ISSN (online): 2250-141X
Vol. 7 Issue 1, February 2017

DESIGN AND FPGA IMPLEMENTATION OF CA BASED FOUR BYTE ERROR DETECTION AND CORRECTION

M. Kiran Kumar Amrita Sajja G.Pravalika
Anurag Group of Institutions

ABSTRACT

In certain memory systems the most common error is a single error and the next most is double error type. Here we propose a methodology and implementation of CA (cellular automata) based error correcting codes (ECCs) those are widely applied in computer memory systems. These increase reliability and data integrity. In this paper, we have proposed a novel approach for designing four byte error detecting and correcting code, based on CA concept, and implemented in VHDL. The CA already accepted as an attractive structure for VLSI implementation because of its parallelism, modularity, high performance and reliability. In this correspondence, a modular architecture of CA based (32, 28) four byte error correcting encoder and decoder has been proposed. By using cellular automata (CA) based codes,we can perform four byte error detection and four byte error correction either in check byte or in information byte.

*Keywords***:** *cellular automata, Byte error correcting*

I.INTRODUCTION

Coding is used in a digital system to detect and correct errors introduced in the data stream by channel noise. Error correcting codes have been used to enhance system reliability and data integrity. The use of error-correction coding to eliminate the necessity of retransmission of the data is called forward error correction (FEC). The basic concept is to add redundancy to messages at the encoder such that the decoder can successfully recover the messages from the received block possibly corrupted by channel noise. Detection and correction of errors in a byte is more relevant for such cases rather than a bit.

In this work, the four byte error-correcting code is designed using CA technique. In general error detection and correction is nothing but adding some redundancy (i.e., some extra data) to a message, which receivers can use decoder to check consistency of the delivered message, and to recover data determined to be erroneous.

Error-detection and correction schemes can be either systematic or nonsystematic i.e. the transmitter sends the original data, and attaches a fixed number of check bits (or parity data), which are derived from the data bits and original data achieved by some deterministic algorithm at decoder side in systematic scheme and in non-systematic Scheme, the original message is transformed into an encoded message that has at least as many bits as the original message, then the systematic decoder is user to recover the original data from the encoded data.

II. CELLULAR AUTOMATA

CA consists of a number of cells arranged in a regular fashion where the state transitions of the cell depend on the state of its neighbors and each cell consists of a D flip-flop and a combinational logic implementing the next-state function.

One-dimensional (1*D*) CA characterization into four broad categories:

Class 1: CA which evolve to a homogeneous state;

ISSN (online): 2250-141X
Vol. 7 Issue 1, February 2017

Class 2: those which evolve to simple separated periodic structures;
Class 3: which exhibit chaotic or pseudo-random behavior; and
Class 4: which yield complex patterns of localized structures and are capable of universal computation. The next-state function for a three-neighborhood one-dimensional (1*D*) CA a cell can be expressed as follows.

$$q_i\,(t + 1) = f\,[q_{i-1}\,(t),\, q_i\,(t),\, q_{i+1}(t)] \qquad (1)$$

Where $q_i\ (t+1)$ and $q_i\ (t)$ are the output state of the i^{th} cell at the $(t+1)^{th}$ and t^{th} time step respectively and *f* denotes the local transition function realized with a combinational logic, and is known as a “rule” of the CA. The most effective application of Group CA is to generation of pseudo-random patterns report that maximal length Group CA, with all non-zero states lying in a single cycle, produces best quality of pseudo-random patterns. Such a maximal length CA is designed only with two CA rules (neighborhood configuration), 90 and 150, and its characteristic polynomial is primitive.
Rule 90: $q_i\ (t + 1) = q_{i-1}\ (t) \oplus q_{i+1}(t)$

