Low-cost Error Resilient Circuits for Digital Control Paths

Aradhana Kumari
STMicroelectronics
Crolles, France
aradhana.kumari01@st.com

***Abstract*— Designs for safety critical applications involves the mitigation of soft errors at multiple stages of the design. This paper proposes multiple error resilient techniques that can be incorporated at the design stage. The vulnerability of the control section of the design to SEUs (Single Event Upsets) is addressed. Control circuits with multiple feedback paths where an error impact could be catastrophic are made fault tolerant. The listed error resilient circuits can be incorporated into contemporary state machines, high speed counters and index generators to make them resistant to bit flip events. It is shown that single bit soft error immunity can be achieved with minimal area and power overheads.**

Keywords—SEU, soft error, fault tolerance, control circuits

I. INTRODUCTION

Radiation-induced soft errors are a major concern for modern digital circuits, especially the ones deployed in safety critical applications. With very large scale integrations, circuits are becoming less tolerant to radiation effects as the power supply voltage is going down resulting in the reduction of the energy need of a particle to induce a soft error. The demand for mitigation techniques and hardening methods is increasing for critical applications of space, medical and automotive. Several techniques can be applied at different abstraction levels for enhancing the reliability of circuits.

At process level, immunity to SEU (Single Event Upset) events is ensured through "rad hard" (radiation hardened) processes[1]. The rad hard devices are slower and limit the operation frequency of the designs. Also, they are expensive because they require special processing steps to fabricate. Multiple latches and flip-flops [2][3][4][5] are proposed for radiation immunity, with DICE (Dual Interlocked Storage Cell) [6] among the more popular ones. These specialized cells are not a part of standard digital cell libraries and are challenging to characterize, and have limited usage during logic synthesis. Another popular method is the Triple Modular Redundancy (TMR). In TMR (as seen from Fig. 1), a module is replicated three times and the output is extracted from a majority voter that usually comprises three AND gates and one OR gate [7]. When a SEU occurs, the majority voter circuit functions by ignoring the failure value (outputs a,b,c,) of a module receiving the SEU and accepts a correct value from the other two modules not receiving it. Lockstep systems[8] are fault-tolerant computer systems that execute the identical cycle accurate operations simultaneously in parallel. This redundancy (duplication) enables error detection and correction: by comparing the outputs of lockstep operations, faults can be identified if there are at least two systems (dual modular redundancy, DMR). Furthermore, errors can be automatically corrected if there are at least three systems (triple modular redundancy, TMR) through a majority voting mechanism. Volatile memories are highly susceptible to SEU errors. There are well established methods [9] [10] for protecting Random Access Memories (RAM) using error-correcting codes and bit interleaving.

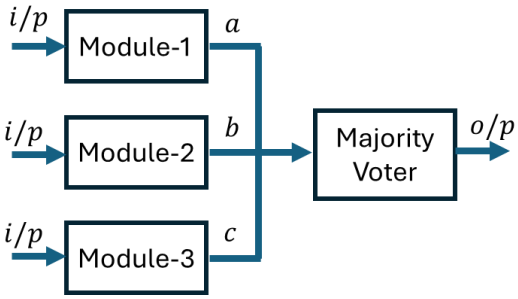

Fig 1. TMR Configuration

A digital block can be clearly partitioned into data and control paths. An error occurring in the control logic can have long term ramifications due to its closed loop architecture. The control loop could be made of multiple complex state machines or could be as simple as an incremental counter. Whatever be the control architecture, a bit flip in a control register will be trapped in the feedback loop forever. Unless detected and corrected, an SEU corrupted control machine is at high risk to run invalid states or counts indefinitely.

On the other hand, an SEU impacted datapath is relatively easy to self-correct as the faulty data and computation will eventually be flushed out of the system. Even in signal processing intensive feedback datapaths, SEU errors are ironed out by digital filtering or they manifest as offsets in final outputs which are easy to correct.

In this paper, multiple techniques for control logic protection and recovery are presented. The proposed techniques have lower overhead than known contemporary methods of DICE, TMR, lockstep among others. Another major advantage being the of reuse of existing process technologies and design flows.

