Trading Off Transient Fault Tolerance and Power Consumption in Deep Submicron (DSM) VLSI Circuits

Atul Maheshwari, *Student Member, IEEE*, Wayne Burleson, *Senior Member, IEEE*, and Russell Tessier, *Member, IEEE*

***Abstract*—High fault tolerance for transient faults and low-power consumption are key objectives in the design of critical embedded systems. Systems like smart cards, PDAs, wearable computers, pacemakers, defibrillators, and other electronic gadgets must not only be designed for fault tolerance but also for ultra-low-power consumption due to limited battery life. In this paper, a highly accurate method of estimating fault tolerance in terms of mean time to failure (MTTF) is presented. The estimation is based on circuit-level simulations (HSPICE) and uses a double exponential current-source fault model. Using counters, it is shown that the transient fault tolerance and power dissipation of low-power circuits are at odds and allow for a power fault-tolerance tradeoff. Architecture and circuit level fault tolerance and low-power techniques are used to demonstrate and quantify this tradeoff. Estimates show that incorporation of these techniques results either in a design with an MTTF of 36 years and power consumption of 102 μW or a design with an MTTF of 12 years and power consumption of 20 μW. Depending on the criticality of the system and the power budget, certain techniques might be preferred over others, resulting in either a more fault tolerant or a lower power design, at the sacrifice of the alternative objective.**

***Index Terms*—Fault sensitivity estimation, fault-tolerance techniques, low-power techniques, transient fault model.**

I. Introduction

TRENDS in CMOS technology, applications, and operating conditions are resulting in circuits with higher power consumption and higher susceptibility to transient faults. In recent years, portable systems like cellphones, PDAs and smart cards have become integral parts of every day life. Many of these systems, such as smart cards, pacemakers, and defibrillators, have critical functionality that warrants the incorporation of fault tolerance. These portable systems rely on embedded batteries as their source of power. Due to limited battery life and the inability of the user to replace or recharge the battery, circuits used in these systems should be extremely low power.

Information in these electronic circuits is stored and communicated as a collection of electric charges. Any event which upsets the stored or communicated charge can cause errors in the circuit output. These errors are called transient faults, soft errors (SE) or single-event upsets (SEU). The event causing the upset can be an energetic nuclear particle or an electrical source. The nuclear particles which create these upsetting events are either cosmic rays which bombard the earth constantly from space or radioactive atoms which exist in trace amounts in all materials due to atomic decay. Atmospheric nuclear particles include *alpha-particles* [2], protons [3], and neutrons [4]. Electrical sources are power supply noise, electromagnetic interference (EMI) or radiation from lightning [1].

Due to their spatial density and the amount of information they store, memories are considered most vulnerable to transients. Memories were the first circuits studied for alpha particle induced SE [2]. Since then, several studies have been performed on SE in memories [5]–[8]. Recently, it has been demonstrated that it is important to consider memory arrays and core logic when estimating microprocessor soft error rate [9]. SE and SEU are considered challenges for high performance and low-power microprocessor design [10]. Hence, it is imperative to consider the total design when estimating or mitigating circuit soft error rate.

Power consumption is a key issue in the design of portable systems. Increased power consumption leads to increased battery size and an increase in product size and weight. Any increase in power consumption for these systems is highly discouraged and methods of reducing power must be explored and incorporated.

In this paper, a methodology to estimate the transient fault tolerance of a circuit in terms of *mean time to failure* (MTTF) is presented. A study of the relationship between circuit fault tolerance and power consumption is performed using this methodology. Both architecture and circuit-level fault-tolerance techniques are studied and a 4-bit real-time counter is used as a circuit example. The counter serves as an effective example since it is a simple finite-state machine (FSM) and is often included in embedded systems. Our analysis can be extended to any FSM or sequential circuit.

The paper is organized as follows: Section II discusses the model used to represent the transients. Section III describes the metrics and methodology used to estimate fault tolerance and power. Section IV discusses the logic implementation of counters used in our analysis. Section V discusses methods of improving counter fault tolerance with architecture-level redundancy and current-block redesign and the impact on power. Section VI studies the impact of standard low-power design techniques on circuit fault tolerance. Finally, Section VII discusses conclusions and future work.

Manuscript received January 28, 2003; revised May 31, 2003. This work was supported in part by the Sharp Corporation and in part by the Semiconductor Research Corporation under Task 766, Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA.

A. Maheshwari and W. Burleson are with the Interconnect Circuit Design Group, Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA (e-mail: amaheshw@ecs.umass.edu; burleson@ecs.umass.edu).

R. Tessier is with the Reconfigurable Computing Group, Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA (e-mail: tessier@ecs.umass.edu).

Digital Object Identifier 10.1109/TVLSI.2004.824302

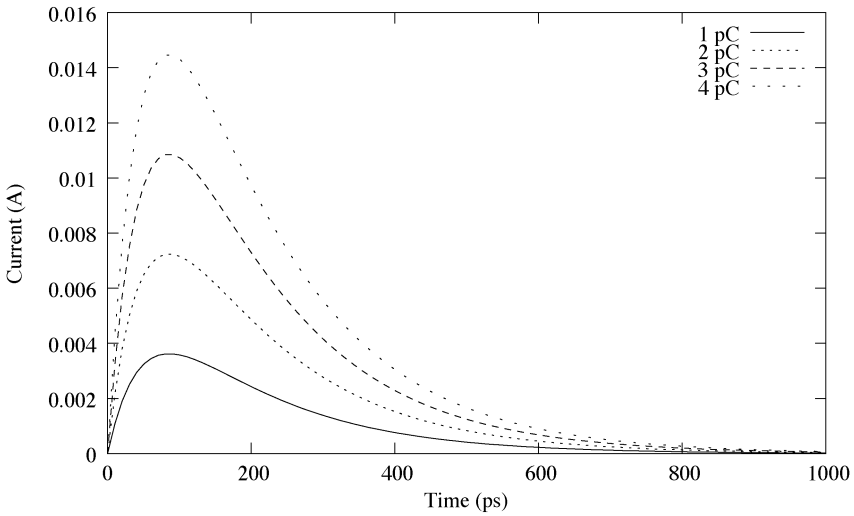

Fig. 1. Current pulse generated as a result of an α-particle strike at time $t = 0$.

II. Fault Model

Faults can be broadly classified into *permanent* and *transient* faults. Permanent faults usually occur during manufacturing and can be repaired while transient faults generally occur in the field. It has been shown that 80% of system failures are due to transient faults [15], [16]. For this paper, we only consider transient faults.

Transients can be represented at the device level by a current or a voltage source. These models accurately represent the electrical impact of the transient. Device-level models of cosmic-particle induced transients have been developed [17], [18]. In [17], a SPICE circuit with a current source was used to represent the collected charges generated by alpha particles. An approximate analytic solution which models a current transient is proposed in [19]. The model includes parameters which represent the maximum current, the collection time constant of the junction, and the time constant for initially establishing the ion track. Fig. 1 shows the current pulse model for a 1.2μ technology for various charge injection levels (in pC). The current injections in Fig. 1 can be used to represent any transient by changing the rise time, fall time, and the peak current of the model.

At the logic-level, transients can be modeled as a momentary *bit-flip* of the propagating signal. Simulations using gate-level transient faults are shown in [20], and a fault simulation method at the register-transfer-level (RTL) is proposed in [21]. In the RTL method, transients are modeled as temporary changes in logic values in circuit memory elements. The corrupted values are either overwritten or propagated causing errors in other parts in the system. A timing simulation technique that approximates the device-level fault waveform has also been proposed [22], [23].

Logic-level approaches are inherently faster than device-level approaches since they do not rely on the evaluation of circuit equations. However, these approaches may not be very accurate. A transient can propagate along multiple paths and cause multiple latch errors. The chance of a faulty pulse propagating to a latch and becoming a latch error is a function of device-level parameters. Moreover, the shape of the pulse may be changed in transit through different gates in the propagation path. It has been shown that a discrete logic-level fault model can result in a 50% error when used to estimate SE [24]. For the purpose of this study, we use a device-level fault model.

Several fault models for transient faults have been proposed [19], [25]. For this study, we use the model presented in [19], which models any transient resulting in a collection of active node charge. This model is preferred over others since it can be used to represent other transients by changing model parameters. This work, however, is limited to *α-particles*. In this model, a transient is modeled as a double exponential injection current

$$I_{\text{inj}}(t) = I_0 \left(e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_2}} \right) \tag{1}$$

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. τ_1 is dependent on the doping concentration and, hence, on the process. τ_2 is relatively independent of the technology. I_0 depends on the process and the charge intensity. Fig. 2 shows the phenomena of an *α-particle* hit on a pMOS transistor and the equivalent current injection model.

