in Fig. 8(a). The error injection block inverts the output bits using XOR gates based on the error rate that is selected. Random errors are generated using a set of linear feedback shift registers (LFSR) by comparing their values to constants: if the LFSR values match the constants, bit errors are injected

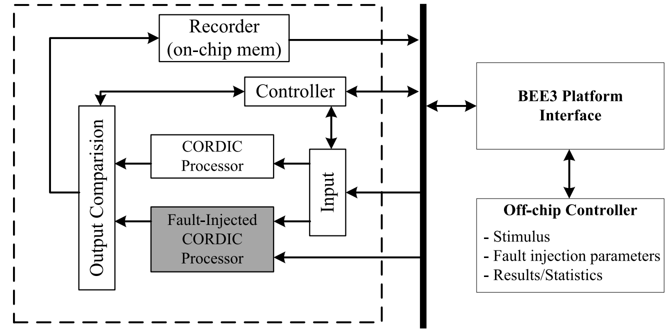

Fig. 7. FPGA-based system emulation platform for quantitative evaluation.

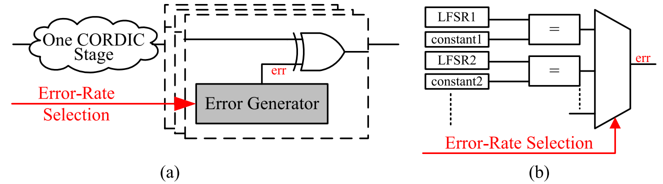

Fig. 8. (a) Error injection mechanism. (b) Error generator.

by inverting the bits. Because each LFSR produces 1 and 0 with nearly equal likelihood, the probability of error being generated is 2^{-x}, where x is the length of the constant. A set of maximal-length LFSRs of various lengths is used, each of which produces a different error probability. An error rate selector, shown in Fig. 8(b), sets the error rate by choosing one of the LFSRs. Using this platform, we were able to collect more than 1000 errors for a reliable estimation of the system error rate at 10^{-10} in one week.

A. Reliability Evaluation

Emulation proves the effectiveness of CDC. The system error rate (the probability of an incorrect system output) as a function of the circuit node error rate (the probability of error of any circuit node) is shown in Fig. 9(a). Without any protection, the error rate of the CORDIC processor is three orders of magnitude higher than the node error rate because of the large number of circuit nodes that are subject to errors. With confidence estimators placed at the end of sixth stage and 12th stage of the CORDIC processor and as we increase the confidence threshold, the system error rate decreases by at least four orders of magnitude when the node error rate is at 10^{-5} or lower.

To translate the error rates to practical terms, if we were to guarantee a mean time between failure (MTBF) of two years for a 1-GHz CORDIC processor, the required node error rate is 10^{-22} if no protection is used. This extremely low node error rate is possible with current mainstream CMOS technology but will likely become difficult with continued device scaling and the new generation of nano devices. Through inserting confidence estimators, the required node error rate is relaxed to 10^{-11} with a confidence threshold of 2 and 10^{-8} with a threshold of 3. The threshold can be adjusted at runtime based on the underlying circuit error rate and the application requirement.

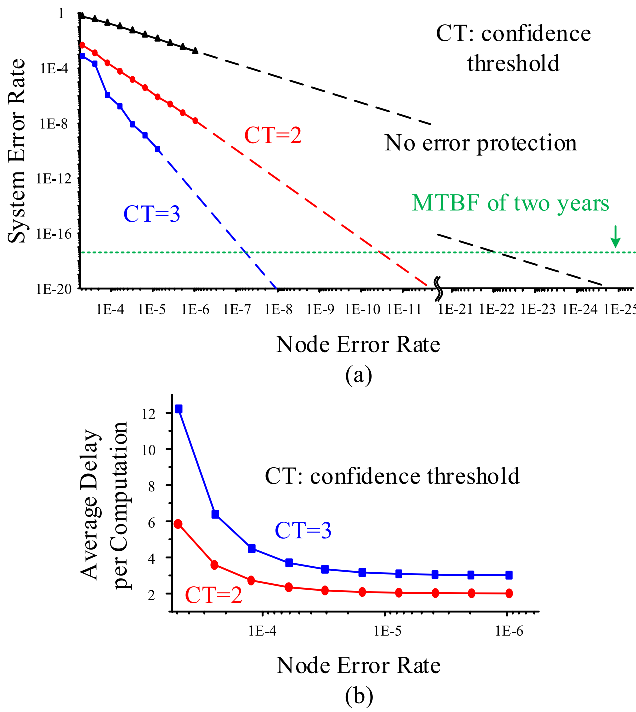

Fig. 9. (a) System error rate based on FPGA emulation results (solid line) and extrapolation (dashed line). (b) Average delay per computation as the confidence threshold is adjusted.

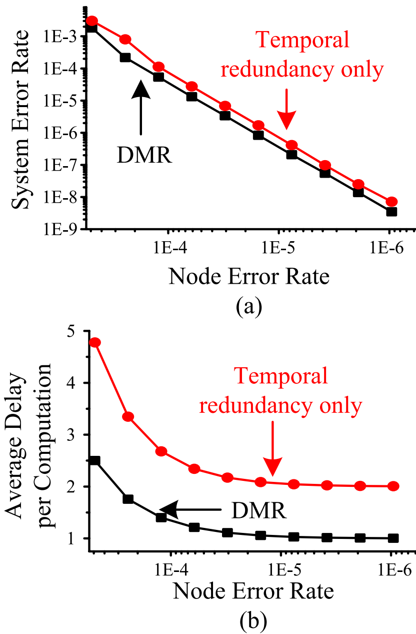

Fig. 10. (a) System error rate and (b) average delay of the CORDIC processor using temporal redundancy only and DMR (assume a confidence threshold of 2).

