LDPC ENCODER FOR OFDM BASED TROPOSCATTERER MODEM

A Project Report

submitted by

VAMSI KRISHNA V

in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2013

THESIS CERTIFICATE

This is to certify that the report titled **LDPC ENCODER FOR OFDM TROPOSCATTERER MODEM**, submitted by **VAMSI KRISHNA V**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology**, is a bonafide record of the project work done by him under our supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr.Nitin Chandrachoodan
Project Guide
Associate Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 23th May 2013

ACKNOWLEDGEMENTS

I wish to express my deep sense of gratitude to my project guide Dr.Nitin Chandrachoodan, Associate Professor in Department of Electrical Engineering at IIT Madras, for his most valuable guidance, discussions, suggestions and encouragement, from the conception to the completion of this project.

My sincere gratitude to Dr.Andrew Thangaraj, Associate Professor in Department of Electrical Engineering for explaiing me the basics of LDPC and Dr. Radha Krishna Ganti, Assistant Professor in Department of Electrical Engineering for being supportive throughout. I would like to thank them for their help and guidance throughout the project.

I would also like to thank my professors and friends from my undergraduate studies while at Indian Institute of Technology, Madras. The preparation and experience I gained here were invaluable.

Finally, I would to like to thank my family for their unconditional love and moral support, which allowed me to achieve what I have and will.

ABSTRACT

KEYWORDS: LDPC, OFDM, Encoder, FPGA implementation

In this era of wireless technologies, OFDM is finding use in many applications such as 4G mobile communications, digital television, audio broadcasting and wireless networks. In OFDM applications that require reliable and highly efficient information transfer - LDPC, a linear error correcting code is gaining popularity.

In this project LDPC encoder to be used for OFDM troposcatterer modem has been developed. The LDPC encoder offers flexibility to choose the code rate, message size and Base matrix. This has been developed along with its peripheral buffers as a separate module to facilitate replacing it with Turbo or other parity check codes if need be.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

LDPC	Low-density parity check
OFDM	Orthogonal frequency-division multiplexing
AM	Amplitude modulatation
FM	Frequency modulation
DVB	Digital video broadcasting
QAM	Quadrature amplitude modulation
QPSK	Quadrature phase shift keying
LDPC	Low-density parity-check code
SISO	Single input and single output
MIMO	Multiple input and multiple output
BEC	Backward error correction

CHAPTER 1

Introduction

1.1 Wireless Communication Systems

The quest for making life comfortable has been instrumental in advancing human civilization. Communication services available at any time and place free people from the limitation of being attached to fixed devices.The world's first wireless telephone conversation occurred in 1880, when Alexander Graham Bell and Charles Sumner Tainter invented and patented the photophone, a telephone that conducted audio conversations wirelessly over modulated light beams (which are narrow projections of electromagnetic waves).

In that distant era, when utilities did not yet exist to provide electricity and lasers had not even been imagined in science fiction, there were no practical applications for their invention, which was highly limited by the availability of both sunlight and good weather. Since then wireless communication has taken huge strides in terms of technology and connectivity. Affordable wireless communication has now become a reality. At the end of 2012, there were 6.8 billion mobile subscriptions (1).

1.2 Digital Broadcasting systems

Digital audio and video broadcasting offers consumers high resolution and better quality programs. The inauguration of AM broadcast radio(as opposed to point-to-point communication) can be traced back to the early 1900s began when Canadian experimenter Reginald Fessenden, conducted the first experimental broadcast this and was used for small-scale voice and music broadcasts in the early years of its usage. AM was the dominant method of broadcasting during the first eighty years of the 20th century and is still used in the 21st century. Around the middle of twentieth century, FM radio programmes were available. These technologies, based on analog communication, brought news, music, drama and much more into our daily lives. To provide more

and better programmes, digital broadcasting techniques such as digital audio broadcasting(DAB) and digital video broadcasting(DVB), began to replace the traditional analog broadcasting technologies in the past several years.

Digital Audio Broadcasting DAB is among the first standards that used the OFDB technique. Based on OFDM, DAB has one distinct benefit: a single-frequency network(SFN). In a single frequency broadcasting network, one carrier frequency can be used for all transmitters to broadcast the same programme to the entire country without suffering from co-channel interference. On the other hand, in the FM system, only one out of approximately 15 possible frequencies can be used, resulting in a very inefficient reuse factor.

In the DAB system, it is not necessary to search for radio stations as is necessary for AM/FM. The programmes of all radio stations are integrated on so called multiplexes, which save on the maintenance cost of individual radio stations. In addition variable bandwidths can be assigned to each programme, fulfilling their respective demands for sound quality. Furthermore, the DAB system features better mobile reception quality thanks to the OFDM technique.

Digital Video Broadcasting DVB is a suite of internationally accepted open standards for digital television. DVB standards are maintained by the DVB Project, an international industry consortium with more than 270 members, and they are published by a Joint Technical Committee (JTC). Indian Government announced the discontinuation of analog signals in the four metropolitan cities of Mumbai, Delhi, Kolkata by March 31st 2012 and for tier II cities like Bangalore, etc. by March 31, 2013. Most houses are now required to use only Digital Set Top box to watch "cable tv" in several cities in India.India's Information and Broadcasting Ministry announced that the deadline for the shift from analogue to digital systems in areas across the country has been set as 31st March 2015(2).

As the DAB system, DVB technology also supports countrywide single frequency networks. In addition, DVB standards offer several modes of operation that are tailored for large scale SFN and high-mobility reception.

The DVB-T standard(for terrestrial transmission) can support a data rate of MPEG-2

high definition TV(HDTV), which is upto 31 Mbps. In DVB-H(for low power handheld terminals), high-speed IP services as an enhancement of mobile telecommunications are offered. DVB standard has allowed for integration with bi-direction data connections through other access technology, thus enabling interactive applications between the viewers and the TV stations.

CHAPTER 2

Digital Modulation

2.1 Introduction

Digital modulation is the process of representing binary information using segments of different sinusoidal waveforms. The parameters that can be adjusted in a sinusoidal wave are it's amplitude, frequency and phase. In digital modulation, an analog carrier signal is modulated by a discrete signal. Digital modulation methods can be considered as digital-to-analog conversion, and the corresponding demodulation or detection as analog-to-digital conversion. The changes in the carrier signal are chosen from a finite number of alternative symbols.

The most fundamental digital modulation techniques are based on keying:

- PSK: a finite number of phases are used
- FSK: a finite number of frequencies are used
- ASK: a finite number of amplitudes are used
- QAM: a finite number of at least two phases and at least two amplitudes are used

2.2 Single-carrier Modulation

Single-carrier modulation techniques use only one sinusoidal wave at all times, while in the multi-carrier modulation techniques, several sinusoidal waves are transmitted simulta- neously. Basic single-carrier modulation techniques modify only one of the three parametersâĂŞamplitude, frequency and phaseâĂŞof the sinusoidal wave according to the binary information to be transmitted.The basic time unit in digital modulation techniques is a symbol, which is composed of a segment of the sinusoidal waveform. If there are only two possible different symbols in a digital modulation, then it is called a binary modulation. Figure 2.1 depicts sample waveforms of binary ASK, binary FSK and binary PSK.