III. Fault Tolerance and Power Dissipation Estimation Methodology

A. Fault-Tolerance Estimation

Fault simulation and fault-tolerance quantification are the key components in developing a fault-tolerance estimation methodology. Fault simulation can be performed at several levels of design abstraction (i.e. RTL, gate level, transistor level, etc.).

The fault simulation abstraction level is determined by the fault model that is used. If a logic-level fault model is used, then the fault simulation is at the logic level. If the fault model is a device-level fault model, then fault simulation is at the circuit level. Mixed-mode simulation using both circuit and logic simulation has been proposed to speed up simulation [26], [27]. In the mixed-mode approach, device-level faults are injected using a circuit simulator such as SPICE. Subsequent logic-level errors are propagated to the gate and higher levels using logic simulation. The use of a gate-level simulator has also been proposed to further speedup fault simulation [20], [23]. These simulation methodologies are not very accurate due to fault-model limitations.

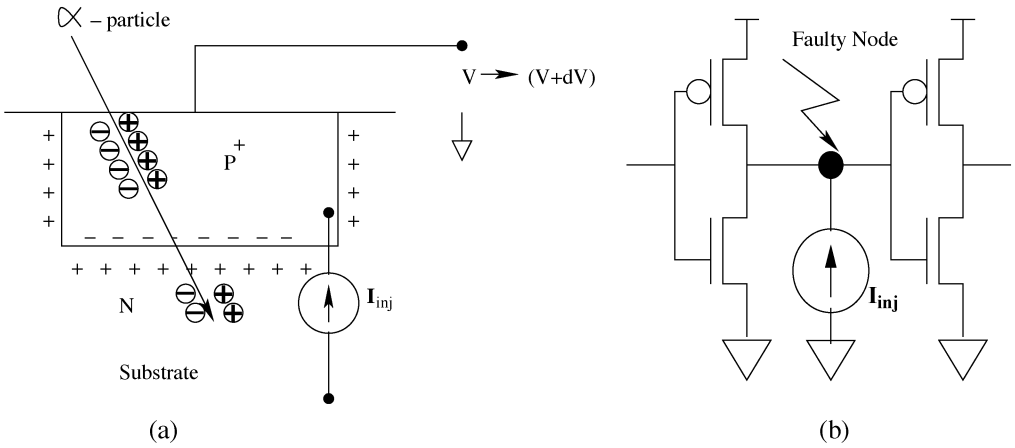

Fig. 2. (a) α-particle hit on a pMOS transistor and (b) the hit modeled as a current source.

The core of our fault simulation approach consists of HSPICE [28] based circuit-level simulations. Recent work [29] has shown that the fault-sensitivity analysis for an alpha-particle induced transient can be performed at an early stage in the design cycle of very large scale integration (VLSI) circuits. Layout level designs are not mandated, as a particle hit creates free charge carriers only if hits occur in an active area [30].

In order to obtain a highly accurate measure of fault tolerance, exhaustive fault simulations considering SEU are performed. The simulation is performed for all possible inputs and states of the circuit, thus, considering them equally probable. The estimation methodology accounts for the fact that a particle strike is equally likely to create positive or negative charge of up to 4 pC. It is assumed that each node in the circuit is equally prone to be hit by an *α-particle*. Simulation accounts for differing active node areas while estimating fault tolerance. Since transients have equiprobable occurrence during a clock cycle, simulations are performed assuming transients occur at equidistant times in a clock cycle. The total number of simulations are given by

$$N = p \cdot q \cdot r \cdot n \tag{2}$$

where

- N total number of simulations;
- p number of input or state combinations;
- q number of particle injection levels considered;
- r number of time instances at which faults are injected;
- n number of nodes in the circuit.

While p and n depend on the circuit under study, q and r are fixed. Particles of charges up to ± 4 pC are considered, hence, $q = 8$. The injections were done at eight different time instances equally dividing the clock cycle, thus, $r = 8$.

A metric which quantifies fault tolerance, the probability of failure (POF_c), was proposed in [29]. The POF_c is given by

$$\mathrm{POF}_c = \sum_{i=1}^{n} w_i \bar{E}_i, \quad \text{where} \quad w_i = \frac{A_i}{\sum_{i=1}^{n} A_i}. \tag{3}$$

Here A_i is the area of the node i. $\bar{E}_i$ is given by

$$\bar{E}_i = \frac{1}{k} \sum_{i=1}^{k} E_i, \quad \text{where} \quad k = p \cdot q \cdot r. \tag{4}$$

E_i, the outcome of a fault injection experiment is given by

$$E_i = \begin{cases} 1, & \text{if the injection into node } i \text{ results in a fault getting latched} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

POF_c is thus a measure of the conditional probability of error given that a particle hits the circuit. By weighing the errors by node area it is possible to account for the higher likelihood of larger nodes being hit by a particle. When comparing two different designs, POF_c fails to account for the higher likelihood of a larger circuit being hit by a particle. A different measure is needed which is independent of circuit parameters.

Measurements have shown that the particle density at sea level (New York) is approximately 100 000/cm^2/yr [31]. The probability of failure per year (POF_y) can then be determined from the conditional probability of error (POF_c) as

$$\mathrm{POF}_y = \mathrm{POF}_c \cdot \text{area of circuit} \cdot 100\,000. \tag{6}$$

Since the particle hit event is memoryless, i.e., future particle hits do not depend on the present particle hit, MTTF is given by

$$\mathrm{MTTF} = \frac{1}{\mathrm{POF}_y}. \tag{7}$$

To estimate POF_c and MTTF, a scripted set of simulations was performed. The overall tool flow and corresponding program flow are shown in Fig. 3. The SPICE netlist of the circuit, the valid input patterns or states and the charge levels are the inputs to the script and the script outputs circuit POF_c and MTTF. This study is done in a 0.18μ technology and the parameters of the fault model (I_0 and τ_1) have been adjusted from the originally proposed model for 1.2μ technology.

B. Estimating Power Dissipation

In order to have an accurate estimate of power dissipation, HSPICE simulations are used. All possible input and/or state combinations are considered equally probable, and hence power dissipation is averaged over all such possible combinations. Care is taken to avoid all unreachable states. A clock frequency of 1 GHz is assumed and simulations are performed for a 0.18μ technology.

