EFFICIENT BLOCK CODES FOR ERROR CORRECTION USING LOW DENSITY PARITY CHECK CODES

M. Shyam, G. Sreekanth and V. Balamurugan
Electronics and Communication Engineering, Sathyabama University, Chennai, India
E-Mail: mshyamkumar619@gmail.com

ABSTRACT

This paper presents a novel high-speed BCH (hamming) decoder that corrects single-bit errors in parallel and multiple-bit errors corrects serial manner. The proposed decoder is constructed by a novel design and is suitable for nanoscale memory systems, in which multiple-bit errors occur at a probability comparable to single-bit errors and multiple errors occur at a higher probability. To prevent such soft or transient fault related attacks, we consider fault tolerance as a method of mitigation. Most of the current fault tolerant schemes are only multiple bit error detectable but not multiple bit error correctable. This paper also shows that the area, delay, and power overheads incurred by the proposed scheme are significantly lower than traditional fully parallelized BCH based hamming decoders capable of correcting any multiple bit error. This error detection and correction algorithm is synthesized and simulated by using XILINX ISE.

Keywords: error, block code, correction.

1. INTRODUCTION

Error control codes (also known as error correcting codes, ECC) have been frequently used to improve the dependability of a memory system. However, the depend ability of a memory system still remains a concern due to neutron-induced single event upsets. ECC dealing with multiple-bit errors are becoming more and more important. The bch code is one of the best-known and widely used multiple-bit error correcting codes. Multiple-bit error correction of a bch code needs a low-speed serial decoding.

BCH codes can be decoded faster by parallelizing the serial operations but parallelization incurs in a large hardware overhead, particularly for long information bit length. Moreover, it is well known that the bch code is less efficient for short information bit lengths. there are few multiple-bit error correcting codes that can be decoded in parallel, product codes and some low-density parity check codes, such as orthogonal Latin square codes. They require longer check bits than bch codes. Wang has proposed a decoder that is smaller than Wilkerson's, but it is only suitable for Hierarchical Double-Error Correcting Codes, not for bch codes. Codes have a worse code rate than bch, so Wang's decoder requires a larger memory for the check bits than a larger hardware overhead because the area of the additional memory is significantly larger than the decoder. An adjacent erroris a specific type of multi-bit error that changes (or flips) the contents of several adjacent cells. Adjacent errors are caused by a particle hitting a memory array that releases enough energy to affect the value of multiple adjacent cells. This effect occurs with a higher probability than other multiple-bit errors. In general, the occurrence of adjacent errors is mitigated by utilizing an interleaving scheme. Interleaving requires along check bit length. However, this multiple-bit error correction incurs in a considerable overhead. Research on Adjacent Error Correcting (AEC) codes and in particular Double-Adjacent Error Correcting (DAEC) codes has been pursued in the technical literature. The different numbers of errors that can be corrected by different (*n, k*) configurations of bch codes. Note that the number of double-adjacent errors is $n-1$, which is almost the same as the number of single bit errors, *n*, and significantly less than the number of double-bit errors, given by_*n*_1 so as a first estimate, the hardware penalty incurred for double-adjacent error correction should be comparable to single-bit error correction, and moreover, it should be significantly less than double-error correction it is also well-known that the odd-weight column single-error correcting, double-error detecting can be used code recently, a few codes have been proposed generic multi-bit errors not necessarily double-adjacent errors occur at a lower probability than double-adjacent error by two orders of magnitude but they still cannot be ignored. Some codes can be provided with the capability of detecting multiple-bit errors in addition to double-adjacent errors however, these codes are not capable of correcting multiple-bit errors, so they are of limited use at nano scales. This paper presents a high-speed bch decoder. The proposed decoder resembles Wilkerson's design with high speed single-bit error correction. Wilkerson's decoder corrects single-bit errors in parallel and multiple-bit errors serially. Instead, we propose a decoder that corrects single-bit errors and double-adjacent errors in parallel and corrects other multiple bit errors serially. In the proposed decoding scheme, the error pattern generator for sec is also used for error generation of double-adjacent errors.

