# ENHANCE ORTHOGONAL CODE CONVOLUTION CAPABILITIES FOR EFFICIENT DIGITAL COMMUNICATION

Akanksha Kant
VLSI DESIGN, Indira Gandhi Delhi Technical University for Women
New-Delhi, India

***Abstract:* Real time transmission of data at high bit rates is one of the increasing demands today. This has inadvertently made data susceptible to noise resulting in impairment during transmission. In this paper, Orthogonal Codes have been developed and realized using Xilinx software for efficient digital communication for not only can they detect and correct errors but also do away with the requirement to send the parity bit. The result shows that the proposed technique enhances the error detection capabilities of orthogonal code convolution from 50% to 99.99% for 32 bit orthogonal code and corrects up to (n/4-1) bits of error in the received impaired code.**

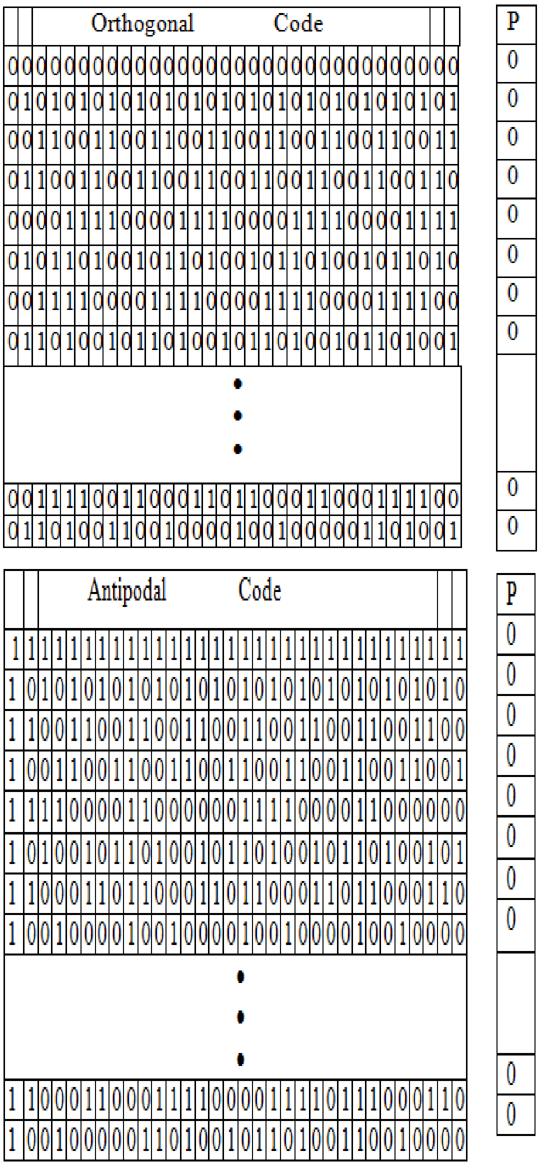
***Key-words:* Orthogonal Codes, Antipodal Codes, Error Detection and Correction.**

## I. INTRODUCTION

In a world with rapidly increasing demands to share large quantities of data in different forms such as text, high quality audio and video, both efficiency and bandwidth requirements come into picture. Reliability of data though is affected by the channel over which it is transmitted and by the noise and crosstalk to which it is highly susceptible. Coding is one of the best solutions to achieve high channel efficiency as they can correct errors. While existing techniques namely Cyclic Redundancy Check [3, 4] can detect errors and Solomon Codes [6] can both detect and correct errors, they fail to meet rising efficiency and bandwidth requirements. Among these techniques, Orthogonal Codes are one of the codes which can detect and correct corrupted data efficiently. Our objective in this paper is to implement orthogonal code convolution technique using Xilinx software and enhance its error control capabilities for efficient digital communication. Section second and third presents theory of orthogonal codes and design methodology respectively. In section fourth results are shown followed by conclusion in fifth section.

