Trading off Reliability and Power-Consumption in Ultra-Low Power Systems

Atul Maheshwari, Wayne Burleson and Russell Tessier
Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, MA 01003
E-mail: {amaheshw,burleson,tessier}@ecs.umass.edu

Abstract

Critical systems like pace-makers, defibrillators, wearable computers and other electronic gadgets have to be designed not only for reliability but also for ultra-low power consumption due to limited battery life. This paper explores architecture, logic and circuit level approaches to this tradeoff. Fault tolerance techniques at the architecture level can be broadly classified into spatial or temporal redundancy. Using an example of counters (Binary and Gray) we show that temporal redundancy is best suited for these ultra-low power and low performance systems as it consumes 30% less power than an area redundant technique. Circuit techniques allow power-reliability tradeoffs of about 50% in each measure. A methodology is developed based on low-level fault simulation using SPICE, which allows detailed circuit models for both power consumption and reliability in current and future CMOS technology.

1. Introduction

Critical systems, ranging from medical applications like pace-makers and defibrillators to the emerging wearable wrist-watch computers, utilize embedded batteries. Due to the limited battery life along with the fact that in many of these systems batteries cannot be replaced or recharged, the circuits used in these systems should be extremely low-power. At the same time the criticality of these systems warrants the use of fault-tolerance techniques to ensure reliable functioning. CMOS technology scaling trends are also resulting in circuits with higher power consumption and lower reliability. This paper uses 4-bit real-time counters as an example to study fault-tolerance techniques at several levels and illustrate the tradeoff between power consumption and reliability of a system.

Traditional fault tolerance schemes like Triple Modular Redundancy (TMR) provide a high degree of fault-tolerance while maintaining performance at the cost of more than two times the area and power consumption. Most of the above mentioned systems are relatively low performance but are power critical and hence motivate the study of the impact on power consumption due to incorporation of fault-tolerance.

The remainder of paper is organized as follows: Section 2 discusses two logic implementations of counter which were considered for this study. Section 3 discusses the metrics and methodology used to measure fault-tolerance and power. Section 4 discusses methods of improving these counters by means of adding redundancy (architecture level) and by redesigning some circuit blocks (circuit level). Finally, Section 5 discusses some conclusions and future work.

2. Counters

In this section we discuss the implementation of Binary and Gray counter which were considered for this study. Fault detection techniques for these counters are also discussed. Binary and Gray counters are studied in this paper but the same technique can be extended to almost any circuit. Counters were taken as an example for this study as they are an integral part of these systems and may be used as a timer or as a address generator as they provide two different state encoding of a counter.

2.1. Binary Counter

The Binary counter is the simplest and most commonly used counter. Goutis in [1] proposed a technique for making the Binary counter fault secure. A property of the T flip-flop was used for detecting errors. Figure 1 shows the circuit implementation of a T flip-flop based counter with fault detection logic.

Typically in CMOS, Binary counters are implemented as shown in Figure 2 rather than using a T flip-flop. Since this implementation uses a D-flip flop (or a latch) and not T-flip flop, the algorithm proposed in [1] cannot be directly applied to this counter. However an alternate scheme can be used where the parity of the enable signals (EN_1, EN_2 ... EN_n) indicate

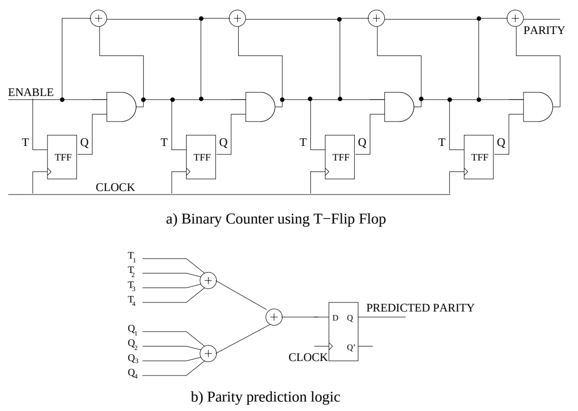


Figure 1: **Binary counter with fault detection using T-Flip ¤op**

whether the parity of the next state will change or not. Hence instead of the T_1, T_2 ... T_n signals in figure 1(b), signals EN_1, EN_2 ... EN_n can be used to detect a fault. It can be seen that Figure 1 and Figure 2 are similar implementations of the Binary counter except for the memory element. We are going to consider the implementation shown in Figure 2 for this study as it is a popular implementation and also makes it easy to compare Binary counters with Gray counters.

