DS-CDMA Transmitter and Receiver

Talha saeed
Department of Electrical Engineering
COMSATS Institute of Information Technology (CIIT)
Islamabad, Pakistan

Muhammad shoaib
Department of Electrical Engineering
COMSATS Institute of Information Technology (CIIT)
Islamabad, Pakistan

Zubair iftikhar
Department of Electrical Engineering
COMSATS Institute of Information Technology (CIIT)
Islamabad, Pakistan

Bushra Ghouri
Department of Physics
COMSATS Institute of Information Technology (CIIT)
Islamabad, Pakistan

***Abstract*—this paper describes the DS-CDMA Transmitter and Receiver for two users. This project has been implemented on Spartan 3 FPGA by using Verilog HDL. Five separate modules have been implemented for transmitter which was the Convolutional Encoder, the Frequency Divider, the Block Interleaver, the Long PN Code, and the Walsh Code. In Receiver Side two new modules have been added the Walsh Code, and the Viterbi Decoder. ModelSim SE 5.7g has been used for simulation. While ISE Webpack 10.1 Xilinx has been used for synthesizing, mapping, and for downloading the modules on FPGA. This DS-CDMA system contains data rates up to 50 Mbps.**

***Index Terms*—DS-CDMA, FPGA, transmitter, receiver**

I.INTRODUCTION

CDMA provides ability to all users to use same frequency and same time at same channel. Users are distinguished by different codes. Even though at same carrier frequency is used at which all stations operate, the receiving channel already knows the specific code which is assigned to the transmitting station [10]. From the desired station it can extract the desired transmitted signal and will reject all other stations that are using other codes. Spread spectrum systems have two major types. These can be termed as Frequency Hopping (FH) and Direct Sequence (DS) system. CDMA belongs to DS spread spectrum system, so we will only stress upon this [5] [12]. The transmitter structure can be divided into different components. At first step is the convolution encoder. In this we are adding some additional bits for making bit error checking more successful and allow for more accurate transfer of data. Convolutional Encoder was used with constraint length of 3 (k = 3) and rate ½. Its mean that for every input bit it can generates 2 output bit. After that Block Interleaver has been used that rearranges the order of a sequence of input Symbols for randomized the location of error, by scrambling or rearranging the sequence of input bits. This was implemented through 24x16 matrixes. Output of this performs XOR operation with Long PN Code. PN code is used here for encryption and spreading. We have implemented the PN Code with a 42-stages shift register [10].

In next step data is multiplied with Walsh Code. The Walsh Code are the orthogonal codes that are assigned each and every user to distinguish each user and for spreading. The receiver structure can be broken down into several components. In first step data is multiplied with Walsh Code. The output of the Walsh Code is then performs XOR operation with Long PN Code. PN Code is used here for privacy of user and for spreading. On receiver side the PN Code is used to despread the signal that has been spreaded at transmitter side. The PN Code is implemented with a 42-stages shift register [10].

After despreading of signal the output is fed into the Deinterlever. It performs operation on a data matrix that is the reversed form of that operation which is performed at data matrix in Block Interlever at transmitting side. The output of Deinterlever is given to Viterbi Decoder. The Viterbi Decoder uses maximum likelihood decoding that decodes the data which has been encoded by the Convolutional Encoder at transmitter side and finally we receive our desire data [14] [15].

II. CONVOLUTIONAL ENCODER

A. Working Principle

Convolutional Encoder is used to take input bits and generate a matrix of encoded outputs. It is used for channel encoding in digital communication systems because noise can change the actual value of the signal. So there have been added additional bits for making bit error checking more successful and allow for more accurate transfer of data. The convolution encoder that was used in this DS-CDMA is of constraint length of 3 (k = 3) and rate ½. This means that for every bit of input it can generates a 2 bit code word. So, for every input sequence the encoder transform in to a unique code word sequence. The Figure 1 shows the working of Convolutional Encoder.

B. Implementation

Convolutional Encoder is a bottom level module. Two bits right shifted register is used in this module, actually three bits shift register is needed for Convolutional Encoder. But here replaced the one bit of shift register with input bit due to avoid any unsynchronization. For every input bit, Convolutional Encoder will generate two output bits LSB and MSB. Input bits of Convolutional Encoder arrive at data rate of ‘clock_a’ (781.250 Kbps) and output bits are generated at data rate of ‘clock_b’ (1.5625 Mbps) as shown in figure 1and 2.

