MATRIX CODE BASED MULTIPLE ERROR CORRECTION TECHNIQUE FOR N-BIT MEMORY DATA

Sunita M.S[1] and Kanchana Bhaaskaran V.S[2]

[1]VIT University, Chennai Campus, Chennai, India
PESIT, Bangalore, India
sunitha@pes.edu
[2]VIT University, Chennai Campus, Chennai, India
kanchana.vs@vit.ac.in

ABSTRACT

Constant shrinkage in the device dimensions has resulted in very dense memory cells. The probability of occurrence of multiple bit errors is much higher in very dense memory cells. Conventional Error Correcting Codes (ECC) cannot correct multiple errors in memories even though many of these are capable of detecting multiple errors. This paper presents a novel decoding algorithm to detect and correct multiple errors in memory based on Matrix Codes. The algorithm used is such that it can correct a maximum of eleven errors in a 32-bit data and a maximum of nine errors in a 16-bit data. The proposed method can be used to improve the memory yield in presence of multiple-bit upsets. It can be applied for correcting burst errors wherein, a continuous sequence of data bits are affected when high energetic particles from external radiation strike memory, and cause soft errors. The proposed technique performs better than the previously known technique of error detection and correction using Matrix Codes.

KEYWORDS

Memory testing, Error correction codes, Matrix codes, multiple error detection, multiple error correction.

1. INTRODUCTION

Embedded memories play an important role in the semiconductor market because the system-on-chip market is booming and almost every system chip contains some type of embedded memory. There is a prediction that embedded memories will dominate more than 90% of the system chip area in the next few years. High-density, low-voltage levels, small feature size and small noise margins make the memory chips increasingly susceptible to faults or soft errors [1]. Errors introduced due to the external radiation or electrical noise rather than the design or manufacturing defects are known as soft errors. They are caused by high energy neutrons and alpha particles hitting the silicon bulk resulting in the production of large number of electron-hole pairs. The accumulated charge may be sufficient to flip the value stored in a cell thus causing bit inversion, resulting in soft error [2]. Hence the effects of radiation are bit-flips occurring in the information stored in memory elements. Due to the relentless shrinkage in the device dimensions, the particles that were once considered negligible are now proving to be significant enough to cause upsets [3]. Such errors are identified as soft errors since, although they corrupt the value stored in the cell, they do not permanently damage the hardware.

Soft errors can be either *single-event upset* (SEU), where an ionizing particle affects a single bit or *multiple-bit upset* (MBU), where more than one bit is upset during a single measurement. Burst error can be defined as an error pattern, generally in a binary signal, that consists of known

DOI : 10.5121/vlsic.2013.4103

positions where the digit is in error (*first and last*) with the intervening digits possibly in error and possibly not. By implication, the digits before the first error in the block and after the last error in the block are correct [4]. The burst errors occur during short intervals of time and hence corrupt a set of adjacent bits in that duration. Depending on the underlying technology and the incident particle, several types of multiple-bit errors are possible [5][6]. It has been shown that incident neutron particles can react with the die contaminant and generate secondary particles with enough energy to create multiple errors.

Testing embedded memory is more difficult than testing independent memory, in general, unless built-in self diagnosis techniques are used. Some of the common approaches to protect memories are

1) Built-in Current sensors (BICS) that can detect occurrence of errors by detecting changes in current. The sensors are placed in the columns of the memory blocks and they detect unexpected current variations on each of the memory bit positions [7]. The BICS consumes very little power during testing and no power when testing is finished. Furthermore, it can screen out defects that escape other test methods and are very effective in defect diagnosis. However, it has the drawback that it can only detect errors and it does not have error correction capability.

2) Built-in Self-Test (BIST), which uses various algorithms such as the March pattern, pseudo-random patterns and MATS patterns to test the functionality of the RAMs [8]. They not only detect the presence of faults, but also specify their locations for repair. Although, very effective in functional testing of the RAMs and subsequent error detection, they do not have error correction capability.

3) Built-in Self Repair (BISR)/Built-in Redundancy Analysis (BIRA) is an extension of BIST. It uses the Replacement Algorithm, wherein the cells of a memory identified as defective by process of BIST are corrected by replacing the corresponding row/column by spare rows/columns [9][10]. Though easier to repair the faulty cells, this approach is inefficient, due to the fact that more redundant rows and columns are required to achieve sufficient chip yield.