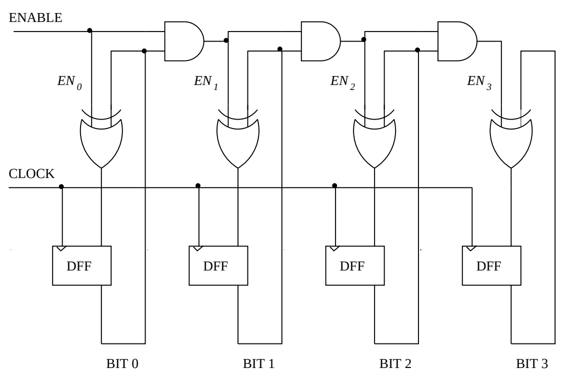


Figure 2: **Binary counter using a D-Flip ¤op**

2.2. Gray counter

Binary counters are not intuitively very power efficient since there are multiple transitions when switching from one state to the next. The Gray counter can be lower-power, since there is transition in only one flip-flop when switching states. This reduction in switching activity can save a significant amount of power although intermediate logic may have more transitions.

Use of Gray code counters has been proposed to save power in the control path of embedded processors [2]. Work has been done to prove the advantages of Gray code addressing in the context of bit changes on the address lines and hence a reduction of switching activity by 30-50% during normal program execution using a Gray code counter [3]. However, implementing a Gray counter is not as simple as designing Binary counters. A significant amount of logic is required to determine the next state of the counter. This logic is usually $(n - 1)$ gates deep for a n-bit counter and depends on all the bits of the previous state and their complements. All this logical complexity might end up making the Gray counter more power hungry. As shown in Figure 3, a segmented Gray counter[4] can be used to reduce the amount of logic required with just a slight increase in the switching activity. Detecting faults in a Gray counter is very easy as with every state change the parity of each segment keeps toggling. A circuit implementation of the Gray counter is shown in Figure 4.

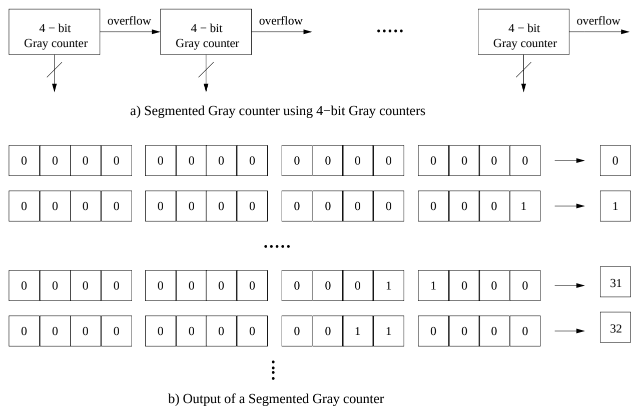
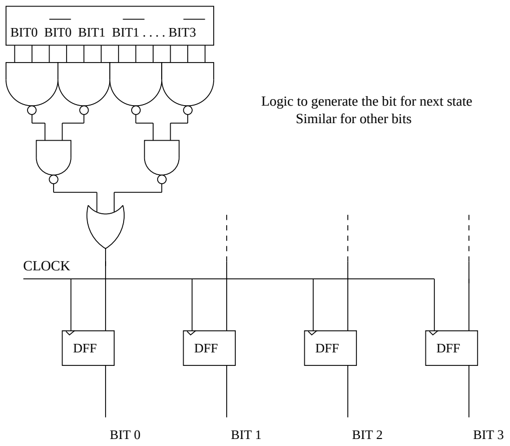

4 − bit Gray counter
overflow
.....
a) Segmented Gray counter using 4−bit Gray counters
0
1
31
32
b) Output of a Segmented Gray counter

Figure 3: **Segmented Gray counter**

3. Fault model and fault sensitivity analysis methodology

Faults can be broadly classified into *permanent* and *transient* faults. Permanent faults are usually caused during the manufacturing stage while transient faults are caused in the field. Reasons for transient fault include radiations, cross talk and power transients. It has been shown that 80% of system failures is due to transient faults [5, 6]. Hence, for this study we are considering only transient faults.

