Soft-Error Tolerance and Mitigation in Asynchronous Burst-Mode Circuits

Sobeeh Almukhaizim, *Member, IEEE*, Feng Shi, *Member, IEEE*, Eric Love, and Yiorgos Makris, *Senior Member, IEEE*

***Abstract*—We discuss the problem of soft errors in asynchronous burst-mode machines (ABMMs), and we propose two solutions. The first solution is an error tolerance approach, which leverages the inherent functionality of Muller C-elements, along with a variant of duplication, to suppress all transient errors. The proposed method is more robust and less expensive than the typical triple modular redundancy error tolerance method and often even less expensive than previously proposed concurrent error detection methods, which only provide detection but no correction. The second solution is an error mitigation approach, which leverages a newly devised soft-error susceptibility assessment method for ABMMs, along with partial duplication, to suppress a carefully chosen subset of transient errors. Three progressively more powerful options for partial duplication select among individual gates, complete state/output logic cones, or partial state/output logic cones and enable efficient exploration of the tradeoff between the achieved soft-error susceptibility reduction and the incurred area overhead. Furthermore, a gate-decomposition method is developed to leverage the additional soft-error susceptibility reduction opportunities arising during conversion of a two-level ABMM implementation into a multilevel one. Extensive experimental results on benchmark ABMMs assess the effectiveness of the proposed methods in reducing soft-error susceptibility, and their impact on area, performance, and offline testability.**

***Index Terms*—Asynchronous burst-mode circuits, soft errors, soft-error mitigation, soft-error susceptibility, soft-error tolerance.**

I. Introduction

SOFT ERRORS are emerging as a serious threat to the reliable operation of integrated circuits (ICs). When high-energy neutrons or alpha particles strike a sensitive region in a semiconductor device, they generate a single-event transient (SET) that may alter the state of the system, resulting in a soft error. The projected increase in the Soft-Error failure Rate (SER) of near-future CMOS technologies has sparked numerous efforts to develop error protection mechanisms for digital ICs [1]–[4]. Since the majority of commercial ICs available in the marketplace follow the clocked design paradigm, most of these efforts target synchronous circuits. Yet, clockless design has recently received increased attention, and several advanced asynchronous design styles have been carving an increasing niche in the market [5]. Moreover, due to their low power consumption and electromagnetic noise emission, asynchronous circuits are gaining a particularly strong foothold in mission-critical applications, wherein reliability is key.

Soft-error protection techniques can be divided into *error tolerance* and *error mitigation* approaches. The former take an expensive holistic approach and attempt to tolerate all SETs in the circuit, while the latter aim to explore the tradeoff between the provided protection and the incurred cost. Unfortunately, tolerance and mitigation methods developed for synchronous circuits are not directly portable to the asynchronous domain. Moreover, and while a few soft-error analysis, tolerance, and design hardening methods have been developed for the class of quasi-delay-insensitive (QDI) circuits [6], [7], their utility is limited in other classes, each of which presents its own challenges.

This paper aims to provide an array of solutions for coping with soft errors in the class of asynchronous burst-mode machines (ABMMs). Specifically, the contributions of this paper include the following.

1) A duplication-based soft-error-tolerant ABMM design methodology, which leverages the inherent functionality of C-elements to reduce the cost and improve the robustness of the triple modular redundancy (TMR) approach.
2) A soft-error susceptibility assessment methodology for ABMMs, based on an enhanced version of a previously developed asynchronous-circuit fault simulator [9].
3) A soft-error mitigation solution, based on the newly developed soft-error susceptibility assessment methodology. Three alternative partial duplication options, which select judiciously among individual gates, complete cones of state/output logic, or partial cones of state/output logic, are proposed in order to explore the tradeoff between area overhead and SER reduction in ABMMs.
4) A gate-decomposition strategy, which maximizes the ability of the decomposed structure to suppress soft errors when a two-level ABMM is converted into a multilevel equivalent. Formulated as an integer linear program (ILP), the proposed method aims at maximizing *logic masking* through efficient distribution of the input signals of each decomposed gate to its constituents.
5) A halting-based test generation method that retains offline testability for most faults in a soft-error-tolerant ABMM, based on an extension of a previously developed test generation tool [10].

This paper is organized as follows. In Section II, we review related work in soft-error tolerance and mitigation in

Manuscript received July 01, 2008; revised November 13, 2008. First published June 05, 2009; current version published June 19, 2009.

S. Almukhaizim is with the Department of Computer Engineering, Kuwait University, Safat 13060, Kuwait (e-mail: sobeeh.almukhaizim@ku.edu.kw).

F. Shi is with the Central Analog Department, Marvell Semiconductor, Santa Clara, CA 95054 USA (e-mail: feng.shi@gmail.com).

E. Love and Y. Makris are with the Department of Electrical Engineering, Yale University, New Haven, CT 06520-8285 USA (e-mail: eric.love@yale.edu; yiorgos.makris@yale.edu).

Digital Object Identifier 10.1109/TVLSI.2009.2014381

asynchronous circuits. In Section III, we briefly introduce ABMMs. In Section IV, we describe TMR and the proposed duplication-based soft-error-tolerant ABMM design method. In Section V, we devise a fault-simulation-based method to compute soft-error susceptibility in ABMMs. Then, in Section VI, we describe the proposed partial-duplication-based soft-error mitigation solutions and their utility in effectively exploring the design space. Next, in Section VII, we present our gate-decomposition strategy for further reducing soft-error susceptibility in multilevel ABMMs. In Section VIII, we discuss the offline testability of soft-error-tolerant ABMMs. Finally, in Section IX, we assess experimentally the overhead incurred by the proposed methods and demonstrate their ability to reduce ABMM susceptibility to soft errors in a cost-effective manner.

II. Related Work

Two main studies related to soft errors in asynchronous circuits have been previously performed [6], [7]. Both target the class of QDI circuits, with the first focusing on soft-error tolerance and the second focusing on soft-error susceptibility analysis and hardening. In the following, we summarize these two studies and outline the reasons due to which they cannot be directly applied to ABMMs.

Jang *et al.* [6] investigated the effect of soft errors on the operation of QDI circuits. Their analysis reveals that a soft error may not only produce erroneous output results but may also lead the circuit to a deadlock state. Thus, a traditional TMR approach cannot be employed to tolerate soft errors, since two soft errors accumulated over time could deadlock two of the replicas, rendering the TMR system ineffective. In order to make a QDI circuit soft error tolerant, the authors propose a gate-level *fine-grain duplication and double-checking* method; every gate is duplicated, and each pair of nominally identical outputs is fed to two C-elements [11]. A C-element is a state-holding component that waits for all of its inputs to agree on a logic value before it changes its state to this value. Hence, a transient error at a gate is blocked by the correct value of the duplicate gate and does not propagate to the output of the C-element. While this method could potentially be ported to ABMMs, the fine granularity at which it is applied would result in very high overhead since it would require two C-elements per gate. Instead, inspired by this method, we propose a coarse-grain variant, which adds significantly fewer C-elements.

Monnet *et al.* [7] were the first to quantify the susceptibility of QDI circuits to soft errors. In their analysis, the circuit is divided into two parts, namely, a computational logic part and a memory part, which implements the communication protocol. The global state of the circuit is defined as the state of all its C-elements implementing the memory part. The sensitivity of each C-element at any given time is defined in terms of the number of errors that need to occur at its current inputs in order for the C-element to enter an erroneous state. Sensitivities are computed through simulating a typical workload profile using standard event-driven simulators and recording the average time that each C-element spends in a sensitive state. The sensitivity of the circuit is then computed as the average time spent by the C-elements in sensitive states. Subsequently, several hardening techniques are proposed, based on duplication of the computational part and expansion of the C-elements, synchronization of linked channels when available, and synchronization using a redundant control circuit, when no linked channels are available.

While the aforementioned sensitivity analysis and hardening methods are effective in QDI circuits, they cannot be applied to ABMMs. First, sensitivity assessment in [7] is performed based on C-elements. ABMMs, however, do not have C-elements but employ combinational feedback instead. Second, ABMMs contain redundant logic, wherein transient errors may result in hazards but no functional discrepancy at the output [8]. Such hazards jeopardize the correct communication of the circuit with its environment. However, since this it not a concern in QDI circuits, such effects are not modeled in the sensitivity metric of [7]. Therefore, new methods for soft-error susceptibility analysis and hardening are required for ABMMs.

III. ABMMs

In this section, we briefly review the fundamentals of ABMMs, outline the synthesis process for realizing an ABMM implementation from a finite-state-machine description, and give an example (adapted from [8]).

A. Fundamentals

ABMMs constitute a class of *Huffman* circuits [12], which consist of a set of combinational functions, computing the next state and output of the circuit, and a set of feedback lines, storing the state of the circuit. No clock and no state registers are used in these circuits; however, delay elements are often added to eliminate *essential hazards*[1] [13]. Given the absence of a clock, *communication protocols* are needed to ensure correct interaction between an asynchronous circuit and its environment. These protocols define the properties of the stimuli (response) that the environment (circuit) is allowed to provide to the circuit (environment). Based on these protocols, various classes of asynchronous circuits are defined.

The key aspect of the protocol used in ABMMs, as indicated by their name, is that the interaction between the circuit and its environment happens in *bursts*. An *input burst* is defined as a set of bit changes in one or more inputs of the circuit, which are allowed to occur in any order and without any constraint in their relative time of arrival. Only after an input burst is complete, does the circuit respond to the environment through a *hazard-free* output change. We emphasize the protocol requirement for hazard-free output changes. Since no clock is used, synchronization is based on the fact that any change in the output of the circuit signifies completion of an evaluation cycle. Therefore, all hazards should be eliminated to ensure correct interaction of an ABMM with its environment.

ABMMs can be designed through burst-mode logic synthesis tools, such as MINIMALIST [14]. While most of those yield two-level ABMMs, some have also recently provided the capability to generate a multilevel implementation.

[1]Essential hazards arise when a state change completes before the input change is fully processed. To prevent this early state change from propagating through the combinational logic, delay may be added to the feedback.