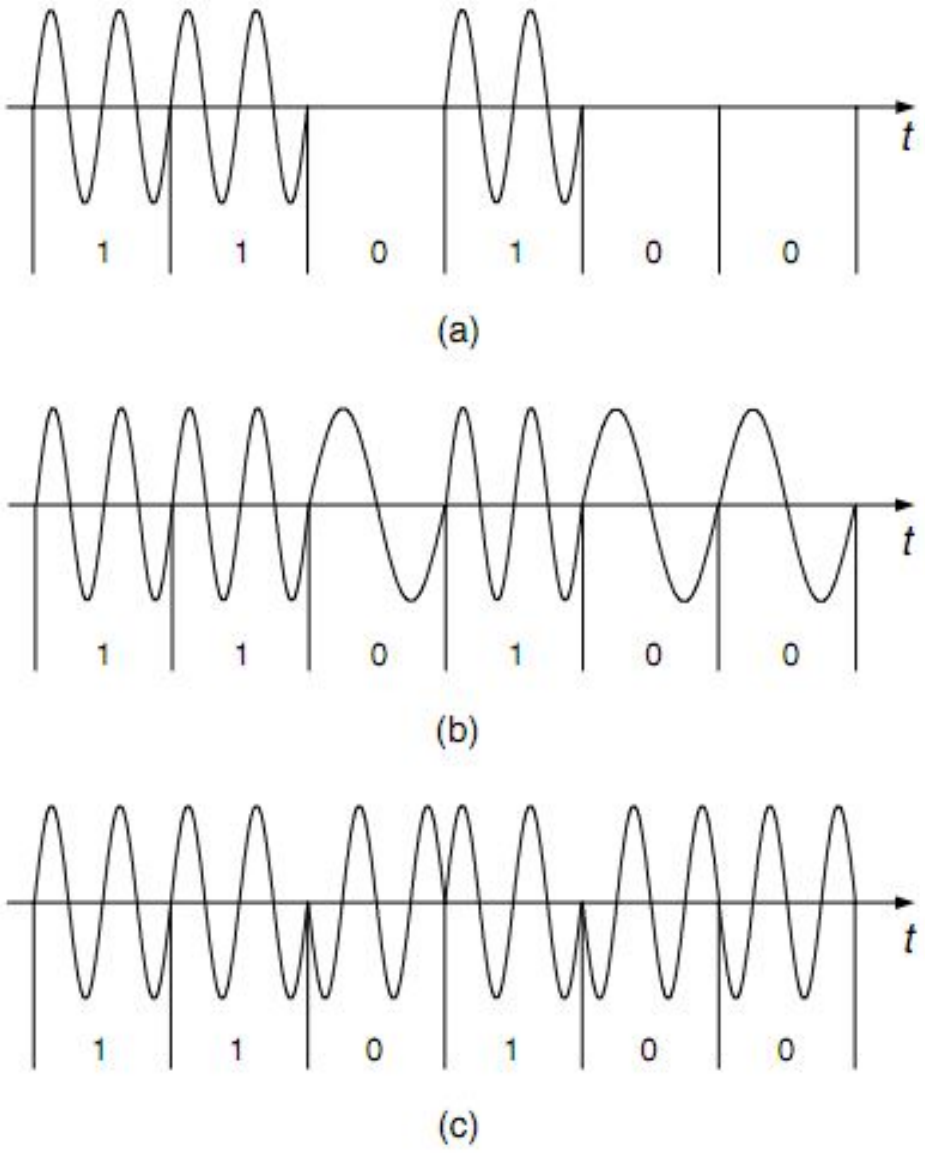

Figure 2.1: Waveforms of (a)ASK (b)FSK and (c)PSK

ASK and PSK are linear modulation, whose symbol waveforms are sinusoidal waveforms of the same frequency. A clearer representation of all possible symbols in such modulation is drawn by the phasor representations of all the possible symbols on a phasor plane. This representation is called the signal constellation of a digital modulation. The signal constellation of the binary ASK is illustrated in figure 2.2.

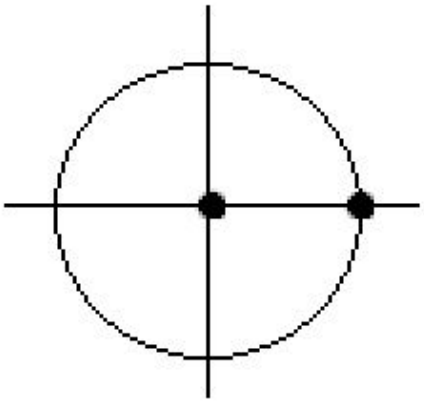

Figure 2.2: Signal constellation of binary ASK

With the signal constellation representation, higher-order modulation techniques that have a large number of possible symbol waveforms can be clearly described. For instance, M-ary PSK has M possible symbol waveforms with different phases, and carries log M bits per symbol. Figure 2.3 shows the signal constellation of binary PSK, quaternary PSK (QPSK) and 8PSK.

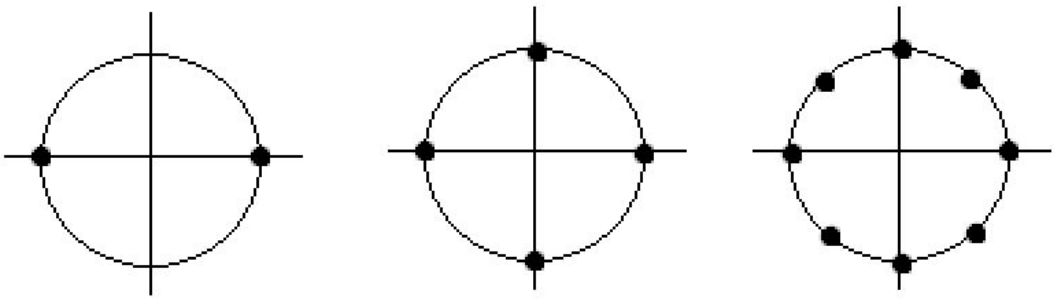

Figure 2.3: Signal constellation of (a) BPSK (b)QPSK and (c) 8PSK

More than one parameter in a segment of sinusoidal wave can also be changed in advanced digital modulation techniques. For example, in quadrature amplitude modulation(QAM), both amplitude and phase of the sine wave are changed. Figure 2.4 shows signal constellations of 16-QAM and 32-QAM.