Several fault models for transient faults have been proposed [7, 8]. For the purpose of this study we are using the model presented in [7], which models any



BIT0 BIT0 BIT1 BIT1 .... BIT3
Logic to generate the bit for next state
Similar for other bits
CLOCK
DFF
DFF
DFF
DFF
BIT 0
BIT 1
BIT 2
BIT 3

Figure 4: **Implementation of a Gray counter**

transient which results in collection of charge on an active node (eg. $\alpha - particles$). The transient is modeled as a double exponential injection current given by :

$$I_{inj}(t) = I_0(e^{-t/\tau_1} - e^{-t/\tau_2}) \qquad (1)$$

where I_0 is the maximum current, τ_1 is the collection time constant for a junction and τ_2 is the ion track establishment time constant.

Using this fault model a fault sensitivity analysis can be done to compare two types of counter implementations. Recent work [9] has shown that the fault sensitivity analysis for an α-particle induced transient can be performed at an early stage in the design cycle of VLSI circuits.

A metric called the Probability of Failure (POF) has been proposed in [9]. The POF is given by

$$POF = \sum_{i=1}^{n} w_i \bar{E}_i \quad , \qquad w_i = \frac{A_{s,i}}{\sum_{i=1}^{n} A_{s,i}} \qquad (2)$$

where $A_{s,i}$ is the area of the node i. $\bar{E}_i$ is given by

$$\bar{E}_i = \frac{1}{k}\sum_{i=1}^{k} E_i \quad , \qquad k = p \cdot q \cdot r \qquad (3)$$

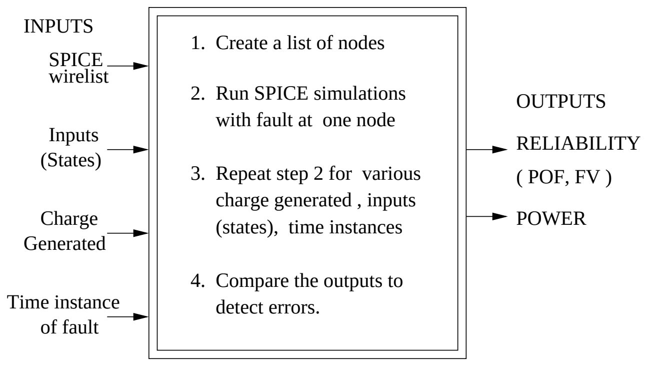
where p is the number of states(2^n for n-bit counter) for which the simulation was performed, q is the number of injection levels considered and r is the number of time instances at which faults were injected. E_i, the outcome of a fault injection experiment is given by

$$E_i = \begin{cases} 1 & \text{if the injection into node } i \text{ results in} \\ & \text{a faulty output} \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

The number of α-particles hitting a circuit is directly proportional to the active area of the circuit (α-particles hit does not create free charge carriers if it hits a non active area [10]). Hence, to compare different implementations of the counter, active area of the counter should be taken into account. Hence we define a new metric called Fault Vulnerability (FV) as

$$FV = POF \; * \; Active\ area\ of\ circuit \qquad (5)$$

To measure POF and FV a scripted set of simulations were used to perform exhaustive analysis. The overall tool flow is shown in Figure 5.



INPUTS
SPICE wirelist
Inputs (States)
Charge Generated
Time instance of fault
1. Create a list of nodes
2. Run SPICE simulations with fault at one node
3. Repeat step 2 for various charge generated , inputs (states), time instances
4. Compare the outputs to detect errors.
OUTPUTS
RELIABILITY (POF, FV)
POWER

Figure 5: **Tool ¤ow for measuring Reliability and Power**

4. Improving fault tolerance of counters

Exhaustive simulations shown in the tool flow (Figure 5) were performed using the above mentioned fault sensitivity analysis method on a 1.2μ CMOS process. All the nodes of the counter were considered for a single fault and all the possible states of a 4-bit counter were covered. Several injection levels were also considered. Simulation results of the counters without any optimization is shown in table 1.

Counter	POF	FV	Power
Binary	0.178	127.6	0.052mW
Gray	0.080	96.6	0.038mW

Table 1: **Comparison of counter without any optimization**

Two approaches were considered for improving fault tolerance of the counter. First method involves incorporating redundancy in the counter to make them fault-tolerant. The second method looks at redesigning some of the blocks to make the counter fault-tolerant.

4.1. Incorporating redundancy in the counters

In general spatial(area) or temporal(time) redundancy can applied to a circuit to make them fault-tolerant. Area redundancy can be applied to counters in a very simple form of Dual Modular Redundancy(DMR) (Figure 6) or Triple Modular Redundancy(TMR) wherein redundant counters are added and the final output is the majority vote of these counters. Clearly adding any type of area redundancy increases power consumption significantly (by a factor of 2 or 3).

