International Journal on Applications in Engineering and Technology
Volume 3: Issue 1: March 2017, pp 5 -11. www.aetsjournal.com ISSN (Online) : 2455-0523

Generation of BCH Error Correction Code in SRAM Processor

B.Aswini, S.Mohammed Nizar,

Abstract — Error correction codes are used to correct the errors occurred the storing of the data in memory. The idea behinds these codes are to add redundancy bits to the data being stored that even if some errors are occurred in due to noise in the memory, these data can be correctly stored in the memory. Bose, Ray- Chaudhuri, Hocquenghem (BCH) codes are one of the error correcting codes. The BCH decoder consists of the four blocks namely syndrome blocks, IBM block, chin search block and error correction block. This paper described new methods for error detection in syndrome and chien search blocks of BCH decoder. The proposed scheme syndrome block is used to reduce the number of computation by calculating the even number of syndromes from their corresponding odd number of syndromes. The new factorization methods are used to implement the algorithms of chien search blocks of enhanced BCH decoder reduces the number of components required. Thus a new model of BCH decoder is proposed to reduce the area and simplifies the computational schedules of both syndrome and chain search blocks without parallelism leading to high throughput. The enhanced BCH decoder is designed for using hardware description languages called VHDL and synthesized in Xilinx ISE 14.3.

Keywords — : Error correction code (ECC), static random access memory (SRAM), and BCH (Bose–Chaudhuri–Hocquenghem) code.

I. INTRODUCTION

In recent years an increasing demands for digital transmission and storage systems. This demand system has been accelerated by the rapid development and availability of VLSI technology and digital processing. It is frequently the digital system must be fully reliable, as a single error may have shutdown the whole system, or cause unacceptable corruption of data, e.g. in a bank account. In situations such as this error control must be employed so that an error may be detected and afterwards corrected. The simplest way of detecting a single error is a parity checksum, which can be implemented using only exclusive or gates. But in some applications method is insufficient and more sophisticated error control strategy must be implemented.

If a transmission system can transfer the data in both directions, an error control strategy may have determined by detecting an error and then, if an error has occurred, retransmitting the corrupted data. These systems are called as Automatic Repeat request (ARQ). If transmission takes place in only one direction, e.g. information recorded on a Compact

Disk, the only way to accomplish an error control is with Forward Error Correction (FEC). In FEC systems some redundant data is concatenated with the information data in order to allow for the detection and correction of the corrupted data without having to retransmit it. One of the most important classes of FEC codes is linear block codes. In block codes data is transmitted and corrected within one block called as (codeword). That is the data proceeding or following a transmitted codeword does not influence in current codeword. Linear block codes are described by the integer n, the total number of symbols is associated as codeword. Block codes are also described as the number k of information symbols within a codeword, and the number of redundant (check) symbols n-k.

As the embedded size of static random access memories (SRAMs) in a chip increases to aggressive area optimization and power optimization approaches have made them highly vulnerable to manufacturing defects and runtime failures, respectively. The vulnerability of SRAMs is even exacerbated due to the increasing process variations in nano -scale CMOS technologies. Due to process variations, SRAM cells are marginally functional during the manufacturing test can undergo runtime failures, especially under low voltage operation modes Error correction codes (ECCs), such as the single-error-correcting and double error detecting (SEC-DED) codes, are widely used to improve the reliability of on-chip SRAM memories. One of the fundamental problems encountered with the conventional uniform ECC approaches are blocks without considering the differences in importance among the data in a word of embedded memory. However, in

B.Aswini, Applied Electronics , IFET College of Engineering, Villupuram, India. (Email : aswinibala33@gmail.com)
Mr.S.Mohammed Nizar, Department of ECE, IFET College of Engineering, Villupuram, India.

many applications, some parts of the data in words are much more important than other parts. For example, in the embedded memory of digital signal processor (DSP), the failures in the memory bit-cells storing higher order bit (HOB), which are also called as most significant bits, give rise to much larger output quality degradation than those of the lower order bit (LOB).

