A Probabilistic Model for Soft-Error Rate Estimation in Combinational Logic

Jung Sub Kim, Chrysostomos Nicopoulos, N. Vijaykrishnan, Yuan Xie and Emanuele Lattanzi*
Microsystems Design Laboratory, The Pennsylvania State University, University Park, PA 16802
* Universita di Urbino, Italy
{jskim, nicopoul, vijay, yuanxie} @cse.psu.edu

Abstract

Single Event Upsets (SEU) arising from atmospheric neutrons and alpha particles are becoming increasingly important in combinational logic circuits. Combinational logic is resilient to soft errors due to three masking phenomena: (1) Logical Masking, (2) Electrical Masking, and (3) Latching-window Masking. This paper concentrates on logical masking, and proposes a probabilistic model which calculates the Soft Error Rate (SER) of any output node in combinational logic circuits, based on inherent logical masking properties.

1. Introduction

Technology scaling, shrinking geometries into the deep sub-micron regime, lower supply voltages, higher operating frequencies, and higher density circuits have all had a negative impact on reliability. The number of occurrences of transient faults has increased dramatically. One major transient fault type is soft errors, caused by two main sources: (1) secondary cosmic rays, especially atmospheric neutrons, and (2) alpha particles emitted by decaying radioactive impurities in packaging and interconnect materials. These highly energetic particles induce Single Event Transients (SET) in digital circuits. The amount of charge injected may be sufficient to invert the logical state at a node, hence introducing a soft error. SER per chip is projected to increase quadratically with decreasing feature size [1].

Traditionally, soft errors were tackled within the context of memory cells. Today, error detection and correction circuits are widely used to protect memory arrays. Combinational logic circuits, on the other hand, have been found to be less susceptible to SEU in equivalent device technologies due to the naturally occurring logical, electrical and latching-window masking effects [2]. However, these phenomena are diminishing as feature size decreases and circuits move to higher operating frequencies. Recent studies predict that the SER per chip of logic circuits will increase exponentially to 2011, at which point it will be comparable to the SER per chip of unprotected memory elements [3].

For an SET induced in a combinational logic circuit to result in a soft error, three conditions have to be satisfied: (1) an active path must exist between the afflicted node and the output of the circuit. (2) The pulse must be wide enough to avoid inertial delay filtration through subsequent gates, and survive electrical attenuation along the active path. Finally (3), the pulse should arrive within the setup and hold time of a latch element to be captured and cause a soft fault [4].

The proposed model in this paper concentrates on the first condition, and specifically on the role of logical masking in SET propagation. The logic state of a gate along the SET path to an output can inhibit the upset propagation as a result of the gate's logical function. The haphazard nature of particle-induced upsets, though, implies a probabilistic approach, which estimates the SER based on input vector probabilities and the statistics of SET in CMOS circuits. This model can easily be incorporated as a component within a complete SER modeling tool.

Several studies have been conducted in the estimation of SER in both storage elements and combinational logic [8, 9, 2]. Hazucha and Svensson developed an empirical model to predict atmospheric neutron SER as a function of technology scale [1]. Tosaka et al. used a Modified Burst Generation Rate (MBGR) empirical model to predict neutron-induced SER [5]. Some work has also been done in SER estimation in combinational logic. Massengill et al. developed a VHDL simulator to analyze SEU effects in combinational circuits [4]. Baze et al. investigated the effects of electrical masking [6], while Buchner et al. investigated latching-window masking [7]. Shivakumar et al. proposed a complete model to account for all three masking phenomena in combinational logic [3].

The rest of the paper is organized as follows: section 2 explains the proposed model algorithm and illustrates, through a simple example, its application to combinational logic. Section 3 analyzes – using the model – two small circuits (the example circuit from Section 2 and a Full Adder) and then compares the results to those obtained from simulations to validate the model. Section 4 concludes the paper and points to future work plans.