4) Design-for-test (DFT) techniques are aids to enable detection of defects. A DFT technique involves modifying a memory design to make it testable.

5) Interleaving in the physical arrangement of memory cells, such that the cells belonging to the same logical word are separated. This can prevent MBUs since the physically adjacent bits that are affected by MBU belong to different words. This causes single errors in different words, which can be easily detected and corrected. However, interleaving can have impact on floor-planning, the access time and power consumption [11].

6) Error Correcting Codes: Most common approach to maintain a good level of reliability. ECC techniques are well understood and relatively inexpensive in terms of the extra circuitry required.

The rest of the paper is organized as follows. Section 2 provides a brief survey on the various error-correcting codes used with memory. Section 3 describes the Matrix Codes and the algorithm used for error detection and correction. The proposed architecture is explained in Section 4. The implementation method is explained in Section 5. Section 6 presents the results and discussion. Section 7 concludes the paper. Finally Section 8 provides an insight into the future work which is in progress.

2. SURVEY OF VARIOUS ECC SCHEMES USED FOR MEMORY

There are various Error Correcting Codes used for error detection and correction in memory. Hamming Codes are largely used to correct SEUs in memory due to their ability to correct single errors with reduced area and performance overhead [12]. Though excellent for correction of single errors in a data word, they cannot correct double bit errors caused by single event upset. An extension of the basic SEC-DED Hamming Code has been proposed to form a special class of codes known as Hsiao Codes to improve the speed, cost and reliability of the decoding logic [13]. One more class of SEC-DED codes known as Single-error-correcting, Double-error-detecting Single-byte-error-detecting SEC-DED-SBD codes were proposed to detect any number of errors affecting a single byte. These codes are more suitable than the conventional SEC-DED codes for protecting the byte-organized memories [14][15]. Though they operate with lesser overhead and are good for multiple error detection, they cannot correct multiple errors.

There are additional codes such as the single-byte-error-correcting, double-byte-error-detecting (SBC-DBD) codes, double-error-correcting, triple-error-detecting (DEC-TED) codes that can correct multiple errors as discussed in [9]. The Single-error-correcting, Double-error-detecting and Double-adjacent-error-correcting (SEC-DED-DAEC) code provides a low cost ECC methodology to correct adjacent errors as proposed in [11]. The only drawback with this code is the possibility of miscorrection for a small subset of multiple errors.

The Reed-Solomon (RS) code and Bose-Chaudhuri-Hocquenghem (BCH) Codes are capable of detecting and correcting multiple bytes of errors with low overhead. However, they work at the block level and normally are applied to multiple words at a time [11]. Hsiao M.Y et. al. [16] also proposed a new class of multiple error correcting codes called Orthogonal Latin Square Code, which belong to the class of one-step-decodable majority code, and which can be decoded at an exceptionally high speed.

The matrix codes combine the Hamming and parity codes to improve reliability and yield of memory chips even in the presence of high defects and multiple bit upsets [17]. This paper presents a new decoding algorithm to detect and correct multiple errors using Matrix Codes.

3. MATRIX CODES

The n-bit data word is stored in a matrix format such that n = k1 x k2, where k1and k2 represent the number of rows and columns respectively. For each of the k1 rows, check bits are added and another k2 bits are added as vertical parity bits. The technique is explained by considering a data word length of 32 bits. The 32-bit word is stored in a 4x8 matrix with 4 rows and 8 columns i.e., k1 = 4 and k2 = 8 as shown in the Figure 1.

X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	C_0	C_1	C_2	C_3	C_4
X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	C_5	C_6	C_7	C_8	C_9
X_{16}	X_{17}	X_{18}	X_{19}	X_{20}	X_{21}	X_{22}	X_{23}	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
X_{24}	X_{25}	X_{26}	X_{27}	X_{28}	X_{29}	X_{30}	X_{31}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}
P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7					

Figure 1. 32-bit logical organization of MC's

X_0 to X_{31} are the data bits, C_0 to C_{19} are the horizontal check bits, P_0 to P_7 are the vertical parity bits. Hamming codes are applied for each row. Since 5 check bits are required for 8 data bits, these are added at the end of each row.