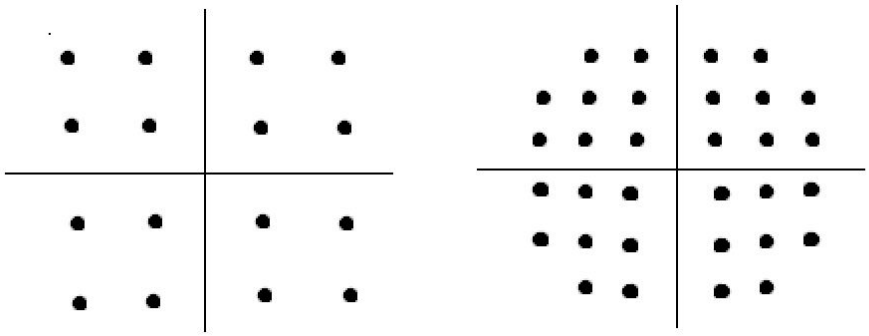

Figure 2.4: Signal constellation of (a) 16-QAM and (b) 32-QAM

2.3 Multi-Carrier Modulation

Wireless signals to be transmitted over the air usually suffer frequency-selective fading, namely different frequency components are faded quite differently by the channel.In conventional single-carrier systems, complex equalization schemes are adopted to combat frequency-selective fading. The ideal equalizer has a frequency response that is the exact inverse of that of the channel. This usually entails an inïňĄnite number of equalizer taps.The ïňĄrst proposal to use parallel data transmission to combat frequency-selective fading channels was published around 1967(3). In that system, only a small number of sub-channels use carriers that fall within each deep-faded frequency band. With the help of error-correcting codes, data along those corrupted sub-channels can be recovered. Thus, error-correcting codes are indispensable in all multi-carrier systems.

2.4 OFDM

Orthogonal frequency-division multiplexing(OFDM) is a method of encoding digital data on multiple carrier frequencies.It is a frequency-division multiplexing scheme used as a digital multi-carrier modulation method. A large number of closely spaced orthogonal sub-carrier signals are used to carry data on several parallel data streams or channels. Each sub-carrier is modulated with a conventional modulation scheme (such as quadrature amplitude modulation or phase-shift keying) at a low symbol rate, maintaining total data rates similar to conventional single-carrier modulation schemes in the

same bandwidth.

The primary advantage of OFDM over single-carrier schemes is its ability to cope with severe channel conditions (for example, attenuation of high frequencies in a long copper wire, narrowband interference and frequency-selective fading due to multipath) without complex equalization filters. Channel equalization is simplified because OFDM may be viewed as using many slowly modulated narrowband signals rather than one rapidly modulated wideband signal. The low symbol rate makes the use of a guard interval between symbols affordable, making it possible to eliminate intersymbol interference (ISI) and utilize echoes and time-spreading (on analogue TV these are visible as ghosting and blurring, respectively) to achieve a diversity gain, i.e. a signal-to-noise ratio improvement.

2.4.1 OFDM Transmitter used in the implementation

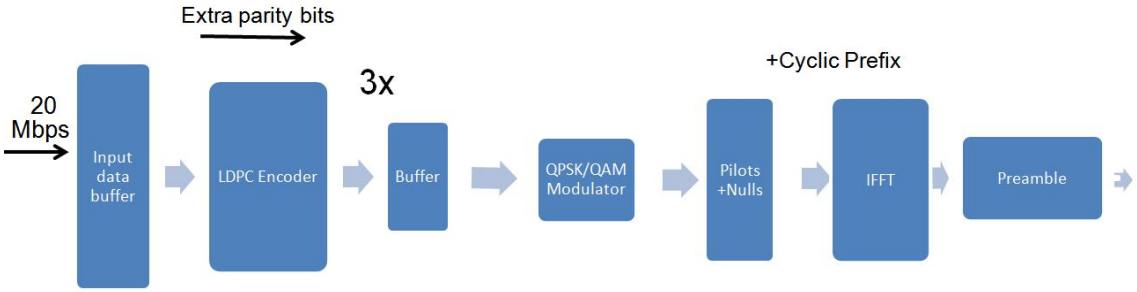

Figure 2.5: Transmitter Block diagram used in the implementation

LDPC encoder: LDPC is forward error correction code used to achieve reliable and efficient information transfer over bandwidth or return channelâĂŞconstrained links in the presence of data-corrupting noise. The encoder

QPSK mapper: It takes the encoded message from the LDPC and gives a corresponding complex mapped sample.

Pilot/Null Adder: Pilot signals for measurement of the channel conditions(i.e., the equalizer gain and phase shift for each sub-carrier). Pilot signals are also used for time synchronization (to avoid intersymbol interference, ISI) and frequency synchronization (to avoid inter-carrier interference, ICI, caused by Doppler shift). The pilots/nulls are

added at specific locations in a symbol by the module.

IFFT:An inverse FFT is computed on each set of symbols, giving a set of complex time-domain samples. Cyclic prefix is added to the symbol in the IFFT block.

Preamble Adder: A preamble/training symbol(64 bits) is added at the beginning of frame. It is used to train the VCO of the receiver to the incoming signals's clock so as to produce a clocking in the reciver that is synchronized to the received signal, and so a perfect sampling and/or demodulation can be done.

2.4.2 Frame configuration

The transmitter transmits data as frames. Each frame consists of 9 symbols and a preamble. A symbol consists of input samples(along with parity), pilots, nulls (pilots and nulls are added to fixed locations within a symbol) and a cyclic prefix.

Preamble : 64 samples/frame

Cyclic Prefix : 32 samples/symbol

Data + Pilots + Nulls= 384 + 46 + 82= 512

Total data length in a frame = 384x9

Frame size = (512+32x9) + 64 samples

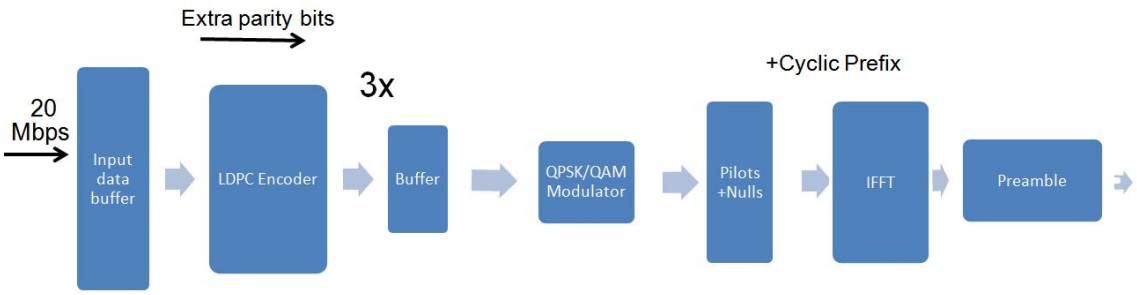

Figure 2.6: Frame configuration

CHAPTER 3

LDPC

3.1 Introduction

Low-density parity-check code is a linear error correcting code, a method of transmitting a message over a noisy transmission channel, and is constructed using a sparse bipartite graph.A sparse parity check matrix has relatively few number of 1s as compared to the number of 0s. LDPC codes have very good error correcting capability. The data transmission rate is close to Shannon limit. LDPC codes are capacity-approaching codes, which means that practical constructions exist that allow the noise threshold to be set very close (or even arbitrarily close on the BEC) to the theoretical maximum (the Shannon limit) for a symmetric memory-less channel and hence they allow higher code rates when compared to other error correcting codes such as Reed-solomon and Turbo.

The feature of LDPC codes to perform near the Shannon limit of a channel exists only for large block lengths. For example there have been simulations that perform within 0.04 dB of the Shannon limit at a bit error rate of 10^6 with a block length of 10^7. Those high performance codes are irregular. The large block length results also in large parity-check and generator matrices. The complexity of multiplying a codeword with a matrix depends on the amount of $1'$s in the matrix.