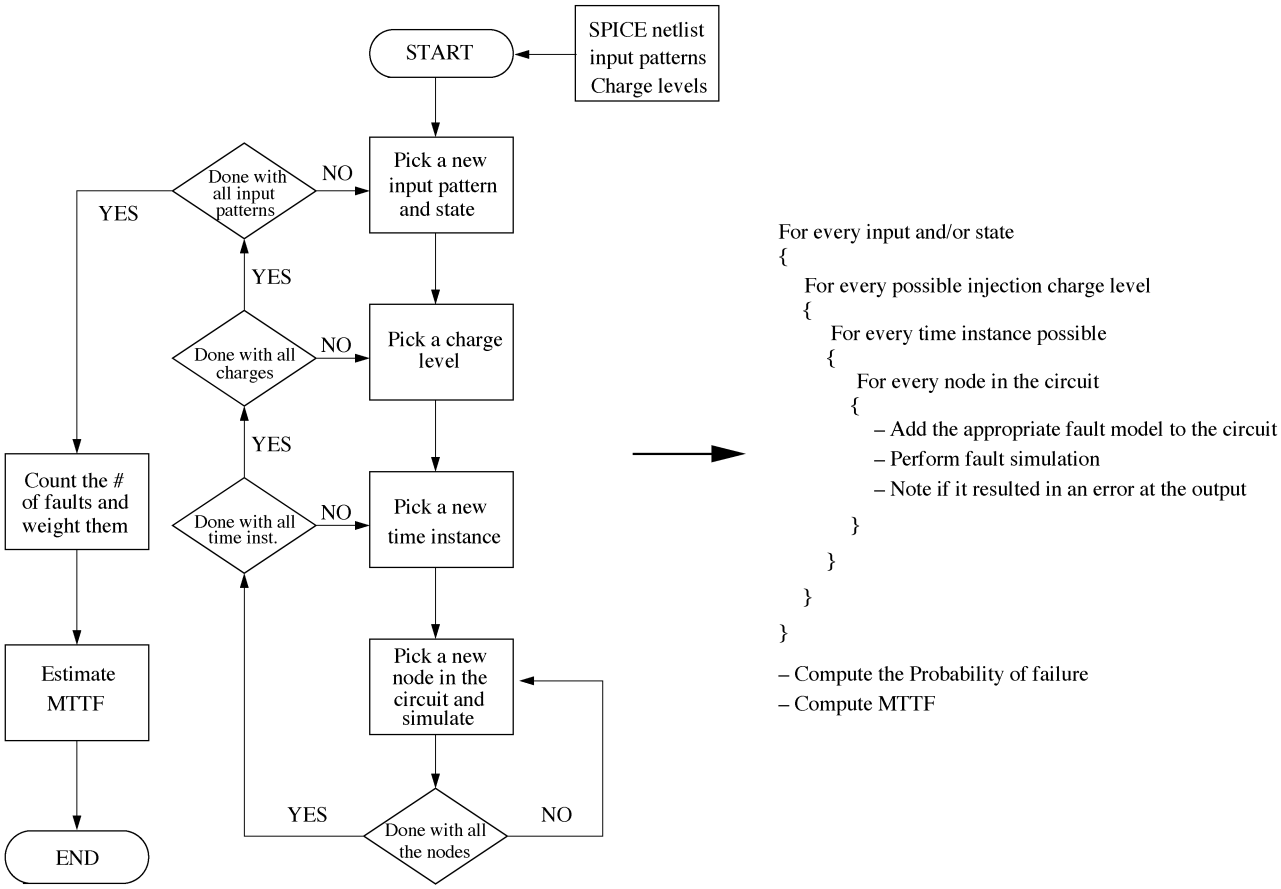

Fig. 3. Tool flow and corresponding program flow for measuring fault tolerance.

SPICE netlist used for power estimation does not include wire loads. Since the analysis is performed for small circuits, the effect of not considering the wire load does not have a significant impact on power dissipation estimation.

IV. The Example Circuit: Counters

In this section, the circuit implementation of the counters is discussed. Fault detection techniques for these counters is also discussed since these techniques are used to incorporate architecture-level fault tolerance. Two types of counters, i.e., binary and Gray counters, are studied in this paper.

These counters provide a realistic example since they often are timers and address generators in low-power embedded systems. As discussed in Section I, both storage elements and combinational logic should be considered in estimating the fault tolerance of a circuit. A counter consists of flip-flop storage elements which hold the present counter state, and combinational logic gates which compute the next counter state. The comparison of two counters demonstrates the impact of state coding and next-state generation logic on both power and fault tolerance.

Although counters are used as examples in this study, our technique can be applied to almost any circuit. However, the exhaustive simulation method used to estimate fault tolerance is only feasible for circuits with a small number of nodes, inputs and states (e.g., 4-bit Gray code counter has 16 states and has about 150 nodes). Most benchmark circuits are too big (large number of nodes and inputs) to be directly feasible for study using our current approach. It can be modified in two ways to make the analysis of bigger circuits feasible. First, since SPICE is very time consuming, simulation can be performed at higher level of abstraction. Another way to estimate fault tolerance quickly would be to estimate fault tolerance by performing simulations for only a few nodes. Note that both these approaches will provide a less accurate estimate of fault tolerance.

A. Binary Counter

A binary counter is the simplest and most commonly used counter. Goutis [11] proposed a technique for making binary counters fault secure using *T* flip-flop properties to detect errors. Fig. 4 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 Fig. 5. This implementation uses a *D*-flip flop (or a latch) and the algorithm proposed in [11] cannot be directly applied to this counter. An alternate scheme uses the parity of the present enable signals $(\mathrm{EN}_1, \mathrm{EN}_2, \ldots, \mathrm{EN}_n)$ to indicate if the parity of the next state $(\mathrm{BIT0}, \mathrm{BIT1}, \ldots, \mathrm{BIT}n)$ will change. Instead of using the $T_1, T_2, \ldots, T_n$ signals in Fig. 4(b), to detect faults, signals $\mathrm{EN}_1, \mathrm{EN}_2, \ldots, \mathrm{EN}_n$ can be used instead. Figs. 4 and 5 are similar binary counter implementations except for the memory element. Our study uses the implementation shown in Fig. 5 since it is a popular implementation that is similar to a Gray code counter.

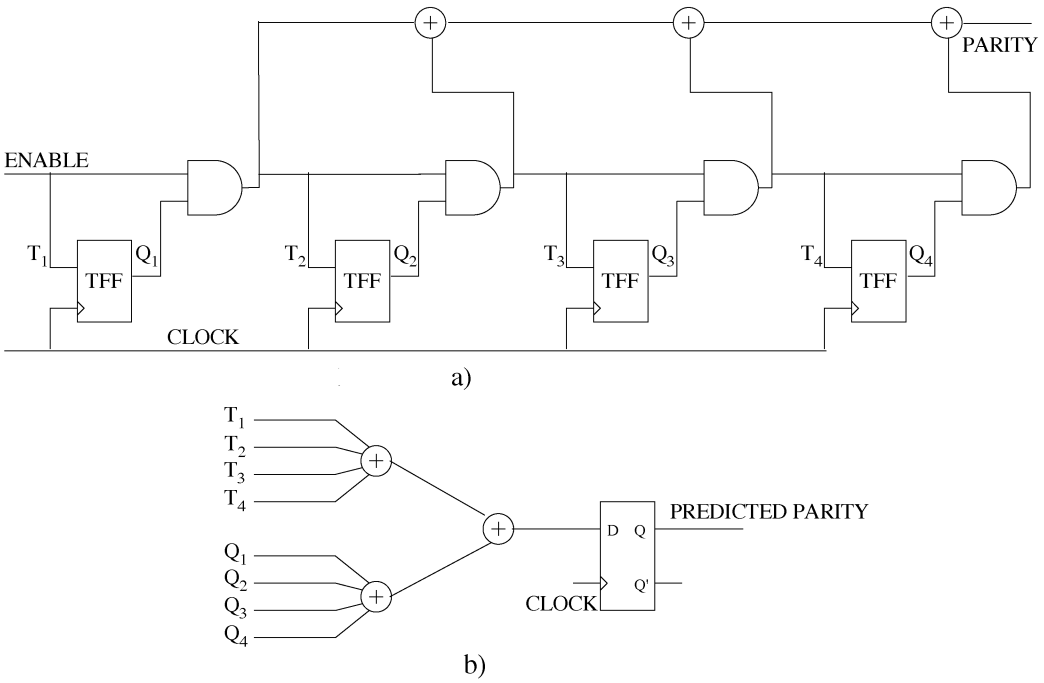

Fig. 4. Binary counter with fault detection using *T*-flip flop. a) Binary counter using *T*-flip flop and b) parity prediction logic.

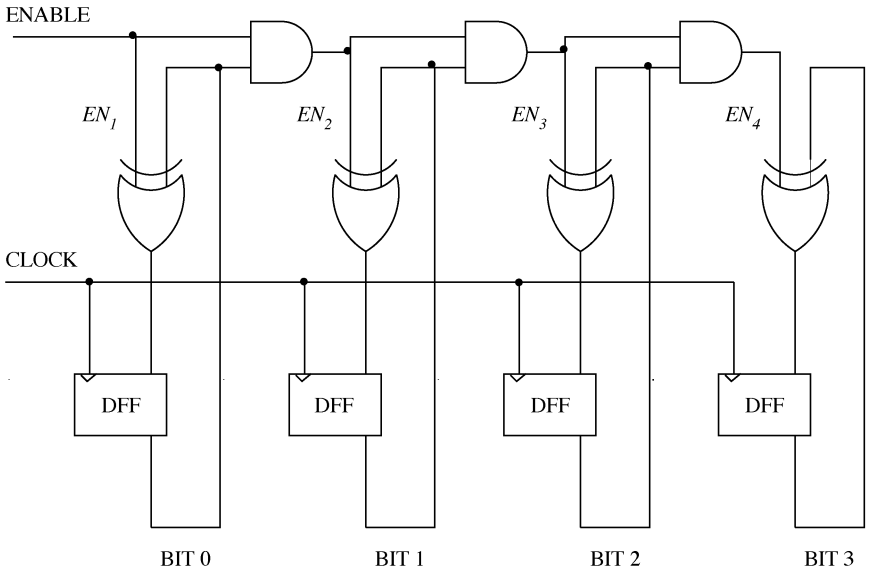

Fig. 5. Binary counter using a *D*-flip flop.