2. Model Description

2.1 Preliminary Calculations

This probabilistic model calculates the soft error probability of any output node in a combinational circuit, based on logical masking principles.

The proposed approach differs from the ones found in the literature in three important ways: (1) this model assumes soft error hits at *individual nodes*, and not on the gate as a whole; this makes the model more realistic and accurate:

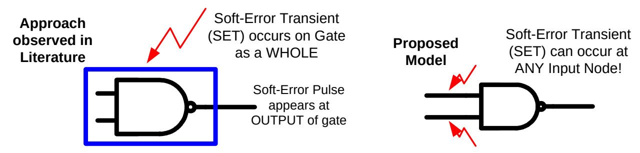
Approach observed in Literature
Soft-Error Transient (SET) occurs on Gate as a WHOLE
Soft-Error Pulse appears at OUTPUT of gate
Proposed Model
Soft-Error Transient (SET) can occur at ANY Input Node!

Fig. 1 *An SET can occur at ANY node*

(2) The model accounts for *input probabilities*, i.e. it can accommodate *unbalanced input vectors*; this allows the designer to estimate soft error resiliency for specific input patterns, as well as random input patterns. (3) The model includes, despite its low likelihood, the possibility of *multiple* soft error events.

The model is based on the Susceptibility Tables of primitive logic gates. The tables for four fundamental gates are shown in Tables 1 and 2 below. The normalized probability of a soft error hit is **Ps**, whereas the normalized probability of NO hit is **1-Ps**. The value of Ps is derived from the literature, based on physical experiments carried out on circuits. The value depends on many parameters including the technology scale used, the altitude and the location worldwide. Estimating this value itself is not the focus of this paper. Our model uses a given probability value (Ps) which allows the analysis to illustrate the relative resilience of a node to soft errors based on input vector values for a given circuit. This model is useful in obtaining a more accurate estimation of soft error susceptibilities of output nodes when more information is known about inputs as is the case in many embedded applications.

Two-Input NAND Gate:

IN	OUT	Prob. for OUT=0 from this input state	Prob. for OUT=1 from this input state
00	1	Ps * Ps	1 – Ps * Ps
01	1	Ps(1-Ps)	1 – Ps(1-Ps)
10	1	Ps(1-Ps)	1 – Ps(1-Ps)
11	0	(1-Ps)*(1-Ps)	1 – (1-Ps)*(1-Ps)

Table 1 *Two-Input NAND Gate Susceptibility Table*

To understand how these values are calculated, take the first table line: For a NAND gate with input 00 the output should be 1. To obtain an erroneous output, i.e. for the output to switch from 1 to 0, both inputs should change to 11 (thus jumping to the last state of the table above). For both inputs to change from 00 to 11, two simultaneous soft error hits are required, hence the probability of Ps * Ps in the table. On the other hand, the probability for the output to equal 1, i.e. no output flip (right column of table), is 1 – Ps * Ps. Because the probabilities in the last two columns of every table line should sum to 1, calculating one value suffices. For the last line of Table 1, if the input is 11, then the output is 0. So, for the output to remain at 0, there should be NO soft error hit on any of the inputs; hence, a probability of (1-Ps)*(1-Ps) shown in the third column. Subtracting this probability from 1 (fourth column of table) gives the probability of the output changing from 0 to 1. This amounts to jumping from this state to any of the other three states.

In a similar fashion, the Susceptibility Table of any gate can be formulated.

