NOTE TO USERS

Design and FPGA Implementation of Trellis Coded Modem

based on DVB Standard

Danial Nikfal

A thesis

in

The Department

of

Electrical and Computer Engineering

Present in Partial Fulfillment of Requirements
for the Degree of Master of Applied Science (Electrical Engineering) at
Concordia University
Montreal, Quebec, Canada

April 2005

Your file Votre référence
ISBN: 0-494-04387-3
Our file Notre référence
ISBN: 0-494-04387-3

ABSTRACT

Design and FPGA Implementation of Trellis Coded Modem

based on DVB Standard

Danial Nikfal

This thesis concentrates on the implementation of a base-band modem specified by Digital Video Broadcasting (DVB) standard. The modem consists of Reed Solomon coding, Convolutional Interleaving, Pragmatic Trellis Coded Modulation (PTCM), and Pulse shaping filters. The system is employed in digital television and related applications based on satellite communications.

A design flow using System On a Programmable Chip (SOPC) is a novel approach that can be considered in the next generation of wireless modems. The design flow starts with modeling of the floating-point representation of the design in Simulink with parameters described in the DVB standard. Fixed-point refinement of the model is developed to compromise arithmetic precision for hardware simplicity. A number of significant simulations are performed to verify adequate precision of the fixed-point model by comparing the bit error rate curves with their floating-point counterparts.

Bottom-up design flow is used to synthesize the modem into the target hardware. The high-level abstraction model of the design is simulated and compiled into an FPGA using Xilinx System Generator for DSP. A novel clock distribution technique is proposed to achieve the highest possible clock frequency. Finally, the modem is synthesized and implemented in Xilinx Virtex II FPGA and post-place&route simulation is performed to

ensure that the timing constraints are met. The implemented modem is able to transmit/receive digital audio and video signals up to 27.778 Mbit/s.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Dr. Yousef Shayan for his enthusiastic supervision, support and assistance in developing this thesis. He has been an invaluable source of knowledge and his constructive comments have helped to inspire many of the ideas in this thesis.

Also, I would like to thank Mr. Tadeusz Obuchowics for promptly answering my questions and for providing technical support for CAD tools used in Concordia University VLSI lab. I would also like to thank the rest of the faculty members of the Department of Electrical and Computer Engineering at the Concordia University. I also appreciate all my friends who patiently helped me and encouraged me throughout my studies. I am especially grateful to Mr. Afshin Nourivand.

Finally, I am forever indebted to my parents for their understanding, endless patience and encouragement when it was most required.

This Thesis is dedicated to my parents .....

Table of content:

List of Figures:

List of Tables:

Chapter one

1 Introduction

In this chapter motivation of Trellis Coded Modulation (TCM) is presented. Moreover, Pragmatic TCM is introduced as a more regular approach to implement the TCM codec. The history of Trellis Coded Modulation and its application is presented. In addition, the motivation and methodology of the research is stated. This chapter also presents the organization of the thesis.

1.1 Coding versus Modulation

Forward Error Correction (FEC) coding is a practical technique that virtually improves the efficiency of any digital communication system by adding redundant bits to the original message sequence. Communication systems can generally be divided into two major categories: power-limited and band-limited systems.

In power-limited systems, the desired performance should be achieved with the smallest possible transmit power. Hence, error correction codes are exploited to achieve the same performance with smaller transmit power. However, higher symbol rate and higher bandwidth are required to transmit the same information per unit of time.

On the other hand, in bandwidth-limited systems, the desired performance should be achieved in the available limited bandwidth. Therefore, higher-level modulation schemes can be used to increase bandwidth efficiency. In this case, larger transmitted power is required to maintain the same system bit-error-rate.

In wireless communications frequency spectrum is limited. Moreover, the more transmitted power a system requires, the more expensive the data transmission costs. Therefore, the aim of any digital communication is to achieve reliable data transmission with the lowest transmitted power within the limited available bandwidth. The increasing demand of higher bit-rate requires efficiently using the available bandwidth even in satellite communication where bandwidth is plentiful. Therefore, FEC became an inevitable part of bandwidth limited systems. The information rate loss, by adding redundant bits to the message sequence, can be compensated by using higher modulation schemes. However, when the modulation and FEC are performed in an independent manner, disappointing results are obtained.

Trellis coded modulation is introduced as a combined coding and modulation scheme that improves reliability of the transmission system without increasing the signal power or required bandwidth. Trellis-coded modulation is different from conventional coding techniques. It performs coding and modulation together. In other words, the coding process involves handling of mapping the codeword to a modulated signal as well.

Since Ungerboeck invented TCM in 1976 and had his papers published in the 1980's [4-6], numerous researches have been working on TCM applications in numerous areas such as: voice band modems, satellite communications, wireless communications trials, digital subscriber loop, HDTV (high definition television), broadcast channels, CATV (community antenna television) and DBS (direct broadcast satellite) in the 1980's and 1990's. Many innovations in TCM technology have been introduced, such as multidimensional TCM (1984-1985), rotationally invariant TCM with M-PSK (1988), TCM with built-in time diversity (1988-1990), TCM with Tomlinson pre-coder (1990-

1991), TCM with unequal error protection (1990), multilevel coding with TCM (1992-1993), and concatenated coding with TCM (1993-present). TCM evolution is depicted in Figure 1-1.

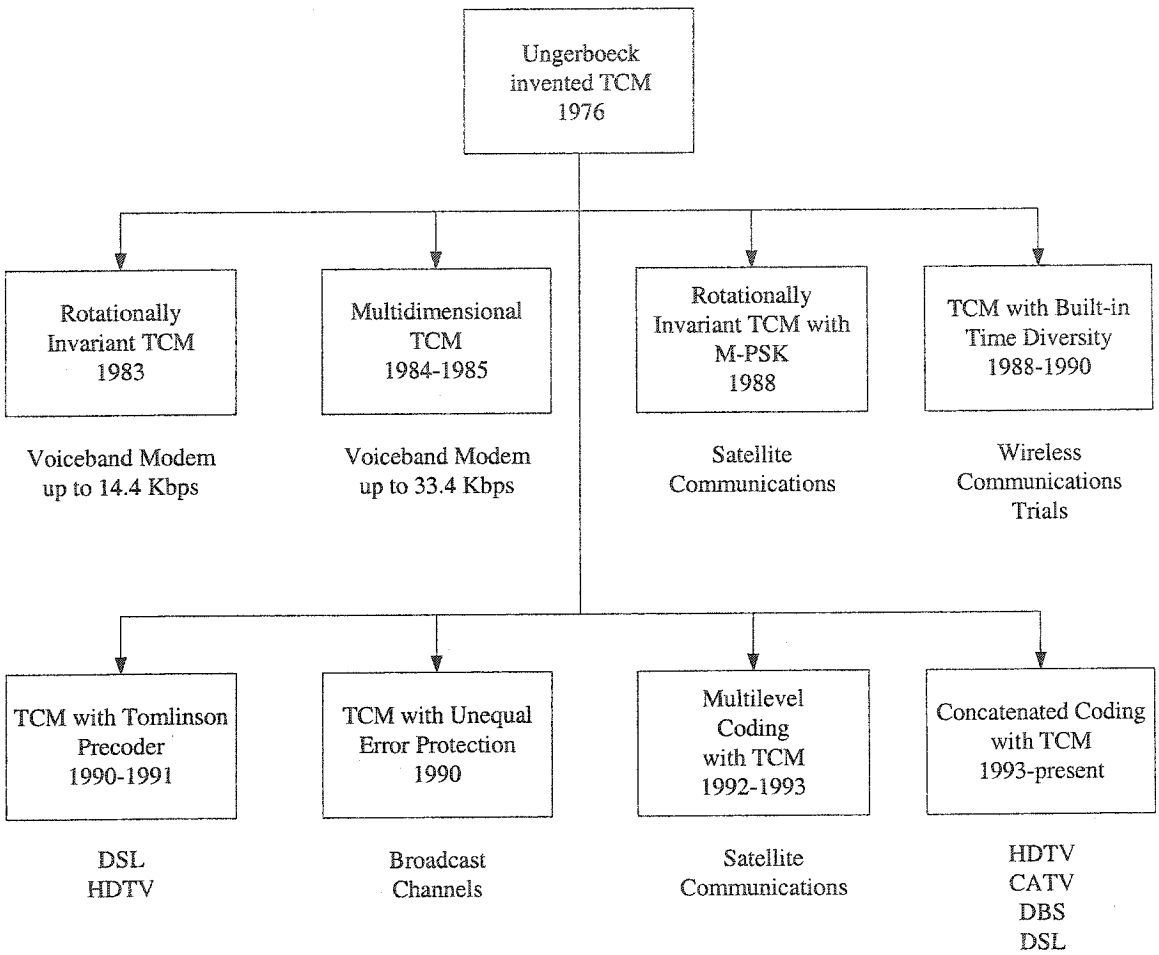


Figure 1-1: TCM evolution and applications

Pragmatic TCM introduced in 1989 by Zehavi and Wolf [8] reduced the complexity of the TCM encoder/decoder. Pragmatic TCM is employed the same de-facto standard rate 1/2 convolutional code and techniques known as "puncturing" to achieve coded modulation with different rates. This method permits the use of single basic convolutional encoder/decoder to achieve respectable coding gain which is virtually equivalent to that of best Ungerboeck approach.

1.2 Motivation and Methodology of the research

Although considerable amount of research works have been conducted on different aspects of TCM, only a few of them are focused on hardware implementation. Furthermore, the available implementations are not designed with the regularity and methodology that System On a Programmable Chip (SOPC) design flow requires. As an example, an ASIC design TCM codec is implemented by Xiao Hu in 2003 [32]. In Xiao Hu's approach, the four dimensional *16*-state TCM codec is proposed and implemented. However, in this thesis we focus on the implementation of a modem with Concatenated Pragmatic TCM codec based on SOPC design flow. The regularity of Punctured Pragmatic TCM structure makes it well suited for the SOPC design flow. Furthermore, in the design flow Intellectual property (IP) cores are used to simplify the design procedure. This approach can be used as a prototype design flow for implementing any DSP algorithm.

The Pragmatic TCM is recommended by Digital Video Broadcasting (DVB) standard as an inner coder of a base-band modem for satellite communication. Digital Video Broadcasting is a well known telecommunications European standard that founded in 1993 to establish global standards for digital television and data services. The specification of the modem is stated at ETSI EN 301 210 [14]. Concatenated code (RS encoder as an outer code, PTCM as an inner encoder) is used to improve the performance and reliability of the data transmission. In the thesis, the two-stage Pragmatic TCM codec is designed and implemented along with the specification given in DVB standard.

A large number of choices are available to implement a Digital Signal Processing (DSP) system; such as programmable DSP, Filed Programmable Gate array (FPGA) and

Application Specific Integrated Circuit (ASIC). Each of these solutions has its own strengths and weaknesses.

The programmable DSP is essentially a microprocessor which is designed for DSP applications. It is an off-the-shelf part that can be programmed with a very high level language such as C, or perhaps with assembly. Programmable DSPs are very flexible because they can be programmed again and again. These characteristics permit fast design iteration and reduce time to market [28]. Programmable DSPs are also well suited for very complex math-intensive tasks, with conditional processing. However, they are limited by number of operations which can be done per clock cycle. As an example, a TMS320c6201 has two multipliers and can perform well up to 200 MHz; hence 400 M multiplications per seconds can be done by the DSP [27].

On the other hand, FPGA is a programmable device which is configured by connecting the gates together. Elements such as registers, adders and multipliers as well as building DSP blocks such as FFT and FIR can be implemented in FPGA. FPGA vendors have developed several LogiCores which can be used in block diagram level to implement very efficient DSP tasks. It can also be configured by means of Hardware Description Language (HDL). Their performance is limited to the number of gates available and clock rate. FPGA is well suited for parallel processing i.e. multiple execution units can be built that operate simultaneously. For example, a 1 M gate Virtex II device has 40 multipliers and can perform as high as 100 MHz. comparing to the DSP counterpart, it executes 4000M multiplications in a second.

When the sampling rate of the design is more than a few MHz and extensive parallel processing is required in the design, FPGA is the best choice. Moreover,

telecommunication systems are usually described by block diagram; translating the block diagram to FPGA may be simpler than converting it to C code for DSP.

A number of customized Matlab scripts and Simulink libraries is prepared in the design flow to model the modem and perform system simulation. Floating point simulation is performed in Simulink platform to verify the telecommunication system. Afterwards, fixed-point simulation is performed and the simulation results are compared with floating-point ones. When the fixed-point representation of the system is chosen, the system is implemented in Xilinx Virtex II FPGA. Xilinx System Generator for DSP is used to translate fixed-point model to VHDL. Xilinx Synthesis Tool (XST) in Integrated Software Environment (ISE) package is used to synthesize the HDL model. In addition, gate level simulation is performed to verify timing and functionality of the post-synthesis design. Finally, the HDL model is configured in to the target FPGA chip.

1.3 Organization of the thesis

The organization of this thesis is as follows: Chapter 2 introduces fundamentals of any communication systems with a focus on concatenated codes and Pragmatic TCM. Chapter 3 provides an overview of DVB and demonstrates the structure of the encoder for the desired specifications. Chapter 4 provides further description on the design flow, model structure and simulation results. Floating point and Fixed-point simulation results are presented in this chapter. Chapter 5 provides hardware implementation, details of the hardware design flow and verification. Finally, the conclusion and future work are provided in chapter 6.

Chapter two

2 Digital Telecommunications System

The objective of any digital communications system is to provide a cost effective facility for transmitting information at a specific rate from one end of the system to the user at the other end; in an acceptable reliability level and quality. Two primary communication resources for the designer are bandwidth and power. In many communication systems, one of these resources may be more precious, thus any system may be classified to either band-limited or power-limited. In band-limited communication systems, spectrally efficient modulation techniques are employed to save bandwidth at the expense of power. However in power-limited systems, power efficient modulation techniques can be used to save power at the expense of bandwidth. In both cases, Error Correction Coding (Channel Coding) might be used to have a more reliable communication medium with the same power by increasing the bandwidth. However, increase in bandwidth is not desirable and therefore Ungerboeck [4-6] proposed Trellis coded Modulation (TCM) schemes to improve the performance of the communication systems without any increase in bandwidth.

In this chapter, non-binary band-limited modulation technique; including pulse-shaping and matched filter is presented. Fundamentals of Error Correction Coding are also discussed and finally Trellis Coded Modulation is thoroughly described.

2.1 QAM Modulation scheme

M-ary non-orthogonal signaling schemes are well known techniques to achieve bandwidth reduction by sending K bits, at each symbol duration T_s. Instead of using binary numbers, an alphabet with $M=2^K$ symbol is employed, hence the data rate with the same bandwidth can be expressed as:

$$R_b = KR_s = \frac{K}{T_s} = \frac{\log_2 M}{T_s} bits/s \qquad (2.1)$$

Therefore according to the Equation (2.1), and knowing that the required bandwidth is proportional to the symbol rate (i.e. R_s); M-ary signaling increases the bit rate by a factor of K with the same bandwidth, comparing to binary case. There are different modulation schemes employing M-ary signaling such as M-ary Quadrature Amplitude Modulation (QAM), Phase Shift Keying (PSK), and Amplitude Shift Keying (ASK). In this thesis, the QAM will be described in details.

QAM can be considered as a logical extension of PSK and ASK, or simply as a two-dimensional ASK with two orthogonal basis. Figure 2-1 illustrates a *16*-ary QAM constellation; each constellation point contains $4=\log_2 16$ bits. It is more likely to receive

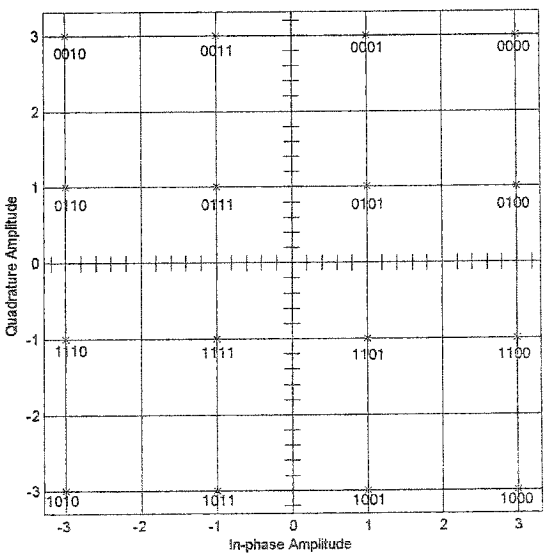


Figure 2-1: QAM constellation with Gray coding

adjacent symbols in error, therefore the bit to signal assignments should be done such that adjacent symbols differ in only one bit position (Gray coding). This provides the best bit error rate (BER). For a rectangular constellation with Gray coding, a Gaussian channel, and matched filter reception, the probability of bit error is expressed by [1]:

$$P_B \approx \frac{2(1-L^{-1})}{\log_2 L} Q\left[\sqrt{\left(\frac{3\log_2 L}{L^2-1}\right)\frac{2E_b}{N_o}}\right] \qquad (2.2)$$

where L represents the number of amplitude level in one dimension and $Q(x)$, called Q-function, is defined as:

$$Q(x) \approx \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \exp(-\frac{u^2}{2}) du \qquad (2.3)$$

The BER performance of *M*-QAM scheme with *M=4, 16* and *256* is sketched in Figure 2-2. It is illustrated that performance degrades with the increase of modulation level. In other words, higher bit rate is achieved by decreasing the performance with the same transmitted power.

Any digital scheme that transmits *K* bits in T_s seconds, using bandwidth of *W Hz*, operates at a bandwidth efficiency of:

$$\frac{R_b}{W} = \frac{\log_2 M}{WT_s} bits/s/Hz \qquad (2.4)$$

Bandwidth efficiency is a factor that shows how the system uses the bandwidth efficiently; for example, a communication system can achieve a bandwidth efficiency of *4 bits/s/Hz* instead of *1 bits/s/Hz*, if it employs *16*-QAM modulation scheme instead of BPSK.

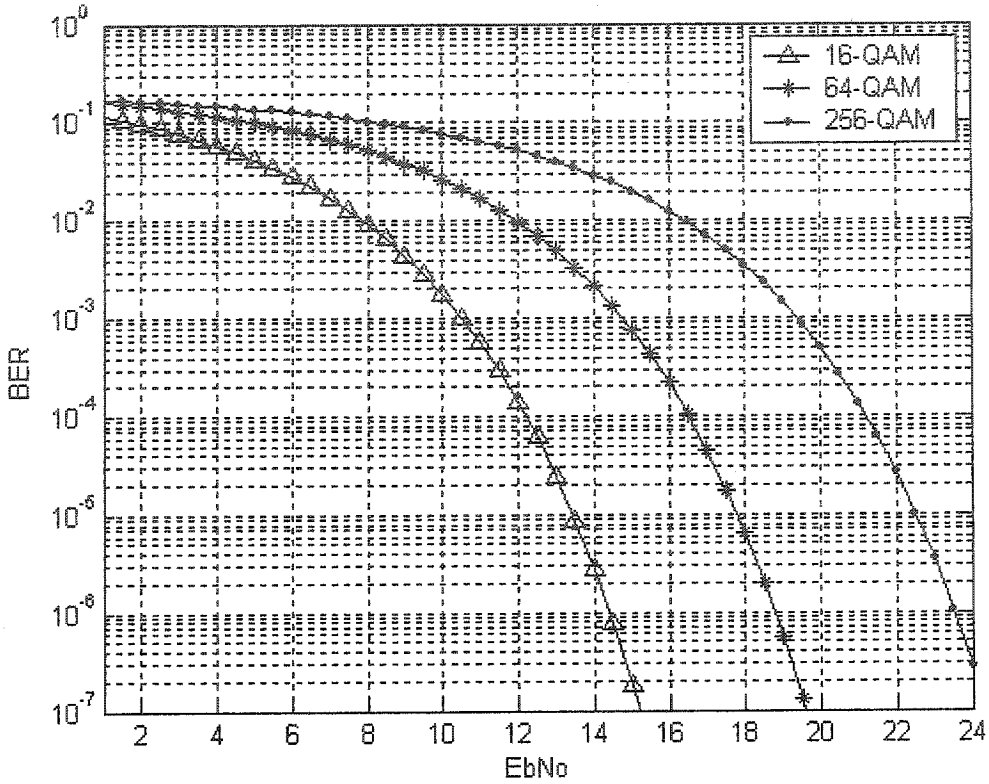


Figure 2-2: Various QAM BER curves

2.2 Pulse-Shaping and Matched filters

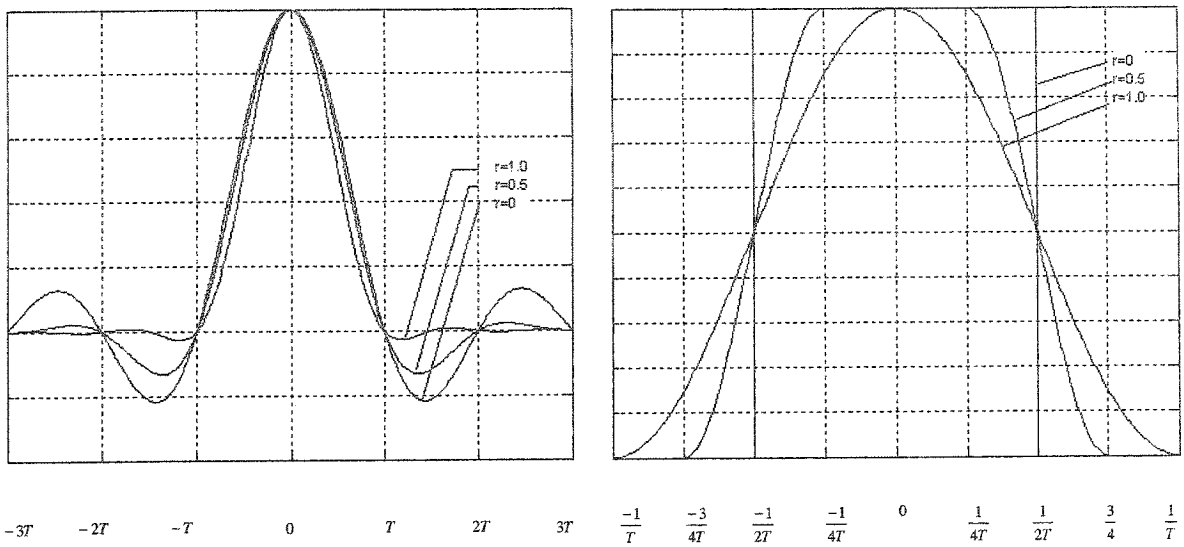
In modern data transmission systems, bits or groups of bits (i.e.: symbols) are typically transmitted in the form of individual pulses of energy. In the receiver the goal is to sample the received signal at an optimal point in the pulse interval to maximize the probability of right decision. This implies that the shape of transmitted pulses be such that they do not interfere with one another and on the other hand, maximum signal-to-noise-ratio (SNR) at the optimal sampling point is required. This can be accomplished by using a pulse-shaping filter in the transmitter and matched-filter in the receiver, respectively.

2.2.1 Pulse-shaping filter

In communication systems bandwidth is limited. The more compact the signal bandwidth is made, the greater the number of users that can simultaneously be served. It is essential to reduce the required system bandwidth; however there is a limitation to such bandwidth reduction. Once a system operates at smaller bandwidth, the pulse would spread in time domain and results in Inter-Symbol-Interference (ISI). Nyquist showed that at based-band communication, theoretical minimum system bandwidth needed to detect R_s Symbols/s without ISI is $R_s/2$ *Hz*. This might lead us to an ideal Nyquist filter with its rectangular-shaped frequency response; however such a filter is not realizable. A practical filter should be introduced to have zero ISI at each sampling time with the reasonably small bandwidth. Raised cosine filter is a candidate that meets above requirements. The raised cosine frequency transfer function can be expressed as:

$$H(f)=\begin{cases} T_s & 0\le|f|\le\dfrac{1-\beta}{2T_s} \\ \dfrac{T_s}{2}\left\{1+\cos\left[\dfrac{\pi T_s}{\beta}\left(|f|-\dfrac{1-\beta}{2T_s}\right)\right]\right\} & \dfrac{1-\beta}{2T_s}\le|f|\le\dfrac{1+\beta}{2T_s} \\ 0 & |f|>\dfrac{1+\beta}{2T_s} \end{cases} \qquad (2.5)$$

where β is called roll-off factor and takes the value in the range of $0\le\beta\le1$. The bandwidth occupied by the signal beyond the Nyquist minimum bandwidth (i.e. $\frac{1}{2T_s}$) is called "excess bandwidth" and is usually expressed at the percentage of Nyquist bandwidth. For example, β of *0.5* or *1* corresponds to the excess bandwidth of *50* percent or 100 percent, respectively. The raised cosine filter impulse response and frequency response are plotted in Figure 2-3 for different roll-off factors.



(a) (b)

Figure 2-3: Raised-cosine filter characteristic

(a) System impulse response (b) System transfer function

2.2.2 Matched filter

A matched filter is a linear filter designed to provide the maximum signal-to-noise- ratio (SNR) at its output for the given transmitted symbol waveform. It is proved [1] that such a filter has an impulse response of the mirror image of the message signal *s(t)*, delayed by the symbol duration *T*. The impulse response of a matched filter is expressed as :

$$h(t) = \begin{cases} Ks(T_s - t) & 0 \le t \le T_s \\ 0 & elsewhere \end{cases} \qquad (2.6)$$

where K is constant.

In digital systems, the sampled matched filter is derived by sampling h(t). The discrete impulse response of sampled matched filter is expressed as:

$$h_d(n) = \begin{cases} Ks(M-n) & 0 \le n \le M \\ 0 & elsewhere \end{cases} \qquad (2.7)$$

where M and n are integer.

Both pulse-shaping filter and matched-filter are essential parts of any communication system. The pulse shaping filter should be designed such that the overall transfer function of the transmitter and receiver filter, produces zero ISI at optimal sampling point. Square-root-raised-cosine (SQRC) is turned out to be the solution, by dividing raised-cosine filter transfer function to two SQRC filters, one in the transmitter and the other one in the receiver. The product of transmitter and receiver filter transfer functions being a raised-cosine will provide zero ISI at optimal sampling point in the receiver. Furthermore, according to the equation (2.6) and the fact that impulse response of SQRC filter is even symmetric, the second SQRC filter is matched to the transmitted symbol. In other words, it provides maximum SNR at its output. The impulse response of SQRC filter is expressed as [2]:

$$h_{srrc}(t) = \frac{(4rt/T_s)\cos[\pi(1+rt/T_s)] + \sin[\pi(1-r)t/T_s]}{(\pi t/T_s)[1-(4rt/T_s)^2]} \qquad (2.8)$$

where r $(0 \le r \le 1)$ is roll-off factor and T_s is symbol period.