Inputs: a, b, c, d

States	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
S_0	S_0,00	-	-	-	-	-	-	-	S_0,01	S_0,00	S_2,00	-	S_1,00	-	-	-
S_1	S_0,00	-	-	-	-	-	-	-	S_1,10	S_1,11	-	-	S_1,00	-	-	-
S_2	-	-	-	-	-	-	-	-	S_2,00	S_0,00	S_2,00	S_2,00	-	-	-	-

Outputs: x, w

Fig. 1. Example of a symbolic state transition table for defining an ABMM.

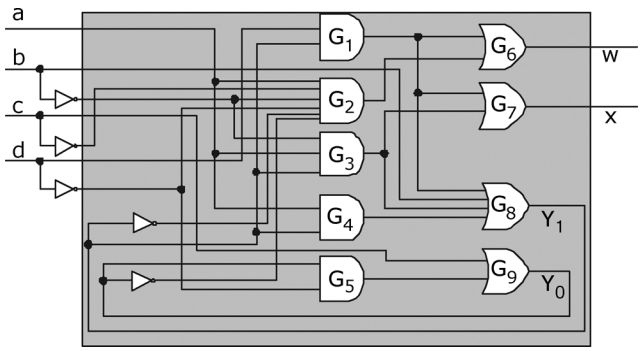


Fig. 2. ABMM implementation of the example.

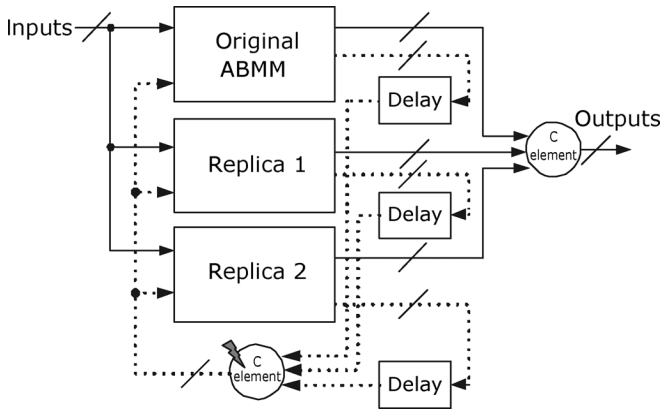


Fig. 3. TMR-based soft-error tolerance.

B. Example

An ABMM is described using a state transition table such as the one shown in Fig. 1. The rows in the table correspond to the current symbolic state, the columns correspond to the inputs, and each table entry indicates the next state and the outputs. For example, suppose that the circuit is in state S_0. Then, an input burst of 1010 will cause a transition to state S_2 and will generate an output of 00. Let us now assume that the next input burst is 1001, i.e., input c is lowered and input d is raised, and that c is lowered first and then d is raised, i.e., $1010 \rightarrow 1000 \rightarrow 1001$. The circuit responds only after the input burst is complete, so between the time that c is lowered and the time that d is raised, the next state and output bits do not change. Once the input burst is complete, the circuit makes a transition to state S_0 and computes the output, which, in this case, happens to remain the same, i.e., 00.

A dash in a table entry signifies that the corresponding combination of current state and input is not permitted by the communication protocol between the circuit and the environment. The synthesis process of MINIMALIST starts by solving the state encoding problem, which is modeled as a set of *dichotomies* [15] in order to derive a state encoding that allows a hazard-free implementation. Solving the dichotomies results in the state encoding $S_0 = 00$, $S_1 = 01$, and $S_2 = 10$ for the example circuit, and the symbolic states are replaced by their binary values. The last step is to generate a minimal-cost *hazard-free* implementation of the circuit [16]. Fig. 2 shows the resulting two-level ABMM, which includes some logic redundancy to ensure hazard-free operation.

IV. Soft-Error Tolerance in ABMMs

Toward designing soft-error-tolerant ABMMs, we first examine the applicability and effectiveness of the traditional TMR paradigm. As we discuss, TMR-based soft-error-tolerant ABMM design is not only overly expensive but also incomplete in terms of the provided soft-error tolerance. Then, building upon the method proposed in [6] for QDI circuits, we introduce a duplication-based method for designing soft-error-tolerant ABMMs. This method not only overcomes the limitations of TMR but also reduces the incurred area overhead, often even below the cost of previous concurrent error detection (CED) methods for ABMMs [8]. The proposed method is demonstrated using the running example of Fig. 2.

A. TMR-Based Soft-Error Tolerance

TMR employs three copies of a given circuit and a majority voter to decide the final output. Thus, any error(s) affecting only one of the copies is tolerated. As has been done for tolerating soft errors in both synchronous [17]–[19] and in asynchronous [6] circuits, the majority voter module can be substituted by a C-element. A C-element generates a rising (falling) transition when rising (falling) transitions have occurred on all of its inputs. Thus, when the inputs to a three-input C-element are nominally identical signals, a transient error in one of them will be suppressed and will not change the output of the C-element. The latter remains in its previous state until all three inputs to the C-element make the same transition, after which the output of the C-element follows.

The TMR-based soft-error-tolerant design method for ABMMs is shown in Fig. 3. The original circuit is triplicated, and C-elements are inserted at the state/output lines. When an SET strikes in any one of the three replicas, these C-elements prevent its effect from propagating to a state line or output. However, transient errors in the newly introduced C-elements cannot be tolerated. Specifically, a transient error that temporarily changes the state of a C-element driving an output may result in a hazard that can jeopardize communication of the circuit with its environment. Moreover, a transient error that

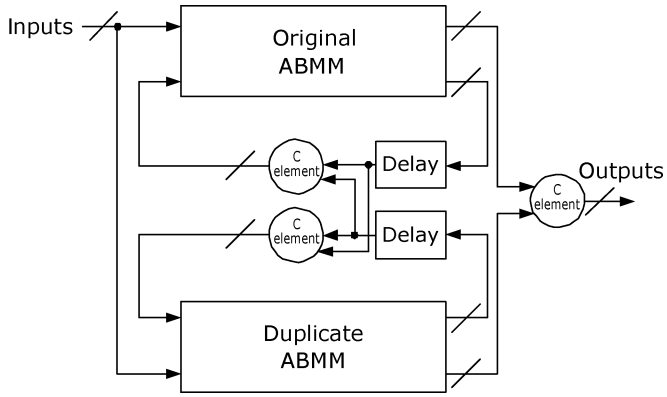


Fig. 4. Duplication-based soft-error tolerance.

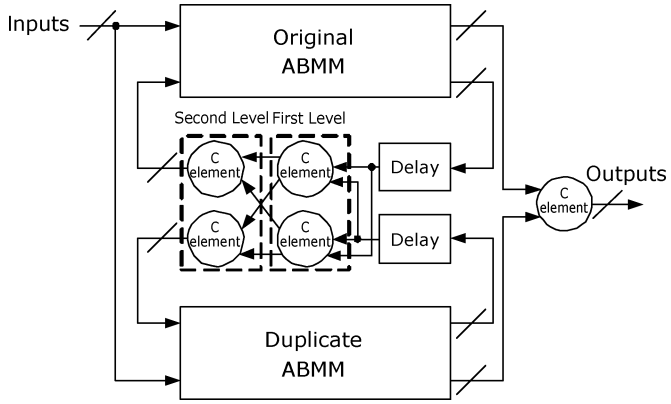


Fig. 5. Tolerating SETs on state-line C-elements.

temporarily changes the output of a C-element on a state line may propagate through the combinational feedback back to its inputs, as shown via the dotted lines in Fig. 3. Hence, all inputs of the C-element will agree on the erroneous value, forcing an incorrect permanent change in the state of the three copies and, by extension, the state/output of the circuit. In other words, an SET in a C-element driving a state line is far worse than an SET in a C-element driving an output since it results in a chain reaction of erroneous states, outputs, and, by extension, miscommunication between the ABMM and the environment. This limitation, along with the excessive cost incurred, makes TMR a rather nonappealing option for designing soft-error-tolerant ABMMs.

B. Duplication-Based Soft-Error Tolerance

In this section, we describe a duplication-based soft-error-tolerant design strategy that not only resolves the TMR problem of soft errors striking the C-elements on state lines but also incurs less area overhead. Unlike the combinational majority voter, a C-element is a sequential element that effectively *waits* until the error has ceased, which is permissible in an asynchronous implementation. In other words, even if two out of the three circuit copies in Fig. 3 produce soft errors, the C-element will suppress both of these errors. Essentially, this implies that tolerating single soft errors requires one replica only. This observation is the basis of duplication-based soft-error tolerance methods previously proposed for synchronous circuits [17]–[19] and for asynchronous circuits [6]. In the latter, a fine-grain duplication and double-checking approach is taken, as discussed in Section II. While applying this method to every gate in an ABMM would be economically infeasible, a coarse-grain variant at the state/output level is plausible, as shown in Fig. 4. A replica of the ABMM and one C-element for every pair of duplicate outputs is added, which ensures that soft errors in one ABMM copy do not reach the outputs. Moreover, a pair of C-elements is added to each state line (one for the original ABMM and one for the replica), which ensures that soft errors in one ABMM copy do not propagate back to the ABMM and change its state/output.

This design still does not address the problem of SETs striking the newly introduced C-elements. Strikes at output C-elements may result in hazards, causing miscommunication with the environment. Unfortunately, without changing the environment (e.g., to process two copies of the output signal), the last element driving the output is destined to be susceptible to SETs. An even more serious problem, however, has to do with SETs striking a C-element on a state line, which may propagate through the ABMM copy driven by this C-element back to its input and permanently change its state, as will be illustrated through an example in the next section.

To resolve this limitation, we enhance the duplication-based soft-error-tolerant design by adding a cross-coupled structure of four C-elements to state lines, as shown in Fig. 5. This structure prevents transient errors occurring in any of the state-line C-elements from resulting in an erroneous state being latched. More specifically, an error affecting a C-element in the first level of the cross-coupled structure is suppressed by the C-elements in the second level. Similarly, an error affecting a C-element in the second level of the cross-coupled structure may propagate through the one ABMM copy driven by this C-element but will not result in an erroneous latched state. The reason for this is that any state change will need to be agreed upon by both ABMM copies, i.e., the output value of the first level of C-elements will not change if its inputs mismatch. Thus, and once the transient error affecting the C-element in the second level disappears, its correct output value is restored.