II. ERROR RESILIENT CIRCUITS

The present section details design techniques for making a range of control circuits immune to single bit flip events. There are no requirements of special cells for their implementation and the functional timing criticality is maintained.

A. High reliability State Machines

We start by developing a framework for a digital state machine to be able to tolerate a bit flip event. State machines are a useful way to represent sequential logic that is controlling a sequence of events based on various internal and external signals. They are standard designs consisting of a state register which is updated on a clock. The outputs of the state register, together with inputs, are used to determine the next state, which will be loaded into the state register on the next clock. The output logic takes the state register value along with optional inputs and determines a set of outputs.

State	Binary	One-Hot	H-2	H-3
S0	000	00000001	0000	000000
S1	001	00000010	0011	000111
S2	010	00000100	0101	011001
S3	011	00001000	0110	011110
S4	100	00010000	1001	101010
S5	101	00100000	1010	101101
S6	110	01000000	1100	110011
S7	111	10000000	1111	110100

Fig 2. State Coding

The state register progresses through a series of predefined states, each symbolically represented, such as S0 or S1. The total number of states is dictated by the unique states required by the circuit. The size of the state machine register is influenced by both the number of states and the state machine encoding, which refers to the method used to represent these symbolic states within the state register. The simplest state machine encoding is binary, which involves a sequential binary count of the state numbers. Table 1 demonstrates the binary and three other possible codes for an 8-state state machine.

An alternative state encoding method is "one-hot" encoding. In this approach, only one bit of the state register is set for any given state. One-hot encoding requires minimal logic to determine the next state, making it efficient and popular in FPGA applications. Although this method results in a larger state register, this is generally not an issue in FPGAs, as they have abundance of registers.

However, the efficiency of one-hot encoding relies on an optimization that is unsuitable for high-reliability logic. In high-reliability circuits, every state must be explicitly defined. The large number of flip-flops required for one-hot encoding significantly increases the total number of possible states (state space), leading to substantial resource usage in highly reliable one-hot encoded state machines.

One of the key advantages of one-hot encoding over binary encoding is its preservation of a Hamming distance of 2. This ensures that each state is at least two bit transitions away from any other state. Consequently, a "single event upset" (SEU) caused by radiation can only force the state register into an illegal state. This illegal state can be predefined to always bring the state machine to an idle or "S0" state. As a result, no incorrect state is entered, and no incorrect outputs are generated, although the state machine may not complete its intended sequence correctly.

A preferred encoding method is to minimize the state register size while maintaining a Hamming distance of 2. H-2 encoding achieves this by cycling the state machine through states that differ by 2 bits, requiring only one more bit in the state register than binary encoding. H-2 encoding can be easily generated by adding a parity bit to binary encoding. Fig. 2 presents an 8-state H-2 encoding scheme. Since H-2 encoding uses fewer flip-flops than one-hot encoding, it is expected to be less prone to errors.

To ensure that a state machine completes correctly even in the event of a SEU resulting in an error state, H-3 encoding can be utilized. In H-3 encoding, there is a Hamming distance of 3 between all states, meaning that 3 bits must change for any state to represent another valid state. This is illustrated in figure 3. In the case of an SEU, the state will change to an

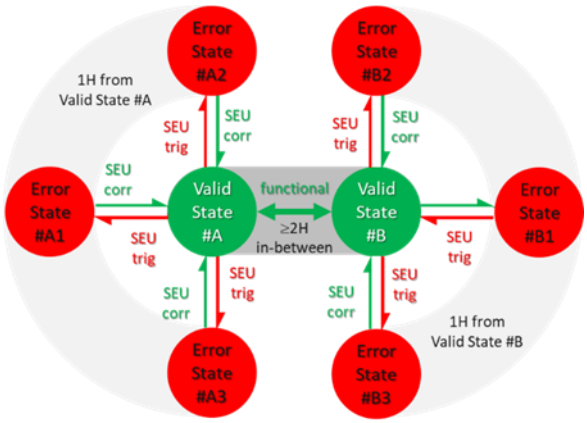

Fig 3. H-3 Coded State Machine

adjacent and unique error state (SEU trig) relative to the original state. With a Hamming distance of 3, an SEU in any other valid state will not cause the state register to assume this value. It is possible to design a state machine will recover (SEU corr) from this intermediate SEU-induced state change. Each legitimate state is defined as a set of values, consisting of the state encoding (Fig. 1) and every value that is a Hamming distance of 1 (also known as an adjacent state) from the state encoding. For the 8 states in the example in Table 1, each set actually consists of 9 values. As state machines have log2(N) bits in their construction, the area overhead is relatively low.