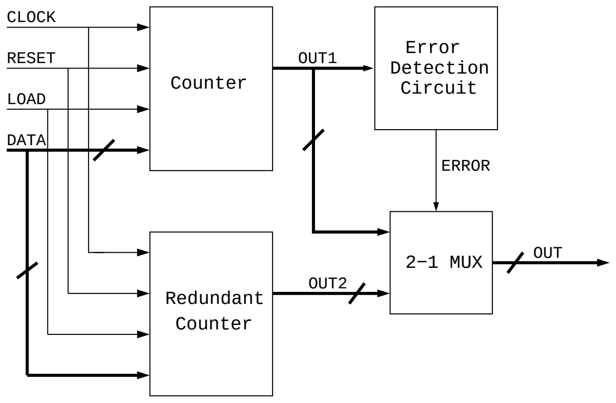


Figure 6: **Architecture of area redundant (DMR) counter**

We propose to incorporate *time-redundancy* in the counters. This approach limits hardware overhead, leading to reduced power consumption. In our *time-redundant* technique, an errant output is recomputed in an attempt to recover from transient faults.

Figure 7 illustrates our implementation of the *time-redundant* technique. During normal operation the counter output is stored in an enabled register. In case of an error, the previously-stored correct value is loaded into the counter. The counter is then given a clock to recompute the next state. The additional hardware required includes an enabled register, multiplexers, and control logic for load and clock signals. Power is saved since error recovery logic is used only **after** an error has been detected. Thus, this technique is appropriate for ultra-low power and low performance systems.

For single faults, area and time redundant techniques discussed above result in 100% reliability. The power consumed by these implementations is shown in table 2.

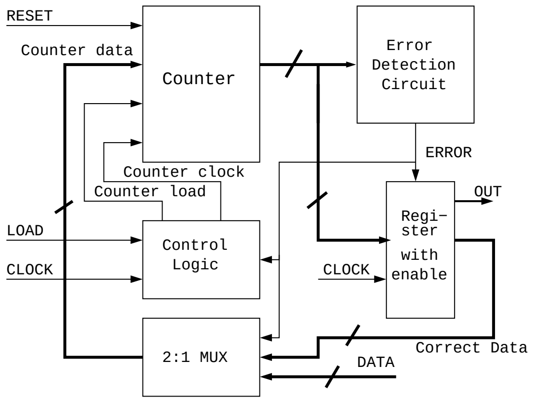


Figure 7: **Architecture of time redundant counter**

Counter Type	Power
Area Redundant Binary Counter	0.1123mW
Time Redundant Binary Counter	0.0822mW
Area Redundant Gray Counter	0.0806mW
Time Redundant Gray Counter	0.0623mW

Table 2: **Power comparison of area and time redundant counters**

4.2. Circuit-level fault tolerance

A counter consists of a set of memory elements (flip-flops) which holds the current state of the counter and logic which determines the next state based on the previous state. Since flip-flops are bistable elements, they are more sensitive to transients than the logic gates. Hence, any attempt to redesign counter components should start from the flip-flop. A Transient Pulse Tolerant Latch (TPTL) proposed in [11] can be used to improve the fault tolerance of the flip-flop and hence of the counter. RC circuit helps in filtering out the high frequency transients(Figure 8). Several RC values were used (ranging from 5-20K ohm, 75ff-300ff) to see the effect of TPTL on fault-tolerance and power (Table 3).

Results show that with increase in resistance and capacitance the reliability of the counter increases at the cost of increased power (Figures 9 and 10). The gain in terms of reliability however saturates at about 10K ohm and any further resistance increase results in additional power cost.

Another technique which might provide improvement in fault-tolerance is to increase the size of transistors [12]. Increasing the size of the transistor reduces the magnitude of the offset node voltage of the

