International Journal of Inventive Engineering and Sciences (IJIES)
ISSN: 2319–9598, Volume-1 Issue-11, October 2013

Design of Fault Tolerant Memory System with Difference Set Cyclic Codes

Swarnalatha Eluri, Hemalatha Rallapalli

***Abstract*: The problem of single event upset (SEU) due to higher integration, smaller dimensions and lower voltages is very common and need to be addressed. The effect of SEU is not only present at the terrestrial environments but also at the ground level. The SEUs also result in silent data corruption which results in the further corruption of data, especially in memories. A special class of LDPC codes called Difference Set Cyclic Codes (DSCC) is used to design a fault tolerant memory system that detects the silent data corruption. The DSCC is simple and easy to implement.**

***Index Terms*: Difference Set Cyclic Codes (DSCC), LDPC, Majority Logic Fault Detector (MLDD), Single Event Upsets (SEU),.**

I. INTRODUCTION

As the dimensions and operating voltages of computer electronics are reduced to satisfy the consumer's insatiable demand for higher density, functionality, and lower power, their sensitivity to radiation increases dramatically. There are a plethora of radiation effects in semiconductor devices that vary in magnitude from data disruptions to permanent damage ranging from parametric shifts to complete device failure [1], [2]. Of primary concern for commercial terrestrial applications are the "soft" single-event effects (SEEs) as opposed to the "hard" SEEs and dose/dose-rate related radiation effects that are predominant in space and military environments. As the name implies, SEEs are device failures induced by a single radiation event.[R]

A soft error occurs when a radiation event causes enough of a charge disturbance to reverse or flip the data state of a memory cell, register, latch, or flip-flop. The error is "soft" because the circuit/device itself is not permanently damaged by the radiation—if new data are written to the bit, the device will store it correctly. The soft error is also often referred to as a single event upset (SEU). If the radiation event is of a very high energy, more than a single bit maybe affected, creating a multi-bit upset (MBU) as opposed to the more likely single bit upset (SBU). While MBUs are usually a small fraction of the total observed SEU rate, their occurrence has implications for memory architecture in systems utilizing error correction.

Another type of soft error occurs when the bit that is flipped is in a critical system control register such as that found in field-programmable gate arrays (FPGAs) or dynamic random access memory (DRAM) control circuitry, so that the error causes the product to malfunction [5]. This type of soft error, called a single event interrupt (SEFI), obviously impacts the product reliability since each SEFI leads to a direct product malfunction as opposed to typical memory soft errors that may or may not affect the final product operation depending on the algorithm, data sensitivity, etc.

A. Errors in Memories- Mitigation techniques

Some commonly used mitigation techniques are:

- triple modular redundancy (TMR);
- error correction codes (ECCs).

TMR is a special case of the von Neumann

method consisting of three versions of the design in parallel, with a majority voter selecting the correct output. As the method suggests, the complexity overhead would be three times plus the complexity of the majority voter and thus increasing the power consumption.

For memories, it turned out that ECC codes are the best way to mitigate memory soft errors.

For terrestrial radiation environments where there is a low soft error rate (SER), codes like single error correction and double error detection (SEC–DED), are a good solution, due to their low encoding and decoding complexity. However, as a consequence of augmenting integration densities, there is an increase number of soft errors, which produces the need for higher error correction capabilities.

The usual multi error correction codes, such as Reed–Solomon (RS) or Bose–Chaudhuri– Hocquenghem (BCH) are not suitable for this task. The reason for this is that they use more

sophisticated decoding algorithms, like complex

algebraic (e.g., floating point operations or

logarithms) decoders that can decode in fixed time, and simple graph decoders, that use iterative

Among the ECC codes that meet the requirements of higher error correction capability

and low decoding complexity, cyclic block codes have been identified as good candidates, due to their property of being majority logic (ML) decodable. A subgroup of the low-density parity

check (LDPC) codes, which belongs to the family of the ML decodable codes, has been researched in.

