Error Detection and Correction for Binary Arithmetic Golay Code based on CRC Technique

Gautam Deo Verma[1], Prof. Suresh. S. Gawande[2], Prof. Satyarth Tiwari[3]

M. Tech. Scholar, Department of Electronics and Communication, Rkdf college of engineering, Bhabha University, Bhopal[1]

Guide, Department of Electronics and Communication, Rkdf college of engineering, Bhabha University, Bhopal[2]

Co-guide, Department of Electronics and Communication, Rkdf college of engineering, Bhabha University, Bhopal[3]

Abstract— VLSI architecture for fast extended Golay encoder and decoder are presented in this paper. The extended golay code encode and decode of the bit is (24, 12, 8) format. The first bit of the format is represent the transmit Golay encoder bit, second bit of the format is represent the polynomial bit and third bit of the format is represent the hamming distance. Extended Golay codes are detection up to eight bit error and correction up to three bits. The extended Golay code is main block of the cyclic redundancy check (CRC). CRC is error detection code commonly used in wireless communication system. The extended Golay code is implemented Xilinx software with vertex-2p device family. The extended Golay code is implemented in term of number of slice, number of LUT and maximum combinational path delay compared with existing Golay code.

Index Terms- Binary Golay Code (23, 12, 7), Extended Golay Code (24, 12, 8), Adder, Weight Measurement Unit

I. INTRODUCTION

With the rapid growth of digital communications, such as Digital Audio Broadcasting (DAB) and ATM systems, increased data rate and advanced error control coding techniques are required. Consequently, the parallelism innate in the unraveling calculation and the territory productive rapid VLSI designs must be abused. The (24,12,8) expanded Golay code is a notable blunder remedying code, which has been effectively applied in a few existing correspondence frameworks to improve the framework bit-error rate (BER) execution. One goal of this research was to provide a strong error protection for the important head information in the transmission of the high quality compressed music signal of the DAB system. The equal Golay decoder can be, obviously, utilized for the most part to ensure the information transmission or capacity against channel mistakes for rapid information preparing.

Various delicate choice interpreting of the (24, 12) twofold Golay code were seriously examined over the most recent couple of years and point by point examination of computational unpredictability were talked about. Notwithstanding, none of these calculations have been acknowledged proficiently with equal VLSI circuits. This paper presents a full equal change interpreting procedure with look-ahead mistake rectification and a quick delicate choice translating for (24, 12, 8) expanded Golay code. The zone proficient equal VLSI models and the PC reproduction results are additionally exhibited. The look-up table used in this improved algorithm consists of syndrome patterns and corresponding error patterns which have one to three errors occurred in the message block of the codeword. Then the look-up table contains only 25 syndrome patterns and corresponding error patterns. Suppose that there are only three or less errors occurred in (15, 5, 7) BCH codeword. Due to the latter part of H is a 10x10 identity matrix and $S = eH^T$, if the weight of S $w(S)\leq 3$, it means at most three errors only occurred in the parity check block and the location of 1 in S is just the error location in the parity check block. Then shift the syndrome right 5 bits to form a 15-bit length word and minus (modulo 2) the received codeword to decode. If $w(S) \geq 4$, it means at least one error occurred in the message block. First, the syndrome minus (modulo 2) all syndrome patterns in the table to obtain the difference and compute the weight of these difference, respectively.

II. GOLAY CODE

The Binary Golay code is spoken to as (23, 12, 7) that delineates that length of codeword is 23 bits, while message is of 12 bits and the base separation between two parallel Golay codes is 7.

$$A=\begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

A Galois field (GF) is important to build double codes. All in all, parallel field is signified by GF (2), which underpins distinctive twofold number juggling activities. The age of coding arrangement needs a generator polynomial. The conceivable generator polynomials [13] over GF (2) for Golay (23, 12, 7) code are x11 + x10 + x6 + x5 + x4 + x2 + x1 and x11+x9+x7+x6+x5+x1+1. Right now, is considered as the trademark polynomial. The rest of the long division gives the necessary check bits. At last, attaching the produced check bits with the message gives us the all-encompassing Golay codeword. The all-inclusive Golay code (24, 12, 8) can be produced by affixing an equality bit with the twofold Golay code or utilizing a generator grid G, which is characterized as [I, B], where I signifies a personality network of request 12.

