A SINGLE BIT AND MULTIPLE BIT ERROR DETECTING AND CORRECTING PARALLEL DECODER

K.K Geethanjali , *J.Kirthika, *K.Deepika, *K.Gomathi , **Sathyah S.V

(* Student, T.J.S Engineering College ** AP-ECE, T.J.S Engineering College)

ABSTRACT:

In all electronics and electrical systems error detection and error correction is necessary for enabling the delivery of data over unreliable communication channels. Errors may be introduced while transmitting a data from the transmitter to the receiver via channel as most communication channels are subjected to noise. The objective of this paper is to detect and correct multiple bit errors using the parallel decoder for Bose Chaudhuri Hocquenghem (BCH) codes. The errors are detected and corrected efficiently with less mathematical complexity. The BCH code can detect and correct single, double, three, four and adjacent bit errors in parallel manner. Multiple error correcting is achieved in real time networking system. The location of the error bits are also found out. BCH code is better when compared to other coding techniques as they require less check bits. The static and the dynamic power consumed are also estimated. The power consumption is less when compared to the existing system. Error detection and correction is mainly applied in memories.

KEY WORDS: BCH codes, error detection, error correction, check bits, channel.

1. INTRODUCTION With the application of computer science and communication in information and coding theory, the message has to be transmitted to the respective receiver without any disturbances. Hence error detection and error correction control techniques are necessary. Error detection techniques allows the detection of errors, while error correction technique enables reconstruction of the original data in many situations. The most widely used method for error detection and correction of multiple bit error is the BCH codes. BCH codes can be decoded faster by performing the serial operations in parallel manner but parallelization results in a large hardware overhead, particularly for long information length bits. There are few other multiple bit error correcting codes that can be decoded in parallel manner, e.g., Low Density Parity Check (LDPC) codes which includes Orthogonal Latin Square (OLS) codes, Difference Set Cyclic Codes (DSCC) and Euclidean Geometry LDPC (EG-LDPC) Codes. However, they require longer check bits than BCH codes. Theoretical Background: Block codes are generally processed block by block. Convolutional codes are suitable for hardware implementation. Traditional examples of block codes are Hamming codes, repetition codes and paritycheck codes. In the existing system, the single bit errors are corrected in parallel manner and double adjacent bit errors are corrected in serial manner at low speed. The data goes to the syndrome generator from there it goes to the error pattern generator which produces the error pattern output and the error detector which gives the error detection output. For the existing system separate error pattern generator circuits is required for both serial and parallel decoder. The high speed BCH decoding scheme in the existing system decodes words in parallel manner if no error or a single bit error occurs and decodes in serial manner if multiple bit errors occur. As the probability of occurrence of a multiple bit error is greater than a single bit error, this decoding scheme does not represents a good compromise, because it does not achieve high speed operation for cases of error occurrence. The syndrome generator generates the syndrome s=(sparity s1s2) from a received word v. The error pattern is generated by the error pattern generator, and the decoder yields the output v+e. The error detector detects uncorrectable errors which are not single bit or double adjacent bits.

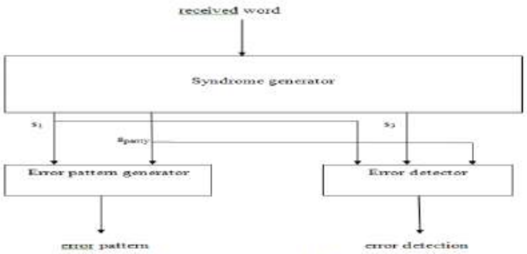


Figure.1. Existing system block diagram

2. METHODS AND MATERIALS OF ERROR CORRECTION AND DETECTION

The proposed scheme is capable of detecting and correcting multiple bit errors in parallel manner. The BCH (hamming) decoders require less checksum bits when compared to other decoding methods. The message bits are exclusively combined with the parity bits and given to the syndrome generator. The message bit is of 16 bits. The parity bit size is 6 bits. Hence the total encoded data is of 22 bits. In the syndrome generator two calculations are done. One is the Parity Checksum calculation and the other is the Syndrome calculation. The syndrome generator and syndrome checker matrix identifies the number of errors bits, the position of the error bit and it simultaneously corrects the error bits. All code word bits are decoded at the same time by having for each code word bit a parity check equation and a correction unit.

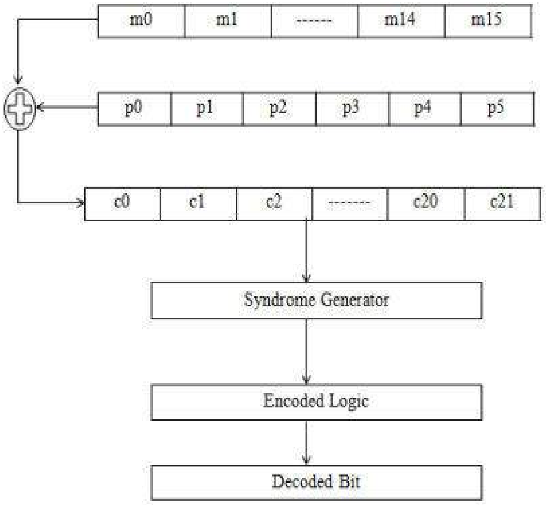
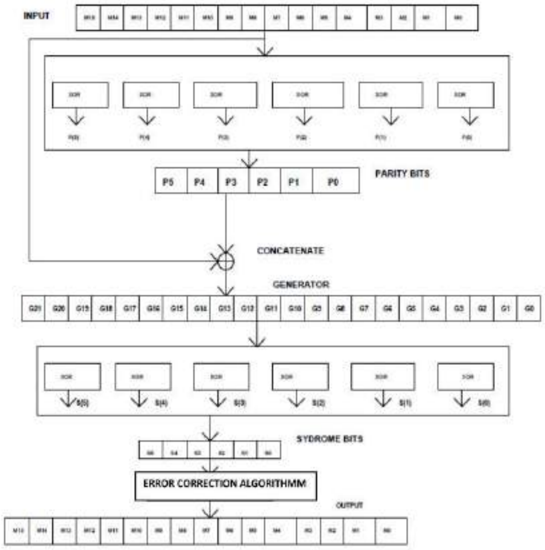


Figure.2. Proposed system block diagram

Here encoder is the encoding of the output to the shift registers. The shift register then shifts each parity bit and are then sent to the parity check equations. If any error is detected it goes to the XOR operations using XOR function for 15 parity checks. This is the process parity check sums calculation. These schemes are used to correct single, double, double adjacent, three and four bit errors.



For parity bits

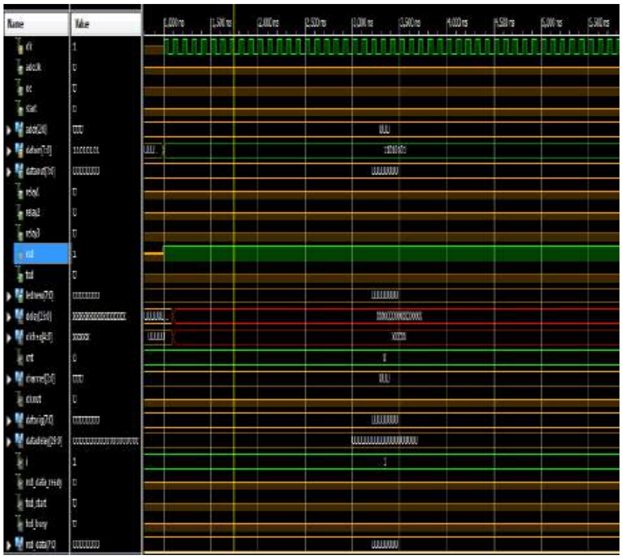
- parity(0) <= message(15) XOR message(13) XOR message(11) XOR message(10) XOR message(8) XOR message(6) XOR message(4) XOR message(3) XOR message(1) XOR message(0);
- parity(1) <= message(13) XOR message(12) XOR message(10) XOR message(9) XOR message(6) XOR message(5) XOR message(3) XOR message(2) XOR message(0);
- parity(2) <= message(15) XOR message(14) XOR message(10) XOR message(9) XOR message(8) XOR message(7) XOR message(3) XOR message(2) XOR message(1);
- parity(3) <= message(10) XOR message(9) XOR message(8) XOR message(7) XOR message(6) XOR message(5) XOR message(4);
- parity(4) <= message(15) XOR message(14) XOR message(13) XOR message(12) XOR message(11)
- parity(5) <= message(15) XOR message(14) XOR message(13) XOR message(12) XOR message(11)
- XOR message(10) XOR message(9) XOR message(8) XOR message(7) XOR message(6) XOR message(5) XOR message(4) XOR message(3) XOR message(2) XOR message(1) XOR message(0);

For syndrome bits

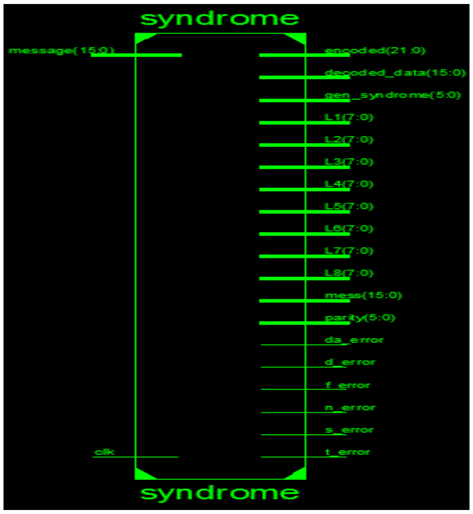
- syndrome(0) <= gen matrix(0) XOR gen matrix(2) XOR gen matrix(4) XOR gen matrix(8);
- syndrome(1) <= gen matrix(10) XOR gen matrix(12) XOR gen matrix(14) XOR gen matrix(16);
- syndrome(2) <= gen matrix(0) XOR gen matrix(4) XOR gen matrix(8) XOR gen matrix(16);
- syndrome(3) <= gen matrix(2) XOR gen matrix(6) XOR gen matrix(10) XOR gen matrix(14);
- syndrome(4) <= gen matrix(2) XOR gen matrix(4) XOR gen matrix(6) XOR gen matrix(8);
- syndrome(5) <= gen matrix(1) XOR gen matrix(3) XOR gen matrix(5) XOR gen matrix(7);

3. RESULT AND DISCUSSION

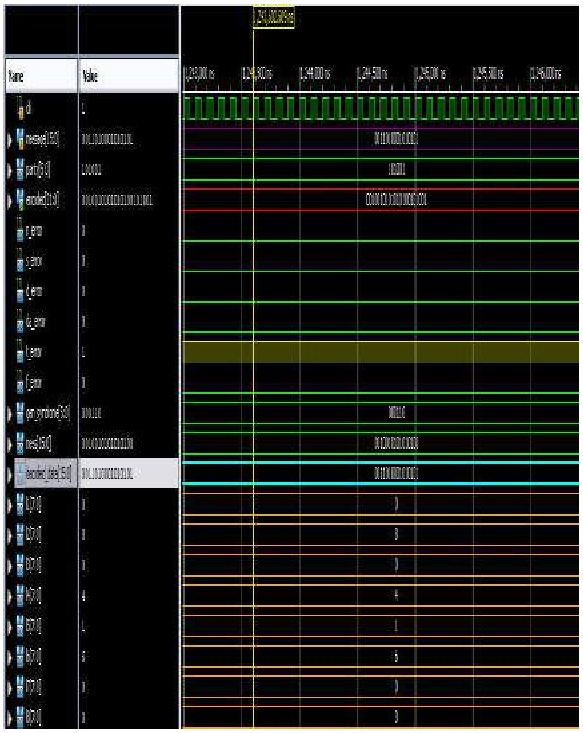
Generation of Syndrome Generator: By giving the message bits as inputs the respective parity bits are generated. By mutually combining the message bits and the parity bits we get the generated matrix as well as the generated syndrome bits. So by applying the message sequence as 1111111111111000 the parity sequence obtained is 111110, generated matrix is 1111111111111000111110.



Error detected and corrected without the location being found: For the message sequence 1111111111111000, single bit error is detected. The error is then corrected and obtained as the decoded data.



Error detected and corrected with the location being found: For the message sequence 1111111111110000 a single bit error is detected and that error is corrected. The corrected error is shown as the decoded data. The location of the error is found out.



4. CONCLUSION

There are various techniques for error correction purpose but the parallel decoder using BCH codes stands out among them all. The checksum bits are reduced by using the BCH codes. Up to four bit errors are detected and corrected using the above technique. In this paper the location of the error is also found out. The power consumption in the parallel decoder is less when compared to other error correction techniques. One of the disadvantages of the parallel decoder is that they add delay. Future work includes the hardware implementation of the above process. The output can be shown using an FPGA board and LEDs.

REFERENCES

1. Liu SF, Reviriego P and Maestro JA, Efficient majority of logic fault detection with difference-set codes for memory applications, IEEE Trans, Very Large Scale Integr, VLSI Syst, 20(1), 2012, 148–156.

2.Maiz J, Hareland S, Zhang K and Armstrong P, Characterization of multibit error events in advanced SRAMs, in IEDM Tech. Dig, 21.4.1–21.4.4, 2003,

3. Pontarelli S and Frost C, A new hardware/software platform and a new 1/E neutron source for error studies, IEEE Trans. Nucl. Sci, 54(4), 2007, 1184–1189.