Considering the differences in importance of memory, modified as ECC ideas have been proposed to strongly protect the HOB data bits in the embedded SRAM memories of the DSPs. Although the tradeoff error correction performance and area overhead is considered as the approaches, the ECC schemes are focused by low latency decoding area overhead due to large parity bits and increasing decoder complexity are still very expensive. This brief presents a novel low complexity and low latency unequal error protection ECC (UEEP-ECC) for the embedded SRAM memories inside the DSPs. An efficient ECC generation algorithms with low area and low latency hardware architecture is also presented as to achieve the minimum power consumption of memory core and ECC encoder/decoder.

II. EXISTING METHOD

The basic concepts for existing UEEP-ECC code decoding procedure is shown in Fig.1. Since the decoding of the repetition code in the HOB part can be simply implemented in using majority voting modules without latency overhead, the repetition code can be easily merged into the BCH decoding process without incurring additional latency. Since repetition code does not incur in additional error even when it fails to correct the errors, BCH decoding can be processed over the decoded data (by repetition decoder) to correct the remaining errors.

One of the fundamental problems is encountered with the conventional uniform ECC approaches is that the ECC protections is equally applied to all memory blocks without considering the differences in importance among them the data in a word of embedded memory. However, in many applications, some parts of the data in a word are much more important than other parts. For example, in the embedded memory of digital signal processor (DSP), the failures in the memory bit-cells storing high-order bits (HOBs), which are also called the most significant bits, give rise to much larger output quality degradation than those of the low-order bits (LOBs). Considering the differences in importance in memory, modified ECC ideas have been proposed to strongly protect the HOB data bits in the embedded SRAM memories of the DSPs.

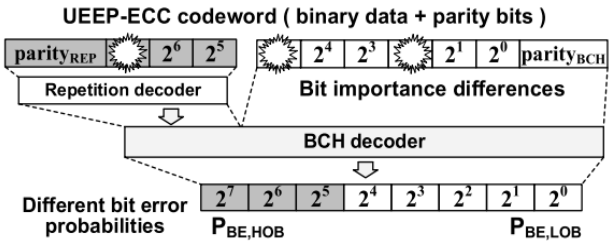

Fig.1. Decoding procedure of the existing UEEP code

Although the considering differences in importance in memory, modified ECC ideas have been proposed to strongly protect the HOB data bits in the embedded SRAM memories of the DSPs. Although the tradeoff between the error correction performance and area overhead is considered in the approaches, since the ECC schemes are focused on low latency decoding, area overhead due to large parity bits and increasing decoder complexity are still very expensive.

This brief presents a novel low-complexity and low-latency unequal error protection ECC (UEEP-ECC) for the embedded SRAM memories inside the DSPs. In proposed ECC scheme, by efficiently merging repetition code over the Bose Chaudhuri Hocquenghem (BCH) code, the UEEP-ECC offers stronger than error corrections on HOB parts without larger area overhead due to parity bits and decoder complexity.

An efficient ECC generation algorithm with low area, latency hardware architecture is also presented to achieve the minimum power consumption of memory core and ECC encoder or decoder.

III. PROPOSED METHOD

In the proposed scheme the error correction code for single error and multiple errors are using the code BCH will be generated. BCH code is a set of ECC that can be used to detect and correct bit errors that can occur when computer data is moved or stored.BCH code is the most widely used in the embedded memory, since it provides low decoding latency with high code rate.BCH can be used to find large number of errors using codes such as the parity bit is added with the data to encode the actual data to avoid the loss of data.

A. BCH ENCODER

The codeword are formed by adding the remainder after division of message polynomial with generator polynomial. All codeword are the multiples of generator polynomial. At the encoding side, the generator polynomials are not usually split as it will demand more hardware and control circuitry.