For a better protection, the DMR method can be applied. The improved reliability shown in Fig. 10(a) is due to the difference between confidence accumulation policies: an error in DMR invalidates one pair of computations from both the primary and duplicate datapath; thus the number of agreements needed on average to reach the confidence threshold is slightly higher. The DMR method is also capable of detecting permanent errors when a confidence estimator consistently reports disagreements and fails to meet the confidence threshold over a large number of clock cycles. An adaptive sparing method can be used to dynamically allocate additional spatial redundancy to overcome permanent errors.

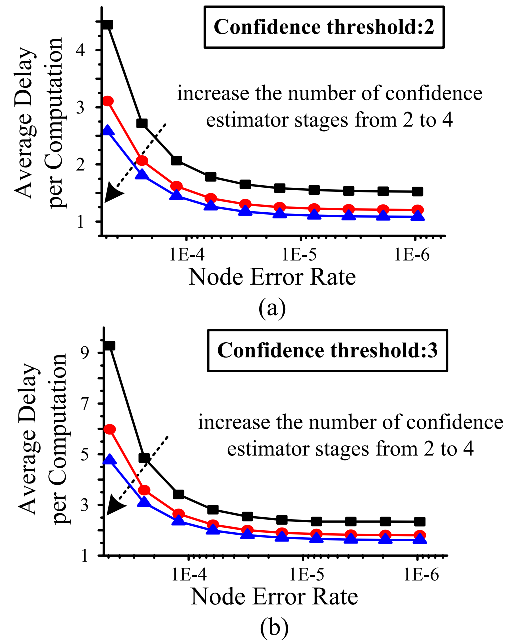

Fig. 11. Average delay per computation improves with more fine-grained placement of confidence estimators (a) at confidence threshold = 2 and (b) at confidence threshold = 3 when only temporal redundancy is considered.

B. Performance Evaluation

A high confidence threshold provides better protection but it also decreases the throughput. Fig. 9(b) shows the average delay per computation (the inverse of throughput): when the node error rate is moderate to low, a confidence threshold of 2 requires at least two clock cycles per computation and a threshold of 3 requires at least three cycles per computation. When the circuit node error rate is high, the delay becomes significant using a higher threshold. Frequent node errors slow down the process of gathering agreements. It is therefore more advantageous to operate CDC at the lowest confidence threshold that provides the necessary reliability. The runtime configurable confidence threshold accommodates device fluctuations and different reliability requirements among applications.

Confidence estimators can be placed in finer grained intervals to reduce the throughput penalty. A fine-grained placement of confidence estimators shortens the path and bounds the probability of error, contributing to a faster convergence toward the required reliability. The sampling clock frequency can also be increased because of the shortened delay per stage. Fig. 11 shows the improved average delay per computation as more stages of confidence estimators are inserted to the same CORDIC processor. The improvement becomes more significant at high circuit node error rates because of the faster convergence toward the confidence threshold. The delay improvement is also attributed to the increased clock frequency as more confidence estimator stages are inserted.

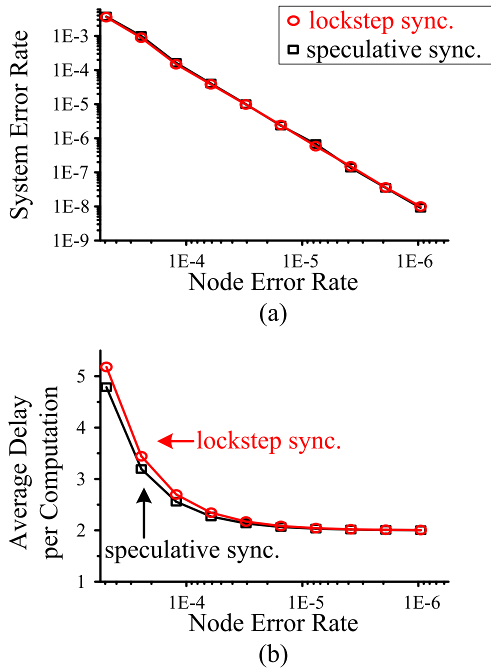

Fig. 12. (a) System error rate and (b) average delay of the CORDIC processor using lockstep and speculative synchronization.

TABLE I
AREA AND ENERGY OF CDC WITH TEMPORAL REDUNDANCY ONLY
(CE: CONFIDENCE ESTIMATOR. CT: CONFIDENCE THRESHOLD)

	1-stg. CE	2-stg. CE	3-stg. CE	4-stg. CE
Norm. Clk Period	1.30	0.76	0.60	0.54
Norm. Area	1.08	1.21	1.34	1.46
Norm. Energy (CT=2)	1.14	1.20	1.36	1.53
Norm. Energy (CT=3)	1.15	1.23	1.46	1.64

These discussions are based on lockstep synchronization. The speculative synchronization further improves the throughput when the circuit node error rate is high as shown in Fig. 12(b). In addition, the speculative scheme shortens the latency of computation: when the circuit node error rate is moderate to low, the latency of computation is almost independent of the confidence threshold because a tentative output is passed along without waiting, followed by parallel checking performed at all the stages. Therefore, the speculative synchronization is especially attractive for latency-sensitive applications.