The check bits are calculated as follows

$C_0 = X_0 \oplus X_1 \oplus X_3 \oplus X_4 \oplus X_6$
$C_1 = X_0 \oplus X_2 \oplus X_3 \oplus X_5 \oplus X_6$
$C_2 = X_1 \oplus X_2 \oplus X_3 \oplus X_7$
$C_3 = X_4 \oplus X_5 \oplus X_6 \oplus X_7$
$C_4 = X_0 \oplus X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus X_6 \oplus X_7$

Accordingly, all the check bits are calculated for all the rows using the formula $C_{new} = C_{j+(cb*r)}$ and $X_{new} = X_{i+(K2*r)}$ where *cb* is the number of check bits per row, *r* is the row number from 0 to 3, j is the corresponding check bit's position in the first row and i is the corresponding data bit's position in the first row.

For the parity row, we use the following formula

$$P_l = X_l \oplus X_{l+8} \oplus X_{l+16} \oplus X_{l+24}$$

where l is the column number from 0 to 7 for eight parity bits.

Algorithm for Error Correction using Matrix Codes

1) Read the saved data bits X as well as the saved check bits C and parity bits P corresponding to the data word.
2) Generate the check bits using saved data bits (C'_0 to C'_{19}).
3) Generate the syndrome bits of check bits by XORing the original check bits (C_0 to C_{19}) with the newly generated check bits (C'_0 to C'_{19}). The syndrome bits are (SC_0 to SC_{19}).
4) Generate the SED (single error detection) and NE (no error) signals for each row by checking if the syndrome check bit SC(r*5+4) =1, where r = 0,1,2,3 (row number). If there is a single error in any row, the corresponding syndrome bit for that row goes high. If none of the syndrome bits are high, then NE signal is generated.
5) Correct the detected single errors as follows

   If $SC_0 * SC_1 * SC_2 = 1$, then X_3 is in error;
   Else if $SC_0 * SC_1 * SC_3 = 1$, then X_6 is in error;
   Else if $SC_0 * SC_1 = 1$, then X_0 is in error;
   Else if $SC_0 * SC_2 = 1$, then X_1 is in error;
   Else if $SC_1 * SC_2 = 1$, then X_2 is in error;
   Else if $SC_1 * SC_3 = 1$, then X_5 is in error;
   Else if $SC_0 * SC_3 = 1$, then X_4 is in error;
   Else if $SC_2 * SC_3 = 1$, then X_7 is in error

   Accordingly all single errors in each of the rows is corrected
6) Next, generate the MED (Multiple error detection) signal for each row from the syndrome check bits as follows

   If (SC(r*5) OR SC(r*5+1) OR SC(r*5+2) OR SC(r*5+3) OR SC(r*5+4)) =1 then MEDr =1;

   where MEDr is the MED signal corresponding to row r.
7) In addition, generate the parity bits (P'_0 to P'_7).

8) Generate the syndrome bits of parity bits by XORing the original parity bits (P_0 to P_{19}) with the newly generated parity bits (P'_0 to P'_7). The syndrome bits are (SP_0 to SP_7).

9) Using the parity syndrome bits, correct the multiple errors in a row as follows

$Xicorr = Xi \oplus (MEDr * SP_l)$

where SP_l is the syndrome of the parity bit corresponding to the bit l.

10) Output the corrected word.

---

4. PROPOSED ARCHITECTURE

In this section, we portray the block schematic employed for implementing the algorithm described in this paper. Figure 2 shows the block diagram of Memory Architecture for Error Detection and Correction. During the m*emory write* operation, the encoder generates the check bits and the parity bits from the data bits. The check bits and the parity bits are stored in the check bit memory while the data is stored in the data memory.

During the *memory read* operation, the check bits and the parity bits are retrieved along with the data bits. New check bits and parity bits are internally generated in the decoder from the data bits. These new check bits are compared with the stored check bits by an Exclusive-OR operation to generate the syndrome bits. To determine whether the data word is corrupted or not, the decoder generates the error signals NE, SED and MED using the syndrome bits. The errors, if any, are corrected and the corrected data is given out of the decoder.