C. Figures

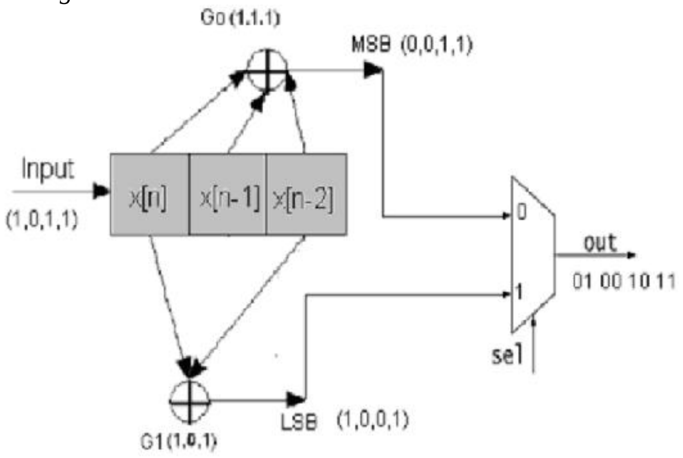

Fig 1 Convolutional Encoder

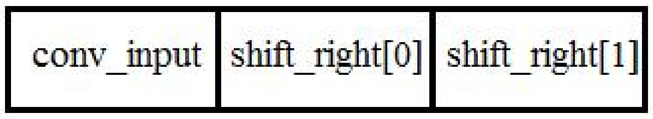

Fig 2 Shift Register

D. Wave Diagram

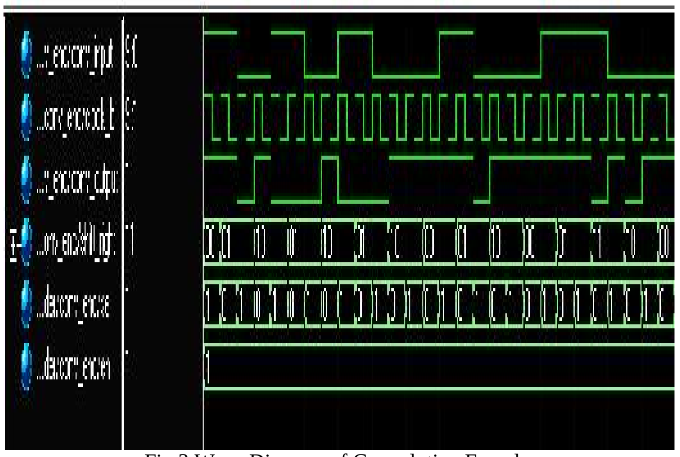

Fig 3 Wave Diagram of Convolution Encoder

III. FREQUENCY DIVIDER

A. Working Principle

Frequency Divider is also bottom level module for generating following three different frequencies which are needed for this project. *50 Mbps (clock_d)* 50 Mbps is the global clock of Spartan 3 FPGA. It has been used in implementation of Walsh code and Dewalsh code. It has also been used for taking data from PC to FPGA and FPGA to PC through UART. Pin number for system clock in xcs200 FT256 Spartan 3 FPGA is T9. *3.125 Mbps (clock_c)* [11]. It is used for generating long PN code and for XOR operation between long PN code and data. This frequency should be 16 times less than the clock_d (system clock) *1.5625 Mbps (clock_b)* Bits pair Convolutional Encoder in figure 3, Viterbi Decoder and Block Interleaver are implemented through this frequency. It should be 64 times less than the clock_c frequency. But due to limitation of FPGA resources here it is two times less than the clock_c frequency. *781.250 Kbps (clock_a)* Arrival of data in system is done at this clock speed. It is two less than the clock_b frequency and four times less than the clock_c frequency shown in figure 4.

B. Wave Diagram

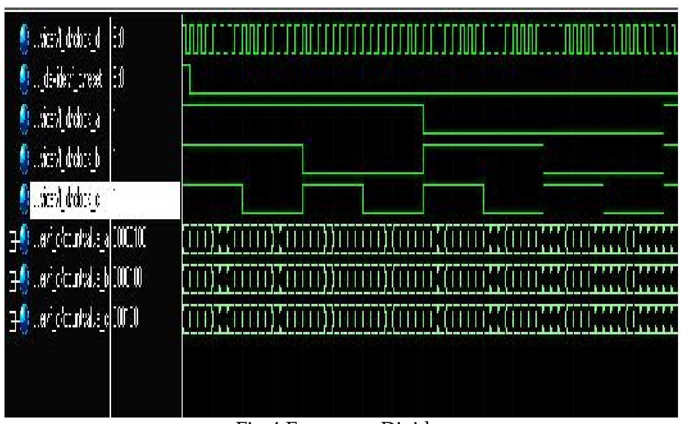

Fig 4 Frequency Divider

IV. BLOCK INTERLEAVER

A. Working Principle

An interleaver rearranges the order of a sequence of input symbols. It is used for randomized the location of error, by scrambling or rearranging the sequence of input bits. Interleaving spreads a burst of errors out so that error correction circuits have a better chance of correcting the data. The interlever which is used at transmitting end must be used the reverse of that interlever at receiving end for recover the original data [3].

B. Implementation

In following order Block Interleaver performs its operation.

1. Input bits in row wise from left to right and start from top row.
2. Inter-row permutations.
3. Inter-column permutations.
4. Column-wise data out, top to bottom, starting with the left Column

Row permutation is used in following manner.

1. 0 row -> 23 row
2. 1 row -> 0 row
3. 2 row -> 1 row
4. 3 row -> 2 row
5. . …….
6. . ...
7. 22 row -> 21 row
8. 23 row -> 22 row

Column permutation is used in following manner.

1. 0 column -> 14 column
2. 14 column -> 15 column
3. 15 column -> 0 column

Block Interleaver is also a bottom level module. It has contained a 24x16 size matrix for randomize the locations of errors within 384 bits. Two 24x16 shift registers 'hold' and 'block' are used to perform the operations of this module.('hold' for input data and 'block for output 'data').It works on data rate of 'clock_b', 'Hold' and 'block' worked in parallel. But at start when hold is taking first 384 input bits then that time no previous data is available in 'block'. So 'block' which is used for doing output will perform no operation and there will be delay (time for 384 bits). After that 'hold' matrix is taking input of current data and 'block' matrix is doing output of previous data at every positive edge of 'clock_b'. Last input bit of current data and last output bit of previous data will occur at same time. So 'block' matrix at this time will give last output bit of previous data and will take new data from 'hold' matrix and last input bit. Also shifts new current data according to row and column permutations at their original places. So at next pulse of 'clock_b' again 'hold' matrix is taking input of current data and 'block' matrix is doing output of previous data.

Control signal 'resetn' indicates (when changes from logic high to logic low) that data from Block Interleaver is ready for XOR operation and it will set to logic '0' when Block Interleaver starts to give outputs.