III. PROPOSED METHODOLOGY

The primary byte of the ROM code is a cyclic excess check. On the off chance that a Golay code is determined from the accompanying 12 message bits, at that point concurrence with this worth infers that the ROM code is without blunder.

The hypothesis behind this is somewhat troublesome, yet is essentially working with polynomials with parallel numbers as coefficients- - that may be, 1 or 0. The xor work is the main straight capacity of two bits. This move register compares to the polynomial $x^8 + x^5 + x^4 + 1$. It is anything but difficult to copy the move register in programming, in spite of the fact that the PIC can just set, clear or test bits, not move them or do tasks with them.

One of the most powerful redundancy checking procedure based on binary division. In this approach, a sequence of redundant bits is appended to the end of the data unit recognized as CRC with two significant standards.

1. It must be one less than the number of its divisor
2. Data units must be exactly divisible by the divisor after appending these bits.

There are two significant steps of CRC procedure, the CRC generator and the CRC Checker.

CRC Generator

In this procedure, the CRC bits are calculated at the sender side using a generating function in a way that the data unit and the generating function are conducted using bit-wise XOR. The generating function is n+1 bits and CRC are of n bits.

CRC Checker

In this procedure, CRC bits at the receiver side are calculated to check whether they are similar to 0 or not. It applies the bit-wise XOR on the data unit received from the sender and generating function. It is accepted when all the CRC bits are 0 else it is discarded.

Polynomial

A polynomial should meet the following circumstance.

1. It should be not divisible by x
2. It should be divisible by x+1

Number one condition assures detection of the burst errors of length equal to the highest degree of polynomial, while number two condition assures detection of all burst error affecting odd number of bits. The Generating polynomial is important in CRC error detection being a mathematical equation used to locate the generating function bits.

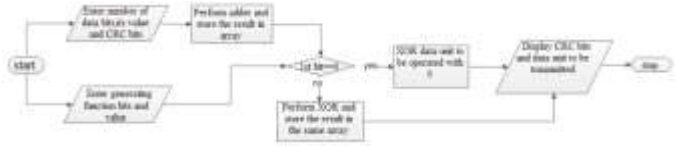

Fig. 1: CRC Generator

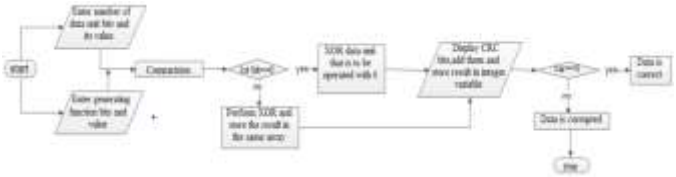

Fig. 2: CRC Checker

Algorithm for CRC checker

1. Input the number of data unit bits (number of data bits + number of CRC bits) received from the sender.
2. Enter generating function bits, its value and data unit bits value.
3. Store generating function bits and data unit (data bits + CRC bits) received from sender in different arrays.
4. Repeat step-5 until each bit of data unit is processed.
5. Check data unit received from sender- if(1st bit==0) XOR the data unit with 0 and store the result in same array of data unit. else XOR the data unit with generating function and store the result in same array of data unit.
6. Display CRC bits, perform adder operation on them and store the result in an integer variable.
7. Check the value of integer variable (var)- if (var ==0) Data unit is correct and accepted after discarding the CRC bits. else Data unit received is corrupted and discarded.

In CRC generator, data bits, number of CRC bits, generating function bits and its value is entered from the user. The data bits and CRC bits are operated under binary addition. The result generated after binary addition is operated under binary division using generating function obtained by generating polynomial. The remainder of the binary division is the actual CRC generated at sender side. The CRC is appended to the data bits and transmitted on the network for the destination.

At the destination end, same generating function is used and received message is operated under binary division using the same procedure to ensure whether the data unit is correct or corrupted.

If the remainder of binary division is all 0s then the data unit is correct and accepted for further processing but if the remainder is non-zero then data has been altered in the mid

way during transmission and it is discarded by the receiver as it is of no use to the receive end.

IV. SIMULATION RESULT

Synthesis is the process of developing a physical system using the abstract descriptions of predefined building blocks such as flip flops, latches and logic gates. It creates a gate-level netlist from a model of a circuit described in VHDL. Finally, the synthesis helps to map VHDL to technologies such as FPGA and ASIC. Most FPGA manufacturers provide free tools to synthesize VHDL to use with their chips.

