Coping with Soft Errors in Asynchronous Burst-Mode Machines

Sobeeh Almukhaizim
Computer Engineering Department
Kuwait University

Feng Shi & Yiorgos Makris
Electrical Engineering Department
Yale University

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 (TMR) error tolerance method and often even less expensive than previously proposed Concurrent Error Detection (CED) 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 exploration of the trade-off between the achieved soft error susceptibility reduction and the incurred area overhead.*

1 Introduction

Soft errors are emerging as a serious threat to the reliable operation of logic circuits. When high-energy neutrons or alpha particles strike a sensitive region in a semiconductor device, they generate a Single Event Transient (SET) which may alter the state of the system, resulting in a soft error. The projected increase in the Soft Error Rate (SER) of near-future CMOS technology has sparked numerous efforts to develop soft error protection mechanisms for digital Integrated Circuits (ICs) [1]. Since the majority of commercial ICs available in the marketplace follow the clocked design paradigm, most of these efforts target synchronous circuits and, with regards to their effectiveness, can be divided into *soft error tolerance* and *soft error mitigation* approaches. The former takes an expensive holistic approach and attempts to tolerate all SETs in the circuit, while the latter aims to explore the trade-off between the provided soft error protection and the incurred cost. Unfortunately, soft error tolerance and mitigation methods developed for synchronous circuits are not directly portable to the asynchronous domain. And while a few soft error analysis, tolerance and design hardening methods have been developed for the class of Quasi-Delay Insensitive (QDI) circuits [2, 3, 4], their utility is limited in other classes, each of which presents its own challenges.

The research described in 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 work include:

- A duplication-based soft error tolerant ABMM design methodology, which leverages the inherent functionality of Muller C-elements to reduce the cost and improve the robustness of the typical Triple Modular Redundancy (TMR) approach. As will be shown, the proposed method is often even less expensive than previously proposed Concurrent Error Detection (CED) methods for ABMMs [5], which only provide detection but no correction.
- A soft error susceptibility assessment methodology for ABMMs, based on an enhanced version of a previously developed asynchronous circuit fault simulator [6].
- 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 trade-off between area overhead and SER reduction in ABMMs.

The paper is organized as follows. In Section 2, we review related work in soft error tolerance and mitigation in asynchronous circuits. In Section 3, we briefly introduce ABMMs. In Section 4, we describe TMR and the proposed duplication-based soft error tolerant ABMM design method. In Section 5, we devise a fault simulation-based method to compute soft error susceptibility in ABMMs. Then, in Section 6, we describe the proposed soft error mitigation solution. Finally, in Section 7, we demonstrate experimentally the ability of the proposed methods to explore the trade-off between area overhead and soft error tolerance on a standard set of benchmark ABMMs.

2 Related Work

To our knowledge, two studies related to soft errors in asynchronous circuits have been previously performed [3, 4]. 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. Below, we summarize these two studies and outline the reasons due to which they cannot be directly applied to ABMMs.

Jang *et al.* [3] 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 in these circuits, 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

1522-8681/08 $25.00 © 2008 IEEE
DOI 10.1109/ASYNC.2008.15

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 [7]. 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. The use of two such C-elements per gate also enables tolerance of soft errors occurring in these C-elements. While this method could potentially be ported to ABMMs, the fine granularity 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.* [4, 8, 9] were the first to quantify the susceptibility of QDI circuits to soft errors. In their analysis, the circuit is divided into two parts, 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 memory C-elements in sensitive states. In order to harden the QDI circuit against soft errors, the authors employ three methods, based on duplication of the computational part and expansion of the C-element in the memory part, synchronization of linked channels when available, and synchronization using a redundant control circuit, when no linked channels are available.

While the above sensitivity analysis and hardening methods are effective in QDI circuits, they cannot be applied to ABMMs. First, sensitivity assessment in [4] 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 only in hazards but no functional discrepancy at the output [5]. 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 [4]. Therefore, new methods for soft error susceptibility analysis and hardening are required for ABMMs.

3 Asynchronous Burst-Mode Machines

In this section, we briefly review the fundamentals of ABMMs, we outline the synthesis process for realizing an ABMM implementation from a Finite State Machine (FSM) description, and we give an example (adapted from [5]).

3.1 Fundamentals

ABMMs constitute a class of *Huffman* circuits [10]. As shown in Fig. 1, Huffman circuits 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] [11]. 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 that the environment is allowed to provide to the circuit, as well as the properties of the responses of the circuit. Based on these protocols, various classes of asynchronous circuits are defined.