In summary, the use of this cross-coupled structure allows the enhanced duplication-based soft-error-tolerant design to suppress all transient errors in the two ABMM copies and in the C-elements added to the state lines, making it more robust and less expensive than TMR.

C. Example

The duplication-based soft-error-tolerant ABMM design for the running example ABMM of Fig. 2 and the enhanced version for suppressing soft errors on state-line C-elements are shown in Fig. 6(a) and 6(b), respectively. Assume that the circuit is in state S_1 (encoded as 01) with an input of 1000. Then, as annotated in the implementation of Fig. 6(a), a transient error changing the state of the C-element that implements Y_1 from 1 to 0 would propagate through the top ABMM copy driven by this C-element back to its input. Therefore, the value of Y_1 in the top circuit copy will remain at 0, even after the transient error disappears, since the current inputs to the C-element have

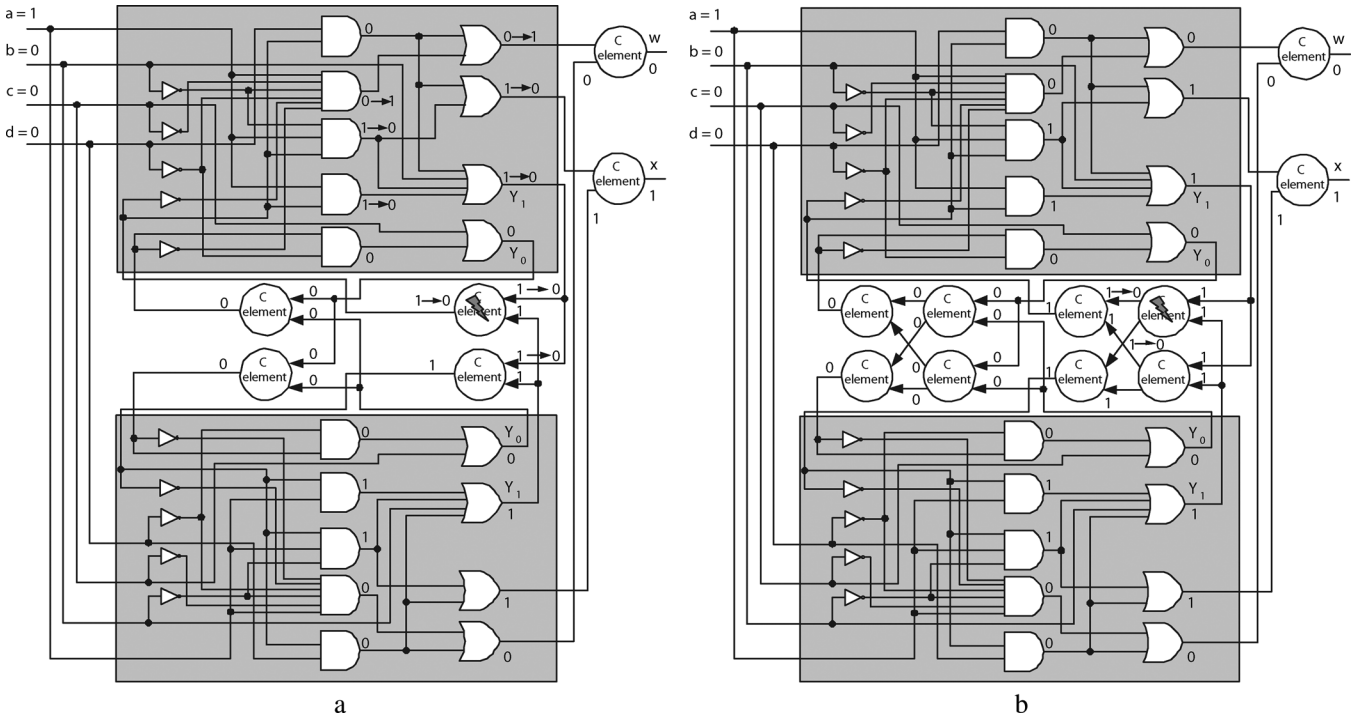


Fig. 6. Examples of duplication-based and enhanced duplication-based soft-error tolerance. (a) Duplication example for the circuit in Fig. 2. (b) Enhanced duplication for the circuit in Fig. 2.

now values of 0 and 1. On the other hand, the same transient error occurring in the enhanced implementation of Fig. 6(b) will change the state of the C-element to 0 but will be suppressed by the C-elements in the second level. Since both inputs to the affected C-element are 0, the erroneous state of the C-element will be corrected once the transient error disappears. One can also easily verify that an SET striking a C-element in the second level of the cross-coupled structure may propagate through one of the ABMM copies but will not reach the inputs of this C-element, as it is suppressed by the C-elements in the first level.

Assuming a rather expensive C-element implementation using three two-input NAND gates [20], the cost of the enhanced duplication design of Fig. 6(b) is three times the cost of the original circuit. In contrast, the cost of the TMR-based soft-error-tolerant design of Fig. 3 is 3.6× the cost of the original circuit and is less effective since it does not tolerate errors in C-elements on the state lines. For an apples-to-apples comparison, we note that if we substitute the cross-coupled structures of four C-elements that provide this additional robustness with single C-elements, the cost drops to 1.8× the cost of the original circuit, which is half the cost of TMR. We also note that the incurred overhead is 10% less expensive than that of the predominant CED method proposed in [8], despite the fact that the latter only provides detection.

V. Soft-Error Susceptibility Assessment

Despite its lower cost over TMR, many applications cannot afford the duplication-based soft-error tolerance method. Instead, there is a need for *partial* solutions that improve reliability to a target level at commensurate cost [3], [4]. Devising solutions that explore this tradeoff calls for the development of a *soft-error susceptibility assessment* method for ABMMs. Similar to methods for synchronous circuits, such soft-error susceptibility assessment should take into account the factors that prevent an SET from causing a soft error. In this section, we first describe the masking factors of SETs in synchronous combinational logic and contrast them to those in ABMMs. Then, we extend a previously developed fault simulator [9] to assess the potential of SETs in causing logic errors or hazards at the outputs of an ABMM. Finally, we describe how to compute the soft-error susceptibility and SER of an ABMM implementation.

A. Masking Factors

Three masking factors determine whether an SET in a synchronous combinational circuit will propagate to the output and result in a soft error [21]. First, there must exist a functionally sensitized path from the SET location to the output of the circuit; otherwise, the SET is *logically* masked. Second, the SET must create a pulse of sufficient amplitude that does not get attenuated by the electrical properties of successive gates before reaching an output; otherwise, the SET is *electrically* masked. Finally, the SET must appear at the output during the clocking window of the output flip-flops; otherwise, the SET is *latching-window* masked.

The specification and implementation properties of ABMMs lead to the exclusion of two of the aforementioned masking factors. ABMMs operate without a global clock, so latching-window masking does not apply to these circuits. Also, we opted to exclude electrical masking from our susceptibility analysis for the following two reasons. First, since ABMMs are high-performance controllers, their shallow paths provide minimal opportunity for electrical masking. Second, the effect of electrical masking is not as significant as the effect of logical masking, as

corroborated in a study by Boeing and SFA, Inc. [22]. The latter concludes that, while there is an observable effect, it cannot be generally assumed that electrical masking will significantly reduce the observed soft-error rate.

This leaves logic masking as the key mechanism for withstanding SETs in ABMMs.[2] In order to account for logical masking, a fault-simulation-based approach is necessary for assessing the soft-error susceptibility. Such fault simulation, however, should take into account that SETs in ABMMs may result not only in logic errors but also in hazards [8]. In order to identify these SETs, a fault simulator tailored to the particularities of ABMMs is needed.

B. Fault Simulation in ABMMs

In order to compute the soft-error susceptibility of ABMMs, we use SPIN-SIM [9]. SPIN-SIM is a logic and fault simulator that extends Eichelberger's classical hazard detection method, improves simulation accuracy through the use of a 13-valued algebra, maintains the relative order of causal signal transitions, and unfolds time frames judiciously. While SPIN-SIM was originally developed for speed-independent circuits, it was later extended [23] for delay-insensitive circuits through insertion of buffers to handle arbitrary delays on both gates and wires, as well as for QDI circuits through transformation of their *isochronic forks* into an equivalent speed-independent circuit form. For the purpose of this work, we further enhanced SPIN-SIM to simulate faults in ABMMs, so that it accounts for both functional discrepancies and hazards occurring due to SETs occurring in these circuits.

1) Simulation of asynchronous circuits: Unlike simulating synchronous circuits, there are generally a set of unique challenges in simulating asynchronous circuits, i.e., detecting hazards, handling timing constraints, and dealing with sequential behavior.

Hazard detection is critical in simulating an asynchronous circuit. Logic simulation algorithms developed for the synchronous domain are inadequate for hazard detection as latches/flip-flops mask their effect. The commonly used technique for hazard detection is multivalued logic, which has been employed in simulation of asynchronous circuits and test generation for path delay faults [24]–[28]. The 13-valued algebra has been particularly explored in SPIN-SIM since it can accurately describe signal transitions. Moreover, the 13-valued algebra is compact, and it avoids unnecessary event proliferation by abstracting the details of multitransition waveforms.

Simulating asynchronous circuits necessitates the handling of the timing constraints in them. Almost all practical asynchronous circuits operate correctly only based on certain timing constraints, such as the fundamental operation mode and isochronic forks, which must be taken into account during simulation. Various timing constraints may be handled by various simulation algorithms. One effective technique among them is to handle relative signal ordering with partially ordered multivalued algebras [9].

Moreover, asynchronous circuits often rely on combinational loops to perform sequential operations. As a result, signals through the feedback paths may even race with signals activated in previous iterations, producing sequential hazards. These autonomous behaviors challenge the traditional sequential simulation algorithm in an iteration by iteration fashion. Judicious time-frame unfolding techniques [9] are usually used to detect all sequential hazards.