Rule 150: $q_i\ (t + 1) = q_{i-1}\ (t) \oplus q_i\ (t) \oplus q_{i+1}(t)$
An efficient Characterization of 1*D* CA based on matrix algebra, where the global states of the CA are generated by repeated use of a linear operator. An n-cell 1D ACA is characterized by a linear operator $[T]_{n\times n}$ matrix. T is the characteristic matrix of the cellular automata. The i^{th} row of T corresponds to the neighborhood relation of the i^{th} cell, where

$$T\,[i, j] = \begin{cases} 1, & \text{if the next state of the } i^{th} \text{ cell depends on the present state of the } j^{th} \text{ cell.} \\ 0, & \text{otherwise.} \end{cases}$$

If S_t and S_{t+r} represent the state of the CA at t^{th} and $(t+r)^{th}$ time instant respectively then $S_{t+r} = T^r S_t$. If for a CA all states transition graph lie in some cycle, it is called a group CA. An n-cell maximum length CA is characterized by the presence of a cycle of length 2n-1 with all non-zero states. Two important properties of maximum length CA are stated below which are required to explain the proposed scheme.

III.DESIGN PRINCIPLE OF CA-BASED FOUR BYTE ERROR CORRECTING CODE

CA-based byte error correcting code is similar to extended RS code. But compared to RS code, the proposed code is much simpler to design and requires much less hardware. Design of encoder and decoder using CA is explained in this section.

A. Encoder: In four byte error correcting code, encoder generates four check bytes from a block of N-byte information. After that the check bytes are appended with information bytes to form the code word. The generator matrix of 4-byte error correcting code is given below.

$$G = \begin{pmatrix} I & 0 & 0 & 0 & . & . & 0 & I & T & T^2 & T^3 \\ 0 & I & 0 & 0 & . & . & . & 0 & I & T^2 & T^4 & T^6 \\ 0 & 0 & I & 0 & . & . & . & 0 & I & T^3 & T^6 & T^9 \\ . & . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0. & 0 & . & . & I & I & T^N & T^{2N} & T^{3N} \end{pmatrix}$$

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Where T is an 8×8 matrix and I is an 8×8 identity matrix. In this work, T is the characteristics matrix of an 8-cell maximum length CA. One such rule vector of the 8-cell maximum length CA is < 150, 150, 90, 150, 90, 150, 90, 150 > and corresponding characteristics matrix T is as above.

The codeword W is defined as $W = DG$, where $D = [D0\ D1 \ldots DN-1]$, $W = [D0\ D1 \ldots DN-1\ C0\ C1\ C2\ C3]$, Di is the i^{th} information byte and $C0$, $C1$, $C2$, $C3$ are four check bytes. The four check bytes are computed using the equations given below.

$C_a = T^a D_{N-1} \oplus T^{a(2)} [D_{N-2}] \oplus \ldots \oplus T^{a(N)} [D_0]$ (2)

Where $0 \leq a \leq 3$

B. Decoder

The function of decoder is to determine the error locations, corresponding error magnitudes and error correction.

The syndrome corresponding to the j^{th} check byte, S_j is defined as

$S_j = C'_j \oplus C''_j$; $0 \leq j \leq 3$. (3)

where C'_j is the j^{th} received check byte and C''_j is the j^{th} check byte recomputed from the received information bytes.

Decoding Logic: Decoding logic is explained below. Table I represents the algorithm. In the table, $S0$, $S1$, $S2$, $S3$ are four syndrome values computed by the syndrome generator. The entry NZ in the table I indicates the non-zero syndrome value and Z indicates zero value.

TABLE I
DECODING LOGIC

$S0$	$S1$	$S2$	$S3$	Error in
Z	Z	Z	Z	No error
Z	NZ	Z	Z	$C1$
Z	Z	NZ	NZ	$C2$ & $C3$
Z	NZ	NZ	NZ	Two or More
NZ	NZ	NZ	NZ	One or Two or More.

Error location identification and magnitude calculation scheme:

For four byte error correcting code three different cases may occur where error corrections are required and the cases are as follows.