2. LITERATURE SURVEY

(H. Naeimi, A. DeHon, 2007) proposed a reliable memory system that can tolerate multiple transient errors in the memory words as well as transient errors in the encoder and decoder (corrector) circuitry. The key novel development is the fault-secure detector (FSD) error-correcting code (ECC) dentition and associated circuitry that can detect errors in the received encoded vector despite experiencing multiple transient faults in its circuitry. The structure of the detector is general enough that it can be used for any ECC that follows our FSD-ECC dentition. We prove that two known classes of Low-

Density Parity-Check Codes have the FSD-ECC property: Euclidean Geometry and Projective Geometry codes. We identify a specific FSD-LDPC code that can tolerate up to 33 errors in each memory word or supporting logic that requires only 30% area overhead for memory blocks of 10 Kbits or larger. Larger codes can achieve even higher reliability and lower area overhead. We quantify the importance of protecting encoder and decoder (corrector) circuitry and illustrate a scenario where the system failure rate (FIT) is dominated by the failure rate of the encoder and decoder.

(S.-F. Liu, P. Reviriego, 2012) proposed a single event upsets (SEUs) altering digital circuits are becoming a bigger concern for memory applications. This paper presents an error-detection method for difference-set cyclic codes with majority logic decoding. Majority logic decodable codes are suitable for memory applications due to their capability to correct a large number of errors. However, they require a large decoding time that impacts memory performance. The proposed fault-detection method significantly reduces memory access time when there is no error in the data read. The technique uses the for Double Error Correction Bose–Chaudhuri Hocquenghem (BCH) codes is presented. The scheme reduces the dynamic power consumption so that it is the same that for error detection in a SEC Hamming code. Majority logic decoder itself to detect failures, which makes the area overhead minimal and keeps the extra power consumption low.

(C. Wilkerson, 2012) proposes the significant impact of variations on refresh time and cache power consumption for large eDRAM caches. We propose Hi-ECC, a technique that incorporates multi- bit error-correcting codes to significantly reduce refresh rate. Multi-bit error-correcting codes usually have a complex decoder design and high storage cost. Hi-ECC avoids the decoder complexity by using strong ECC codes to identify and disable sections of the cache with multi-bit failures, while providing efficient single-bit error correction for the common case. Hi-ECC includes additional optimizations that allow us to amortize the storage cost of the code over large data words, providing the benefit of multi-bit correction at same storage cost as a single-bit error-correcting (SECDED) code (2% overhead). Our proposal achieves a 93% reduction in refresh power vs. a baseline eDRAM cache without error correcting capability, and a 66% reduction in refresh power vs. a system using SECDED codes.

(A. Dutta, 2012) proposed amultiple Bit Upsets (MBUs) have become increasingly more frequent with continued increase in memory density. The existing adjacent error correcting codes suffer from high probability of miscorrection for non-adjacent double errors. Miscorrection of a non- adjacent double error as an adjacent double error can reduce the reliability of the memory incorporating such codes. A constraint driven methodology is proposed here for deriving an error correcting code that can correct all single errors and correct the most likely double bit errors i.e., double adjacent errors in a memory while completely eliminating the miscorrection of the most likely non- adjacent double errors. The check bit overhead is minimal and comparable to SEC-DED and SEC-DAEC codes. The encoding and decoding schemes along with the associated hardware for the proposed code are also presented.