2.3 Coding Fundamental

Forward Error Correction (FEC) Coding introduces more reliable digital transmission by increasing the "distance properties" in expense of higher bandwidth. In other words, the price paid for channel coding is bandwidth because of the extra redundant bits. This appended redundant bits, increase distance in signal space vector between each coded vector, i.e. comparing two sequences before and after coding is applied, the coded sequences have more differing bits than unencoded sequence. The decoder in the receiver is then able to detect and/or correct a certain number of errors to retrieve the original data correctly.

For a system with a fixed power the only practical option to increase the performance is to encode transmitted bits in the transmitter. In the receiver, according to the coding scheme, the decoder would be able to detect and correct specific number of bits in error. Another practical motivation for the use of coding is to reduce the required E_b/N_o for a fixed bit error rate. This reduction in E_b/N_o may, in turn, be exploited to reduce the required transmitted power and reduce the hardware cost.[3]

There are many different error-correcting codes that can be used. These codes have been classified into block codes and convolutional codes.

2.3.1 Block codes (RS code)

To generate an *(n ,k)* binary block code, the encoder accepts successive blocks of *k-bits* data; for each block, it adds *n-k* bits to the information bits and produces an *n*-bit *(n>k)* codeword. In other words, a block encoder produces bits at a rate of $R_c = \frac{n}{k} R_b$;

where R_b *bits*/sec is bit rate and the dimensionless ratio $r = \frac{k}{n}$ $(0 < r < 1)$ is called the code rate. There are a total number of 2^k different possible messages fed to the block encoder. Corresponding to these 2^k different possible messages, there are only 2^k different possible *n-bit* codewords exist at the encoder output. For clarifying the error correction and detection capability of a block code, a term *minimum distance* (i.e. $d_{\min}$) has to be defined; $d_{\min}$ of a block code is the minimum number of bits which are different between two codewords in set of all codewords. Thus, in block codes, no error pattern of $d_{\min} - 1$ or fewer errors can change any received codewords to another one. Hence, the received sequence is not a valid codeword. When the decoder detects an invalid received codeword, the error is detected. In other words, the decoder is able to detect any error patterns of $d_{\min} - 1$ and less. Moreover, a block code with *minimum distance* $d_{\min}$ is not only able to detect a large number of error patterns, but also guarantees correcting all error patterns of $t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$ or fewer errors in each code word; where $\left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$ denotes the largest integer no greater than $\frac{d_{\min} - 1}{2}$.

Reed-Solomon codes are an important subclass of non-binary cyclic codes. The encoder of *RS(n,k)* code, is different from the binary encoder. It operates on multiple bits rather than individual bits. A *t*-error correction *RS(n,k)* code with *m*-bit per symbols has the following parameters:

Block length:	$n = 2^m - 1$ symbols	
Message size:	k symbols	(2.9)
Symbol size:	m bits	
Parity-check size:	$2t = n - k$ symbols (for even value of n - k)	
Minimum distance:	$d_{\min} = 2t + 1$ symbols	

This code is able to correct up to t, m-bit symbols in a block of n symbols. Unlike binary block codes where each bit is corrected individually in codewords, RS decoder corrects a complete symbol of m-bit in case of symbol error. The symbol error correction is carried out irrelevant to the number and the position of error patterns in the symbol. Hence, this gives RS codes a capability of correcting burst of error [1]. For example, consider *(n,k)=(255,239)* with $m=8$ bits per symbol; since $2t = n - k = 16$, this code, in the worst case scenario, can correct a burst of error of *57* bits in a block of *255* symbols of *8*-bit long. However, this code can only correct up to *8* random bits in the block of *255* symbols of *8*-bit long.

Any block code (in systematic form) can be shortened without affecting the number of errors that can be corrected within the block length [1]. These shortened codes are used in systems that shorter code words are required. In terms of *RS(255,239)* $t=8$, imagine that 51 of 239 message symbols are a set of all-zero symbols, which are not actually transmitted and hence are not subjected to any errors. Thus the shortened Reed-Solomon code i.e. *RS(204,188)* of that of *RS(255,239)*, with the same $t = 8$ symbol error correction capability, is obtained.

2.3.2 Convolutional codes

A convolutional code is represented as (n,k,l) where parameters n, k, as well as ratio k/n has the same significance as that of block codes. The integer l is a parameter known as constraint length; it represents the number of k-tuple stages in the encoding shift register. Unlike block codes, convolutional encoders have memory. In other words, output at each time unit is not only a function of input k-tuple but also it is a function of the previous l-1 input k-tuple. A convolutional code can completely be described by its encoding function, nevertheless there are several methods used to represent a convolutional encoder, such as connection representation and trellis diagram.

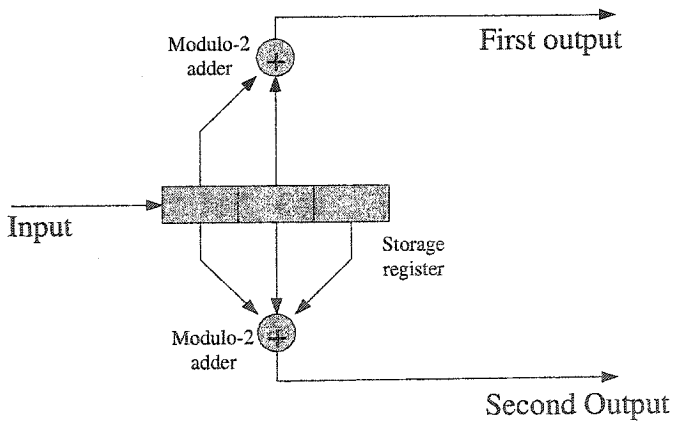


Figure 2-4: Connection representation of the Convolutional encoder

In Figure 2-4 the connection representation of a very simple convolutional encoder with three shift registers is depicted. The encoder works by passing a continuous stream of input bits through a memory device such as shift register. The register length, i.e. constraint length, is an important parameter as it provides the distance properties of the resulting coded sequence. The constrain length of the encoder in Figure 2-4 is $l=3$.

The number of output bits (n) for every input bit is determined by the number of modulo-2 adders. In this example there are two adders, thus making the code rate k/n equal to $1/2$. Each of modulo-2 adders is connected to specific cells of the shift register. Once the modulo-2 addition is completed, the output bits are produced. A new bit enters the shift register; while the content of registers shift one place to the right, then the right most bit is discarded. Each output bit is not only dependent on the present input bit, but the past l-1 bits as well.

Binary vector representation of the generator polynomials of this encoder can be expressed by $G_1=(110)$ and $G_2=(111)$; for the first and second output respectively. A straightforward approach to illustrate an encoder is to show each encoder output with its octal value, instead of that of binary vector. For example; this encoder can be expressed as $G_1=6$ and $G_2=7$.

Trellis diagram representation of a convolutional encoder shows output and each state transition, when a new information bit enters the encoder. Trellis diagram of the encoder of Figure 2-4 is depicted in Figure 2-5. It illustrates the complete characteristic of the encoder; i.e. state transition and coded bits can be derived easily by pursuing the trellis diagram according to the input bit at each unit of time. Trellis diagram is also a manageable way to illustrate Viterbi Algorithm (VA).

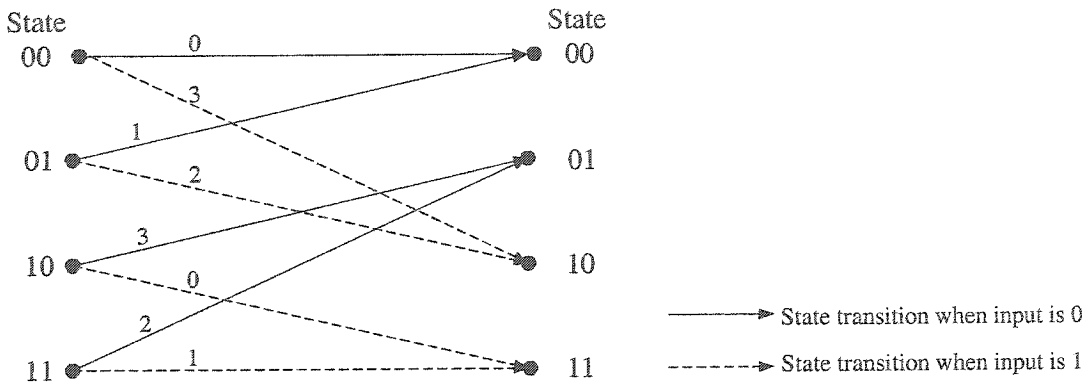


Figure 2-5: Trellis diagram representation of the Convolutional encoder

Viterbi Algorithm, discovered by Viterbi in 1967, is used at the receiver end to perform decoding. The advantage of the VA is that it reduces the computational complexity of the brute-force decoding by taking advantage of the special structure of the code trellis. Viterbi Algorithm essentially performs maximum likelihood decoding [1]. The algorithm involves calculating a *measure of similarity* or *distance*, called metric, between all possible input sequences. The viterbi chooses the most likely sequence that has the same pattern as the received sequence. This can be done by measuring metric of the received sequence and all other sequences at time *t*, and eliminating all those that could not possibly be candidates. The early rejection of unlikely path reduces the decoder complexity [1]. The process is then repeated from bit to bit while the metric of "surviving path" is accumulated among all possible sequences. Once the trellis grows large enough (usually five times the constrain length) *trace-back* begins, by choosing the path with *maximum likelihood metric*, or *minimum distance metric*. A complete tutorial of VA is given by Viterbi in [7].

The demodulator can also be configured to feed the decoder quantized value of the received signal, instead of hard value of '0' and '1'. These soft values are used by

decoder to recover the message sequences. This provides the decoder more information than hard-decision case, therefore it results in performance improvement. For example, for the Gaussian channel, soft-decision decoding by eight-level of quantization results in performance improvement of approximately *2* dB in required signal to noise ratio comparing to hard-decision. In fact, feeding such a *3*-bit word instead of single binary symbol is equivalent to providing the decoder a *measure of confidence* along with codeword decision [1]. In other words, "*000*" is strong indication of "*0*" and "*111*" gives the decoder very high confidence that "*1*" has been sent.

2.4 Interleaving and Concatenated Codes

A channel is called memory-less channel when it affects transmitted signal with random independent noise. On the other hand, a channel that has memory is one that exhibits mutually dependant signal transmission impairments (i.e. the disturbances cause errors in bursts). In these bursty channels interleaving can be used to randomize the errors.

Interleaving the coded message before transmitting and de-interleaving in the receiver cause burst of error to be spread out in time and thus to be handled by the decoder as if they were random error. Two types of interleaver are commonly used, Block and Convolutional interleaver.

2.4.1 Block interleaving

A block interleaver accepts coded symbols from the encoder, permutes the symbols and feeds the rearranged symbols to the next block in the transmitter; in the

receiver the de-interleaver performs the reverse operation. The usual permutation of the block is accomplished by filling the column of an *M*-row by *N*-column (*MxN*) array with the encoded sequence, when the array is completely filled; the symbols are fed to the next block in the transmitter one row at a time. The most important characteristics of such a block interleaver are [1]:

- Any burst of *N* continuous channel symbol errors, results in isolated errors that is separated in the de-interleaver by at least *M* symbols.
- The interleaver/de-interleaver end-to-end delay is approximately *2MN* symbol times, and the memory requirement is *MN* symbols for each of interleaver and de-interleaver.

Interleaver parameters (i.e. number of columns *N* and that of rows *M*), are selected such that interleaving protects transmitted data from the expected burst of error in the channel. Generally speaking, for *t* error correction code, *N* is selected such that its value exceeds the expected burst length in the channel divided by *t*. The choice of row *M* is dependent on the coding scheme used. For block codes, *M* should be larger than the codeword length, while for convolutional codes *M* should exceed the constrain length.

2.4.2 Convolutional interleaving

The structure of Convolutional interleaver that has been proposed by Forney [11] appears in Figure 2-6. Interleaver consists of *N* branches of storage registers. Each successive bank of registers provides *J* symbols more storage than the preceding one. As illustrated, the zeroth register provides no storage while the *(N-1)*th register delays its output by *(N-1)J* symbols. The interleaver works by sequentially passing a symbol of

coded data to each branch of registers. At each new code symbol, the commutator switches to the next register and feeds the register a new symbol, while the previous symbols are shifted one place to the right. After *(N-1)* registers, the commutator switches to the zeroth register again. In the receiver, the deinterleaver performs the inverse operation. Note that input and output commutator for both interleaver and deinterleaver have to be synchronized.

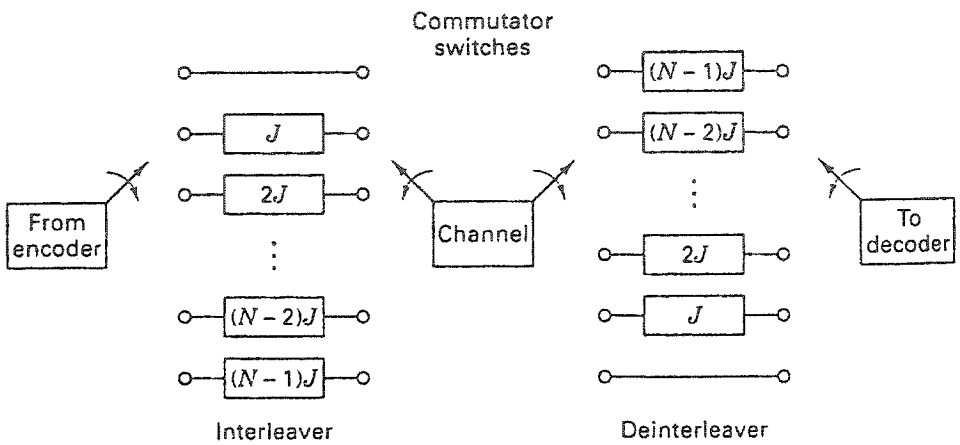


Figure 2-6: Fundamental structure of a Convolutional Interleaver [1]

Performance of a convolutional interleaver is very similar to that of a block interleaver. The important advantage of the convolutional interleaver over block interleaver is that the end-to-end delay with convolutional interleaver is equal to $M(N-1)$ symbols, where $M = NJ$, and its required memory is $M(N-1)/2$ at both end of the channel. Therefore, it is a reduction by one-half in the memory size and delay compared to the block interleaving.

2.4.3 Concatenated codes

A concatenated code is one that uses two levels of coding, an inner code and outer code to achieve the desired error performance. The primary reason for using the concatenated code is to achieve a better performance with an overall low implementation complexity. Figure 2-7 depicts a general modem block diagram based on concatenated codes. The inner code is usually configured to correct most of the channel errors and the outer code is usually a lower rate code, improves the overall performance by correcting the imminent burst of errors at the output of the inner decoder. An interleaver is required to randomize burst of errors produced by inner decoder.

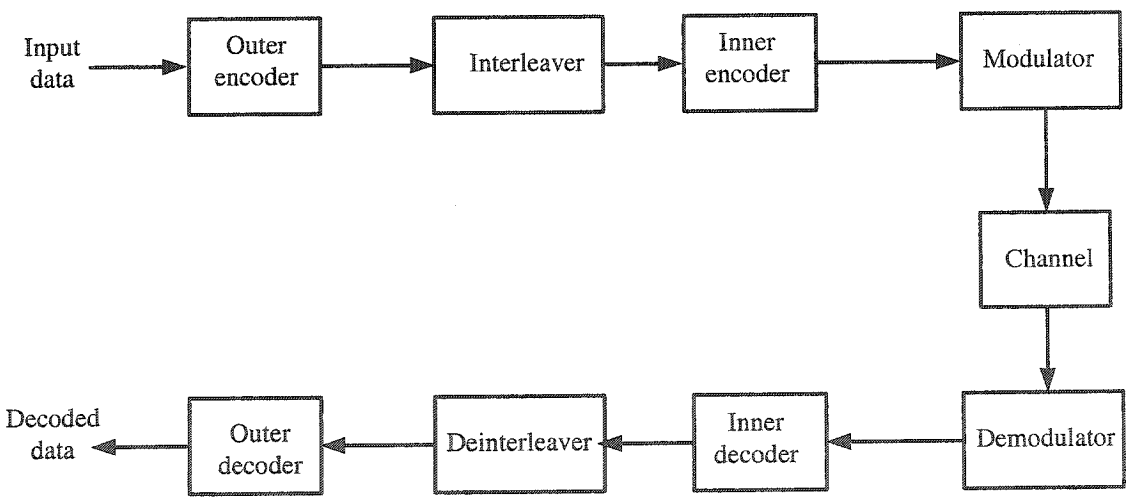


Figure 2-7: General Modem block diagram based on concatenated codes

2.5 Trellis Coded Modulation

In classical Digital Communications System, the functions of modulation and coding are separated, however Trellis Coded Modulation is a combined modulation and coding scheme that achieves remarkable coding gain over conventional multi-level modulation scheme with the same bandwidth efficiency.[4]

2.5.1 Ungerboek Trellis coded modulation

A Convolutional Code with rate of k/n, by adding n-k redundant bits to the message is able to correct up to $\left\lfloor \frac{d_{free}-1}{2} \right\rfloor$ bits in error, where d_{free} is minimum free distance of the convolutional code. Hence, coding sacrifices spectral efficiency of the system to improve the performance. Generally, there exist two possibilities to compensate for this loss; by either increasing the bandwidth or enlarging the signal set of the modulation system if the channel is band-limited. Therefore, to maintain the same spectral efficiency the idea of enlarging the signal set of the modulation is worthy for band-limited channels. However, when the modulation and Error-Correction are performed in the independent manner, disappointing results are obtained. As an illustration, consider four-phase modulation (*4*-PSK) without coding, and eight-phase modulation (*8*-PSK) used with a binary convolutional code of rate *2/3*. Note that both systems transmit two information bits per modulation interval (*2 bit/sec/Hz*). If the *4*-PSK system operates at an error rate of 10^{-5}, at the same signal-to-noise ratio the "raw" error rate at the *8*-PSK demodulator exceeds 10^{-2}, because of the smaller spacing between the *8*-PSK signals. Patterns of at least three bit errors must be corrected to reduce the error rate to that of the uncoded *4*-PSK system. A rate *2/3* binary convolutional code with constraint length K= *6* has the required value of d_{free}. For decoding, a fairly complex *64*-state binary Viterbi decoder is needed. However, after all this effort, error performance will be equal to that of uncoded *4*-PSK.[4]

One problem of former *8*-PSK coded-system is hard decision detection prior to the decoder, which causes irreversible loss of information in the receiver. The remedy for

this will be soft decision decoding, such that decoder operates directly on unquantized soft output samples of the channel. Consider samples at the input of the demodulator as $r_n = a_n + w_n$, where; r_n and a_n are samples of received and transmitted signals respectively, and w_n represents samples of AWGN process.

The decision rule of the optimum sequence decoder is to determine sequence $\{\hat{a}_n\}$ (output of cascaded decoder and demodulator) with minimum squared Euclidean distance from $\{r_n\}$. In other words, decoded sequence is determined among the set C of all coded signal sequence such that sum of squared errors will be minimum, that is, the sequence of $\{\hat{a}_n\}$ which satisfies [4]:

$$|r_n - \hat{a}_n|^2 = Min \sum_{\{\hat{a}_n\} \in C} |r_n - a_n|^2 \qquad (2.10)$$

The Viterbi Algorithm can be used to determine the coded signal sequence closest to the received quantized signal. However, when optimum sequence decisions are made directly in terms of Euclidean distance, another problem becomes apparent. Mapping of binary data to non-binary symbols according to maximized *hamming distance*, do not guarantee maximum *Euclidean distance* as well. This was the motivation for Ungerboek to propose a rule called "mapping by set partitioning" to achieve maximum desired Euclidean distance between signal sets. It is based on successive partitioning of the expanded 2^{m+1}-ary signal set into subsets with increasing Euclidean distance, $\Delta_0 < \Delta_1 < \Delta_2 < \ldots$, between the signal points of these subsets. As an illustrative example consider the set partitioning of the 8-PSK signal set in Figure 2-8.

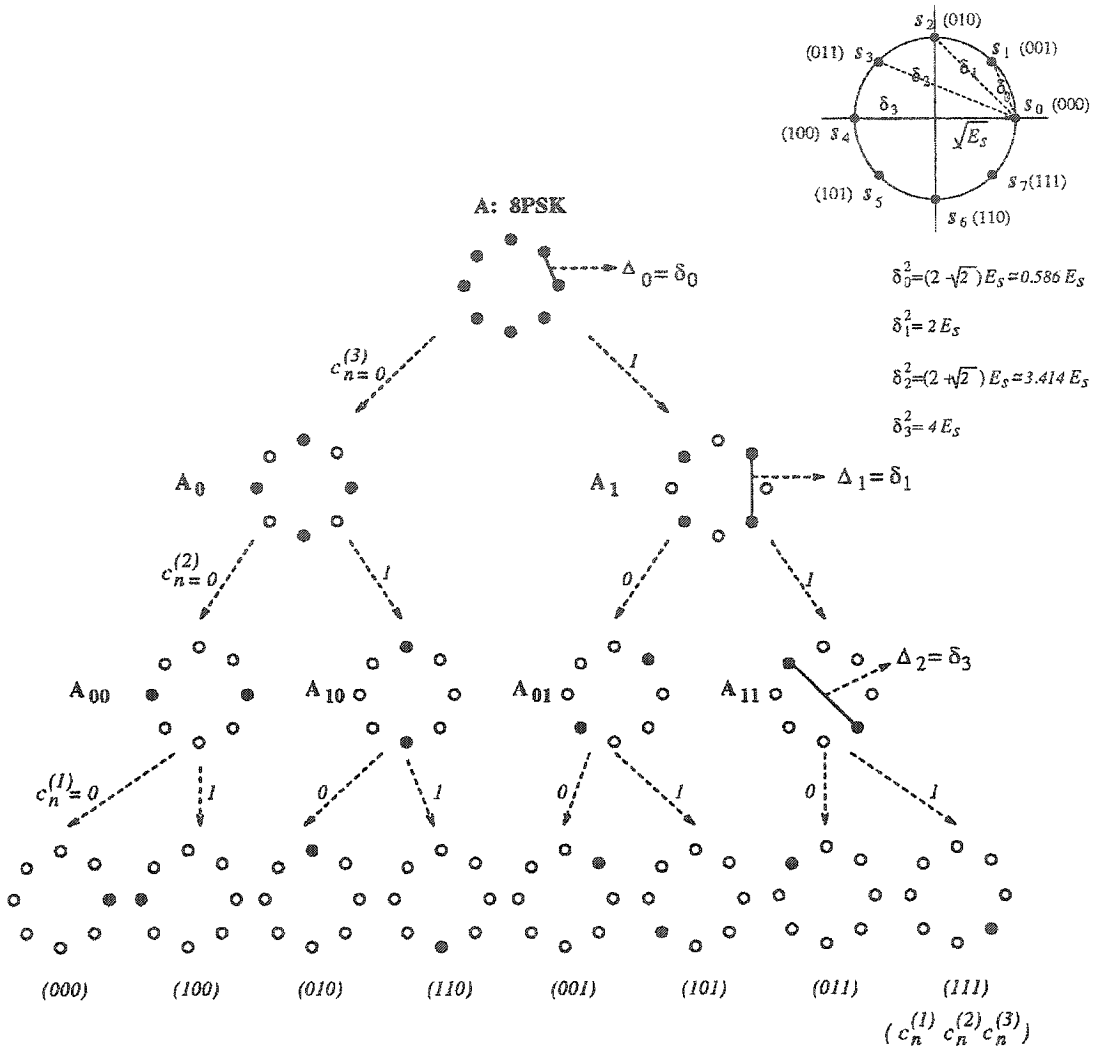


Figure 2-8: Signal set partitioning of *8*-PSK [30]

In the first step, the *8*-PSK signal set is partitioned into two subsets A_0 and A_1 each having four points. The minimum ED of these subsets is Δ_1, which is larger than that of original *8*-PSK. The second step of partitioning divides each of A_0 and A_1 to two subsets with two signal points. The minimum ED of the subsets in this level is Δ_2. Further partitioning of these subsets leads to eight subsets with a single signal point.

In general, it is not necessary to continue partitioning up to a single signal point in each subset, since this leads to the value of Δ_i that exceeds the free ED that one expects to achieve at a given code complexity.[5]

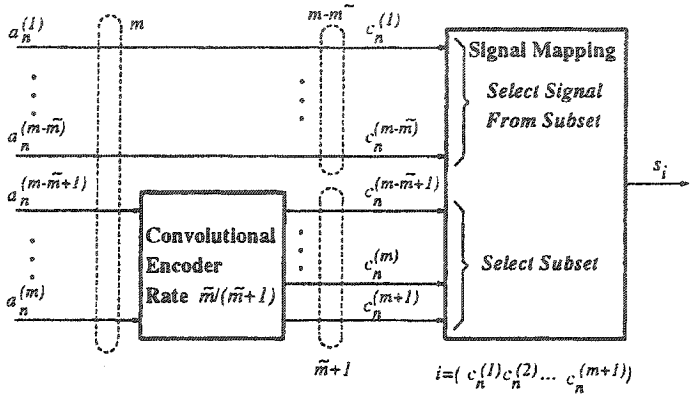


Figure 2-9: General structure of a TCM encoder [30]

A general structure of a TCM encoder is shown in Figure 2-9, in this figure a block of m information bits enters the encoder. From these m bits, $\tilde{m} \leq m$ bits are encoded by a rate $\tilde{m}/(\tilde{m}+1)$ binary convolutional encoder into $\tilde{m}+1$ bits, while the remaining $m-\tilde{m}$ bits are left uncoded. The $(\tilde{m}+1)$ coded bits are used to select one of the $2^{(\tilde{m}+1)}$ possible subsets, while the remaining $m-\tilde{m}$ uncoded bits are used to select one of the $2^{(m-\tilde{m})}$ signal points in this subset.

The number of states in the trellis diagram of a TCM scheme depends on the number of registers v in the convolutional encoder, i.e. it is equal to 2^v. The greater the v is, the better encoder can be achieved, with higher code complexity. An extensive research has been done to find the best convolutional codes with different constraint length for each TCM scheme.[5] The number of transitions between two states depend on the number of uncoded bits, i.e. for $m = \tilde{m}$ the states are joined by single transition while

for the case of $m > \tilde{m}$ there exists $2^{(m-\tilde{m})}$ parallel transitions between states. These parallel transitions are associated with $2^{(m-\tilde{m})}$ signals of the subsets in the lowest layer of the set partitioning tree. A trellis diagram of a four state encoder for the rate 2/3 8-PSK TCM is depicted in Figure 2-10.