## II. ORTHOGONAL CODES

Orthogonal codes, being binary valued consists of equal number of 1's and 0's i.e. for n bit orthogonal code there n/2 1's and n/2 0's. Due to equal number of 1's and 0's, orthogonal codes will generate zero parity bits and are referred to as zero parity bits. Antipodal codes are simply the inverse of orthogonal codes and possess the same properties. Fig.1 illustrates the concept of orthogonal codes via 32 bit orthogonal code as shown.



| Orthogonal Code | P |
|---|---|
| 00000000000000000000000000000000 | 0 |
| 01010101010101010101010101010101 | 0 |
| 00110011001100110011001100110011 | 0 |
| 01100110011001100110011001100110 | 0 |
| 00001111000011110000111100001111 | 0 |
| 01011010010110100101101001011010 | 0 |
| 00111100001111000011110000111100 | 0 |
| 01101001011010010110100101101001 | 0 |
| •<br>•<br>• | |
| 00111100110001101100011000111100 | 0 |
| 01101001100100001001000001101001 | 0 |

| Antipodal Code | P |
|---|---|
| 11111111111111111111111111111111 | 0 |
| 10101010101010101010101010101010 | 0 |
| 11001100110011001100110011001100 | 0 |
| 10011001100110011001100110011001 | 0 |
| 11110000110000001111000011000000 | 0 |
| 10100101101001011010010110100101 | 0 |
| 11000110110001101100011011000110 | 0 |
| 10010000100100001001000010010000 | 0 |
| •<br>•<br>• | |
| 11000110001111000011110111000110 | 0 |
| 10010000011010010110100110010000 | 0 |

Fig.1 32- bit orthogonal code has 32 orthogonal and 32 antipodal codes.

There are 32 orthogonal codes and 32 antipodal codes for a total of 64 bi-orthogonal codes that constitutes a 32 bit orthogonal code. It is noted that with this method, transmitter does not have to send an additional parity bit since their parity bit will always remain zero. This property is exploited in digital communication where any error occurred during transmission can be detected at the receiver end by generating a parity bit. In this method, a k bit data to be transmitted is first mapped into a unique n bit orthogonal code before transmission. In this paper, a 6 bit data set is considered which is mapped into a unique 32 bit orthogonal code and transmitted devoid of parity bit. When received it is decoded to retrieve the original data sent based on code correlation. This is achieved by setting a threshold midway between two orthogonal codes, as given by the following equation

$$d_{th} = n/4 \qquad (1)$$

Here n indicates code length and d $_{th}$ represents threshold midway between two orthogonal codes. Therefore threshold midway for a 32 bit orthogonal code is 8. Fig. 2 illustrates encoding and decoding during digital communication.

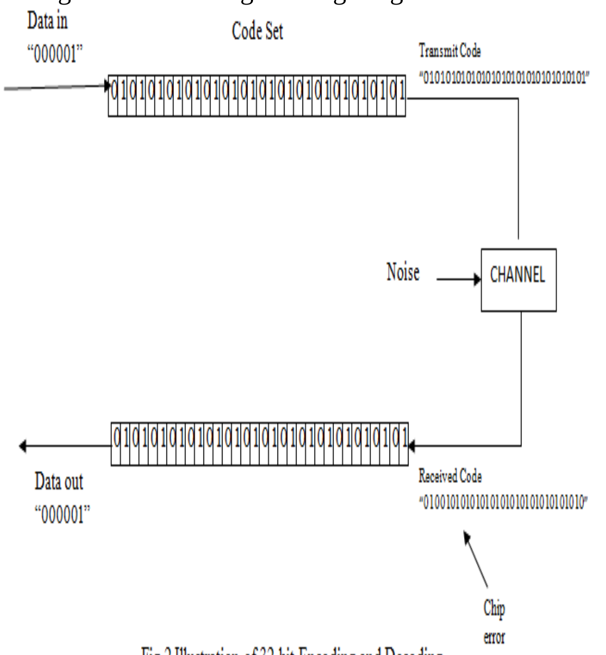


Fig.2 Illustration of 32-bit Encoding and Decoding.