(A. Neale, M. Sachdev, 2013) The reliability concern associated with radiation induced soft errors in embedded memories increases as semi- conductor technology scales deep into the sub-40 nm regime. As the memory bit-cell area is reduced, single event upsets (SEUs) that would have once corrupted only a single bit-cell are now capable of upsetting multiple adjacent memory bit- cells per particle strike. While these error types are beyond the error handling capabilities of the commonly used single error correction double error detection (SEC-DED) error correction codes (ECCs) in embedded memories, the overhead associated with moving to more sophisticated double error correction (DEC) codes is considered to be too costly. To address this, designers have begun leveraging selective bit placement to design SEC- DED codes capable of double adjacent error correction (DAEC) or triple adjacent error detection (TAED). These codes can be implemented for the same check-bit overhead as the conventional SEC-DED codes; however, no codes have been developed that use both DAEC and TAED together. In this work, a new ECC scheme is introduced that provides not only the basic SEC-DED coverage, but both DAEC and scalable adjacent error detection (xAED) with a reduction in miscorrection probability as well. Codes capable of up to 11-bits AED have been developed for both 16- and 32-bit standard memory word sizes, and a (39, 32) SEC-DED-DAEC-TAED code implementation that uses the same number of check-bits as a conventional 32 data-bit SEC- DED code is presented.

(Z. Wang, 2013) As the technology moves into the nano-realm, traditional single-error-correcting, double-error-detecting (SEC-DED) codes are no longer sufficient for protecting memories against transient errors due to the increased multi-bit error rate. The well-known double-error- correcting BCH codes and the classical decoding method for BCH codes based on Berlekamp-Massey algorithm and Chien search cannot be directly adopted to replace SEC-DED codes because of their much larger decoding latency. In this paper, we propose the hierarchical double-error- correcting (HDEC) code. The construction methods and the decoder architecture for the codes are described. The presented error correcting algorithm takes only 1 clock cycle to finish if no error or a single-bit error occurs. When there are multi-bit errors, the decoding latency is O(log2m) clock cycles for codes defined over GF(2m). This is much smaller than the latency for decoding BCH codes using Berlekamp Massey algorithm and Chien search, which is O(k) clock cycles - k is the number of information bits for the code and m ~ O(log2k). Synthesis results show that the proposed (79, 64) HDEC code requires only 80% of the area and consumes < 70% of the power compared to the classical (78, 64) BCH code. For a large bit distortion rate (10-3 ~ 10-2), the average decoding latency for the (79, 64) HDEC code is only 36% ~ 60% of the latency for decoding the (78, 64) BCH code.

3. SYNDROME ERROR PATTERN

It consists of a syndrome generator, an error pattern generator and an error detector. The syndrome generator generates the syndrome $s = (s\text{parity}s1\ s3)$ from a received word v. The error pattern generator generates the error pattern e, and the decoder then outputs $v + e$ as a decoded word.The error detector detects uncorrectable errors, that are neither single-bit nor double-adjacent. Specifically, if $\alpha i$is the input, the output vector $e = (e0,\ e\ (k-1))$ satisfies the following conditions ei= 1 and ej= 0 for i= j. The input value of the generator for SEC is received for error pattern generates thus, the detector outputs a 0 no uncorrectable error is detected. As a further example, consider a double-bit error occurring on the 0th and second bits. The structure of the detector is general enough that it can be used for any ECC that follows our FSD-ECC dentition. We prove that two known classes of Low-Density Parity-Check Codes have the FSD-ECC property: Euclidean Geometry and Projective Geometry codes. We identify a specific FSD-LDPC code that can tolerate up to 33 errors in each memory word or supporting logic that requires only 30% area overhead for memory blocks of 10 Kbits or larger. Larger codes can achieve even higher reliability and lower area overhead. We quantify the importance of protecting encoder and decoder (corrector) circuitry and illustrate a scenario where the system failure rate (FIT) is dominated by the failure rate of the encoder and decoder the syndrome is given by $s = (0(\alpha + \alpha 2)\ (\alpha 3 + \alpha 6))$. As $s$parity = 0, the MUX outputs $s1/\ (1 + \alpha) = \alpha$. The decoder also includes AND–OR gates. The output vector of the generator for SEC $e = (e0,\ e\ (k-1))$ and the output of the AND–OR gates. The error is detected to be uncorrectable and the serial decoder for single error and parallel decoder for multiple error correcting The well-known double-error-correcting BCH codes and the classical decoding method for BCH codes based on Berlekamp-Massey algorithm and Chien search cannot be directly adopted to replace SEC-DED codes because of their much larger decoding latency. In this paper, we propose the hierarchical double-error-correcting (HDEC) code. The construction methods and the decoder architecture for the codes are described. The presented error correcting algorithm takes only 1 clock cycle to finish if no error or a single-bit error occurs. When there are multi-bit errors, the decoding latency is O(log2m) clock cycles for codes defined over GF(2m). This is much smaller than the latency for decoding BCH codes using Berlekamp Massey algorithm and Chien search, which is O(k) clock cycles - k is the number of information bits for the code and m ~ O(log2k). Synthesis results show that the proposed (79, 64) HDEC code requires only 80% of the area and consumes < 70% of the power compared to the classical (78, 64) BCH code.

