International Journal of Computational Research and Development (IJCRD)
Impact Factor: 4.775, ISSN (Online): 2456 - 3137
(www.dvpublication.com) Volume 1, Issue 2, 2016

EVALUATING TRANSIENT ERRORS IN ELECTRICAL CIRCUITS

V. Sharmilaraj* & Natarajan Somasundaram**
* Associate Professor, Department of Electronics and Communication Engineering, SSM College of Engineering, Komarapalayam, Tamilnadu
** Professor, Department of Electronics and Communication Engineering, Karpagam College of Engineering, Coimbatore, Tamilnadu

Cite This Article: V. Sharmilaraj & Natarajan Somasundaram, "Evaluating Transient Errors in Electrical Circuits", International Journal of Computational Research and Development, Volume 1, Issue 2, Page Number 22-25, 2016.

Abstract:
With the advent of VLSI technology, the systems fabricated in deep sub micron technology are more prone to errors like intermittent or transient faults, causing unidirectional errors. Research has been established to provide protection against these transient faults to improve the overall reliability of the system. This paper provides a comprehensive survey of fault detection techniques and describe in brief the promising techniques comes under coding techniques that allow the detection of these errors and their performance evaluation.

Key Words: Unidirectional Errors, Transient Faults, Error Detection Coding, Resource Overhead, Performance Overhead, SRAM, Latency & Scalable Error Detection Coding (SEDC)

1. Introduction:

A fault in a system is some deviation from the expected behavior of the system. Faults can be classified into, Permanent faults in which the type of failure is persistent and continues to exist until the faulty component is repaired or replaced , Intermittent faults where a malfunction of a device or system that occurs at intervals, usually irregular and Transient faults that occur once and then disappear. Comparing transient types, intermittent faults are the most annoying of component faults. In today's integrated circuits dominant type of faults are Transient faults and 98% of Random Access Memories (RAM) faults are transient faults. But Dynamic RAMs (DRAM) experience one single bit transient fault out of its memory per day. These faults are mostly caused by environmental effects such as alpha particles, atmospheric neutrons, electro static discharge, electrical power drops, overheating, cosmic ray particles and random noise due to signal integrity (e.g. wire cross talk). Faults in VLSI circuits can give rise to many types of errors and they occur in logic circuits and memory systems. Such errors belong to one of the following classes: symmetric, unsymmetric, and unidirectional. Symmetric errors have both 0 to 1 and 1 to 0 transitions errors and occur with equal probability in a code word whereas unsymmetric errors have only one type of error i.e 0 to 1 or 1 to 0 but not both can occur in a code word. Unidirectional errors have both 0 to 1 and 1 to 0 errors but they do not occur simultaneously in any code word. Transient faults are likely to cause a limited number of symmetric errors or multiple unidirectional errors. Also, intermittent faults, because of short duration, are expected to cause a limited number of errors. On the other hand, permanent faults cause either symmetric or unidirectional errors, depending on the nature of the faults. The most likely faults in developed LSI/VLSI, ROM, and RAM memories (such as the faults that affect address decoders, word lines, power supply, and stuck-fault in a serial bus, etc.) cause unidirectional errors. This paper provides an overview of several techniques that allow the detection of transient error in a circuit.

2. Related Works:

2.1 Various Fault Detection Techniques: Electric circuit testing is the standard approach to detect the transient errors where the defective circuit is compared with simulated circuit model with selected logic inputs. The dissimilar output implies the defect in the circuit, and the defect is also due to the feeding of inputs with high/low temperature at different frequencies that are performed at the production site. These errors can be detected by BIST (Built-in-Self Test), Roving technique, redundancy method, logic implications and error coding techniques. BIST is a fault détection technique in which parts of a circuit are used to test the circuit itself using online or offline testing and the design for test techniques cause a significantly larger area overhead. Roving detection utilizes run-time reconfiguration to carry out BIST techniques on-line, in which the computational array split into equal-sized regions. One of these regions is configured to carryout self-test, while the others carry out the design function of the array and the test region is swapped with functional regions one at a time so that the entire array can be tested while the system remains functional. Due to swapping process the testing speed is slow. The redundancy techniques protect systems against operational faults like, hardware (additional components), software (special programs), and time (repetition of operations). Hardware redundancy, such as dual modular redundancy (DMR) and triple modular redundancy (TMR) uses voting logic to determine the faults and to improve the reliability of a system. Space redundancy and time redundancy are the two kinds. Space redundancy provides additional components, functions, or data items that are redundant for fault-free operation. But in time redundancy the computation is repeated and the result is compared to a stored copy of result. Redundancy techniques have limitation with area or latency overhead. Logic implication method uses existing design and searches all internal circuit nodes for consistent logic patterns between them. When a

distinct pattern is found, it will append some simple checker hardware that will reinforce the validity of that relationship. The drawback of this technique is that it is not well suited for concurrent error detection.