In operation, an SEU in the state register will bring the state machine to an adjacent state. However, since this adjacent state is treated the same as the original state, the state machine continues to complete its sequence normally, rendering the SEU ineffective. For managing more than 1-bit flip, it is possible to create codes with greater hamming distance than 3, with more fault tolerance.

B. High Speed Counter Design

The control path with index generators are very different from the state machine codes. An H-3 coded index counter (from previous section) would be extremely area inefficient and slow. This section presents efficient error resilient counter architectures. The major difference between a state machine and a counter is the probability of determinism of the consecutive states. The sequence of an index counter is always predetermined and fixed, whereas the state machine could have multiple sequence trajectories. This key distinction is used to develop efficient index counters.

A high speed index counter is usually realized as a ring counter. Ring counters offer some advantages over binary counters; these require an adder circuit, which is substantially more complex. Additionally, the worst-case propagation delay on an adder circuit is proportional to the number of bits in the code (due to the carry propagation). The complex combinational logic of an adder can create timing errors which may cause erratic hardware performance. The ring counter propagation delay is constant regardless of the number of bits in the code. The registers cycle through a sequence of one hot coded bit-patterns. The MOD of the ring counter is n, if n flip-flops are used.

It is proposed to use the ring counter to implement the LSBs or Least Significant bits of the index counter. This

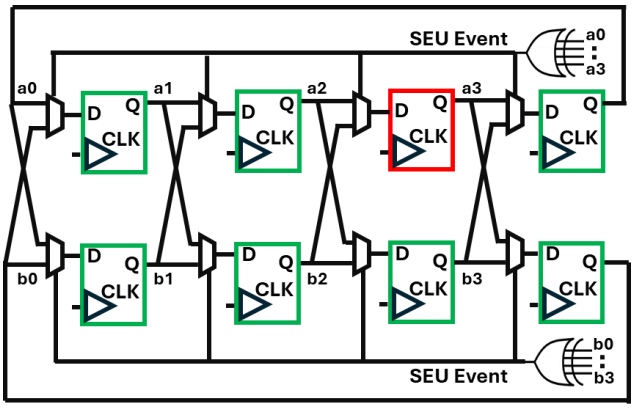

Fig 4. DMR Ring Counter

reduces the area overhead drastically as the majority of the counter is implemented as a grey counter.

For example, in a high speed 256:1 multiplexor serializer running at fs=3GHz, a high speed 8b counter is needed. This is a challenge to implement even without any redundancy or error correction. The LSB of the counter will toggle at fs, the LSB-1 at fs/2, LSB-3 at fs/4 and so on. The counter can be partitioned into two parts, a high speed section and a low speed section. The high speed section can be implemented as a ring counter. As an example, a 4 bit ring counter is used to implement the 2 LSBs of the 8-bit counter. Fig. 4 illustrates one such self-correcting counter. The ring counter registers are connected in a closed loop shift register arrangement. A shadow ring counter runs concurrently with the main counter. This could be labelled as a DMR configuration with a dual redundancy. Contemporary DMR circuits bring the system to a reset state during an SEU event. The unique feature of the proposed DMR configuration is that it is not only able to detect, but also correct any SEU error in real time, thus avoiding any reset or restart condition.