Two-Input NOR Gate:

IN	OUT	Prob. for OUT=0 from this input state	Prob. for OUT=1 from this input state
00	1	1 – (1-Ps)*(1-Ps)	(1-Ps)*(1-Ps)
01	0	1 – Ps(1-Ps)	Ps(1-Ps)
10	0	1 – Ps(1-Ps)	Ps(1-Ps)
11	0	1 – Ps * Ps	Ps * Ps
Two-Input XOR Gate:			
00	0	1 – 2*Ps(1-Ps)	2*Ps(1-Ps)
01	1	2*Ps(1-Ps)	1 – 2*Ps(1-Ps)
10	1	2*Ps(1-Ps)	1 – 2*Ps(1-Ps)
11	0	1 – 2*Ps(1-Ps)	2*Ps(1-Ps)
NOT (Inverter) Gate:			
0	1	Ps	1 – Ps
1	0	1 – Ps	Ps

Table 2 *Susceptibility Tables for more Fundamental Gates*

It can be observed that the NAND gate is the logical opposite of the NOR gate. This approach can be extended to any kind of gate, regardless of the number of inputs involved. Note that the Ps for the different gates can be different.

2.2 Model Algorithm and Analysis

This model analyzes the circuit by employing a two-pass methodology, as follows:

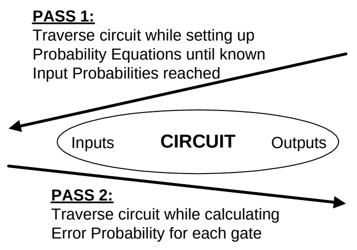
PASS 1:
Traverse circuit while setting up Probability Equations until known Input Probabilities reached
Inputs
CIRCUIT
Outputs
PASS 2:
Traverse circuit while calculating Error Probability for each gate

Fig. 2 *Two-Pass Nature of Model*

The model starts from the output node whose error probability is to be calculated, and then moves backwards through all the gates on the path. While doing so, Error Probability Equations are set up which involve unknown quantities, and the gate's connection information is pushed in a "stack". As soon as known input values are reached, i.e. the start of the circuit, the model can start moving forward again, popping the connection information from the "stack" and calculating the error probabilities based on the now known parameters. As soon as the second pass is finished, the model can calculate the final error probability for the output under investigation.

This algorithm is summarized in pseudo-code form below:

```
for (different input pattern)
{
        pick output node
        while(the input probabilities of the gate are unknown)
        {
              push this gate in the stack
              go backwards to the preceding gate
        }
        while(not having reached the output node)
        {
              pop gate from stack
              calculate the error probability of the gate output
                 using known input probabilities and the gate's
                 susceptibility table
              go forward to the following gate
        }
}
calculate total error probability for the final output using
   given input vector probabilities
```

Fig. 3 *Probabilistic Model Algorithm*

The following example circuit will help illustrate how the proposed model works:

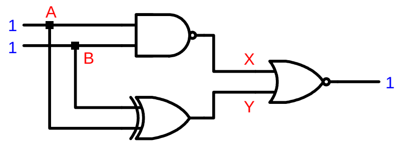

Fig. 4 *Example circuit with Input State=11*

As described in the algorithm, the model starts from the output, and then propagates to the input. Hence, the two input nodes of the last gate (on the right of Fig. 4) are identified as X and Y for the analysis to begin. The probability of an error in the output, given an 11 input, is, by the Total Probability Theorem:

$$\begin{aligned} P\{Error\}_{In\,11} &= P\{Error \mid X{=}0 \,\&\, Y{=}0\} * P\{X{=}0\} * P\{Y{=}0\} \\ &+ P\{Error \mid X{=}0 \,\&\, Y{=}1\} * P\{X{=}0\} * P\{Y{=}1\} \\ &+ P\{Error \mid X{=}1 \,\&\, Y{=}0\} * P\{X{=}1\} * P\{Y{=}0\} \\ &+ P\{Error \mid X{=}1 \,\&\, Y{=}1\} * P\{X{=}1\} * P\{Y{=}1\} \end{aligned} \tag{1}$$