The noise threshold defines an upper bound for the channel noise, up to which the probability of lost information can be made as small as desired. Using iterative propagation techniques, LDPC codes can be decoded in time linear to their block length.

3.2 Code Description

The LDPC code is based on a set of one or more fundamental LDPC codes. Each of the fundamental codes is a systematic linear block code.Making changes to the Code Rate and Block Size in the fundamental codes can accommodate various code rates and

packet sizes.In telecommunication and information theory, the code rate (or information rate) of an error correction code is defined as the proportion of the data-stream that is useful (non-redundant). An LDPC code is a linear block code represented by a sparse parity check matrix **H** having n columns and m rows. The length of codeword is n bits and the message length is k = n - m bits where m is number of parity bits. Code rate is $r = \frac{k}{n}$. In other words, if the code rate is $r = \frac{k}{n}$, for every k bits of useful information, the coder generates totally n bits of data, of which n-k are redundant. The matrix **H** is given by:

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \cdots & \mathbf{P}_{0,n_b-2} & \mathbf{P}_{0,n_b-1} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \cdots & \mathbf{P}_{1,n_b-2} & \mathbf{P}_{1,n_b-1} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \cdots & \mathbf{P}_{2,n_b-2} & \mathbf{P}_{0,n_b-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \mathbf{P}_{m_b-1,0} & \mathbf{P}_{m_b-1,1} & \mathbf{P}_{m_b-1,2} & \cdots & \mathbf{P}_{m_b-1,n_b-2} & \mathbf{P}_{m_b-1,n_b-1} \end{bmatrix} = \mathbf{P}^{H_b}$$

where $\mathbf{P}_{i,j}$ is one of a set of z-by-z permutation matrices or a z-by-z zero matrix.

The matrix **H** is expanded from a binary base matrix $\mathbf{H_b}$ of size $m_b - by - n_b$, where n = z.n_b and n = z.m_b, with z an integer greater than or equal to 1. The base matrix is expanded by replacing each 1 in the base matrix with a z-by-z permutation matrix, and each 0 with a z-by-z zero matrix. The base matrix size n is an integer equal to 24.The permutations used are right shifts, and the set of permutation matrices contains the *zxz* identity matrix and circular right shifted versions of the identity matrix. Because each permutation matrix is specified by a single circular right shift, the binary base matrix information and permutation replacement information can be combined into a single compact model matrix $\mathbf{H_{bm}}$. The model matrix $\mathbf{H_{bm}}$ is the same size as the binary base matrix $\mathbf{H_b}$, with each binary entry(i,j) of the base matrix $\mathbf{H_b}$ replaced to create the model matrix $\mathbf{H_{bm}}$. Each 0 in $\mathbf{H_b}$ is replaced by a blank or negative value (e.g., by -1) to denote a *zxz* all-zero matrix, and each 1 in $\mathbf{H_b}$ is replaced by a circular shift size p (i,j)$\geq$0. The model matrix $\mathbf{H_{bm}}$ can then be directly expanded to **H**.

Output Packet Size, n(bits)	Output Packet Size, n(bytes)	Block Size (bits)	Input Packet size, k (bytes)			
			Rate= 1/2	Rate= 2/3(A/B)	Rate= 3/4(A/B)	Rate = 5/6
576	72	24	36	48	54	60
672	84	28	42	56	63	70
768	96	32	48	64	72	80
864	108	36	54	72	81	90
960	120	40	60	80	90	100
1056	132	44	66	88	99	110
1152	144	48	72	96	108	120
1248	156	52	78	104	117	130
1344	168	56	84	112	126	140
1440	180	60	90	120	135	150
1536	192	64	96	128	144	160
1632	204	68	102	136	153	170
1728	216	72	108	144	162	180
1824	228	76	114	152	171	190
1920	240	80	120	160	180	200
2016	252	84	126	168	189	210
2112	264	88	132	176	198	220
2208	276	92	138	184	207	230
2304	288	96	144	192	216	240

Figure 3.1: LDPC block size for various code rates

There are six different LDPC code rates (1/2, 2/3A, 2/3B, 3/4A, 3/4B and 5/6) defined in IEEE P802.16e(draft)(4)., and shown in the above Fig. The standard also defines the Block Size for each code. The Block Size controls the length of the output packet, and the code rate determines the ratio of the input packet length to the output packet length. Each encoding solution is defined by selecting one of the Code Rates and an associated Block Size.

Each of the six LDPC codes in the standard is defined by the matrix H of size m-by-n, defined above..

3.3 Encoding

An (n,k) LDPC code with an (n-k)xn parity check matrix $\mathbf{H}$ is called regular if the number of 1's in each of the n-k rows is always w_r and the number of $1'$s in each of the n-k rows is always w_c. w_r(n$-$k) = w_cn nad the density of the matrix is w_r/n. All LDPC codes have relatively small w_r when compared to the block size. In most of the LPDPC codes, the block size is greater than 1000(we will be using 2304) and w_r and w_c are kept less than 10. $\mathbf{H}$ can be considered to be of the form $\mathbf{H}$ =[$\mathbf{A}_1^T$ $\mathbf{A}_2^T$], where $\mathbf{A}_1$ is a kx(n-k) matrix and $\mathbf{A}_2$ is a (n-k)(n-k) matrix. Then the correseponding generator matrix of LDPC code is given by : [$\mathbf{I}_{kxk}\mathbf{A}_1{\mathbf{A}_2}^{-1}$]

CHAPTER 4

Hardware Implementation

4.1 Introduction

The LDPC and the surrounding buffers are implemented on a Virtex-5 FPGA.

Family	Virtex 5
Device	XC5VTX150T
Package	FF1156
Speed grade	-1
Size(mmxmm)	35x35
Max. Available I/Os	360
Gigabit transceivers	40 GTXs

Figure 4.1: Device Properties

The following parameters have been used for the LDPC. These values are flexible and can be changed by modifying the *parametes.v* file.

Parameter	**Value**
Z factor	96
Code Rate	2/3
Size of encoded message	2304 bits
Base Matrix	Fig. below

Figure 4.2: Parameters

$$
\begin{pmatrix}
2 & -1 & 19 & -1 & 47 & -1 & 48 & -1 & 36 & -1 & 82 & -1 & 47 & -1 & 15 & -1 & 95 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & 69 & -1 & 88 & -1 & 33 & -1 & 3 & -1 & 16 & -1 & 37 & -1 & 40 & -1 & 48 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 \\
10 & -1 & 86 & -1 & 62 & -1 & 28 & -1 & 85 & -1 & 16 & -1 & 34 & -1 & 73 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 \\
-1 & 28 & -1 & 32 & -1 & 81 & -1 & 27 & -1 & 88 & -1 & 5 & -1 & 56 & -1 & 37 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\
23 & -1 & 29 & -1 & 15 & -1 & 30 & -1 & 66 & -1 & 24 & -1 & 50 & -1 & 62 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 \\
-1 & 30 & -1 & 65 & -1 & 54 & -1 & 14 & -1 & 0 & -1 & 30 & -1 & 74 & -1 & 0 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 \\
32 & -1 & 0 & -1 & 15 & -1 & 56 & -1 & 85 & -1 & 5 & -1 & 6 & -1 & 52 & -1 & 0 & -1 & -1 & -1 & -1 & -1 & 0 & 0 \\
-1 & 0 & -1 & 47 & -1 & 13 & -1 & 61 & -1 & 84 & -1 & 55 & -1 & 78 & -1 & 41 & 95 & -1 & -1 & -1 & -1 & -1 & -1 & 0
\end{pmatrix}
$$

Figure 4.3: Base matrix P

Xilinx ISE Design suite 14.3 has been used for the implementation. Xilinx Synthesis Technology(XST), a Xilinx application that synthesizes HDL designs to create Xilinx specific netlist files called NGC files is used for synthesis.The NGC file is a netlist that contains both logical design data and constraints. ISIM, an HDL simulator integrated within ISE is used for behavioral, post-map and post-route simulations. HDL language Verilog is used for the hardware implementation of the blocks.