2) Simulation of ABMMs: Burst-mode circuits operate with certain timing constraints. ABMMs assume arbitrary gate and wire delays in the combinational logic; therefore, signal ordering is unnecessary during the evaluation of the combinational logic. The 13-valued algebra suffices to detect combinational hazards under the arbitrary delay assumption. Moreover, the fundamental operation mode prevents signal racing across iterations. Therefore, the sequential behavior of the circuit can be simulated in an iteration-by-iteration fashion until a stable state is reached. The main steps of the simulation algorithm are described in Algorithm 1. Feedback loops are cut, with one end being marked as a pseudoprimary input (PPI) and the other as a pseudoprimary output (PPO). During the simulation, these are used in a similar fashion as the primary inputs (PI) and primary outputs (PO). The simulation iterates until no changes occur in the PPIs/PPOs, implying that the circuit has reached a stable state, or until a maximum number of iterations, (Max), has been reached, implying that the circuit is in oscillation.

Algorithm 1: Simulation of burst-mode machines

Identify and break feedback loops

$i \Leftarrow 0$; PI^0, $PPI^0 \Leftarrow$ initial values

repeat

evaluate PO^i, PPO^i using 13-valued algebra

$i \Leftarrow i + 1$

$PPI^i \Leftarrow PPO^{i-1}$

$PI^i \Leftarrow$ final state of PI^0

until $PPI^i = PPI^{i-1}$, or $i = Max$

C. Soft-Error Susceptibility Computation

Using the enhanced fault simulation capabilities of SPIN-SIM, we can now examine the impact of SETs in an ABMM and quantify the susceptibility of individual gates and the SER

[2]With two of the three masking factors excluded from significantly reducing the soft-error susceptibility of ABMMs, one might think that ABMMs are more soft-error prone than synchronous designs, making the need for developing soft-error tolerance and mitigation solutions for such circuits even more imperative. Indeed, since ABMMs operate without a global clock, they are susceptible to any SET that affects the correctness of its response at any point in time. Synchronous designs, on the other hand, are susceptible to SETs that arrive during the latching window of the memory elements only. However, synchronous controller designs may have more sequential elements (latches or flip-flops) than ABMMs since the latter store the state in the combination feedback. Such memory elements are typically much more susceptible to SETs than logic. In addition, signal interference (another source of transient errors) is typically lower in asynchronous circuits than in synchronous circuits [5], possibly reducing further the SET susceptibility of ABMMs. In short, the jury is still out on this, and a comparative study between the susceptibility of an asynchronous circuit and its synchronous counterpart is necessary before a conclusion can be drawn on the immunity of each design style to SETs.

TABLE I
SOFT-ERROR SUSCEPTIBILITY TABLE

State & Input Burst Pairs (SIB)	Potential SETs			
	f_1	f_2	...	f_p
SIB_1	11...0	01...1	...	00...0
SIB_2	01...0	11...1	...	00...1
⋮	⋮	⋮	⋮	⋮
SIB_m	01...1	00...0	...	01...0

of the circuit. Toward this end, we construct the soft-error susceptibility table illustrated in Table I. The rows in the table correspond to the combinations of state and input bursts (SIBs) that are allowed by the communication protocol of the ABMM with its environment. The columns represent potential SETs[3] in the circuit. Each table entry contains a bit string that reflects the output and state lines of the ABMM that are affected when an SET occurs during a SIB. A value of 1 (0) in a bit of this bit string implies that the corresponding output or state line is erroneous (correct). The table is constructed through fault simulation of all possible SETs over the entire input space of the ABMM. We note that, unlike synchronous circuits where exhaustive simulation of all possible input patterns is prohibitive, ABMMs only have a much smaller set of permitted SIBs in their protocol, which allows quick construction of the table.

Once the table is constructed for an ABMM with n gates, the susceptibility of each gate is computed as follows. Assume that the table is stored as an $m \times p$ matrix $sest$. Let k_q denote the total number of possible SETs in gate G_q, where $q \in [1, \ldots, n]$, and let $sest[i,j]$ denote the (i,j)th entry of the soft-error susceptibility table for all $i \in [1, \ldots, m]$, $j \in [1, \ldots, p]$. Also, let $E(sest[i,j])$ be a function that returns a 1 (0) if any (none) of the output and state bits in $sest[i,j]$ is 1, i.e., if the combination of a SIB and an SET results in an error (or not). Then, the susceptibility of G_q is defined as

$$susc(G_q) = \frac{\sum_{i=1}^{m} \sum_{j=s+1}^{s+k_q} E\left(sest[i,j]\right)}{m \times k_q}, \qquad s = \sum_{l=1}^{q-1} k_l \tag{1}$$

and the SER of the ABMM is defined as

$$SER(ABMM) = \sum_{q=1}^{n} susc(G_q). \tag{2}$$

Essentially, the susceptibility of a gate reflects the percentage of SIB and SET combinations that produce an observable error at an output or a state line of the ABMM. By extension, the SER of the ABMM reflects its vulnerability to SETs.

[3]Potential SETs affecting the initialization circuitry (i.e., the $reset$ and $reset\_bar$ signals) are accounted for in the construction of the soft-error susceptibility table, and hence, the proposed soft-error susceptibility computation method models all SETs that may appear during the *operation* of the circuit. However, transient errors appearing during initialization (i.e., while the next state and output functions are forced to their initial values) may prevent the correct initialization of the circuit. Nonetheless, and since the $reset/reset\_bar$ signals are connected to many gates, only a particle strike with significant charge would alter the value of these signals (similar to a transient error affecting the clock network in synchronous designs). Hence, the high load of these signals provides naturally an increased immunity to most potential SETs during initialization.

VI. SOFT-ERROR MITIGATION IN ABMMS

Based on the susceptibility assessment capability of the previous section, we devise a soft-error mitigation solution for ABMMs. The proposed method is based on partial duplication and aims to explore the tradeoff between area overhead and soft-error susceptibility reduction by judiciously selecting and replicating individual gates, complete state/output logic cones, or partial state/output logic cones. The three alternative selection methods are presented herein and illustrated using the example of Fig. 2. Finally, we discuss how these methods enable integration of soft-error susceptibility into a design-space exploration framework, which may include not only area but also other design constraints such as performance, power consumption, and offline testability.

A. Duplication of Sensitive Gates

Due to the asymmetric susceptibility [1], gates at the second level of an ABMM are significantly more susceptible to transient errors than gates at the first level. This observation reveals an opportunity for reducing duplication overhead by replicating only gates that have high soft-error susceptibility. In order to preserve the functionality of the partial replica, however, signals from the nonduplicated gates in the original ABMM need to drive some gates in the partial replica. As a result, transient errors affecting shared gates will affect both ABMM copies and will not be suppressed; thus, the cost reduction comes at a loss of transient-error tolerance. Yet, due to the asymmetry in susceptibility, judicious selection can lead to a favorable outcome.

Selection of gates to be replicated commences with construction of the duplication-based soft-error-tolerant ABMM, as described in Section IV-B. Then, the susceptibility of each gate in the original ABMM is computed using (1). Finally, gates at the first level of the duplicate ABMM are removed in an increasing order of susceptibility. Every time a gate is removed, its fan-outs in the partial replica are driven by the corresponding gate in the original ABMM. Accordingly, the overhead and the soft-error tolerance of the circuit are reduced by the cost and the susceptibility of the removed gate, respectively. The process is repeated until a target area overhead constraint is satisfied or no more first-level gates are left to remove.

B. Duplication of Sensitive Complete Logic Cones

In contrast to the previous method, which exploits the asymmetric susceptibility of gates in different levels of an ABMM circuit, this method builds upon the asymmetric susceptibility of state/output logic cones to transient errors. In essence, it selects a subset of state/output cones that meets an area target and whose replication maximizes the number of tolerated pairs of SIBs and SETs in Table I. This problem can be formulated as an ILP. Assume that the ABMM has m SIBs, p SETs, and r state/output lines, denoted by $\{x_1, x_2, \ldots, x_r\}$. Let any subset of state/output logic cones be represented by an r-dimensional 0–1 vector denoted Y_k, and let its implementation cost be C_k, $1 \leq k \leq 2^r - 1$. For example, the subset $\{x_1, x_2, x_4\}$ is represented as [1 1 0 1] and denoted by Y_{13}, and the cost of implementing the state/output functions x_1, x_2, and x_4 is denoted by C_{13}. Also, let

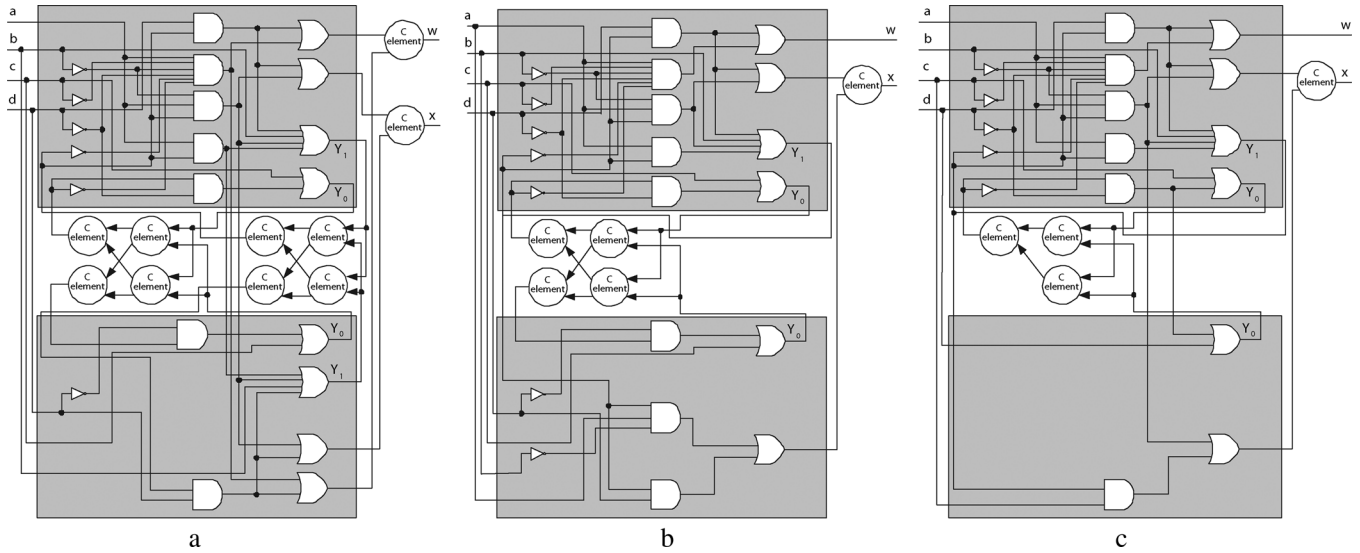


Fig. 7. Examples of circuit implementations produced using the mitigation methods. (a) Duplication of sensitive gates. (b) Duplication of complete cones of logic. (c) Duplication of partial cones of logic.