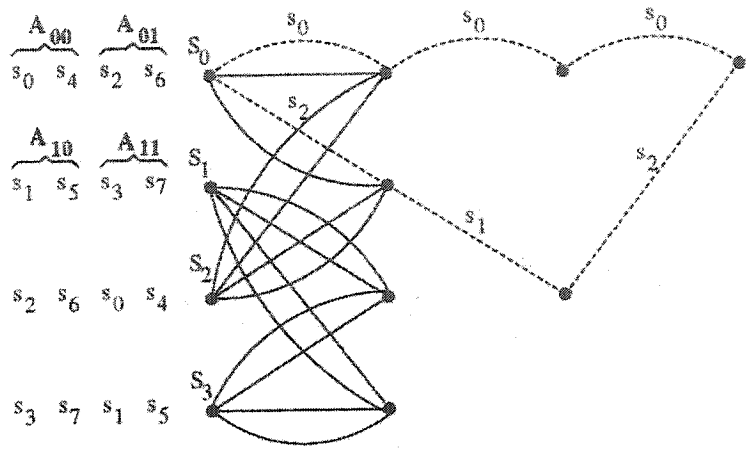


Figure 2-10: Trellis diagram of the four state 8-PSK TCM scheme [30]

Denoting the minimum ED between parallel transition by $\Delta_{\tilde{m}+1}$ and the minimum ED between the non-parallel paths in the TCM encoder by $d_{free}(\tilde{m})$, the free ED of the TCM code can be expressed as:

$$d_{free} = Min[\Delta_{\tilde{m}+1}, d_{free}(\tilde{m})] \qquad (2.11)$$

At this point it would be useful to summarize the Ungerboeck TCM design rules in three steps:

1. Parallel transitions, if exist, are associated with the signals in subsets of the lowest layer of the set partitioning tree. These signals have the minimum ED of $\Delta_{\tilde{m}+1}$.

2. The transitions originating from or merging into one state are associated with the signals in the first step of set partitioning. The ED between these signals is at least Δ_1.
3. All signals are used with equal frequency in the trellis diagram.

2.5.2 Pragmatic Trellis-Coded Modulation

Although the TCM scheme proposed by Ungerboek can achieve Asymptotic Coding Gain (ACG) as much as 6 dB within precisely the same spectral efficiency, it has major drawbacks as follows [8]:

- Each signal configuration requires different codes, so no single encoder-decoder could be used for a wide set of parameters e.g. different spectral efficiency of *2,3* and *4* $b/s/Hz$.
- Implementation of a convolutional decoder with high number of states in its trellis diagram is not practical. Therefore, encoder with low number of states has to be used which results in low ED. This affects performance of the trellis code.

These drawbacks were the motivation to develop the Pragmatic TCM (PTCM). PTCM exploits the same convolutional code for all rates $n/n+1$ TCM code with different spectral efficiency. A technique known as "puncturing" permits use of modified version of the optimum code for rate $1/2$ to produce any rate of $n/n+1$. The resultant punctured code is moderately less efficient than the optimum code for that rate. Therefore, by using a punctured version of this convolutional code, different TCM

schemes, with wide set of parameters can be produced. Note that, only one encoder-decoder is used.

Punctured codes are obtained by periodically deleting a fraction of the symbol generated by encoder. Thus, this results in higher code rate and higher spectral efficiency than that of the original code. In the receiver, prior to the decoder, the deleted symbols are replaced by *erasure*. These null symbols do not affect the decision rule of the decoder. Therefore, the same decoder of un-punctured code can be used.[9] puncturing technique is very attractive not only for simplifying the Viterbi decoder for high-rate codes but also for implementing a rate-selectable convolutional encoder-Viterbi decoder.

PTCM takes advantage of the *de facto* standard convolutional code of rate $1/2$ and its punctured version of rate $3/4$ and $7/8$ with Constraint length of $K=7$. This convolutional code which is optimum in the sense of maximum free distance ($d_{free}=10$), can be fully defined by its two generators $G_1 = (133)$ and $G_2 = (171)$. This pragmatic approach to all coding applications permits the use of a single basic encoder and decoder to achieve respectable coding gains for the bandwidth efficiencies from *1 b/s/Hz to 6 b/s/Hz*.

Assume that we desire a rate $(n-1)/n$ pragmatic trellis coded modulation for a signal constellation of 2^n points. The $(n-1)$ binary vectors, $C_1, C_2, ..., C_{n-1}$ is encoded to binary n-bit vector, $D_1, D_2, ..., D_{n-1}, D_n$ according to the rule; $D_i = C_i$ for i=1 to $(n-2)$ and $(D_{n-1}, D_n) = (B_1, B_2)$, where the B_1 and B_2 are two outputs of the rate 1/2 convolutional encoder corresponding to C_{n-1} [10]. This general pragmatic encoder has been shown in Figure 2-11.

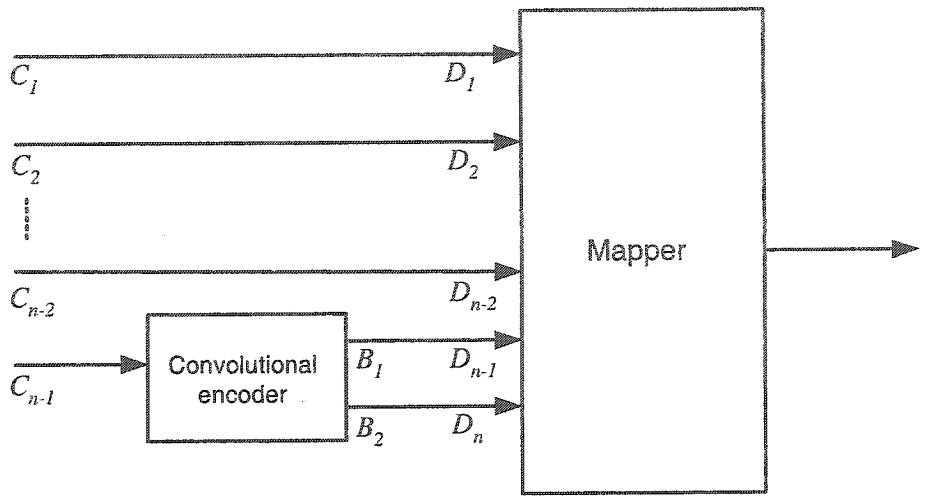


Figure 2-11: General structure of PTCM encoder

Figure 2-12 illustrates the pragmatic coded modulation system for *M=4, 8* and *16*-PSK. In this general encoder, the two encoded bits select one of the four phases according to the following Gray coding [8]:

$$00 \rightarrow 0 \quad \text{rad.}$$

$$01 \rightarrow \pi/2^{n-1} \quad \text{rad.}$$

$$11 \rightarrow 2\pi/2^{n-1} \quad \text{rad.}$$

$$10 \rightarrow 3\pi/2^{n-1} \quad \text{rad.}$$

The remaining (uncoded) *n-2* bits select a subset lexicographically, i.e. the binary vector whose decimal equivalent is *j*; where $(0 \le j \le 2^{n-2}-1)$ selects the *(j+1)st* sector. This mapping provides the maximum possible Euclidean distance between uncoded bits within different subsets with the same coded bits, while the coded bits of adjacent signal points differ in only one bit. A generalized *M*-ary PSK mapper, is illustrated in this section, since it is easier to follow. A *16*-QAM mapper for Pragmatic TCM will be explained thoroughly in the next chapter.

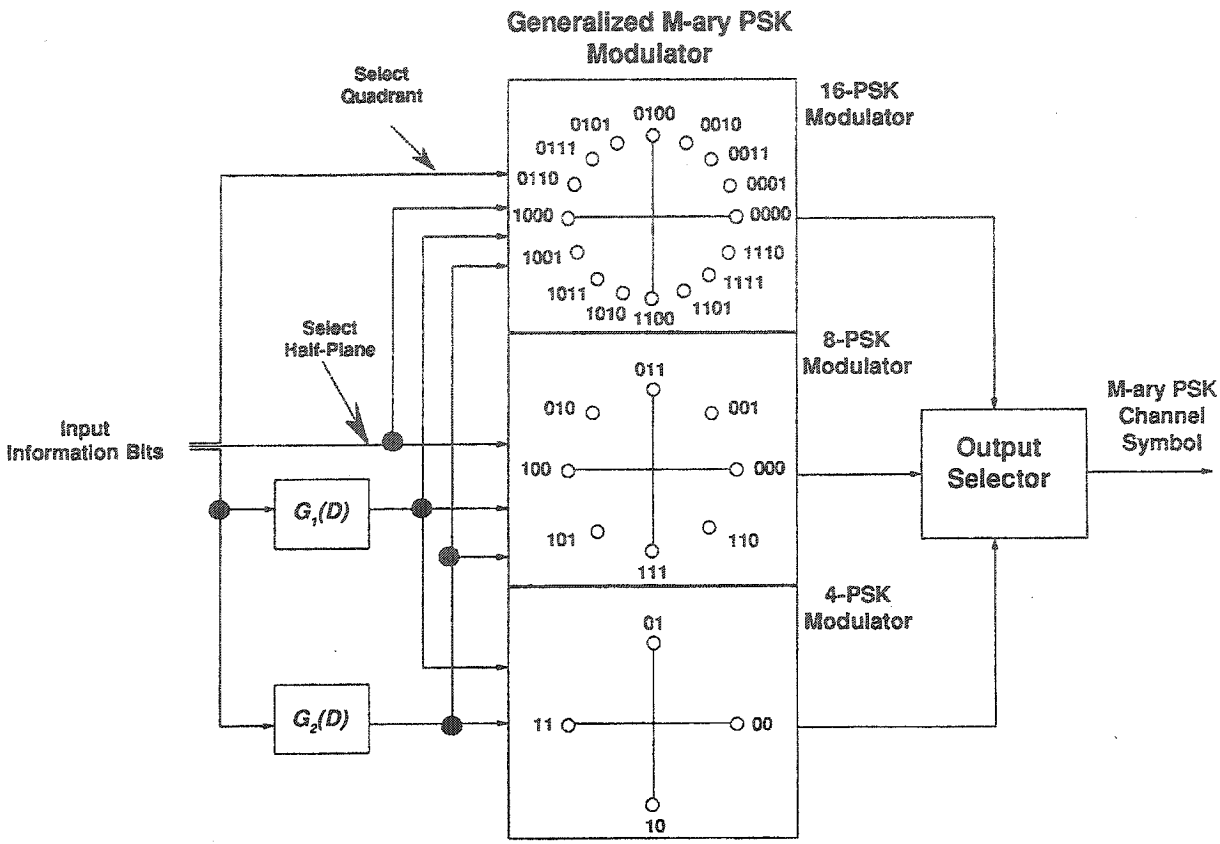


Figure 2-12: Pragmatic encoder/modulator of QPSK, *8*-PSK and *16*-PSK [8]

Chapter three

3 Digital Video Broadcasting European standard

Digital Video Broadcasting (DVB) is a well known telecommunications European standard that has been established by the joint technical committee broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI). This joint group of industry, operators and broadcasters provided the necessary information on all relevant technical matters, as well.

In this chapter, based on ETSI EN 301-210 [14], the Framing structure, channel coding and modulation scheme of the implemented modem is explained in Figure 3-1. The system is presented as a functional block of equipment performing the adaptation of the base-band TV signal, from the output of the MPEG-2 transport multiplexer to the satellite channel characteristics. This standard, proposes variety of modulation schemes, however in this thesis *16*-QAM scheme is implemented. Concatenation of pragmatic trellis coded modulation (PTCM) with shortened RS code provides the modem a capability to perform in quasi-linear satellite channels.

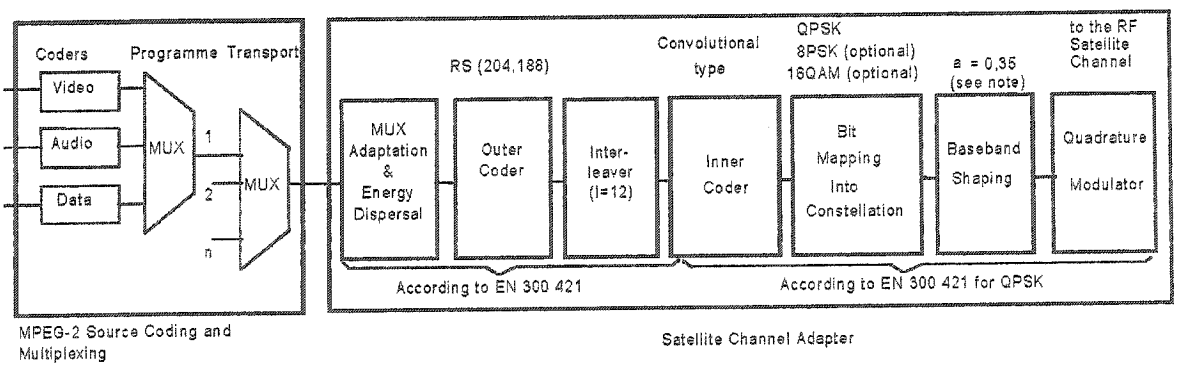


Figure 3-1: Functional block diagram of the System [14]

The following characteristics of the modem will be defined in this chapter based on DVB standard:

- Reed-Solomon code (outer code)
- Convolutional interleaving
- Pragmatic TCM (inner code)
- Squared root raised cosine pulse-shaping filters with roll-off factor $\alpha = 0.35$

3.1 Reed-Solomon code

Reed-Solomon *RS(204,188, t = 8)* shortened code is used as an outer code in the modem as shown in Figure 3-2. A block of 188 bytes (message symbols) is fed to the RS encoder. Then, the encoder generates a codeword of 204 bytes. The message and codeword are depicted in Figure 3-3. The message contains 187 byte of information data appended by a sync byte. Note that, Reed-Solomon coding is also applied to the sync byte which is either non-inverted (i.e. 47HEX) or inverted (i.e.B8HEX).

Figure 3-2: A simple block diagram of the transmitter

The proposed RS code has the following code generator and field generator polynomial:

$$g(x) = (x + \lambda^0)(x + \lambda^1)(x + \lambda^2)...(x + \lambda^{15}) \qquad \text{where } \lambda = 02 \text{ HEX}$$

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1$$

Shortened Reed-Solomon code i.e. *RS(204,188)*, as it was explained in chapter two, is implemented by adding 51 bytes of zero to the message bytes of *RS(255,239)* encoder. Then, when encoding procedure is done, these null bytes are discarded.

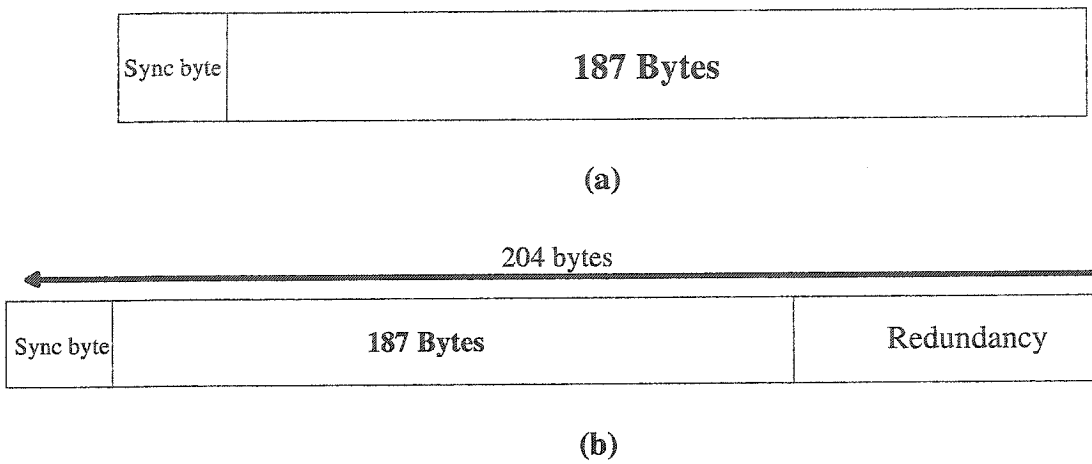


Figure 3-3: Message symbol and codeword of RS code

(a) MPEG-2 message block (b) error protected codeword block

3.2 Convolutional interleaving

The concept of convolutional interleaver was explained in the sub-section 2.4.2 of the thesis. For the recommended *RS(204,188)* encoder, a convolutional interleaver with depth $I = 12$, and memory block of $M = 12$ symbols is used to interleave the entire $N = 12 * 17 = 204$ codeword. The interleaver is composed of $I = 12$ branches, cyclically connected to the input byte stream by the input switch. Each branch consists of a First-In, First-Out (FIFO) shift register, with depth Mj cells, where $M = 17 = N/I, N = 204$ (error protected frame length), $I = 12$ (interleaving depth) and j is the branch index. Each cell of the FIFO contains *1* byte of information, and the input

and output switches are synchronized. The recommended interleaver is shown in Figure 3-4.

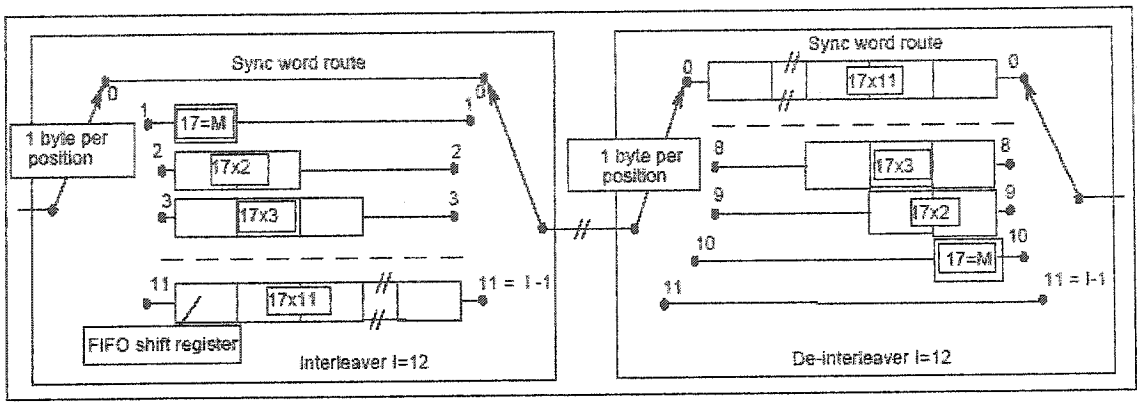


Figure 3-4: Conceptual diagram of the convolutional interleaver and de-interleaver [29]

3.3 Inner coding Pragmatic TCM

The recommended inner coding scheme is PTCM with different rates and modulation schemes, which is explained thoroughly in the subsection 2.5.2 of the thesis. A general block diagram of this PTCM is illustrated in Figure 3-5.

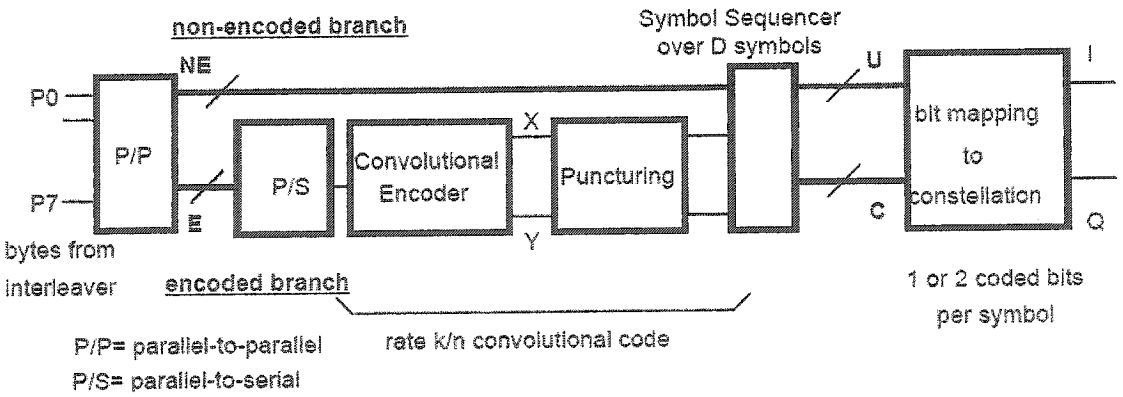


Figure 3-5: inner PTCM principle [14]

In general, PTCM requires two branches of encoded bits and unencoded bits in order to map bit streams to the selected constellation. With the aid of puncturing,

different high rate PTCMs can be produced with moderately acceptable performance loss but higher spectral efficiency. The proposed convolutional encoder is the best rate $r = 1/2$ convolutional encoder. This *64*-state convolutional encoder is totally described with its two generator polynomials i.e. $G_1 = (133) \text{and} G_2 = (171)$ which has a free distance of $d_{free} = 10$ prior to puncturing.

	Input P									Output
MODE	LAST							FIRST		
QPSK	A0	A1	A2	A3	A4	A5	A6	A7	⇒	E1
8PSK - 2/3	B0	B1	B2	B3	B4	B5	B6	B7	⇒	NE1
	A0	A1	A2	A3	A4	A5	A6	A7	⇒	E1
8PSK - 5/6	G3	G7	F3	F7	D3	D7	B3	B7	⇒	NE4
	G2	G6	F2	F6	D2	D6	B2	B6	⇒	NE3
	G1	G5	F1	F5	D1	D5	B1	B5	⇒	NE2
	G0	G4	F0	F4	D0	D4	B0	B4	⇒	NE1
	A0	A1	A2	A3	A4	A5	A6	A7	⇒	E1
8PSK - 8/9					F5	F7	B1	B7	⇒	NE6
					F4	F6	B0	B6	⇒	NE5
					F3	D3	D7	B5	⇒	NE4
					F2	D2	D6	B4	⇒	NE3
					F1	D1	D5	B3	⇒	NE2
					F0	D0	D4	B2	⇒	NE1
					A1	A3	A5	A7	⇒	E2
					A0	A2	A4	A6	⇒	E1
16QAM - 3/4	D1	D3	D5	D7	B1	B3	B5	B7	⇒	NE2
	D0	D2	D4	D6	B0	B2	B4	B6	⇒	NE1
	A0	A1	A2	A3	A4	A5	A6	A7	⇒	E1
16QAM - 7/8	L3	L7	G3	G7	D3	D7	B3	B7	⇒	NE4
	L2	L6	G2	G6	D2	D6	B2	B6	⇒	NE3
	L1	L5	G1	G5	D1	D5	B1	B5	⇒	NE2
	L0	L4	G0	G4	D0	D4	B0	B4	⇒	NE1
	H2	H5	F0	F3	F6	A1	A4	A7	⇒	E3
	H1	H4	H7	F2	F5	A0	A3	A6	⇒	E2
	H0	H3	H6	F1	F4	F7	A2	A5	⇒	E1

Table 3-1: parallel-to-parallel conversion [14]

Byte parallel streams in the output of the convolutional interleaver enter the Parallel-to-parallel (*P/P*) block, as it is illustrated in Figure 3-5. The *P/P* block splits the input bits into two branches depending on the selected modulation/inner coding mode. The order of parallel-to-parallel converters is selected to reduce, on average, the byte error-rate at the input of Reed-Solomon decoder, i.e. high concentration of bit-errors in

bytes. Therefore, the bit-error-rate (BER) after RS correction is improved. Parallel-to-parallel converter is also synchronized such that MPEG sync-bytes, in the normal form (47HEX) or bit-wise inverted form (B8HEX), appear in the first byte (**A**) shown in Table 3-1. When an MPEG sync byte (47HEX) is transmitted, **A** is coded as follows: **A** = [01000111]. The signal **NE** of the non-encoded branch generates a sequence of signals **U**, through the Symbol Sequencer, each to be transmitted in a modulated symbol. These bits generate parallel transitions in the trellis diagram of the encoder, and are only protected by large Euclidean distance in the signal space. The signal **E** in the encoded branch is processed by the punctured convolutional encoder according to the puncture code definition of Table 3-2. These bits generate, through the Symbol Sequencer, a sequence of signals **C**, each to be transmitted in a modulated symbol. It is worthwhile to mention that **NE**, **E**, **U** and **C** are set of parallel bits which are sent out at unit of time.