Table 1: Summary of Error Detection Techniques

Error Detection Technique	Speed of Detection	Resource Overhead	Performance Overhead	Fault Coverage
Modular Redundancy	Fast	Very Large – Triplicate plus voting logic	Very Small – Latency of voting logic	Good All manifest errors are detected.
Off-line BIST	Slow	Very small	Small – Slight start-up delay	Very Good All faults including dormant.
Roving	Medium	Medium – Empty test block plus test controller	Large – Clock must be stopped to swap blocks.	Very Good Multiple Manifest and latent faults are detected.

2.2 Detection of Transient Errors with Coding Techniques:

2.2.1 Parity Code: During data transmission an electrical noise can corrupt the transmission signal and can alter the logic level of the signal, introducing errors in the transmitted signal. For the concurrent error detection the first step is to select a parity check bit which may be even or odd for encoding the outputs of the circuit. This parity bit is chosen such that the total number of 1's in the transmitted signal should produce an odd or even number of 1's according to their type odd parity or even parity. It adds one parity bit P with transmitted signal which is computed by the modulo-2 sum of all the information bits during its encoding as,

$$P = x_1 \oplus x_2 \oplus x_3 \oplus \ldots\ldots\ldots\ldots \oplus x_n \quad \text{----------(1)}$$

Where P is the encoded value for the x_k binary input values of the transmitted signal. Though Parity bit checking is the simplest form of error checking and requires limited hardware overhead, this system will not catch any double errors. When even numbers of bits in the information are flipped it will produce proper parity bit with an erroneous output.

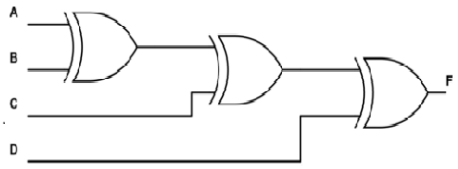

Figure 1: Basic Parity Generator

2.2.2 Hamming Code: Hamming code was invented by Richard Hamming in 1950 is a linear code for error detection that can detect up to two simultaneous bit errors and is capable of correcting single-bit errors. Reliable communication is assured if the hamming distance between the transmitter and receiver is less than or equal to one. Hamming codes extend the idea of parity to include an error correction as well as an error-detection scheme. The crucial idea behind the Hamming code system is that a unique number is generated by parity error which uniquely identifies the bit which is in error.

2.2.3 RS-Codes: Reed-Solomon codes are error correcting codes, in which redundant information is added to data so that it can be recovered reliably despite errors in transmission or storage and retrieval. RS differs from a Hamming code in that it encodes groups of bits instead of one bit at a time and rely on special properties of finite-arithmetic Galois Field (GF) operations. They are subset of BCH codes specified as RS (n, k), with m bit symbols. i.e the encoder takes k data symbols of m bits each, appends n – k parity symbols, and produces a code word of n symbols (each of m bits. For an RS (n, k) code where n – k = 2T, the decoder can correct up to T symbol errors in the code word and they are best for correcting burst errors.

2.2.4 Berger Code: Berger code (BC) is an optimal separable code which can detect all unidirectional errors .The check bits(k) for these codes are detected from word with a length of n bits which has m information bits as $k=\log_2 (m+1)$ & n = m+k. Two different schemes used to generate the check bits for Berger codes are B1 encoding scheme and B0 encoding scheme. First scheme counts the number of 1's in the information bits and bit-by-bit complement of its equivalent binary value is considered as the check bits. Similarly the later performs the same task but it counts only the number of 0's in the information bits. If the number of information bits m = (2k -1), where k≥1, then it is called as maximal length Berger code; otherwise it is called as the non-maximal length Berger code. For example, if m = 1010101, $k = [\log_2 (7+1)] =3$ and the Berger code must have a length of 10. For B1 encoding scheme the check bits are derived as 011 [Number of 1s in the information bits = 4(in binary 100) & bit-by-bit complement of 100 is 011]. Thus, the code word is 1100101 011. Berger code not offers error correction capability and only used to detect all unidirectional errors. They are optimal and no extra decoders are required to extract the information bits.

2.2.5 Bose-Lin Code: A Bose-Lin code is an optimal systematic code detects t unidirectional errors that require a fixed number of check bits, independent of the number of information bits. An efficient technique is used to append the check bits to the output bits together with a single checker thereby reducing overhead. Similar to

Berger codes the codes are designed to count the number of zeros in the information bits and the counts are then modified depending on t. For the double and triple errors, the counts are performing modulo 4 and 8 to detect the unidirectional errors. If it counts the number of one's then the error detection would fail for the Bose-Lin code. Similar to Berger error correction capability is not available in Bose Lin.