The polynomial is used as such for encoding. The generator polynomial for BCH (63, 5, t = 2) is given by the parity bits are obtained by computing the remainder polynomial r(x) as: where m(x) is the message polynomial. ri = 0,…., 11 and mi = 0,….,50 are elements of GF(64). M50 is the first message bit to be transmitted and m0 is the last message bit, may be a shortened bit. The parity bits follow the order as follows: r11 first, followed by r10 and so on. BCH codes are implemented as systematic cyclic codes. Hence can be easily implemented and the logic which implements encoder and decoder is controlled into shift register circuits. The remainder r(x) can be calculated in the (n-k) linear stage shift register with the feedback connection at the coefficient of generator polynomial. (63, 51) BCH codeword are encoded as follows: During the 1st clock cycle m50 is given as input to the shift register.

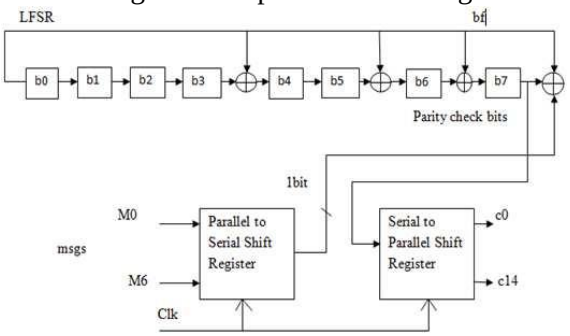

Fig.2. BCH encoder

LFSR is initialized with seed value 0. From 1 to k clock cycles all the 51 messages bits are transmitted and the linear feedback shift register calculates the parity bits. The parity bits generated in the linear feedback shift register are taken in parallel. Thus the 12 parity bits are appended to the message bits.

B. BCH DECODER

Determining where the errors are in a received codeword is a rather complicated process. The decoding process for BCH codes consists of four major steps.

1. Syndrome computation
2. Determine coefficients of error locator polynomial
3. Find the roots of error locator polynomial and it will indicate the erroneous bits in the received codeword.
4. Error correction

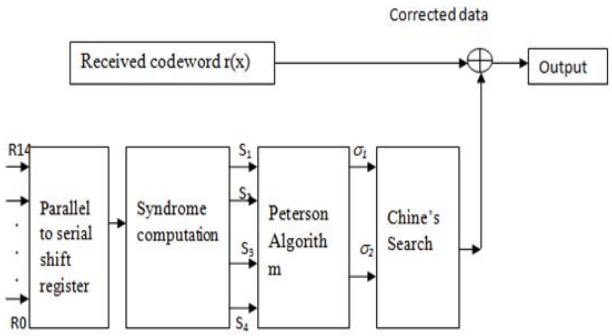

Fig.3. BCH decoder

C. BCH CODING

BCH Codes form a large class of multiple random error-correcting codes. They were first discovered by A. Hocquenghem in 1959 and independently by R. C. Bose and D. K. Ray-Chaudhuri in 1960. BCH codes are cyclic codes. Only the codes, not the decoding algorithms, were discovered by these early writers. The original applications of BCH codes were restricted to binary codes of length $2m - 1$ for some integer m. These were extended later by Gorenstein and Zieler (1961) to the non binary codes with symbols from Galois field GF(q)**.** The first decoding algorithm for binary BCH codes was devised by Peterson in 1960. Since then, Peterson's algorithm has been refined by Berlekamp, Massey, Chien, Forney, and many others.

1. ***Primitive BCH Codes***

For any integer $m \geq 3$ and $t < 2^{m-1}$ there exists a primitive BCH code with the following parameters:

$n = 2^m - 1$

$n - k \leq mt$

$d_{min} \geq 2t + 1$

This code can correct t or fewer random errors over a span of 2^{m-1} bit positions. The code is a t-error-correcting BCH code.

For example, for m=6, t=3

$n = 2^6 - 1 = 63$

$n - k = 6 \times 3 = 18$

$d_{min} = 2 \times 3 + 1 = 7$

This is a triple-error correcting (63,45)BCH code.

2. ***Generator polynomial of binary BCH codes***

Let α be a primitive element in $GF(2^m)$.

For $1 \leq i \leq t$, let $\varphi_{2i-1}(x)$ be the minimum polynomial of the field element α^{2i-1}. The degree of $\varphi_{2i-1}(x)$ is m or a factor of m.