Case-I**:** Single information byte error

` Suppose k^{th} information byte is in error and E_k is error magnitude, then syndrome equations for single information byte error are

$S_0 = E_k$; $S_1 = T^i E_k$; $S_2 = T^{2i} E_k$; $S_3 = T^{3i} E_k$ (4)

Where S_0, S_1, S_2, S_3 are the four syndrome bytes and $i+k = N$. From the above four equations we get

$T^i S_0 = S_1$; $T^i S_1 = S_2$; $T^i S_2$ (5)

$E_k = S_0$ Error magnitude (6)

Value of i from the equation (5) gives the error location $k = N-i$ and equation (6) determines the error magnitude.

Case-II: Double information byte errors

Suppose E_k and E_l are the errors in k^{th} and l^{th} information bytes, then the corresponding syndrome equations are

$S_a = T^{(i)a}[E_k] \oplus T^{(j)a} [E_l]$ (7)

Where $0 \leq a \leq 3$, $i + k = N$ and $j + l = N$.

From the equation (7), error locations $k=N-i$, $l=N-j$ are determined using the following:

$T^i[S_2] \oplus S_3 = T^{2j}(T^i[S_0] \oplus S_1)$ (8)

ISSN (online): 2250-141X
Vol. 7 Issue 1, February 2017

$T^{2i}[S_1] \oplus S_3 = T^{j}(T^{2i}[S_0] \oplus S_2)$
(9)

The error magnitudes are determined using the two equations given below.

$E_l = T^{z}(T^{i}[S_0] \oplus S_1)$ (10)

$E_k = S_0 \oplus E_l$

(11) Error location identification scheme is achieved as below and given in table II,

$F_1 = S_3 \oplus T^{i}S_2, \quad F_2 = S_3 \oplus T^{2i}S_1,$

$F_3 = S_1 \oplus T^{i}S_0, \quad F_4 = S_2 \oplus T^{2i}S_0.$

TABLE II
LOGIC FOR ERROR LOCATION IDENTIFICATION

F1	F2	F3	F4	Error in
Z	Z	Z	Z	*Dk*
Z	Z	NZ	NZ	*C0 & Dk.*
Z	NZ	NZ	Z	*C1 & Dk.*
NZ	Z	Z	NZ	*C2 & Dk.*
NZ	NZ	Z	Z	*C3 & Dk.*

Case-III: Four information byte errors

Suppose E_k , E_l ,E_p *and* E_q are the errors in k^{th} , l^{th} ,p^{th} *and* q^{th} information bytes, then the corresponding syndrome equations are

$S_a = T^{(i)a}[E_k] \oplus T^{(j)a}[E_l] \oplus T^{(s)a}[E_p] \oplus T^{(t)a}[E_q]$
(7)

Where $0 \leq a \leq 3$, $i + k = N$, $j + l = N$, s + p = N and t +q = N.

Knowing the error locations and error magnitudes, errors are corrected using XOR operation. Suppose D'_k, E_k are the received k_{th} information byte and the calculated k_{th} error byte respectively then the correct information byte can be obtained as

$D_k = D'_k \oplus E_k;$ *where* $0 \leq k < N$
(15)

IV.RESULTS AND CONCLUSION

The Project presents a design and implementation of four byte error correcting code using CA. The CA based encoding and decoding scheme for correcting and detecting four byte errors of such a code is suitable for VLSI design outlook and attractive for its uncomplicatedness and uniformity. The basic cell of a four byte Error Location and four byte Error Correction code has been designed, that are simulated and synthesized by Xilinx Vivado tools.