Synthesis tools mainly focus on the logic design of FPGA and ASIC. They do not consider sensitivity list as they focus on three basic logics: combinational logic, edge sensitive storage (flip flops and some RAM) and level sensitive storage (latches and some RAM).

Moreover, some VHDLs are non-synthesizable. Thus, the programmer can write VHDL code he can simulate but not synthesize. For a design description to be synthesizable, the constructs should be acceptable to the synthesis tool.

CRC Technique

The architecture of CRC Golay code encoder has been implemented in field programmable gate array using Xilinx ISE tool.

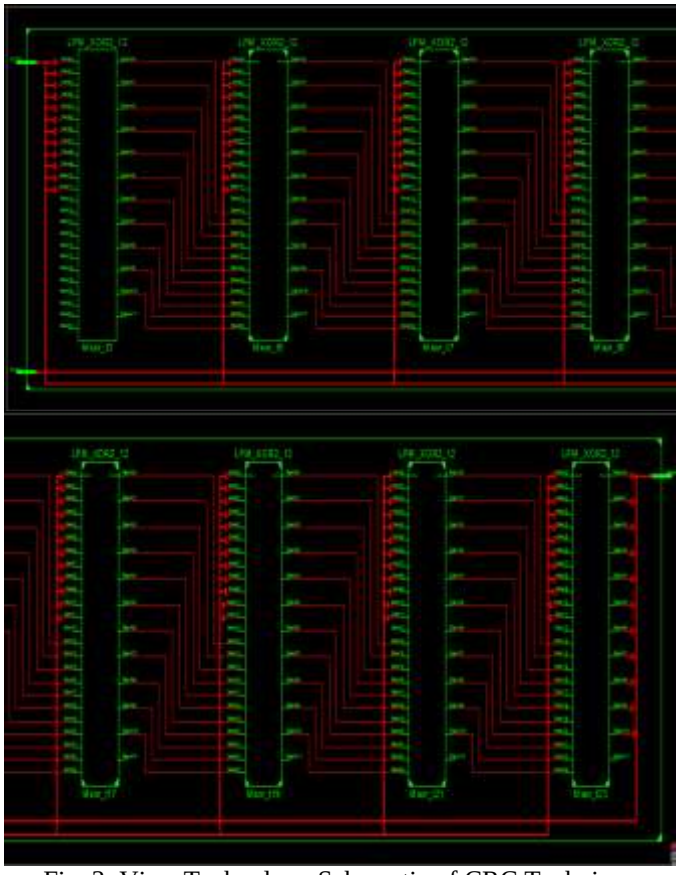

Fig. 3: View Technology Schematic of CRC Technique

```
Device utilization summary:
---------------------------

Selected Device : 3s50pq208-4

 Number of Slices:                        6  out of    768     0%
 Number of 4 input LUTs:                 10  out of   1536     0%
 Number of IOs:                          47
 Number of bonded IOBs:                  46  out of    124    37%
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 15.107ns
```

Fig. 4: synthesize Result by CRC Technique

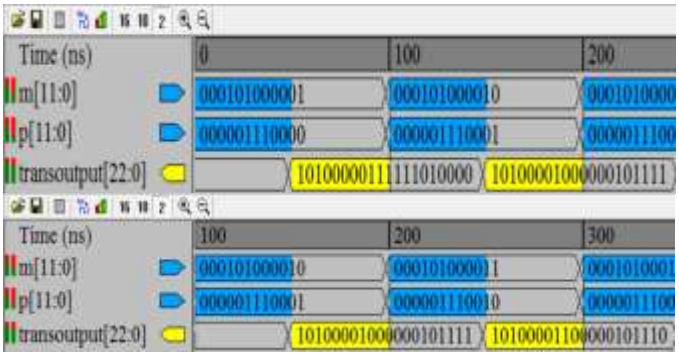

Fig. 5: Output Waveform of CRC Technique

Weight Measurement Unit:

The weight measurement unit primarily counts the number of binary 1 in the sequence, which can be efficiently done by the circuit shown in Fig. 6, which results in less critical path delay.