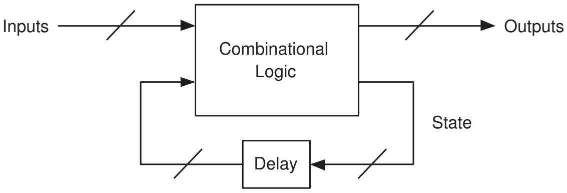


Figure 1. Huffman Asynchronous Circuits

The key aspect of the protocol used in Asynchronous Burst-Mode Machines, 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. Once an input burst is complete, and only then, the circuit responds 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 between the circuit and its environment 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 and optimization tools, such as MINIMALIST [12], which are available in the public domain.

3.2 Example

An ABMM is described using a state transition table such as the one shown in Fig. 2. 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,

[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

Figure 2. Example of a Symbolic State Transition Table for Defining an ABMM

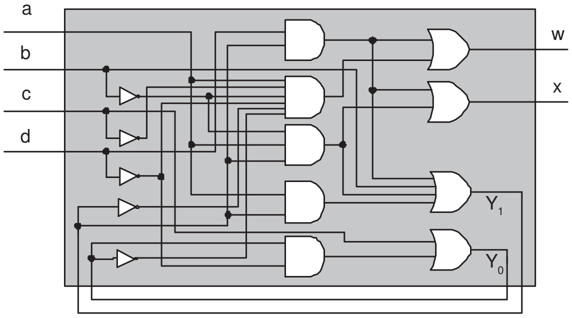


Figure 3. ABMM Implementation of the Example

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. For example, if the circuit is in state S_1, an input burst of 0010 is not allowed to occur. The synthesis process of MINIMALIST starts by solving the state encoding problem, which is modelled as a set of *dichotomies* [13] 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 [14]. Fig. 3 shows the resulting ABMM, which includes some logic redundancy to ensure hazard-free operation.

4 Soft Error Tolerance in ABMMs

Towards 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 [3] 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 CED methods for ABMMs [5], despite the fact that the latter provide only detection but no correction. The proposed method is demonstrated and contrasted to TMR and CED using the running example of Fig. 3.

4.1 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 both in synchronous [15, 16, 17] and in asynchronous [3] circuits, the majority voter module can be substituted by a Muller C-element [7]. 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 3-input C-element are three 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, in which case the output of the C-element follows.

The TMR-based soft error tolerant design method for ABMMs is illustrated in Fig. 4. 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 temporarily changes the output of a C-element on a state-line may propagate through the combinational feedback back to its inputs, as illustrated via the dotted lines in Fig. 4. 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 a C-element driving an output, since it results in a chain-reaction

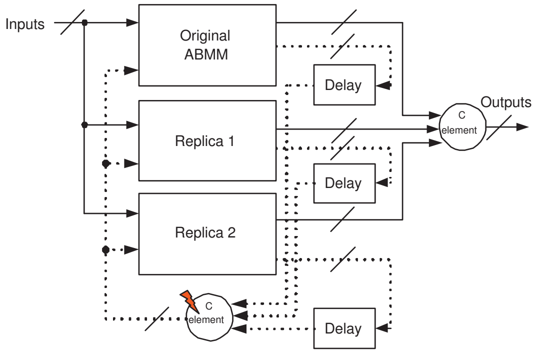


Figure 4. TMR-Based Soft Error Tolerance

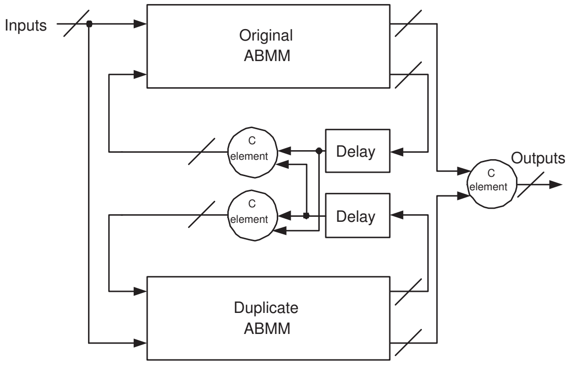


Figure 5. Duplication-Based Soft Error Tolerance

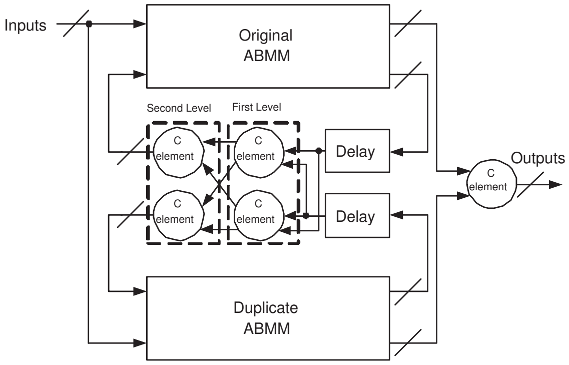


Figure 6. Tolerating SETs on State-Line C-elements

of erroneous states, outputs, and, by extension, miscommunication between the ABMM and the environment. This limitation, along with the excessive cost incurred, make TMR a rather non-appealing option for designing soft error tolerant ABMMs.