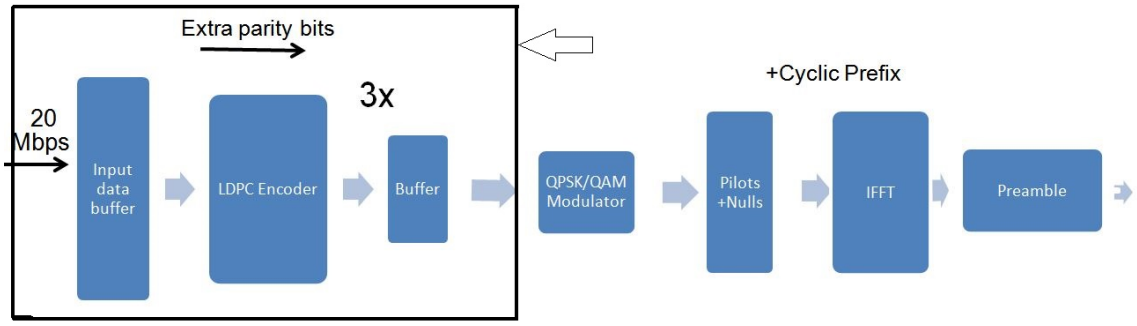

Figure 4.4: Transmitter Block

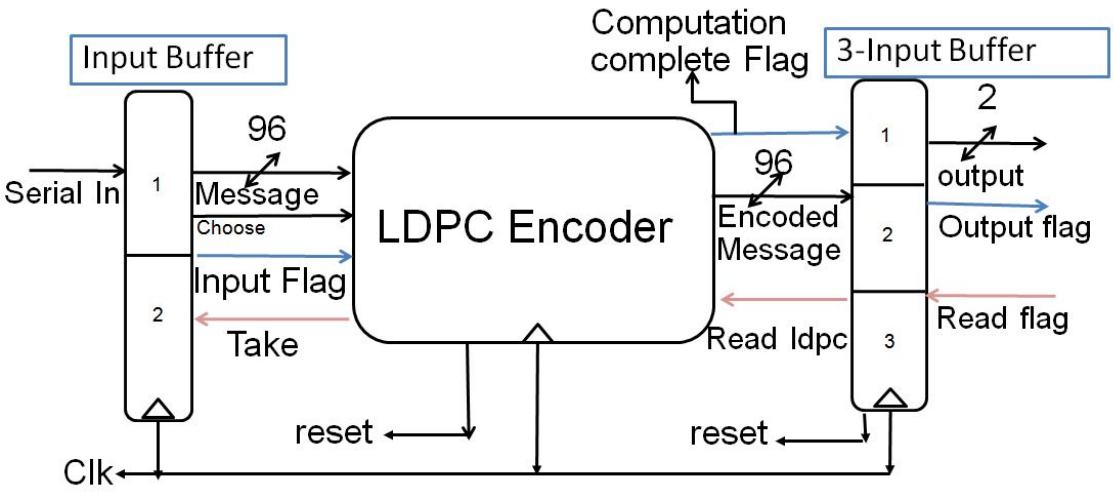

Figure 4.5: LDPC with serial input

4.2 Modules

4.2.1 Input buffer

It is a buffer to store the serial input data bits(20Mbps) before being sent to the LDPC module.

Inputs :

Reset: A common master reset which is the same for all blocks.

Clk: A clock which operates at 20MHz - clock rate is equal to the input bit rate which is 20Mbps for our transmitter.

Clock: A clock which operates at 25MHz - This is the common clock is used for the entire transmitter block. This must always be faster than *Clk*.

in: It is the serial input data given to the transimitter at 20MHz. For simulation purposes,input is given through a pseudo-random number generator block.

take: It is sent by the LDPC module when it is ready to take data. Data is read from the data only when *take* is high.

Outputs :

Message: 96-bit parallel data sent from the buffer to the LDPC module.

Choose - Indicates the part of the buffer being filled.

input_flag - Stays high when message is being transmitted.

Functioning :

Input serial data *in* is written into the buffer at 20MHz. The buffer consists of two 1536 bit blocks which are stacked individually as 96x16 registers. At master reset, the contents of the buffer are initialized to zero and data is written once the reset goes to zero. The data is a continuous stream and there can't be a lag while data is being read from the buffer. Hence two(96x16) buffers are chosen to ensure continuity of data. As data is filled in one half of the buffer, couters- count_row and count_column are incremented. When the first buffer gets filled, *choose* goes high and when second buffer gets filled *choose* goes low. In other words, if the first buffer is being filled *choose* stays low and when second buffer is being filled *choose* stays high.

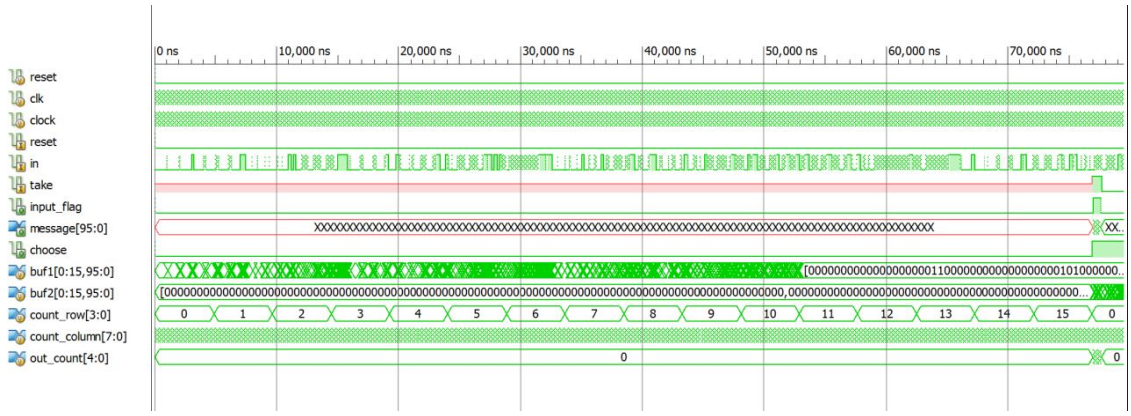

Figure 4.6: Overview of the Input buffer simulation

When *choose* goes high , the LDPC sends a *take* signal if it is ready to accept data. On recieving *take*, input buffer sets *input_flag* high and message bits are transmitted. When *out_counter*, which indicates the 96-bit register being read equals 16(implies all the message bits in one half(1536) have been read) the *input_flag* goes low and the cycle continues.