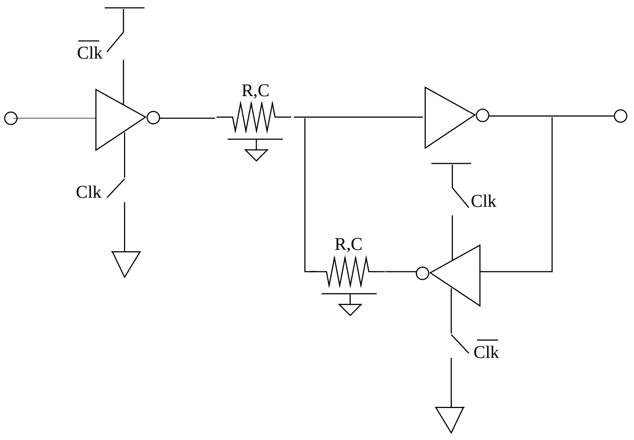


Figure 8: **Transient Pulse Tolerant Latch (TPTL)**

Counter		POF	FV	Power
Binary	Normal	0.178	127.6	0.052mW
	5K ohm	0.152	109.0	0.058mW
	10K ohm	0.103	73.8	0.065mW
	20k ohm	0.100	71.7	0.073mW
Gray	Normal	0.08	96.6	0.038mW
	5K ohm	0.062	74.9	0.044mW
	10K ohm	0.053	64.0	0.051mW
	20k ohm	0.052	62.8	0.061mW

Table 3: **Comparison of counters using various resistance values**

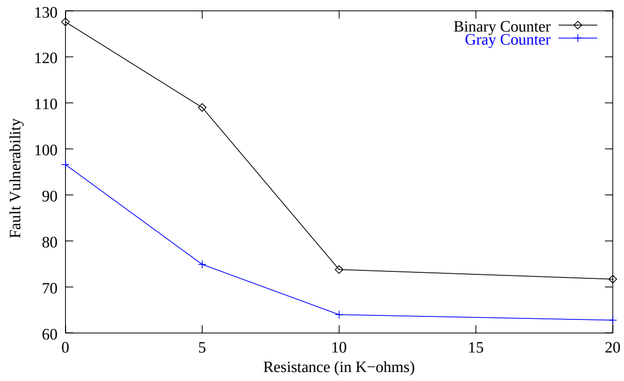


Figure 9: **Effect of resistance on reliability**

transient and thus improves reliability. Only selected highly sensitive nodes should be resized to improve the fault-tolerance without increasing the area significantly. Critical nodes were sized by a factor of 2-6 and their effect on fault-tolerance and power was studied (Table 4). This technique resulted in an area penalty

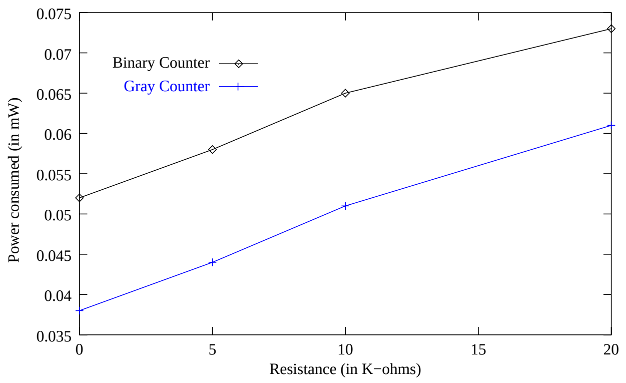


Figure 10: **Effect of resistance on power**

ranging from 7% to 20%.

Counter		POF	FV	Power
Binary	Normal	0.178	127.6	0.052mW
	2X	0.147	111.7	0.066mW
	4X	0.080	64.2	0.082mW
	6X	0.040	33.8	0.098mW
Gray	Normal	0.08	96.6	0.038mW
	2X	0.055	70.3	0.055mW
	4X	0.029	39.2	0.072mW
	6X	0.015	21.3	0.088mW

Table 4: **Comparison of counters using various transistor sizes**

Results show that the reliability of the counter increases with increase in the size of the transistors, again at the cost of increased power (Figures 11 and 12). A significant gain in reliability is observed as the size of the critical devices is increased. If the size of devices is further increased, reliability will improve but it might not be power efficient to increase the size of the devices beyond a certain limit. A sizing factor can be determined based on the power budget for the system (or the circuit).

5. Conclusions and Future work

From this work we can conclude that conventional low-power and fault-tolerance design techniques are at odds (Figure 13) and some guidelines and novel design techniques are required to address both objectives simultaneously. At the architecture level *time redundancy* should be employed wherever possible to improve the fault-tolerance of the system. It was shown that time redundant techniques end up consuming up