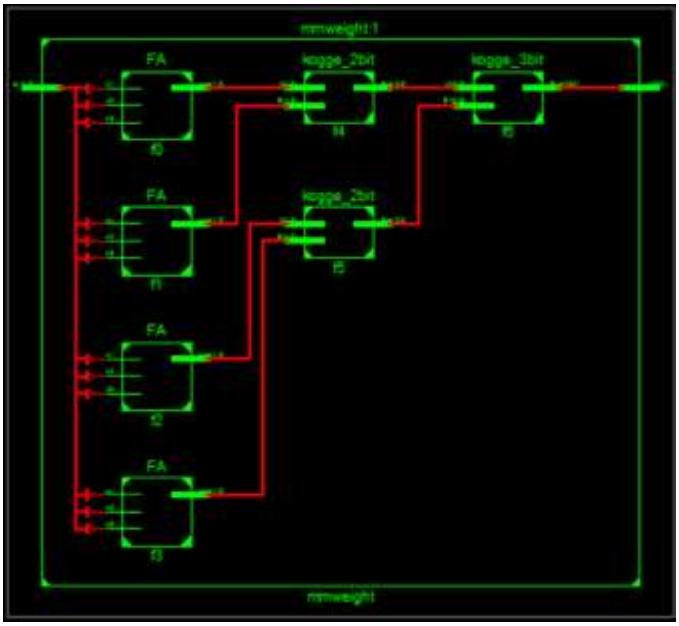

Fig. 6: View Technology Schematic of Weight Measurement

```
Device utilization summary:
---------------------------

Selected Device : 3s50pq208-4

 Number of Slices:                       10  out of    768     1%
 Number of 4 input LUTs:                 18  out of   1536     1%
 Number of IOs:                          16
 Number of bonded IOBs:                  16  out of    124    12%
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 12.627ns
```

Fig. 7: synthesize Result by Weight Measurement

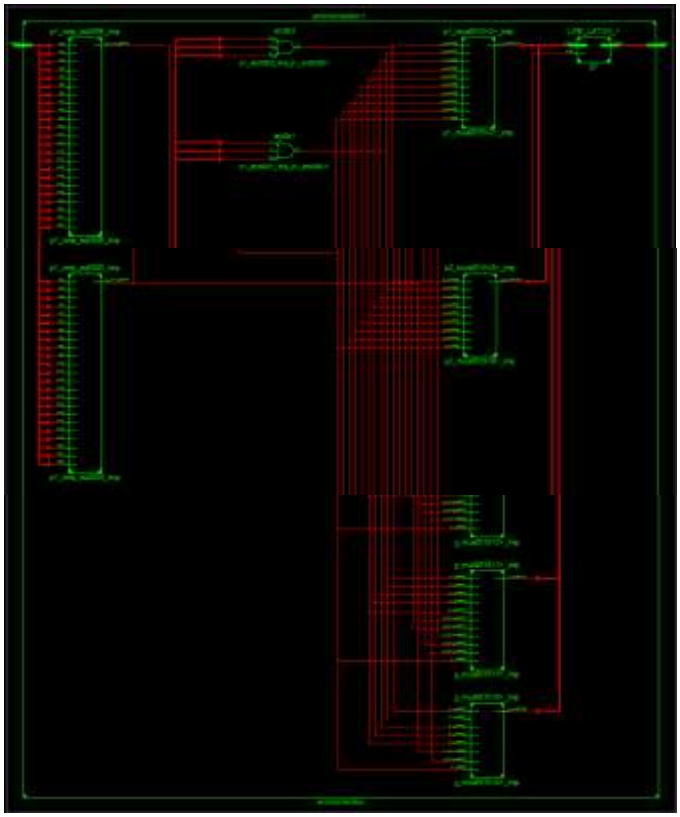

Fig. 8: View Technology Schematic of Error Detection and Correction

```
Device utilization summary:
---------------------------

Selected Device : 3s50pq208-4

 Number of Slices:                       54  out of    768     7%
 Number of 4 input LUTs:                 92  out of   1536     5%
 Number of IOs:                          47
 Number of bonded IOBs:                  47  out of    124    37%
    IOB Flip Flops:                      23
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: 16.334ns
   Maximum output required time after clock: 7.078ns
   Maximum combinational path delay: No path found
```

Fig. 7: synthesize Result by Error Detection and Correction

V. CONCLUSION

This paper presented the error detection golay technique based on CRC binary division using basic XOR bitwise operation and algorithm to implement it. It works on the concept of redundancy that is to append some extra bits in data unit known as redundant bits to detect the error and implemented at the Data Link layer of OSI model. There are a number of techniques used for error detection at data link layer among which CRC provides desirable efficiency. It provides good performance in terms of accuracy and security compared to other techniques. The future challenge is to make the code applicable to real world entity for example, audio, image etc.