B. Gray Counter

A binary counter is not intuitively power efficient since there are multiple signal transitions during state transitions. A Gray counter can be lower power, since there is only one flip-flop transition during state transitions. This switching activity reduction can save a significant amount of power, although, intermediate logic may have more transitions. The use of Gray code counters has been proposed to save power in the control path of embedded processors [12]. Gray code addressing reduces address line switching activity by 30–50% during normal program execution [13]. A significant amount of logic is required to determine the next state of a Gray code counter. This logic is usually $(n-1)$ gates deep for a n-bit counter and depends on all previous state bits and their complements. This logical complexity can make the Gray counter less power efficient in some situations. A segmented Gray counter [14] can be used to reduce the amount of logic required at the cost of a slight increase in switching activity. Fault detection in a Gray counter is easy, since the parity of each segment toggles with every state change. A circuit implementation of a 4-b segment of a Gray counter is shown in Fig. 6.

C. Latch Implementation

Latch is the most critical component of the counter design. Being a bistable element it is most vulnerable to any transients. In order to facilitate fair comparison, both the counters are implemented using a *D* Flip-flop. An edge-triggered flip-flop will provide a higher tolerance to transient faults since it will latch only the faults which appear at the input of the flip-flop when the data

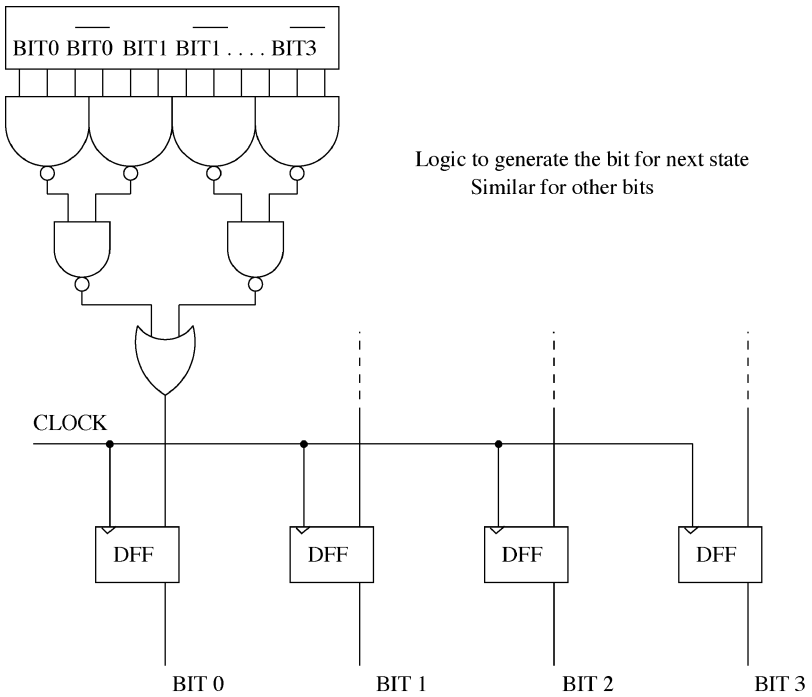

Fig. 6. Implementation of a Gray counter.

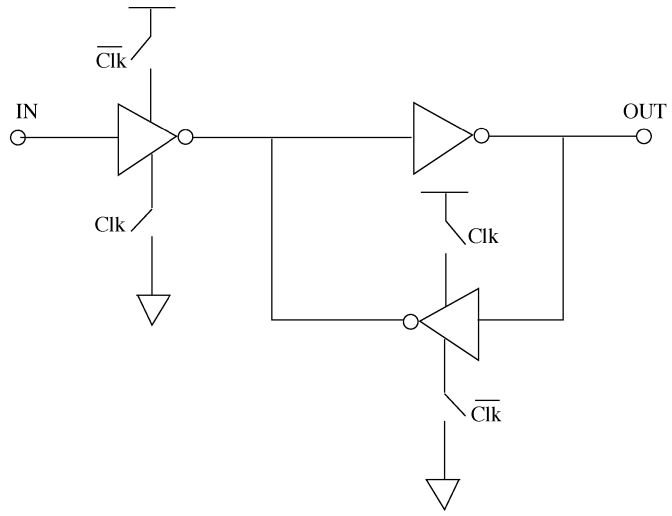

Fig. 7. Latch used in the counters.

is getting latched. In this study, we have used a simple flip-flop shown in Fig. 7. The inherent fault tolerance of this latch might be improved but this design serves the purpose of our study.

V. Improving Fault-Tolerance of Counters

Using the method described in Section III, the fault tolerance of 4-b binary and Gray counters is measured in terms of POF_c and MTTF. The binary counter has about $n = 85$ nodes while the gray counter has about $n = 150$ nodes. Both counters have 16 possible states ($p = 16$). Fault tolerance and power dissipation results are shown in Table I.

Note that the MTTF of the two counters differs by only 18% despite vastly different (41%) POF_c. This is due to the large area difference of the Gray counter which compensates for the Gray counter's low fault sensitivity result. Gray counters are inherently more fault-tolerant because they have a balanced design. However, a 4-b gray counter consumes more power than the 4-b binary counter. Since the counters are very small (only 4-b wide), the gain due to fewer gray counter transitions is small compared to the loss of power due to circuit area increase.

TABLE I
Comparison of Counter Without any Optimization

Counter	POF_c	$MTTF$ (years)	Power (μW)
Binary	0.524	18.49	68.85
Gray	0.308	21.84	73.58

Two approaches are considered for improving counter fault tolerance 1) incorporation of redundancy at the architecture-level and 2) circuit-level optimization for part or all of the circuit.

A. Incorporating Redundancy in the Counters

In general, spatial (area) or temporal (time) redundancy can be applied to a circuit to make it fault tolerant [32]. Area redundancy can be applied to counters using dual modular redundancy (DMR) (Fig. 8) or triple modular redundancy (TMR). Redundant counters are required for these approaches and the final output is the majority vote of these counters. Adding area redundancy can increase power consumption significantly (e.g., by a factor of 2 or 3).

In a *time-redundant* technique, an errant output is recomputed in an attempt to recover from transient faults. This approach limits hardware overhead, leading to reduced power consumption. Fig. 9 illustrates our implementation of the *time-redundant* technique. During normal operation the counter output is stored in a register. In case of an error, the previously stored correct value is loaded into the counter. The counter then recomputes the next state. The additional hardware required includes an enabled register, multiplexers, and control logic. Power is

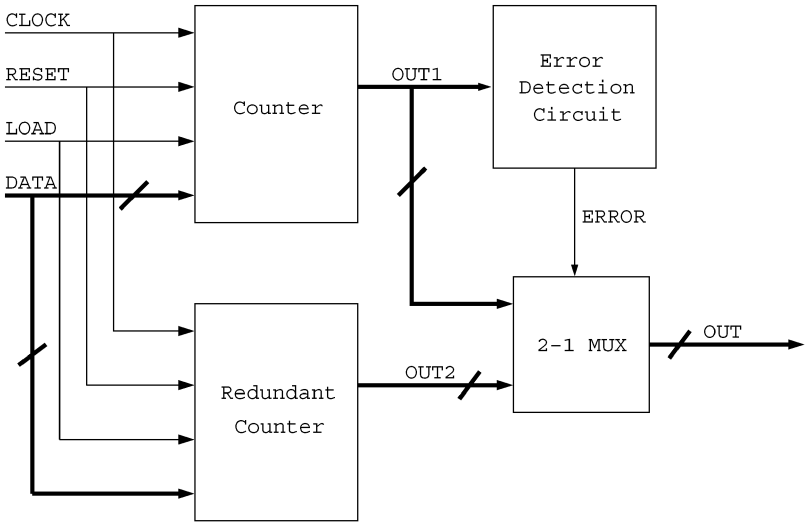

Fig. 8. Architecture of an area redundant (DMR) counter.

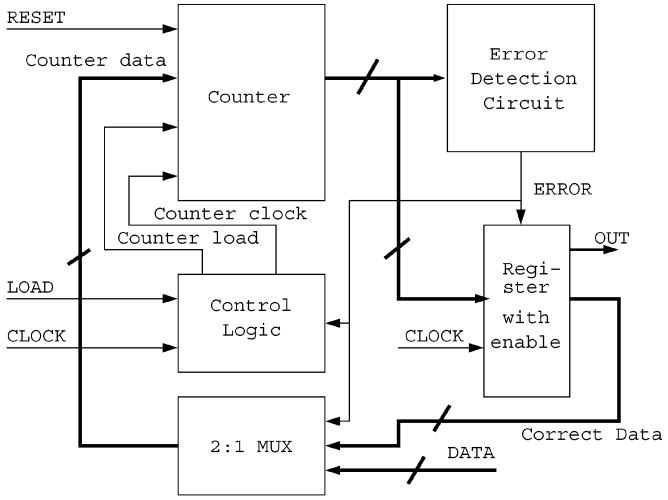

Fig. 9. Architecture of a time redundant counter.