Now, the conditional probabilities for the different value combinations for X and Y can be calculated using the NOR gate Susceptibility Table (see Table 2):

$$\begin{aligned} &P\{Error \mid X{=}0 \,\&\, Y{=}0\} = 1 - (1\text{-}Ps)^2 \\ &P\{Error \mid X{=}0 \,\&\, Y{=}1\} = 1 - Ps\,(1\text{-}Ps) \\ &P\{Error \mid X{=}1 \,\&\, Y{=}0\} = 1 - Ps\,(1\text{-}Ps) \\ &P\{Error \mid X{=}1 \,\&\, Y{=}1\} = 1 - Ps^2 \end{aligned} \tag{2}$$

However, the input probabilities for X and Y, i.e. P{X=0,1} and P{Y=0,1}, are unknown at this point. Therefore, as per the algorithm, the right-most gate is pushed in the stack, and the preceding gates are analyzed.

$$\begin{aligned} P\{X{=}0\} &= P_X\{Error \mid A{=}0 \,\&\, B{=}0\} * P\{A{=}0\} * P\{B{=}0\} \\ &+ P_X\{Error \mid A{=}0 \,\&\, B{=}1\} * P\{A{=}0\} * P\{B{=}1\} \\ &+ P_X\{Error \mid A{=}1 \,\&\, B{=}0\} * P\{A{=}1\} * P\{B{=}0\} \\ &+ P_X\{Error \mid A{=}1 \,\&\, B{=}1\} * P\{A{=}1\} * P\{B{=}1\} \end{aligned} \quad (3)$$

Once again, the conditional probabilities of equation (3) can be calculated using the Susceptibility Table of the NAND gate:

$$\begin{aligned} &P_X\{Error \mid A{=}0 \,\&\, B{=}0\} = Ps^2 \\ &P_X\{Error \mid A{=}0 \,\&\, B{=}1\} = Ps\,(1\text{-}Ps) \\ &P_X\{Error \mid A{=}1 \,\&\, B{=}0\} = Ps\,(1\text{-}Ps) \\ &P_X\{Error \mid A{=}1 \,\&\, B{=}1\} = (1 - Ps)^2 \end{aligned} \quad (4)$$

At this point, however, the input probabilities for A and B, i.e. P{A=0,1} and P{B=0,1}, are known, since the start of the circuit has been reached:

$$\begin{aligned} &P\{A{=}0\} = 0 \qquad P\{A{=}1\} = 1 \\ &P\{B{=}0\} = 0 \qquad P\{B{=}1\} = 1 \end{aligned} \quad (5)$$

So, plugging (4) and (5) into (3),

$$\begin{aligned} &P\{X{=}0\} = (1 - Ps)^2 \\ &P\{X{=}1\} = 1 - P\{X{=}0\} = 1 - (1 - Ps)^2 \end{aligned} \quad (6)$$

Similarly,

$$\begin{aligned} P\{Y{=}0\} &= P_Y\{Error \mid A{=}0 \,\&\, B{=}0\} * P\{A{=}0\} * P\{B{=}0\} \\ &+ P_Y\{Error \mid A{=}0 \,\&\, B{=}1\} * P\{A{=}0\} * P\{B{=}1\} \\ &+ P_Y\{Error \mid A{=}1 \,\&\, B{=}0\} * P\{A{=}1\} * P\{B{=}0\} \\ &+ P_Y\{Error \mid A{=}1 \,\&\, B{=}1\} * P\{A{=}1\} * P\{B{=}1\} \end{aligned} \quad (7)$$

The conditional probabilities of equation (7) can be calculated using the Susceptibility Table of the XOR gate:

$$\begin{aligned} &P_Y\{Error \mid A{=}0 \,\&\, B{=}0\} = 1 - 2{*}Ps(1\text{-}Ps) \\ &P_Y\{Error \mid A{=}0 \,\&\, B{=}1\} = 2{*}Ps(1\text{-}Ps) \\ &P_Y\{Error \mid A{=}1 \,\&\, B{=}0\} = 2{*}Ps(1\text{-}Ps) \\ &P_Y\{Error \mid A{=}1 \,\&\, B{=}1\} = 1 - 2{*}Ps(1\text{-}Ps) \end{aligned} \quad (8)$$