C. Wave Diagram

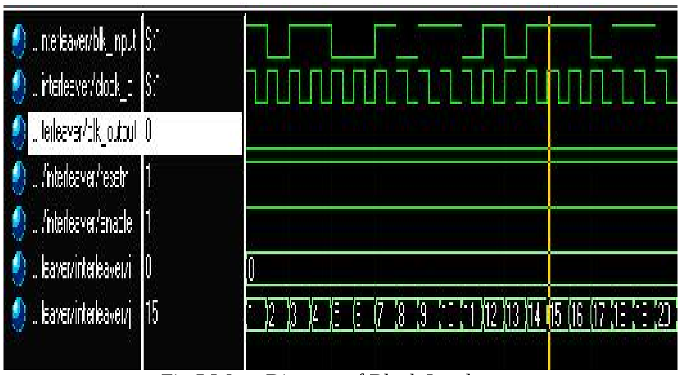

Fig 5 Wave Diagram of Block Interleaver

In the above wave diagram 'blk_output' is zero its means that Block Interleaver has not completed his matrix of 24x16. So at start when hold is taking first 384 input bits then at that time no previous data is available in the matrix. Therefore 'block' matrix which is used for doing output will perform no operation. So that's why 'blk_output' is zero.

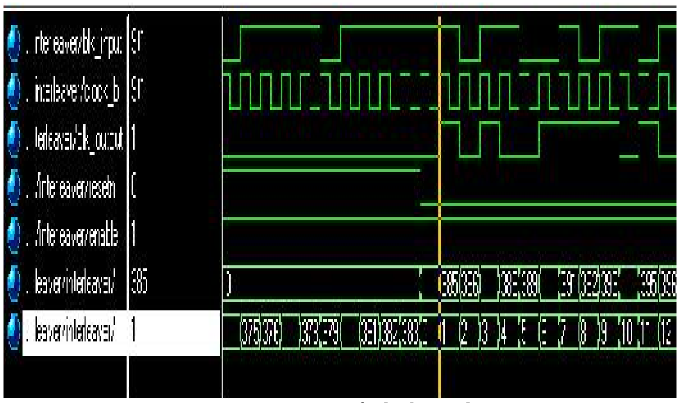

Fig 6 Wave Diagram of Block Interleaver

In the above figure 6 yellow line indicates that now 'block' matrix is doing output through 'blk_output' of previous data at every positive edge of 'clock_b'. And hold' matrix is taking input of current data if data is available at input.

V. LONG PN CODE

A. Working Principle

In the forward channel, direct sequence is used for data scrambling. Each user has assigned the long PN sequence code with period 2^42-1 chips [12]. The long PN code is specified by the following characteristic polynomial.

P(x) = X^42 + X^35 + X^33 + X^31 + X^27 + X^26 +X^25+ X^22 + X^21 + X^19 + X^18 + X^17 +X^16 + X^10+X^7 + X^6 + X^5 + X^3 + X^2 + X^1 + 1…….. (1)

It is used to both spread the signal and to encrypt it [8] as in figure 7.

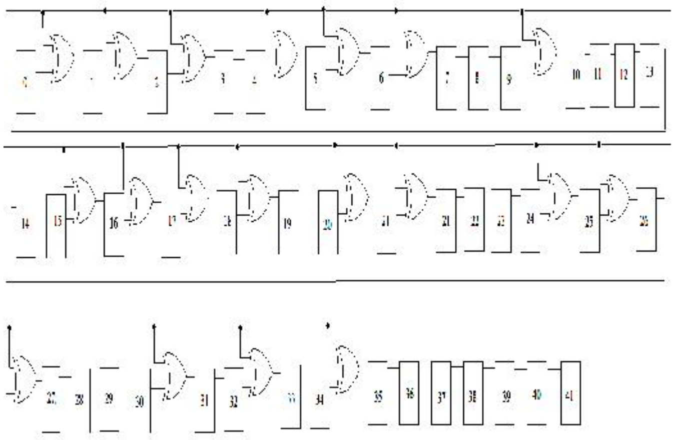
Figure 7 Architecture of Long PN Code

B. Implementation

Long PN Code is the bottom level module. It is implemented with 42 bits shift register and 19 XOR operations. This specification is according to IS-95 standards.
This module works at data rate of 'clock_c'. When control signal 'resetn' changes from logic high to logic low then at start all 42 shift registers store the logic high value '1'. After that according to architectural of shift registers, values are shifted. Last shift register 'sr[41]' is used for 'long_pn_out' and for feedback mechanism shown in figure 8.

C. Wave Diagram

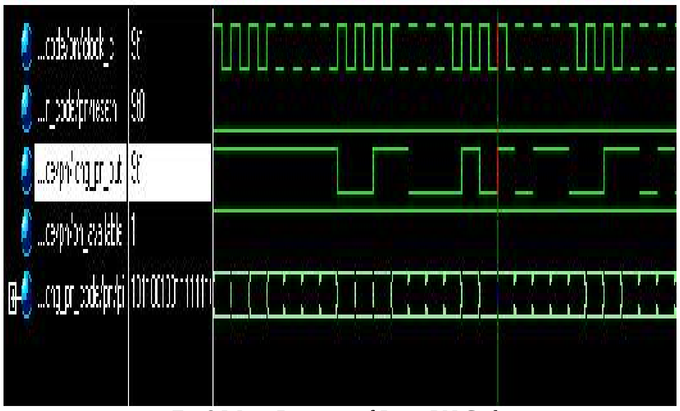
Fig 8 Wave Diagram of Long PN Code

VI. WALSH CODE

A. Working Principle

All the Walsh Codes are orthogonal to each other, which means that the cross correlation between Walsh Code with each other is zero [6].The main functionality of Walsh codes in CDMA is to provide orthogonally among all the users in a cell so that they can uniquely identifies. Each user traffic channel is assigned a different Walsh code by the base station [6]. For IS-95, 64 codes are available. In other words, a base station can talk to a maximum of 64 mobiles at the same time. Whereas CDMA 2000 can use up to 256 such codes. Walsh Code is constructed by Haddamard matrices. The code length is the size of the matrix. Each row is one Walsh Code of size N. The first matrix gives us two codes; 11, 10. The second matrix gives: 1111, 1010, 1100, 1001 and so on [6].