In this paper, we will focus on one specific type of LDPC codes, namely the difference-set cyclic codes (DSCCs), which is widely used in the Japanese teletext system or FM multiplex broadcasting systems. The main reason for using ML decoding is that it is very simple to implement and thus it is very practical and has low complexity.

Manuscript received October 2013.

Swarnalatha Eluri, Assistant Professor, Department of ECE, Jawaharlal Nehru Institute of Technology, Hyderabad., (swarna.koli@gamil.com).

Hemalatha Rallapalli, Assistant Professor, Department of ECE, University College of Engineering, Osmania University, Hyderabad., (hemalatha.rallapalli@gmail.com.

.

The drawback of ML decoding is that, for a coded word of -bits, it takes cycles in the decoding process, posing a big impact on system performance. One way of coping with this problem is to implement parallel encoders and decoders. This solution would enormously increase the complexity and, therefore, the power consumption. As most of the memory reading accesses will have no errors, the decoder is most of the time working for no reason. This has motivated the use of a fault detector module that checks if the codeword contains an error and then triggers the correction mechanism accordingly. In this case, only the faulty code words need correction, and therefore the average read memory access is speeded up, at the expense of an increase in hardware cost and power consumption. A similar proposal has been presented in for the case of flash memories. The simplest way to implement a fault detector for an ECC is by calculating the syndrome, but this generally implies adding another very complex functional unit. This paper explores the idea of using the ML decoder circuitry as a fault detector so that read operations are accelerated with almost no additional hardware cost. The results show that the properties of DSCC-LDPC enable efficient fault detection. This paper explores the idea of using a simple and effective coding technique called Difference Set Cyclic Codes, for the detection of silent data corruption. The remainder of this paper is organized as follows. Section II presents the Methodology for designing the proposed memory system. Section III presents the Design of Fault Tolerant Memory System. Section IV gives the comparison of different MLD techniques. Section V discusses the simulation results for the proposed MLDD. Section VI discusses the conclusion and future scope.

II. METHODOLOGY

The Fault Tolerant Memory System is modelled using Verilog HDL, and the functionality is verified using Active HDL 7.1 / Xilinx 9.1i simulation tool. The synthesis is also done in Xilinx 9.1i.

III. DESIGN OF FAULT TOLERANT MEMORY SYSTEM

The memory system designed using Difference Set Cyclic Codes is depicted in Fig.1. The message is encoded using the encoder and stored in the memory. The suspected memory words are read out to the Memory and checked for errors in the MLDD.

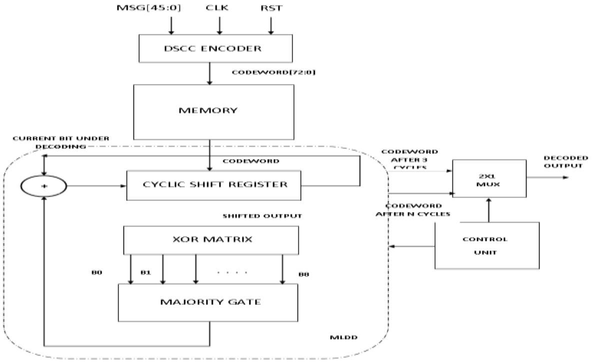


Fig.1. Fault Tolerant Memory System.

A. DSCC Encoder

The generator polynomial for the given difference set is

$$Z(X) = 1 + X^2 + X^4 + X^6 + X^8 + X^{12} + X^{16} + X^{22} + X^{25} + X^{28}$$

The code word is generated by the encoder. The 45-bit message is connected at IN and the 28-bit parity is generated with the help of LFSR which is shown in Fig.2. This 28-bit is combined with the 45-bit message in order to make the 73 bit Difference Set Cyclic Code.