4. SYNDROME GENERATOR LOGIC

The check bits are generated using an XOR-tree corresponding to the G matrix. The proposed code is systematic. During encoding, the data bits can be directly copied and the check bits are generated using an XOR network corresponding to the *G*-matrix. Generate the syndrome using an XOR network corresponding to the *H*-matrix. If the syndrome is the all zero vector, then no error is detected, otherwise one or more errors occurred. If the syndrome matches any of the *H*-matrix columns then a single error is detected and the error position is the corresponding column position. The corresponding bit should be flipped to correct the error. Else if the syndrome matches any of the *n-1* adjacent double error syndromes, then a double adjacent error is detected and the corresponding bit positions are generated using the error correction logic. Else an uncorrectable error (UE) a double nonadjacent error or more than two errors has occurred.

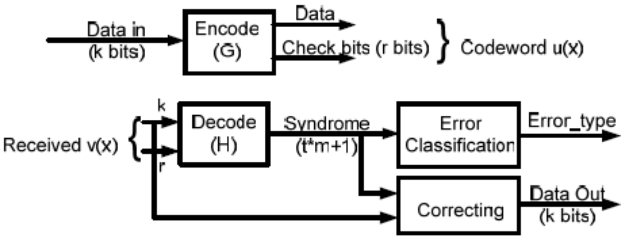
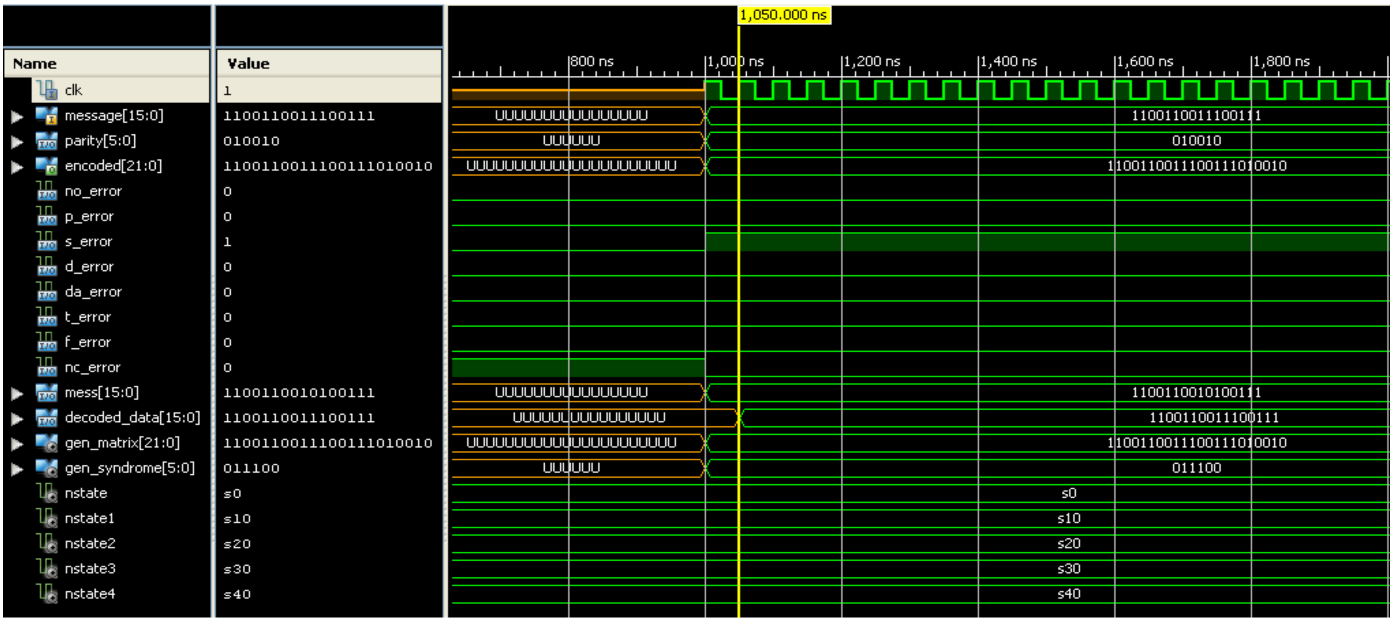