In this technique a decision process takes place when the received and impaired orthogonal code is examined for correlation with the neighbouring codes for a possible match. Only a good auto-correlation value for n bit comparison can ensure the acceptance of a valid code, else a false detection will occur. This involves a correlation process between a pair of n-bit codes $x_1, x_2, x_3,\ldots\ldots x_n$ and $y_1, y_2, y_3, \ldots.y_n$ . The two are compared to yield an auto- correlation value represented by the following equation

$$R(x,y) = \sum_{i=1}^{n} x_i y_i \leq \frac{n}{4} - 1 \qquad (2)$$

Here R(x, y) is auto-correlation function, n is code length and d $_{th}$ is threshold midway. Since threshold (d $_{th}$) is between a pair of valid codes, an additional 1 bit offset is added as shown in (2) for reliable detection. Both (1) and (2) are further combined to represent the average number of errors which can be corrected by (3).

$$t = n - R(x,y) = \frac{n}{4} - 1 \qquad (3)$$

Where t gives the number of errors that can be corrected via n-bit orthogonal code. Table 1 has tabulated different bits of orthogonal codes and their respective error correction capabilities as per (3).

TABLE 1

Orthogonal Codes (n bit) and their respective Error Control Capabilities.

| n | t |
|---|---|
| 8 | 1 |
| 16 | 3 |
| 32 | 7 |
| 64 | 15 |

## III. DESIGN METHODOLOGY

In earlier techniques, error detection percentage was only 50% because the parity bit remained unchanged for even number of errors making number of detectable errors equal to $2^n$ /2 combinations of received code. However in our approach instead of going for parity generation, the received code is compared with a set of combination of orthogonal codes, already stored in a look up table. Orthogonal codes have zero parity. Hence if data is corrupted during transmission, it will not generate zero parity at the receiver end thereby enabling error detection. The entire design can be disintegrated into transmitter and receiver.

### *A. Transmitter*

The transmitter contains an encoder which encodes a k-bit data to n= 2k-1 bits of the orthogonal code and a shift register which transforms this code to serial bits of data to be transmitted serially as shown in block diagram Fig.3. Here 6-bit data is encoded to 32-bit (25) orthogonal code, shown in Fig.2.

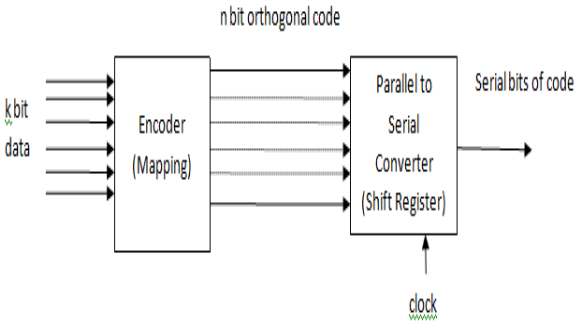


Fig.3. Block Diagram of the transmitter.

Copyright 2014.All rights reserved.

*B. Receiver*

At the receiver end the incoming serial bits are converted into n bit parallel code via another shift register as shown in Fig.4. This received code is then compared with all combinations of orthogonal codes stored in the look up table. This is achieved by performing XOR operation between the received code and each combination of orthogonal codes in the look up table and counting the number of ones in the result. A counter counts the number of ones and keeps a tab on the minimum count that can be corrected. A zero count indicates uncorrupted data and the orthogonal code from the look up table associated with the minimum count is the closest match for the corrupted received code. The matched orthogonal code is the correct code. It is decoded to give back the k bit data initially transmitted. Minimum number of bits that can be corrected is given by (3).