REFRENCES

[1] Kristjane Koleci, Paolo Santini, Marco Baldi, Franco Chiaraluce, Maurizio Martina And Guido Masera, “Efficient Hardware Implementation of the LEDAcrypt Decoder”, IEEE Access 2021.

[2] P. Santini, M. Battaglioni, M. Baldi, and F. Chiaraluce, “Analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding and application to cryptography,”, IEEE Trans. Communication, vol. 68, no. 8, pp. 4648_4660, Aug. 2020.

[3] J. Hu, M. Baldi, P. Santini, N. Zeng, S. Ling, and H. Wang, “Lightweight key encapsulation using LDPC codes on FPGAs”, IEEE Trans. Comput., vol. 69, no. 3, pp. 327_341, Mar. 2020.

[4] D. Zoni, A. Galimberti, andW. Fornaciari, ``Efficient and scalable FPGA oriented design of QC-LDPC bit-flipping decoders for post-quantum cryptography," *IEEE Access*, vol. 8, pp. 163419_163433, 2020.

[5] K.Koleci, M. Baldi, M. Martina, and G. Masera, ``Ahardware implementation for code-based post-quantum asymmetric cryptography," in *Proc. 3rd Italian Conf. Cybersecurity (ITASEC)*, vol. 2597, Ancona, Italy, Feb. 2020, pp. 141_152.

[6] D. Zoni, A. Galimberti, and W. Fornaciari, ``Flexible and scalable FPGA oriented design of multipliers for large binary polynomials," *IEEE Access*, vol. 8, pp. 75809_75821, 2020.

[7] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini, ``LEDAcrypt: QC-LDPC code-based cryptosystems with bounded decryption failure rate," in *Code-Based Cryptography*, M. Baldi, E. Persichetti, and P. Santini, Eds. Cham, Switzerland: Springer, 2019, pp. 11_43.

[8] Shivani Tambatkar, Siddharth Narayana Menon, Sudarshan. V, M. Vinodhini and N.S.Murty, “Error Detection and Correction in Semiconductor Memories using 3D Parity Check Code with Hamming Code”, International Conference on Communication and Signal Processing, April 6-8, 2017, India.

[9] Wenhe JIN, Shaohua WU, Erpeng YANG and Jian JIAO, “LDPC Convolutional Codes Coded

Cooperation Based on Puncturing", ISBN 978-89-968650-9-4 ICACT2017 February 19 ~ 22, IEEE 2017.

[10] Pallavi Bhoyar, "Design of Encoder and Decoder for Golay code", International Conference on Communication and Signal Processing, April 6-8, IEEE 2017, India.

[11] Manikandan J, Shruthi S, Mangala SJ and Agrawal VK, "Design and Implementation of Reconfigurable Coders for Communication Systems", International Conference on VLSI Systems, Architectures, Technology and Applications (VLSI-SATA) IEEE 2016.

[12] Sindhuaja Muppalla and Koteswara Rao Vaddempudi, "A Novel VHDL Implementation of UART with Single Error Correction and Double Error Detection Capability", SPACES-2015, Dept of ECE, KL University.

[13] Satyabrata Sarangi and Swapna Banerjee, "Efficient Hardware Implementation of Encoder and Decoder for Golay Code", IEEE Transactions On Very Large Scale Integration (VLSI) Systems 2014.

[14] Nitin Patel, Naresh Patel, " VHDL Implementation of UART with BIST capability, IEEE 4th ICCCNT July 4 - 6, 2013, Tiruchengode, India.

[15] Shumit Saha, Md. Ashikur Rahman, Amit Thakur, "Design and Implementation of a BIST Embedded High Speed RS-422 Utilized UART over FPGA", IEEE 4th ICCCNT July 4 - 6, 2013, Tiruchengode, India.