Figure-3. Error correcting scheme.

5. RESULTS AND DISCUSSIONS

The simulation results are shown from Figure 4, 5, 6 and 7 with using xilinx software.

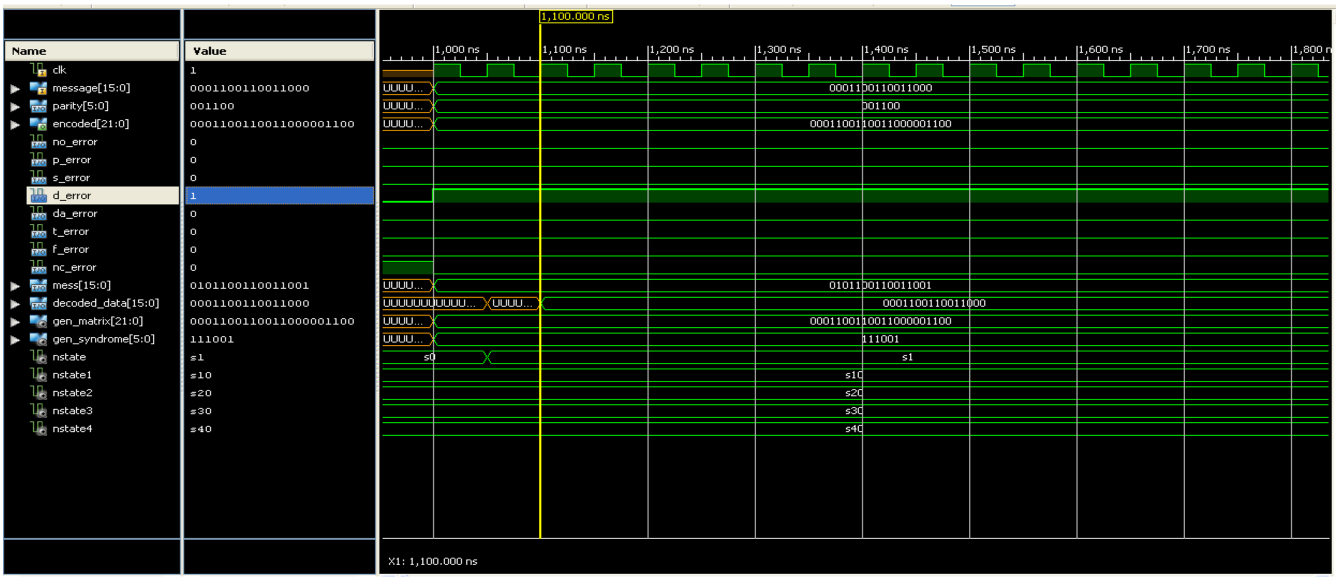


Name	Value
clk	1
message[15:0]	1100110011100111
parity[5:0]	010010
encoded[21:0]	1100110011100111010010
no_error	0
p_error	0
s_error	1
d_error	0
da_error	0
t_error	0
f_error	0
nc_error	0
mess[15:0]	1100110010100111
decoded_data[15:0]	1100110011100111
gen_matrix[21:0]	1100110011100111010010
gen_syndrome[5:0]	011100
nstate	s0
nstate1	s10
nstate2	s20
nstate3	s30
nstate4	s40