C. Implementation Results and Case Study

We estimate the performance, area, and energy of the CDC design by synthesizing it along with a CORDIC processor using a 45-nm CMOS technology. Table I lists the comparison. Having one stage of confidence estimator increases the clock period by 30% and introduces an 8% area overhead. Additional stages of confidence estimators reduce the clock period, but the area overhead rises. There is, however, a limit to how fine-grained confidence estimators can be efficiently placed due to the diminishing improvement in throughput and the escalating cost of area and energy. Design time decisions need to be made based on the expected range of circuit error rates along with the area and energy constraints imposed by the design. DMR provides better throughput, but the energy and area are 58% and 64% higher, respectively, by comparing the results in Tables I and II (when using four stages of confidence estimators).

TABLE II
AREA AND ENERGY OF CDC WITH DMR

	1-stg. CE	2-stg. CE	3-stg. CE	4-stg. CE
Norm. Area	2.03	2.15	2.28	2.40
Norm. Energy	2.12	2.17	2.27	2.42

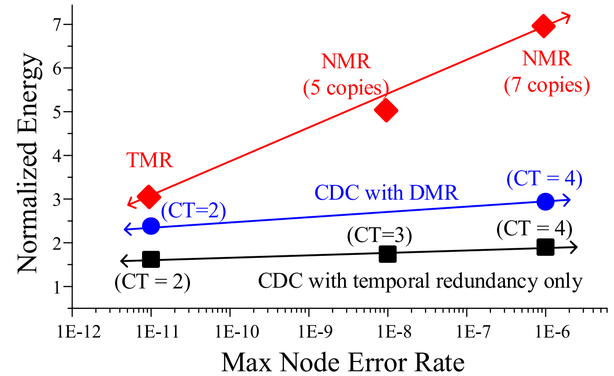

Fig. 13. Normalized energy overhead of three schemes for a highly reliable general purpose processor that requires an MTBF of two years.

We show a case study based on a highly reliable general-purpose processor that requires two years of MTBF. Fig. 13 shows the tradeoff between the energy and system reliability of CDC compared with the conventional NMR scheme. CDC guarantees a given system error rate of 10^{-17}, even when the underlying circuit node error rate fluctuates between 10^{-11} and 10^{-6}. The protection can be accomplished in several ways: temporal redundancy only or DMR with either lockstep or speculative synchronization. When circuit node error rate increases, the reliable system is achieved by setting a higher confidence threshold with a gradual increase of energy. In comparison, the energy of N-module redundancy is much higher.

V. EARLY CHECKING

In CDC, error checking holds back the datapath. Errors increase the delay to reach the required confidence, resulting in an inefficient checking for short-duration errors, such as soft errors and errors because of coupling noise and voltage droop. To overcome this challenge, we propose an EC technique, built on top of the CDC model. The EC technique exploits the delay slack in the vast majority of the datapaths for fast error checking and confidence accumulation.

We first show in Fig. 14 a realistic example of path delay profile of a synthesized 16-bit 12-step CORDIC processor assuming uniform random inputs. The delay profile can be modeled using a Gaussian distribution with a mean of 0.75 and a standard deviation of 0.08, normalized to the longest path delay. The clock cycle boundary is drawn at 1 to contain the longest path. Extra margins are often added in practice

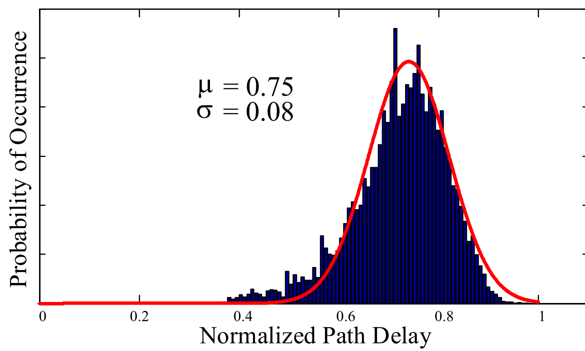

Fig. 14. Path delay distribution of a 16-bit 12-step CORDIC processor.

to accommodate process variation and runtime fluctuation. A research in [16] also shows that 70% of the datapaths in a commercial processor have at least 20% timing slack. Therefore, the confidence estimator can start sampling early, well before the clock edge, thus the name EC.

A. Early Checking

Using EC, glitches because of transient and soft errors will be detected and invalidated and a required confidence can be met before the clock edge without using an extra clock cycle. If the required confidence cannot be met by the clock edge because of long path delay or errors or both, an additional clock cycle is needed. Long path delay is statistically less likely, thus the performance penalty will, however, remain low.

To implement EC, we decouple datapath from error checking and place them on separate clocks. Error checking will operate on a possibly faster sampling clock sclk, so the confidence level can start accumulating early and quickly. The datapath samples the confidence level at the main clock clk and makes the propagation decision. EC can be shown using the same block diagram in Fig. 4, except that the checking path runs on sclk instead of clk. The interface between the datapath and error checking is the confidence level—error checking accumulates the confidence level at the sclk frequency and the datapath inspects the confidence level at the clk frequency. It is not necessary to have sclk and clk synchronized or phase aligned; thus the sclk generation can be simplified or self timed using a small ring oscillator. The sclk of different stages do not need to be synchronized either, and sclk can also be shared among multiple stages to save cost.