4.2 Duplication-based Soft Error Tolerance

In this section, we describe a duplication-based soft error tolerant design strategy which 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 preserves its state until all inputs agree to a new value. In other words, even if two out of the three circuit copies in Fig. 4 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 [15, 16, 17] and for QDI asynchronous circuits [3]. In the latter, a fine-grain duplication and double-checking approach is taken, as discussed in Section 2. 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 illustrated in Fig. 5. A replica of the ABMM and one C-element for every pair of duplicate outputs is added to the design. This ensures that soft errors in one of the ABMM copies 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. This ensures that soft errors in one of the ABMM copies does not propagate back to the ABMM, so it cannot 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 4 C-elements to state-lines, as shown in Fig. 6. 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.

4.3 Example

The duplication-based soft error tolerant ABMM design for the running example ABMM of Fig. 3, as well as the enhanced version for suppressing soft-errors on state-line C-elements, are shown in Fig. 7 and Fig. 8 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. 7, a transient error changing the state of the C-element which 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 0 even after the transient error disappears, since the current inputs to the C-element have now values of 0 and 1. On the other hand, the same transient error occurring in the enhanced implementation of Fig. 8 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 easily also verify that a SET striking a C-element in the second-level of the cross-coupled structure may propagate through one of the

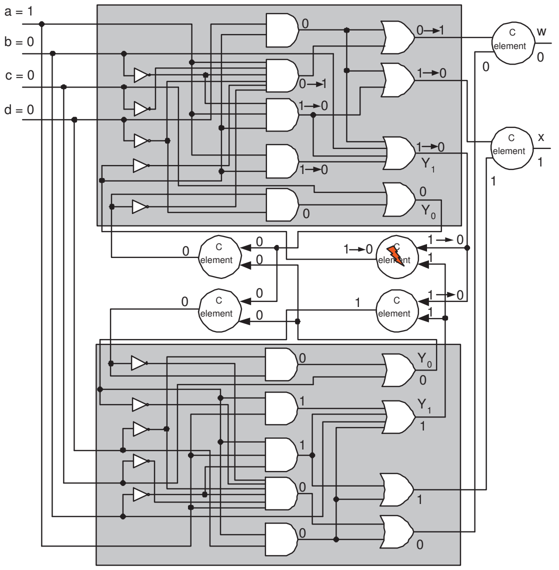


Figure 7. Duplication Example for Circuit in Fig. 3

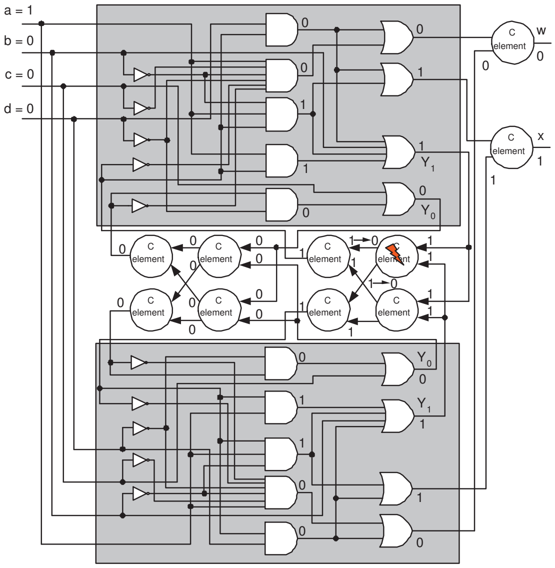


Figure 8. Enhanced Duplication for Circuit in Fig. 3

ABMM copies but will not reach the inputs of this C-element, as it will be suppressed by the C-elements in the first level.

Assuming a rather expensive C-element implementation using three 2-input NAND gates [18], the cost of the enhanced duplication design of Fig. 8 is 3x the cost of the original circuit. In contrast, the cost of the TMR-based soft error tolerant design of Fig. 4 is 3.6x 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 4 C-elements that provide this additional robustness with single C-elements, the cost drops to 1.8x 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 [5], despite the fact that the latter only provides detection.

5 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 [19, 20]. Devising solutions that explore this trade-off 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 [6] 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.

5.1 Masking Factors

Three masking factors determine whether a SET striking an internal gate of a synchronous combinational circuit will propagate to the output and result in a soft error [21, 22]. 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 in *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 above masking factors. ABMMs operate without a global clock, so latching-window masking does not apply to these circuits. Also, while electrical masking in ABMMs can be assessed in a fashion similar to synchronous combinational logic (i.e. computationally expensive Spice simulations), we opted to exclude this factor from our susceptibility analysis for the same two reasons as outlined in [19]. First, since ABMMs are high performance controllers implemented in a two-level logic fashion, 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. [23]. The latter concludes that, while there is an observable effect, it cannot be generally assumed that electrical masking will significantly reduce the observed error rate.