The generator polynomial g(x) of a t-error correcting primitive BCH codes of length $2^m - 1$ is given by

$g(x) = LCM\{\varphi_1(x), \varphi_3(x), \varphi_{2t-1}(x)\}$

Note that the degree of g(x) is mt or less.

Hence the number of parity checks bits; n-k, of the code is at mostmt.

Example: m=4, t=3

Let α be a primitive element in $GF(2^4)$ which is constructed based on the primitive polynomial $p(x) = 1 + x + x^4$

$\varphi_1(x) = 1 + x + x^4$ Corresponding to α

$\varphi_3(x) = 1 + x + x^2 + x^3 + x^4$ Corresponding to $\propto^3$

$\varphi_5(x) = 1 + x + x^2$ Corresponding to $\propto^5$

$g(x) = LCM\{\varphi_1(x), \varphi_3(x), \varphi_5(x)\}$

$= \varphi_1(x)\varphi_3(x)\varphi_5(x)$

$= 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$

The code is a (15,5) cyclic code.

3. ***Properties of binary BCH codes***

Consider a t-error correcting BCH code of length $n = 2^m - 1$ with generator polynomial $g(x)$.

$g(x)$ has as $\propto, \propto^2, \propto^3 \ldots, \propto^{2t}$ roots, i.e.

$g(\alpha \wedge i) = 0 \; for \; 1 \leq i \leq 2t$

Since a code polynomial c(x) is a multiple $g(x)$, $c(x)$ also has $\propto, \propto^2, \ldots, \propto^{2t}$ as roots, i.e. $c(\alpha^i) = 0 \; for \; 1 \leq i \leq 2t$.

A polynomial $c(x)$ of degree less than $2^m - 1$ is a code polynomial if and only if it has $\propto, \propto^2, \ldots, \propto \wedge 2t$ as roots.

4. ***Decoding of BCH Codes***

Consider a BCH code with $n = 2^m - 1$ and generator polynomial $g(x)$.

Suppose a code polynomial $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ is transmitted.

Let $r(x) = r_0 + r_1 x + \cdots + r_{n-1}x^{n-1}$ be the received polynomial.

Then $r(x) = c(x) + e(x)$, where $e(x)$ is the error polynomial.

To check whether $r(x)$ is a code polynomial or not, we simply test whether $r(\propto) = r(\propto^2) = \cdots = r(\propto^{2t}) = 0$.

If yes, then $r(x)$ is a code polynomial, otherwise $r(x)$ is not acode polynomial and the presence of errors is detected.

Decoding procedures are:

- Syndrome Computation.
- Determination of the error pattern.
- Error correction.

5. ***Syndrome computation***

The syndrome consists of 2t components in $GF(2\wedge m)$ $\bar{s} = (s_1 s_2 \ldots s_{2t})$ and $s_i = r(\propto^i) \; for \; 1 \leq i \leq 2t$.

6. **Computation**

Let $\varphi_i(x)$ be the minimum polynomial of $\propto^i$.

Dividing $r(x)$ by $\varphi_i(x)$, we obtain $r(x) = a(x)\varphi_i(x) + b(x)$

Then $s_i = b(\propto^i)$

$s_i = b(\propto^i)$ Can be obtained by linear feedback shift-register with connection based on $\varphi_i(x)$.

7. ***Syndrome and Error Pattern***

Since $r(x) = c(x) + e(x)$ then $s_i = r(\propto^i) = c(\propto^i) + e(\propto^i) = e(\propto^i)$

for $1 \leq i \leq 2t$.

This gives a relationship between the syndrome and the error pattern.

Suppose $e(x)$ has errors v ($v \leq t$) at the locations specified by $x\,j1$, $x\,j2$, , $x\,jv$.

i.e. $x\,j1 + x\,j2 + \ldots + x\,jv$.

where $0 \leq j1 < j2 < \ldots < jv \leq n - 1$.

It we can solve the $2t$ equations, we can determine $\propto^{j1} \propto^{j2}, \propto^{j3} \ldots, \propto^{jv}$.