Figure-4. Simulation of single bit error.

i.e., Message bit: 1100001100001111; Parity bit: 000101; Coded bit: 1100001100001111 000101 Syndrome bit: 111011; Encoded bit: 1000101000011111; Decoded bit: 1100001100001111

Then compare the message bit and encoded bit, but the 7th position of the encoded data is changed. Instead of ‘1’. Here received ‘0’. but remaining bits are unaffected. So it is a single bit error. In decoded section. That error corrected by single clock cycle itself.



Name	Value
clk	1
message[15:0]	0001100110011000
parity[5:0]	001100
encoded[21:0]	0001100110011000001100
no_error	0
p_error	0
s_error	0
d_error	1
da_error	0
t_error	0
f_error	0
nc_error	0
mess[15:0]	0101100110011001
decoded_data[15:0]	0001100110011000
gen_matrix[21:0]	0001100110011000001100
gen_syndrome[5:0]	111001
nstate	s1
nstate1	s10
nstate2	s20
nstate3	s30
nstate4	s40

Figure-5. Simulation of double bit error.

i.e., Message bit: 1100001100001111; Parity bit: 000101; Coded bit: 1100001100001111 000101 Syndrome bit: 111011; Encoded bit: 1000101000011111; Decoded bit: 1100001100001111

Then compare the message bit and encoded bit, but the LSB bit and 15th position of the encoded data is changed. For both the bits instead of ‘0’. Here received ‘1’ but remaining bits are unaffected. So it is a Doublebit error. In decoded section. That error corrected by using two clock cycles.

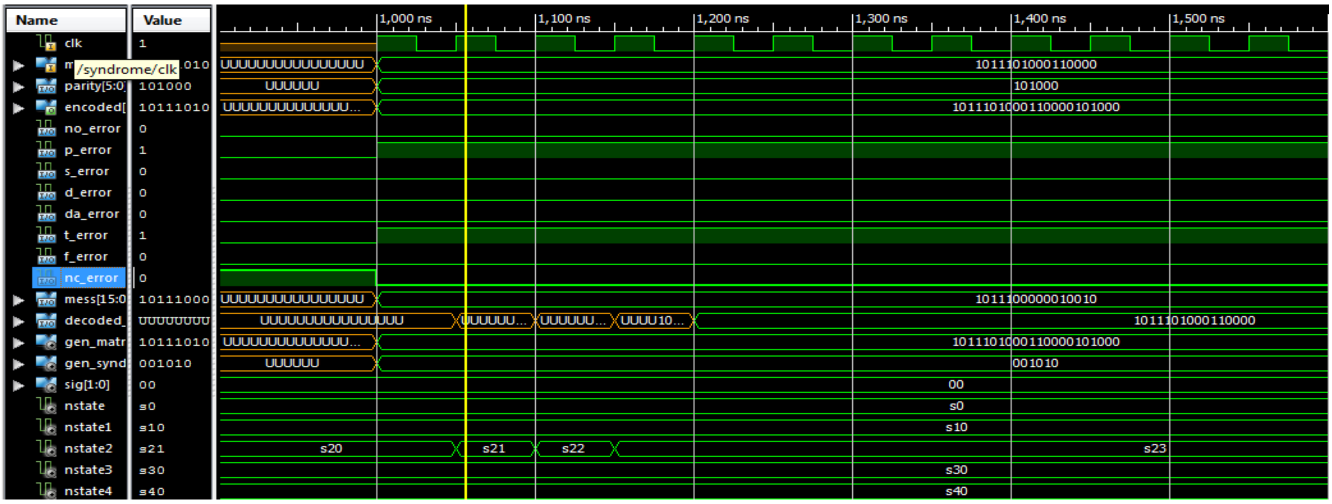


Figure-6. Simulation of triple bit error.