Original code			Code rates									
			1/2		2/3		3/4		5/6		7/8	
K	G1 (X)	G2 (Y)	P	dfree	P	dfree	P	dfree	P	dfree	P	dfree
7	171_{OCT}	133_{OCT}	X: 1 Y: 1 I=X_1 Q=Y_1	10	X: 1 0 Y: 1 1 I=X_1 Y_2 Y_3 Q=Y_1 X_3 Y_4	6	X: 1 0 1 Y: 1 1 0 I=X_1 Y_2 Q=Y_1 X_3	5	X: 1 0 1 0 1 Y: 1 1 0 1 0 I=X_1 Y_2 Y_4 Q=Y_1 X_3 X_5	4	X: 1 0 0 0 1 0 1 Y: 1 1 1 1 0 1 0 I=X_1 Y_2 Y_4 Y_6 Q=Y_1 Y_3 X_5 X_7	3
NOTE: 1 = transmitted bit 0 = non transmitted bit												

Table 3-2: Puncture code definition [29]

MODE	LAST SYMBOL			FIRST SYMBOL		Output
QPSK - 1/2				Y1	⇒	C2
				X1	⇒	C1
QPSK - 2/3		Y4	X3	Y1	⇒	C2
		Y3	Y2	X1	⇒	C1
QPSK - 3/4			X3	Y1	⇒	C2
			Y2	X1	⇒	C1
QPSK - 5/6		X5	X3	Y1	⇒	C2
		Y4	Y2	X1	⇒	C1
QPSK - 7/8	X7	X5	Y3	Y1	⇒	C2
	Y6	Y4	Y2	X1	⇒	C1
8PSK - 2/3				NE1	⇒	U1
				Y1	⇒	C2
				X1	⇒	C1
8PSK - 5/6			NE2	NE4	⇒	U2
			NE1	NE3	⇒	U1
			Y1	X1	⇒	C1
8PSK - 8/9		NE2	NE4	NE6	⇒	U2
		NE1	NE3	NE5	⇒	U1
		Y2	Y1	X1	⇒	C1
16QAM - 3/4				NE2	⇒	U2
				NE1	⇒	U1
				Y1	⇒	C2
				X1	⇒	C1
16QAM - 7/8			NE2	NE4	⇒	U2
			NE1	NE3	⇒	U1
			X3	Y1	⇒	C2
			Y2	X1	⇒	C1

Table 3-3: Puncturing and symbol sequencer definitions [14]

According to Table 3-1 and Table 3-3 there are different modulation schemes and bit mapping available for both M-PSK and M-QAM modulations. Bit mapping into constellation is carried out by associating $k = \log_2^M$ input bits, i.e. U, C in Figure 3-5 with corresponding vector in the Hilbert signal space belonging to the chosen constellation. Possible constellations are *8*-PSK (k = *3* bit) and *16*-QAM (k = *4* bit). Optimum mapping into constellation for coded and uncoded bits is different in the case of *1* coded bit per symbol (CBPS) or *2* CBPS schemes. The Cartesian representation of each vector will be indicated by *I, Q*, i.e. the in-phase and quadrature components. Since we

are interested in 16-QAM constellation, only $r = 3/4$ and $r = 7/8$ code rate are explained in detail:

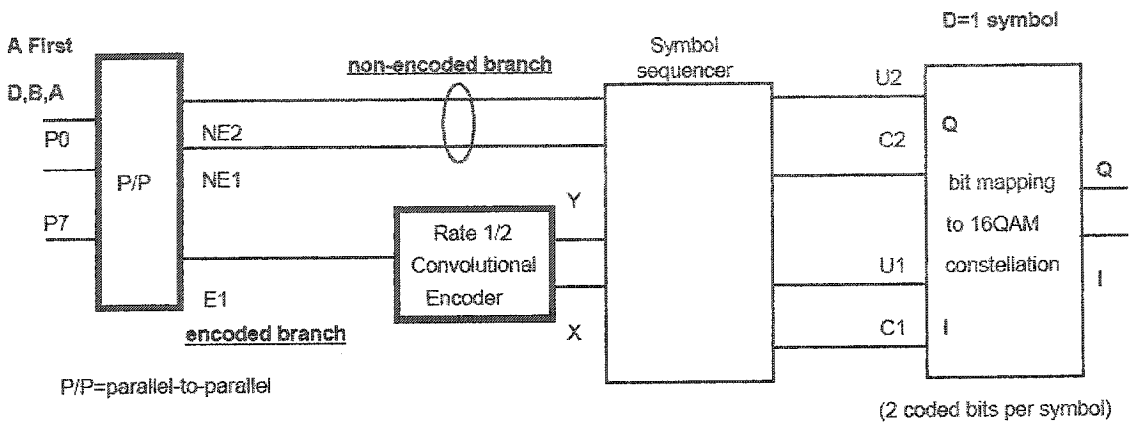


Figure 3-6: Inner coding principle of rate *3/4* non-punctured code [14]

According to the Figure 3-6, three bytes i.e. **A**, **B** and **D** from output of convolutional interleaver are fed to the *P/P* block. Note that **A** is the first byte fed to the *P/P* converter and **D** is the last one. NE_i *(i=1,2)* branches are directly connected to the symbol sequencer, however E_1 is fed to the convolutional encoder to produce two encoded bits i.e. *X*, *Y*. Symbol sequencer arranges four bits required for *16*-QAM mapper along with Table 3-3. Finally, two coded and uncoded bits are mapped to constellation in compliance with Figure 3-7. The proposed mapping not only achieves the maximum Euclidean distance between uncoded bits, (parallel transition in the code trellis) but also allows a simple implementation approach i.e. both *I* and *Q* are mapped with the same method.

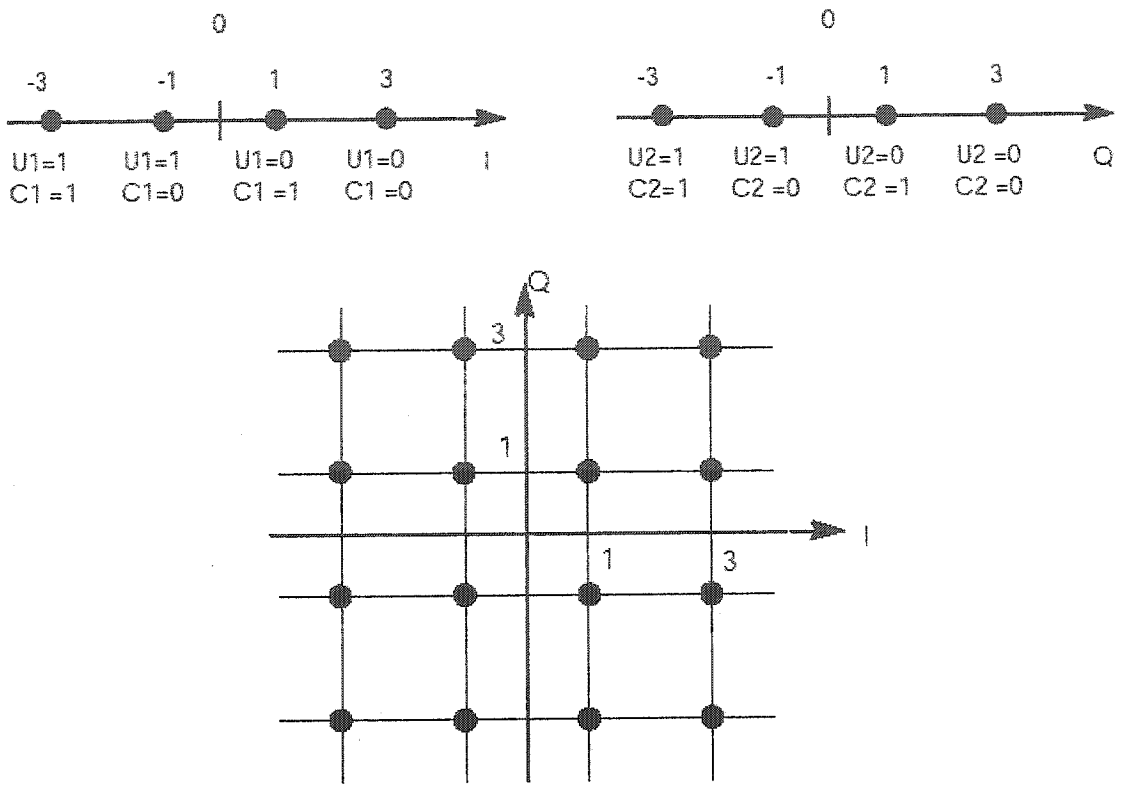


Figure 3-7: Bit mapping to the *16*-QAM constellation for both rates *3/4 & 7/8* [14]

Punctured trellis encoder of rate $r = 7/8$ is illustrated in Figure 3-8. Seven bytes i.e. **A,B,D,F,G,H** and **L** from the output of convolutional interleaver are fed to the *P/P* converter. Output branches of the converter are associated with parallel inputs according to Table 3-1. Uncoded bits i.e. NE_i *(i=1,2,3,4)* are directly connected to the symbol sequencer, but the three encoded branches i.e. E_i*(i=1,2,3)* are fed to a parallel to serial converter followed by convolutional encoder producing two encoded-bit i.e. *X, Y* at unit of time. Timing table of the *P/S* converter and its coded-bits output is illustrated in Figure 3-9. At the output of the convolutional encoder two bits out of each block of six bits, are deleted in compliance with punctured matrix defined in Table 3-2. This makes a

punctured convolutional encoder of rate $r = 3/4$. Symbol sequencer arranges four bits required for the *16*-QAM mapper along with Table 3-3. Finally, coded bits and uncoded bits are mapped to constellation of two separated symbol according to Figure 3-7.

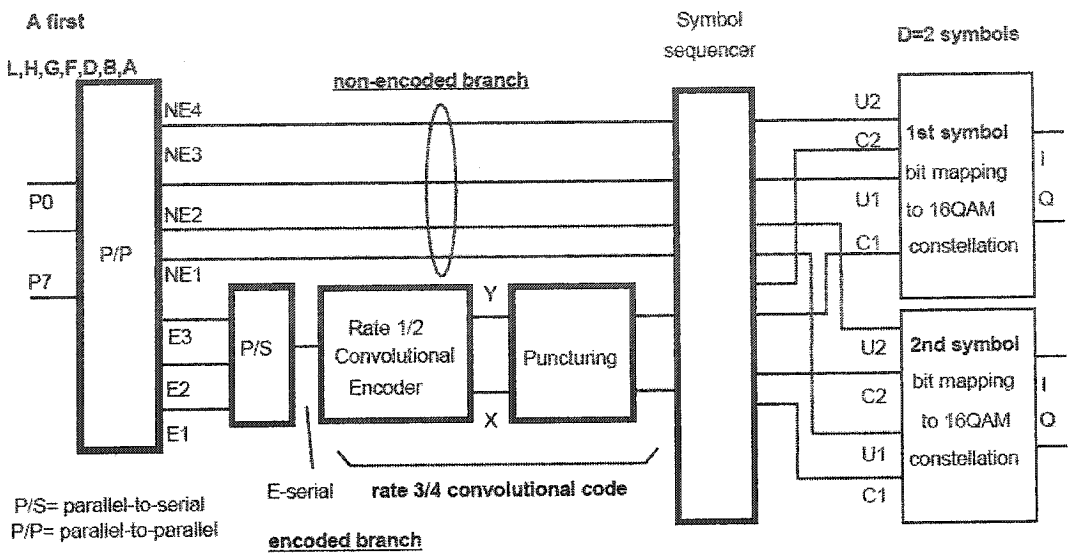


Figure 3-8: Inner coding principle of rate *7/8* punctured code [14]

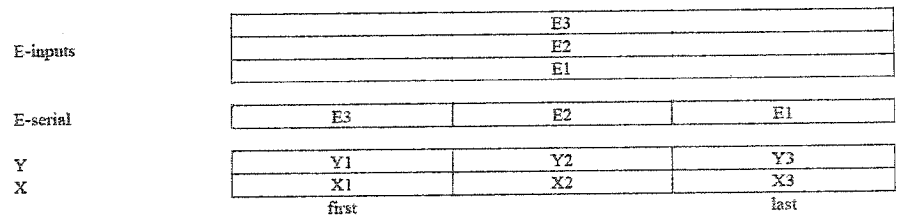


Figure 3-9: Timing table of the P/S converter and convolutional encoder [14]

3.4 Squared root raised cosine pulse-shaping

Prior to modulation, *I* and *Q* signals, mathematically represented by a succession of Dirac delta functions, multiplied by the amplitudes *I* and *Q*, spaced by the symbol

duration $T_s = 1/R_s$, is filtered by a square root raised cosine filter to have no ISI in the receiver. The roll-off factor of this SQRC filter is $\alpha = 0.35$.

$$H(f) = \begin{cases} 1 & f_N(1-\alpha) \le |f| \le f_N(1+\alpha) \\ \frac{1}{2} + \frac{1}{2}\sin\frac{\pi}{2f_N}\left[\frac{f_N - |f|}{\alpha}\right]^{\frac{1}{2}} & |f| > f_N(1+\alpha) \\ 0 & \\ |f| < f_N(1-\alpha) & \end{cases}$$

where; $f_N = \frac{1}{2T_s} = \frac{R_s}{2}$ is the Nyquist sampling frequency and α is the roll-off factor.

Chapter four

4 DVB Modem Design and Simulation

In order to design and implement the system, a model based on mathematical requirements of the system should be proposed. The proposed system based on the standard should be verified by means of computer simulations. If necessary, the design should be modified and reevaluated to meet desired characteristics. Eventually designer should take into account number of implementation issues such as timing analysis and hardware cost of the design.

In this chapter, modem design flow is proposed, block diagrams of the design are illustrated thoroughly and finally both floating-point and fixed-point simulation results of the system are depicted.

4.1 Implementation design flow

The design flow for this project is illustrated in Figure 4-1. As shown, design flow is divided to four steps. At the first step floating point modeling and simulation is performed using Matlab Simulink. A model is designed based on the mathematical basis expressed in earlier chapters. The system consists of basic logical/mathematical components as well as complex blocks which are available as part of Simulink Blocksets library. When the design is completed, it is simulated to obtain the BER curve. As it is mentioned earlier, Bit Error Rate versus bit energy to noise spectral density (E_b/N_o) curve is an important measurement of the performance of any communication system.

Therefore, the simulation and theoretical curves are compared to ensure the feasibility of the design. Design must be modified if the system simulations are not desirable.

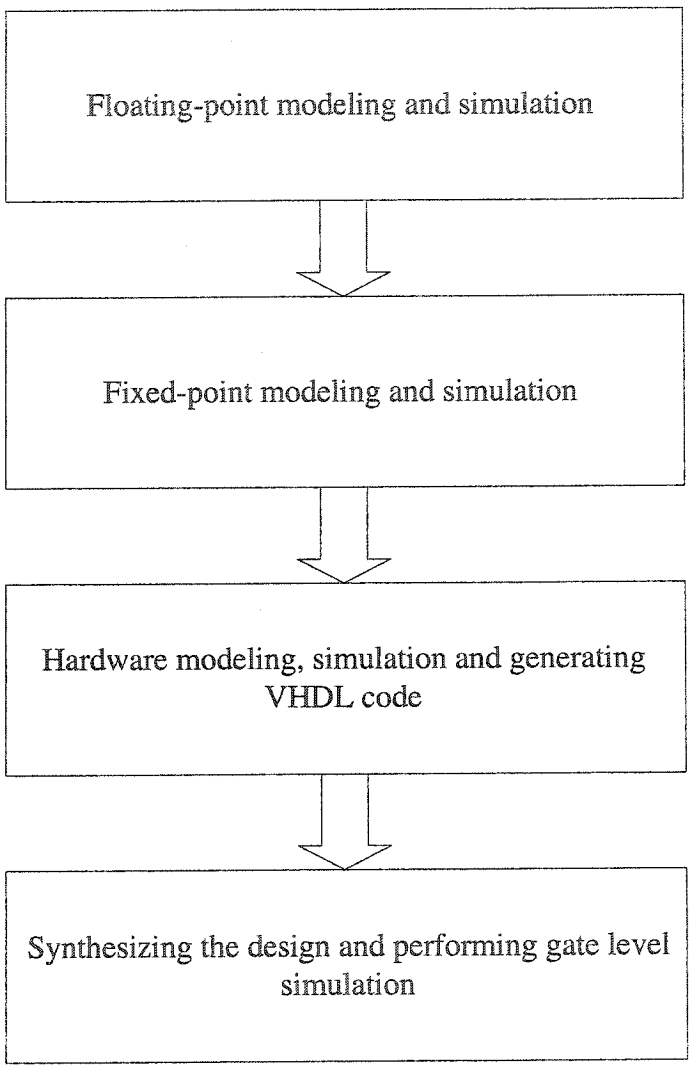


Figure 4-1: Implementation design flow

As depicted in Figure 4-1, second step is fixed-point simulation. At this point, it is worth explaining the concept of fixed-point and floating-point numbers. IEEE 754-1985 standard [31] is the most common floating-point representation of real numbers used in computer software/hardware architecture. According to the standard, floating-point

numbers are shown either by 32 bits (single precision) or 64 bits (double precision) depending on the desired precision. Fixed-point representation on the other hand, employs a sliding-window with a fixed radix-point to show real numbers. This allows a simple representation of real number by scarifying accuracy. In other word, fixed-point representation of a real number compromises arithmetic precision for hardware simplicity. The higher the number of bits are chosen, the more complicated the hardware is, which results in more accurate representation of the model. Hence, numbers of significant fixed-point simulations are performed to determine the minimum number of bits required to maintain the same performance as of floating point design.

When the reliability of the fixed-point design is proved, by knowing the number of assigned bits, it is time to generate Register Transfer Level (RTL) module. We took advantage of an intermediate tool i.e. System Generator for DSP. One of the key features of this tool is the ability to generate RTL module of the corresponding System Genarator model. System Generator is part of Xilinx Xtreme DSP solution which combines state-of-the-art FPGAs, design tools and intellectual property cores. It is also possible to import HDL modules directly into Simulink-based design and co-simulate them using ModelSim. After modeling, and simulating the RTL design the final step is to synthesize and place &route the design.

Last step of the design flow is synthesis. In this project, we performed bottom-up synthesis flow. In other words, each block is synthesized separately and imported as a black-box into an upper-level design. The level of hierarchy is maintained up to top-level design. For each block, a designated test bench is written and the module is simulated after synthesis i.e. gate level simulation. Eventually, top level design is mapped to the

target FPGA chip. This task is called place&route and the final report generated by the tools gives us the exact timing report and hardware cost. If this final report does not meet the constraints, the design should be changed and RTL module should be re-generated with a new design criteria.

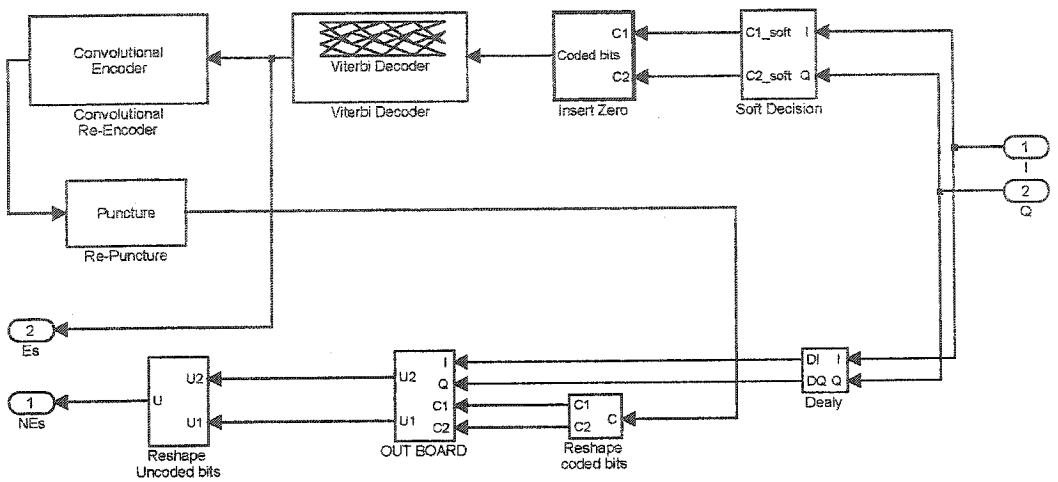
4.2 Multi-rate Pragmatic TCM Decoder structure

General block diagram of PTCM decoder for two different rates is depicted in Figure 4-2. It is worth mentioning that structure of the encoder is thoroughly expressed in section 3.3 of this thesis. However, standard does not specify the decoder structure. Therefore, the decoder is developed based on one of the available Pragmatic TCM structures given by Zehavi and Wolf approach [8].

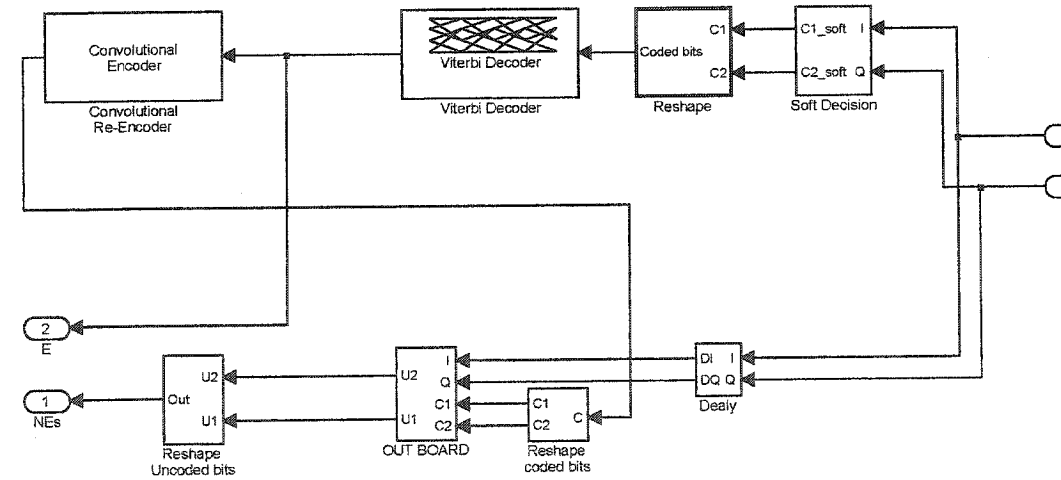
It is well known that with puncturing a de facto rate $1/2$ convolutional code, encoder with different rates can be achieved. These different rates can be decoded with the same viterbi decoder as of rate $1/2$. This is illustrated for two different rates of punctured and non-punctured $3/4$ and $7/8$, respectively.

Each symbol in PTCM encoder consists of some coded-bits and uncoded-bits. Probability of detecting un-coded and coded bits is different in the decoder [8]. This difference could be exploited in some applications where some bits are more important than other bits. In other words, it is proved that the probability of incorrectly decoding coded-bit becomes insignificant comparing to un-coded bits. Thus the overall probability of error reduces to the probability of outboard decision error [20],[21]. The outboard decision logic detects the most likely un-coded bit based on received symbol vectors along with the companion re-encoded bits. A computer simulation is carried out for both rates. The simulation results are depicted for rate $3/4$ in Figure 4-3. In this figure, BER of

the uncoded bits tends to the expected bit error rate at higher SNR. In other words, BER



(a)



(b)

Figure 4-2: Decoder structure of PTCM (a) rate *3/4* (b) rate *7/8*

of uncoded bits is reduced to the probability of an incorrect outboard decision provided that the estimated coded bits are correct. Note that In Figure 4-2, PTCM is fed by two components of the Cartesian signal space i.e. In-phase and quadrature. They represent real and imaginary part of signal vector, respectively. Since both I and Q have the same mapping, in the decoder both branches have the same structure. Received signals fed to

the decoder, passes through soft decision logic block. Since viterbi decoder requires soft decision input to be quantized, a reasonable approach is to quantize the signal set space to different levels and assign a soft decision weight to each point. Both I and Q components are mapped to nearest soft decision level. Decoder will perform well as long as the soft decision inputs are reasonably accurate indication of the coded bit likelihoods. Computer simulation proved that a 3-bit soft decision with 8-level of quantization is accurate enough [1]. Soft decision for 8-level quantization is depicted in Figure 4-4.

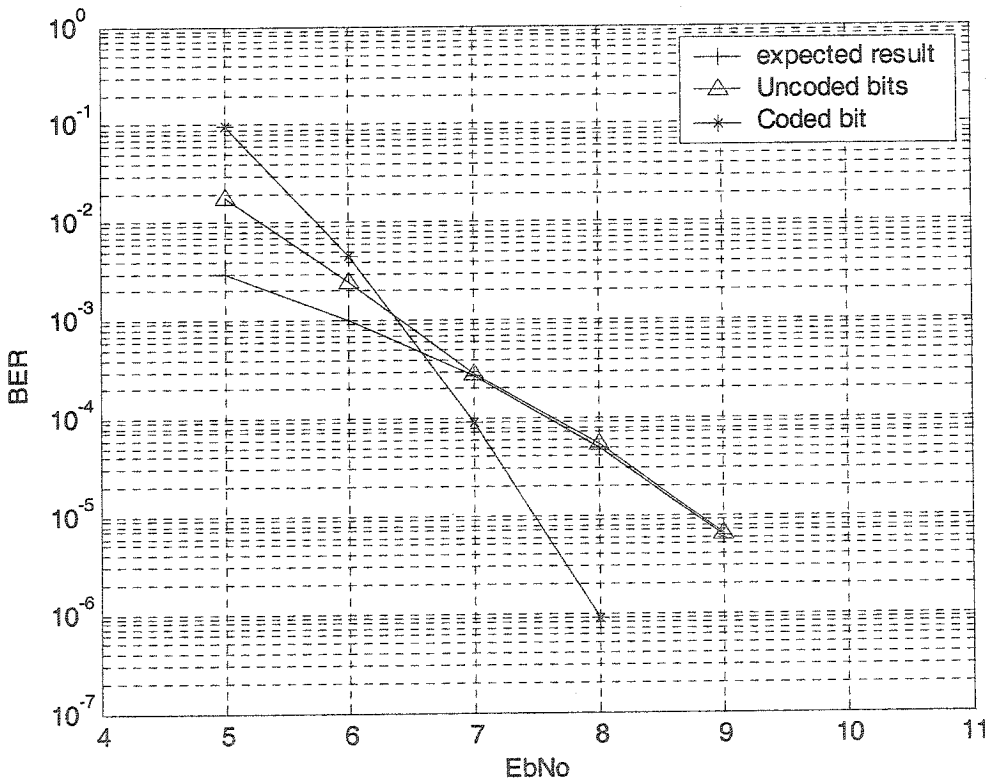


Figure 4-3: Simulation results of rate *3/4* PTCM

Soft decision weights are fed to the viterbi decoder. This yields to E (encoded bit), as it is shown in the Figure 4-2. Output of the viterbi decoder is re-encoded to generate estimated coded bits. This estimated coded bits together with signal space information, is

used to estimate the uncoded bits, on a symbol-by-symbol basis. This is the function of outboard decision logic.

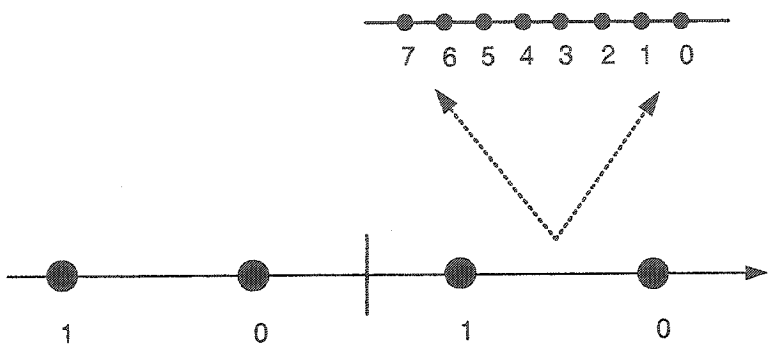


Figure 4-4: Soft decision assignments

Outboard decision logic requires coded sequence as input to set the correct threshold. Depending on the coded sequence there are two threshold levels to be chosen. They are depicted in Figure 4-5. When the threshold level is assigned, a threshold decision is needed to identify the most likely uncoded-bit for each symbol. For example, if viterbi decoder determines coded bit as C=0 and the signal vector is '1.6' in Figure 4-5, then outboard determines the most likely uncoded-bit as U=0.