B. Implementation

4x4 Hadamard metrics was used for assigning Walsh Code to each user. Row number one contains all '1' so this row did not assign to any user [17]. Row number two is assigned to user one and row number three is assigned to user two as shown in figure 9.

$$[1]$$

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

$$\begin{array}{c l} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} & \begin{array}{l} \\ \texttt{walsh code for user 1} \\ \texttt{walsh code for user 2} \\ \\ \end{array} \end{array}$$

Figure 9 Walsh Code for User 1 and 2

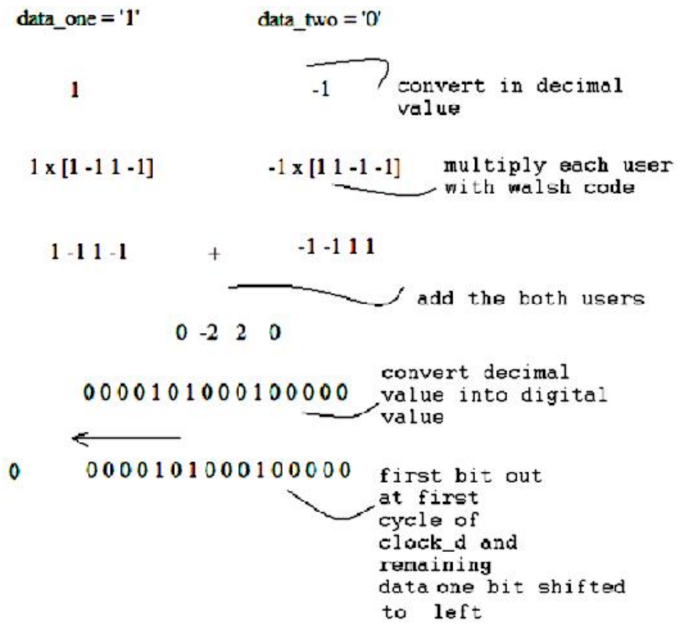

Fig 10 Procedure of Walsh Code

Calculated sixteen bits values for all four possible combinations of two users have been used as we can see in the Figure 10. 'clock_d' is sixteen times greater than the 'clock_c'. Therefore time duration at which input data remains (occurring at the rate of 'clock_c'), so at that time sixteen Walsh output bits (occurring at the rate of 'clock_d') will be generated for that particular input. As shown in Figure 11.

C. Wave Diagram

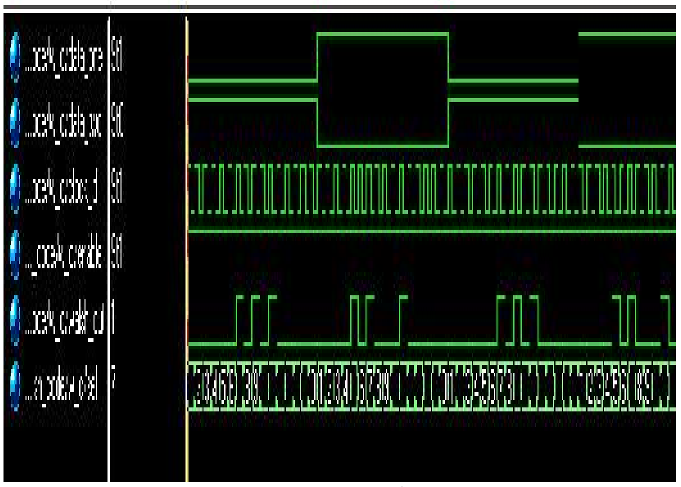

Fig 11 Wave Diagram of Walsh Code

VII. INTERFACING OF TRANSMITTER MODULES

A. Implementation

PC was used for storing data in FPGA and for showing results back to PC. For this purpose UART module was implemented in FPGA with two FSM for transmitter and receiver [11]. This serial communication system communication is done through serial cable so did not need for modulation like QPSK. So here is the communication of Baseband Transmitter and Receiver in this project. In the following Diagram Fig. 12, we can see interfacing of blocks in Transmitter side.