$V(sest[i,j])$ be the r-dimensional vector constructed from the r bits in $sest[i,j]$. We define function $Tol(Y_k,i,j)$ as follows:

$$Tol(Y_k,i,j)=\begin{cases}1, & \text{if } \overline{Y_k}\cdot V^{\mathsf{T}}(sest[i,j])=0\\ 0, & \text{if } \overline{Y_k}\cdot V^{\mathsf{T}}(sest[i,j])>0\end{cases} \quad (3)$$

where $\overline{Y_k}$ is the binary complement of Y_k, $\cdot$ is the dot multiplication operation, and T is the transpose operation. $Tol(Y_k,i,j)$ returns a 1 if and only if Y_k tolerates SET j occurring during SIB i (i.e., if the transient error propagates to the state/output logic cones in Y_k). The following ILP formulation finds the state/output subset Y_k that maximizes the number of tolerated entries in Table I for a given area overhead constraint $(COST)$:

$$\texttt{Maximize}\sum_{i=1}^{m}\sum_{j=1}^{p}Tol(Y_k,i,j)$$
$$\texttt{subject to}:$$
$$\text{(a) } C_k\le COST$$
$$\text{(b) } x_s\in\{0,1\}$$
$$\text{for all s}:\ 1\le s\le r.$$

While ILP is NP complete, it can be efficiently approximated through a well-known method combining linear program relaxation and randomized rounding [29] in order to obtain near-optimal solutions.

C. Duplication of Sensitive Partial Logic Cones

The third partial duplication method aims to combine the first two and leverages the asymmetric susceptibility of both gates and state/output logic cones. In other words, it explores solutions that include a subset of *partial* state/output logic cones. Similar to the first method, in order to preserve the functionality of the partial replica, the fan-outs of the missing gates in these partial cones are driven by the corresponding gates in the original ABMM.

Our algorithm starts by solving the ILP of Section VI-B for a higher $COST$ value than the targeted area cost $(COST_{\text{target}})$. Then, the state/output cones returned by the ILP are pruned by applying the method of Section VI-A until $COST_{\text{target}}$ is met, in which case the partial state/output cones and the corresponding soft-error tolerance are recorded. $COST$ is then increased,[4] and the process is repeated until all cones are included in the ILP solution, at which point the best recorded solution is reported.

D. Examples

Fig. 7 shows instances of circuits produced by the proposed soft-error mitigation method for the ABMM example of Fig. 2. For the first partial duplication option, all second-level gates and the corresponding C-elements appear in the replica in Fig. 7(a), but some of them are driven from the original ABMM due to removal of first-level gates in the replica. For the second partial duplication option, only some second-level gates and the corresponding C-elements appear in the replica in Fig. 7(b), along with their complete cone of logic, which eliminates the need for tapping signals from the original ABMM. For the third partial duplication option, only a subset of second-level gates and the corresponding C-elements appear in the replica in Fig. 7(c), but also, only a subset of their cones of logic is replicated, creating the need for signal tapping from the original ABMM. In comparison to the duplication-based ABMM design shown in Fig. 6(b), the implementations in Fig. 7(a)–(c) require 87%, 60%, and 50% of its cost while providing 68%, 47%, and 24% of its transient-error tolerance, respectively.

E. Design-Space Exploration

In the three partial duplication methods discussed previously, the key objective driving the corresponding search algorithms is the maximization of the circuit's ability to withstand soft errors. During this search, $COST$ has so far been used as a synonym

[4]In our experiments, we start with $COST=COST_{\text{target}}+1\%$, and we increment $COST$ by 1% in each iteration.

with **area overhead** incurred by the proposed soft-error mitigation solution, without consideration to the impact on performance, power consumption, and offline testability of the produced design. Even under this simplistic assumption, the proposed soft-error mitigation methods can be used to select the best circuit instance for the following three objectives.

1) Identify the circuit instance attaining the maximum soft-error mitigation capability for a given area overhead.
2) Identify the circuit instance incurring the lowest area cost for a target level of soft-error mitigation.
3) Identify the circuit instance that optimizes the return on investment, i.e., the ratio of attained soft-error mitigation over incurred area overhead, or any other similarly defined metric.

While the problem formulation given in the previous sections reflects the first objective, we describe in the sequel how the latter two objectives are supported. A close look at the description of the algorithm of Section VI-C, along with the optimization objective in the ILP formulation of Section VI-B, reveals that the search yields a list of circuit instances that are sorted in a monotonically increasing order with respect to their area overhead and soft-error mitigation level. Ultimately, however, a single circuit instance is selected.

To satisfy the first objective, among all circuit instances that do not exceed the target area overhead, we select the one that maximizes soft-error mitigation; however, there may exist another instance offering just slightly lower soft-error mitigation yet at significantly lower cost. Similarly, to satisfy the second objective, among all circuit instances that exceed the aimed soft-error mitigation level, we select the one with the lowest area overhead; however, there may exist another instance incurring just slightly higher area overhead yet providing significantly higher soft-error mitigation. The third objective alleviates these concerns by combining soft-error mitigation and area overhead in a single metric and comparing circuit instances across both dimensions. Hence, we select among all circuit instances the one that maximizes the return on investment, as expressed through the ratio of soft-error mitigation over area overhead.

Furthermore, the problem formulation can be easily extended to other design parameters. Specifically, performance, power consumption, and offline testability may also be seamlessly integrated in the search framework along with area and soft-error mitigation, thus enabling exploration of the various design-space tradeoffs [3]. In other words, each of the circuit instances produced by the search algorithm can be characterized and annotated with the corresponding design parameters. Hence, constraints on all of these parameters can be used to define various objectives, based on which the optimal circuit instance may be selected.

VII. Logic Masking Optimization Through Gate Decomposition in Multilevel ABMMs

In order to satisfy the constraints that are necessary to produce an ABMM, such as the hazard-free property, most existing synthesis packages generate a two-level implementation. When mapping the circuit to a specific technology, however, each of the gates in these two levels may need to be decomposed (in a hazard-nonincreasing manner) into multiple levels to satisfy the fan-in limitations of the target library. Such decomposition may offer opportunities for further improvement of various design aspects. Among those, we examine the impact of gate decomposition on the ability of an ABMM to tolerate soft errors. Specifically, we first show that the distribution of input signals of a decomposed gate to its constituents affects its ability to logically mask errors. Then, we formulate the problem of optimally assigning signals to gates as an ILP, and we describe the overall method used to minimize soft-error susceptibility while decomposing a two-level ABMM into a multilevel equivalent.

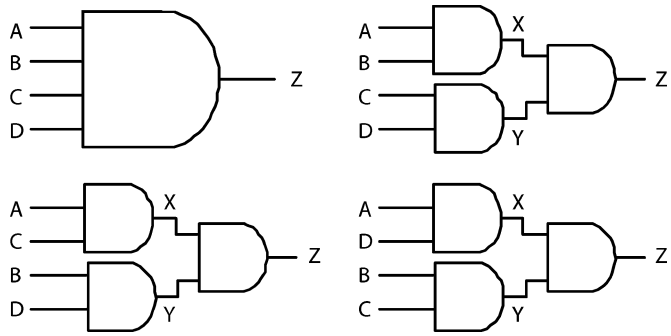


Fig. 8. Four-input AND gate and its decomposed structures using two-input AND gates.

TABLE II
Transition List and TSM for a Four-Input AND Gate

#	(ABCD) → (ABDC)$_{next}$	TSM			
1.	0000 → 1100	0	0	1	1
2.	1100 → 1111	0	0	0	0
3.	1111 → 0010	0	0	0	0
4.	0010 → 0110	1	0	0	1
5.	0110 → 0000	1	0	0	1

A. Impact of Gate Decomposition on Soft-Error Tolerance

When decomposing a gate, there are many choices as to how the input signals are distributed to the constituent gates. The *key observation* is that each of these choices offers a different level of soft-error tolerance. Specifically, while all of them will mask the same number of errors occurring outside the gate, each one will mask a different subset of errors striking the gates within the decomposed structure. Hence, judicious distribution of the input signals can enhance the ability of the decomposed structure to withstand soft errors.

Example: Consider the four-input AND gate shown in Fig. 8, and let us assume that it is part of a two-level ABMM and that the transitions that it goes through based on the ABMM definition are the ones shown in Table II. The last column provides its transition stability matrix (TSM), which indicates whether, during a transition, an input will retain a controlling value. For example, during the first transition, inputs A and B change from 0 to 1, which is not a controlling value for the AND gate, while inputs C and D remain at 0, which is a controlling value, so the corresponding TSM entry is 0011. Let us also assume that we are mapping the ABMM to a library that only offers AND gates with a maximum of two inputs, which implies that we need to use three two-input gates, and that, in the interest of maintaining speed, we choose a balanced-tree implementation of depth two. As shown in Fig. 8, there are three options for distributing the

four inputs A, B, C, and D to the two two-input AND gates at the first level of the decomposed structure. In larger gates, after assignments are made at the first level, more options may occur in the subsequent levels. For each of the five transitions, any of the three gates may suffer a transient error that will flip its value, so we have a total of 15 possible errors. Out of these, the decompositions (AB)(CD) and (AD)(BC) will mask 7, while the decomposition (AC)(BD) will mask 8 and is therefore a better option.