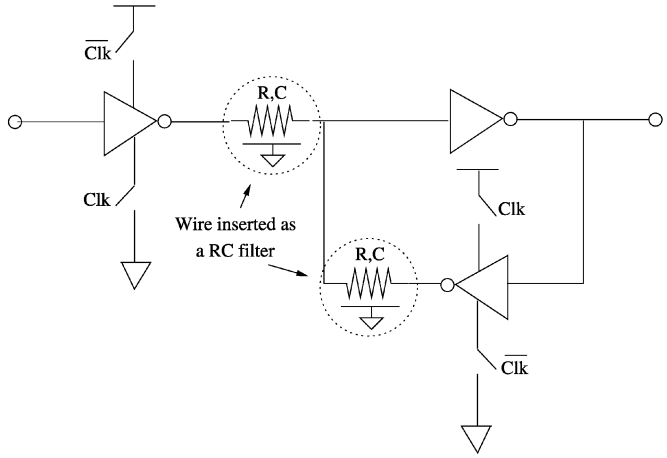

Fig. 10. TPTL.

TABLE II
COMPARISON OF AREA AND TIME REDUNDANT COUNTERS

Counter	POF_c	$MTTF$ (years)	Power (μW)
DMR Area Redundant Binary Counter	0.058	167.0	195.8
Time Redundant Binary Counter	0.058	167.0	158.5
DMR Area Redundant Gray Counter	0.043	156.1	190.4
Time Redundant Gray Counter	0.043	156.1	181.3

saved since error recovery logic is used only *after* an error has been detected. Thus, this technique is most appropriate for ultralow-power systems. Since the counter might have to recompute the next state, additional time may be required to determine the correct output, limiting the use of this technique to low performance systems.

For single faults, the area and time redundant techniques discussed here result in a high-transient fault tolerance. The fault tolerance depends on the efficiency of the error-detection circuit in detecting faults. If an error detection circuit can detect all the faults, then 100% transient fault tolerance can be achieved. Fault tolerance of the above discussed implementations and the power consumed be them is shown in Table II.

TABLE III
COMPARISON OF COUNTERS USING TPTL WITH FILTERS OF VARIOUS WIRELENGTHS

Counter	Wire length used (μm)	POF_c	$MTTF$ (years)	Power (μW)
Binary	0	0.524	18.49	68.85
	10	0.468	20.70	77.81
	25	0.399	24.28	86.65
	50	0.391	24.78	102.43
Gray	0	0.308	21.84	73.58
	10	0.266	25.29	82.71
	25	0.205	32.82	91.93
	50	0.202	33.31	109.35

B. Circuit-Level Fault Tolerance

Circuit-level fault tolerance can be used as an alternative to architecture-level fault tolerance. At the circuit level, certain circuit components are redesigned and optimized to make the overall circuit more fault tolerant. In order to have a fair comparison, the circuit operating frequency must be fixed. The overall size of the circuit can change as the metric used to represent fault tolerance is independent of circuit area. In this section, circuit-level fault-tolerance techniques are evaluated in terms of improvement in fault tolerance and their impact on power dissipation using the methodology described in Section III.

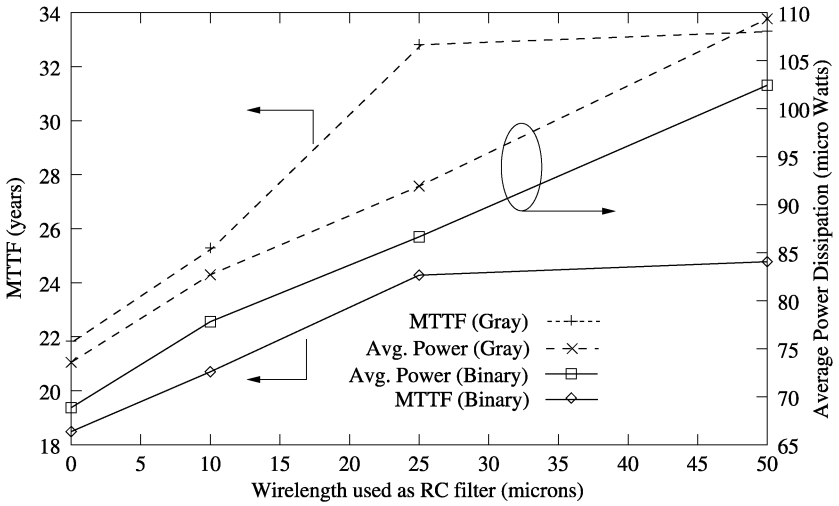

Fig. 11. Effect of adding wire as a filter in the latches on fault-tolerance (MTTF) and power dissipation. (Arrows point toward the axis relevant to the plot).

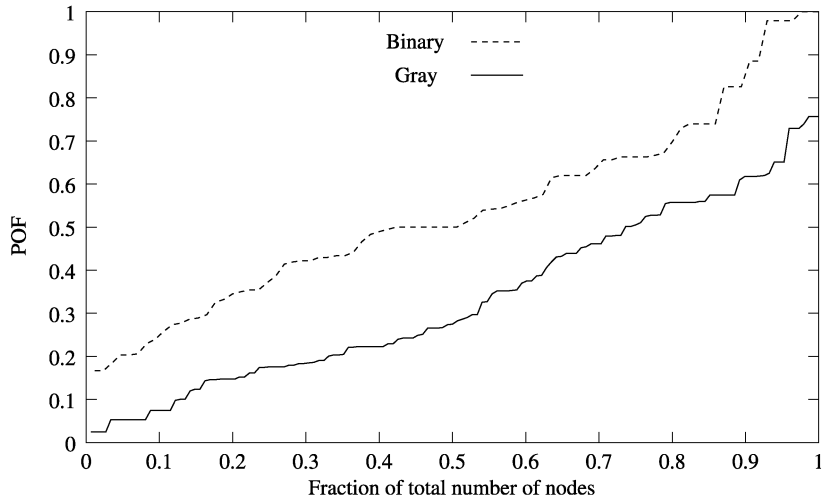

Fig. 12. POF_c as a function of fraction of total number of nodes for the two counters showing the criticality of the counters and number of critical nodes.

1) Transient Pulse Tolerant Latch (TPTL): A counter consists of a set of memory elements (flip-flops), which hold the current state of the counter, and logic, which determines the next state based on the current state. Since flip-flops are bistable elements with positive feedback, they are more sensitive to transients than logic gates. A TPTL, proposed in [33], can be used to improve the fault tolerance of the flip-flop. A low-pass filter is added to the latch in order to filter out the high-frequency transients. Fig. 10 shows the implementation of a TPTL which is used to improve the fault tolerance of the counters. Wires acting as *RC* low-pass filters are added to the latch to help filter out high frequency transients. Several *RC* values corresponding to small wirelengths ranging from 10μ to 50μ are used to study the effect of TPTL on fault tolerance and power (Table III). Although the latch shown in Fig. 10 is an elementary latch, the technique can be used with any latch containing feedback circuitry.

Results in Fig. 11 show that with the increase in resistance and capacitance due to increased wirelength, the fault tolerance of the counter (measured as MTTF) increases at the cost of increased power. The gain in terms of fault tolerance, however, saturates at about a wirelength of 25μ and any further wirelength increase results only in additional power cost. Moreover, a wirelength increase slows down the latch. For the wirelengths used in these simulations the additional delay did not result in any change in the clock frequency.