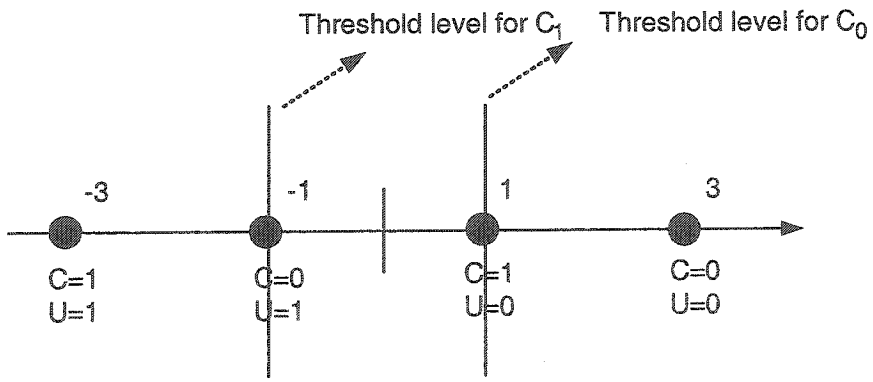


Figure 4-5: Outboard decision logic threshold levels

4.3 Digital filters

Digital filters are categorized into two classes known as finite-length impulse response (FIR) and infinite-length impulse response (IIR). Advantage of FIR filters over IIR filters is that they are stable and can be designed to have linear phase. It makes FIR filters attractive for digital communication applications where non-linear phase distortion is not tolerated [16]. Therefore, two required digital filters, pulse-shaping and matched filters, are realized as an FIR filter. FIR digital filters can be designed easily, by knowing the filter specifications. Then, it is implemented using adder, multiplier and registers.

FIR filter design and implementation techniques are explained in this section, and then ploy phase implementation technique is presented as a practical approach in our design.

4.3.1 SQRC filter design and implementation techniques

There are two considerations to design the required digital filters, pulse-shaping and matched filters i.e. sample rate and time span of the filter impulse response [15]. These two factors govern number of filter coefficients and thus filter order. For example, if time span of the SQRC sets to be $S=7$ symbols and it up-samples the signal by a factor of $L=4$, then the resultant filter order is $N=28$. Obviously, trade-off exists between number of taps (filter order plus one) and filter impulse response. More over-sampling yields a more accurate frequency response. However, more filter coefficients are required for the given time span. A general block diagram of pulse-shaping and matched filter is depicted in Figure 4-6. Note that symbols prior to pulse-shaping filter are required to be up-sampled by factor of two or preferably more [15]. In other words, pulse shaping filter

works as an interpolator filter. On the other hand, matched filter followed by a downsampler i.e. matched filter works as a decimator filter.

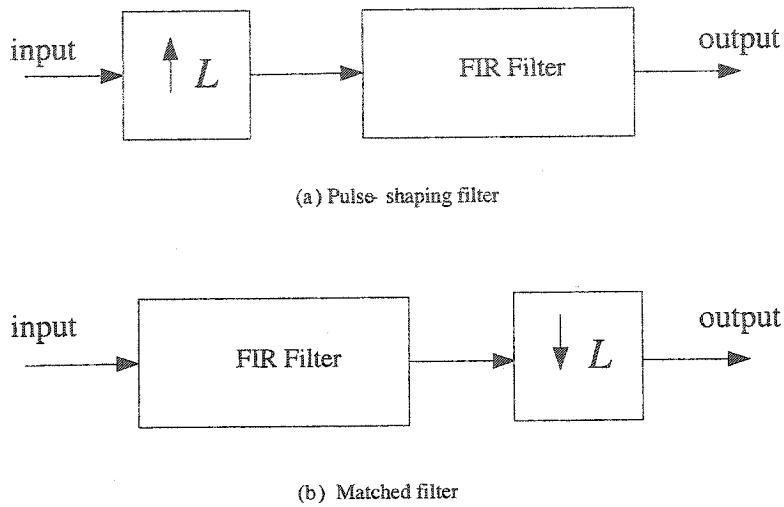


Figure 4-6: A general block diagram of (a) pulse-shaping and (b) match filter

Knowing these facts, design and implementation of an SQRC can be summarized in the following steps:

- Find impulse response of the required SQRC (i.e. with appropriate roll-off factor). This can by done, by using Equation (2.8) or many digital filter design tools available such as Matlab.
- Choose sampling rate and time span of the filter impulse response found in the first step. This can be done by the aid of computer simulations i.e. compare the simulation results of the system with theoretical system performance.
- Repeat the previous step with different parameters until it meets system requirements.

The designated SQRC filter with given roll-off factor (i.e. $\alpha = 0.35$) is simulated with different parameters. The simulation results are then compared with theoretical BER of *16*-QAM modulation scheme. After extensive simulation, up-sampling rate and symbol span time are chosen to be four and seven respectively. It provides an accurate

spectral shape of the required filter. The BER curve versus E_b/N_o with up-sampling $L=4$ is depicted for different span time in Figure 4-7. It is illustrated that for selected sample time, the performance is acceptable for symbol span $S=7$ and by increasing symbol span to 9 no performance improvement is achieved. The same simulation is done for up-sampling rate $L=5$ and 6, with different symbol span times. Comparing all simulation results, the best performance with optimum hardware implementation complexity (number of coefficients) turns out to be $L=4$ and $S=7$. The filter coefficients are depicted in Table 4-1.

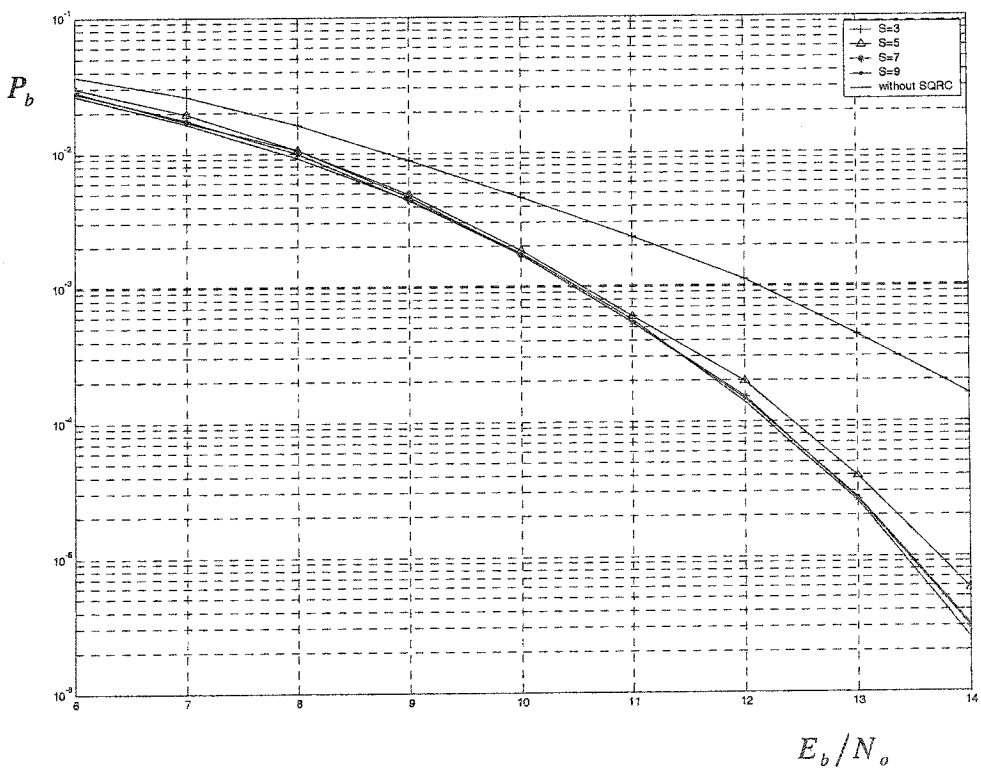


Figure 4-7: Simulation result of SQRC filters with $L=4$ and different span time

N	0	1/-1	2/-2	3/-3	4/-4	5/-5	6/-6	7/-7
H[n]	0.5478	0.4786	0.03039	0.1034	-0.0423	-0.0943	-0.0676	-0.0110
N	8/-8	9/-9	10/-10	11/-11	12/-12	13/-13	14/-14	
H[n]	0.0286	0.0327	0.0128	-0.0074	-0.0127	-0.0048	0.0048	

Table 4-1: SQRC filters coefficients

As shown in Table 4-1 all coefficients of the filters have floating point value. Practically, FIR filter is rarely realized with floating point representation of its coefficients. In other words, particular number of bits (fixed-point design) is chosen to represent filter coefficients. The fixed-point simulation results are presented at the end of this chapter.

There exist different structures to realize an FIR filter such as Direct Form, Linear Phase and polyphase realization. Direct Form or Transversal Structure is one of the simplest structures depicted in Figure 4-8. This is in fact realization of Convolution operation. In other words, filtering by using simple operations (mutiply/add/delay) is

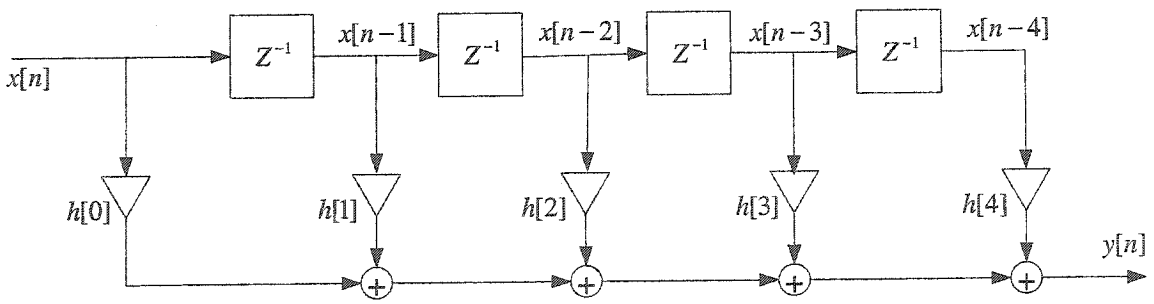


Figure 4-8: Direct form FIR structure [17]

realization of convolution of sampled input signal ($x[n]$) with sampled impulse response of the filter ($h[n]$). It can be shown by Equation (4.1):

$$y[n] = \sum_{k=0}^{N} h[k]x[n-k] \qquad (4.1)$$

where for the given filter with $N=4$ yields:

$$y[n] = h[0]x[n] + h[1]x[n-1] + h[2]x[n-2] + h[3]x[n-3] + h[4]x[n-4]$$

An N^{th} order FIR implemented in Direct Form requires $N+1$ multipliers and N two-input adders [17].

4.3.2 Polyphase realization of pulse-shaping filter

In the transmitter prior to pulse-shaping filter, symbols are up-sampled by factor of L thus only every L^{th} sample entered to the filter is non-zero. This is depicted in Figure 4-6. This implementation has major drawbacks as follows:

- FIR filter followed by up-sampling has to operate at clock rate of L times faster than that of input data stream.
- Filter coefficients are multiplied by $L-1$ zeros stuffed by the up-sampler. It results in extra unnecessary computation load.

Polyphase decomposition is remedy to these drawbacks. In this approach, the operation of filtering and up-sampling is interchanged, then filter is decomposed into L parallel filters each followed by an up-sampler. Finally each decomposed branch is followed by chain of delay element prior to addition. A block diagram of the Polyphase structure is depicted in Figure 4-9.

Consider an arbitrary FIR filer with impulse response $h[n]$, sub-filter decomposition of this filter can be shown as:

$$E_k[n] = h[nL+k]; \qquad 0 \le k \le L-1 \qquad (4.2)$$

Therefore, $h[n]$ can easily be reconstructed from Equation (4.2):

$$h[n] = \sum_{k=0}^{L-1} E_k(n-k) \tag{4.3}$$

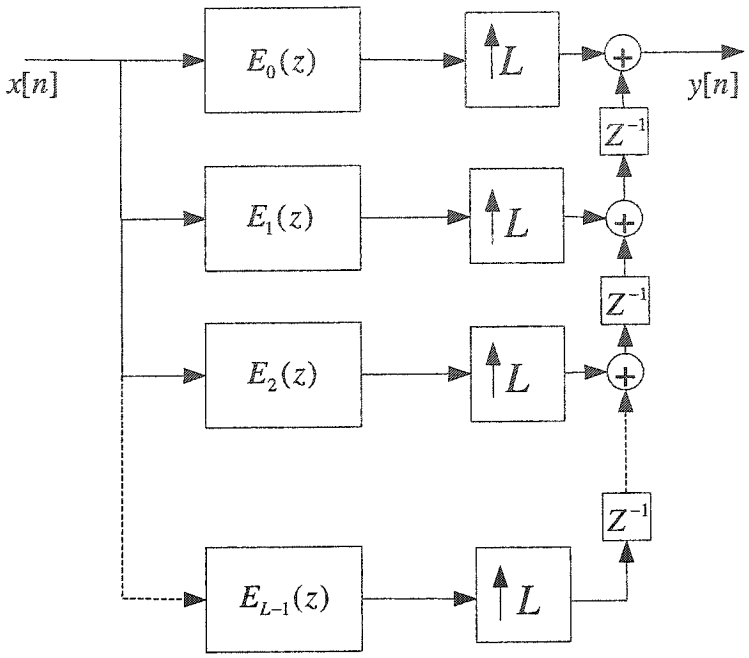


Figure 4-9: Polyphase decomposition of interpolator filter

For example, in this thesis pulse-shaping filter is designed with $L=4$, thus it is decomposed of four sub-filters with filter coefficients as follows:

$$\begin{aligned} E_0[n] &= h[j]; & j &= 0,4,8,12,16,20,24,28 \\ E_1[n] &= h[j]; & j &= 1,5,9,13,17,21,25,29 \\ E_2[n] &= h[j]; & j &= 2,6,10,14,18,22,26 \\ E_3[n] &= h[j]; & j &= 3,7,11,15,19,23,27 \end{aligned} \tag{4.4}$$

To illustrate the advantage of polyphase structure, consider an FIR filter of length N where input symbols are clocked at rate of $1\ Symb/Sec$. Transversal structure requires NL multiplications and $(NL-1)$ additions per second. On the other hand, Figure 4-9

requires $L(N/L)$ multiplications and $L(N/L-1)$ additions per second. Therefore, polyphase structure results in computation efficiency by rearranging the operations so that the filtering is done at low sampling rate.

4.3.3 Polyphase realization of matched filter

Another important application of Polyphase structure is implementation of a filter whose output is followed by downsampling. In Figure 4-6 matched filter computes output samples but only one of each L samples is retained. Intuitively, an efficient implementation is expected to exist which does not compute the samples that are thrown away [18]. This kind of structure is polyphase decomposition of the matched filter which is depicted in Figure 4-10.

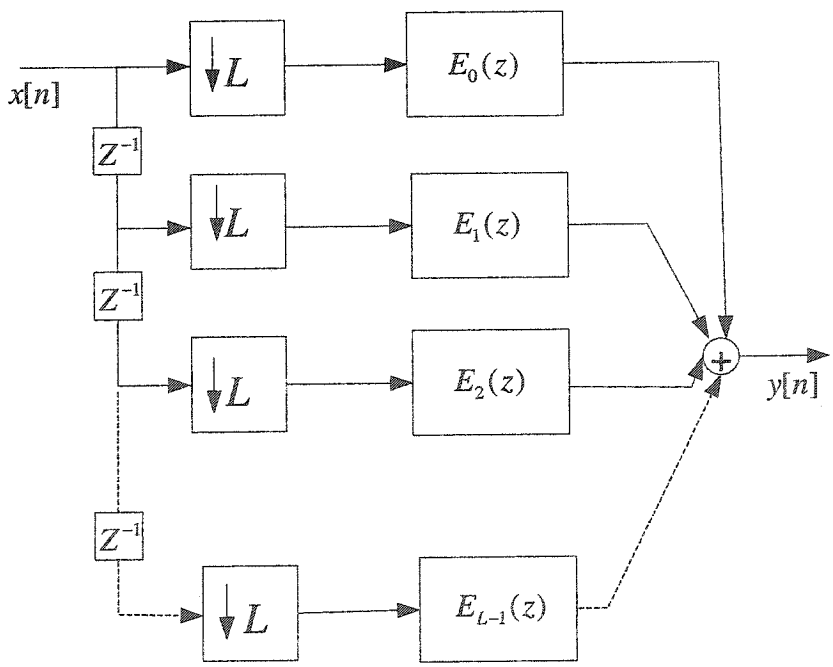


Figure 4-10: Polyphase implementation of decimation filters

Like pulse-shaping filter, the FIR filter is decomposed of L sub-filters shown in Equation (4.2). By interchanging downsampling and filtering operation sub-filters operate only on output data of downsampler. To illustrate the advantage of Polyphase structure, consider an FIR filter of length N where input symbols are clocked at rate of $1\,Symb/Sec$. Transversal structure requires N multiplications and $(N\text{-}1)$ additions per second. On the other hand, polyphase structure requires (N/L) multiplications and $(N/L-1)(L-1)$ additions per second.

4.4 Simulation result of the modem

In order to verify performance of the design, it should be simulated. One of the factors that illustrate performance of a communication system is BER curve versus E_b/N_o . The simulation results are compared with the performance required by the standard. A general block diagram used for simulation is depicted in Figure 4-11. A sequence of random binary numbers is fed to the modem. At output of the transmitter i.e. pulse shaping filter, both I and Q components of the transmitted signal are added with AWGN. It simulates our memory-less noisy channel. At the receiver, this noisy received signal is fed to the matched filter. Finally, at output of the RS decoder, decoded bit sequence is compared in a bit-by-bit basis to the originally generated random binary sequence. Note that received bit sequence has certain amount of delay with respect to the transmitted bit. Task of error calculator block is to count number of uncorrected bits (N_e) and eventually calculate probability of bit in error (P_b). P_b is determined as soon as N_e reaches a certain threshold. The larger this threshold is chosen, the more accurate the

simulation result will be, which requires longer simulation time. Probability of the uncorrected bits is given by Equation (4.5):

$$P_b = \frac{N_e}{Total \quad bit \quad transmitted} \tag{4.5}$$

Modem is simulated with different power of noise in Simulink environment to obtain performance curve.

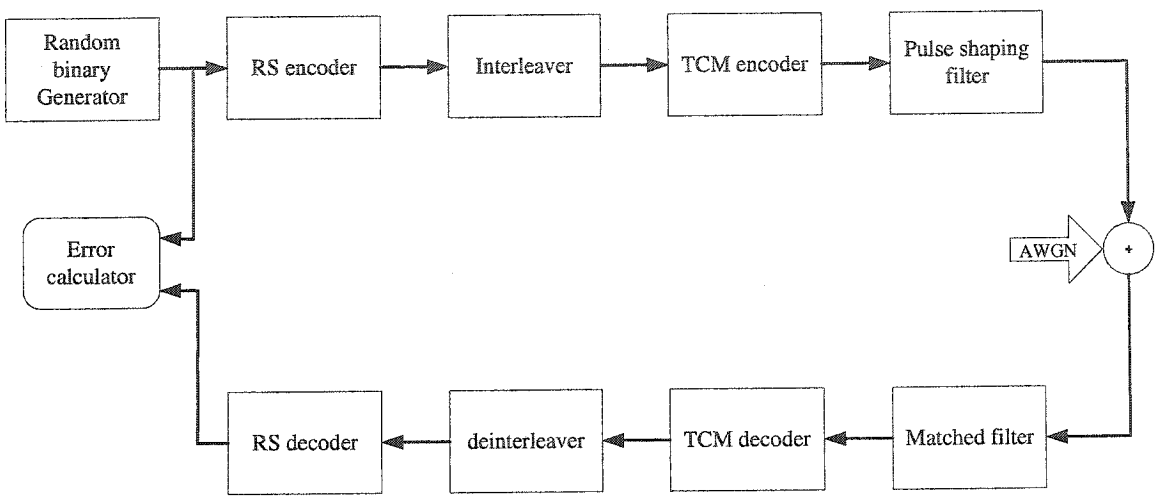


Figure 4-11: Block diagram of the Modem

In Table 4-2, IF-Loop performance of the modems for two different rates is depicted based on DVB. It is shown that when modem functions at certain BER, required E_b/N_o should not exceed the value that is given in Table 4-2. In this table, Spectral Efficiency can be calculated by Equation (4.6):

$$Spec.Efficiency = \frac{188}{204} * innner\ code\ rate * log_2(M) \tag{4.6}$$

where M is number of different symbols in M-ary signaling. In our implementation it is equal to *16*.

Inner Code Rate	Spectral Efficiency (bit/symbol)	Modem Implementation Margin (dB)	Required E_b/N_o for BER = $2*10^{-4}$ before RS, QEF after (dB)
3/4	2.76	1.5	9.0
7/8	3.22	2.1	10.7

Table 4-2: Performance of the modem with different rates [14]

Error performance curve of the PTCM modem with two different rates is depicted in Figure 4-12. In this figure, simulation results are compared with the expected value given by Table 4-2. Noting that in Table 4-2 required E_b/N_o is given for the useful-bandwidth (R_u). In other words, the noise bandwidth increase and modem implementation margin should be considered i.e. required E_b/N_o is decreased by $10\log(188/204) \approx 0.36\,dB$ due to RS outer encoder. Also modem implementation margin has to be taken into account. Modem implementation margins which increase with spectral efficiency, are adopted to confront higher sensitivity associated with two schemes [14]. For example, for a modem performing at rate of $3/4$, required E_b/N_o in order to have the acceptable performance should not exceed *9-.036-1.5=7.14* dB. It is depicted for both rates at Figure 4-12.

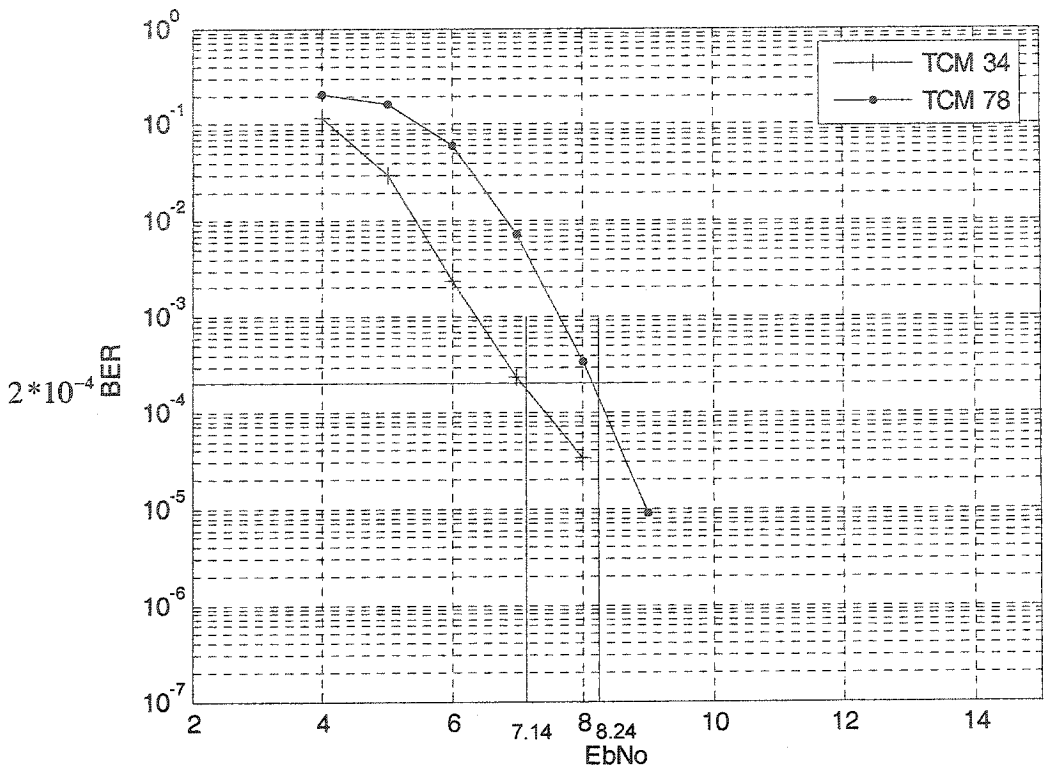
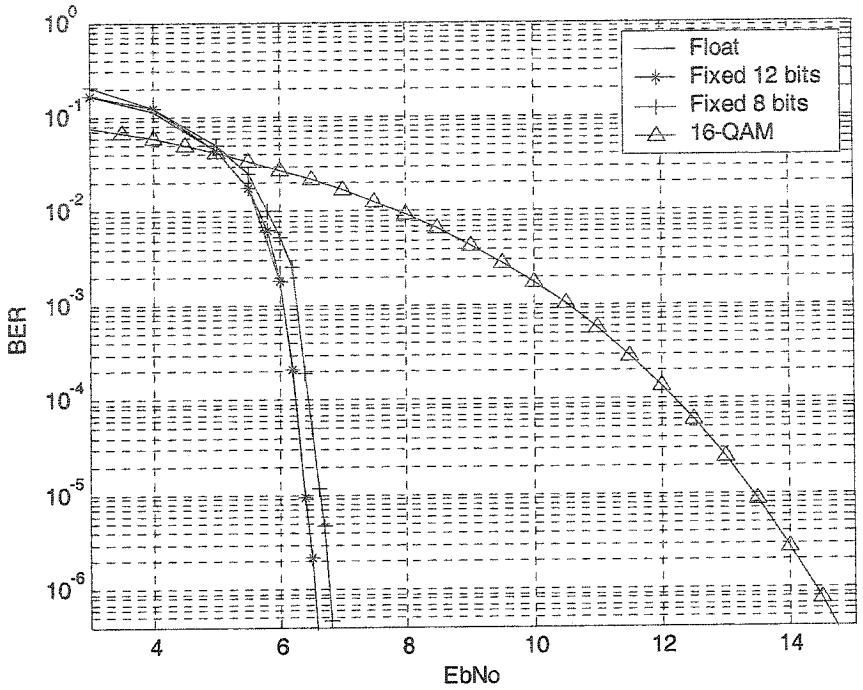
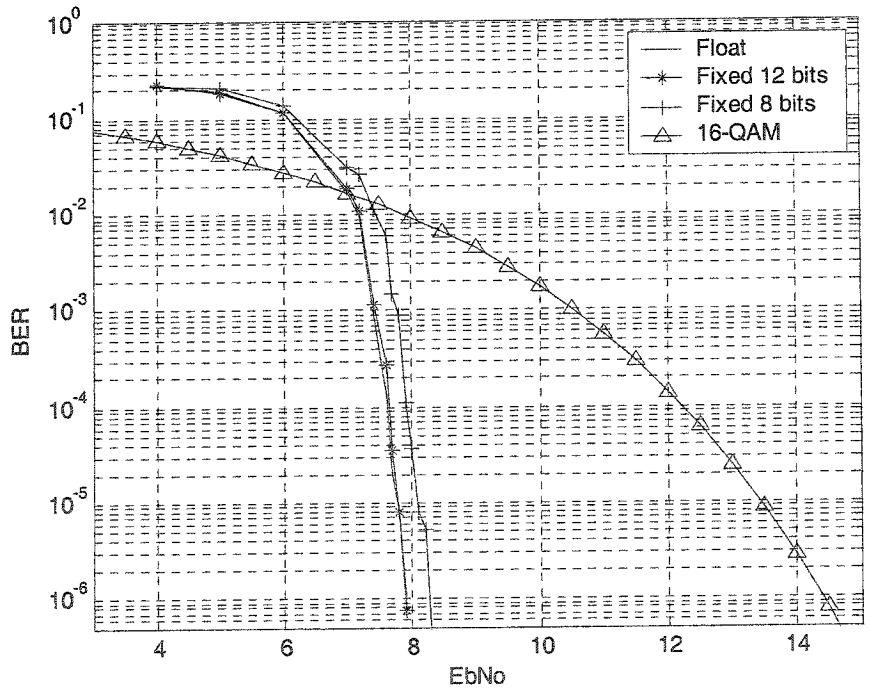


Figure 4-12: Comparison of the error performance curves with expected values

When the functionality of the PTCM encoder and decoder is verified, it is time to authenticate the performance of the complete design. System with concatenated RS code and PTCM is simulated for both rates. It is depicted in Figure 4-13. In Figure 4-13-a, performance curve of the modem with PTCM rate 3/4 as an inner code, and convolutional interleaver and RS encoder as an outer code is illustrated. Fixed-point simulation is also performed for two precision i.e. 8-bit and 12-bit. It comes out that *12*-bit Fixed-point model is accurate enough. The same simulation is carried out for PTCM of rate 7/8 , as an inner code. The performance curve is sketched in Figure 4-13-b.