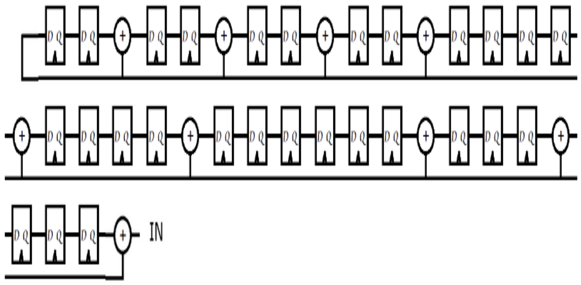


Fig.2. DSCC Encoder

B. Majority Logic Detector/Decoder

A detailed schematic of the existing Majority logic detector/decoder is shown in Fig.3. As described before, the ML decoder is a simple and powerful decoder, capable of correcting multiple random bit flips depending on the number of parity check equations. It consists of four parts:

1) A cyclic shift register;
2) An XOR matrix;
3) A majority gate;
4) An XOR for correcting the codeword bit under decoding;
5) The Control Unit;
6) The Output tri-state buffer;

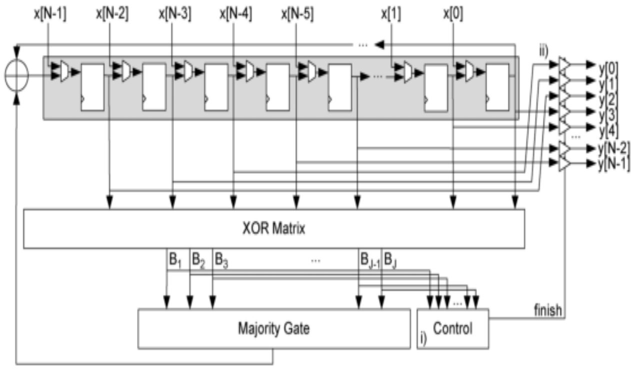


Fig.3. Majority Logic Detector/Decoder

The Fig.3 shows the basic ML decoder with an N tap shift register, an XOR array to calculate the orthogonal parity check sums and a majority gate for deciding if the current bit under decoding needs to be inverted.

C.Control Unit for MLDD

The control unit triggers a finish-flag when no errors are detected after the third cycle

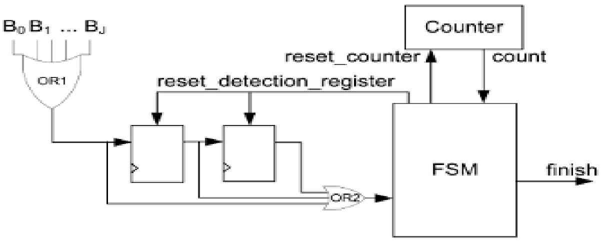


Fig.4. Control Unit for MLDD

The control schematic is illustrated in Fig.4. The control unit manages the detection process. It uses a counter that counts up to three, which distinguishes the first three iterations of the ML decoding. In these first three iterations, the control unit evaluates the $\{B_j\}$ by combining them with the OR1 function. This value is fed into a three-stage shift register, which holds the results of the last three cycles. In the third cycle, the OR2 gate evaluates the content of the detection register. When the result is "0," the FSM sends out the finish signal indicating that the processed word is error-free. In the other case, if the result is "1," the ML decoding process runs until the end.

The output tri-state buffers are always in high impedance unless the control unit sends the finish signal so that the current values of the shift register are forwarded to the output.

D.MLDD Flow Chart

The flow chart for the above MLDD is shown in Fig.5. MLDD Algorithm checks for the error in the first 3 cycles. If the error is detected in the first three cycles only then all the N-nits are detected and decoded. Otherwise, the codeword is declared error free.

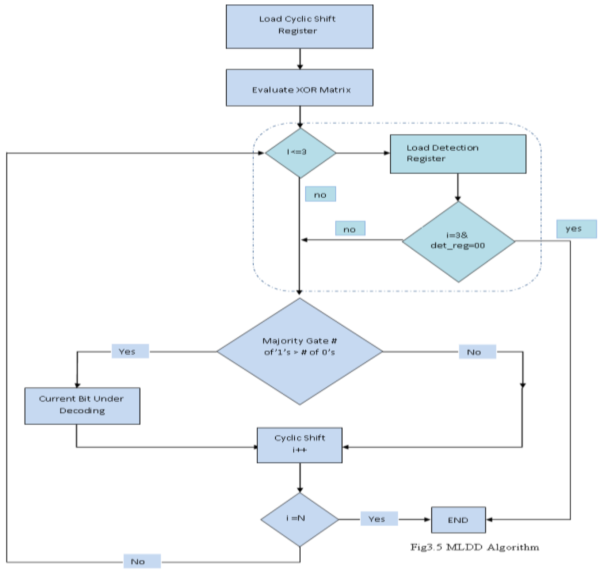


Fig.5. MLDD flow chart

E. Proposed Algorithm

The flow chart for the proposed Novel MLDD system is shown in Fig.6.