i.e., Message bit: 1100001100001111; Parity bit: 000101; Coded bit: 1100001100001111 000101 Syndrome bit: 111011; Encoded bit: 1000101000011111; Decoded bit: 1100001100001111

Then compare the message bit and encoded bit, but the 2nd bit, 6th and 10th position of the encoded data is changed. For 2nd bit instead of '0' and here received '1', 6th bit instead of '1' and here received '0' and 10th bit instead of '1' and here received '0'.but remaining bits are unaffected. here 3 different positions generate error. So it is a triple error. In decoded section. That error corrected by using four clock cycles.

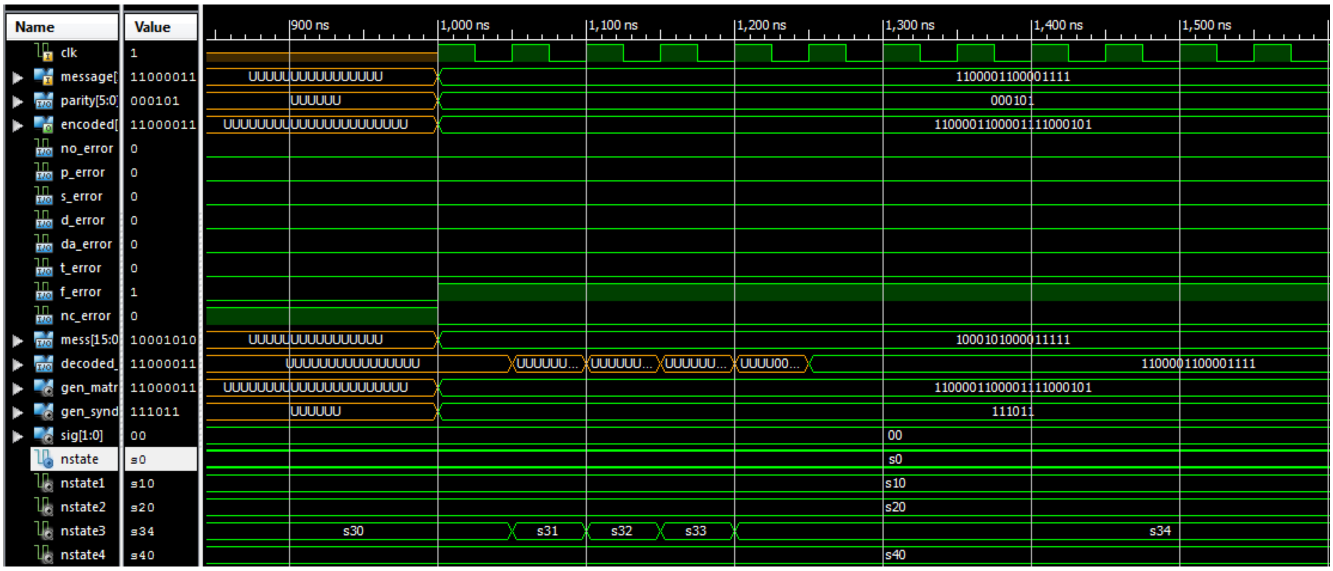


Figure-7. Simulation of four bit error.

i.e., Message bit: 1100001100001111; Parity bit: 000101; Coded bit: 1100001100001111 000101Syndrome bit: 111011;Encoded bit: 1**0**00**1**01**0**000**1**1111; Decoded bit: 1100001100001111

For 5nd bit instead of '0' and here received '1' ,9th bit instead of '1' and here received '0' and 12th bit instead of '0' and here received '1' and 15th bit instead of '1' and here received '0'.but remaining bits are unaffected. Here 4 different positions generate error. So it is a four error. In decoded section. That error corrected by using five clock cycles.