B. Functional Analysis and Comparison

The conceptual timing diagram of EC is shown in Fig. 15, where the sampling clock sclk runs independently of the main clock clk and the two are not necessarily phase aligned. The confidence estimator starts accumulating confidence at the first possible sclk edge. If sclk is fast, the confidence level quickly reaches the threshold and saturates. After some time, D makes the final transition to the correct value. The transition is detected by the confidence estimator as a disagreement, which resets the confidence level to 0. The confidence estimator continues to sample D and looks for additional agreements. On the next edge of clk, the confidence level is inspected and

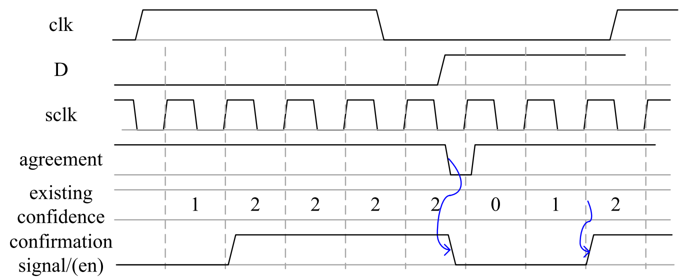

Fig. 15. Conceptual timing diagram illustrating EC.

as it matches the confidence threshold, the output of the current stage is allowed to propagate through the main flip-flop.

In an error-free operation, a late arriving output will be treated as tentative if an insufficient confidence level is accumulated before the clk edge, and the output will be held in the current pipeline stage for one more cycle. The probability of holding depends on how often the long paths are exercised. This probability can be very low for delay distributions with long tails, as in the case of deep submicron circuits running at a reduced supply voltage [20]. We expect that the performance loss will be low if the critical paths are only infrequently exercised and the confidence level is relatively low.

The effect of a transient fault or soft error depends on its arrival time and duration. If an error arrives before the output makes the final transition, it is not different from the error-free case that is discussed. If an error arrives at the same time or after the output makes the final transition, it will prolong the final confidence accumulation and increase the probability of holding. A high error rate or long error duration will further increase the holding probability, leading to performance loss.

The EC technique can be compared with other common approaches such as Razor and BISER that also target errors of short duration. The comparison is shown in Table III. Razor double samples and looks for an agreement. It introduces a small area overhead, and the impact on performance is small. The protection level is, however, dictated by the checking duration and it is fixed during design time. BISER has been presented to monitor errors by detecting suspicious transitions. Its area overhead is small but the impact on performance is decided by the checking duration. A longer checking duration results in more performance degradation. Similar to Razor, the checking duration of BISER is fixed and cannot be changed to adapt to various protection levels. In contrast, the EC technique offers a tunable protection level by setting the confidence threshold. The impact on performance depends on the confidence threshold and the runtime delay distribution.

VI. Reliability and Performance Evaluation Using Event-Based Simulation

The effectiveness of a resilient design is highly dependent on the complex interplay of error events (error occurrence and duration), path events (path start and end), and the error protection mechanism. The FPGA-based emulation is cycle-based and events can only occur at clock cycle boundaries. Such a platform is sufficient for a sample-based technique such as CDC that is described. The EC technique is, however,

TABLE III
COMPARISONS OF ERROR DETECTION TECHNIQUES WITH SHORT CHECKING DURATIONS

Technique	Razor [6]	BISER [7]	CDC with Early Checking
Mechanism	post-clock-edge checking	pre-clock-edge checking	multiple checkings before the clock edge
Protection level	fixed in design time	fixed in design time	tunable by changing confidence threshold
Area overhead	small	small	moderate
Performance impact	small	larger impact with longer checking window	depends on confidence threshold and runtime delay distribution

more fine grained, and it requires a transient simulator for the quantitative evaluation.

To speed up the transient simulation, we propose an FPGA-based event simulator. The simulator consists of path delay model, error model, and the EC technique to be tested. The simulation operates on the FPGA system clock. The main clock period, sampling clock period, path delays, and error durations are in units of the FPGA system clock period T_{sys}. For example, we can set the main clock period $T_{clk} = 1000T_{sys}$, and the sampling clock (sclk) period T_{sclk} is set independently, e.g., $T_{sclk} = 100T_{sys}$. The path delay model is implemented as a random number generator based on its statistical distribution (see Fig. 14). The transient and soft error models are implemented similarly.

The simulation is conducted in steps of T_{sys}. In each T_{sys}, the simulation controller determines whether the path under test gives a valid output based on the time elapsed, delay generated by the delay model, and error occurrence and duration given by the error model. The confidence estimator samples the output at sclk, and increments or resets the confidence level. The confidence level is sampled at clk to decide whether to propagate the output or to hold it for one more cycle.

The FPGA simulation platform operates at a frequency of 100 MHz or higher, thus a 100 kHz simulation throughput is possible if T_{clk} is set to $1000T_{sys}$. This setup allows us to collect about 20 errors in two weeks for an estimation of the system error rate at 10^{-10}. A finer time resolution is possible, but the simulation throughput will be reduced proportionally.

A. Error Simulation

A transient error flips the validity of the path output (from valid to invalid) and causes an error. Both the error arrival time and duration are tunable in T_{sys} steps. Fig. 16 shows the construction of the error model. In each T_{sys} step, it produces a conditional error using an LFSR. To model error duration, a counter is started upon error assertion to keep the error for the error duration. The error duration can also be randomly generated.

Fig. 17 illustrates the timing of the FPGA-based event simulation platform. All events are lined up with the FPGA system clock. Error events can be programmed to model their natural occurrence and duration. The simulation platform is general purpose and not tied to any circuit implementation. It provides a high simulation throughput and can be easily extended to other related work.