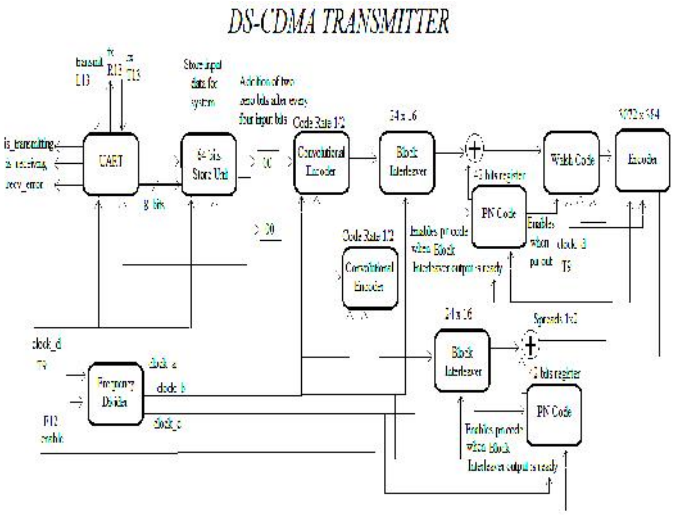

Figure 12 Interfacing of Transmitter Modules

Initially data is taken from PC to UART module at Transmission line 'tx'. It is T13 pin number in Spartan 3 FPGA. It will take data at a frame size of 8 bits at the rate of 'clock_d'. UART module gives each frame at this rate to store unit. Store Unit is the small portion in main module of Transmitter. Due to memory and resources limitation of FPGA we can take only 32 bits for each user. F12 pin of FPGA has been used here as enabling pin of system so when presses this switch then whole system will starts to work. 'Enable' signal will be given at all modules. But before press 'enable' we must need to store data for two units in store unit. So at start of system Store Units gives a bit at a rate of 'clock_a' to the Convolutional Encoder. Stores unit also performs the addition of two zeros after every four output bits as a requirement of Viterbi Decoder. Because Viterbi Decoder must need two zeros at the end of frame for going back to initial state at next frame. After addition of two zeros total bits for each user will be 48 bits. Frequency Divider provides the frequencies to different blocks according to their requirements [14].

At the next stage Convolutional Encoder will generate the output at a rate of 'clcok_b'. And after that total number of each user bits will be 96. Meanwhile when Convolutional Encoder output is not ready Block Interleaver will not start to work [15]. We have put delay in Block Interleaver for this purpose. In addition to these two Convolutional Encoders and all next blocks from here to Long PN Code will work parallel for both users. So here total bits of each user will become 192. At next when Block Interleaver output is ready it will enable the PN code and then Block Interleaver output and PN code chip will perform XOR operation at a spreading factor of two[10].

After this, spreaded data of both users will come to Walsh Code. So that the time at which each bit of two users remains in this module the Walsh Code performs Scrambling and generates sixteen output bits. At this output, system total bits will be 3072, reduced it into 384 bits by using Encoder due to FPGA resources limitation. After that it stored the data in UART by LIFO (Last in First Out) technique. And at the end by pressing 'transmit' switch L13 pin in Spartan 3 FPGA, data from UART will go back to PC. And PC will show the resultant values at Matlab with fread(s) command [15].

B. Wave Diagram

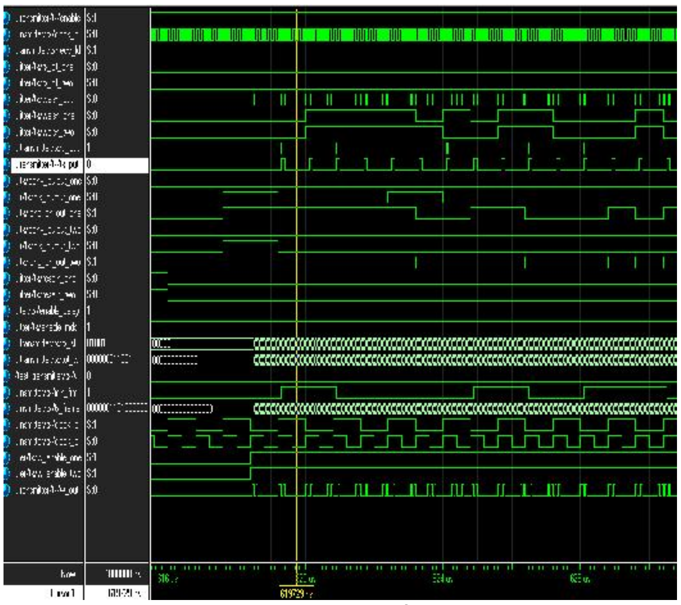

Fig 13 Wave Diagram of Transmitter

From figure 13 when 'resetn' at time 616us changes from logic '1' to logic '0' then 'block_long_PN_sum', 'blk_output' and long_pn_out' will be started their functions. Block Interleaver output is available at third cycle of 'clock_c' when 'resetn' goes from '1' to '0' logic, Synchronization between 'long_pn_out' and 'blk_output, 'long_pn_out', which works on 'clock_c' will be delayed until Block Interleaver output will become available. When both arrive on third 'clock_c' cycle, then on fourth cycle PN operation will start due to next state of flip flops. Each sixteen Walsh output values are encoded into two values which have been shown in wave form by 'u_out' signal. So when each sixteen bits frame of Walsh output completes then 'tx_put' signal goes high for two 'clock_d' pulses and for that time it will stores the value of 'u_out' on UART. Basically 'tx_put' enables the UART for two 'clock_d' pulses for storing resultant values of transmitter after every sixteen 'clock_d' pulses.

VIII. DE WALSH

A. Implementation

De Walsh is bottom level module at receiver side. It performs the operation for separating two user data. Each sixteen bits frame which has been decoded by Decoder arrives at De Walsh input on sixteen 'clock_d' pulses. At each pulse it will store input data in two different registers or matrices with the size of 4x4 for each user de Walsh operation. And then De Walsh performs multiplication of data with each user Walsh Code. After that it performs the addition of each user data (addition of rows). During this addition and multiplication procedure, it also stores incoming data in two matrices. Finally when sixteen input bits are completed then it gives output bit for each user at the rate of 'clock_c' on next 'clock_d' pulse. According to the decision, this has been taken on the MSB bit of 'sum_one' and 'sum_two'. If the MSB of these registers is '1' then output will be '0' and if the MSB of these registers is '0' then output will be '1'.