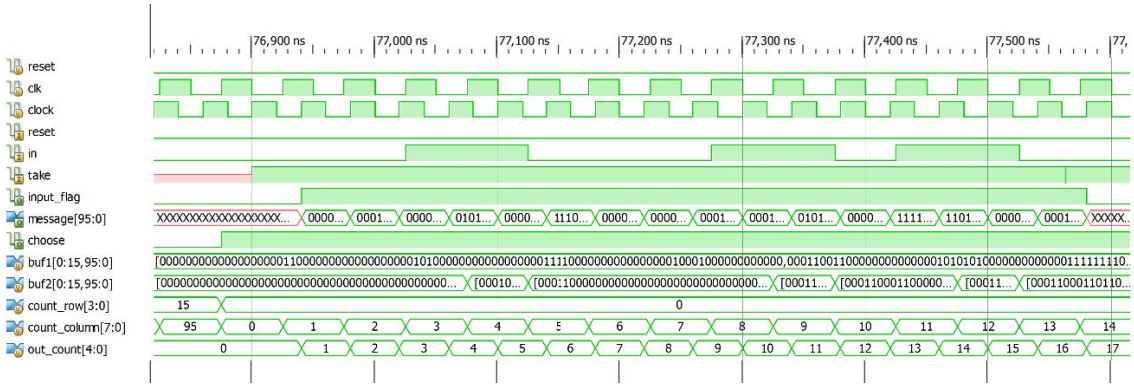

Figure 4.7: Input buffer after *take* is high

4.2.2 LDPC

LDPC takes in parallel FIFO message bits and gives encoded message as output.

Inputs :

reset : Master reset

clock : 25 Mhz clock used throughout the transmitter.

message: 96-bit parallel FIFO message bits recieved from the *input buffer*.

input_flag : This flag stays high when *message* is being transmitted.

choose:Indicates the part of the input buffer being filled, in other words the part of the input buffer to be read.

read_ldpc : Part of the handshake mechanism between the LDPC and 3-input buffer block. Indicates that the buffer is ready to accept data from the LDPC.

Outputs :

take: Indicates that it is ready to take in messge bits from the input buffer.

computation_complete_flag: It is a flag bit which indicates completion of the encoding of 1 block of data(1536 message bits + 768 parity bits for a code rate of 2/3).

par_out: A 96-bit parallel output for sending the encoded message(Message bits

plus parity bits)

Functioning of LDPC :

The elements of base matrix P (lie between -1 and z-1) are first hard-coded in a ROM using readmemh. Note: All elements of the must be converted to hexadecimal before reading from the external file.

When the value of choose changes, i.e.,sufficient data required for 1 block of data(1536 input bits) is available and the LDPC is ready to take in data, it sets *take* high. Handshake mechanism ensues and input message which is of the form of 96 bit parallel data is read from the input buffer. Once the message data has been written into the LDPC, *take* goes low. The parity bits are computed using the algorithm in Chapter 3. It takes 24 clock cycles once the input_flag goes low to compute the parity bits.

As soon as the 768 parity bits are calculated, computation_complete_flag is set high. It waits for read_ldpc signal to transmit data to 3-input buffer. Once read_ldpc is high, the encoded message(2304 bits) is transmitted as 96-bit output.

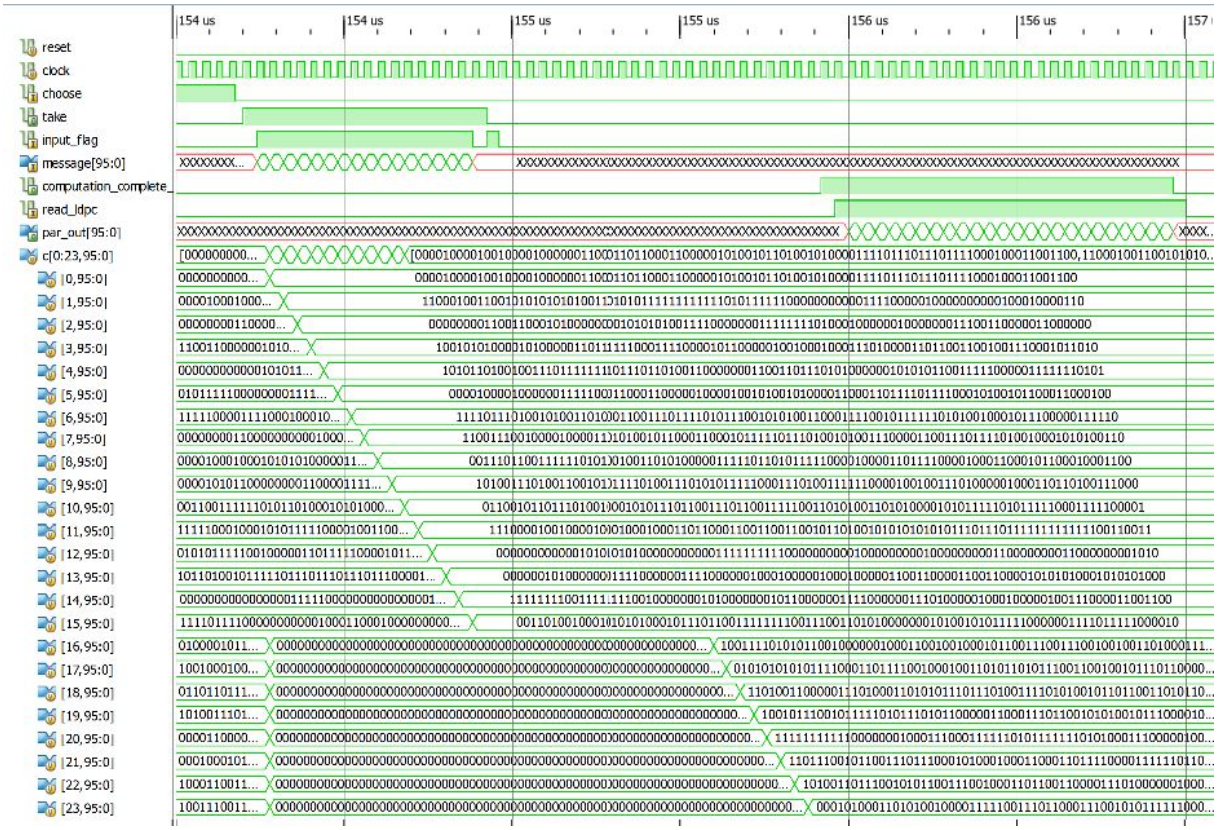

Figure 4.8: Input buffer after *take* is high

Parity bits computation : The input 1536 bits are stored in 96-bit registers. For each row of parity bits(8 in all), it is calculated by *xor*ing the row with the input message row circular-shifted by the value of P(i,j) if P(i,j) is ≥ 0.

```
for (j=0; j<ROWS_P; j=j+1 ) begin
if (P[i+j*COLS_P][7]==0) begin //if (P[i+j*COLS_P]>=0) begin
temp1[j] = (c[i]<<P[i+j*COLS_P]);
temp2[j] = (c[i]>>(Z-P[i+j*COLS_P]));
circ_shifted_c[j] = temp1[j] | temp2[j];
par[j] = ( par[j] ^ circ_shifted_c[j]);
end
end

//Not the entire algorithm - only the part which involves
circular shift and xor.
```

Circular shift is implemented by doing an *or* operation of two registers, one obtained by left-shifting it by P(i,j) and the other obtained by right shifting the register by Z - P(i,j) bits.