Plugging (8) and (5) into (7),

$$\begin{aligned} &P\{Y{=}0\} = 1 - 2{*}Ps(1\text{-}Ps) \\ &P\{Y{=}1\} = 1 - P\{Y{=}0\} = 2{*}Ps(1\text{-}Ps) \end{aligned} \quad (9)$$

Pass 1 of the algorithm has now been concluded and Pass 2 can commence moving forward toward the output node. The right-most gate can now be popped from the stack and calculated, since all equation parameters are now known. Plugging (2), (6) and (9) into (1) gives:

$$\begin{aligned} P\{Error\}_{In\,11} &= \left[1 - (1\text{-}Ps)^2\right] * (1\text{-}Ps)^2 * \left[1 - 2{*}Ps(1\text{-}Ps)\right] \\ &+ \left[1 - Ps\,(1\text{-}Ps)\right] * (1\text{-}Ps)^2 * 2{*}Ps(1\text{-}Ps) \\ &+ \left[1 - Ps\,(1\text{-}Ps)\right] * \left[1 - (1\text{-}Ps)^2\right] * \left[1 - 2{*}Ps(1\text{-}Ps)\right] \\ &+ \left[1 - Ps^2\right] * \left[1 - (1\text{-}Ps)^2\right] * 2{*}Ps(1\text{-}Ps) \end{aligned} \quad (10)$$

The stack is now empty, since the output node has been reached. Pass 2 of the algorithm has been concluded for Input Vector 11. The two-pass algorithm is repeated for all input combinations. This way, $P\{Error\}_{In\,00}$, $P\{Error\}_{In\,01}$ and $P\{Error\}_{In\,10}$ are also calculated.

After all input combinations are calculated, the last two lines of the algorithm of Fig. 3 are executed. This involves calculating the Total Error Probability for the output node under investigation:

$$\begin{aligned} P\{Total\ Error\} &= P\{Input{=}00\} * P\{Error\}_{In\,00} \\ &+ P\{Input{=}01\} * P\{Error\}_{In\,01} \\ &+ P\{Input{=}10\} * P\{Error\}_{In\,10} \\ &+ P\{Input{=}11\} * P\{Error\}_{In\,11} \end{aligned} \quad (11)$$

The input vector probabilities are accounted for in the terms P{Input=00}, P{Input=01}, P{Input=10}, and P{Input=11}. In arithmetic circuits, the inputs are usually random, but in control circuits the input vectors could be skewed toward 1 or 0. This model provides the flexibility of accounting for such unbalanced input values.

Hence, by repeating the two-pass algorithm for all input patterns, the model fully analyzes the circuit and calculates a soft error probability for any output node, based on logical masking principles.

A potential problem to the proposed methodology arises when the number of inputs increases dramatically. Because the model exhaustively analyzes all input combinations, the calculation time grows exponentially with the number “n” of inputs (2^n). However, there are two methods, one of which is used extensively in the Functional Verification field and known as *Pseudo-Exhaustive Testing*, which alleviate this conundrum. (1) Pseudo-Exhaustive Testing breaks a circuit having n primary inputs into smaller, overlapping partitions, each with less than n inputs. Each of these partitions is then tested exhaustively in parallel. Our model inherently exhibits one such pseudo-exhaustive technique known as *Cone Segmentation*. The latter allows the partitioning of large circuits into fan-in cones by backtracking from each primary output, through the circuit, to the inputs which influence the output. Each fan-in cone can then be tested exhaustively in parallel. *Sensitized Path Segmentation* is another technique which breaks the inputs affecting the output into logical segments and then analyzes each segment individually [10]. (2) By studying the Susceptibility Tables of individual gates, a designer can identify *Equivalent Input Vectors*. For example, Table 2 indicates that inputs 01 and 10 are equivalent for a two-input XOR gate. Thus, only one of them needs to be tested. Using input equivalence, input vectors can collapse to manageable sizes.