This leaves logic masking as the key mechanism for withstanding SETs in ABMMs. Therefore, a fault simulation-based approach is necessary for assessing their soft error susceptibility. Such fault simulation, however, should take into account that

State & Input	Potential SETs			
Burst Pairs (SIB)	f_1	f_2	$\dots$	f_p
SIB_1	11…0	01…1	…	00…0
SIB_2	01…0	11…1	…	00…1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
SIB_m	01…1	00…0	…	01…0

Table 1. Soft Error Susceptibility Table

SETs in ABMMs may result not only in logic errors but also in hazards [5], which jeopardize the communication of the ABMM with its environment. In order to identify these SETs, a fault simulator tailored to the particularities of ABMMs is needed.

5.2 Fault Simulation in ABMMs

In order to compute the soft error susceptibility of ABMMs, we use SPIN-SIM [6]. SPIN-SIM is a logic and fault simulator which 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 [24] for delay-insensitive circuits through insertion of buffers to handle arbitrary delays on both gates and wires, and QDI circuits, through transformation of their *isochronic forks* into an equivalent speed-independent circuit form. For the purpose of this work, we enhanced SPIN-SIM to simulate faults in ABMMs, which operate correctly based on the *fundamental mode* assumption, i.e. that outputs and state variables must stabilize before new inputs or feedback state variables arrive. Fundamental mode operation for internal state variables is usually guaranteed by inserting sufficient delay in the feedback. While an ABMM can be handled, in general, as a delay-insensitive circuit, special care is required during time-frame unfolding. Therefore, we adapted the time-stamping method of SPIN-SIM to account for the fundamental mode operation and report both functional discrepancies and hazards occurring due to SETs.

5.3 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 *Soft Error Rate* (SER) of the circuit. Towards this end, we construct the soft error susceptibility table illustrated in Table 1. 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. Columns represent potential SETs in the circuit. Each table entry contains a bit-string which 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 $q \in [1, \dots, n]$, and let $sest[i, j]$ denote the (i, j)-th entry of the soft error susceptibility table for all $i \in [1, \dots, m]$, $j \in [1, \dots, p]$. 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 at an output or state-line (or not). Then, the susceptibility of G_q is defined as:

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

and the SER of the ABMM is defined as:

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

Essentially, the soft error 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.

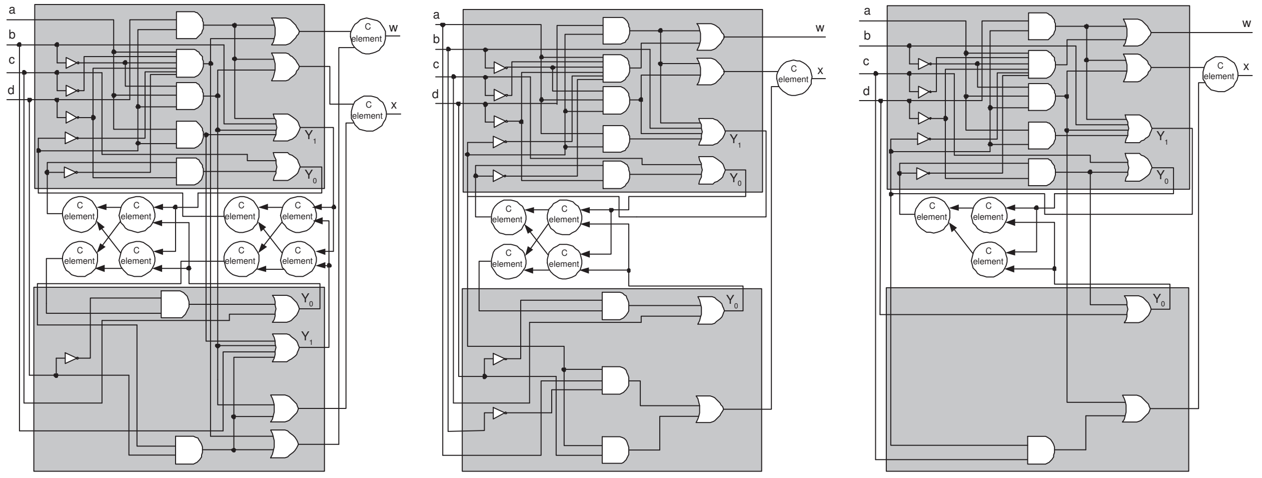
6 Soft Error Mitigation in ABMMs

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 trade-off 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 discussed herein and illustrated using the example of Fig. 3.

6.1 Duplication of Sensitive Gates

Due to asymmetric susceptibility [22], 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 non-duplicated 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 4.2. Then, the soft error susceptibility of each



a. Duplication of Sensitive Gates
b. Complete Cones of Logic
c. Partial Cones of Logic

Figure 9. Examples of Circuit Implementations Produced using the Mitigation Methods

gate in the original ABMM is computed using equation 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.