4.2.3 3-Input Buffer

3-input buffer stores 3 blocks of encoded message (2304x3 bits $\Leftrightarrow$ 9 symbols $\Leftrightarrow$ 1 frame). It takes in encoded message from the LDPC and stores it till data for one entire frame is available. It then sends 2-bit output to the QPSK mapper module.

Inputs :

reset : Master reset

clock : 25 Mhz clock used throughout the transmitter.

computation_complete_flag: It is a flag bit which indicates completion of the encoding of 1 block of data(1536 message bits + 768 parity bits for a code rate of 2/3).

par_out: A 96-bit parallel output for sending the encoded message(Message bits plus parity bits)

read_flag:Data is read from the buffer when read_ldpc is high and stops when it goes low.

Outputs :

out : 2-bit output to the QPSK mapper block.

output_flag : Is high when data required for an entire frame is ready to be read.

read_ldpc : Part of the handshake mechanism between the LDPC and 3-input buffer block. Indicates that the buffer is ready to accept data from the LDPC.

Functioning At master reset, all the registers in the buffer are set to 0. While the first frame is being filled, the output_flag is set low till the data is being written into the buffer. When entire (72x16) registers fill up output_flag goes high. Data is read when read_flag is set high. It is usually high for 15360ns(1 symbol $\Leftrightarrow$ 384 samples $\Leftrightarrow$ 768 bits). Data is not written into the buffer while data from part(i) of the 3-Input buffer is being read. When data has been read from the part(i) (read_counter $\geq$ 24 $\Leftrightarrow$ 3 symbols) data can be filled in part(i). When data is being read from part(iii)(read_counter $\geq$ 48) data can be filled in part(ii) of the buffer. When the reading is complete i.e., read_counter = 72 output_flag goes low again till the part (iii) is filled up. The cycle continues till master reset is pressed.

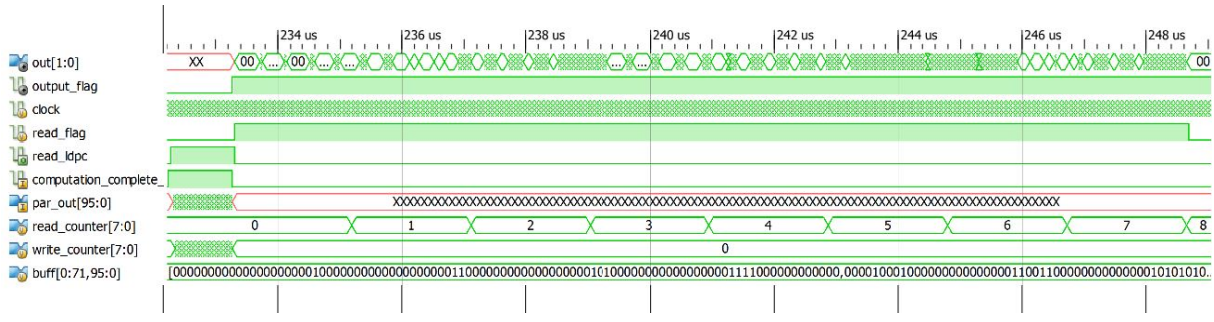

Figure 4.9: 3-Input Buffer behavior

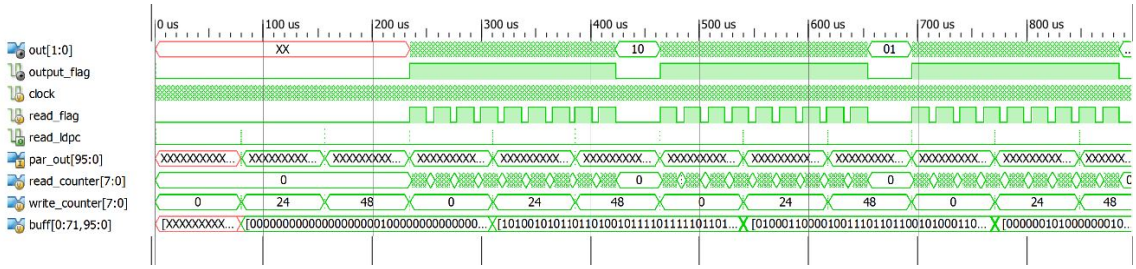

Figure 4.10: 3-Input Buffer for 1 complete frame

CHAPTER 5

Conclusion

5.1 Salient features of the design

:

- Flexibility to choose the code rate and message size as they are defined as parameters
- Designed as an independent module to facilitate replacement with turbo codes if required. Only the flag bits need to be synchronized.
- Base matrix P is hard-coded and can be changed directly from the code.
- Handshake mechanism for communicating with other modules.
- Encoding is done in minimum time possible(24 clock cycles). This design involves using 16 Logical Shifters and 8 xors(96-bit) to be used in parallel. Even though hardware intensive it is a good trade off as it reduces the no. of intermediate memory elements by a huge amount and hence the area.
- The LDPC module and the buffers have been paramaterized to make the design flexible. Designed as an independent module to facilitate replacement with turbo codes if required. Only the flag bits need to be synchronized.

5.2 Design Summary

Mapping Report

```
Slice Logic Utilization:
  Number of Slice Registers:      10,243 out of  92,800   11%
    Number used as Flip Flops:    10,243
  Number of Slice LUTs:           18,796 out of  92,800   20%
    Number used as logic:         18,334 out of  92,800   19%
      Number using O6 output only:18,042
      Number using O5 output only:      162
```

```
    Number using O5 and O6:          130
   Number used as exclusive route-thru:462
  Number of route-thrus:             647
    Number using O6 output only:     624
    Number using O5 output only:      23

Slice Logic Distribution:
  Number of occupied Slices:          7,908 out of  23,200   34%
  Number of LUT Flip Flop pairs used:24,936
    Number with an unused Flip Flop: 14,693 out of  24,936   58%
    Number with an unused LUT:        6,140 out of  24,936   24%
    Number of fully used LUT-FF pairs:4,103 out of  24,936   16%
    Number of unique control sets:     122
    Number of slice register sites lost
      to control set restrictions:      21 out of  92,800    1%
```

5.3 Place and route summary

```
Device Utilization Summary:

   Number of BUFGs                    2 out of 32      6%
   Number of External IOBs            7 out of 360     1%
      Number of LOCed IOBs            0 out of 7       0%

   Number of Slices                7908 out of 23200  34%
   Number of Slice Registers      10243 out of 92800  11%
      Number used as Flip Flops   10243
      Number used as Latches          0
      Number used as LatchThrus       0

   Number of Slice LUTS           18796 out of 92800  20%
   Number of Slice LUT-Flip Flop pairs 24936 out of 92800  26%
```