In order to maximize the ability of the decomposed structure to suppress such errors, we propose an *algorithm that distributes the input signals in a way that maximizes the number of transitions for which two or more gates have at least one controlling value as an input*. In any transition where this is the case, an error at a single gate in the level being optimized will be masked, as there will always be at least one other controlling value that is present at the inputs of the following level. For example, in the first transition of Table II, inputs C and D remain 0, while inputs A and B both transition from 0 to 1. Thus, in the (AB)(CD) network shown in Fig. 8, a soft error causing Y to become 1 will propagate all the way to the output. On the other hand, in the network with structure (AC)(BD), a strike at either gate X or Y will not affect the output because they both contain controlling values.

B. Problem Formulation

To perform the aforementioned process of optimally allocating the input signals of a decomposed gate to its constituents, we formulate the problem as a set of binary ILP constraints and an objective function to be maximized. We represent connections between signals and gates as a set of binary variables x_{ig}. The value of x_{ig} is 1 when input i is connected to gate g and 0 otherwise. If we are assigning I inputs to G gates that have a fan-in limit of F, then we first need to generate two sets of constraint equations for the ILP, i.e., one to ensure that each input is assigned to exactly one gate, and one to ensure that no more than F inputs are assigned to any gate

$$\sum_{g=1}^{G} x_{ig} = 1 \qquad \forall i \in [1, \ldots, I] \tag{4}$$

$$\sum_{i=1}^{I} x_{ig} \leq F \qquad \forall g \in [1, \ldots, G]. \tag{5}$$

The next two sets of constraints allow us to calculate the number of transitions for which this level has two or more gates with one or more controlling values as inputs. To do this, we use the TSM values, such as the ones shown in Table II, where the columns represent the I inputs and the rows represent the T transitions. If we denote the TSM values as TSM_{it}, then TSM_{it} is 1 if input i is a controlling value during transition t and 0 otherwise. The set of equations described by (6) uses the TSM to determine whether gate g has one or more controlling values during transition t. If so, then the binary variable P_{gt} is set to 1; otherwise, it is 0

$$\sum_{i=1}^{I} TSM_{it} \cdot x_{ig} \geq P_{gt} \qquad \forall (g,t) \in [1, \ldots, G] \times [1, \ldots, T]. \tag{6}$$

Thus, in a given instance of (6), the ILP can only set P_{gt} to 1 when the total number of inputs that are both connected to gate g and have controlling values during transition t is greater than or equal to 1. The next set of equations uses these P_{gt} values to calculate the number of transitions for which two or more gates at the level being optimized have at least one controlling input

$$\sum_{g=1}^{G} P_{gt} \geq 2P'_t \qquad \forall t \in [1, \ldots, T]. \tag{7}$$

These equations determine the values of the binary variables P'_t, which are 1 when the level being optimized has at least two gates with one or more controlling values during transition t. Finally, we generate the objective function, which calls for the maximization of the sum of all the P'_t variables

$$Maximize : \sum_{t=1}^{T} P'_t. \tag{8}$$

The ILP instances that we are dealing with here are rather small and can be either solved explicitly or approximated using randomized rounding [29]. This process is carried out sequentially for each level in the decomposed structure, starting at the one closest to the inputs. Each time that assignments are made at a given level, the TSM for the next level is generated so that the next ILP can be formulated, and the process is repeated until the output is reached.

VIII. Production Testing of Soft-Error-Tolerant ABMMs

In this section, we first discuss the adverse impact that a soft-error-tolerant design method may have on the offline testability of a circuit (whether synchronous or asynchronous). Then, we describe the unique opportunity that the use of C-elements in the proposed soft-error-tolerant ABMMs offers for offline testing through input sequences that halt the circuit, as well as a possible method for generating these test sequences.

Production testing aims to weed out chips with *permanent* defects incurred during the manufacturing process, by applying input sequences to which the faulty and fault-free chips respond differently at their outputs. In a soft-error-tolerant design, however, the purpose of the added hardware is to ensure that the circuit continues to operate correctly in the presence of an error, thereby suppressing it before it reaches the output. Evidently, the objectives of testing and soft-error tolerance are contradictory, and it is well known that a soft-error-tolerant design method may jeopardize the testability of a circuit. Consider, for example, a permanent fault in a TMR implementation of a synchronous circuit. While this fault may cause an erroneous response at the output of one of the three replicas, the outputs of the remaining two will prevail, and the error will be suppressed by the voter. Hence, faults in the TMR implementation become untestable; yet, while the TMR faulty circuit will function correctly, its ability to withstand errors will be diminished. The typical solution to address this problem is by inserting design-for-testability (DFT) hardware in the form of controllability/observability points and/or conversion of memory elements to scan chains. In TMR, for example, multiplexing the outputs of the three replicas with the output of

the voter and selecting which one to observe would resolve the problem. On the other hand, such addition of DFT logic incurs an area overhead and possible performance degradation, so it should be used judiciously.

Testability considerations in ABMMs have been previously investigated in [30]. As mentioned in Section III-A, ensuring the hazard-free property of the state/output functions in an ABMM calls for some form of logic redundancy in the original design, which could possibly result in a number of faults being untestable. Nevertheless, as detailed in [30], hazard-free machines that are fully testable with regard to both stuck-at and delay faults can be obtained by appropriately constraining the synthesis process. Hence, our discussion focuses on faults that are testable in the original ABMM, i.e., there exists a test sequence that, in the presence of this fault, will cause an incorrect transition at a state/output line.

Consider now the addition of the hardware that is necessary for soft-error tolerance, as shown in Fig. 4. Given the previous observation on the contradictory objectives of testability and error tolerance, one might expect that the proposed soft-error-tolerant ABMM implementation will result in a large percentage of these faults becoming untestable, unless DFT hardware is employed [31], [32]. Nonetheless, the use of two copies and a C-element, instead of three copies and the voter that is used in a TMR implementation of a synchronous circuit, offers an opportunity to retain, at least partially, the testability of the ABMM, without the addition of DFT logic. Specifically, unlike a TMR voter, where two correct replicas suffice to control the output and mask the error of the third replica, the output of the C-element retains its previous value until both of its inputs make a transition. Hence, if a permanent fault is excited and reaches the outputs of the original ABMM, then it will cause a discrepancy with the output of the replica, and therefore, the driven C-element will halt indefinitely in its previous state. If the previous state is set to the opposite value, such that a transition should occur in a fault-free circuit, the absence of this transition will signify the existence of a fault. Of course, this raises the question of how long one should wait prior to calling the fault. While the lack of a clock blurs the line of a computational cycle, it is common practice in asynchronous circuit testing to define an upper bound after technology mapping and rely on circuit halting for fault detection.

Example: Consider the circuit in Fig. 2 and assume that the circuit is initially in state S_0 (encoded as 00) with an input of 0000 and an output of 00. Then, a permanent stuck-at-1 fault at the output of G_2 sets output w to the incorrect value of 1, and the fault is testable in the original design, with no further input vectors needed. In the soft-error-tolerant ABMM version, however, and under the standard single-fault assumption, one of the two copies (for example, the original) is faulty and the other (for example, the replica) is fault free, so w in the fault-free circuit $(w_{\text{fault-free}})$ is different from w in the faulty circuit (w_{faulty}). Therefore, the output of the C-element driven by $w_{\text{fault-free}}$ and w_{faulty} is equal to its previous state (i.e., 0), which is identical to the fault-free state. Hence, the stuck-at-1 fault on w_{faulty} does not propagate through the C-element and cannot be observed at the output using the test vector generated for the stand-alone original machine. However, if we first apply the test vector 1000, both $w_{\text{fault-free}}$ and w_{faulty} are set to 1, setting the output of the C-element driven by these two signals to 1. Then, applying the test vector 1001 forces $w_{\text{fault-free}}$ to 0, while w_{faulty} remains at 1 due to the stuck-at fault. At this point, the C-element driven by $w_{\text{fault-free}}$ and w_{faulty} retains its previous state (i.e., 1), which is different from the expected fault-free response. Hence, the fault is testable since it prevents an expected transition from occurring at the output of the circuit (i.e., the fault halts the circuit).

A key limitation in leveraging this testability opportunity offered by C-elements is the pronounced lack of native-mode asynchronous test generation algorithms and tools, as well as the fact that their synchronous domain counterparts are not geared toward such mode of detection. Nevertheless, in order to obtain at least some quantitative information as to how effectively the soft-error-tolerant ABMMs can be tested, we extended the capabilities of SPIN-TEST [10], a previously developed test generation algorithm for the class of speed-independent circuits. Based on SPIN-SIM [9] and its extensions to handle ABMMs, which we described in Section V-B, SPIN-TEST can now find test sequences for faults in the original ABMM and can take advantage of halting to detect faults in the soft-error-tolerant ABMM.

We note that SPIN-TEST is a fault-simulation-based test generation algorithm, so it relies on objective-driven perturbation of pseudorandom input patterns to identify the appropriate test vectors. Such algorithms are very fast in identifying patterns for a large percentage of faults yet have a hard time to generate patterns for the random-pattern-resistant faults. Furthermore, they cannot prove whether a fault is untestable since it is impossible to simulate exhaustively the complete input space. Even with these limitations, SPIN-TEST is able to generate test sequences that retained the ability to test a very high percentage of faults. The remaining faults are either hard to test (i.e., using the current test generation procedure of SPIN-TEST) or untestable. The attained fault coverage level may be further improved by adding a deterministic test generation phase to SPIN-TEST or by adding DFT to target the remaining untestable faults. We also note that the ability to assess fault coverage in an ABMM with soft-error mitigation hardware would enable integration of testability in the design-space exploration framework [3] described in Section VI-E.