B. Wave Diagram

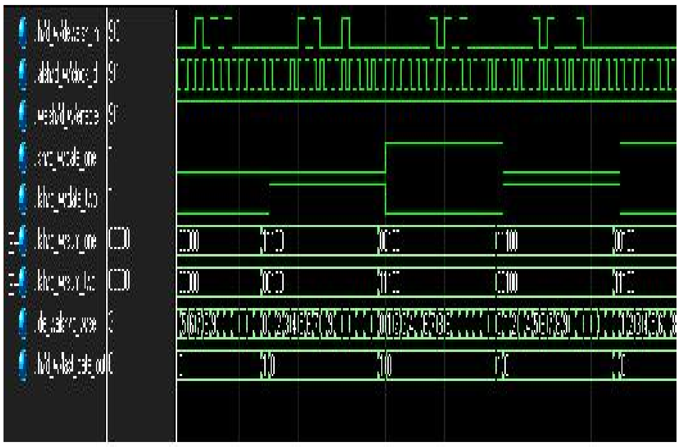

Fig 14 Wave Diagram of De Walsh

After every sixteen 'clock_d' cycles it will generates the corresponding output value for each user at the frequency of 'clock_c'. As shown in the Figure 14. Output which is generated in this diagram has been the same sequence as it was given at the input of Walsh Code at transmitter Side.

IX. VITERBI DECODER

A. Working Principle

In this system Viterbi Decoding algorithm is using as a Convolutional Decoder, to decode the code sequence that has been encoded by the Convolutional Encoder at transmitter side. For decoding of code sequences either be represented by tree or trellis structure. Here we are only using trellis structure for purpose of decoding.

B. Implementation

Viterbi Decoder is also bottom level module of receiver. Viterbi Decoder has main advantage is to remember the only four paths from thirty-two paths at every transition. Decision of choosing best four paths is according to paths hamming metric. The Viterbi Decoder here decodes the code word sequence of input 12 bits into 6 output bits, whereas first 4 bits are information bits while remaining 2 bits are zeros. Because it starts every frame from first state so it has needed two zeros at end of previous frame for going back to first state for next state as we can see in Fig. 15.

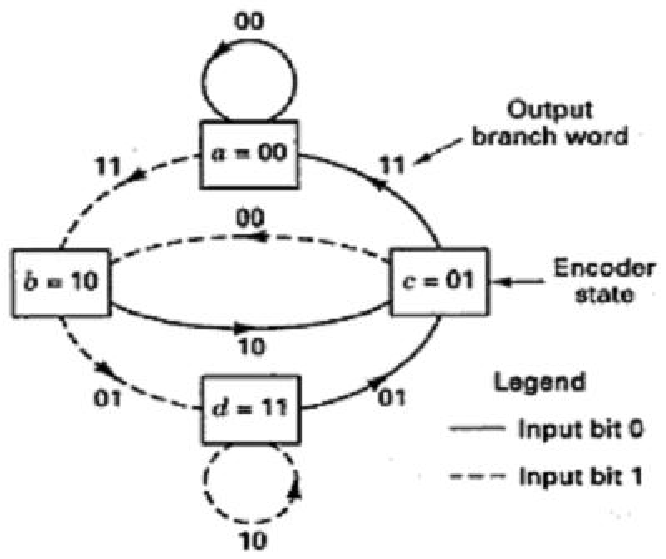

Fig 15 FSM of Convolutional Encoder

C. Wave Diagram

Input Convolved data is given to Viterbi Decoder at the rate of ‘clock_b’. And it will generate six output bits for each twelve input bits frame at the rate of ‘clock_a’. Containing four data bits and two zero bits in output frame. Here is the example where an input frames with adding error at six and eight bits. But Viterbi Decoder decodes the frames and gives accurate output as we can see in wave diagram Fig. 16. Output data sequence: 1 1 0 1 Original code word: 11 01 01 00 Transmitted code word: 11 01 00 01.

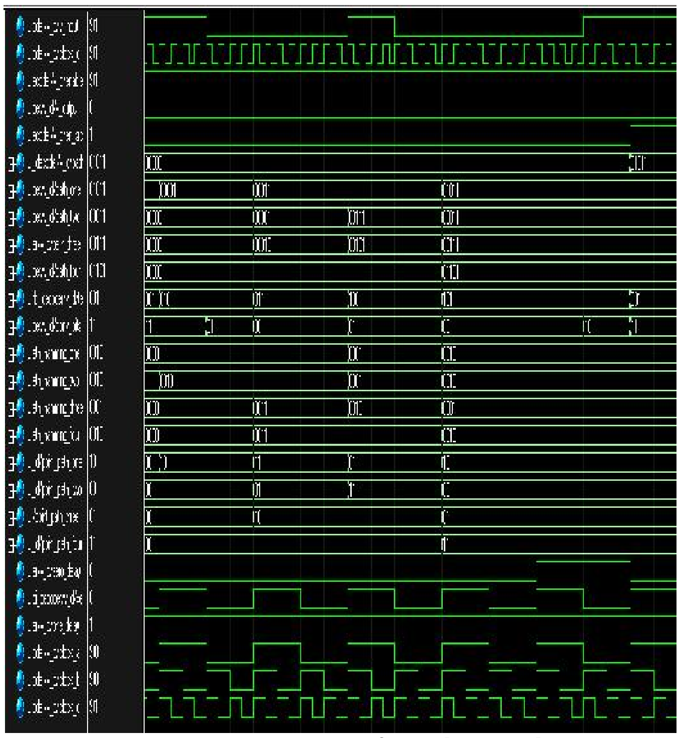
Fig 16 Wave Diagram of Viterbi Decoder

Here signal ‘path’ gives the output frame of six bits after completing twelve bits input frame.