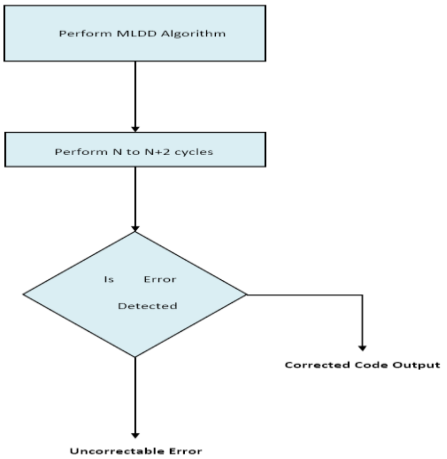


Fig.6. Proposed Algorithm

IV. COMPARISON OF DIFFERENT MLD TECHNIQUES

The Decoder for MLDD memory system is designed based on Difference Set Cyclic Codes. Difference Set Cyclic Codes is one of the LDPC codes which is the very simplest and most suitable method for correction in memory applications [11]. In this proposed technique two more error detection cycles are included, which will detect the silent data corruption. This is a great improvement over the earlier Majority Logic Decoding Algorithm.

The MLDD is a simple and powerful EDAC technique. Using MLDD up to 16 single event upsets may be corrected for a (1057, 813) DSCC. The DSCC error correction capability is given in Table-I.

TABLE I Difference Set Code Parameters

Code Length(N)	Message Bit(k)	Correctable Errors
21	11	2
73	45	4
273	191	8
1057	813	16

The Novel MLDD requires two extra cycles which will enable us to detect more than five errors also. The comparison of the Novel MLDD with the previous MLD techniques is given in Table II and Table III.

TABLE II Error Correction Capability of Different MLDs

Algorithm	No. of Cycles to detect the error	No of Errors Corrected
Plain MLD	N	2^{s-1}(4 errors for s=3)
MLDD	3	2^{s-1}(4 errors for s=3)
Novel MLDD	3	2^{s-1}(4 errors for s=3)

TABLE III Error Detection Capability of Different MLDs

Algorithm	No. of Cycles to detect the error	No of Errors Detected
Plain MLD	N	5
MLDD	N+3	5
Novel MLDD	N+5	More than 5

IV. SIMULATION RESULTS

This Section presents the simulation results for the proposed memory system. Fig.7(a) to 7(d) shows the simulation results for error detection and correction in the proposed memory system.