IX. Experimental Results

The proposed soft-error tolerance and mitigation methods were applied on a standard suite of 13 benchmark circuits. The circuits are first synthesized using MINIMALIST [14] to generate an ABMM implementation. Then, the TMR- and duplication-based soft-error-tolerant implementations are constructed, as described in Section IV. Next, the soft-error susceptibility table of the original ABMM is generated using the enhanced version of SPIN-SIM [23], as discussed in Section V-C, and the partial-duplication-based soft-error mitigation solution described in Section VI is applied. In Section IX-A, we compare the results of the proposed duplication-based soft-error tolerance method to TMR and CED methods for ABMMs. Then, in Section IX-B, we present and compare the results of the three partial duplication options of Section VI for two-level and multilevel ABMMs.

TABLE III
EXPERIMENTAL RESULTS FOR DUPLICATION-BASED SOFT-ERROR TOLERANCE

Circuit		Original		C-elements			Total		TMR		CED [9]		Performance	Testability
Name	I/S(Bits)/O	Lit.	Gates	#	Lit.	Gates	Lit.	Gates	Lit.	Gates	Lit.	Gates	Overhead	
hp-ir	3/2(1)/2	13	8	6	36	18	**62**	**34**	75	42	102	57	46.15%	66.67%
martin-q-element	2/2(1)/2	14	9	6	36	18	64	36	78	45	**59**	**31**	42.86%	70.00%
tangram-mixer	3/2(1)/2	17	10	6	36	18	**70**	**38**	87	48	78	44	35.29%	100%
concur-mixer	3/3(2)/3	26	16	11	66	33	**118**	**65**	138	78	152	85	38.46%	90.16%
while	4/3(2)/3	27	16	11	66	33	120	65	138	78	**104**	**56**	37.04%	94.74%
while_concur	4/4(2)/3	41	24	11	66	33	**148**	**81**	183	102	158	88	24.39%	90.16%
opt-token-distributor	4/6(3)/4	74	41	16	96	48	244	130	306	165	**201**	**109**	18.92%	68.60%
rf-control	6/6(3)/5	75	37	17	102	51	252	125	321	159	**224**	**121**	21.33%	98.10%
pe-send-ifc	5/5(3)/3	110	58	15	90	45	**310**	**161**	402	210	383	211	10.91%	100%
barcode	13/11(4)/17	327	172	33	198	99	**852**	**443**	1233	642	1022	547	12.84%	100%
diffeq	14/9(4)/20	345	189	36	216	108	**906**	**486**	1323	711	1019	556	13.91%	92.21%
p2	8/13(4)/16	349	192	32	192	96	890	480	1287	696	**818**	**436**	11.46%	99.80%
p1	13/11(4)/14	458	238	30	180	90	**1096**	**566**	1590	822	1116	599	7.86%	62.10%

A. Soft-Error Tolerance Results

The results for duplication-based soft-error tolerance are presented in Table III, including the following details of the circuits that were used: name, number of inputs (I), number of states (S), number of state bits (Bits), and number of outputs (O). The fourth major heading summarizes the total literal and gate count of the soft-error-tolerant ABMM. While for small circuits (e.g., $hp\text{-}ir$ and $tangram\text{-}mixer$) the area overhead may seem excessive (i.e., over 300%), we raise caution that this cost is significantly inflated due to the proportionately large number of C-elements over logic gates. Indeed, in larger circuits, such as $p2$ and $p1$, this proportion changes, and the percentile overhead reduces drastically (i.e., less than 150%). Thus, we anticipate the area overhead to be even lower for larger and more complex ABMMs. More importantly, assessing the overhead of the proposed duplication-based soft-error-tolerant ABMM design method should not be done in absolute terms but rather in comparison to the best known alternative for these circuits. The area cost of TMR and the minimum-cost CED method in [8] are summarized in the fifth and sixth major headings in Table III, respectively. For each circuit, the solution with the lowest area cost is shown in boldface. The area cost of duplication-based tolerance is, on average, 24% less than that of TMR. We also note that, for 8 out of the 13 benchmark circuits, duplication-based soft-error tolerance incurs lower overhead, even in comparison to the CED methods in [8].

With respect to performance, the duplicate circuit utilized in the soft-error tolerance approach, or the partial duplicate in the mitigation methods, operates in parallel with the original circuit. However, and similar to that when majority voters are employed in a typical soft-error-tolerant TMR design, the addition of C-elements at the output/state functions increases the delay of the circuit and equivalently reduces the performance of the controller. The performance of the soft-error-tolerant ABMM is compared to that of the original ABMM under the sixth major heading in Table III.[5] The results indicate that the soft-error-tolerant ABMM is, on average, 25% slower than the original ABMM. Similar to the previous observation on the area overhead of soft-error-tolerant ABMMs, the performance overhead reduces for larger and more complex ABMMs.

[5]We note that the performance overhead of any circuit produced by the mitigation methods is upper bounded by the performance overhead of the soft-error-tolerant implementation reported in Table III.

The last column in Table III summarizes the fault coverage obtained by the modified version of SPIN-TEST [10] on the soft-error-tolerant ABMM. The results indicate that an average of at least 87% of all single stuck-at faults remain testable without the use of any DFT circuitry.

B. Soft-Error Mitigation Results

In this section, we first present the results of the three partial duplication options of Section VI for two-level ABMMs, followed by the results for multilevel ABMMs. Then, we compare the area overhead and soft-error susceptibility reduction in two-level and multilevel ABMMs, and we discuss cost-efficient tradeoff points.

1) Two-level ABMMs: In Fig. 9, we show the reduction in the soft-error susceptibility achieved by the proposed soft-error mitigation methods on several two-level benchmark circuits: method 1 (M_1) for duplication of sensitive gates, method 2 (M_2) for duplication of sensitive complete state/output cones, and method 3 (M_3) for duplication of sensitive partial state/output cones. The results are presented with respect to the targeted area overhead of the mitigation logic, where 100% reflects the cost of the complete duplication-based soft-error tolerance method.

Several observations are supported by these results. First, the reduction in soft-error susceptibility is commensurate with the incurred area overhead. This can be inferred from the linearly monotonic shape of the plots in Fig. 9 until the limit of 100% susceptibility reduction is reached at the cost of complete duplication. Second, M_3 is able to yield mitigation logic implementations for very low targets of area overhead, which neither M_1 nor M_2 can achieve. This can be observed on the lower left corner of the plots in Fig. 9, where the leftmost point is always a triangular shape. Finally, M_3 always yields a mitigation logic implementation that achieves higher soft-error susceptibility reduction at lower area overhead, as compared to M_1 and M_2. This is expected since M_3 was developed by combining M_1 and M_2.

2) Multilevel ABMMs: Fig. 10 compares the reduction in the soft-error susceptibility achieved by the proposed soft-error mitigation method on two-level and multilevel benchmark circuits. The results indicate that the mitigation methods in multilevel ABMMs always yield a logic implementation that achieves a higher soft-error susceptibility reduction than that in two-level ABMMs. In a multilevel ABMM, the decomposition of a large

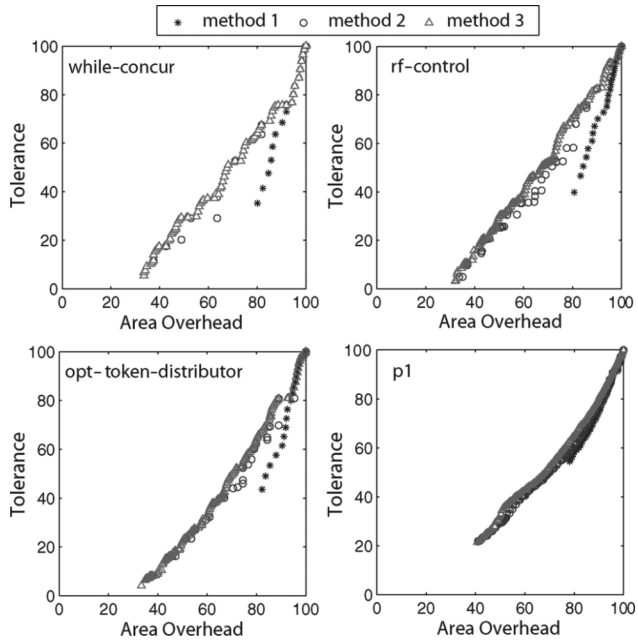


Fig. 9. Tradeoff between area cost and soft-error susceptibility reduction.

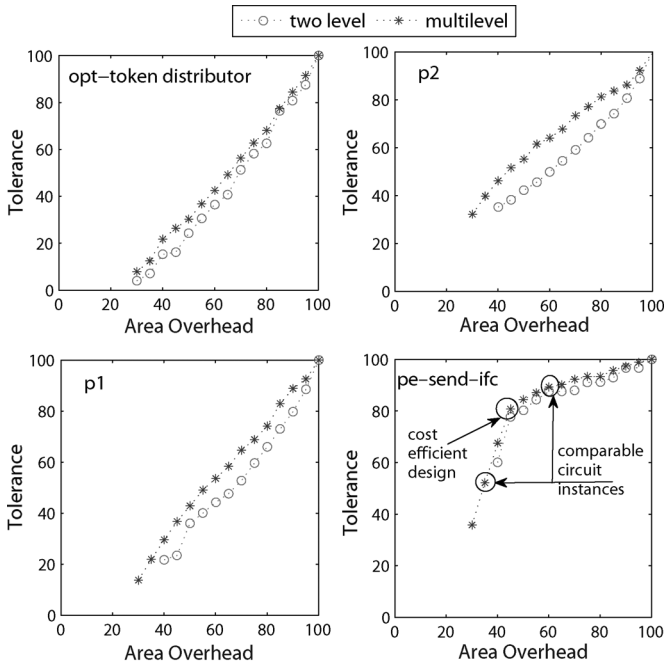


Fig. 10. Comparison between two-level and multilevel ABMMs.

gate into smaller ones provides more choices on which gates to include in the mitigation logic, which enables a more efficient exploration of the tradeoff space.