6.2 Duplication of Sensitive Complete Logic Cones

In contrast to the previous method, which exploits the asymmetric susceptibility of gates, this method builds upon the asymmetric susceptibility of state/output logic cones. In essence, it aims to select 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 1. This can be formulated as an Integer Linear Program (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 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 $V(sest[i,j])$ be the r-dimensional vector constructed from the r bits in $sest[i,j]$. We define function $Tol(Y_k, i, j)$ as follows:

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

where $\bar{Y_k}$ 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 the selected state/output subset Y_k tolerates SET j occurring during SIB i. The following ILP formulation finds the state/output subset Y_k that maximizes the number of tolerated entries in Table 1 for a given area overhead constraint ($COST$):

Maximize $\sum_{i=1}^{m} \sum_{j=1}^{p} Tol(Y_k, i, j)$
subject to:
(i) $C_k \leq COST$
(ii) $x_s \in \{0, 1\}$ for $1 \leq s \leq r$

While ILP is NP-Complete, it can be efficiently approximated through a well-known method combining linear program relaxation and randomized-rounding [25].

6.3 Duplication of Sensitive Partial Logic Cones

The third partial duplication method aims to combine the first two and leverages both the asymmetric susceptibility of gates and the asymmetric susceptibility of state/output logic cones. In other words, it explores solutions that include a subset of *partial* state/output logic cones. Similarly 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 6.2 for a higher $COST$ value than the targeted area cost ($COST_{target}$). Then, the state/output cones returned by the ILP are pruned by applying the method of section 6.1 until ($COST_{target}$) is met, in which case the partial state/output cones and the corresponding soft error tolerance are recorded. $COST$ is, then, increased[2] and the process is repeated until all cones are included in the ILP solution, at which point the best recorded solution is reported.

6.4 Examples

Fig. 9 shows instances of circuits produced by the proposed soft error mitigation method for the ABMM example of Fig. 3. For the first partial duplication option, all second-level gates and the corresponding C-elements appear in the replica in Fig. 9.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

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

Circuit		**Original**		**Duplicate**		**C-elements**			**Total**	
Name	I/S(Bits)/O	Lit.	Gates	Lit.	Gates	#	Lit.	Gates	Lit.	Gates
hp-ir	3/2(1)/2	13	8	13	8	6	36	18	62	34
martin-q-element	2/2(1)/2	14	9	14	9	6	36	18	64	36
tangram-mixer	3/2(1)/2	17	10	17	10	6	36	18	70	38
concur-mixer	3/3(2)/3	26	16	26	16	11	66	33	118	65
while	4/3(2)/3	27	16	27	16	11	66	33	120	65
while_concur	4/4(2)/3	41	24	41	24	11	66	33	148	81
opt-token-distributor	4/6(3)/4	74	41	74	41	16	96	48	244	130
rf-control	6/6(3)/5	75	37	75	37	17	102	51	252	125
pe-send-ifc	5/5(3)/3	110	58	110	58	15	90	45	310	161
barcode	13/11(4)/17	327	172	327	172	33	198	99	852	443
diffeq	14/9(4)/20	345	189	345	189	36	216	108	906	486
p2	8/13(4)/16	349	192	349	192	32	192	96	890	480
p1	13/11(4)/14	458	238	458	238	30	180	90	1096	566

Table 2. Experimental Results for Duplication-Based Soft Error Tolerance

Circuit	Area Cost (**Tolerance Methods**)		Area Cost (**CED Methods [5]**)			**Reduction in Area Cost Over (%)**			
	TMR -Based	**Proposed** **Duplication-Based**	Duplication -Based	Transition Triggered	Berger Code	TMR -Based	Duplication -Based	Transition Triggered	Berger Code
hp-ir	42	**34**	61	63	57	19.05	44.26	46.03	40.35
martin-q-element	45	36	44	43	**31**	20.00	18.18	16.28	-16.13
tangram-mixer	48	**38**	46	48	44	20.83	17.39	20.83	13.64
concur-mixer	78	**65**	87	88	85	16.67	25.79	26.14	23.53
while	78	65	70	69	**56**	16.67	7.14	5.80	-16.07
while_concur	102	**81**	104	108	88	20.59	22.11	25.00	7.96
opt-token-distributor	165	130	132	119	**109**	21.21	1.52	-9.24	-19.27
rf-control	159	125	154	145	**121**	21.38	18.83	13.79	-3.31
pe-send-ifc	210	**161**	233	211	252	23.33	30.90	23.70	36.11
barcode	642	**443**	639	611	547	31.00	30.67	27.50	19.01
diffeq	711	**486**	690	660	556	31.65	29.57	26.37	12.59
p2	696	480	581	510	**436**	31.03	17.38	5.88	-10.09
p1	822	**566**	725	599	613	31.14	21.93	5.51	7.67
					Averages	**23.43%**	**21.94%**	**17.97%**	**7.38%**

Table 3. Comparison Between Duplication-Based Tolerance, TMR-Based Tolerance and the CED Methods in [5]

duplication option, only some second-level gates and the corresponding C-elements appear in the replica in Fig. 9.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. 9.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. 8, the implementations in Fig. 9.a, Fig. 9.b, and Fig. 9.c require 87%, 60%, and 50% of its cost while providing 68%, 47%, and 24% of its transient error tolerance, respectively.