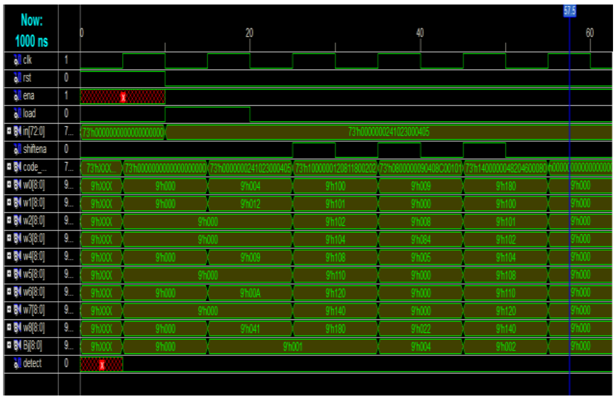


Fig.7(a) Novel MLDD output for Error free Codeword

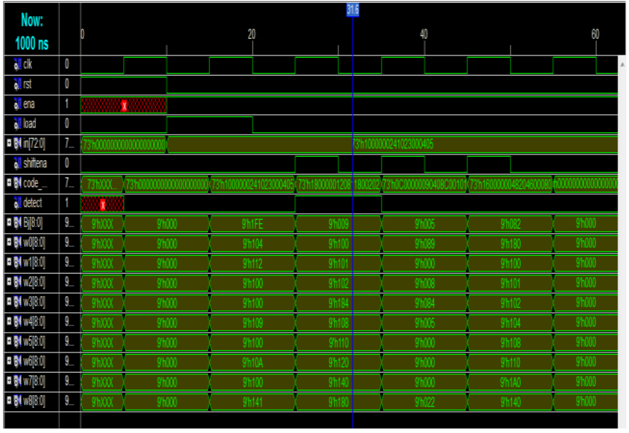


Fig.7(b) Single Error Detection

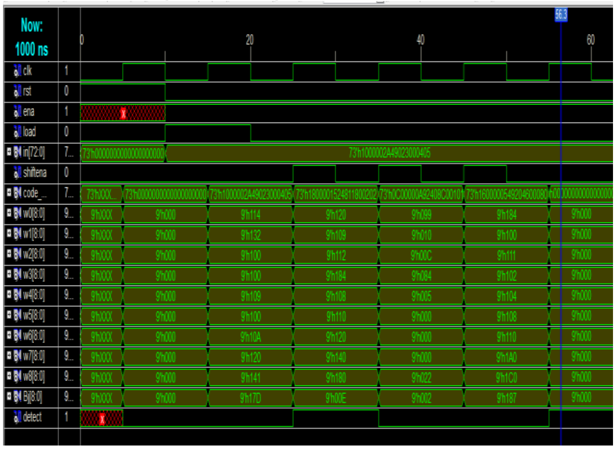


Fig.7(c) Detection of four single bit errors.

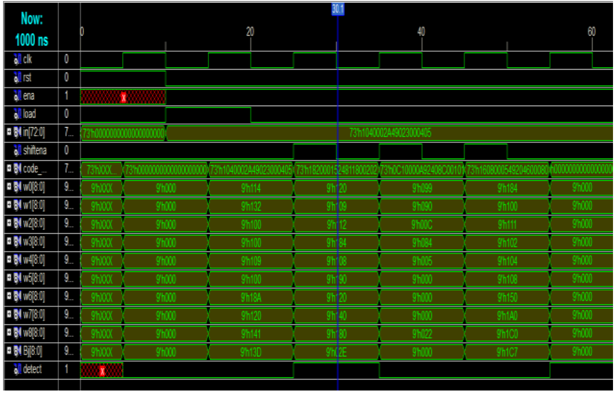


Fig.7 (d) 5-bit error Detection

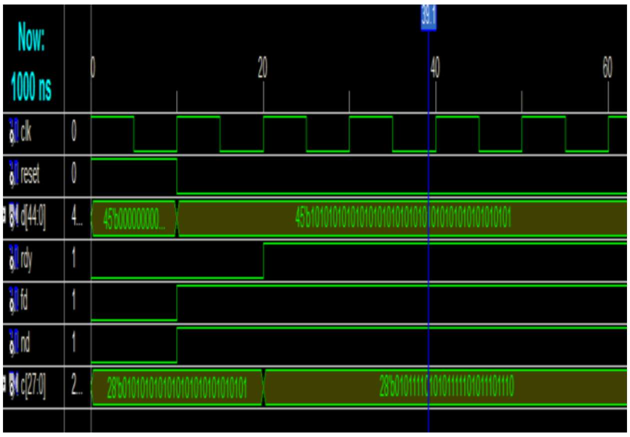


Fig.8. DSCC Encoder Output

V. CONCLUSIONS AND FUTURE SCOPE

In this paper, a Fault Tolerant Memory System has been presented based on ML decoding using the DSCCs. Exhaustive simulation test results show that the proposed technique is able to detect any pattern of up to five bit-flips in the first three cycles of the decoding process.