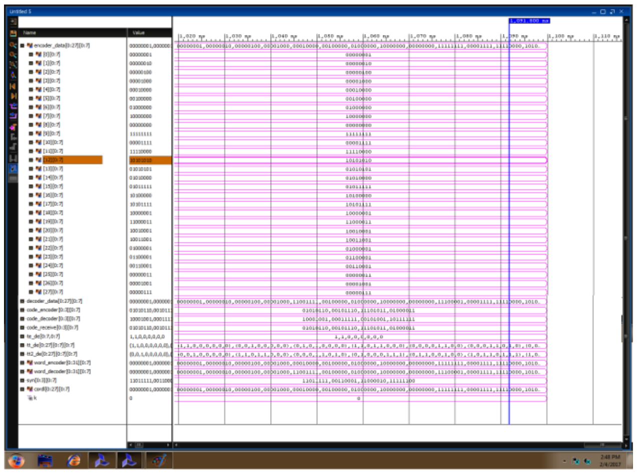

Fig1: Encoder data and syndrome values for four byte error detection and correction

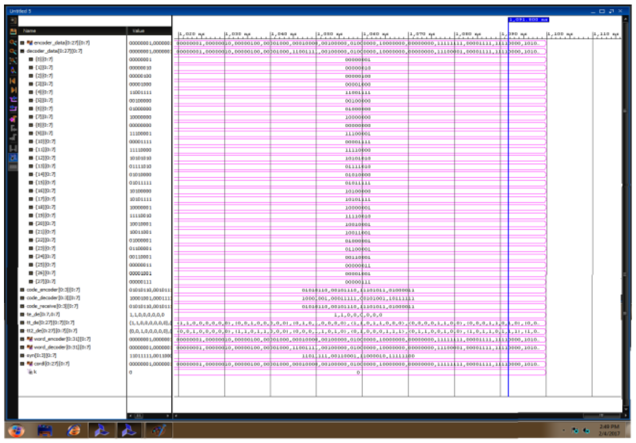

Fig2: Decoder data for four byte error detection and correction

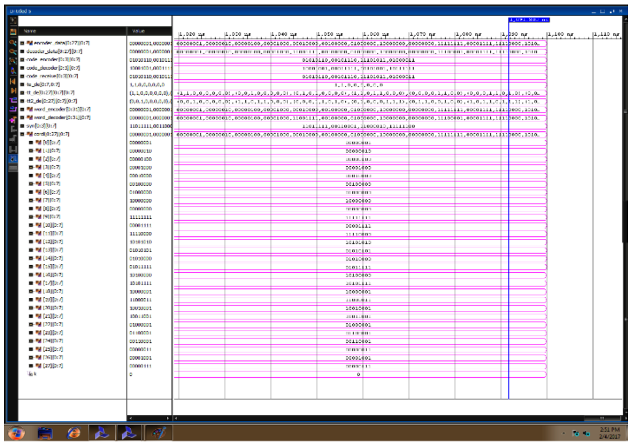

Fig4: Corrected data for four byte error detection and correction

V. REFERENCES

I. B.Anburaj,A. Muthukrishnan, " Fault Tolerant Secured System using Novel Cellular Automata," International Conference on Computing, Electronics and Electrical Technologies [ICCEET],IEEE,2012.

II. D. Roy Chowdhury, I. Sen Gupta, P. P. Chaudhuri, " CA-Based Byte Error-Correcting code," IEEE Transaction on Computers, vol. 44, no. 3,Mar. 1995.

III. P. P. Chaudhuri, D. Roy Chowdhury, S. Nandi and S. Chattopadhyay, Additive Cellular Automata: Theory and Applications, IEEE Computer Society press, California, USA, 1997.

IV. S. Nandi, Ch. Rambabu and P. P. Chaudhuri, "A VLSI Architecture for Cellular Automata Based Reed-Solomon Decoder," 4th Int. Symp.Parallel Architectures, Algorithms and Networks (I-SPAN'99), June 1999.

V. J. Bhaumik, D. Roy Chowdhury and I. Chakrabarti "An Improved Double Byte Error Correcting Code using Cellular Automata," ACRI 2008, LNCS 5191, Sept. 2008.

VI. S. Lin and D. J. Costello, Error Control Coding: Fundamentals and Applications, Prentice-Hall, Engle wood Cliffs, N.J., 1983.

VII. T. R. N. Rao and E. Fujiwara, Error-Control Coding for Computer Systems, Prentice-Hall, Engle wood Cliffs, N.J., 1989.