Table-1. Device utilization summary.

Device utilization summary	Existing	Proposed	Available
Number of Slices	26	15	960
Number of 4 input LUTs	49	27	1920
Number of bonded IOBs	50	38	66
Power Consumption	0.044W	0.35W	

6. CONCLUSIONS

This paper has presented a high-speed BCH based hamming decoder for correcting multiple error as well as single-bit errors in serial and parallel manner. When a multiple-bit error occurs including a double-adjacent error. High-speed correction of double-adjacent errors is included in the proposed scheme, because double-adjacent and single-bit errors are much more frequent in a memory system.

The RTL schematic diagrams are show in the figure. Successful implementation of the BCH based Hamming Code Single, Two, Three, Four Bit Error Detection and Correction using the Xilinx ISE.12.4 version and output wave forms are shown in figure above. Check the output waveform is shown in figure. This Hamming Encoder can be implemented for any Memory circuit and in communication devices to prevent Data loss.

The experimental results showed that the proposed scheme has a lower complexity in terms of area compared with the normal BCH based techniques as well as more number of bits are correctly serially in fast manner with less mathematically complexity.

REFERENCES

[1] Dutta. 2012. Low cost adjacent double error correcting code with complete elimination of miscorrection within a dispersion window for multiple bit upset tolerant memory. in Proc. IEEE/IFIP Int. Conf. VLSI-SoC. pp. 287-290.

[2] Neale and M. Sachdev. 2013. A new SEC-DED error correction code subclass for adjacent MBU tolerance in embedded memory. IEEE Trans. Device Mater. Reliab. 13(1): 223-230.

[3] Wilkerson, A. R. Alameldeen, Z. Chishti, W. Wu, D. Somasekhar. 2010. Reducing cache power with low-cost, multi-bit error-correcting codes. In: Proc. Annu. Int. Symp. Comp. Archit. pp. 83-93.

[4] H. Naeimi and A. DeHon. 2007. Fault secure encoder and decoder for memory applications. In: Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. pp. 409-417.

[5] Hosseini E., Perrins E. 2011. FPGA implementation of a coherent SOQPSKTG demodulator, 2011 IEEE Military Communications Conference, MILCOM 2011; Baltimore, MD; United States, pp. 471-476, ISBN: 978-146730081-0.

[6] S.-F. Liu, P. Reviriego, and J. A. Maestro. 2012. Efficient majority logic fault detection with difference-set codes for memory applications. IEEE Trans. Very Large Scale Integr. VLSI Syst. 20(1): 148-156.

[7] T. Fujiwara *et al.* 1989. Error Detecting Capabilities of the Shortened Hamming Codes Adopted for Error Detection in IEEE Standard 802.3. IEEE Trans. Communications. 37(9): 986-989.

[8] W. W. Peterson and E. J. Weldon, Jr., Error-Correcting Codes (2nd Ed.). Cambridge, MA: MIT Press, 1972.

[9] W. Xiong and D. W. Matolak. Performance of Hamming Codes in Systems Employing Different Code Symbol Energies. IEEE Communications Society, pp. 1055-1058 [Wireless and Communications and Networking Conference (WCNC)].

[10] Yang H., Yang Q.-Q., Fang Y., Zhou X., Sobelman G.E. 2012. A novel hardware-oriented decoding algorithm for non-binary LDPC codes, 2012 IEEE Asia Pacific Conference on Circuits and Systems, APCCAS 2012; Kaohsiung; Taiwan, pp. 400-403, ISBN: 978-145771729-1.

[11] Z.Wang. 2013. Hierarchical decoding of double error correcting codes for high speed reliable memories. In: Proc ACM/EDAV/IEEE Des. Autom. Conf. pp. 1-7.