This improves the performance of the design with respect to the traditional MLDD approach. On the other hand, the modified MLDD error detector module has been designed in a way that is independent of the code size. This makes its area overhead quite reduced compared with other traditional approaches such as the syndrome calculation (SFD).

The application of the proposed technique to memories that use scrubbing is an interesting topic and was in fact the original motivation that led to the MLDD scheme.

APPENDIX

Code word construction using DSCC

The Decoder for MLDD memory system is designed based on Difference Set Cyclic Codes. Difference Set Cyclic Codes is one of the LDPC codes which is the very simplest and most suitable method for correction in memory applications [11].

A. DSCC Parameters:

- Code length : $N = 2^{2s} + 2^{s} + 1$.
- Message bits: $k = 2^{2s} + 2^{s} - 3^{s}$.
- Parity-check bits: $(N - k) = 3^{s} + 1$.
- Minimum distance: $d = 2^{s} + 2$.

For s=3 the DSCC parameters may be calculated as N=73, k=45, (N-k) =28, d=10. Therefore s=3 resulted in a code word of size (N, k) = (73, 45).

The Difference Set Cyclic Codes for (73, 45) the generator polynomial is

$$\boldsymbol{Z(X) = 1 + X^{2} + X^{4} + X^{6} + X^{8} + X^{12} + X^{16} + X^{22} + X^{25} + X^{28}}$$

For a given Difference Set in (73, 45)

Let

$$P(X) = 1 + X^{2} + X^{10} + X^{24} + X^{25} + X^{29} + X^{36} + X^{42} + X^{45}$$

be a Difference Set in (73, 45) DSCC.

$$P = \{0, 2, 10, 24, 25, 29, 36, 42, 45\}$$

The check sum equations are derived using DSCC algebra as

$$w_0(X) = X^{72} + X^{70} + X^{62} + X^{48} + X^{47} + X^{43} + X^{36} + X^{30} + X^{27})$$
$$w_1(X) = X^{72} + X^{64} + X^{50} + X^{49} + X^{45} + X^{38} + X^{32} + X^{29} + X)$$
$$w_2(X) = X^{72} + X^{58} + X^{57} + X^{53} + X^{46} + X^{40} + X^{37} + X^{9} + X^{7})$$
$$w_3(X) = X^{72} + X^{71} + X^{67} + X^{60} + X^{54} + X^{51} + X^{23} + X^{21} + X^{13})$$
$$w_4(X) = X^{72} + X^{68} + X^{61} + X^{55} + X^{52} + X^{24} + X^{22} + X^{14} + 1)$$
$$w_5(X) = X^{72} + X^{65} + X^{59} + X^{56} + X^{28} + X^{26} + X^{18} + X^{4} + X^{3})$$
$$w_6(X) = X^{72} + X^{66} + X^{63} + X^{35} + X^{33} + X^{25} + X^{11} + X^{10} + X^{6})$$
$$w_7(X) = X^{72} + X^{69} + X^{41} + X^{39} + X^{31} + X^{17} + X^{16} + X^{12} + X^{5})$$
$$w_8(X) = X^{72} + X^{44} + X^{42} + X^{34} + X^{20} + X^{19} + X^{15} + X^{8} + X^{2})$$

These parity checksums are orthogonal on X^{72} and are able to correct up to

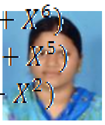
$$t_{ML} = 2^{(s-1)} = 2^{3-1} = 4$$

Therefore (73, 45) DSCC can correct up to 4 errors in the codeword.