Table 2: Comparative Performance of Codes

Coding Technique	**Parity Code**	**Hamming Code**	**RS Code**	**Berger Code**	**Bose-Lin Code**
Resource overhead	Very less	High	Very High	Very High	Medium compared to Berger
Error detection	Single or all odd number of errors	Single and double errors	Not all unidirectional errors	All unidirectional errors	Not all unidirectional errors
Performance overhead	Very less	Less	High	Very High	Very High
Number of Check bits(k) & Information(m)	P=1	2^k-k-1>m	k =2 × t-number of errors to be corrected	k = $\log_2(m+1)$	$(5\times2^{k-4})$+k-4
Speed	High	low	low	High	High

2.3 Scalable Error Detection Coding (SEDC): In space applications SRAM based FPGA are subjected to radiation environments which make vulnerable effects in SRAM cells. When high energy particles in terrestrial cosmic rays strike the SRAM cells they change the logic values of memory from 0 to 1 or 1 to 0 , but not both at the same time. An error detection scheme must provide good fault coverage against these unidirectional errors and error coding techniques are more efficient than various fault detection techniques. Among different types of codes, Berger codes have 100% unidirectional fault detection where the CRC generator circuits are bulky and produce much delay. Other error detection circuits consume much area for constructing the circuit. SEDC (Scalable Error Detection coding) improve the performance in terms of area, speed and delay by portioning the data into segments of 2, 3 and 4 bits and encode these segments using SEDC codes in parallel fashion. Though SEDC and TMR methods have bigger code size than Berger Scheme, the total area utilized by both SEDC and TMR takes less area than Berger Scheme. If overall area including the comparator part is considered, then SEDC requires less area and delay than Berger and TMR methods. With the increase in input binary data, latency and area increases whereas the maximum latency of SEDC checker limited to 2 gate levels does not affect the optimized LUT performance. While considering unidirectional error in programming bits of FPGA or pre-stored check bits, SEDC scheme is faster, easily scalable and area efficient compared to Berger and TMR techniques.

3. Conclusion:

Detection of transient errors using the reviewed methods have their strength and weakness and the reliability of the system can be improved by these codings. Some of these codings are effective and may require cost, more area overhead, or impractical power and time requirements. To achieve 100% unidirectional error detection in terms of speed, latency and area SEDC scheme found efficient than others and it is up to designers to choose the effective method.

4. References:

1. Natarajan Somasundaram, Jeong A Lee, Farhad Mehdipour, Ramadass Narayanadass, Y V Ramana Rao, (2013),'Scalable Error Detection Coding© Algorithm for Totally Self-Checking (TSC) Circuits SEDC© Algorithm for TSC Circuits"Consumer Electronics Times, Vol. 2 Iss. 3, PP. 116-123.
2. N. Alves, "State-of-the-Art Techniques for Detecting Transient Errors in Electrical Circuits", in IEEE Potentials, vol. 30, no. 3, pp. 30- 35, 2011.
3. E. Stott, P. Sedcole, and P. Cheung, "Fault tolerant methods for reliability in FPGAs", in Proc. International Conference on Field Programmable Logic (FPL), pp.415-420, 2008.
4. D. K. Pradhan, and J. J. Stiffler, "Error-Correcting Codes and Self-Checking Circuits", in Computer, vol. 13, no. 3, pp. 27-37, 1980.
5. J. M. Emmert, C. E. Stroud, and M. Abramovici, "Online Fault Tolerance for FPGA Logic Blocks", in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, no. 2, pp. 216-226, 2007.
6. X. She, and K. S. McElvain, "Time Multiplexed Triple Modular Redundancy for Single Event Upset Mitigation", in IEEE Transactions on Nuclear Science, vol. 56, no. 4, pp. 2443-2448, 2009.
7. K. Nepal, N. Alves, J. Dworak, and R. I. Bahar, "Using Implications for Online Error Detection", in Proc. IEEE International Test Conference (ITC 2008), 2008, pp. 1-10.
8. S. J. Piestrak, "Design of Fast Self-Testing Checkers for a Class of Berger Codes", in IEEE Transactions on Computers, vol. C-36, no. 5, pp. 629-634, 1987.

9. A. Morozov, V. V. Saposhnikov, VI. V. Saposhnikov, and M. Gossel, "New self-checking circuits by use of Berger-codes", in Proc. 6th IEEE International On-Line Testing Workshop, 2000, pp. 141-146.
10. A. Bose, and D. J. Lin, "Systematic Unidirectional Error-Detecting Codes", in IEEE Transactions on Computers, vol. C-34, no. 11, pp. 1026-1032, 1985.
11. H. Dong, "Modified Berger Codes for Detection of Unidirectional Errors", in IEEE Transactions on Computers, vol. C-33, no. 6, pp. 572-575, 1984.
12. D.A. Pierce Jr, and P.K. Lala, "Modular Implementation of Efficient Self-Checking Checkers for the Berger Code," J. of Electronic Testing: Theory and Applicat., vol. 9, no. 3, pp. 279-294, 1996.
13. R.C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," IEEE Trans. Device Mater. Reliabil., vol.5, no3, pp. 301-316, sep. 2005.
14. J. R. Schwank, M. R. Shaneyfelt, D. M. Fleetwood, J. A. Felix, P. E. Dodd, P. Paillet, and V. F. Cavrois, "Radiation Effects in MOS Oxides", in IEEE Transactions on Nuclear Science, vol. 55, no. 4, pp. 1833-1853, 2008.