(a)



(b)

Figure 4-13: System simulation (a) rate *3/4* PTCM (b) rate *7/8* PTCM

Chapter five

5 Implementation of the Modem

When fixed-point model of the system is determined i.e. fixed-point simulation result is obtained, modem can be implemented into hardware. Field programmable gate array (FPGA) is selected as a practical solution. FPGA is a custom chip that allows a very specific implementation of a digital circuitry to be implemented. FPGA chips are programmed with a binary bit file that thoroughly describes the feature of digital circuitry. This bit file is generated by a synthesis tool after place&route.

The Xilinx Virtex-II family is platform FPGA developed for high performance designs that are based on IP cores and customized modules. The leading-edge *0.12μm/0.15μm* CMOS *8*-layer metal process which is optimized for high speed and low power is used for VLSI implementation of Virtex-II family. As shown in Figure 5-1, the Virtex-II architecture is comprised of Input/Output Block (IOBs) and internal Configurable Logic. Programmable IOBs provide interface between Configurable Logic Blocks (CLB) and package pins. The internal configurable logic includes four major elements: CLB blocks, Block SelectedRam, Multiplier blocks and DCM blocks [33].

In this chapter, Digital Clock Manager (DCM) is demonstrated as an embedded component of Xilinx Virtex II. Xilinx System Generator is introduced as a practical tool which is employed to generate and simulate HDL model of the design. Modem implementation methodology, hardware design flow and data synchronization is also proposed. Finally, timing and area report of the synthesis tool is tabulated.

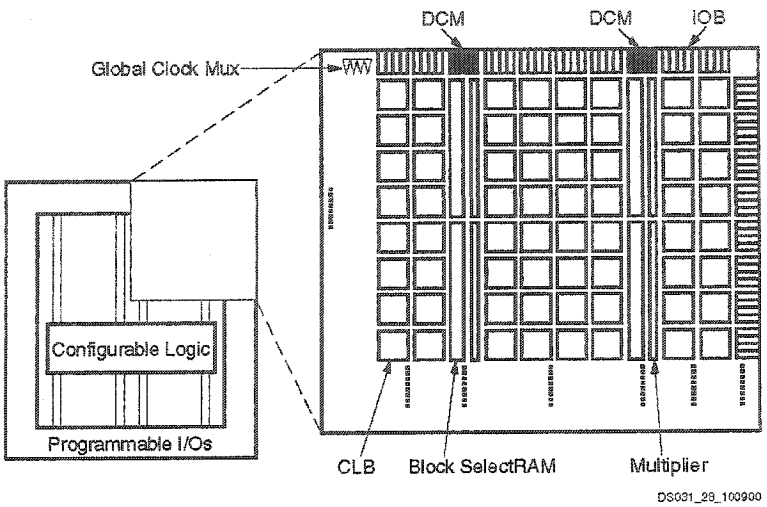


Figure 5-1: Virtex-II architecture overview [33]

5.1 Digital Clock Manager in Virtex-II FPGA

Virtex-II devices have *16* clock input pins. Each input pad should either drive Global Clock Multiplexer Buffers (BUFGMUX) or Digital Clock Manager (DCM). It is depicted in Figure 5-2. Input clock pin cannot directly drive any logic in the device. BUFGMUXs are used to distribute low-skewed clock to synchronous logic elements in the device. As it is shown in Figure 5-3, global clock multiplexer buffers have two input clocks i.e. I0 and I1, output clock (O) and select input (S). The select input chooses between the input clocks without generating glitches.

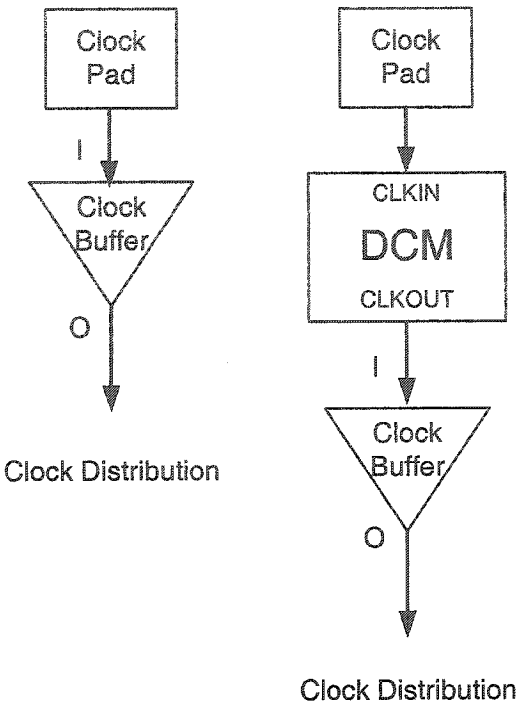


Figure 5-2: Virtex-II clock distribution configuration [22]

Two input clocks of BUFGMUX can be connected to any asynchronous or synchronous clocks. Multiplexer selects I0 when S is low and I1 is distributed to its output when S is high.

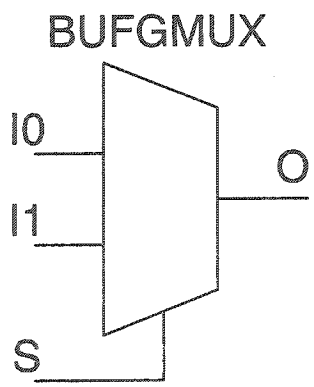


Figure 5-3: Virtex-II BUFGMUX Function [22]

DCM is a fully digital embedded component of Virtex-II devices that offers wide range of clock management features such as:

- Clock De-skew: DCM generates new system clocks which are phase-aligned with the input global clock.
- Frequency Synthesis: DCM can synthesize wide range of the output clock frequency i.e. a very flexible output clock frequency by multiplication and division of input global clock.
- Phase Shifting: DCM provides a very dynamic phase shift control.
- EMI Reduction: DCM provides Electromagnetic Interface reduction by broadening the output clock spectrum.

In this thesis, the first two features of the DCM are used in order to distribute required clock frequency to each component of the design. A Block diagram of the input and output ports of the DCM is illustrated in Figure 5-4.

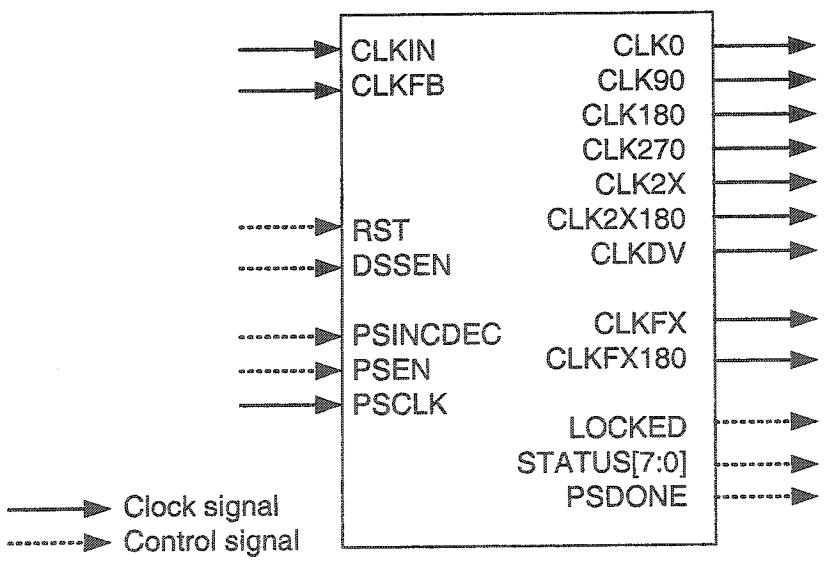


Figure 5-4: input and output ports of the DCM

Virtex-II device has up to 16 embedded DCM depending on the size of the chip. Up to four DCM clock outputs can drive global clock multiplexer buffers, simultaneously. As shown in Figure 5-4, the DCM has three control signals such as: RST, LOCKED and STATUS. RST is an active high reset input. This input signal is asynchronous and it should be held high *2ns*. It takes approximately *120ns* for the DCM to achieve lock after a reset. In other words, when the output clock signals of the DCM become stable, LOCKED control signal becomes high.

DCM not only provides a de-skewed output clock, but also provides output clocks with different frequency. CLK2X output provides twice as high as input clock frequency in its CLK2X output. The clock divide output (CLKDV) of the DCM, also provides a lower version frequency of the input clock. The list of valid divider is given in the data sheet. These features are automatically set the duty cycle of the output clock to *50/50* if it is applicable [22].

One of the interesting capabilities of the DCM is frequency synthesis. CLKFX output provides any function of input clock frequency described by *M/D*; where *M* is multiplication factor and *D* is division factor. *M* and *D* can each range from *1* to *4096*. However, *M* and *D* should be chosen such that for the certain input frequency, output frequency of the sensitizer will be in the range given in the data sheet. Output signal of the CLKFX also automatically has *50/50* duty cycle. As an example, consider input frequency = *55 MHZ*, *M=333* and *D=100*. Note that *M* and *D* do not have common factors. The output frequency is *55* 333/100=183.15 MHZ*, which is in the functional frequency range of the DCM given by Xilinx Virtex-II data sheet. However, both *55 MHZ *333= 18.315 GHZ* and *55 MHZ/100=550 KHZ* are far outside the practical

frequency range. A complete functionality, configuration and pin description of the DCM is given at Xilinx Virtex-II data sheet [22].

5.2 System level design using System Generator

System Generator is a tool that extends Simulink capabilities to facilitate hardware design and accelerate system simulation. A model designed in the System Generator is translated to HDL by just pushing a button. The designated model can also be simulated in System Generator platform much faster than any hardware simulator. System Generator models are bit-true cycle-true i.e. each fixed-point value in System Generator design translates to the same Standard Logic Vector (Stdlogicvector) value in hardware. This means System Generator is bit-true. Moreover, note that System Generator models are discrete thus signals are sampled with an associated sample period. When the design is translated to hardware, its clock period is guaranteed to be multiple of the System Generator sample period. This means that System Generator is cycle-true. Therefore, system designer can determine the behavioral of the hardware and make any necessary changes before generating the hardware.

Timing and clock management is an essential part of any design process, therefore System Generator clock managements is explained thoroughly in the following section. A novel methodology that improves the speed of the top level design is also proposed.

5.3 Timing and clock managements in System Generator

It is important to understand how System Generator manages timing and clock distribution. Time management becomes more significant when the design is multi-rate i.e. somewhere in the design components such as up/down sampler, serial to parallel and parallel to serial converters are used. In our design, due to the special structure of the Pragmatic TCM encoder/decoder and pulse shaping/matched filters, the design is multi-rate. Simulink platform manages clock distribution automatically in the simulation. Since "Simulink sample period" is associated with "hardware clock period", when HDL model is generated by System Generator, different clock frequencies are required. In order to reduce clock-skew and use one system clock (SYS_CLK), System Generator associates input clocks (CLK) of each component with companion clock enable (CE). In other words, the design is driven by single system clock and different clock enable signals that provides different clock frequencies required for the design. Note that the period of CE signal is integer multiple of system clock. For example, consider a design that requires three different rate of 1/2, 1/3 and 1/4 of SYS_CLK . The designated CE signals are CE2, CE3 and CE4 which are depicted in Figure 5-5. Although System Generator generates CLK signals required for the design, optimum CLK designation can be achieved manually. The manual CLK distribution increases clock frequency up to *12* times in our implementation.

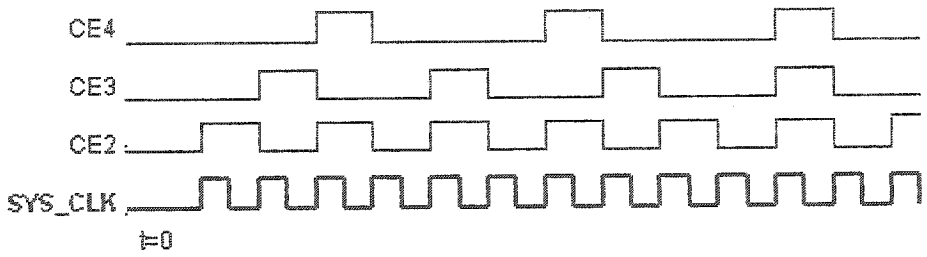


Figure 5-5: CE signals associated with three different rates of *1/2*, *1/3* and *1/4* [23]

A general block diagram of the proposed clock management is shown in Figure 5-6. The idea is based on the System Generator approach with some modifications. In the System Generator based design there is only one DCM that distributes required low-skew ed clock signals. An HDL model generated by the tool is also driven just by CLK0 output of the DCM. Thus the clock synthesis capability of the DCM component is remained unused. Although Virtex-II family has up to *16* DCMs, the tool cannot take advantage of all DCM components available in the chip to achieve optimum clock distribution. These drawbacks restrict multi-rate designs more where a very low clock frequency is chosen by the tool to drive top level design. In other words, input clock is divided by the greater common divisor of all different clock rates required in the design. Therefore, the more different rates the design has, the less frequency is assigned to the top-level deign. In this thesis, clock management is performed manually. Clock distribution is divided to three levels of hierarchy such as: Design, <Design>_CLK_Wrapper and <Desgin>_DCM_Wrapper as shown in Figure 5-6. In this figure, for simplicity one module on each level is illustrated. However, in our design, the top level design consists of number of blocks such as Pragmatic encoder/decoder, pulse shaping and matched filters etc. and depending on the design structure; one or two DCMs are used. As it is illustrated in Figure 5-6, task of CLK_driver is significant. It provides the required CE

signals associated with the input SYS_CLK to construct large range of clock frequency required in the design. The input clock with the companion *CE* signals provide all clock frequency that the design required in designated level of hierarchy. It is worth mentioning that the CLK_driver blocks consist of simple counter that divide input clock and construct required CE signals. For example, in Figure 5-7 block diagram of the CLK_driver of Pragmatic TCM encoder of rate 3/4 is depicted. Three CE signals are constructed such that their clock frequency is 1/3, 1/8 and 1/24 of the SYS_CLK.

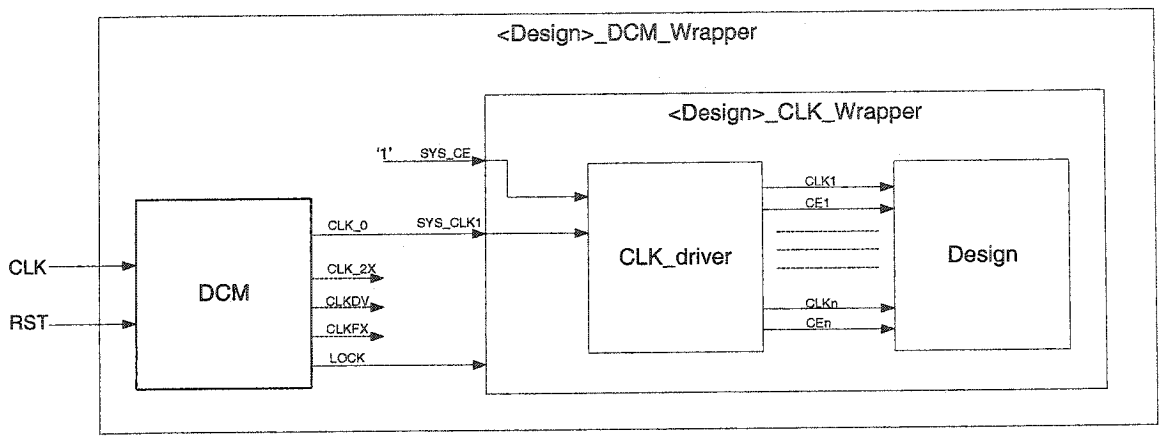


Figure 5-6: General block diagram of system clock distribution using DCM

The input clock signal of the CLK_driver block is provided by DCM. Digital clock manager provides different low-skewed synchronized clock signals. In the proposed approach, we can take advantage of all outputs of the DCM if necessary. It is also possible to employ as many DCMs as available in the target FPGA chip. In this thesis up to two DCMs are exploited to achieve the highest clock frequency possible. Since in this project two different Pragmatic Codes are implemented, clock distribution of the top level designs is explained separately in the following subsection.

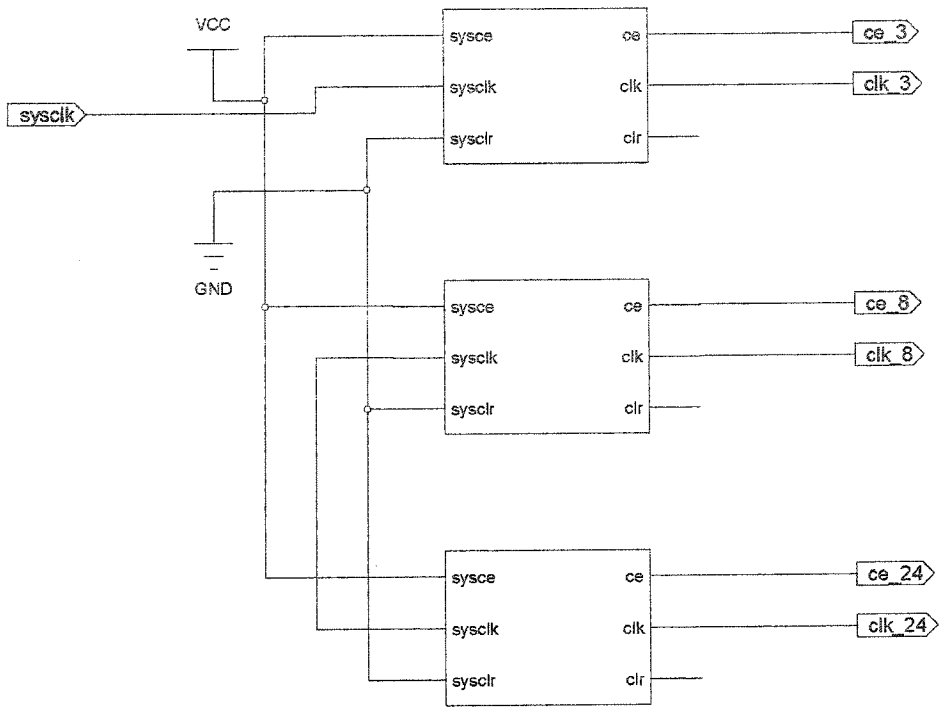


Figure 5-7: CLK_driver block diagram of Pragmatic TCM encoder rate *3/4*

5.3.1 Clock distribution of top level design with rate *3/4* PTCM

RTL Schematic of the top level design with rate *3/4* Pragmatic TCM is depicted in Figure 5-8. As mentioned earlier, top level design consists of eight black boxes such as RS encoder/decoder, convolutional interleaver/de-interleaver, Pragmatic TCM encoder/decoder and pulse shaping/matched filters. The functionality of each black box is verified in the lower level of hierarchy. The time constrain is also verified by performing place&route and gate level simulation. Therefore the maximum input clock frequency of each block is determined. It is also essential to understand that the throughput of different blocks is not the same. In other words, due to block structure, the block input rate might be different from that of output. Therefore, the adjacent block should perform at a different rate. In order to perform the clock distribution manually, the structure of each

black box should be well known and all CE signals should be considered. Note that the combination of each CLK with its CE constructs the genuine clock signal that the inner entity is driven. The maximum input clock that each component can be driven is tabulated in Table 5-1. The table is determined by Xilinx Synthesis Tool (XST) and all frequencies are from the post-place&route timing report. By knowing the maximum frequency of each building block, time constraint of the top level can be determined. Based on Table 5-1 and the inner structure of each block, clock distribution is performed as illustrated in Figure 5-8. Due to the fact that, the top level consists of up-samplers and parallel to serial converter, bottle neck of the design cannot be identified just by Table 5-1. In other words, Table (5.1) provides enough information for each sub-block when it functions individually. However, the aim is to design the modem such that it transmits and receives real time date at as high bit rate as possible.

Component	Minimum Period	Maximum Frequency
RS encoder	5.646 ns	177.116 MHz
RS decoder	9.410 ns	106.269 MHz
Interleaver	4.935 ns	202.634 MHz
De-interleaver	4.196 ns	238.322 MHz
Pragmatic TCM encoder	5.397 ns	185.288 MHz
Pragmatic TCM decoder	9.810 ns	101.936 MHz
Pulse shaping filter	8.554 ns	116.904 MHz
Matched filter	5.610 ns	178.253 MHz

Table 5-1: Timing report of the Modem with rate *3/4*

By knowing the function of CLK_driver, it is time to elaborate on Figure 5-6 in our design. There are two components that increase the rate in the transmitter i.e. byte to bit converter in the Pragmatic TCM encoder and up-sampler in the pulse shaping filters. Thus the pulse shaping filters function with the highest bit rate i.e. it should be driven

with the highest clock. One simple approach is to drive the design with three different clocks as *1X, 8X* and *32X*, where *X* is the input clock and drive two first blocks with *1X*, Pragmatic TCM with 8X and pulse shaping filters with *32X*. There are two drawbacks for this approach; first each DCM has only one CLKFX that can provide any rational multiple of its input clock. Thus, two DCMs are required to provide two multiples of *SYS_CLK*. Besides, Virtex-II FPGA cannot support 32X of the maximum applicable clock of the *RS* encoder. Another approach is to drive pulse shaping filters with the highest possible frequency and with aid of CLK_driver blocks, drive other blocks with lower clock. This is practical but since CLK_driver itself has time constraint the modem perfumes at low bit rate. In other words, RS encoder is clocked with as low as $1/32X$; where *X* is the maximum functional clock frequency of the pulse-shaping filters. A practical trade off is to drive *RS* encoder, interleaver and Pragmatic TCM with *1X Hz* and pulse shaping filter with *4X Hz*. Inside the <design>_CLK_Wrapper, CLK_driver should be designed such that it provides required CE signals to different sub-blocks within the block. The clock distribution is depicted in Figure 5-9. It is shown that pulse shaping filters are driven with frequency four times as much as other blocks. Input and output rate of each block is also shown bellow each block. In other words, CLK_driver divides *1X* frequency and provides *1/8* of its input frequency to the first three blocks. Since the pulse shaping filter is driven by *4X* clock, in order to have a consistent flow in the transmitter, input clock is divided by *12* instead of *3*. In other words, *4X* clock of the pulse shaping filters is compensated by dividing the input frequency by *12*. This not only ensures consistent bit flow but also provides higher frequency that up-samplers require in the

pulse shaping filters. Note that Pragmatic TCM encoder has a multi-rate structure i.e. output rate of the encoder is increased by $8/3$.

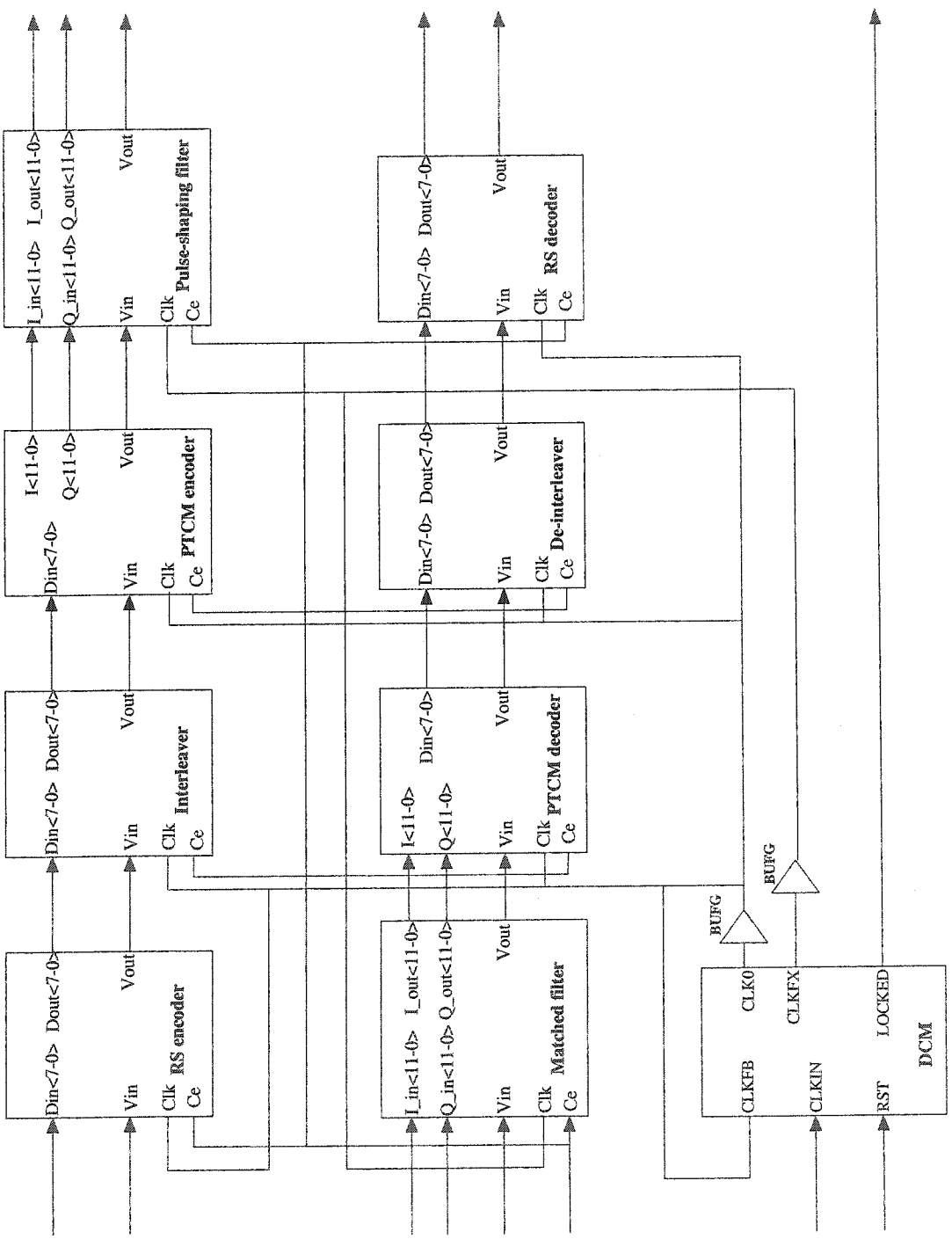


Figure 5-8: RTL schematic of the modem for rate *3/4*

Since in the receiver, the decoder has the same clock structure as its encoder, the clock distribution should be performed similarly. In other words, matched filter is driven by *4X* and Pragmatic TCM decoder, de-interleaver and RS decoder are driven by *1X* Hz.