Moreover, the proposed mitigation methods are able to yield logic implementations for lower targets of area overhead than in two-level ABMMs. This is attributed to the cost of the OR gates driving the output/state functions in an ABMM, which is higher in a two-level ABMM due to the large number of inputs. Hence, in a two-level ABMM, the inclusion of an OR gate in the mitigation logic implementation increases the cost significantly, and thus, no solution exists for low targets of area overhead. In contrast, multilevel ABMMs decompose the gate into multiple ones, which enable the inclusion of part(s) of the decomposed gate at a lower cost than that of the monolithic gate. Therefore, mitigation logic implementations for low targets of area overhead can be derived.

Finally, we illustrate the ability to explore the design space and select a solution that maximizes the ratio of soft-error mitigation ability over the incurred overhead, as discussed in objective 3 of Section VI-E. Given this objective, and among the available options from the multilevel results of benchmark *pe-send-ifc*, which are shown in the upper plot of the lower right corner of Fig. 10, one would choose the point labeled "cost-efficient design," where this ratio obtains its maximum value of 1.793. On the same figure, we also pinpoint two design instances that offer very different levels of soft-error tolerance, incur very different overhead, yet yield similar return-on-investment ratios (1.490 and 1.488).

X. Conclusion

Careful examination of the impact of transient errors in ABMMs reveals the limitations of traditional error tolerance methods, such as the standard TMR approach, in protecting these circuits. Toward soft-error-tolerant ABMMs, the solution proposed herein leverages the inherent functionality of C-elements and extends a duplication-based error tolerance method to withstand more soft errors than the typical TMR method, including errors that jeopardize communication of the ABMM with its environment and errors within the C-elements themselves. At the same time, the proposed solution incurs less area overhead, even when compared to previous CED methods. Furthermore, based on a newly developed soft-error susceptibility assessment method for ABMMs, soft-error mitigation solutions can also be devised. Indeed, as demonstrated experimentally, partial duplication through careful selection of individual gates, complete state/output logic cones, or partial state/output logic cones enables efficient exploration of the tradeoff between the incurred overhead and the achieved soft-error susceptibility reduction.

References

[1] K. Mohanram and N. A. Touba, "Cost-effective approach for reducing soft error rate in logic circuits," in *IEEE Int. Test Conf.*, 2003, pp. 893–901.

[2] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005.

[3] S. Almukhaizim, Y. Makris, Y.-S. Yang, and A. Veneris, "Seamless integration of SER in rewiring-based design space exploration," in *IEEE Int. Test Conf.*, 2006, pp. 29.3.1–29.3.9.

[4] Q. Zhou and K. Mohanram, "Gate sizing to radiation harden combinational logic," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 1, pp. 155–166, Jan. 2006.

[5] ARM Ltd., "ARM996HS Processor," Jan. 18, 2009 [Online]. Available: http://www.handshakesolutions.com/products_services/ARM996HS/Index.html

[6] W. Jang and A. Martin, "SEU-tolerant QDI circuits," in *Proc. IEEE Int. Symp. Asynchronous Circuits Syst.*, 2005, pp. 156–165.

[7] Y. Monnet, M. Renaudin, and R. Leveugle, "Designing resistant circuits against malicious faults injection using asynchronous logic," *IEEE Trans. Comput.*, vol. 55, no. 9, pp. 1104–1115, Sep. 2006.

[8] S. Almukhaizim and Y. Makris, "Concurrent error detection methods for asynchronous burst-mode machines," *IEEE Trans. Comput.*, vol. 56, no. 6, pp. 785–798, Jun. 2007.

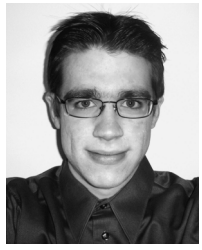
[9] F. Shi and Y. Makris, "SPIN-SIM: Logic and fault simulation for speed-independent circuits," in *Proc. IEEE Int. Test Conf.*, 2004, pp. 597–606.
[10] F. Shi and Y. Makris, "SPIN-TEST: Automatic test pattern generation for speed-independent circuits," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Des.*, 2004, pp. 903–908.
[11] D. E. Muller, "Asynchronous logics and application to information processing," in *Symp. Appl. Switching Theory Space Technol.*, Stanford, CA, 1962, pp. 289–297.
[12] D. A. Huffman, *The Synthesis of Sequential Switching Networks*. Reading, MA: Addison-Wesley, 1964.
[13] S. H. Unger, *Asynchronous Sequential Switching Circuits*. New York: Wiley-Interscience, 1969.
[14] R. M. Fuhrer and S. M. Nowick, *Sequential Optimization of Asynchronous and Synchronous Finite-State Machines: Algorithms and Tools*. Norwell, MA: Kluwer, 2001.
[15] A. Saldanha, T. Villa, R. K. Brayton, and A. Sangiovanni-Vincentelli, "A framework for satisfying input and output encoding constraints," in *Proc. ACM/IEEE Des. Autom. Conf.*, 1991, pp. 170–175.
[16] S. M. Nowick and D. L. Dill, "Exact two-level minimization of hazard-free logic with multiple-input changes," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 15, no. 8, pp. 986–997, Aug. 1995.
[17] Y. Zhao and S. Dey, "Separate dual-transistor registers—A circuit solution for on-line testing of transient error in UDSM-IC," in *Proc. IEEE Int. On-Line Testing Symp.*, 2003, pp. 7–11.
[18] M. Omaña, D. Rossi, and C. Metra, "Novel transient fault hardened static latch," in *Proc. IEEE Int. Test Conf.*, 2003, pp. 886–892.
[19] S. Mitra, M. Zhang, N. Seifert, B. Gill, S. Waqas, and K. S. Kim, "Combinational logic soft error correction," in *Proc. IEEE Int. Test Conf.*, 2006, pp. 29.2.1–29.2.10.
[20] M. Shams, J. C. Ebergen, and M. I. Elmasry, "Modeling and comparing CMOS implementations of the C-element," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 4, pp. 563–567, Dec. 1998.
[21] P. Lidén, P. Dahlgren, R. Johansson, and J. Karlsson, "On latching probability of particle induced transients in combinational networks," in *Proc. Symp. Fault-Tolerant Comput.*, 1994, pp. 340–349.
[22] M. P. Baze and S. P. Buchner, "Attenuation of single event induced pulses in CMOS combinational logic," *IEEE Trans. Nucl. Sci.*, vol. 44, no. 6, pp. 2217–2223, Dec. 1997.
[23] F. Shi and Y. Makris, "Enhancing simulation accuracy through advanced hazard detection in asynchronous circuits," *IEEE Trans. Comput.*, vol. 58, no. 3, pp. 394–408, Mar. 2009.
[24] E. B. Eichelberger, "Hazard detection in combinational and sequential switching circuits," *IBM J. Res. Develop.*, vol. 9, no. 2, pp. 90–99, 1965.
[25] J. P. Hayes, "Digital simulation with multiple logic values," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. CAD-5, no. 2, pp. 274–283, Apr. 1986.
[26] S. Sur-Kolay, M. Roncken, K. Stevens, P. P. Chaudhuri, and R. Roy, "Fsimac: A fault simulator for asynchronous sequential circuits," in *Proc. Asian Test Symp.*, 2000, pp. 114–119.
[27] T. J. Chakraborty, V. D. Agrawal, and M. L. Bushnell, "Delay fault models and test generation for random logic sequential circuits," in *Proc. ACM/IEEE Des. Autom. Conf.*, 1992, pp. 165–172.
[28] J. Brzozowski and Z. Esik, "Hazard algebras," *Form. Methods Syst. Des.*, vol. 23, no. 3, pp. 223–256, Nov. 2003.
[29] P. Raghavan and C. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, Dec. 1987.
[30] S. M. Nowick, N. K. Jha, and F.-C. Cheng, "Synthesis of asynchronous circuits for stuck-at and robust path delay fault testability," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 16, no. 12, pp. 1514–1521, Dec. 1997.
[31] K. Keutzer, L. Lavagno, and A. Sangiovanni-Vincentelli, "Synthesis for testability techniques for asynchronous circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 14, no. 12, pp. 1569–1577, Dec. 1995.
[32] C. Wey, M.-D. Shieh, and P. D. Fisher, "ASLCScan: A scan design technique for asynchronous sequential logic circuits," in *Proc. Int. Conf. Comput. Des.*, 1993, pp. 159–162.

Sobeeh Almukhaizim (S'99–M'07) received the Diploma degree in electrical and computer engineering from Kuwait University, Kuwait City, Kuwait, in 1999, the M.S. degree in computer science and engineering from the University of California, San Diego, in 2001, and the M.S., M.Phil., and Ph.D. degrees in electrical engineering from Yale University, New Haven, CT, in 2003 and 2007, respectively.

He is currently an Assistant Professor with the Department of Computer Engineering, Kuwait University. His research interests include VLSI design and test, computer architecture, microprocessor testing, fault tolerance, test and reliability of synchronous and asynchronous circuits, and soft-error mitigation and low-power test in digital circuits.

Feng Shi (S'02–M'09) received the B.Eng. and M.Eng. degrees in electrical engineering from Tsinghua University, Beijing, China, in 2000 and 2002, respectively, and the Ph.D. degree in electrical engineering from Yale University, New Haven, CT, in 2007.

He is currently with the Central Analog Department, Marvell Semiconductor, Santa Clara, CA. His current research interests include asynchronous circuit design and testing, mixed-signal IC design, computer architecture, and low-power design.



Eric Love is currently working toward the B.S. degree in electrical engineering and computer science at Yale University, New Haven, CT.

In the summer of 2008, he participated in an NSF-funded Research Experience for Undergraduates program. His research interests include reliable digital systems.

Yiorgos Makris (S'96–M'03–SM'08) received the Diploma degree in computer engineering and informatics from the University of Patras, Patras, Greece, in 1995 and the M.S. and Ph.D. degrees in computer science and engineering from the University of California, San Diego, in 1997 and 2001, respectively.

He is currently an Associate Professor of electrical engineering and computer science with Yale University, where he leads the Testable and Reliable Architectures (TRELA) Research Group. His current research interests include soft-error mitigation in digital circuits, machine-learning-based testing of analog/RF circuits, as well as test and reliability of asynchronous circuits.