The unknown parameter $\alpha \; ju = Z$ for 1,2,... ,v are called the "error location number".

When $\propto^{ju}, 1 \leq u < v$ are found, the powers ju, where u = 1,2,…give us the error locations in $e(x)$.The above equations Are known as power-sum symmetric function.

8. ***Error-Location Polynomial***

Suppose that $v \leq t$ errors actually occur.

Define error-locator polynomial $L(z)$ as

$L(z) = (1 + Z_1 z)(1 + Z_2 z) \ldots (1 + Z_v z)$

$= \prod_{i=1}^{v} (1 + Z_i z)$

$= \sigma_0 + \sigma_1 z + \sigma_2 z^2 + \cdots + \sigma_v z^v$

Where $\sigma_0 = 1$.

$L(z)$ has $Z_1^{-1}, Z_2^{-1}, \ldots, Z_v^{-1}$ as roots.

Note that $Z_u = \propto^{ju}$

If we can determine $L(z)$ give us the error location numbers.

9. ***Relationship between S and L(Z)***

From the equation $L(z) = \prod_{i=1}^{v}(1 + Z_i z)$. We find the following relationship between the coefficients of $L(z)$ and the error-locator numbers:

$\sigma_0 = 1$

$\sigma_1 = Z_1 + Z_2 + \cdots + Z_v$

$\sigma_2 = Z_1 Z_2 + Z_2 Z_3 + \cdots + Z_{v-1} Z_v$

$\vdots$

$\sigma_v = Z_1 Z_2 \ldots Z_v$

The above equation is called as "elementary symmetric functions".

The relationship between syndrome and the coefficients of $L(z)$:

$s_1 + \sigma_1 = 0$

$s_2 + \sigma_1 s_1 + 2\sigma_2 = 0$

$s_3 + \sigma_1 s_2 + \sigma_2 s_1 + 3\sigma_3 = 0$

$\vdots$

$s_v + \sigma_1 s_{v-1} + \sigma_2 s_{v-2} + \cdots + \sigma_{v-1} s_1 + v\sigma_v = 0$

Here for binary case $i\sigma_i = \sigma_i$ when I is odd, and $i\sigma_i = 0$ otherwise.

The above equations are called the Newton's identities. If we can determine $\sigma_1, \sigma_2, \dots \sigma_v$, from the Newton's identities, Then we can determine the error-location numbers $Z_1, Z_2, \dots Z_v$, by finding the roots of $L(z)$.

Note that the Newton's identities equation can be expressed also in the following single-equation form:

$s_i + \sigma_1 s_{i-1} + \sigma_2 s_{i-2} + \cdots + \sigma_{i-1} s_1 + i\sigma_i = 0$ for $i = 1,2, \dots, v$

10.

Peterson's Direct-solution (w. w. Peterson, 1960)

Consider the case for $i > v$

First multiply $L(z)$ in the equation $L(z) = \prod_{i=1}^{v}(1 + Z_i z)$ by Z^{-i}, we have

$z^{-i}L(z) = z^{-i} + \sigma_1 z^{1-i} + \sigma_2 z^{2-i} + \cdots + \sigma_{v-1} z^{v-1-i} + \sigma_v z^{v-i}$

Next substituting the roots of $L(z)$ (i.e. $Z_1^{-1}, Z_2^{-1}, \dots, Z_v^{-1}$) into the above equation, produces the following set of equations:

$$Z_1^i + \sigma_1 Z_1^{i-1} + \sigma_2 Z_1^{i-2} + \cdots + \sigma_{v-1} Z_1^{i-v+1} + \sigma_v Z_1^{i-v} = 0$$
$$Z_2^i + \sigma_1 Z_2^{i-1} + \sigma_2 Z_2^{i-2} + \cdots + \sigma_{v-1} Z_2^{i-v+1} + \sigma_v Z_2^{i-v} = 0$$
$$\vdots$$
$$Z_v^i + \sigma_1 Z_v^{i-1} + \sigma_2 Z_v^{i-2} + \cdots + \sigma_{v-1} Z_v^{i-v+1} + \sigma_v Z_v^{i-v} = 0$$