X. INTERFACING OF RECEIVER MODULES

A. Implementation

PC is used for storing data in FPGA and for showing results back to PC. For this purpose UART module was implemented in FPGA with two FSM for transmitter and receiver. In the following Diagram Fig.17, we can see interfacing of blocks in Receiver side.

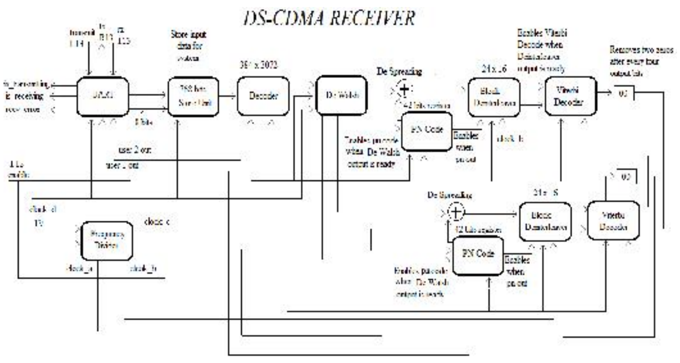

Figure 17 Interfacing of Receiver Modules

Initially data is taken from PC to UART module at transmission serial line ‘rx’. It is T13 pin number in Spartan 3 FPGA. It will take data at a frame size of 8 bits at the rate of ‘clock_d’. UART module gives each frame at this rate to store unit. Store Unit is the small portion in main module of

Receiver. Due to memory and resources limitation of FPGA we can take only 32 bits for each user. F12 pin of FPGA has been used here as enabling pin of system so when we presses this switch then whole system will starts to work. 'Enable' signal is given at all modules. But before pressing 'enable' switch we must need to store data of transmitter output to store unit. Through Frequency Divider, provides the frequencies to different blocks according to their requirements. So at start of system Store Units gives a bit at a rate of 'clock_d' to the Decoder. This performs decoding of 384 bits into 3072 bits and also gives input bit at 'clock_d' pulse to De Walsh. De Walsh separates two user data at the frequency of 'clock_c'. In addition to this when De Walsh output is ready then it enables PN Code for PN operation. After that XOR operation will be performed between De Walsh output and PN Code chip with the spreading rate of two. So, when PN Codes enables Block De Interleaver, then it starts work and finally data goes to Viterbi Decoder. Here in system each module will enable next module when its output is ready. After decoding at Viterbi Decoder we remove two zero bits after every four output bits, which have been added due to Viterbi Decoder, which starts its procedure from first state. Here this procedure is running in parallel for both users. At the end data is stored back to UART for both users (32 bits for each user) and by pressing 'transmit' switch L13 pin in Spartan 3 FPGA data from UART will go back to PC. And PC will show the resultant values at matlab with fread(s) command.

B. Wave Diagram

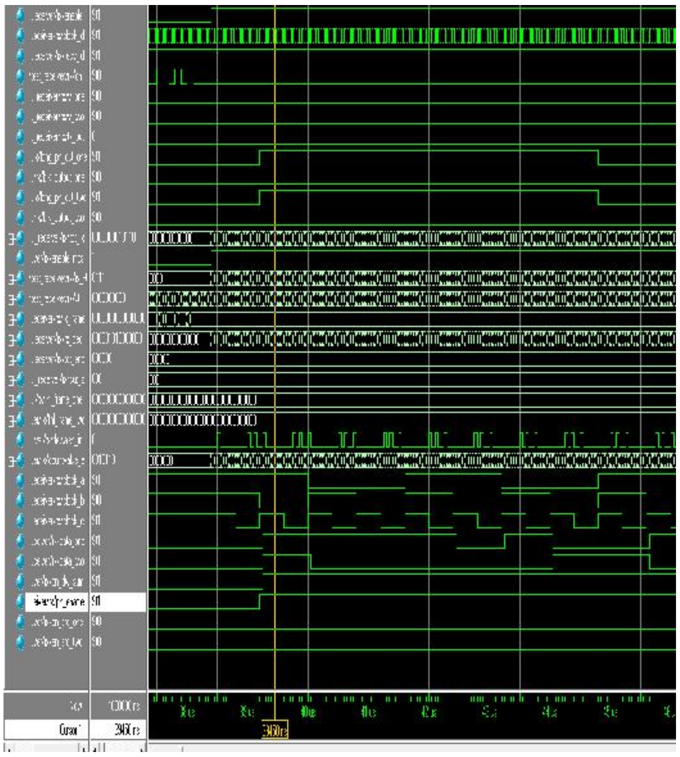

Fig 18 Wave Diagram of Receiver

From figure 18 we see that before De Walsh gives it first output bit it enables the PN Code for initial position ('pn_enable') at before one 'clock_d' pulse. And when De Walsh gives output signal for both users ('data_one' and 'data_two') then there is operation of XOR between individual bits of each user and PN Chip ('long_out_one', long_out_two').