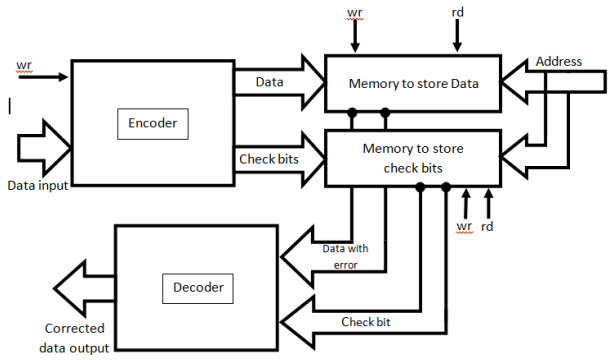

Figure 2. Memory architecture for error detection and correction system

5. IMPLEMENTATION

The method described in the previous section is coded in VHDL. The design was simulated using Xilinx ISim Simulator for both 16-bit and 32-bit data. It was tested for correct functionality by giving various inputs through test benches. The architectures were synthesized on Spartan 6 FPGA which uses 45nm low-power copper process technology. These devices have half the power consumption and are much faster than the previous Spartan families. The LUTs used are dual-register 6-input LUTs. Xilinx Power analyzer tool was used to estimate the power consumption.

6. RESULTS

Figure 3 shows the simulator outputs during a read operation. Here, '*x*' is the 32-bit data whose value is FFFF FFFFH in the waveforms shown. '*er*' is the same data with 11 errors in bit positions 0,14,20,24,25,26,27,28,29,30,31. Thus, there are 3 errors in rows 0, 1, 2 and 8 errors in

row 3. On the positive edge of the *read* signal the data is read, the check bits and parity bits are recalculated. Upon calculation of the syndrome bits, the 'NE', 'SED' and 'MED' signals are determined. The above waveforms show that NE = 0000, SED= 0111 and MED =1000. They imply that there are 3 single errors in the first 3 rows and one multiple error in the last row. Finally, after error correction, the corrected data realizes the value FFFF FFFFH.

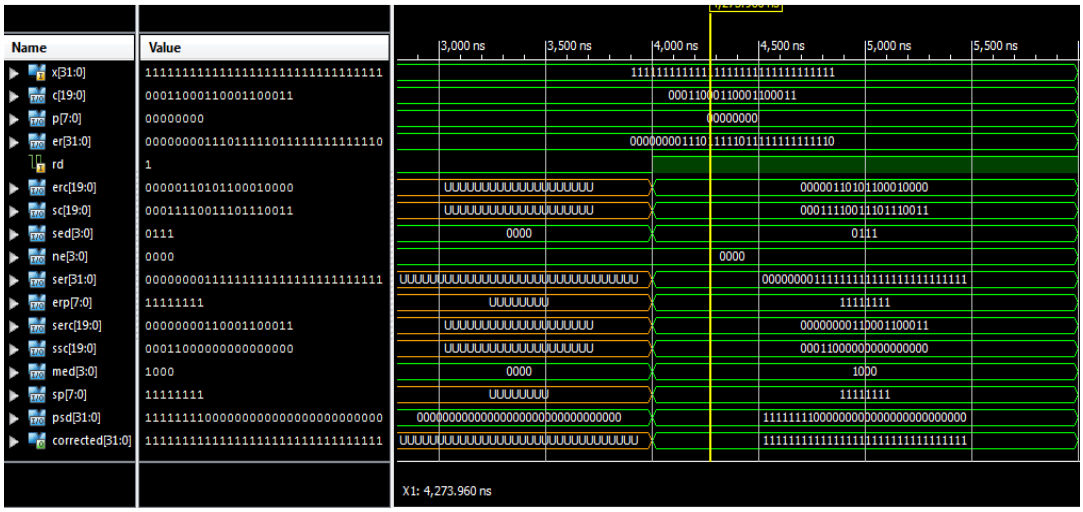

Figure 3. Simulated output for Memory Read and Correct

The same algorithm was applied to a 16-bit data. A comparison is made between the results obtained on a 16-bit data and on a 32-bit data as shown in Table 1.

Parameter	16-bit data	32-bit data
No. of errors corrected	9	11
No. of redundant bits	18	28
Area (No. of 6-input LUT's)	86	192
Power consumed (mW)	54	54

Table 1. Comparison of results

The number of redundant bits per data bit is found to be 1.25 bits for a 16-bit data and 0.875 bits for a 32-bit data. Thus the number of redundant bits decreases as data size increases. Furthermore, it is seen that the major component of power dissipation was that due to leakage and is found to be the same for any data size. The quiescent current was found to be 37mA.