7 Experimental Results

The proposed soft error tolerance and mitigation methods were applied on a suite of 13 benchmark circuits, which have been previously proposed and used by the asynchronous design community [12, 14, 26, 27, 28, 29]. The circuits are first synthesized using MINIMALIST[12] to generate an ABMM implementation. Then, the TMR-based and the duplication-based soft error tolerant implementations are constructed, as described in Section 4. Next, the soft error susceptibility table of the original ABMM is generated using the enhanced version of SPINSIM [6], as discussed in Section 5.3, and the partial duplication-based soft-error mitigation solution described in Section 6 are applied. In Section 7.1, we compare the results of the proposed duplication-based soft error tolerance method to TMR and CED methods for ABMMs. Then, in Section 7.2, we present the results of the three partial duplication options of Section 6.

7.1 Soft Error Tolerance Results

In Table 2, we present the results for duplication-based soft error tolerance, including 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). We also report the cost of the original circuit and its duplicate, and the number and cost of the C-elements. Then, we summarize the total literal and gate count of the duplication-based soft error tolerant ABMM. The gate count of the circuits is normalized to the equivalent number of 2-input NAND-gates. While for small circuits, such as $hp - ir$, $martin - q - element$, and $tangram - mixer$, 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 benchmarks, such as $diffeq$, $p2$, 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

		Soft Error Susceptibility Reduction Achieved For Target Area Overhead (%)													
Circuit	**Method**	**30%**	**35%**	**40%**	**45%**	**50%**	**55%**	**60%**	**65%**	**70%**	**75%**	**80%**	**85%**	**90%**	**95%**
hp-ir	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	-	42.66	61.54	81.82
	$\mathbf{M_2}$	-	-	-	46.86	46.86	46.86	46.86	62.24	62.24	62.24	62.24	62.24	62.24	62.24
	$\mathbf{M_3}$	11.19	15.38	30.07	46.86	46.86	56.64	62.24	62.24	62.24	62.24	62.24	62.24	72.73	81.82
martin-q-element	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	-	39.44	54.93	77.46
	$\mathbf{M_2}$	-	-	25.35	32.39	32.39	57.75	57.75	57.75	57.75	57.75	57.75	57.75	57.75	57.75
	$\mathbf{M_3}$	-	5.63	29.58	32.39	38.03	57.75	57.75	57.75	57.75	57.75	57.75	57.75	67.61	77.46
tangram-mixer	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	-	45.57	56.96	81.01
	$\mathbf{M_2}$	-	13.29	29.75	29.75	29.75	57.59	57.59	57.59	70.89	70.89	70.89	70.89	70.89	70.89
	$\mathbf{M_3}$	10.13	22.15	29.75	29.75	47.47	57.59	63.29	70.89	70.89	70.89	70.89	70.89	70.89	81.01
concur-mixer	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	-	48.81	71.43	85.71
	$\mathbf{M_2}$	-	9.52	26.19	26.19	35.71	35.71	45.24	45.24	61.31	61.31	70.83	70.83	71.43	71.43
	$\mathbf{M_3}$	9.52	19.64	33.93	36.31	45.24	45.24	45.24	48.81	61.31	69.64	70.83	70.83	71.43	85.71
while	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	-	40.00	56.54	80.77
	$\mathbf{M_2}$	-	10.00	24.62	24.62	34.62	44.62	44.62	48.81	63.08	69.64	73.08	73.08	73.08	73.08
	$\mathbf{M_3}$	6.15	17.69	24.62	31.54	38.46	44.62	44.62	50.77	63.08	70.77	73.08	73.08	73.08	86.15
while_concur	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	35.27	58.56	68.49	83.56
	$\mathbf{M_2}$	-	11.64	17.47	20.21	29.11	29.11	37.67	37.67	52.74	52.74	67.47	67.47	67.47	75.68
	$\mathbf{M_3}$	6.85	15.04	23.97	29.45	29.79	37.33	37.67	41.44	52.74	60.96	67.47	67.47	75.34	83.56
opt-token-distributor	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	43.58	53.38	61.49	85.30
	$\mathbf{M_2}$	-	7.09	15.20	16.22	23.31	27.03	31.08	37.84	40.03	53.72	60.14	63.05	80.74	80.74
	$\mathbf{M_3}$	4.05	7.09	15.20	16.22	24.16	30.57	36.49	40.71	51.18	58.11	62.50	76.35	80.74	87.50
rf-control	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	39.77	57.77	70.08	82.12
	$\mathbf{M_2}$	-	4.92	10.62	20.34	25.26	30.57	35.36	40.54	47.02	52.46	58.16	68.03	82.51	87.18
	$\mathbf{M_3}$	3.11	7.12	15.67	23.32	30.44	38.73	40.28	48.19	54.79	58.29	69.82	76.55	82.51	93.65
pe-send-ifc	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	42.07	52.88	65.88	80.32
	$\mathbf{M_2}$	-	-	-	77.72	80.21	84.40	87.51	87.51	87.91	91.02	91.13	92.86	96.69	96.69
	$\mathbf{M_3}$	-	-	60.02	77.72	80.21	84.40	87.51	87.51	87.91	91.02	91.13	92.86	96.69	96.69
barcode	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	41.01	48.13	58.61	66.38	82.86
	$\mathbf{M_2}$	-	-	56.50	58.59	60.55	62.69	75.24	81.17	84.96	88.83	92.49	94.89	96.87	97.64
	$\mathbf{M_3}$	-	54.65	57.38	59.67	63.02	74.87	80.26	82.67	85.59	89.58	92.49	95.18	96.87	97.64
diffeq	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	50.33	59.70	70.78	82.68
	$\mathbf{M_2}$	-	-	78.86	80.27	81.92	83.99	85.34	86.54	88.31	90.35	92.08	93.62	95.15	97.40
	$\mathbf{M_3}$	-	76.55	79.78	81.28	83.32	84.70	85.74	87.29	89.16	90.67	92.08	93.62	95.27	98.05
p2	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	-	55.55	63.21	72.31	83.42
	$\mathbf{M_2}$	-	-	35.17	37.86	42.22	45.51	49.87	54.47	59.06	63.91	69.82	74.12	80.65	88.28
	$\mathbf{M_3}$	-	-	35.17	38.09	42.22	45.51	49.87	54.50	59.06	64.04	69.82	74.12	80.65	88.90
p1	$\mathbf{M_1}$	-	-	-	-	-	-	-	-	-	54.41	59.13	66.74	77.07	88.06
	$\mathbf{M_2}$	-	-	21.71	23.35	28.67	32.98	40.40	45.59	50.07	55.04	61.35	68.49	77.18	87.87
	$\mathbf{M_3}$	-	-	21.71	23.35	35.96	39.96	44.25	47.65	52.76	59.57	65.94	72.97	79.75	88.49