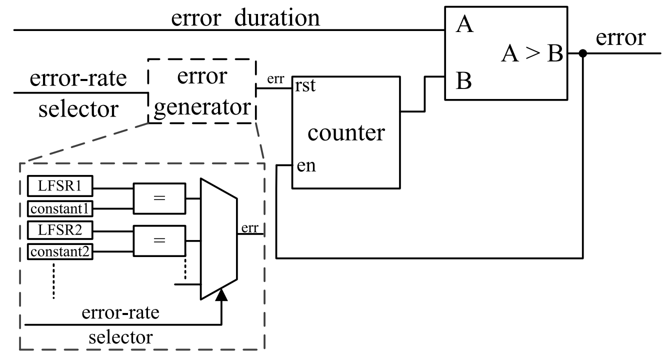

Fig. 16. Error generator based on error rate and duration.

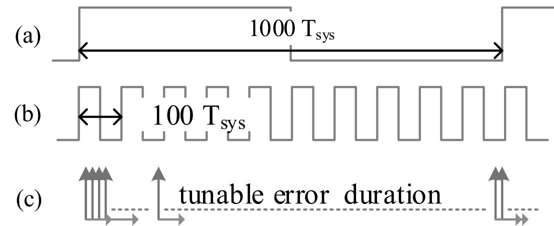

Fig. 17. Timing charts of the FPGA-based event simulation. (a) Main clock. (b) Sampling clock. (c) Error generation.

TABLE IV
NORMALIZED THROUGHPUT OF EC IN ERROR-FREE OPERATIONS

	CT=1	CT=2	CT=3	CT=4
T_{sclk}=0.05T_{clk}	0.997	0.965	0.841	0.683
T_{sclk}=0.1T_{clk}	0.981	0.723	0.510	0.503

B. Experimental Evaluation

To set up the experiment, we set $T_{clk} = 1000T_{sys}$, and created a Gaussian path delay model following the path delay distribution shown in Fig. 14, with a mean and standard deviation set to $0.8T_{clk}$ and $0.0625T_{clk}$. The tail of the distribution beyond 3.2σ is saturated to meet T_{clk}.

We first evaluate the performance of EC under error-free operations. Table IV shows the normalized throughput. In an error-free operation, the throughput is determined by the length of the error detection window, or the product of the sampling clock period T_{sclk} and the confidence threshold. Increasing the confidence threshold prolongs the error detection window, degrading the throughput because of long paths that need extra time to meet a required threshold. Nevertheless, the throughput can be improved by shortening T_{sclk}. For example,

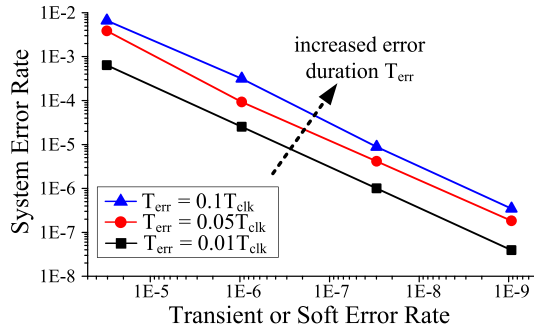

Fig. 18. System error rate without any error protection (error duration T_{err} is normalized to T_{clk}).

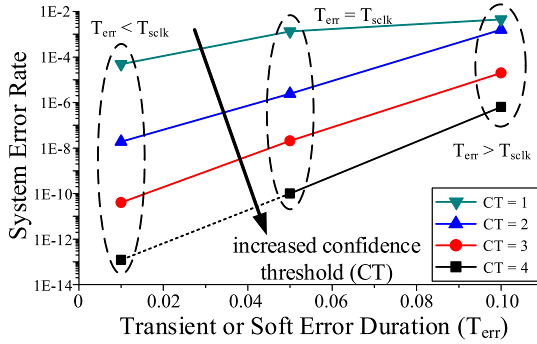

Fig. 19. System error rate as the confidence threshold is adjusted (assume a transient or soft error rate of 10^{-5} and $T_{sclk} = 0.05T_{clk}$. Error duration T_{err} is normalized to T_{clk}). FPGA simulation results (solid line) and extrapolation (dotted line).

at a confidence threshold of 3, shortening T_{sclk} from $0.1T_{clk}$ to $0.05T_{clk}$ improves the throughput by 65%.

As errors are injected to the system, the reliability starts to degrade. Fig. 18 shows that the CORDIC processor's error rate increases with transient and soft error rate and duration. Fig. 19 shows the effect of adjusting the confidence threshold while holding T_{sclk} and the transient error rate constant. As the confidence threshold is raised by 1, the system reliability is improved by three orders of magnitude or more. It is, however, important to notice from Fig. 19 that the sampling clock period T_{sclk} should be chosen to match or exceed the error duration T_{err} to guarantee a minimum error detection window for an appreciable improvement in reliability.

Fig. 20 shows the effect of errors of long and short durations. If errors last for a relatively long duration, e.g., $T_{err} = 0.1T_{clk}$, it is necessary to select the sampling clock period $T_{sclk} \geq T_{err}$ for a more effective improvement of the system reliability. On the other hand, if errors are known to be very short, e.g., $T_{err} = 0.01T_{clk}$, the length of T_{sclk} has little impact on the system reliability. The throughput is given in Fig. 21. A long T_{sclk} or a higher confidence threshold degrades the throughput. Interestingly, the normalized throughput saturates to approximately 0.5 as one additional (main) clock cycle provides more than sufficient time for accumulating a high enough confidence using EC.