3. Applications

3.1 Circuits and Methodology

To validate the proposed model, two circuits were analyzed using the model algorithm, and the results compared to Verilog HDL simulations. MATLAB was used to implement the model algorithm. To simulate the circuits, short pulses were injected randomly and errors in the output observed. Simulations of 2000 transient-event injections were run to obtain a stochastic soft error rate. The circuits used were the example circuit from Section 2 of this paper and a Full Adder (see Fig. 5 below). The latter was chosen because of its three inputs and two outputs, which are ideal to illustrate the capabilities of the model.

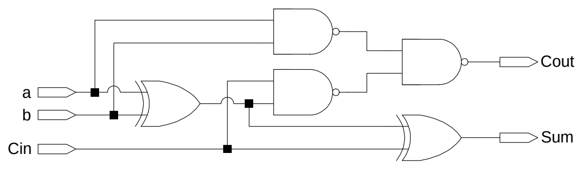

Fig. 5 *Full Adder Circuit*

Ps is a normalized SER value. The normalization was deemed necessary to accelerate the number of errors observed in reasonable amount of simulation time and provide enough errors to make the correlation between the simulation and the model meaningful. The normalized Ps value is significantly larger, yet directly related to the actual SER value. Ps is the average Neutron Flux at sea-level. It was chosen to be 0.0045, which is the average Neutron Flux in neutrons*cm^{-2}sec^{-1} between New York City and Tokyo [5]. The Neutron Flux value varies with location and altitude around the world. The values mentioned in the literature are for New York City and Tokyo. The true SER value is the Neutron Flux value multiplied by the Atmospheric Neutron Cross Section [1]. This Ps value is appropriate to show the relative soft error resilience of any output node for any input combination. In this evaluation, the same value of Ps is used and our model is used to compare resiliency of nodes in different circuits. However, it must be observed that our model is more flexible and can permit the use of different Ps values for different gate primitives.

3.2 Results

The results for both circuits are summarized in Tables 3 and 4 below:

Example Circuit from Section 2:

Input Pattern	Model Prediction (0 < P < 1)	Normalized Model Prediction	Simulation Results (No. of SE)	Normalized Simulation Results
00	0.0045	37.63	418	46.44
01	1.1960e-4	1	9	1
10	1.1960e-4	1	9	1
11	0.0266	222.41	1999	222.11

Table 3 *Simulation Results for Example Circuit in Section 2*

The normalized columns indicate the relative soft error susceptibility of the output for each input pattern. The normalizing value (i.e. 1, the smallest value in the column) indicates the input pattern that is most resilient to errors at the output node. For example, the model predicts that the output of the circuit is about 222 times more susceptible to soft errors for input pattern 11 as compared to using input patterns 01 and 10 due to the logical masking capability. The latter two are the most resilient input patterns.

Full Adder (Sum Output):

Input Pattern	Model Prediction (0 < P < 1)	Normalized Model Prediction	Simulation Results (No. of SE)	Normalized Simulation Results
000	0.0178	1	1014	1
001	0.0178	1	1014	1
010	0.0178	1	1014	1
011	0.0178	1	1014	1
100	0.0178	1	1014	1
101	0.0178	1	1014	1
110	0.0178	1	1014	1
111	0.0178	1	1014	1

Full Adder (Carry-Out Output):

Input Pattern	Model Prediction (0 < P < 1)	Normalized Model Prediction	Simulation Results (No. of SE)	Normalized Simulation Results
000	0.0091	1	416	1
001	0.0221	2.43	1223	2.94
010	0.0178	1.95	787	1.89
011	0.0220	2.41	1214	2.92
100	0.0178	1.95	811	1.95
101	0.0220	2.41	1214	2.92
110	0.0133	1.46	579	1.39
111	0.0132	1.44	579	1.39

Table 4 *Simulation Results for Full Adder Circuit*

The top portion of Table 4 concerns the Sum output of the Full Adder. The model predicts that all input combinations will be equally susceptible to SEU. The reason is the symmetrical nature of the XOR gate which affects the sum output (see Fig. 5). As seen in the XOR Susceptibility Table (Table 2), the probability of a soft error from any input state of an XOR gate is exactly the same, i.e. 2*Ps(1-Ps). The model prediction is verified by the simulation results on the right-hand side of Table 4 above. The Carry-Out output path includes a series of cascaded gates, and, thus, does not exhibit any symmetrical behavior.