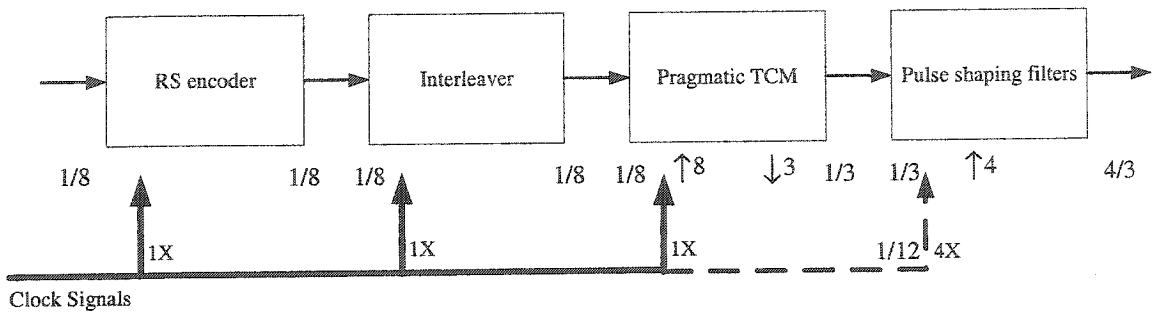


Figure 5-9: Clock distribution of the modem with Pragmatic TCM of rate *3/4*

5.3.2 Clock distribution of top level design with rate *7/8* PTCM

RTL schematic of the top level design with rate *7/8* Pragmatic TCM is depicted in Figure 5-10. Each component is verified by means of behavioral simulation. When the functionality of each block is verified, gate level simulation is performed with the same arbitrary test bench. The time constrain is also verified by performing place&route and gate level simulation. Maximum input clock frequency that each component can be driven is tabulated in Table 5-2. The table is determined by Xilinx Synthesis Tool (XST) and all frequencies are from the post-place&route timing report of the tool. A similar approach to rate 3/4 is exploited to distribute clock signals manually. However, Pragmatic TCM encoder/decoder has different structure than that of rate *3/4*. As shown in Table 5-2, two DCMs are used to achieve the highest throughput in the modem.

Component	Minimum Period	Maximum Frequency
RS encoder	5.646 ns	177.116 MHz
RS decoder	9.671 ns	103.401 MHz
Interleaver	4.649 ns	215.100 MHz
De-interleaver	4.196 ns	238.322 MHz
Pragmatic TCM encoder	7.748 ns	129.065 MHz
Pragmatic TCM decoder	9.185 ns	108.973 MHz
Pulse shaping filter	7.860 ns	127.226 MHz
Matched filter	6.595 ns	151.630 MHz

Table 5-2: Timing report of the Modem with rate *7/8*

In top level design, there are two components that increase the rate in the transmitter i.e. Pragmatic TCM encoder and pulse-shaping filters. Thus the pulse-shaping filters, which function with the highest bit rate, should be driven with the highest clock. In other words, in transmitter data rate is increased in the Pragmatic TCM encoder by 3 and 8. The designated up-samplers in the pulse-shaping filters structure also increase the sampling rate by factor of 4. Therefore, pulse-shaping filter should be driven by a clock 96 times faster than RS encoder. A Comprehensive block diagram of the clock distribution of the modem is depicted in Figure 5-11. As shown, first two blocks in the transmitter are driven by *1X Hz*, however Pragmatic TCM and pulse shaping filters are driven by *3X Hz* and *4X Hz*, respectively. Note that, inside the <design>_CLK_Wrapper, CLK_driver is designed such that it provides required CE signals to different sub-blocks within the block. It is shown that pulse-shaping filters are driven with frequency four times as much as SYS_CLK and Pragmatic TCM encoder is driven with that of three times as much as SYS_CLK. Input and output rate of each block is also shown bellow each block. In other words, CLK_driver divides *1X* frequency and provides *1/8* of its input frequency to the first two blocks. However, Pragmatic TCM is driven by *3X Hz*.

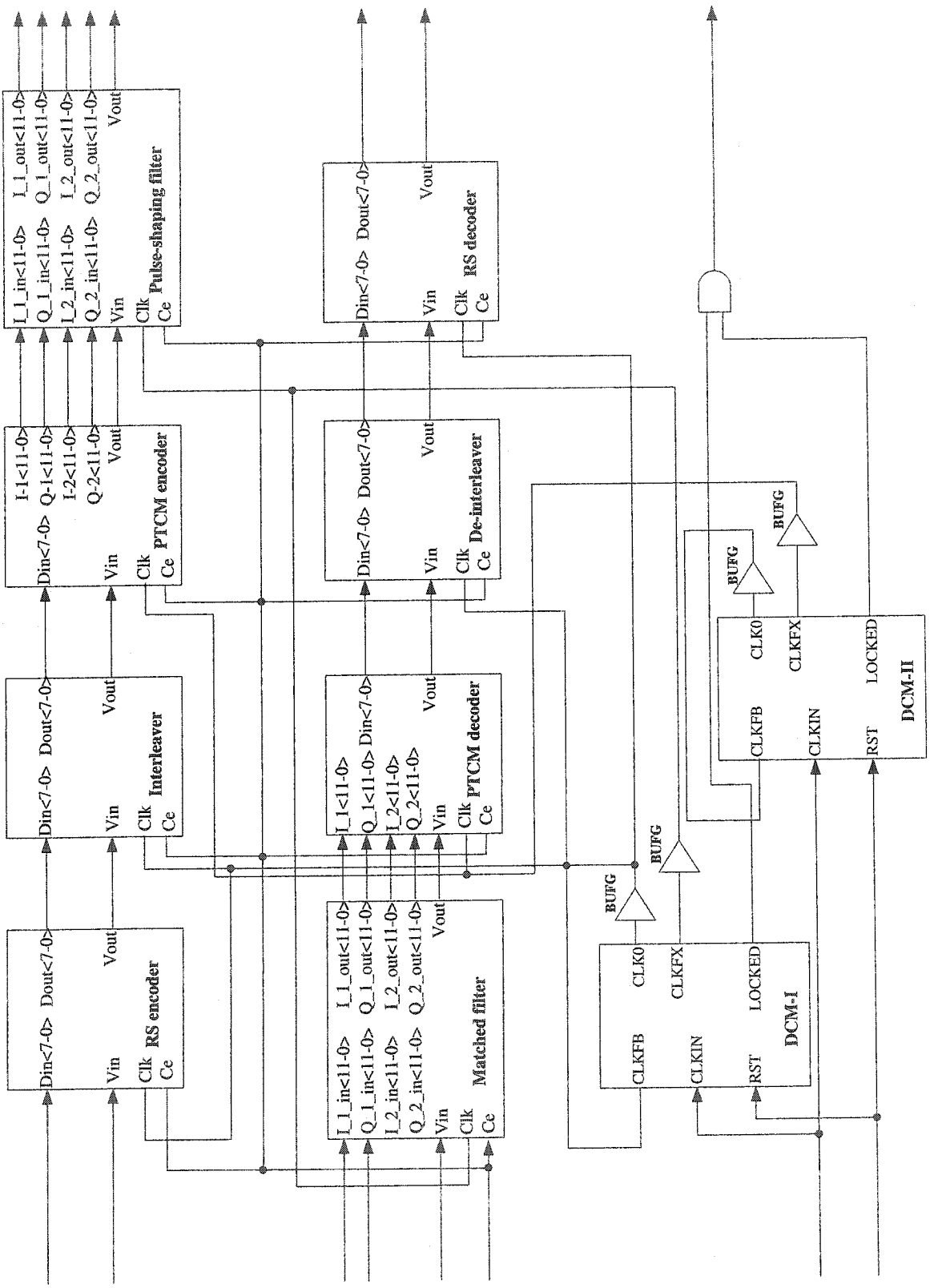


Figure 5-10: RTL schematic of the modem for rate *7/8*

Therefore, CLK_driver at input of the block divides input clock frequency by *24* instead of *8*. This guarantees a consistent flow of data in the encoder. Note that Pragmatic TCM encoder has a multi-rate structure i.e. output rate of the encoder is increased by *8/7*.

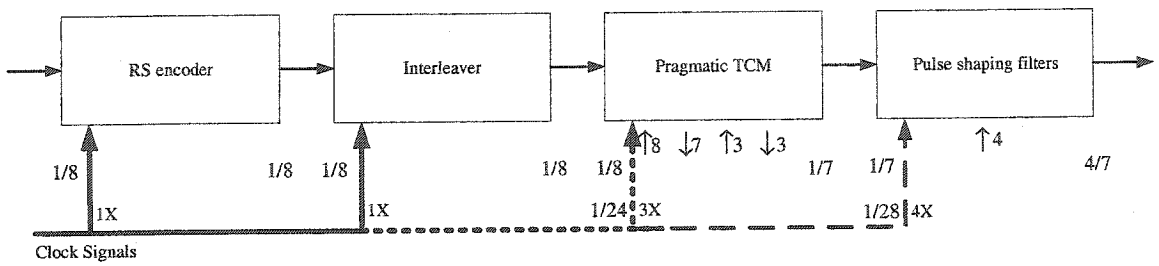


Figure 5-11: Clock distribution of the modem with Pragmatic TCM of rate *7/8*

Likewise, pulse-shaping filters are driven by *4X* clock, thus the input clock frequency is divided by *28* instead of *7*. In other words, *4X* clock of the pulse shaping filters is compensated by dividing the input frequency by *28*. This not only ensures consistent bit flow but also provides higher frequency that up-samplers require in the pulse shaping filters.

In the receiver, the decoder has the same clock structure as the encoder; therefore the clock distribution should be performed similarly. In other words, matched filter is driven by *4X Hz*, Pragmatic TCM decoder by *3X Hz* and de-interleaver and RS decoder are driven by *1X* Herz.

5.4 Hardware design flow using System Generator

It is mentioned earlier that the proposed modem is implemented based on bottom-up design methodology. Each leaf component is either imported from System Generator library or the VHDL code is written manually. If the VHDL code of the component is written manually, the functionality and timing of the component is verified before it is imported in the block. The functionality and timing of the component is verified by gate level simulation. Note that gate level simulation is performed after synthesis. After each component is verified individually, it is imported in System Generator platform as a black box. In the higher level of hierarchy, each block of the modem is implemented either based on the imported black boxes or other components available in the System Generator library. When the block is implemented, it is simulated in Simulink platform. Note that HDL model is co-simulated with ModelSim. ModelSim is one of the popular HDL simulators. Thus the behavior of the implemented block can be compared with its Simulink counterpart. Note that simulation results in System Generator do not give us any timing information. Eventually, the VHDL model of the block is generated by means of System Generator. The tool translates the model to VHDL by using Xilinx LogiCORE or the synthesizable VHDL codes which are imported as black boxes. The Generated VHDL model is synthesized and gate level simulation is performed. The timing report of the synthesis tool after place&route gives us final timing information of the circuitry. This methodology is used to construct each building block in our design.

5.5 Data synchronization

At this point it is necessary to introduce data synchronization in the design. The aim of data synchronization is to develop a simple approach such that the performance and functionality of each block remains consistent regardless of the presence of other blocks. In other words, the performance of the component should be consistent regardless of its position in the design.

In order to achieve the best timing result, the design is pipelined to the greatest extent possible. Parallel processing is also exploited to get better results. Therefore, each component produces latency in the data flow. Data manipulation in the modem is dependent to the latency of the real time data. For example, consider a serial to parallel converter in the Pragmatic TCM encoder. The output of the converter is different depending on the latency that the preceding components produce in the data flow. In other words, the converter should realize the validity of its input bits and start conversion while the data is valid. This simple example illustrates the significance of data synchronization in the design. The synchronization is performed by two control signals i.e. valid input (Vin) and valid output (Vout). Vin is an active high signal i.e. only data at the input ports of each component is considered valid when the Vin is asserted high. Each block also provides a valid output data port. Vout is high when the data at output port of the component is valid. A gate level simulation result of parallel to parallel converter of the TCM encoder is illustrated in Figure 5-12. Note that the component starts capturing input bytes when Vin is asserted high and when output data is ready at Es and NEs ports, Vout goes high. Note that Vin and Vout signals are synchronized with clock signal. It is

illustrated that Vout signal provides enough information about latency of the block, and with aid of Vin, data synchronization can be achieved easily.

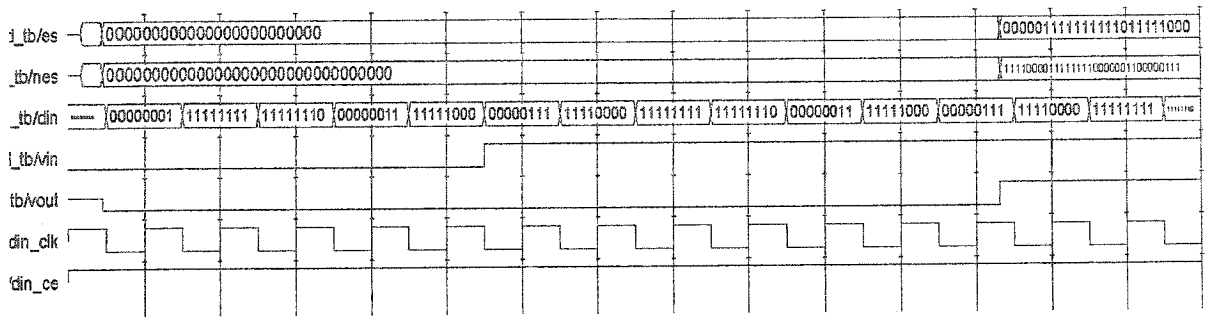


Figure 5-12: Simulation result of P/P converter

In our implementation, each component is equipped with Vin and Vout control ports. Thus, data synchronization is performed from the lowest level of hierarchy to the top level. In other words, when each block is constructed in top level design, simply each component should be cascaded and the data synchronization is performed automatically with the aid of Vout and Vin signals.

5.6 Pragmatic TCM encoder

Pragmatic TCM encoder is implemented based on the specifications stated in previous chapters. Depending on the coding rate, either punctured or non-punctured encoder is exploited in the design. The encoder can be divided to number of building blocks as it is illustrated in Figure 3-5. To elaborate on the general block diagram, The RTL schematic of Pragmatic TCM encoder of rate *7/8* is shown in Figure 5-13.

The first block is parallel to parallel converter. VHDL code of this block is written and the block is constructed manually. This block performs two tasks:

- Performs byte to bit conversion based on Table 3-1.Required Es and NEs bits are provided by parallel to parallel converter and are fed to the following blocks in the encoder.
- Performs block synchronization. Block synchronization is performed by means of Vin control signal.

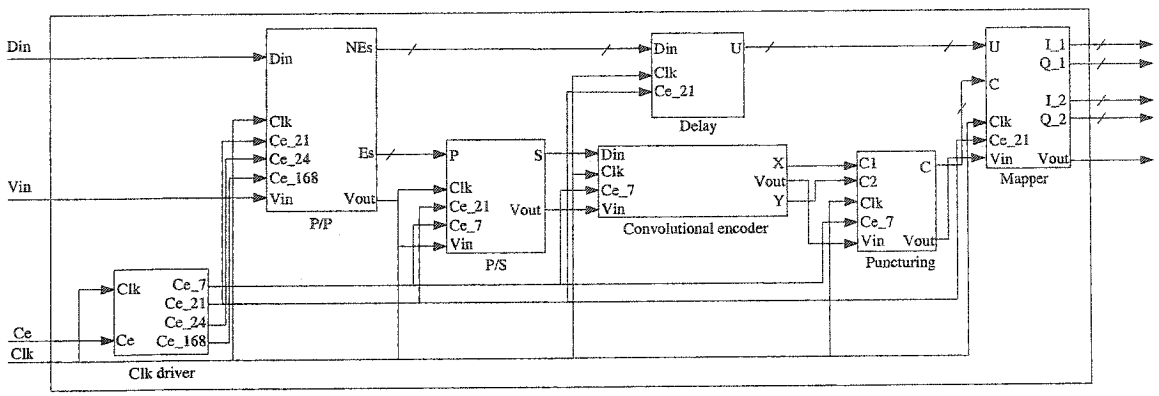


Figure 5-13: RTL schematic of the Pragmatic TCM encoder rate *7/8*

Convolutional encoder is imported from System Generator library. In order to puncture output of convolutional encoder, a serial to parallel (S/P) is used in the puncturing block. One bit out of each three bits word of data is erased based on Table 3-2. Eventually, "bit mapping to constellation" is performed based on look up table for inphase and quadrature individually. Depending on the Pragmatic encoder rate, two to four look up tables are employed. Note that the output of the mapper is *12*-bit signed fixed point with fraction point in bit *11* and the most significant bit is sign bit. The fixed point *(12,11)* representation of constellation point of the *16*-QAM is tabulated in Table 5-3. Note that in order to normalize symbol energy each constellation point is divided by $\sqrt{10}$.

Constellation point	Normalized Decimal representation	Fixed point (12,11) representation
3	0.94868329805051	"011110010110"
1	0.31622776601684	"001010000111"
-1	-0.31622776601684	"110101111001"
-3	-0.94868329805051	"100001101010"

Table 5-3: Binary fixed point representation of 16 QAM constellation point

Top level schematic of Pragmatic TCM encoder is depicted in Figure 5-14. Since Pragmatic TCM encoder/decoder blocks for different rate are driven with different clock, the latency of the blocks are expressed based on TCM input clock. Note that input clock of Pragmatic TCM encoder of rate *3/4* is SYS_CLK and that of rate *7/8* is three times of SYS_CLK (*3X*). Vout of the encoder block of rate 3/4 goes high 34 SYS_CLKs after Vin of the block is asserted high for the first time. On the other hand, the Pragmatic TCM encoder block of rate *7/8* produces a latency of *241* cycles of its input clocks.

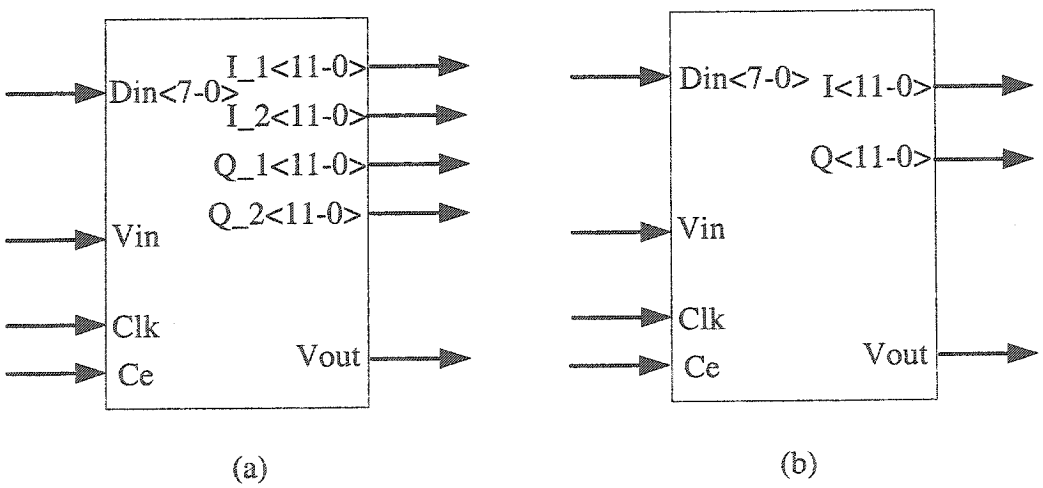


Figure 5-14: Top level schematic of Pragmatic TCM encoder (a) rate 7/8 (b) rate 3/4

5.7 Pragmatic TCM decoder

A general block diagram of Pragmatic TCM decoder is proposed in Figure 4-2. To elaborate on the figure, RTL schematic of the Pragmatic TCM encoder is shown in Figure 5-15. Despite the coding rate, HDL model of soft decision and outboard decision logic are prepared manually in the decoder. Soft decision logic is designed based on soft decision assignments illustrated in Figure 4-5. In the encoder, "bit mapping to constellation" is performed such that both inphase and quadrature components have the same mapping for coded-bit. Therefore, the same soft decision block is employed for both components of the received symbols. Twenty one soft decision points are chosen to determine a *3*-bit soft decision weight for each coded-bit. Note that inphase and quadrature components are *12*-bit signed fixed point with *9* bits fraction i.e. *Sfix(12,9)*, therefore; twenty one *12*-bit comparators are exploited to determine the most likely soft decision weight. It takes *3* CLK cycles for the soft decision to determine each soft decision weight. In other words, soft decision block adds *3* CLK cycles of latency to data flow in the design. Then, depending on the structure of the decoder these soft decision weights are fed to viterbi decoder either directly or through insert zero block (de-puncturing). Viterbi decoder decodes coded bits of each symbol. The Viterbi decoder generates latency in the data flow according to its configuration. One of the factors that affect the latency is trace back length of the decoder. Since punctured code bits are more vulnerable to error, longer trace back length is chosen for the same decoder. Hence, decoder with rate *7/8* generates more latency than that of rate *3/4*. Xilinx LogiCore is exploited to implement the viterbi decoder [34] and convolutional encoder [35] in the structure of Pragmatic TCM decoder.

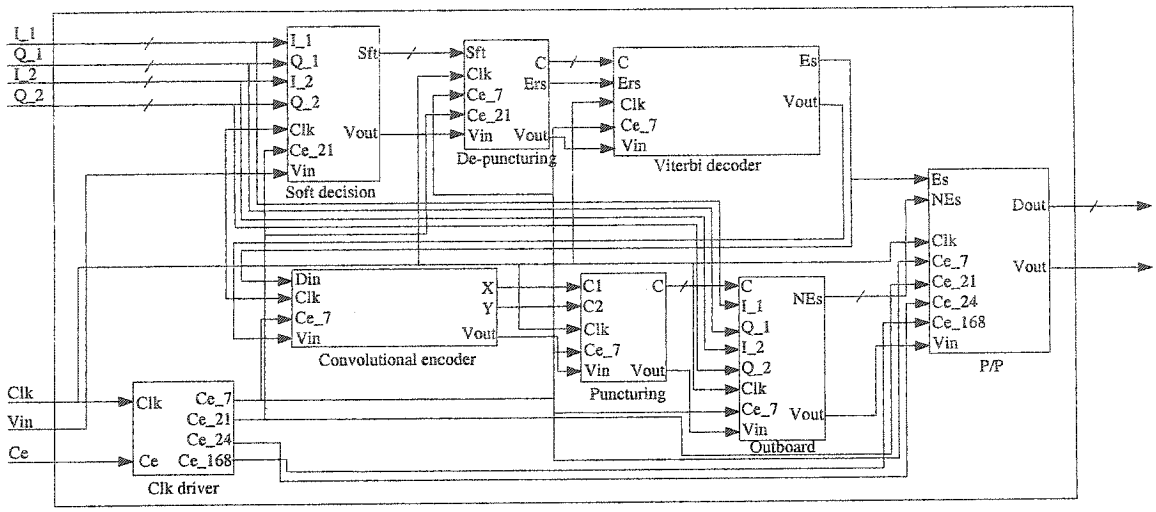


Figure 5-15: RTL schematic of the Pragmatic TCM decoder rate *7/8*

The VHDL code of the outboard decision logic is prepared base on threshold levels that are illustrated in Figure 4-5. Like soft decision, special mapping which is employed in the encoder lets us to use the same outboard decision block to detect uncoded bit in both inphase and quadrature. The outboard decision block is required two pieces of information to estimate the uncoded bit; either projection of the received signal vector on Cartesian axis and the companion estimated coded-bit. Outboard decision logic estimates the most likely uncoded-bit in the designated symbol. The hardware is implemented with two *12*-bit comparators, a multiplexer and number of flip flops. Outboard decision block needs *3* CLK cycle to determine first uncoded-bit. In other words, the latency of the block is *3* CLK cycle.

Finally, a parallel to parallel block is required to descramble Es and NEs input data and prepare output bytes based on Table 3-1. The HDL model of this bock is also written and imported as a black box in the Pragmatic TCM decoder. For rate *7/8*, since the Es and NEs bit have different rates i.e. due to down sampling in the data flow, the *NEs* is fed to the parallel to parallel block with lower rate. In other words, Es bits clocked

3 times faster than NEs bits. Note that *56* bits of the parallel to parallel converter consist of *32* bits NEs and *24* bits Es which are buffered. Afterwards, *7* bytes of descrambled data is appeared at output of the block at rising edge of output clock. It completely shows that the design is multi-rate.

Top level schematic of Pragmatic TCM encoder is depicted in Figure 5-16.The Pragmatic TCM decoder of rate *3/4* requires *664* SYS_CLKs to completely decode input symbols and produce the first output byte. In other words, the latency of the Pragmatic decoder is *664* cycles of SYS_CLK for coding rate of 3/4. On the other hand, Pragmatic TCM decoder of rate *7/8* produces a latency of *3097* cycles of its input clocks.