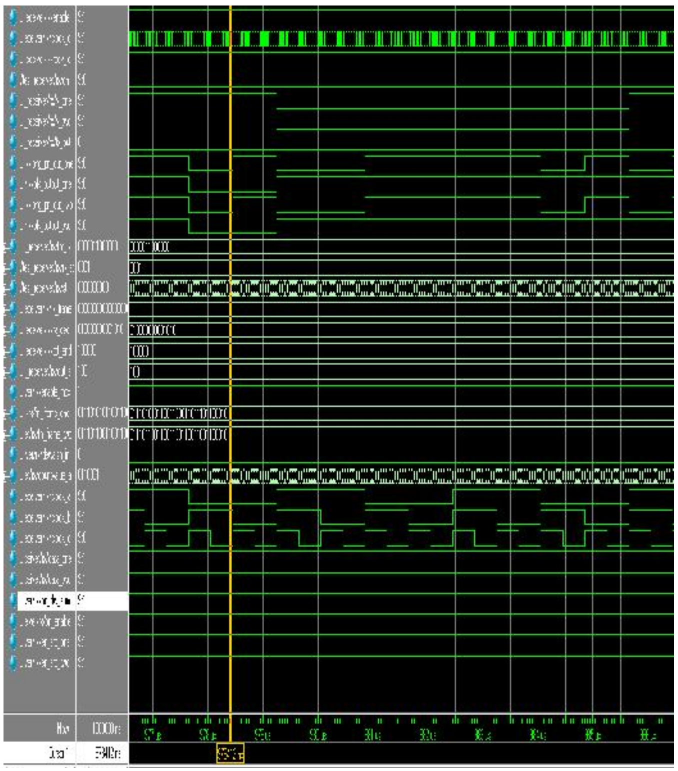

Fig 19 Wave Diagram of Receiver

Here in Fig. 19 'fnl_frame_one' shows the thirty two output bits of user one and 'fnl_frame_two' shows the thirty two output bits of user two. This is same sequence as it was given at the transmitter input for each user.

ACKNOWLEDGMENT

Our great reverence and humble thanks to Almighty Allah, who bestowed upon us enough vigor and urge to work and to finish the task undertaken by us. We are highly grateful to our supervisor Fahad Sharif, without whom this project would never have been transformed into reality. He gives us an opportunity for uninhibited work and their thoughts provoking feedback, instilled upon us new ideas.

CONCLUSION

CDMA is different approach in wireless communication systems. It has gained widespread international acceptance by cellular radio systems.

Its spread spectrum technology is more secure and less probable to intercept and jam. Moreover it is highly private and it also provides high transmission quality than TDMA. Clearly the CDMA is the next generation technology in terms of Voice and data transmissions over the AIR. Even though the cryptographic algorithms for CDMA have been broken, CDMA interception has a long way to go. This means that the CDMA transmissions will remain secure at least for few years from now.

REFERENCES

[1] Imamura, Kimihiko. "Wireless communication transmitter and receiver." U.S. Patent No. 7,251,288. 31 Jul. 2007,

[2] Michelson, Arnold M., and Allen H. Levesque. "Error-control techniques for digital communication." *New York, Wiley-Interscience, 1985, 483 p.* 1 (1985).

[3] Student, Mrs Kokila KS PG. "Physical Layer Implementation Of Orthogonal Frequency Division Multiplexing For Software Defined Radio On FPGA."

[4] Vucetic, Branka, and Savo Glisic. *Spread spectrum CDMA systems for wireless communications*. Artech House, Inc., 1997.

[5] Schilling, Donald L., Raymond L. Pickholtz, and Laurence B. Milstein. "Spread spectrum goes commercial." *Spectrum, IEEE* 27.8 (1990): 40-41.

[6] Lee, Kwang-Su. "Method for allocating Walsh codes by group in a CDMA cellular system." U.S. Patent No. 6,473,395. 29 Oct. 2002.

[7] Dinan, Esmael H., and Bijan Jabbari. "Spreading codes for direct sequence CDMA and wideband CDMA cellular networks." *Communications Magazine, IEEE* 36.9 (1998): 48-54.

[8] Embedded System Design: A Unified Hardware/Software Approach by Frank Vahid and Tony Givargis.

[9] Vanhaverbeke, Frederik, Marc Moeneclaey, and Hikmet Sari. "DS/CDMA with two sets of orthogonal spreading sequences and iterative detection."*Communications Letters, IEEE* 4.9 (2000): 289-291.

[10] Shao, Xiaoyin, and Dong Sun. "A FPGA-based motion control IC design."*Industrial Technology, 2005. ICIT 2005. IEEE International Conference on*. IEEE, 2005.

[11] Chouly, Antoine, Americo Brajal, and Sabine Jourdan. "Orthogonal multicarrier techniques applied to direct sequence spread spectrum CDMA systems."*Global Telecommunications Conference, 1993, including a Communications Theory Mini-Conference. Technical Program Conference Record, IEEE in Houston. GLOBECOM'93., IEEE*. IEEE, 1993.

[12] Ikeda, Yasunari, Tamotsu Ikeda, and Takahiro Okada. "Transmission of data by using convolutional coding of different code rates and the encoded data reception including decoding of the received data." U.S. Patent No. 5,691,995. 25 Nov. 1997.

[13] Viterbi, Andrew J., and Nagabhushana T. Sindhushayana. "Soft decision output decoder for decoding convolutionally encoded codewords." U.S. Patent No. 5,933,462. 3 Aug. 1999.

[14] Astrachan, Paul M. "Convolutional encoder for use on an integrated circuit that performs multiple communication tasks." U.S. Patent No. 5,612,974. 18 Mar. 1997.

[15] Wisal, Nausheen. "Comparative Analysis of MIMO Simulation with Transmit and Receive Diversity." *International Journal of Technology and Research* 1.3 (2013).

[16] Khan, M. Umar, Imad Siraj, and Nadeem Javaid. "Analytical Evaluation of Proactive Routing Protocols with Route Stabilities under two Radio Propagation Models." COCORA 2013, The Third International Conference on Advances in Cognitive Radio. 2013.

[17] Ghauri, Sajjad Ahmed. "Priority based Bandwidth Allocation in Cognitive Radio Network using Cooperative Game Theory." *International Journal of Technology and Research* 1.3 (2013).