To summarize the results, the throughput of a system is determined by the length of the error detection window, or the product of T_{sclk} and confidence threshold. A short window enables a higher throughput as less time is needed to attain

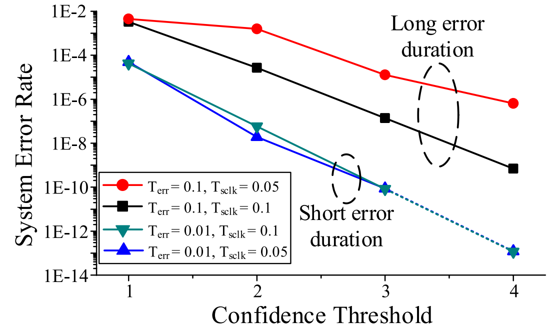

Fig. 20. System error rate versus confidence threshold [assume a transient (or soft) error rate of 10^{-5}]. FPGA simulation results (solid line) and extrapolation (dotted line).

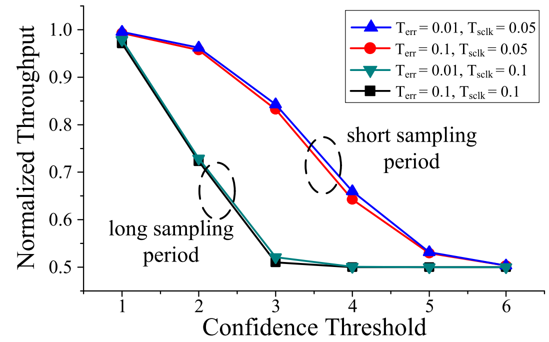

Fig. 21. Normalized throughput versus confidence threshold [assume a transient (or soft) error rate of 10^{-5}].

TABLE V
NORMALIZED ENERGY OF SYSTEM BASED ON EC

	CT=1	CT=2	CT=3	CT=4
Norm. Energy	1.191	1.203	1.210	1.222

the required confidence. For a good protection, T_{sclk} needs to match or exceed T_{err}. Therefore, we formulate a two-step design strategy: 1) tune T_{sclk} to match the expected error duration and 2) set the confidence threshold to obtain the required system error rate.

C. Implementation Results and Case Study

A 16-bit, 12-step CORDIC processor is synthesized in a 45-nm CMOS technology. The area overhead of implementing EC is only 12%. The energy overhead is as low as 20% at a low confidence threshold, and it increases marginally with a higher confidence threshold as listed in Table V.

We formulate a case study based on a general purpose processor that requires a very low system error rate of 10^{-20}. Transient and soft error rates are assumed to vary between 10^{-9} and 10^{-5}, and the error duration varies between $0.01T_{clk}$ and $0.1T_{clk}$. Resilient computing can be accomplished by tuning the confidence threshold and sampling clock period T_{sclk}, the results of which are shown in Fig. 22. For example, when the transient (or soft) error rate is 10^{-9}, the confidence threshold can be set to 3, 4 or 5 at $T_{sclk} = 0.05T_{clk}$ to accommodate error durations ranging from $0.01T_{clk}$ to $0.1T_{clk}$. The long error duration of $0.1T_{clk}$ warrants an increase of

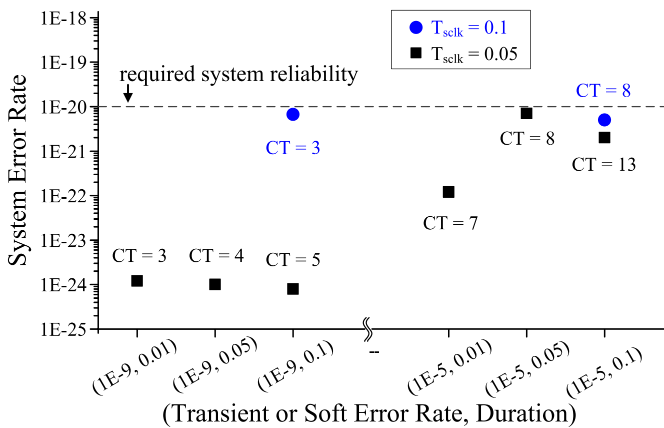

Fig. 22. Adjustment of confidence threshold and sampling clock period to deliver the required system error rate of 10^{-20}.

T_{sclk} to $0.1T_{clk}$ and the confidence threshold can be reduced to 3 while still satisfying the system error rate requirement. A higher transient (or soft) error rate of 10^{-5} requires a higher confidence threshold of 7, 8, and 13 for error durations ranging from $0.01T_{clk}$ to $0.1T_{clk}$ to meet the system reliability.

VII. Conclusion

We present a CDC model to protect the circuits built on highly variable and nondeterministic devices. The reliability is enhanced by confidence estimators that rely on either temporal redundancy only or the combination of temporal and spatial redundancies. The area and energy cost can be kept low using temporal redundancy only, whereas the combined temporal and spatial redundancy provides a higher throughput. The two methods follow one of the two synchronization schemes: lockstep or speculative. The speculative scheme permits a lower latency and higher throughput with a small increase in energy.

The CDC model uses cycle-based checking targeting nondeterministic nanodevices such as memristors. To achieve a better performance for deeply scaled CMOS as memristor circuits that are mostly affected by transient faults and soft errors, we present a variation of CDC using EC. The EC technique recycles unused cycle time to accumulate confidence level, allowing fast response to randomly occurring errors of short duration.