Adding these v equation term by term yield

$$\left(Z_1^i + Z_2^i + \dots + Z_v^i\right) + \sigma_1\left(Z_1^{i-1} + Z_2^{i-1} + \dots + Z_v^{i-1}\right) + \cdots + \sigma_v\left(Z_1^{i-v} + Z_2^{i-v} + \dots + Z_v^{i-v}\right) = 0$$

Now express the above equations in terms of syndrome components, then

$s_i + \sigma_1 s_{i-1} + \cdots + \sigma_{v-1} s_{i-v+1} + \sigma_v s_{i-v} = 0$ for $i > v$

In particular, for $i = v + 1$, we obtain

$s_{v+1} + \sigma_1 s_v + \cdots + \sigma_{v-1} s_2 + \sigma_v s_1 = 0$

Thus, the Newton's identities can be extended to the unknown syndrome s_i for $i > v$.

From the above equations, we can see that the σ_j for $0 \le j \le v$ are closely related to the syndrome components s_i, $1 \le i \le v + 1$.

Thus, σ_j can be determined by solving the set of syndrome equations. Then the error-locations can be found by solving $\alpha^{j_u} = Z_u$ for $1 \le u \le v$ the root of $L(z)$. This $L(z)$ produces an error-pattern $e(x)$ with the minimum number of errors. Hence if $v \le t$ errors occurs, $L(z)$ will give the actual error pattern $e(x)$.

Finally, the error-correcting procedure for the binary BCH codes can be outlined as follows:

1. Compute the syndrome components , $1 \le j \le$, from the received polynomial $r($
2. Set each $\sigma_j =$ for $v + 1 \le j \le$ and solve the first v equations for the , $1 \le j \le$ in terms of .
3. Determine the error-locator polynomial $L($ from these in terms of syndrome components for $0 \le j \le$.
4. Find the error-location numbers $Z_1, Z_2, \dots,$ by solving for the roots of $L($. Use these roots to correct the errors in $r($

11. ***Direct Solutions of Some Simple Cases:***

1. Single-error correction:

$\sigma_1 =$ $L(z) = 1 + s$ $s_1 \neq 0, s_3 =$

2. Double-error correction:

$\sigma_1 =$ $s_1 \neq 0, s_3 \neq$

$\sigma_2 = s_1^{-1}(s_3 + s$

$L(z) = 1 + s_1 z + \left[\frac{s_1^3 + s_3}{s_1}\right]$

3. Triple-error correction:

$\sigma_1 =$ $s_1 \neq$

$\sigma_2 = \frac{s_1^2 s_3 +}{s_1^3 +}$ $s_3 \neq$

$\sigma_3 = (s_1^3 + s_3) + s_1$

Example: $m = 4, t = 3$ BCH code over $GF(2^4)$.

The primitive polynomial for $m = 4$ is $\varphi(x) = 1 + x + x^4$

The minimum polynomials of α, α^3 *and* α^5 are

$\varphi_1(x) = 1 + x + x^4$

$\varphi_3(x) = 1 + x + x^2 + x^3 + x^4$

$\varphi_5(x) = 1 + x + x^2$

$n = 2^4 - 1 = 15$

$g(x) = LCM\{\varphi_1(x), \varphi_3(x), \varphi_5(x)\}$

$= \varphi_1(x)\varphi_3(x)\varphi_5(x)$

$= 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$

The code is a (15,5) cyclic code.

The generator polynomial $g(x)$ has $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6$ as roots. The roots α, α^3 *and* α^4 have the same polynomial,

$\varphi_1(x) = \varphi_2(x) = \varphi_4(x) = 1 + x + x^4$

The root α^3 *and* α^6 have the same minimum polynomial

$\varphi_3(x) = \varphi_6(x) = 1 + x + x^2 + x^3 + x^4$

The minimum polynomial of α^5 is

$\varphi_5(x) = 1 + x + x^2$

Suppose all-zero code word $\bar{c} = (0\ 0\ 0 \dots 0)$ is transmitted and $r(x) = x^2 + x^5 + x^{12}$ is received.