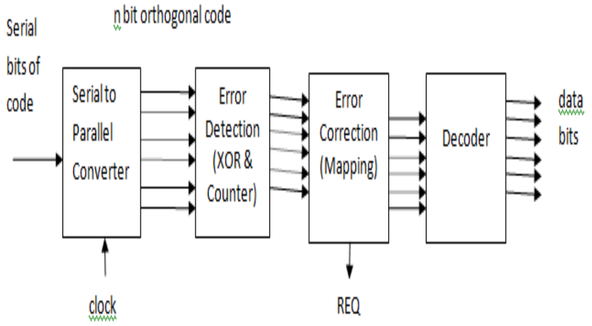


Fig.4 Block Diagram of the receiver

## IV. IMPLEMENTATION AND RESULTS

Simulation has been performed using Xilinx software. The simulation results for 32-bit orthogonal code are explained for the transmitter and receiver as follows.

*A. Transmitter*

Fig.5. shows the simulation results of transmitter. The 6-bit input data ain “000001” is encoded to the corresponding 32-bit orthogonal code labelled as tout “01010101010101010101010101010101” to be transmitted serially.

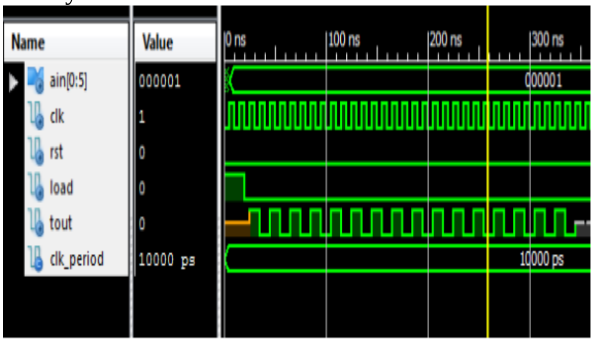


Fig.5 Simulation results of transmitter 1

*B. Receiver*

At the receiver end, the incoming serial bits labelled as ‘din’ are transformed into a parallel 32-bit received code. XOR operation is performed between this received code and each combination of orthogonal code in the look up table. A counter counts the number of 1’s in the resulting value and error signal is raised to ‘1’ if the value of count is not zero. The orthogonal code from the look up table associated with the minimum count is the closest match for the corrupted received code which is further decoded to yield the output data labelled as ‘aout’. Fig.6. shows that the received code “01001010101010101010101010101010” is corrupted giving error equal to 1. Fig.7 shows that the matched orthogonal code zo associated with minimum count is decoded to give back final data aout = “000001”.