5.3.1 Simulation

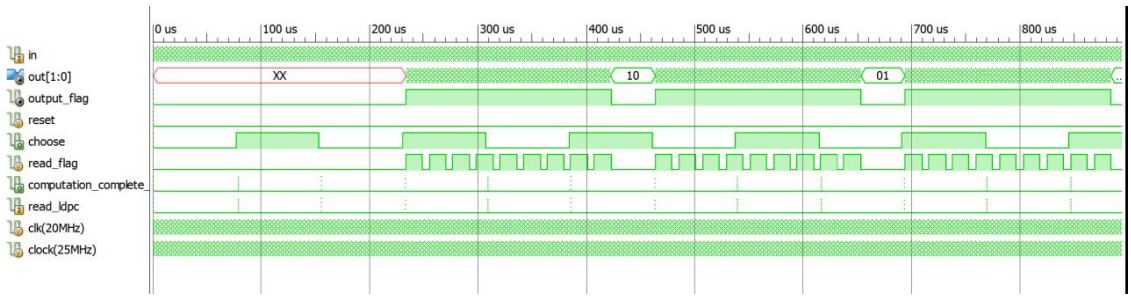

Figure 5.1: Overall simulation

REFERENCES

[1] Subscriber details *http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2013.pdf*.

[2] Fully digital by 2015 *http://www.dvb.org/about_dvb/dvb_worldwide/india/index.xml*.

[3] First multi-carrier *3GPP, Deployment Aspects, TR 25.943, v2.1.0, Jun. 2001*.

[4] *IEEE P802.16e/D8, May 2005 - Section 8.4.9.2.5.1*

[8] *OFDM Baseband receiver design for Wireless Communications − Chiueh and Tsai*

[8] *http://en.wikipedia.org/wiki/Orthogonal_frequency-division_multiplexing*.

[8] *http://http://en.wikipedia.org/wiki/Low-density_parity-check_code*.

[8] *Ramachandran, P. (2004). LATEX class for dissertations submitted to IIT-M . Ph.D. thesis, Department of Aerospace Engineering, IIT-Madras, Chennai âĂŞ 600036.*

APPENDIX A

Coding Matrix

The following are the base matrices(P) for various code rates.

Code rate = 1/2

-1	94	73	-1	-1	-1	-1	-1	55	83	-1	-1	7	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	27	-1	-1	-1	22	79	9	-1	-1	-1	12	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	24	22	81	-1	33	-1	-1	-1	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
61	-1	47	-1	-1	-1	-1	-1	65	25	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	39	-1	-1	-1	84	-1	-1	41	72	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	46	40	-1	82	-1	-1	-1	79	0	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	95	53	-1	-1	-1	-1	-1	14	18	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	11	73	-1	-1	-1	2	-1	-1	47	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
12	-1	-1	-1	83	24	-1	43	-1	-1	-1	51	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	-1	-1	-1	94	-1	59	-1	-1	70	72	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	7	65	-1	-1	-1	-1	39	49	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
43	-1	-1	-1	-1	66	-1	41	-1	-1	-1	26	7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

Code rate = 2/3A

3	0	-1	-1	2	0	-1	3	7	-1	1	1	-1	-1	-1	-1	1	0	-1	-1	-1	-1	-1	-1
-1	-1	1	-1	36	-1	-1	34	10	-1	-1	18	2	-1	3	0	-1	0	0	-1	-1	-1	-1	-1
-1	-1	12	2	-1	15	-1	40	-1	3	-1	15	-1	2	13	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	19	24	-1	3	0	-1	6	-1	17	-1	-1	-1	8	39	-1	-1	-1	0	0	-1	-1	-1
20	-1	6	-1	-1	10	29	-1	-1	28	-1	14	-1	38	-1	-1	0	-1	-1	-1	0	0	-1	-1
-1	-1	10	-1	28	20	-1	-1	8	-1	36	-1	9	-1	21	45	-1	-1	-1	-1	-1	0	0	-1
35	25	-1	37	-1	21	-1	-1	5	-1	-1	0	-1	4	20	-1	-1	-1	-1	-1	-1	-1	0	0
-1	6	6	-1	-1	-1	4	-1	14	30	-1	3	36	-1	14	-1	1	-1	-1	-1	-1	-1	-1	0

Code rate = 2/3B

2	-1	19	-1	47	-1	48	-1	36	-1	82	-1	47	-1	15	-1	95	0	-1	-1	-1	-1	-1	-1
-1	69	-1	88	-1	33	-1	3	-1	16	-1	37	-1	40	-1	48	-1	0	0	-1	-1	-1	-1	-1
10	-1	86	-1	62	-1	28	-1	85	-1	16	-1	34	-1	73	-1	-1	-1	0	0	-1	-1	-1	-1
-1	28	-1	32	-1	81	-1	27	-1	88	-1	5	-1	56	-1	37	-1	-1	-1	0	0	-1	-1	-1
23	-1	29	-1	15	-1	30	-1	66	-1	24	-1	50	-1	62	-1	-1	-1	-1	-1	0	0	-1	-1
-1	30	-1	65	-1	54	-1	14	-1	0	-1	30	-1	74	-1	0	-1	-1	-1	-1	-1	0	0	-1
32	-1	0	-1	15	-1	56	-1	85	-1	5	-1	6	-1	52	-1	0	-1	-1	-1	-1	-1	0	0
-1	0	-1	47	-1	13	-1	61	-1	84	-1	55	-1	78	-1	41	95	-1	-1	-1	-1	-1	-1	0

Code rate = 3/4A

6	38	3	93	-1	-1	-1	30	70	-1	86	-1	37	38	4	11	-1	46	48	0	-1	-1	-1	-1
62	94	19	84	-1	92	78	-1	15	-1	-1	92	-1	45	24	32	30	-1	-1	0	0	-1	-1	-1
71	-1	55	-1	12	66	45	79	-1	78	-1	-1	10	-1	22	55	70	82	-1	-1	0	0	-1	-1
38	61	-1	66	9	73	47	64	-1	39	61	43	-1	-1	-1	-1	95	32	0	-1	-1	0	0	-1
-1	-1	-1	-1	32	52	55	80	95	22	6	51	24	90	44	20	-1	-1	-1	-1	-1	-1	0	0
-1	63	31	88	20	-1	-1	-1	6	40	56	16	71	53	-1	-1	27	26	48	-1	-1	-1	-1	0

Code rate = 3/4B

-1	81	-1	28	-1	-1	14	25	17	-1	-1	85	29	52	78	95	22	92	0	0	-1	-1	-1	-1
42	-1	14	68	32	-1	-1	-1	-1	70	43	11	36	40	33	57	38	24	-1	0	0	-1	-1	-1
-1	-1	20	-1	-1	63	39	-1	70	67	-1	38	4	72	47	29	60	5	80	-1	0	0	-1	-1
64	2	-1	-1	63	-1	-1	3	51	-1	81	15	94	9	85	36	14	19	-1	-1	-1	0	0	-1
-1	53	60	80	-1	26	75	-1	-1	-1	-1	86	77	1	3	72	60	25	-1	-1	-1	-1	0	0
77	-1	-1	-1	15	28	-1	35	-1	72	30	68	85	84	26	64	11	89	0	-1	-1	-1	-1	0