A new performance metric, namely Correction efficiency, is defined to compare the efficiencies of the two codes.

$$\text{Correction efficiency} = \frac{\text{Number of errors corrected}}{\text{Area} \times \text{Number of redundant bits per data bit}} \times 100\%$$

where area is defined as the number of LUT's required.
For 16-bit data the correction efficiency is found to be 9.3% while for 32-bit data it is found to be 6.54%.
Thus it is seen that the efficiency decreases as the number of bits increases.

$$\text{Correction efficiency} \propto \frac{1}{\text{Number of data bits}}$$

The simulation results are shown in the charts in Figure 4

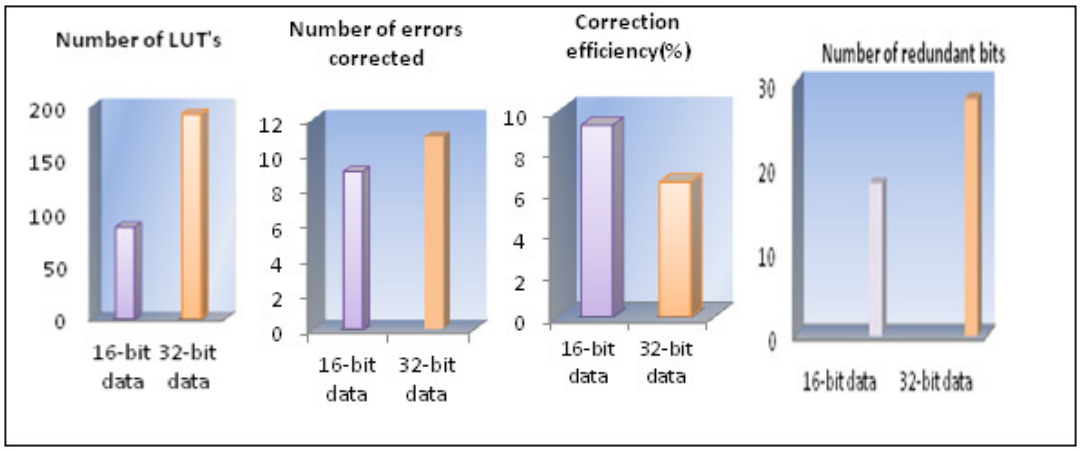

Figure 4. Comparison of results for 16-bit data and 32-bit data

The results obtained with the proposed technique are also compared with the results obtained with the existing technique as given in [17] for a 32-bit data as shown in Table 2.

Parameter	Matrix Codes (Existing)	Matrix Codes (Proposed)
Number of redundant bits	28	28
Number of errors corrected	2	11
Power consumed	207.9mW	54mW

Table 2. Comparison of results with the existing technique

It can be noted that though the overhead (number of redundant bits) remains the same, there is a considerable increase in the number of errors corrected and a significant decrease in the power consumed. Thus the correction efficiency of the proposed technique is better than the existing technique using Matrix codes.

7. CONCLUSION

This paper presented a new technique for multiple error detection and correction in memories, based on the matrix codes. The proposed algorithm can detect and correct multiple errors more efficiently than the earlier known technique. In the method proposed in [17], a maximum of three errors could be corrected wherein two errors occur in a single row and one error in any of the other rows. The technique proposed in this paper can correct up to eight errors in one row and a single error in any of the other rows. The algorithm presented in [17] is simpler wherein both single and double errors are corrected in a single step. The proposed algorithm is more efficient than the one presented in [17]. This is due to the fact that it uses a two-step approach to correct errors. In the first step, all single errors are corrected and in the second step, multiple errors if present, in any row are corrected. It can correct any subset of eight errors in a row as long as the other rows have only a single error. The only drawback is that, when multiple errors do occur in multiple rows, only a few errors are corrected and others remain uncorrected.

8. FUTURE WORK

The work is being extended towards the application of our results for a 1KB memory. The results obtained using the proposed technique for a 1KB memory will be compared with that obtained on application of a few other ECC techniques for a 1KB memory. The comparison would be done based on the area, correction efficiency, performance and power.