TABLE IV
COMPARISON OF COUNTERS USING VARIOUS TRANSISTOR SIZES FOR CRITICAL NODES

Counter	Times minimum size	POF_c	$MTTF$ (years)	Power (μW)
Binary	1X	0.524	18.49	68.85
	2X	0.491	18.62	76.63
	4X	0.433	19.97	86.54
	6X	0.403	20.37	91.21
Gray	1X	0.308	21.84	73.58
	2X	0.261	24.32	83.32
	4X	0.197	30.49	95.23
	6X	0.158	36.09	102.64

2) Selective Sizing of Transistors: Increasing the size of transistors also improves fault tolerance [34]. Increasing the size of the transistor reduces the magnitude of the node voltage

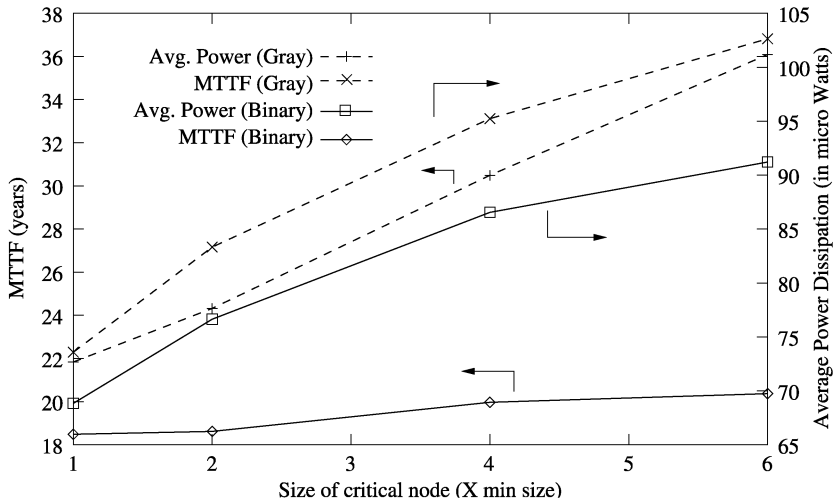

Fig. 13. Effect of resizing critical nodes on fault-tolerance (MTTF) and power dissipation (arrows point toward the axis relevant to the plot).

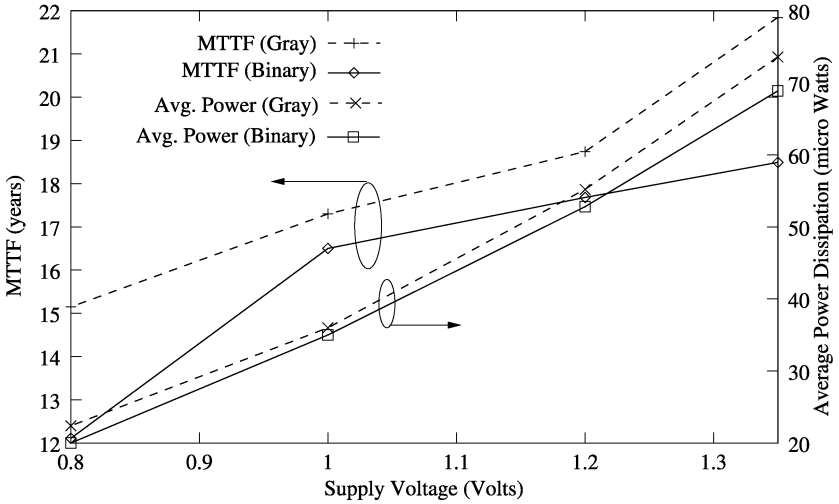

Fig. 14. Impact of power supply scaling on fault-tolerance (MTTF) and power dissipation (arrows point toward the axis relevant to the plot).

offset (due to the transient) at the drain of the transistor and thus improves transient fault tolerance. Note that an increase in the transistor size results in an increase in the diffusion and the gate capacitance. An increase in diffusion capacitance increases susceptibility to transient faults due to the increase in the fault vulnerable area. An increase in the size of the transistor will also increase the power dissipation due to an increase in load capacitance. Hence, only selected highly sensitive nodes should be resized to improve the fault tolerance without increasing the power and area penalty significantly. As shown in Fig. 12, the fraction of nodes with a POF_c higher than a certain value depends on the circuit. The individual POF_c for nodes in the gray counter is much less than those of the binary counter.

In this work, the top 5% critical nodes were sized up by a factor of 2–6 times the minimum, and their effect on fault tolerance and power was studied (Table IV). This technique resulted in an area penalty ranging from 6%–18%.

Results show that the fault tolerance of the counter increases with the size of the transistors, at the cost of increased power (Fig. 13). The MTTF of the binary counter increased by 10%, increasing the power consumption by 32%. This technique is more beneficial to the Gray counter as shown by the 65% increase in MTTF and a 39% increase in power consumption. If the size of devices is further increased, fault tolerance will improve, at the cost of power efficiency until a limit is reached. A sizing factor can be determined based on the power budget for the system (or the circuit).

TABLE V
COMPARISON OF COUNTERS USING VARIOUS POWER SUPPLIES

Counter	Supply Voltage (volts)	POF_c	$MTTF$ (years)	Power (μW)
Binary	1.35	0.524	18.49	68.85
	1.2	0.548	17.68	52.80
	1.0	0.587	16.50	35.00
	0.8	0.615	12.11	20.00
Gray	1.35	0.308	21.84	73.58
	1.2	0.359	18.74	55.20
	1.0	0.389	17.30	36.00
	0.8	0.444	15.15	22.40

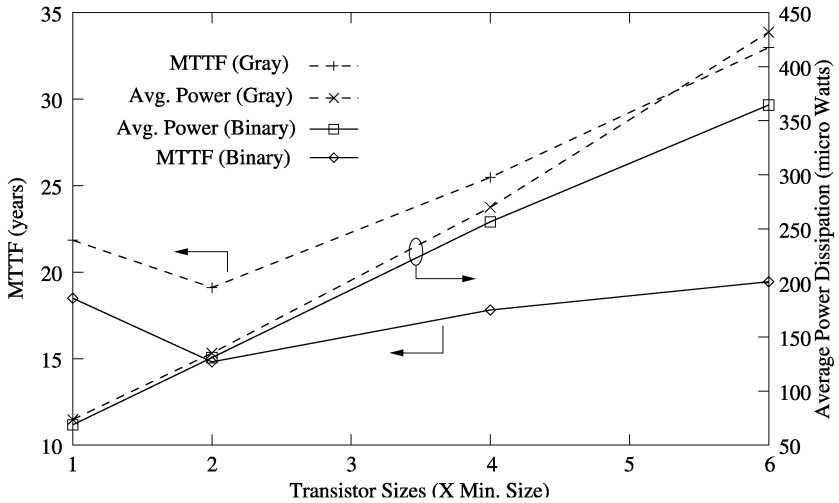

Fig. 15. Impact of sizing the circuit on fault tolerance (MTTF) and power dissipation (arrows point toward the axis relevant to the plot).

VI. Impact of Power Reduction Techniques

Classical power reduction techniques like power supply voltage reduction and load capacitance reduction have been proposed to reduce power dissipation in CMOS VLSI circuits [35], [36]. In this section, the effect of these techniques on fault tolerance is quantified using the method and measures described in Section III.

A. Power Supply Voltage Reduction

Most of the power dissipated in CMOS VLSI circuits is due to dynamic power dissipation

$$P_{\text{dyn}} = C_L {V_{dd}}^2 f \tag{8}$$

where

C_L load capacitance;
V_{dd} power supply voltage;
f frequency of switching.

From (8), supply voltage reduction is the most effective way to reduce power consumption, due to its squared dependence on supply voltage.

To evaluate the effect of power supply scaling on circuit fault tolerance, simulations were performed to calculate POF_c and MTTF for power supply voltage varying from 1.35–0.8 V. Since the threshold voltage of the transistor is 0.35 V, the supply voltage was not reduced further to maintain the noise margins.

As shown in Fig. 14 and Table V, as the supply voltage is scaled down, the power consumed by the counters is reduced. Scaling a supply voltage from 1.35–0.8 V results in about a 70% reduction in power consumption while the MTTF is reduced by 30%–35%.

B. Reducing the Load Capacitance

Load capacitance (C_L) reduction is another way to reduce the power consumption, since the dissipation is proportional to C_L. Load capacitance can be reduced by reducing the size of the transistors, leading to smaller gate and diffusion capacitance.

TABLE VI
Comparison of Counters Using Different Transistor Sizes

Counter	Times Min Size	POF_c	$MTTF$ (years)	Power (μW)
Binary	1X	0.524	18.49	68.85
	2X	0.327	14.81	130.95
	4X	0.136	17.81	256.50
	6X	0.083	19.45	364.50
Gray	1X	0.308	21.84	73.58
	2X	0.176	19.11	135.00
	4X	0.066	25.48	270.00
	6X	0.034	32.98	432.00

To evaluate the effect of load capacitance on the fault tolerance of the circuit, simulations were performed on four different counter designs. The designs were implemented using either minimum size transistors or transistors which were 2, 4, or 6 times the minimum size.