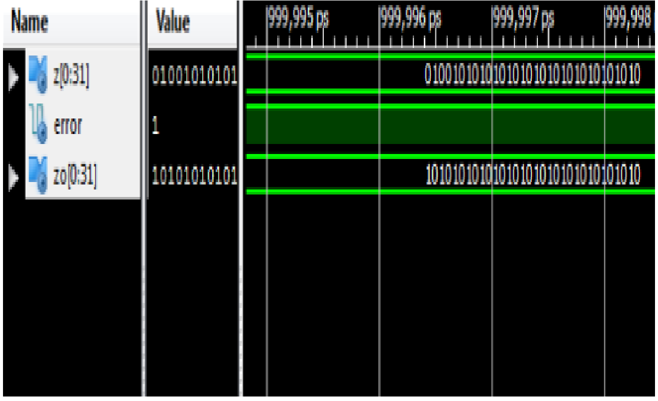


Fig.6 Error detection and correction

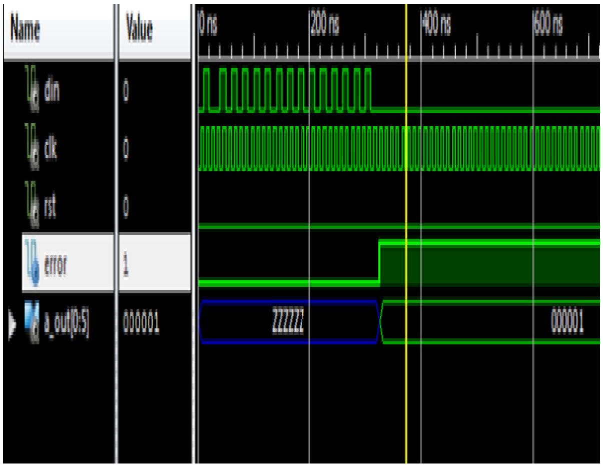
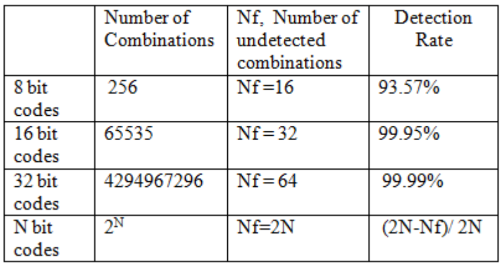


Fig.7 Simulation results of receiver

*C. Results*

The simulation results show that the n-bit orthogonal code is able to correct up to (n/4)-1 bits of error. The detection percentage is (2n – 2k) / 2n % where 2k is the number of combinations of orthogonal code stored in the look up table which remain undetected. Thus excluding 2k combinations, any faulty combination can be detected by the proposed technique. For 32- bit orthogonal code, detection percentage is 99.99% clearly an improvement over 50% obtained in previous techniques of parity generation methods. It is also able to correct up to 7 bit errors. Table 2 has summarized the detection percentage for 8-bit, 16-bit and N bit orthogonal codes.

TABLE 2
Error Detection Capabilities of Orthogonal Code (N-bit)



| | Number of Combinations | Nf, Number of undetected combinations | Detection Rate |
|---|---|---|---|
| 8 bit codes | 256 | Nf=16 | 93.57% |
| 16 bit codes | 65535 | Nf=32 | 99.95% |
| 32 bit codes | 4294967296 | Nf=64 | 99.99% |
| N bit codes | $2^N$ | Nf=2N | (2N-Nf)/ 2N |

## V. CONCLUSION

Orthogonal Code implementation is presented to ensure efficient digital communication. Feasibility of the design is ensured by implementing the modules of transmitter and receiver using synthesizable VHDL codes. The results of orthogonal code implementation show that this technique improved the error detection from 50% to 99.9% for 32 bit orthogonal code. Finally this work has the future scope of further improvement in correction capabilities of orthogonal code, speed up data processing through parallel implementation and improve orthogonal coding for large digital data transmission.

## REFERENCES

[1] N. Kaabouch, A. Dhirde, Member, S. Faruque, “Improvement of the Orthogonal Code Convolution Capabilities Using FPGA Implementation”, IEEE EIT 2007 Proceedings.

[2] R. Dubey & P. Sarojini, “FPGA Implementation of Orthogonal Code Convolution for Efficient Digital Communication”, International Journal of Engineering Research & Technology (IJERT), Vol. 1 Issue 10, December- 2012.

[3] Baicheva, T., S. Dodunekov, and P. Kazakov, “Undetected error probability performance of cyclic redundancy- check codes of 16-bit redundancy,” IEEE Proc. Comms., Vol. 147, No. 5, Oct. 2000, pp. 253-256.

[4] A. Hokanin, H. Delic, S. Sarin, “Two dimensional CRC for efficient transmission of ATM Cells over CDMA,” IEEE Communications Letters, Vol. 4, No. 4, April 2000.

[5] Saleh Faruque , “Error Control Coding Based on Orthogonal Codes”, Wireless Proceedings, Vol. 2, pp. 608-615, 2004.

[6] Stylianakis V., Toptchiyski S, “A Reed-Solomon coding/decoding structure for an ADS modem,” Electronics, Circuits and Systems, 1999. Proceedings of ICECS apos ; 99.The 6th IEEE International Conference, Volume 1, Issue , 1999, pp. 473 – 476.

Copyright 2014.All rights reserved.