The CDC model is emulated on an FPGA platform with realtime error injection to prove its effectiveness. Quantitative evaluations of reliability and performance shed light on the choice of confidence threshold, placement of confidence estimators, and synchronization.

To evaluate the performance of the EC technique with transient errors, an FPGA-based event simulator is presented to incorporate both path delay and error models at much finer time scale. The simulation captures complex interactions between path delays, errors, and protection scheme. The EC technique is demonstrated to be highly effective against short transient errors. It improves the system reliability by more than four orders of magnitude if the errors are of short duration, while the performance loss is kept as low as 0.3%.

The CDC model and the EC technique are the promising solutions to bridge the gap between different applications and fluctuating device behavior in deeply scaled CMOS and future device technologies. The design insights will allow us to construct a reliability diverse computer architecture with computing elements that provide a range of reliability levels at appropriate energy cost to deliver the required performance.

Acknowledgment

The authors would like to thank Y. Kim for help implementing the CDC model and H. Naeimi, S. Sandhu, F. Sheikh, and K. Bowman for suggestions.

References

[1] S. Borkar, "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Dec. 2005.

[2] J. W. McPherson, "Reliability challenges for 45 nm and beyond," in *Proc. ACM/IEEE Design Autom. Conf.*, Jul. 2006, pp. 176–181.

[3] S. Borkar, "Design perspectives on 22 nm CMOS and beyond," in *Proc. ACM/IEEE Design Autom. Conf.*, Jul. 2009, pp. 93–94.

[4] M. Horowitz, E. Alon, D. Patil, S. Naffziger, R. Kumar, and K. Bernstein, "Scaling, power, and the future of CMOS," in *IEEE IEDM Tech. Dig.*, Dec. 2005, pp. 7–15.

[5] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proc. Int. Conf. Dependable Syst. Netw.*, 2002, pp. 389–398.

[6] S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D. Bull, and D. Blaauw, "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.

[7] M. Zhang, S. Mitra, T. M. Mak, N. Seifert, N. J. Wang, Q. Shi, K. S. Kim, N. R. Shanbhag, and S. J. Patel, "Sequential element design with built-in soft error resilience," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 12, pp. 1368–1378, Dec. 2006.

[8] J. Crop, E. Frimer, N. Moezzi-madni, T. Ruggeri, R. Pawlowski, P. Chiang, and M. Erez, "Error detection and recovery techniques for variability-aware CMOS computing: A comprehensive review," *J. Low Power Electron. Appl.*, vol. 1, no. 3, pp. 334–356, Oct. 2011.

[9] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *IEEE J. Solid-State Circuits*, vol. 37, no. 2, pp. 183–190, Feb. 2002.

[10] D. Frank, R. Dennard, E. Nowak, P. Solomon, Y. Taur, and H.-S. P. Wong, "Device scaling limits of Si MOSFETs and their application dependencies," *Proc. IEEE*, vol. 89, no. 3, pp. 259–288, Mar. 2001.

[11] (2008). *International Technology Roadmap for Semiconductors* [Online]. Available: http://public.itrs.net/

[12] S. H. Jo, K.-H. Kim, and W. Lu, "Programmable resistance switching in nanoscale two-terminal devices," *Nano Lett.*, vol. 9, no. 1, pp. 496–500, 2009.

[13] S. E. Savelev, A. S. Alexandrov, A. M. Bratkovsky, and R. S. Williams, "Molecular dynamics simulations of oxide memristors: Crystal field effects," *Appl. Phys. Lett.*, vol. 99, no. 5, pp. 053108-1–053108-3, Aug. 2011.

[14] K. Shin and H. Kim, "A time redundancy approach to TMR failures using fault-state likelihoods," *IEEE Trans. Comput.*, vol. 43, no. 10, pp. 1151–1162, Oct. 1994.

[15] A. Ajami, K. Banerjee, A. Mehrotra, and M. Pedram, "Analysis of IR-drop scaling with implications for deep submicron P/G network designs," in *Proc. 4th Int. Symp. Quality Electron. Design*, Mar. 2003, pp. 35–40.

[16] 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.*, Mar. 2010, pp. 1554–1559.

[17] A. Pellegrini, K. Constantinides, D. Zhang, S. Sudhakar, V. Bertacco, and T. Austin, "CrashTest: A fast high-fidelity FPGA-based resiliency analysis framework," in *Proc. IEEE Int. Conf. Comput. Design*, Oct. 2008, pp. 363–370.