Comparing the Normalized Model Prediction and the Normalized Simulation Results columns, it can be seen that the model is fairly accurate in predicting the relative soft error susceptibility of the output node for each input pattern. The small inaccuracy is attributed to the relatively small number of simulations for such small error probabilities. The experimental values slowly converge to the model estimation as the number of simulations increases.

4. Conclusions and Future Work

A soft error rate estimation model for combinational logic, based on logical masking properties, was proposed. The model is able to determine the soft error resiliency of any output node in a circuit, for any input pattern. The strength of this model lies in its ability to produce individual SER values for any input combination, thus offering the designer the ability to accurately estimate SER susceptibility for unbalanced input vectors, such as those found in control-signal circuitry. Moreover, the designer can identify highly susceptible input patterns and modify the design accordingly. Simulation results validate the functionality and accuracy of the model.

Work is currently under way to apply this model to more complex combinational logic, including sequential circuits. The model will also be extended to include electrical and window-latching masking effects, by using a variable Ps value and additional parameters.

References

[1] P. Hazucha and C. Svensson. “Impact of CMOS Technology Scaling on the Atmospheric Neutron Soft Error Rate,” *IEEE Transactions on Nuclear Science, Vol. 47, No. 6,* pages 2586–2594, Dec 2000.

[2] P. Liden, P. Dahlgren, R. Johansson, and J. Karlsson. “On Latching Probability of Particle Induced Transients in Combinational Networks,” *Proceedings of the 24th Symposium on Fault-Tolerant Computing (FTCS-24)*, pages 340–349, 1994.

[3] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi. “Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic,” *Proeedings of the.Inernational. Conference on Dependable Systems and Networks (DSN’02).*

[4] L. W. Massengill, A. E. Baranski, D. O. V. Nort, J. Meng, and B. L. Bhuva. “Analysis of Single-Event Effects in Combinational Logic – Simulation of the AM2901 Bitslice Processor,” *IEEE Trans. on Nuclear Science,* 47(6):2609–2615, Dec 2000.

[5] Y.Tosaka, H.Kanata, S.Satoh, and T.Itakura. “Simple Method for Estimating Neutron-Induced Soft Error Rates Based on Modified BGR Model,” *IEEE Electron. Device Letters, Vol. 20,* pp. 89-91, Feb 1999.

[6] M. Baze and S. Buchner. “Attenuation of Single Event Induced Pulses in CMOS Combinational Logic,” *IEEE Transactions on Nuclear Science*, 44(6), Dec 1997.

[7] S. Buchner, M. Baze, D. Brown, D. McMorrow, and J. Melinger. “Comparison of Error Rates in Combinational and Sequential Logic,” *IEEE Transactions on Nuclear Science*, 44(6):2209–2216, Dec 1997.

[8] T. Juhnke and H. Klar. “Calculation of the soft error rate of submicron CMOS logic circuits,” *IEEE Journal of Solid State Circuits,* 30:830–834, Jul 1995.

[9] J. Pickel. “Effect of CMOS miniaturization on cosmic-rayinduced error rate,” *IEEE Transactions on Nuclear Science*, 29:2049–2054, Dec 1982.

[10] M. L. Bushnell and V. D. Agrawal, *Essential of Electronics Testing forDigital, Memory and Mixed-Signal Circuits*. Kluwer Academic Publishers, 2001.