Table 4. Percentile Soft Error Susceptibility Reduction of the Proposed Mitigation Methods

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 duplication-based soft error tolerance is summarized in the second and third columns of Table 3, respectively. As an additional point of reference, we also provide the area cost of the previously proposed CED methods for ABMMs [5] in the fourth to sixth columns. For each circuit, the solution with the lowest area cost is shown in boldface. The last major column illustrates the reduction in area cost of duplication-based tolerance over the TMR and the CED methods. As can be observed, the area cost of duplication-based tolerance is, on average, 24% less than that of TMR. We remind that, as explained in section 4.1, TMR is less robust. 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 known CED methods, which only report detection of an error. On average, this hardware reduction is 22%, 18% and 7% over the 3 CED methods reported in [5], respectively. In short, the reduced cost and the increased effectiveness of the duplication-based soft error tolerant method makes it the current method of choice for ABMMs.

7.2 Soft Error Mitigation Results

In Table 4, we provide the reduction in the soft error susceptibility achieved by the the proposed partial duplication-based soft error mitigation method on the benchmark circuits. The first column provides the circuit name, the second column indicates the employed partial duplication method (M_1 for duplication of sensitive gates, M_2 for duplication of sensitive complete state/output cones, and M_3 for duplication of sensitive partial state/output cones), and the last major column presents the achieved soft error susceptibility reduction for a given area overhead target. The results are presented for 5% increments in the targeted area overhead of the mitigation logic, where 100% reflects the cost of the complete duplication-based soft error tolerance method. Dashes in the table indicate that no solution can be achieved by the corresponding method at the targeted area overhead. Three observations are supported by these results: i) the reduction in soft error susceptibility is commensurate with the incurred area overhead, ii) 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, and iii) 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. In short, duplication of partial sensitive state/output cones enables the most efficient exploration of the trade-off between area overhead and soft error susceptibility reduction.

8 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. Towards 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, which only detect but do not correct errors. 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 trade-off between the incurred area overhead and the achieved soft error susceptibility reduction.

References

[1] M. Nicolaidis, "Design for soft error mitigation," *IEEE Transactions on Device and Materials Reliability*, vol. 5, no. 3, pp. 405–418, 2005.