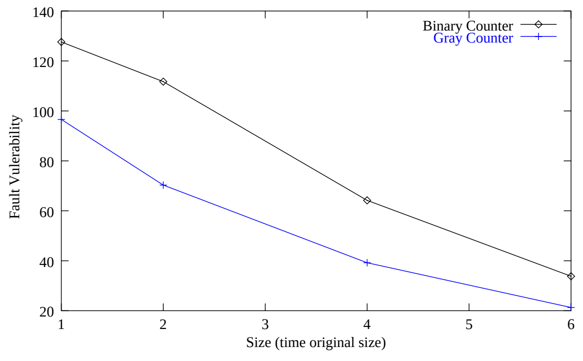


Figure 11: **Effect of sizing on reliability**

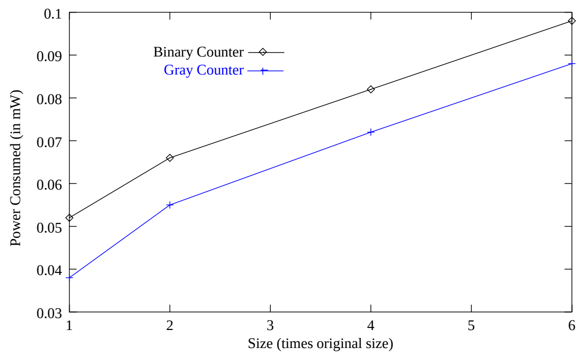


Figure 12: **Effect of sizing on power**

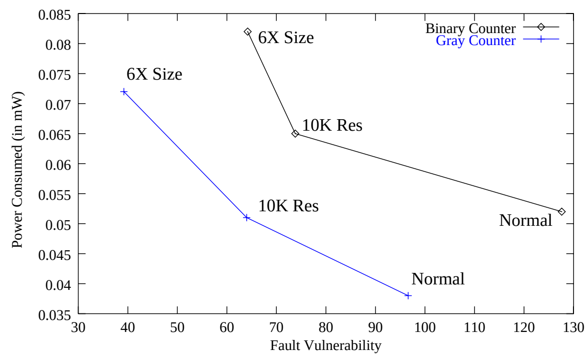


Figure 13: **Power and reliability trade off**

to 35% less power as compared to the area redundant technique. Two types of circuit techniques were discussed and a need for exploring other techniques was motivated. Circuit techniques allow power-reliability tradeoffs of about 50% in each measure.

Future work involves developing novel circuit techniques which address both the low-power and fault-tolerance issues. A faster fault-simulator based on logic simulation will help in making this analysis feasible for larger circuits. To perform the study for modern and future technologies there is a need to develop fault models for latest technologies. Also many of the ultra-low power systems contain fault-sensitive mixed-signal circuits and hence it is important to perform similar studies for mixed-signal circuits.

Acknowledgment: This work was supported in part by Sharp Corporation.

6. References

[1] Karaolis E., Nikolaidis S., Goutis C.E., "Fault Secure Binary Counter Design," in *IEEE International Conference on Electronics, Circuits and Systems*, 1999, pp. 1659-1662.

[2] Su C. L., Tsui C. Y. and Despain A. "Saving power in the control path of embedded processors," in *Proceedings of IEEE Design and Test of Computers*, 1994, pp. 24-31.

[3] Mehta H., Owens R. M. and Irwin M. J. "Some issues in Gray code addressing," in *Proceedings of Sixth Great Lake Symposium on VLSI*, 1996, pp. 178-181.

[4] Hakenes R. and Manoli Y., "A segmented Gray code for low-power microcontroller address buses," in *EUROMICRO Conference*, 1999, pp. 240-243.

[5] Ball H. and Hardy F., "Effects and detection of intermittent failures in digital systems," in *Proceedings of FLCC, AFIPS conference*, 1969, pp. 329-335.

[6] Iyer R. and Rosetti D., "A statistical load dependency of CPU errors at SLAC," in *Proceedings of FTCS*, 1982.

[7] Messenger G. C., "Collection of charge on junction nodes from ion tracks," in *IEEE Transactions on Nuclear Science*, 1982, pp. 2024-2031.

[8] Laguna G. and Treece R. "VLSI modeling and design for radiation environments," in *Proceedings of IEEE International Conference on Computer Design*, 1986, pp. 380-384.

[9] Singh M., Rachala R., and Koren I., "Transient Fault Sensitivity Analysis of Analog-to-Digital Converters (ADCs)," in *IEEE Annual Workshop on VLSI*, Apr. 2001, pp. 140-145.

[10] Kang S. and Chu D., "CMOS circuit design for the prevention of single event upset," in *IEEE International Conference on Computer Design*, 1986, pp. 385-388.

[11] Cha H., and Patel J.H., "A Logic-Level Model for α-Particle Hits in CMOS Circuits," in *Proceedings of IEEE International Conference on Computer Design*, Oct 1993, pp. 538–542.

[12] Singh M., and Koren I., "Reliability Enhancement of Analog-to-Digital Converters (ADCs)," in *IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, Oct. 2001.