[18] C. Lopez-Ongil, M. Garcia-Valderas, M. Portela-Garcia, and L. Entrena, "Autonomous fault emulation: A new FPGA-based acceleration system for hardness evaluation," *IEEE Trans. Nuclear Sci.*, vol. 54, no. 1, pp. 252–261, Feb. 2007.
[19] M. Breuer, S. Gupta, and T. Mak, "Defect and error tolerance in the presence of massive numbers of defects," *IEEE Des. Test Comput.*, vol. 21, no. 3, pp. 216–227, Jun. 2004.
[20] R. Hegde and N. Shanbhag, "Energy-efficient signal processing via algorithmic noise-tolerance," in *Proc. Int. Symp. Low Power Electron. Design*, Aug. 1999, pp. 30–35.
[21] S. Narayanan, J. Sartori, R. Kumar, and D. Jones, "Scalable stochastic processors," in *Proc. Design, Autom. Test Eur. Conf.*, Mar. 2010, pp. 335–338.
[22] H. Cho, L. Leem, and S. Mitra, "ERSA: Error resilient system architecture for probabilistic applications," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 31, no. 4, pp. 546–558, Apr. 2012.
[23] K. Nikolic, A. Sadek, and M. Forshaw, "Fault-tolerent techniques for nanocomputers," *Nanotechnology*, vol. 13, no. 3, pp. 357–362, May 2002.
[24] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, and M. Orshansky, "BulletProof: A defect-tolerant CMP switch architecture," in *Proc. 25th Int. Symp. High-Perform. Comput. Archit.*, Feb. 2006, pp. 5–16.
[25] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *Proc. 17th IEEE VLSI Test Symp.*, Apr. 1999, pp. 86–94.
[26] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: A low-power pipeline based on circuit-level timing speculation," in *Proc. 36th Annu. IEEE/ACM Int. Symp. Microarchit.*, Dec. 2003, pp. 7–18.
[27] H. Naeimi and A. DeHon, "Fault-tolerant sub-lithographic design with rollback recovery," *Nanotechnology*, vol. 19, no. 11, pp. 357–362, Feb. 2008.
[28] J. D. Davis, C. P. Thacker, and C. Chang, "BEE3: Revitalizing computer architecture research," Microsoft Res., New York, NY, USA, Tech. Rep. MSR-TR-2009-45, 2009.
[29] C.-H. Chen, Y. Kim, Z. Zhang, D. Blaauw, D. Sylvester, H. Naeimi, and S. Sandhu, "A confidence-driven model for error-resilient computing," in *Proc. Design, Autom. Test Eur. Conf.*, Mar. 2011.

Chia-Hsiang Chen (S'10) received the B.S. degree in electrical engineering and computer science through the honor program of National Chiao-Tung University, Hsinchu, Taiwan, in 2008, and the M.S. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, MI, USA, in 2012, where he is currently pursing the Ph.D. degree in electrical engineering.

He was with the Circuit Research Laboratory and Integrated Platform Research, Intel, Hillsboro, OR, USA, in 2012 and 2012. His current research interests include resilient circuits, architecture, and co-optimization.

David Blaauw (M'94–SM'07–F'12) received the B.S. degree in physics and computer science from Duke University, Durham, NC, USA, in 1986, and the Ph.D. degree in computer science from the University of Illinois, Urbana, IL, USA, in 1991.

He was with Motorola, Inc., Austin, TX, USA, in 2001, where he was the Manager of the High Performance Design Technology Group. Since August 2001, he has been with the Faculty of the University of Michigan, Ann Arbor, MI, USA, where he is a Professor. He has published over 350 papers and holds 40 patents. His work has focused on VLSI design with particular emphasis on ultralow power and high performance design.

Dr. Blaauw was the Technical Program Chair and a General Chair for the International Symposium on Low Power Electronic and Design. He was the Technical Program Co-Chair of the ACM/IEEE Design Automation Conference and a member of the ISSCC Technical Program Committee.

Dennis Sylvester (S'95–M'00–SM'04–F'11) received the Ph.D. degree in electrical engineering from the University of California, Berkeley, CA, USA. His dissertation was recognized with the David J. Sakrison Memorial Prize as the most outstanding research in the UC-Berkeley EECS Department.

He is a Professor of electrical engineering and computer science with the University of Michigan, Ann Arbor, MI, USA, and the Director of the Michigan Integrated Circuits Laboratory, a group of ten faculty and 60-plus graduate students. He previously held research staff positions with the Advanced Technology Group of Synopsys, Mountain View, CA, USA, and Hewlett-Packard Laboratories, Palo Alto, CA, USA, and a visiting professorship of electrical and computer engineering with the National University of Singapore, Singapore. He has published over 350 articles with one book and several book chapters. He holds 16 U.S. patents. His current research interests include the design of millimeter-scale computing systems and energy efficient near threshold computing.

Dr. Sylvester serves as a Consultant and Technical Advisory Board Member for electronic design automation and semiconductor firms. He co-founded Ambiq Micro, a fabless semiconductor company developing ultralow power mixed-signal solutions for compact wireless devices. He received the National Science Foundation CAREER Award, the Beatrice Winner Award at ISSCC, the IBM Faculty Award, the SRC Inventor Recognition Award, and eight best paper awards and nominations. He was a recipient of the ACM SIGDA Outstanding New Faculty Award and the University of Michigan Henry Russel Award for distinguished scholarship. He has served on the technical program committee of major design automation and circuit design conferences, the Executive Committee of the ACM/IEEE Design Automation Conference, and the Steering Committee of the ACM/IEEE International Symposium on Physical Design. He has served as Associate Editor for the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS.

Zhengya Zhang (S'02–M'09) received the M.S. and Ph.D. degrees in electrical engineering from the University of California, Berkeley, CA, USA, in 2005 and 2009, respectively.

He has been with the Faculty of the University of Michigan, Ann Arbor, MI, USA, as an Assistant Professor, Department of Electrical Engineering and Computer Science, since 2009. His current research interests include low-power and high-performance VLSI circuits and systems for computing, communications and signal processing to leverage emerging nanodevices, architectures, and signal processing algorithms.

Dr. Zhang was a recipient of the National Science Foundation CAREER Award in 2011, the Intel Early Career Faculty Honor Program Award in 2013, the David J. Sakrison Memorial Prize for Outstanding Doctoral Research in EECS at UC Berkeley, and the Best Student Paper Award at the Symposium on VLSI Circuits. He is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS..