ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 12, December 2013

Implementation of a Novel Difference Set Codes for Major Fault Detection

Tanuja arja[1] , Narasimha reddy Bathala [2],Madhusudhana Rao.Adigarla[3]

Student (M.Tech), VLSI(SD), Department of ECE, Avanthi Institute of Engineering &Technology[1]

Asst.Professor, Department of ECE, Avanthi Institute of Engineering &Technology[2,3]

Abstract: This Majority Logic decoding (MLD) is very simple to implement and thus it is very practical and has low complexity. The drawback of ML decoding is that, for a coded word of -bits, it takes cycles in the decoding process, posing a big impact on system performance. One way of coping with this problem is to implement parallel encoders and decoders. The solution would enormously increase the complexity and, therefore, the power consumption. method presents a modified version of the ML decoder that improves the designs. The proposed ML detector/decoder (MLDD) has been implemented using the difference-set cyclic codes (DSCCs). This code is part of the LDPC codes and based on their attributes.

Keywords: MLD, MLDD, DSCC, fault detection

I. INTRODUCTION

A single event upset (SEU) is a change of state caused by ions or electro- magnetic radiation striking a sensitive node in a micro-electronic device, such as in a microprocessor, semiconductor memory, or power transistors [2]. The state change is a result of the free charge created by ionization in or close to an important node of a logic element (e.g. memory "bit"). The error in device output or operation caused as a result of the strike is called an SEU or a soft error.

The SEU itself is not considered permanently damaging to the transistor's or circuits' functionality unlike the case of single event latch up (SEL), single event gate rupture (SEGR), or single event burnout (SEB). These are all examples of a general class of radiation effects in electronic devices called single event effects. Terrestrial SEU arise due to cosmic particles colliding with atoms in the atmosphere, creating cascades or showers of neutrons and protons, which in turn may interact with electronics. At deep sub-micro meter geometries, this affects semiconductor devices in the atmosphere.

In space, high energy ionizing particles exist as part of the natural background, referred to as galactic cosmic rays (GCR). Solar particle events and high energy protons trapped in the Earth's magnetosphere (Van Allen radiation belts) exacerbate the problem. The high energies associated with the phenomenon in the space particle environment generally render increased spacecraft shielding useless in terms of eliminating SEU and catastrophic single event phenomena (e.g. destructive latch-up).

II. ERROR CONTROL CODING

The general idea for achieving error detection and correction is to add some redundancy (i.e., some extra data) to a message, which receivers can use to check consistency of the delivered message, and to recover data determined to be corrupted. Error-detection and correction schemes can be either systematic or non-systematic: In a systematic scheme, the transmitter sends the original data, and attaches a fixed number of check bits (or parity data), which are derived from the data bits by some deterministic algorithm [4]. If only error detection is required, a receiver can simply apply the same algorithm to the received data bits and compare its output with the received check bits; if the values do not match, an error has occurred at some point during the transmission. In a system that uses a non-systematic code, the original message is transformed into an encoded message that has at least as many bits as the original message.

Good error control performance requires the scheme to be selected based on the characteristics of the communication channel. Common channel models include memory-less models where errors occur randomly and with a certain probability, and dynamic models where errors occur primarily in bursts. Consequently, error-detecting and correcting codes can be generally distinguished between random-error detecting/correcting and burst-error-detecting/correcting. Some codes can also be suitable for a mixture of random errors and burst errors.

III. ERROR CORRECTION SCHEMES

Error detection is most commonly realized using a suitable hash function (or checksum algorithm). A hash function adds a fixed-length tag to a message, which enables

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 12, December 2013

receivers to verify the delivered message by re computing the tag and comparing it with the one provided .There exist a vast variety of different hash function designs [1]. However, some are of particularly widespread use because of either their simplicity or their suitability for detecting certain kinds of errors (e.g., the cyclic redundancy check's performance in detecting burst errors).

Random-error-correcting codes based on minimum distance coding can provide a suitable alternative to hash functions when a strict guarantee on the minimum number of errors to be detected is desired. Described below, are special cases of error-correcting codes: although rather inefficient, they find applications for both error correction and detection due to their simplicity.

A. Repetition Codes

A repetition code is a coding scheme that repeats the bits across a channel to achieve error-free communication. Given a stream of data to be transmitted, the data is divided into blocks of bits. Each block is transmitted some predetermined number of times. For example, to send the bit pattern "1011", the four-bit block can be repeated three times, thus producing "1011 1011 1011". However, if this twelve-bit pattern was received as "1010 1011 1011" where the first block is unlike the other two – it can be determined that an error has occurred. Repetition codes are very inefficient, and can be susceptible to problems if the error occurs in exactly the same place for each group (e.g., "1010 1010 1010" in the previous example would be detected as correct).

B. Parity Bits

A parity bit is a bit that is added to a group of source bits to ensure that the number of set bits (i.e., bits with value 1) in the outcome is even or odd. It is a very simple scheme that can be used to detect single or any other odd number (i.e., three, five, etc.) of errors in the output [6]. An even number of flipped bits will make the parity bit appear correct even though the data is erroneous. Extensions and variations on the parity bit mechanism are horizontal redundancy checks, vertical redundancy checks, and "double," "dual," or "diagonal" parity

C. Check sums

A checksum of a message is a modular arithmetic sum of message code words of a fixed word length (e.g., byte values). The sum may be negated by means of a ones'-complement operation prior to transmission to detect errors resulting in all-zero messages. Checksum schemes include parity bits, check digits, and longitudinal redundancy checks. Some checksum schemes, such as the Luhn algorithm and the Verhoeff algorithm, are specifically designed to detect errors commonly introduced by humans in writing down or remembering identification numbers

D. Cyclic Redundancy Checks

A cyclic redundancy check (CRC) is a single-burst-error-detecting cyclic code and non-secure hash function designed to detect accidental changes to digital data in computer networks. It is not suitable for detecting maliciously introduced errors. It is characterized by specification of a so-called generator polynomial, which is used as the divisor in a polynomial long division over a finite field, taking the input data as the dividend, and where the remainder becomes the result. Cyclic codes have favorable properties in that they are well suited for detecting burst errors. CRCs are particularly easy to implement in hardware, and are therefore commonly used in digital networks and storage devices such as drives. Even parity is a special case of a cyclic redundancy check, where the single-bit CRC is generated by the divisor x + 1

E. Cryptographic Hash Function

The output of a cryptographic hash function, also known as a message digest, can provide strong assurances about data integrity, whether changes of the data are accidental or maliciously introduced. Any modification to the data will likely be detected through a mismatching hash value. Furthermore, given some hash value, it is infeasible to find some input data (other than the one given) that will yield the same hash value. If an attacker can change not only the message but also the hash value, then a keyed hash or MAC can be used for additional security. Without knowing the key, it is infeasible for the attacker to calculate the correct keyed hash value for a modified message.

F. Error Correcting Codes

Any error-correcting code can be used for error detection. A code with minimum Hamming distance, d, can detect up to d − 1 errors in a codeword. Using minimum-distance-based error-correcting codes for error detection can be suitable if a strict limit on the minimum number of errors to be detected is desired. Codes with minimum Hamming distance d = 2 are degenerate cases of error- correcting codes, and can be used to detect single errors. The parity bit is an example of a single-error-detecting code.

The Berger code is an early example of a unidirectional error(-correcting)code that can detect any number of errors on an asymmetric channel, provided that only transitions of cleared bits to set bits or set bits to cleared bits can occur. A constant-weight code is another kind of unidirectional error- detecting code

IV. ERROR CORRECTION METHOD WITH LEAST DELAY

Some specific type of LDPC codes, namely the difference-set cyclic codes (DSCCs), which is widely, used in the Japanese Teletext system or FM multiplex broadcasting systems. This paper explores the idea of using the ML decoder circuitry as a fault detector so that read operations are accelerated with almost no additional hardware cost. The

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

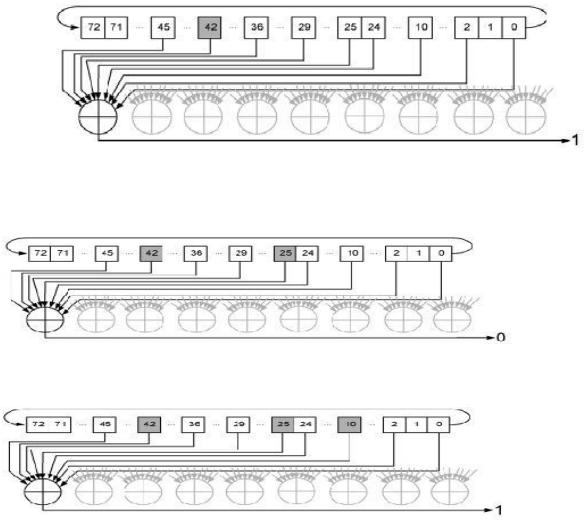
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 12, December 2013

results show that the properties of DSCC-LDPC enable efficient fault detection.

We had proposed the method presents a modified version of the ML decoder that improves the designs presented before. Starting from the original design of the ML decoder introduced in Fig-2, the proposed ML detector/decoder (MLDD) has been implemented using the difference-set cyclic codes (DSCCs). This code is part of the LDPC codes and based on their attributes, they have the following properties

Fig-1 Single, double, triple bit flip

Since performance is important for most applications, we



have chosen an intermediate solution, which provides a good reliability with a small delay penalty for scenarios where up to five bit-flips may be expected. This proposal is one of the main contributions of this paper, and it is based on the following hypothesis: Given a word read from a memory protected with DSCC codes, and affected by up to five bit-flips, all errors can be detected in only three decoding cycles. This is a huge improvement over the simpler case, where decoding cycles are needed to guarantee that errors are detected.

The decoding algorithm is still the same as the one in the plain ML decoder version [7]. The difference is that, instead of decoding all codeword bits by processing the ML decoding during cycles, the proposed method stops intermediately in the third cycle, as illustrated in Fig-2.

If in the first three cycles of the decoding process, the evaluation of the XOR matrix for all {Bj}'is "0," the

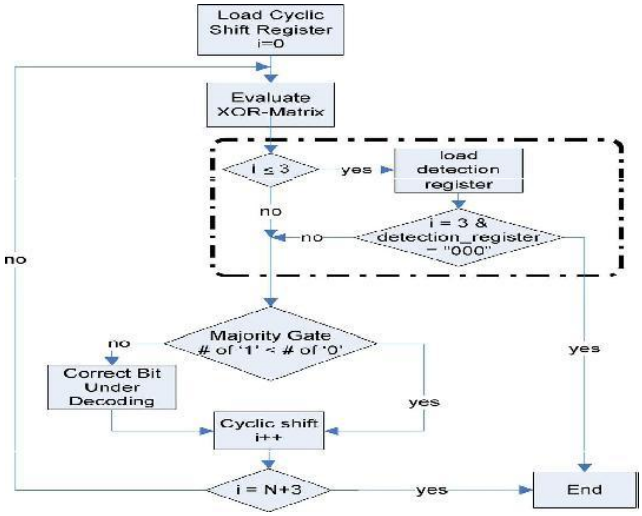


Fig-2 Flow diagram of MLDD

codeword is determined to be error-freeand forwarded directly to the output. If the {Bj} contain in any of the three cycles at least a "1," the proposed method would continue the whole decoding process in order to eliminate the errors [8]. A detailed schematic of the proposed design is shown in Fig-3.

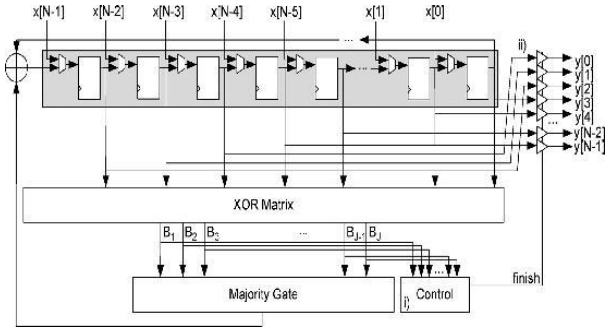


Fig-3 Schematic of Proposed MLDD with Control unit and Tristate Buffers

The figure shows the basic ML decoder with an -tap shift register, an XOR array to calculate the orthogonal parity check sums and a majority gate for deciding if the current bit under decoding needs to be inverted [9]. Those components are the same as the ones for the plain ML decoder in Fig. 2. The additional hardware to perform the error detection is shown in Fig. 4.5 as: i) the control unit which triggers a finish flag when no errors are detected after the third cycle and ii) the output tri state buffers. The output tri state buffers are always in high impedance unless the control unit sends the finish signal so that the current values of the shift register are forwarded to the output.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 12, December 2013

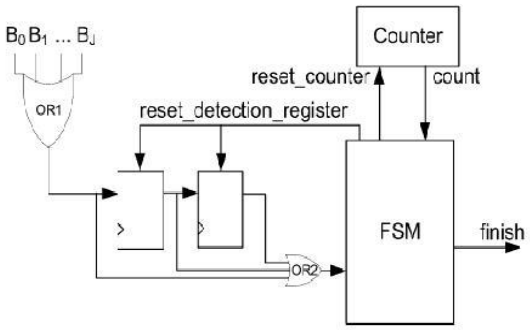


Fig-4 Control unit with finite state machine.

The control unit manages the detection process. It uses a counter that counts up to three, which distinguishes the first three iterations of the ML decoding. In these first three iterations, the control unit evaluates the {Bj} by combining them with the OR1 function. This value is fed into a three-stage shift register, which holds the results of the last three cycles. In the third cycle, the OR2 gate evaluates the content of the detection register. When the result is "0," the FSM sends out the finish signal indicating that the processed word is error- free. In the other case, if the result is "1," the ML decoding process runs until the end.

This clearly provides a performance improvement respect to the traditional method. Most of the words would only take three cycles (five, if we consider the other two for input/output) and only those with errors (which should be a minority) would need to perform the whole decoding process. The schematic for this memory system (see Fig. 5) is very similar to the ML decoder with additional control logic in the MLDD module.

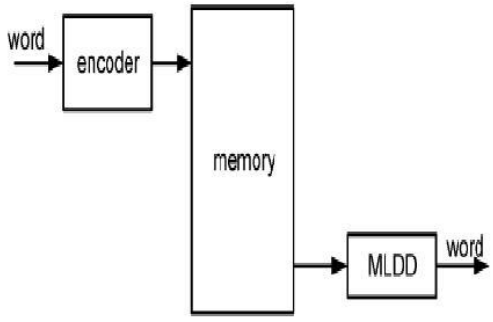


Fig-5 Schematic of MLDD encoding

paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified.

V. RESULTS AND DISCUSSIONS

The input to the ML encoder will be 73 bit code word. By giving the value '0'to the selection signal and '1' to the clock signal, the error will not be generated.

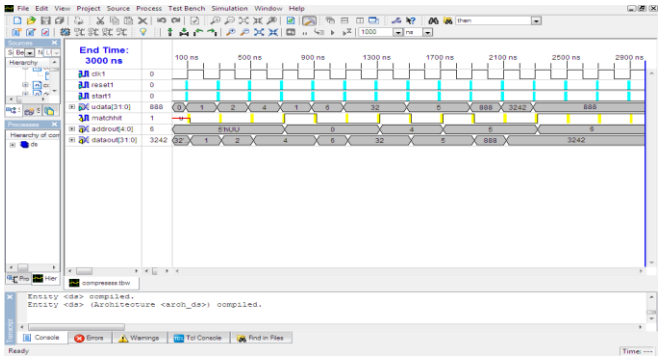


Fig-6 Encoding waveform

The input to the ML Decoder will be 73 bit code word.By giving the value '0'to the selection signal and '1' to the clock signal,the error will not be generated.

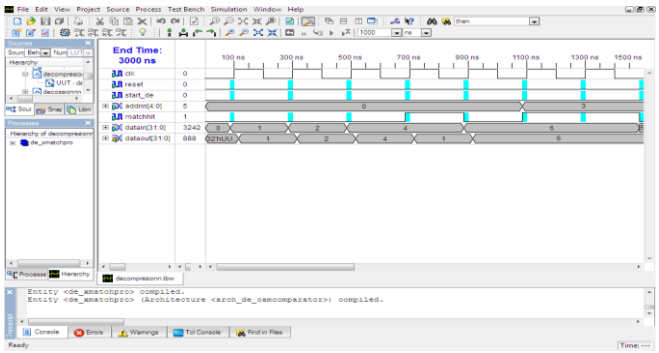


Fig-7 Decoding waveform

VI. CONCLUSION

We conclude that the fault-detection mechanism, MLDD, has been presented based on ML decoding using the DSCCs. Exhaustive simulation test results show that the proposed technique is able to detect any pattern of up to five bit-flips in the first three cycles of the decoding process. This improves the performance of the design with respect to the traditional MLD approach. On the other hand, the MLDD error detector module has been designed in a way that is independent of the code size. The extension of this proof to the case of four errors would confirm the validity of the MLDD approach for a more general case, something that has only been done through simulation . This is, therefore, an interesting problem for future research. The application of the proposed technique to memories that use scrubbing is also an interesting topic and was in fact the original motivation that led to the MLDD scheme.

REFERENCES

[1] P. Ankolekar, S. Rosner, R. Isaac, and J. Bredow, "Multi-bit error correction methods for latency-contrained flash memory

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 12, December 2013

systems," IEEETrans. Device Mater.Reliabil., vol. 10, no. 1, pp. 33–39, Mar. 2010.
[2] M. A. Bajuraet al., "Models and algorithmic limits for an ECC-basedapproach to hardening sub-100-nm SRAMs," IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 935–945, Aug. 2007.
[3] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," IEEE Trans. Device Mater. Reliabil., vol. 5, no. 3, pp. 301–316, Sep. 2005
[4] S. Ghosh and P. D. Lincoln, "Low-density parity check codes for error correction in nanoscale memory," SRI Comput. Sci. Lab. Tech. Rep. CSL- 0703, 2007.
[5] Y. Kato and T. Morita, "Error correction circuit usingdifference-set cyclic code," in Proc. ASP-DAC, 2003, pp. 585–586.
[6] T. Kuroda, M. Takada, T. Isobe, and O. Yamada, "Transmission scheme of high-capacity FM multiplex broadcasting system," IEEETrans. Broadcasting, vol. 42, no. 3, pp. 245–250, Sep. 1996.
[7] S. Lin and D. J. Costello, Error Control Coding, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004
[8] J. L. Massey, Threshold Decoding. Cambridge, MA: MIT Press,1963.
[9] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for NanoMemory applications," IEEE Trans. Very LargeScale Integr. (VLSI) Syst., vol. 17, no. 4, pp. 473–486, Apr. 2009.
[10] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100 nm technologies," in Proc. IEEE ICECS, 2008, pp. 586– 589.