Dividing $r(x)$ by $\varphi_1(x), \varphi_3(x)$ *and* $\varphi_5(x)$ respectively.

We obtain the remainders:

$b_1(x) = 1$

$b_3(x) = 1 + x + x^3$

$b_5(x) = x^2$

The syndrome components are

$s_1 = b_1(\alpha) = 1$

$s_2 = b_1(\alpha^2) = 1$

$s_4 = b_1(\alpha^4) = 1$

$s_3 = b_3(\alpha^3) = 1 + \alpha^6 + \alpha^9 = \alpha^{10}$

$s_6 = b_3(\alpha^6) = 1 + \alpha^{12} + \alpha^{18} = \alpha^5$

$s_5 = b_3(\alpha^5) = \alpha^{10}$
Hence $\bar{s} = (1, 1, \alpha^{10}, 1, \alpha^{10}, \alpha^5)$.

IV. OVERVIEW ERROR CORRECTION CODE

A. Hamming Code

In the year of 1950, Richard Hamming proposed the Hamming code. It is also known as Single Error Detection Codes, which is a popular code and widely used for error control for its variation in system. This code has been considered as the first class of linear codes for error correction. The Hamming code parameters are shown in the table1.

Table.1. Hamming code parameters

Parameters	Equations
Code length, n	n = 2^m – 1
Number of information symbols, k	k =2^m – m – 1
Number of parity-check symbols, m	n - k = m
Error-correcting capability, t	t =[(dmin - 1)/2] = 1

The first hamming code was introduced as (7, 4) code. This represents that there are 4 data bits in a 7-bit packet with 3-bit parity bits, which is shown in Fig. 4. For many applications, a single error correcting code would be considered unsatisfactory, because it accepts all blocks received and only capable to detect and correct a single error. However, it can provide simplicity in coding and has low latency.

Bit position	D7	D6	D5	P4	D3	P2	P1
Data bit	1	0	1	1	0	1	1

Fig. 4.Hamming code (7,4)

B. Single-Error-Correction Double-Error-Detection Code

In the year of 1970, Hsiao has proposed a single error correction and double-error-detection (SEC-DED) code. This code widely used for improving computer reliability. It has been derived from the extended Hamming Code thereby it converts from the Hamming Code by adding an extra parity bit. The SED-DED code was introduced as (8,4) code. It consisting of 4 data bits with 4 extra parity bits is shown in the fig. 5. The parameters for SED-DED code are shown in the table 2.

Table.2. SED-DED parameters

Parameters	Equations
Code length, n	n = 2^m – 1
Number of information symbols	k =2^m – m – 1
Number of parity-check symbols, m	n-k =m
Error-correcting capability	dmin=3

Bit position	P8	D7	D6	D5	P4	D3	P2	P1
Data bit	1	1	0	1	1	0	1	0

Table.2 SED-DED code (8,4)

C. Reed Solomon (RS) Code

In the year of 1960, Irving S. Reed and Gustave Solomon proposed the Reed Solomon (RS) codes. This code is a subclass of BCH codes. The RS code starts with an explanation about the theory of Finite field (FF) or Galois Field (GF). RS code, defined with symbols from GF(q), has the parameters are given in the table 3.

Table.3. Reed-Solomon code parameters

Parameters	Equations
Code length, n	n = q – 1
Number of parity-check symbols, m	n-k = 2m
Error-correcting capability	dmin = 2t+1

D.Low-Density Parity-Check (LDPC) Code

In the year of 1962, Robert G. Gallager proposed the Low Density Parity-Check (LDPC) codes. This LDPC codes are linear error correction code, which is defined by sparse bipartite graphs. Low density parity-check codes are codes specified by a matrix containing mostly 0's and only a small number of 1's. In particular, a (n, j, k) low-density code is a code of block length, n with a matrix, where each column contains a small fixed number, j, of 1 's and each row contains a small fixed number, k, of 1's. Note that this type of matrix does not have the check digits appearing in diagonal form. As for coding purposes, the equations can be represented as in the form of matrices to check on the sums of information digits. These codes are not suitable in minimizing the probability of decoding errors like for a given block length as the codes can only be used below the capacity of the channel when it is at a maximum rate. As to

improve the optimality of codes, decoding scheme need to be simplified for low-density codes. Most LDPC codes have complex encoder. In processor architecture, the connectivity among the decoder component can be large and difficult to be moved.