REFERENCES

1. C. W. Slayman, “Cache and memory error detection, correction, and reduction techniques for terrestrial servers and workstations,” *IEEE Trans. Device Mater. Reliabil.*, vol. 5, no. 3, pp. 397–404, Sep. 2005.
2. R. C. Baumann, “Radiation-induced soft errors in advanced semiconductor technologies,” *IEEE Trans. Device Mater. Reliabil.*, vol. 5, no.3, pp. 301–316, Sep. 2005.
3. J. von Neumann, “Probabilistic logics and synthesis of reliable organisms from unreliable components,” *Automata Studies*, pp. 43–98, 1956.
4. M. A. Bajura *et al.*, “Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs,” *IEEE Trans. Nucl. Sci.*,vol. 54, no. 4, pp. 935–945, Aug. 2007.
5. R. Naseer and J. Draper, “DEC ECC design to improve memory reliability in sub-100 nm technologies,” in *Proc. IEEE ICECS*, 2008, pp.586–589.
6. S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004.
7. I. S. Reed, “A class of multiple-error-correcting codes and the decoding scheme,” *IRE Trans. Inf. Theory*, vol. IT-4, pp. 38–49, 1954.
8. J. L. Massey, *Threshold Decoding*. Cambridge, MA: MIT Press,1963.
9. S. Ghosh and P. D. Lincoln, “Low-density parity check codes for error correction in nanoscale memory,” SRI Comput. Sci. Lab. Tech. Rep.CSL-0703, 2007.
10. B. Vasic and S. K. Chilappagari, “An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes,” *IEEE Trans. Circuits Syst.I, Reg. Papers*, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.
11. H. Naeimi and A. DeHon, “Fault secure encoder and decoder for NanoMemory applications,” *IEEE Trans. Very Large Scale Integr.(VLSI) Syst.*, vol. 17, no. 4, pp. 473–486, Apr. 2009.
12. Y. Kato and T. Morita, “Error correction circuit using difference-set cyclic code,” in *Proc. ASP-DAC*, 2003, pp. 585–586.
13. T. Kuroda, M. Takada, T. Isobe, and O. Yamada, “Transmission scheme of high-capacity FM multiplex broadcasting system,” *IEEE Trans. Broadcasting*, vol. 42, no. 3, pp. 245–250, Sep. 1996.
14. O. Yamada, “Development of an error-correction method for data packet multiplexed with TV signals,” *IEEE Trans. Commun.*, vol. COM-35, no. 1, pp. 21–31, Jan. 1987.
15. P. Ankolekar, S. Rosner, R. Isaac, and J. Bredow, “Multi-bit error correction methods for latency-contrained flash memory systems,” *IEEE Trans. Device Mater. Reliabil.*, vol. 10, no. 1, pp. 33–39, Mar. 2010.
16. E. J.Weldon, Jr., “Difference-set cyclic codes,” *Bell Syst. Tech. J.*, vol.45, pp. 1045–1055, 1966.
17. C. Tjhai, M. Tomlinson, M. Ambroze, and M. Ahmed, “Cyclotomic idempotent-based binary cyclic codes,” *Electron. Lett.*, vol. 41, no. 6, Mar. 2005.
18. T. Shibuya and K. Sakaniwa, “Construction of cyclic codes suitable for iterative decoding via generating idempotents,” *IEICE Trans. Fundamentals*, vol. E86-A, no. 4, pp. 928–939, 2003.
19. F. J. MacWilliams, “A table of primitive binary idempotents of odd length $n, 7 \le n \le 511$,” *IEEE Trans. Inf. Theory*, vol. IT-25, no. 1, pp. 118–123, Jan. 1979.
20. Actel’s Understanding Soft and Firm Errors in Semiconductor Devices Questions and Answers

AUTHORS PROFILE



Swarnalatha Eluri received B.Ein Electronics and Communications Engineering from Osmania University, Hyderabad, Andhra Pradesh in 2005, Pursuing M.E from Osmania University, Hyderabad. She is currently working Assistant Professor in the Department of ECE, Jawaharlal Nehru Institute of Technology, Hyderabad, Andhra Pradesh.

Hemalatha Rallapalli received B.Tech. in Electronics and Communication Engineering, Sri Krishna Deva Raya University, Anathpur, Andhra Pradesh, in 1992, M.Tech., **Embedded Systems**, Jawaharlal Nehru Technological University, Hyderabad, Andhra Pradesh, and Ph.D.,

Wireless Communication (Cognitive Radio), Jawaharlal Nehru Technological University, Hyderabad, Andhra Pradesh. She is currently working as Assistant Professor in the Department of ECE, University College of Engineering, Osmania University, Hyderabad, Andhrapradesh.