The proposed ring counter monitors the ring flip-flops in real time for any deviations from its primary property of holding a single '1'. In the 4 bit ring counter of Fig. 4, the legal state at any point in time is 3 zeros and 1 one. During an SEU event, the two possible illegal configurations are all zeros or two ones. More rigorously, the 4 bit ring counter has 4 legal states and 12 illegal states. The ring counter register outputs (a0..a3 and b0..b3) are continuously monitored in real time by an XOR gate. During an SEU event, any deviation of the ring counters from any of their 4 legal states is detected immediately by an XOR gate which triggers a transfer of legal states from the clone ring counter running in parallel. The critical path overhead of this circuit is an XOR gate and a 2:1

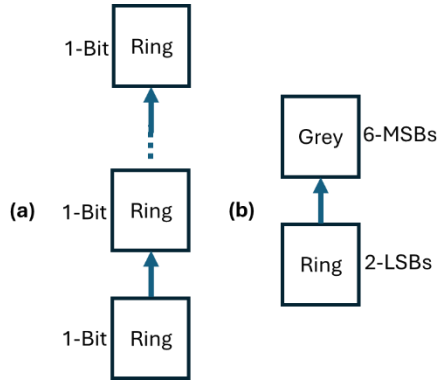

Fig 5. Counter Partitioning

Index	6b GC	Parity
0	000000	0
1	000001	1
2	000011	0
3	000010	1
4	000110	0
5	000111	1
6	000101	0
7	000100	1

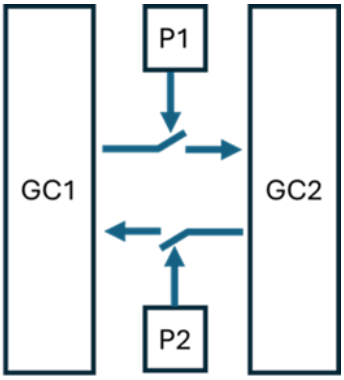

Fig 6. DMR Grey Counter

multiplexer. The critical path can be further reduced by splitting the present 4-bit ring counters to two stages of 2-bit ring counters. This would reduce the critical path and the high speed circuit by half. Another possible solution is to configure multiple 2-bit ring counters as asynchronous counters. Fig. 5(a) illustrates such a configuration where the asynchronous interface will select output of the non-effected counter to drive the next stage. This can be extended to 8 stages for the 8-bit counter example. The successive stages of this asynchronous ring counter arrangement will run at lower frequency, saving substantial power. But this could be a problem where synchronous controls are required.

A higher reliability synchronous solution would be to drive a 6b grey counter with a 2b LSB ring counter. As seen in Fig. 6, the 6 MSBs or most significant bits of the example 8-bit counter are proposed to be implemented as a contemporary grey counter configured as a dual redundancy module comprising GC1 and GC2. Initial eight of the 6b grey codes are shown in Fig. 6. Only one output bit is ever toggling at a time in a Gray code "counter", as opposed to possibly multiple bits in a binary counter. The result is that Gray code counters consume only half the power of an equivalent binary counter and they generate correspondingly less noise. Actually, while the power (and average noise) difference between a Gray and a binary counter asymptotically approaches two, the peak noise difference is equal to the number of bits, since a Gray counter toggles only one bit at a time while a binary counter toggles all of its bits simultaneously two times over the course of a full-count cycle with fewer bits toggling proportionally more times. We can use the property of a grey counter of a single bit change for every consecutive code to build a self-correcting DMR system. Secondly, the 6-bit grey counter will operate at one-fourth of the full speed counter frequency fs. This will relax the implementation of the grey counter and its critical paths.

As shown in Fig. 6, it is proposed to use two identical grey counters GC1 and GC2 running in parallel. In case of an SEU event, one of the grey counter registers will be subject to a bit-flip. The faulty bit-flipped counter is loaded in real time with the clone counter running in parallel. The fault is detected by monitoring the parity of the grey counter outputs. The table of Fig.6 lists initial few 6b gray codes (GC) along with their parity. We note that the parity of a grey counter alternates between one and zero every alternate cycle. We can keep a track of this parity change every cycle and detect any break in the alternating pattern. In case of a deviation, the faulty grey counter registers are replaced with the ones from the shadow grey counter running in parallel. This ensures that the two grey

counters monitor consecutive parities and replicate contents of the unaffected counter into the faulty one. This ensures real time tracking and correction of a DMR system comprising two grey counters

III. Conclusions

Widely used control topologies were proposed for SEU resilience using minimal overheads in terms of power, performance, area and reusability. DMR circuits were proposed and shown to perform operations usually executed by contemporary TMR methods.