Fig. 15 shows the results for various transistor sizes (which correspond to various load capacitances). Increasing the size of the devices initially reduces MTTF and then increases MTTF values while consistently increasing power consumption. As seen from Table VI the POF_c is reduced almost linearly with increases in transistor size. The increase in area results in a loss or slight gain in MTTF. It is not optimal to increase transistor size even for MTTF gain, since a DMR system provides a more power efficient solution.

C. Increasing the Threshold Voltage

In order to improve CMOS circuit performance with technology scaling, the transistor threshold voltage (V_{th}) can be reduced at the same rate as the supply voltage. A reduction in V_{th} causes transistor subthreshold leakage current (I_{off}) to increase exponentially. This results in a corresponding increase in leakage power. Modern CMOS processes provide transistors with different V_{th} levels. High V_{th} transistors can thus be used to reduce leakage power.

In order to estimate the impact of threshold voltage increase on circuit fault tolerance, circuits containing transistors with different V_{th} were evaluated for fault tolerance.

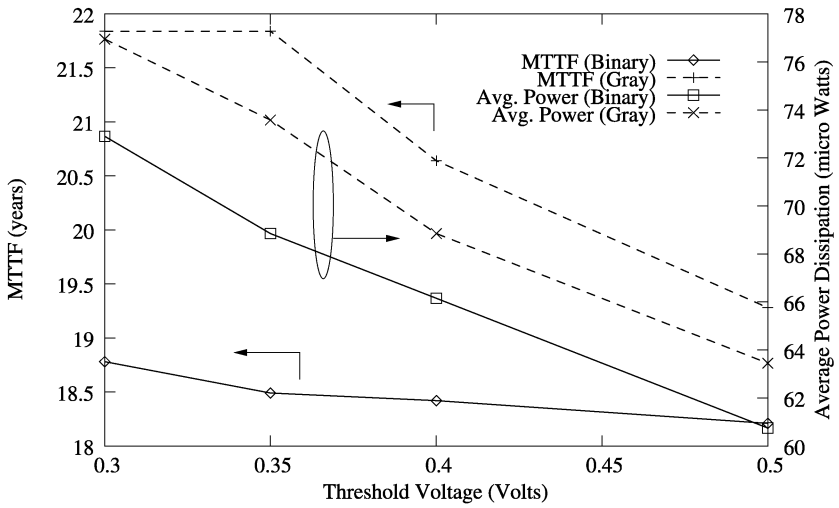

Fig. 16. Impact of threshold voltage on fault-tolerance (MTTF) and power dissipation (arrows point toward the axis relevant to the plot).

TABLE VII
COMPARISON OF COUNTERS USING VARIOUS THRESHOLD VOLTAGES

Counter	Threshold Voltage (volts)	POF_c	$MTTF$ (years)	Power (μW)
Binary	0.3	0.516	18.78	72.90
	0.35	0.524	18.49	68.85
	0.4	0.526	18.42	66.15
	0.5	0.532	18.21	60.75
Gray	0.3	0.308	21.84	76.95
	0.35	0.308	21.84	73.58
	0.4	0.326	20.64	68.85
	0.5	0.349	19.28	63.45

Fig. 16 and Table VII shows a tradeoff between power and fault tolerance that is similar to the one seen with the previous two techniques (Figs. 14 and 15). For 0.18μ, the leakage power is a small percentage of the total power. Thus, increasing the threshold voltage from 0.3–0.5 V reduces power dissipation by 15%. Increasing the threshold voltage, however, reduces MTTF by small amounts (3%–10%). Fault tolerance degrades due to a reduced noise margin. Even the smallest transient results in an offset voltage height greater than V_{DD}. Fault-tolerance improvement is negligible since an increase in threshold voltage makes it relatively difficult for transients to switch the transistors.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, a method to estimate fault tolerance in terms of a well-known metric (mean time to failure) is presented. Although time consuming in its present form, the method is highly accurate since exhaustive simulations are performed using a circuit simulator (HSPICE).

Using this methodology, an attempt is made to relate the transient fault tolerance and power dissipation in functionally critical low-power circuits. Binary and Gray counter designs in 0.18μ technology are used as example circuits. It is noticed, that although the Gray counter has inherently fewer critical nodes when compared to the binary counter due to a balanced design, the MTTF was very similar due to a large implementation area.

Several architecture level and circuit level fault-tolerance techniques were studied for both counters. At the architecture level, the *time-redundant* technique provides power benefits over the *area-redundant* technique when applied to circuits with low-performance requirements. Circuit-level solutions like the fault-tolerant latch and critical node sizing result in increased MTTF at the cost of increased power dissipation.

Several low-power techniques such as scaling power supply, load-capacitance reduction and threshold voltage increase were also studied. The incorporation of the techniques reduced power dissipation. However, power supply voltage reduction also reduced the fault tolerance. It was concluded that the use of minimally sized devices was the best option to save power and to increase fault tolerance. Fault tolerance has little dependence on the threshold voltage.

Fig. 17 is the plot evaluating the different counter designs for the two objectives (i.e. fault tolerance and power dissipation). This plot consists of all the design variations which were studied in this work. Fig. 17 also points out designs with relatively low-power dissipation and high fault tolerance.

From this work, we can conclude that conventional low power and fault tolerance design techniques are at odds and some guidelines and novel design techniques are required to address both objectives simultaneously. As seen from the plot, a design either has high fault tolerance (having high MTTF) or has low-power dissipation. A counter design could have a MTTF of 36 years and burn 102 μW of power or could have a MTTF of only 12 years and burn 20 μW of power.

Future work involves exploration and development of novel circuit and coding 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 the latest technologies. With technology scaling, leakage power is on an increase and a similar study can be performed

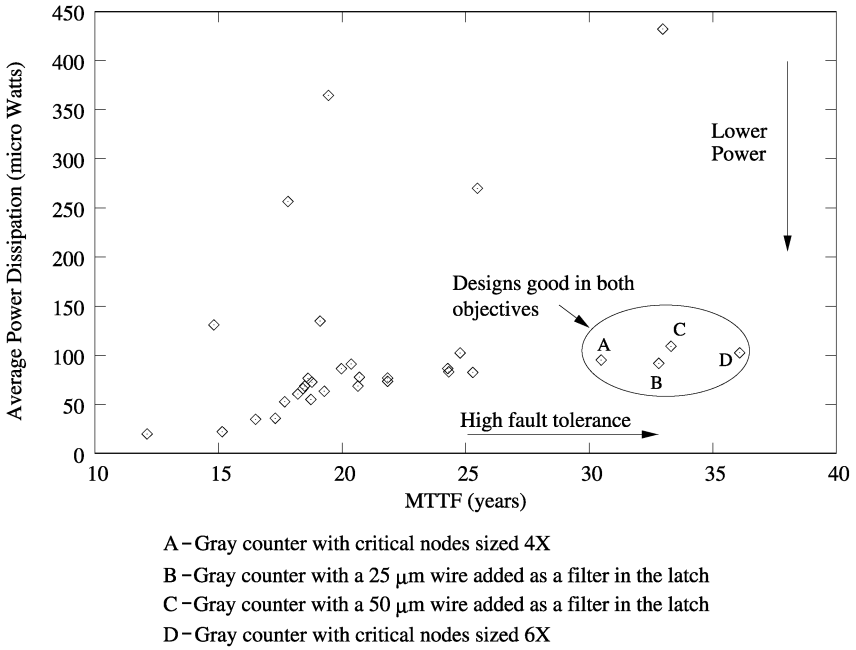

Fig. 17. Counter designs with their corresponding MTTF and power dissipation showing the tradeoff between the two objectives.

to evaluate leakage mitigation techniques for transient fault tolerance. Many low-power systems contain fault sensitive mixed-signal circuits. It is important to perform similar studies for mixed-signal circuits where fault propagation and manifestation have different mechanisms.

REFERENCES

[1] J. F. Ziegler *et al.*, "IBM experiments in soft fails in computer electronics (1978–1994)," *IBM J. Res. Develop.*, vol. 40, pp. 3–17, 1996.

[2] T. C. May and M. H. Woods, "Alpha-particle-induced soft errors in dynamic memories," *IEEE Trans. Electron. Devices*, vol. 26, pp. 2–9, Jan. 1979.