REFERENCES

[1] ITRS 2002. [Online]. http://public.itrs.net.

[2] P.Shivkumar, M.Kristler, S.W. Keckler, D. Burger and L.Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic". Proc. of the Int. Conf. on Dependable systems and Networks, pp.389-398, 2002.

[3] P. Hazucha, C. Svenson, "Impact of CMOS technology scaling on the atmospheric neutron soft error rate", IEEE Trans. on Nuclear Science, Vol. 47, no.6, pp. 2586-2594, Dec 2000.

[4] John Daintith. "A Dictionary of Computing", 2004.

[5] Satoh S, Y.Tosaka, S.A.Wender, "Geometric effect of Multiple-bit Soft Errors Induced by Cosmic-ray Neutrons on DRAMs", Proc. of IEEE Int'l Electronic Device Meeting, pp.310-312, Jun 2000.

[6] Makhira.A, et al., "Analysis of Single-Ion Multiple-Bit Upset in High-Density DRAMS", IEEE Trans. On Nuclear Science, Vol. 47, No.6, Dec.2000.

[7] M. Nicolaidis, F.Vargas and B. Coutois, "Design of Built-in current sensors for concurrent checking in radiation environments", IEEE Trans. Nucl.Sci. vol.40, No.6, pp. 1584-1590, Dec. 1993.

[8] R. Dean Adams, "High Performance Memory Testing: Design Principles, Fault Modeling and Self-Test", Kluwer Academic Publishers, USA, 2003.

[9] D.K. Bhavsar " An algorithm for row-column self-repair of RAM's and its implementation in the ALPHA 21264" Proc. Int. Test Conf, pp. 311-318, 1999.

[10] S.K. Lu and S.C Huang, "Built-in self-test and repair (BISTR) Techniques for Embedded RAM's", Proc, Int. Workshop on Memory Technology, Design and Testing", pp. 60-64, Aug 2004.

[11] A.Dutta, N.A. Touba, "Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code" Proc. IEEE VLSI Test Symposium (VTS), 2007, pp. 349-354.

[12] A.D Houghton "The Engineer's Error Coding Handbook" London, UK; Chapman and Hall, 1997.

[13] Hsiao M.Y, "A class of Optimal Minimum Odd-weight-column SEC-DED codes", IBM Journal of Research and Development, Vol. 14, pp. 395-401, 1970.

[14] Reddy S.M., "A class of linear codes for error control in Byte-per-Package Organized Memory Systems" IEEE trans. On Computers, Vol. C-27, pp. 455-458, May 1978.

[15] Chen C.L, "Error Correcting Codes with Byte Error Detection Capability" IEEE Trans. On Computers, Vol. C-32, pp. 615-621, May 1983.

[16] Hsiao M.Y, Bossen D.C, Chien R.T, "Orthogonal Latin Square Codes", IBM Journal of Research and Development, Vol. 14, pp. 390-394, 1970.

[17] Argryides et al., "Matrix Codes for Reliable and Cost Efficient Memory Chips", IEEE Trans. on VLSI Systems, Vol. 19, pp. 420-428, March 2011.

Authors

Sunita M.S obtained her undergraduate degree from Bangalore University, received her M.Sc (physics) degree from Bangalore University and is currently pursuing her M.S (by Research) at VIT University, Chennai Campus, Chennai, India. She has 22 years of teaching experience and is currently working as an Associate Professor in the Department of Electronics and Communication Engineering, P.E.S Institute of Technology, Bangalore, India.

V.S.Kanchana Bhaaskaran obtained her under graduation degree from Institution of Engineers (India), received her M.S. degree in Systems and Information from Birla Institute of Technology and Sciences, Pilani and PhD degree in the field of Low Power Design of VLSI Circuits from VIT University. She is a Fellow of the Institution of Engineers (India) and a Fellow of the Institution of Electronics and Telecommunication Engineers and Member of IEEE and IET. She has 34 years of industry, research and teaching experience through serving the Department of Employment and Training, Government of Tamil Nadu, Indian Institute of Technology Madras, Salem Cooperative Sugar Mills' Polytechnic College, SSN College of Engineering and currently she serves as the Professor and Dean of the School of Electronics Engineering of VIT Chennai, India.