[2] S. J. Piestrak and T. Nanya, "Towards totally self-checking delay-insensitive systems," in *International Fault-Tolerant Computing Symposium*, 1995, pp. 228–237.

[3] W. Jang and A. Martin, "SEU-tolerant QDI circuits," in *IEEE International Symposium on Asynchronous Circuits and Systems*, 2005, pp. 156–165.

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

[5] S. Almukhaizim and Y. Makris, "Concurrent error detection methods for asynchronous burst-mode machines," *IEEE Transactions on Computers*, vol. 56, no. 6, pp. 785–798, 2007.

[6] F. Shi and Y. Makris, "SPIN-SIM: Logic and fault simulation for speed-independent circuits," in *IEEE International Test Conference*, 2004, pp. 597–606.

[7] D. E. Muller, "Asynchronous logics and application to information processing," in *Symposium on the Application of Switching Theory to Space Technology*. 1962, pp. 289–297, Stanford University Press.

[8] Y. Monnet, M. Renaudin, and R. Leveugle, "Asynchronous circuits transient faults sensitivity evaluation," in *ACM/IEEE Design automation conference*, 2005, pp. 863–868.

[9] Y. Monnet, M. Renaudin, and R. Leveugle, "Hardening techniques against transient faults for asynchronous circuits," in *IEEE International On-Line Testing Symposium*, 2005, pp. 129–134.

[10] D. A. Huffman, *The Synthesis of Sequential Switching Networks*, Addison-Wesley, 1964.

[11] S. H. Unger, *Asynchronous Sequential Switching Circuits*, Wiley-Interscience, 1969.

[12] R. M. Fuhrer and S. M. Nowick, *Sequential Optimization of Asynchronous and Synchronous Finite-State Machines: Algorithms and Tools*, Kluwer Academic Publishers, 2001.

[13] A. Saldanha, T. Villa, R. K. Brayton, and A. Sangiovanni-Vincentelli, "A framework for satisfying input and output encoding constraints," in *ACM/IEEE Design Automation Conference*, 1991, pp. 170–175.

[14] S. M. Nowick and D. L. Dill, "Exact two-level minimization of hazard-free logic with multiple-input changes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 8, pp. 986–997, 1995.

[15] Y. Zhao and S. Dey, "Separate dual-transistor registers - a circuit solution for on-line testing of transient error in UDSM-IC," in *IEEE International On-Line Testing Symposium*, 2003, pp. 7–11.

[16] M. Omana, D. Rossi, and C. Metra, "Novel transient fault hardened static latch," *IEEE International Test Conference*, pp. 886–892, 2003.

[17] S. Mitra, M. Zhang, N. Seifert, B. Gill, S. Waqas, and K.S. Kim, "Combinational logic soft error correction," in *IEEE International Test Conference*, 2006, pp. 29.2.1–29.2.10.

[18] M. Shams, J. C. Ebergen, and M. I. Elmasry, "Modeling and comparing CMOS implementations of the c-element," *IEEE Transactions on Very Large Scale Integration VLSI Systems*, vol. 6, no. 4, pp. 563–567, 1998.

[19] Q. Zhou and K. Mohanram, "Gate sizing to radiation harden combinational logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 1, pp. 155–166, 2006.

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

[21] P. Lidén, P. Dahlgren, R. Johansson, and J. Karlsson, "On latching probability of particle induced transients in combinational networks," in *Symposium on Fault-Tolerant Computing*, 1994, pp. 340–349.

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

[23] M. P. Baze and S. P. Buchner, "Attenuation of single event induced pulses in CMOS combinational logic," *IEEE Transactions on Nuclear Science*, vol. 44, no. 6, pp. 2217–2223, 1997.

[24] F. Shi, Y. Makris, S. M. Nowick, and M. Singh, "Test generation for ultra-high-speed asynchronous pipelines," in *IEEE International Test Conference*, 2005, pp. 39.1.1.–39.1.10.

[25] P. Raghavan and C. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[26] A. Marshall, B. Coates, and P. Siegel, "Designing an asynchronous communications chip," *IEEE Design & Test of Computers*, vol. 11, no. 2, pp. 8–21, 1994.

[27] K. S. Stevens, S. V. Robison, and A. L. Davis, "The post office-communication support for distributed ensemble architectures," in *International Conference on Distributed Computing Systems*, 1986, pp. 160–166.

[28] S. M. Nowick and D. L. Dill, "Automatic synthesis of locally-clocked asynchronous state machines," in *ACM/IEEE International Conference on Computer-Aided Design*, 1995, pp. 318–321.

[29] R. M. Fuhrer, S. M. Nowick, M. Theobald, N. K. Jha, B. Lin, and L. Plana, "MINIMALIST: An environment for the synthesis, verification and testability of burst-mode asynchronous machines," TR CUCS-020-99, Columbia University, Computer Science Department Technical Report, 1999.