[3] E. Normand *et al.*, "Single event upset and charge collection measurements using high energy protons and neutrons," *IEEE Trans. Nucl. Sci.*, vol. 45, pp. 2904–2914, Dec. 1998.

[4] C. A. Gossett *et al.*, "Single event phenomena in atmospheric neutron environment," *IEEE Trans. Nucl. Sci.*, vol. 40, pp. 1845–1852, Dec. 1993.

[5] Y. Tosaka, K. Suzuki, and T. Sugii, "Alpha particle induced soft errors in submicron SOI SRAM," in *Proc. Symp. VLSI Technology Dig. Tech. Papers*, 1995, pp. 39–40.

[6] T. J. O'Gorman *et al.*, "Field testing for cosmic ray soft errors in semiconductor memories," *IBM J. Res. Develop.*, vol. 40, pp. 41–49, Jan. 1996.

[7] P. J. Griffin *et al.*, "The role of thermal and fission neutrons in reactor neutron-induced upsets in commercial SRAMs," *IEEE Trans. Nucl. Sci.*, vol. 44, pp. 2079–2086, Dec. 1997.

[8] K. Johansson *et al.*, "In-flight and ground testing of single event upset sensitivity in static RAM's," *IEEE Trans. Nucl. Sci.*, vol. 45, pp. 1628–1632, June 1998.

[9] N. Cohen *et al.*, "Soft error considerations for deep-submicron CMOS circuit applications," in *Proc. Tech. Dig. Int. Electronic Devices Meeting (IEDM)*, 1999, pp. 315–318.

[10] V. De and S. Borkar, "Technology and design challenges for low power and high performance," in *Proc. IEEE Symp. Low-Power Electronics and Design*, 1999, pp. 163–168.

[11] E. Karaolis, S. Nikolaidis, and C. E. Goutis, "Fault secure binary counter design," in *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 1999, pp. 1659–1662.

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

[13] H. Mehta, R. M. Owens, and M. J. Irwin, "Some issues in Gray code addressing," in *Proc. 6th Great Lake Symp. VLSI*, 1996, pp. 178–181.

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

[15] H. Ball and F. Hardy, "Effects and detection of intermittent failures in digital systems," in *Proc. Fall Joint Computer Conf. (FJCC), American Federation of Information Processing Societies (AFIPS) Conf.*, 1969, pp. 329–335.

[16] R. Iyer and D. Rosetti, "A statistical load dependency of CPU errors at SLAC," in *Proc. Fault Tolerance Computing Systems (FTCS)*, 1982, pp. 363–372.

[17] R. McPartland, "Circuit simulations of alpha-particle-induced soft errors in dynamic RAM's," *IEEE J. Solid State Circuit*, vol. 16, pp. 31–34, Feb. 1981.

[18] R. Johnson, S. Diehl-Nagle, and J. Hauser, "Simulation approach for modeling single event upsets on advanced CMOS SRAMS," *IEEE Trans. Nucl. Sci.*, vol. 32, pp. 4122–4127, Dec. 1985.

[19] G. C. Messenger, "Collection of charge on junction nodes from ion tracks," *IEEE Trans. Nucl. Sci.*, pp. 2024–2031, 1982.

[20] E. W. Czeck and D. P. Siewiorek, "Effects of transient gate-level faults on program behavior," in *Dig. Int. Symp. Fault-Tolerant Computing*, 1990, pp. 236–243.

[21] M. Rimen and J. Ohlsson, "A study of the error behavior of a 32-bit RISC subjected to simulated transient fault injection," in *Proc. Int. Test Conf.*, 1992, pp. 696–704.

[22] H. Cha and J. H. Patel, "A logic-level model for α-particle hits in CMOS circuits," in *Proc. IEEE Int. Conf. Computer Design*, Oct. 1993, pp. 538–542.

[23] H. Cha *et al.*, "A fast and accurate transient fault simulation environment," in *Dig. Int. Symp. Fault-Tolerant Computing*, June 1993, pp. 310–319.

[24] G. L. Ries, G. S. Choi, and R. K. Iyer, "Device-level transient fault modeling," in *Dig. Papers, Int. Symp. Fault-Tolerant Computing*, 1994, pp. 86–94.

[25] G. Laguna and R. Treece, "VLSI modeling and design for radiation environments," in *Proc. IEEE Int. Conf. Computer Design*, 1986, pp. 380–384.

[26] G. Choi *et al.*, "A fault behavior model for an avionic microprocessor: a case study," in *Proc. Int. Working Conf. Dependable Computing for Critical Applications*, Aug. 1989, pp. 71–77.

[27] F. L. Yang and R. A. Saleh, "Simulation and analysis of transient faults in digital circuits," *IEEE J. Solid-State Circuits*, vol. 27, pp. 258–264, Mar. 1992.

[28] HSPICE User's Guide. [Online]. Available: www.synopsys.com.

[29] M. Singh, R. Rachala, and I. Koren, "Transient fault sensitivity analysis of analog-to-digital converters (ADC's)," in *Proc. IEEE Annual Workshop VLSI*, Apr. 2001, pp. 140–145.

[30] S. Kang and D. Chu, "CMOS circuit design for the prevention of single event upset," in *Proc. IEEE Int. Conf. Computer Design*, 1986, pp. 385–388.
[31] J. F. Ziegler *et al.*, "Terrestrial cosmic rays," *IBM J. Res. Develop.*, vol. 40, pp. 19–39, 1996.
[32] D. K. Pradhan, *Fault Tolerant Computing*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
[33] H. Cha and J. H. Patel, "Latch design for transient pulse tolerance," in *Proc. IEEE Int. Conf. Computer Design*, 1994, pp. 385–388.
[34] M. Singh and I. Koren, "Reliability enhancement of analog-to-digital converters (ADC's)," in *IEEE Int. Symp. Defect and Fault Tolerance VLSI Systems*, Oct. 2001, pp. 347–353.
[35] J. Rabaey, *Digital Integrated Circuits, a Design Perspective*. Englewood Cliffs, NJ: Prentice-Hall, 1996.
[36] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low power CMOS digital design," *IEEE J. Solid-State Circuits*, pp. 1082–1087, Apr. 1992.

Atul Maheshwari (S'99) received the B.E. degree in electronics and communication engineering from the L.D. College of Engineering, Gujarat University, India, in 1998, and the M.S. degree in electrical and computer engineering from the University of Massachusetts, Amherst, in 2001, and is currently pursuing the Ph.D. degree in electrical and computer engineering at University of Massachusetts, Amherst.

From 1998 to 1999, he worked for Larsen and Toubro Biomedical Equipments Research and Development, Mysore, India. From May 2000 to March 2001, he was with the Alpha Development Group, Compaq, Shrewsbury, MA. His research interests include high-speed on-chip interconnects, transient faults and soft errors (SE) in VLSI circuits and clock distribution networks.

Wayne Burleson (M'84–SM'01) received the B.S.E.E. and M.S.E.E. degrees from the Massachusetts Institute of Technology, Cambridge, in 1983, and the Ph.D. degree from the University of Colorado, Boulder, in 1989.

He is currently an Associate Professor of Electrical and Computer Engineering with the University of Massachusetts at Amherst, where he has been since 1990. He was a custom digital signal processor chip designer for four years with VLSI Technology Inc., and Fairchild Semiconductor. He was a Consultant with the Alpha Development Group of Compaq, Shrewsbury, MA, and the Datafusion Corporation. He was a Visiting Professor with the Ecole Nationale Superieure des Telecommunications, Paris, France, from September 1996 to August 1997. His previous work includes research, development, teaching, and industrial work at a variety of levels including theory, algorithms, architectures, circuits, and computer-aided design tools. His current research interests include very large scale integration signal processing, digital complementary metal-oxide-semiconductor circuit design, and multimedia educational innovations.

Dr. Burleson is a Member of the Association for Computing Machinery (ACM) and Sigma Xi.

Russell Tessier (M'00) received the B.S. degree in computer engineering from Rensselaer Polytechnic Institute, Troy, N.Y. in 1989. and S.M. and Ph.D. degrees in electrical engineering from Massachusetts Institute of Technology, Cambridge, in 1992 and 1999, respectively.

He is currently an Assistant Professor of electrical and computer engineering at the University of Massachusetts, Amherst. He is a founder of Virtual Machine Works, a logic emulation company, and has also worked at BBN and Ikos Systems. He currently leads the Reconfigurable Computing Group at University of Massachusetts. His research interests include computer architecture, field-programmable gate arrays, and system verification.