Testing the possible four different state machine encodings with multiple SEU fault injections gives an indication of which encoding method is preferred. As expected, Hamming-3 (H-3) encoding gave by far the best fault tolerance, recording 0 errors in single fault injection tests. However, it requires relatively higher resources, and is the slowest of the encoding methods. Hamming-2 (H2) encoding has less errors than binary encoding, and has zero of the critical false-positive errors. One-hot encoding has the most errors, due to its large number of target flip-flops. One-hot also shows poor use of resources, and slow speed, although it is better than binary for false-positive errors. For fault tolerant designs H-2 is the best compromise in terms of size, speed and fault-tolerance and should always be preferred over both binary and one-hot state machine encoding. H-3 encoding is robustly fault tolerant to single faults and is preferred when ultimate reliability is required in a critical application.

The determinism of index counter outputs is used to architect multiple structures that can be suited for various applications. Ring counter and grey counter based DMR circuits are proposed to track and correct SEU errors in real time. It is shown that the critical paths and area overheads of the proposed structures is minimal. All combinational and sequential cells used in the error resilient circuits are from the usual standard cell technology libraries and don't need any special treatment.

References

[1] K. M. Chavali, "Mitigation of Soft Error Rate using Design, Process and Material Improvements," *2019 IEEE International Integrated Reliability Workshop (IIRW)*, South Lake Tahoe, CA, USA

[2] R. Rajaei, M. Niemier and X. S. Hu, "Low-Cost Sequential Logic Circuit Design Considering Single Event Double-Node Upsets and Single Event Transients," *2021 IEEE 39th International Conference on Computer Design (ICCD)*, Storrs, CT, USA

[3] R. Reis, C. Meinhardt, A. L. Zimpeck, L. H. Brendler and L. Moraes, "Circuit Level Design Methods to Mitigate Soft Errors," 2020 IEEE Latin-American Test Symposium (LATS), Maceio, Brazil

[4] Y. Bai, S. Yu, L. Ma, N. Wang, J. Sun and X. Chen, "A Novel Latch Circuit against Single Event Upset," *2020 International Conference on Artificial Intelligence and Electromechanical Automation (AIEA)*, Tianjin, China

[5] T. Uemura, S. Lee, D. Min, I. Moon, S. Lee and S. Pae, "SEIFF: Soft Error Immune Flip-Flop for Mitigating Single Event Upset and Single Event Transient in 10 nm FinFET," *2019 IEEE International Reliability Physics Symposium (IRPS)*, Monterey, CA, USA, 2019

[6] Y. Chen et al., "Physical Insight into Single-Event Upsets of DICE Circuits and Hardening Strategy," *2024 2nd International Symposium of Electronics Design Automation (ISEDA)*, Xi'an, China

[7] J. Liu, X. Li, J. Zhang and J. Li, "An Area-Efficient Design of Enhanced Space-Time Redundant DFF (IEST_TMR DFF)," *2022 7th International Conference on Computer and Communication Systems (ICCCS)*, Wuhan, China

[8] S. Kasap, E. W. Wächter, X. Zhai, S. Ehsan and K. D. McDonald-Maier, "Novel Lockstep-based Approach with Roll-back and Roll-forward Recovery to Mitigate Radiation-Induced Soft Errors," 2020 *IEEE Nordic Circuits and Systems Conference (NorCAS)*, Oslo, Norway, 2020

[9] P. Sen, M. S. Sadi, N. Ashab and D. Rossi, "A New Error Correcting Coding Technique to Tolerate Soft Errors," *2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, Khulna, Bangladesh

[10] J. Li, P. Reviriego, L. Xiao and H. Wu, "Protecting Memories against Soft Errors: The Case for Customizable Error Correction Codes," in *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 2, pp. 651-663, 1 April-June 2021