E. Bose-Chaudhuri-Hocquengham (BCH) Code

In year 1959, Hocquenghem initially discovered the Bose Chaudhuri and Hocquenghem (BCH) codes and subsequently the beginning of the year 1960. For m (m ≥3) and t(t < 2/\(m-l)), the binary BCH code parameters are shown in table 4.

Table.4 BCH code parameters

Parameters	Equations
Code length, n	n =2^m – 1
Number of parity-check symbols, m	n - k ≤ mt
Error-correcting capability	dmin ≥ 2t+1

The biggest advantage of BCH codes is the existence of efficient decoding methods due to the special algebraic structure introduced in the codes. It can detect and correct multiple errors with low latency. However, it is considered as a complicated code because of the complexity of the combinational logic circuits in the error detector.

V. CONCLUSION

In conclusion the generation of BCH codes is rather than very difficult and the BCS system most significantly shortens the time taken to generation of BCH codes. In addition the system has been thoroughly simulated and is less error prone than hand crafted designs. However it should be noted that in some cases a modification of a basic BCH code is required, e.g. shortened or extended BCH codes. Furthermore a different input/output format may be required or an alarm signal needed to be asserted if a non-correctable error pattern has occurred. BCH code can be easily adopted by changing only VHDL files that do not contain any information about finite field operators. Finally, it is worth noting that the final VHDL files are almost behavioural descriptions and that the resultant circuits are as hardware efficient as hand-crafted ones developed for just one set of parameters. Hence there are no hardware penalties incurred by using the BCS system instead of designing a BCH codes on-self, but obviously using the BCS system saves many design man hours.

REFERENCES

[1] David Haughton and Felix Balado “Repetition Coding as an Effective Error Correction Code for Information Encoded in DNA,” IEEE International Conference on Bio-Informatics and Bio-Engineering, 2011.

[2] Meng Zhang, Fei Wu, Changsheng Xie, You Zhou, “A Novel Optimization Algorithm for Chien Search of BCH Codes in Flash Memory Devices”, Huazhong university of science and technology, Wuhan, China.

[3] Kazuteru Namba, Member and Fabrizio Lombardi, Fellow,“A Single and Adjacent Symbol Error Correcting Parallel Decoder for Reed-Solomon codes,” IEEE Transactions on Device and Materials Reliability, 2014.

[4] Carlo Condo Guido Masera, Paolo Montuschi,“Unequal Error protection of Memories in LDPC Decoders,”IEEE Transactions on computers,2013.

[5] Hoyoung Tang and Jongsun Park, “Unequal-Error-Protection Error Correction Codes for Embedded Memories in Digital Signal Processors, IEEE Transactions on Very Large Scale Integration Systems.

[6] Kazuteru Namba, Member and Fabrizio Lombardi,“Parallel Decodable Multi Level Unequal Burst Error Correcting Codes for Memories of Approximate Systems,” IEEE Transactions on computers, 2016

[7] A Avizienis, "Fault-tolerance: The Survival Attribute of Digital Systems", Proc. IEEE, vol. 66, no. 10, pp.l109-1125, Oct 1978.

[8] T. Rao, and E. Fujiwara, "Error Control Coding for Computer Systems", Prentice-Hall Inc. 1989.

[9] R. Hamming, "Error Detecting and Error Corection Code", The Bell System Techninal Journal, vol. 29, no. 2, April 1950.

[10] J. S. Chitode, "Information Coding Technique", Technical Publications, 2007.

[11] Costello, D., and Shu Lin, "Error control coding", Pearson Higher Education, 2004.

[12] Moon, Todd K. "Error Correction Coding", Mathematical Methods and Algorithms, John Wiley and Son 2005.