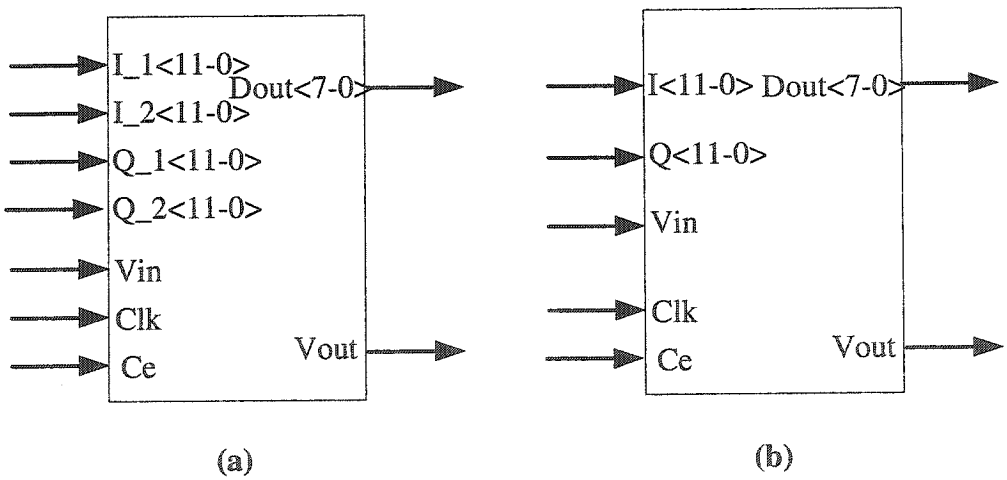


Figure 5-16: Top level schematic of Pragmatic TCM decoder (a) rate *7/8* (b) rate *3/4*

5.8 RS encoder and decoder

In this project HDL model of both RS encoder and decoder are imported from Xilinx logiCore. In other words, System Generator exploits Xilinx LogiCore [12] to generate the HDL model. The particular encoder chosen based on DVB is the shortened

RS (*204,188*). Top level schematic of the RS encoder and decoder are depicted in Figure 5-16. In the transmitter, RS encoder has a latency of *5* clocks. Hence, Vout of the RS encoder is high after *40* SYS_CLKs. Note that in our design, regardless of the inner coding rate, RS encoder core is driven by *1/8* of SYS_CLKs rate.

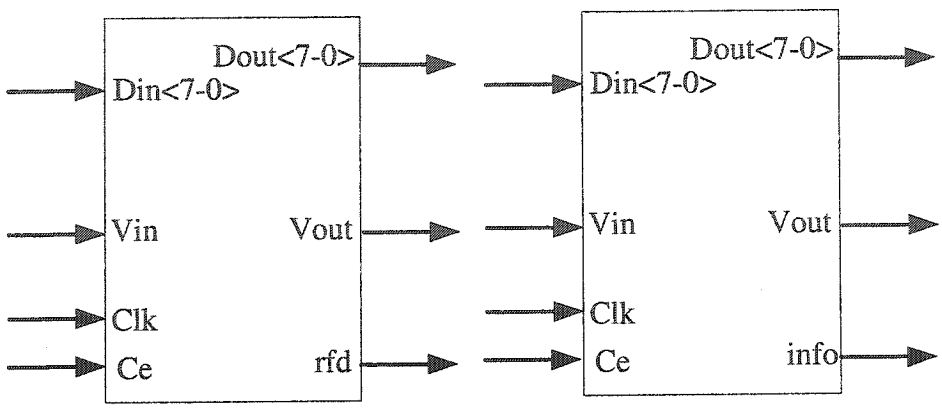


Figure 5-17: Top Level schematic (a) RS encoder (b) RS decoder

It is essential to understand the core functionality. RS encoder block, is fed by *188* consecutive bytes of data and produces output code words of length *204* bytes. Thus frame of *188* bytes is followed by *16* parity bytes at the output of the encoder. Hence, the core is not ready to receive input bytes for 16 consecutive clock cycles while producing parity bytes. An *rfd* pin shows the status of the core i.e. when rfd pin is equal to *'1'*, the encoder is accepting and producing information symbols, and when it is *'0'*, the encoder is producing the parity symbols. In our design, rdf goes low *1464* CLK_SYSs after Vout is high for the first time. rfd remains low for *128* CLK_SYSs before it goes high again. Note that rfd is a periodic signal i.e. it is high for 188 clock cycles, when the encoder is accepting and producing the symbols and it is low when the encoder is producing the

parity symbols. Timing of the control signals of RS encoder is depicted in Figure 5-17. Note that in the figure core clock is chosen as time scale instead of SYS_CLK.

On the other hand, RS decoder is fed by successive symbols of a coded frame. A frame of *204* symbols is fed to RS decoder. It takes the decoder block *1904* SYS_CLK to decode the first frame and produce the first symbol at its output. As shown in Figure 5-16, like RS encoder, the status of the decoder is also depicted by a status pin i.e. info. The info pin is high when the decoder is producing the decoded symbols and it is low when the decoder is producing random symbols. Note that the decoder is fed by a frame of 204 bytes which consists of *188* bytes of message symbol and *16* byes of parity. Therefore, the info pin is high for *1504* SYS_CLK and then it goes low for *128* SYS_CLK. Note that the info signal is a periodic signal. The control signals of the RS decoder are sketched in Figure 5-17.

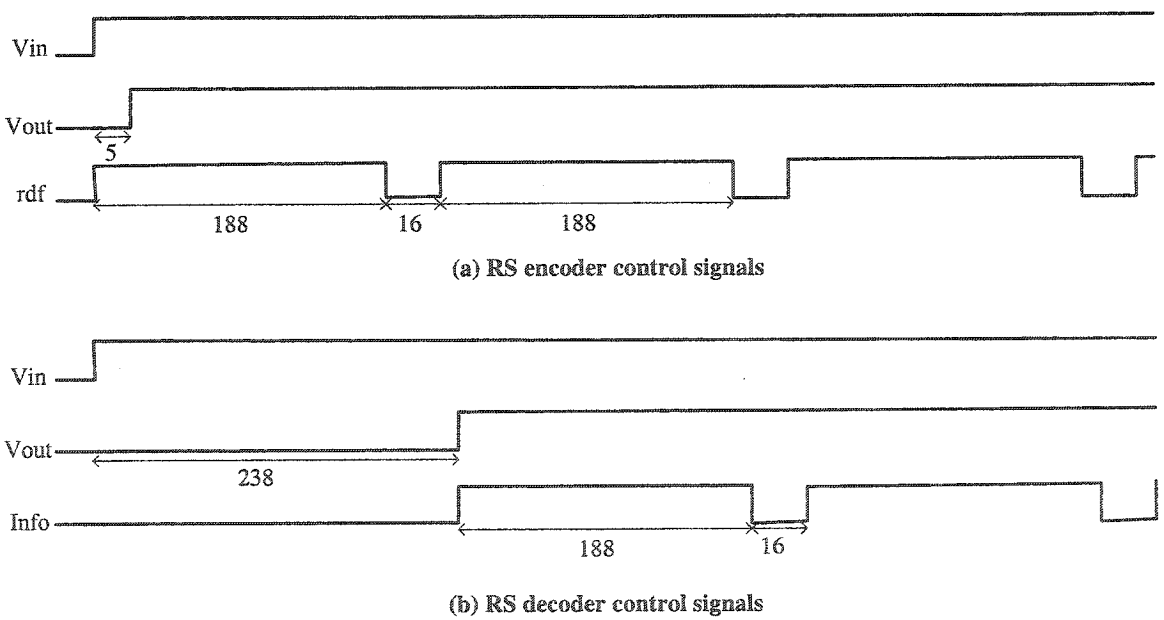


Figure 5-18: Control singnals of (a) RS encoder (b) RS decoder

5.9 Interleaver and De-interleaver

In this project HDL model of both Interleaver and De-interleaver are imported from Xilinx LogiCore. In other words, System Generator exploits Xilinx LogiCore to generate the HDL model of the designated blocks. The structure of Forney convolutional interleaver is shown in Figure 2-6. The core operates as a delay line shift registers [24]. Input symbols are presented to the input commutator from Din. Output interleaved symbols are extracted from the output commutator to Dout. Both commutator arms are synchronized i.e. both commutators start from the top branch. Note that in Figure 2-6, although first branch appears to be zero-delay connection, because of the core structure there is a delay between Din and Dout. Top level schematic of Interleaver and De-interleaver is depicted in Figure 5-18.

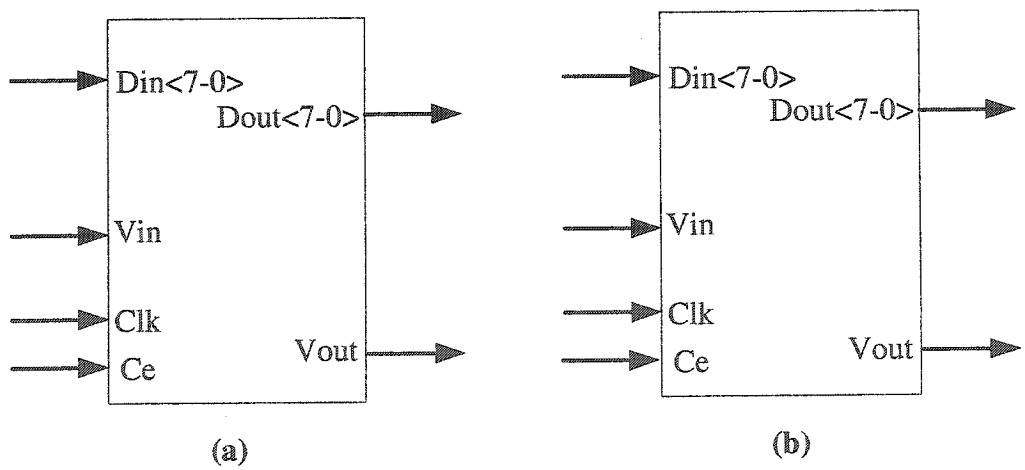


Figure 5-19: Top level schematic of (a) Interleaver (b) De-interleaver

Both Interleaver and De-interleaver have the same latency. In other words, both blocks produce *48* SYS_CLK delay i.e. Vout status signal of both blocks goes high after *48* SYS_CLK when Vin is asserted high for the first time.

5.10 Pulse-shaping and matched filters

The value of the coefficients and number of filter taps are evaluated in the previous chapters. The polyphase implementation of up-sampling filter (pulse-shaping filer) and down-sampling filter (matched filter) are also stated in chapter 4. The pulse-shaping and matched filers are implemented based on the polyphase structure illustrated in Figure 4-9 and Figure 4-10, respectively. As shown in Figure 5-20, inphase and quadrature component of each symbol are filtered individually and the same polyphase filter is used for both components. Depending on the coding rate, the pulse-shaping and matched filters consist of two or four parallel polyphase filters. In other words, according to the coding rate of the encoder one or two symbols are fed to the filter blocks.

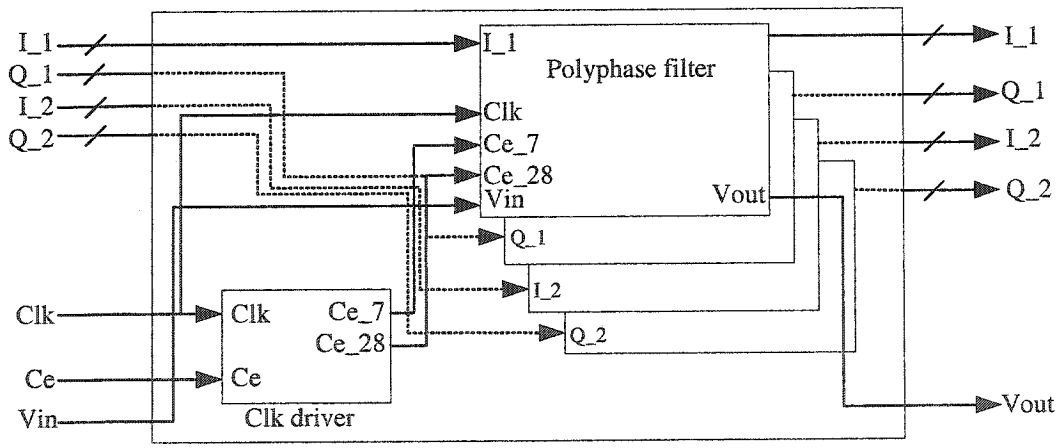
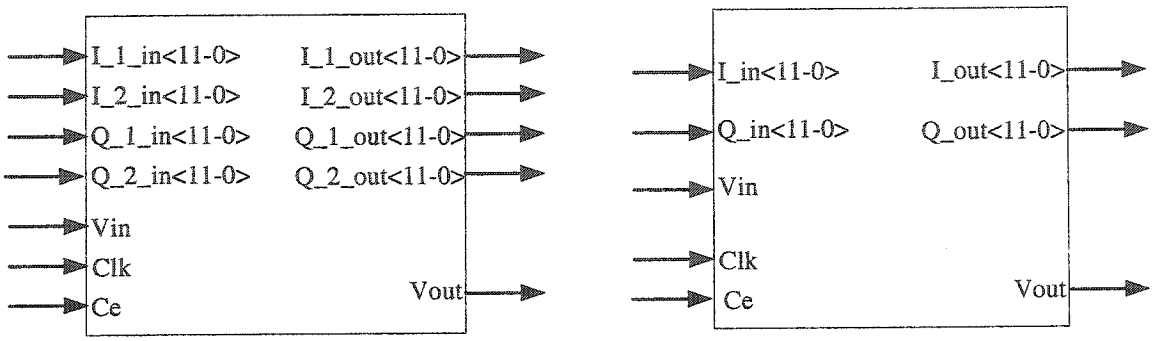


Figure 5-20: RTL schematic of pulse-shaping and matched filters for rate *7/8*

Although the filters for both coding rate is implemented based on the same sub-filters with the same specifications, each one requires different number of SYS_CLK to produce the first output. In other words, the latency of the blocks is different with respect to SYS_CLK. This is due to the fact that the sub-filter of each block is driven with different internal clocks which are extracted from CE signals. Pulse shaping filters of

either rate *3/4* or *7/8* produces latency of *54* and *118* filter clock cycles, respectively. Likewise, matched filters in the receiver produce latency of *203* and *475* filter clock cycles for rate *3/4* and *7/8*, respectively. In other words, it takes *203* and *475* filter clock cycles for Vout to go high when Vin is asserted high for the first time. Top level schematic of the pulse shaping and matched filters are depicted in Figure 5-21.



(a) **(b)**

Figure 5-21: Top level schematic of pulse shaping and matched filters

(a) rate 7/8 (b) rate 3/4

5.11 Modem Verification

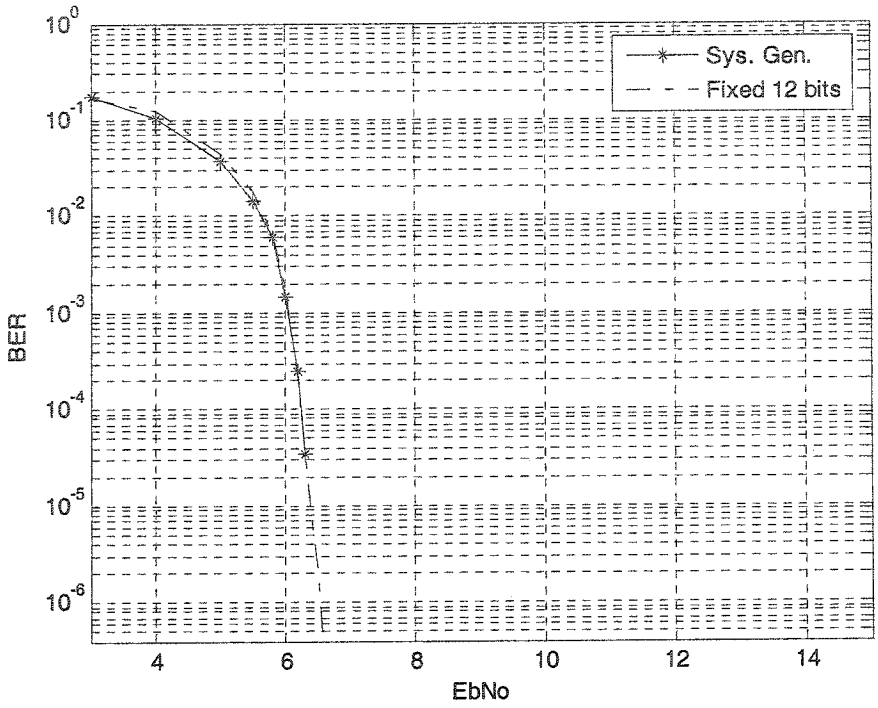
The next step in the implementation flow is to simulate the top level design and verify the circuitry. The behavioral simulation is carried out in System Generator platform. Top level design of the modem is imported individually as a transmitter and receiver black boxes in the tool. The model is simulated with presence of AWGN channel and a bit error rate versus noise power curve is obtained. Since the simulation is performed with ModelSim and Simulink, the simulation takes longer time than a pure Simulink model. Hence, the simulation is performed up to BER of 10^{-4} .When the behavioral simulation of the circuitry is obtained, it is compared with the fixed-point

simulation results of the model. Afterwards, post-place&route simulation is performed to verify the final circuitry. A test bench is written manually and the simulation is performed with ModelSim.

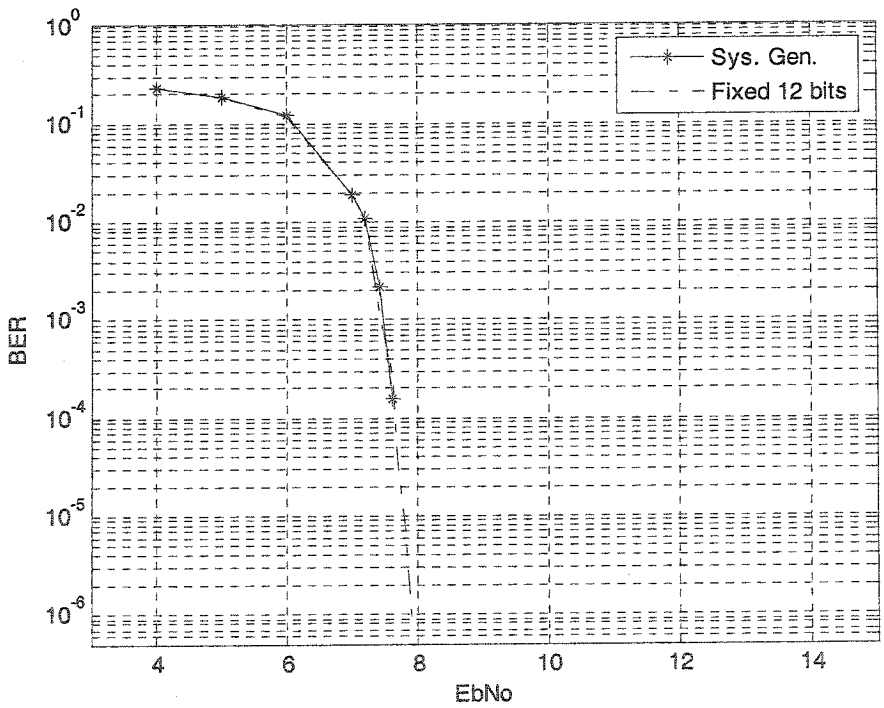
In the following subsection behavioral simulation result of the modem for both rates are shown. Post-place&route simulation process is also briefly expressed. Finally, the timing and area report of the circuitry is stated.

5.11.1 Simulation of the modem in System Generator platform

Top level design is imported in a System Generator platform as two black boxes i.e. Transmitter and receiver. A mixed model is designed to verify the functionality of the modem. In other words, AWGN channel, Random binary number generator and Error rate calculator are imported from Simulink library. After the model is implemented, it is simulated with different power of noise to obtain probability of uncorrected bits versus noise power. The simulation result of the modem with both rates is depicted in Figure 5-22. In the figure, *12*-bit Fixed-point simulation is also sketched for comparison. As shown, the HDL model and fixed point have the same performance.



(a)



(b)

Figure 5-22: Simulation result of the modem (a) rate *3/4* (b) rate *7/8*

5.11.2 Simulation of the modem in the Xilinx ISE with ModelSim

Although the simulation results which are obtained in System Generator platform verify the design functionality, it does not guarantee post synthesis characteristic of the modem. Post-place&route simulation is performed to verify the functionality of the design before it is downloaded in the target FPGA chip. The target FPGA chip is XC2V4000. In order to simulate the post-place&route design, a test bench is prepared for the designated top level model. In the test bench the "uniform" procedure of *math_real* package in *IEEE* library is used to generate random bytes. This procedure generates successive values between *0.0* and *1.0* [25]. These randomly generated real numbers are converted to natural numbers between 0.0 and $2^8 - 1$. Finally, these natural numbers are converted to *std_logic_vector*. These random test vectors are fed to the transmitter input. In the receiver, the output of the receiver is compared to the generated bytes. If "*Error*" is detected, severity message will be reported.

In order to verify the modem, *100,000* successive random bytes are generated to verify the modem functionality. The simulation results confirm the modem functionality. When the modem is completely verified, the top level design is ready to download to the targeted FPGA chip with Xilinx ISE.

5.11.3 Timing and area report of the circuitry

The area report of the Xilinx ISE shows number of used components in the target FPGA chip. The Xilinx XC2V4000 is chosen as the target FPGA chip. The selected chip approximately comprises of *4* million equivalent gates. The area report of the Xilinx ISE is tabulated in Table 5-4. In the table, area report of the modem with both rates is

illustrated. Since the RS encoder/decoder and interleaver/de-interleaver blocks for both rates have the same structure, the same report is given by the tool. As shown in Table 5-4, depending on the coding rate of the modem, it uses 914,882 and 1,251,296 equivalent gates of the target FPGA chip for rate *3/4* and *7/8* respectively. Therefore, the implemented modem approximately uses 30% of the available resources of the target FPGA chip.

Synthesis tool also provides maximum delay of the critical path. Note that, maximum achievable baud rate of the modem is restricted to the maximum applicable clock frequency of the modem. On the other hand, according to the design methodology, low de-skewed clock signal is distributed by DCM. Depending on the operating mode of the DCM and the speed grade of the FPGA target chip, input and output frequency range of the DCM is bounded [22]. Hence, the minimum achievable baud rate of the modem is bounded by the DCM minimum applicable clock frequency.

According to Table 5-1 and Table 5-2 bottle neck of the design is pulse-shaping and matched filers. Note that both filters are driven by *4X* of the SYS_CLK frequency. Hence, SYS_CLK is selected to be *27.778* MHz (*36*ns). The necessary timing constraints are set along with the selected system clock frequency. Then, the modem is synthesized and configured to the target FPGA chip. Post-place&route timing report of the synthesis tools ensure that all timing constraints are met. Therefore, the maximum achievable bit rate is 27.778 Mbit/s. It shows *12* times speed improvement with the proposed clock distribution.

Block	Total Gates
RS encoder	3,381
RS decoder	150,115
Interleaver	73,281
De-interleaver	73,281
PTCM encoder 3/4	133,323
PTCM decoder 3/4	334,815
PTCM encoder 7/8	267,549
PTCM decoder 7/8	351,980
Pulse shaping 3/4	78,535
Matched filters 3/4	63,867
Pulse shaping 7/8	156,839
Matched filters 7/8	127,359
Total Modem 3/4	914,882
Total Modem 7/8	1,251,296

Table 5-4: Final area report of the synthesis tool

Bit rate of 8.448 Mbit/s is recommended by DVB standard for data transmission with audio signal (MPEG 1 Layer 2, 48KHz stereo) along with video signal (25 frame of 720x576 pixels or 30 frame of 720x480 pixels). The post-synthesis timing report of the modem demonstrates that the modem meets the desired bit rate which is required by the standard.

Chapter six

6 Conclusion

In recent years, the underlying FPGA technology has advanced rapidly. As a result, new opportunities such as System On a Programmable Chip (SOPC) are emerging. SOPC design flow provides rapid prototyping of complex algorithms in a programmable chip such as FPGA. This methodology also enables system-level to FPGA design flow and hardware/software co-design and co-verification. Additionally, FPGA design technology reduces time-to-market by bringing the concept to implementation in a very short time.

In this thesis, we model a base-band modem, specified by DVB standard, at the system level. After the fixed-point model is verified, it is translated to lower level of abstraction in Hardware Description Language (HDL). The modem consists of Reed Solomon code, Convolutional Interleaving, Pragmatic Trellis Coded Modulation (PTCM), and Pulse- shaping filters. Once the HDL model is verified, it is implemented in the target FPGA chip. The design flow can be utilized to design, verify and implement any digital communication and DSP algorithm, resulting in accelerated hardware verification and implementation.

The design flow starts with system level modeling of a modem parameterized with DVB standard. A floating-point model of the system is modeled, and simulated in Simulink platform. Comparing the simulation results with the desired performance specified by the standard, the functionality of the modem is verified. Subsequently, the model is converted to fixed-point. The required number of bits to achieve adequate

precision of each building block is evaluated by performing fixed-point simulation. The fixed-point representation of the model compromises performance for hardware simplicity.

In order to translate the system-level design to HDL netlist, the fixed-point model is re-designed with Xilinx System Generator blocksets. The HDL model is either instantiated from IP blocks of Xilinx LogiCore libraries or generated from a customized synthesizable library. Customized VHDL code of the synthesizable library is verified by post-synthesis simulation before it is imported as a black box into System Generator. ModelSim is exploited to co-simulate each System Generator building block in Simulink platform, verifying block functionality before its HDL netlist is generated. Finally, the complete modem is co-simulated and BER curves are compared with that of the fixed-point model.

To increase the throughput of the modem, a novel clock distribution is proposed. Digital Clock Manager (DCM), an embedded component of Xilinx Virtex II FPGA, is employed to distribute the de-skewed clock signal in the top level design. Random test vectors are generated by a customized test bench and post-place&route simulation is performed to verify the synthesized design. Finally, the design is configured to Xilinx FPGA Xc2V4000. The Place&route timing report confirms that the design meets the timing constraints. The implemented modem is able to transmit/receive digital video and audio signals up to 27.778 Mbit/s which is adequate for High Definition TV (HDTV) based on DVB standard. The implemented modem approximately uses 30% of the available equivalent gates of the designated target FPGA chip.

Addressing future work, unused available resources in the FPGA chip can be employed to complete the base-band system specified by DVB standard. Additional blocks perform MPEG-2 source coding, audio and video multiplexing and randomization for energy dispersal. Clock recovery and frame synchronization can also be added to complete the base-band modem. In addition, implementing the system in newer FPGA technology such as Virtex-4 with updated IP cores significantly increases the maximum input clock. As a result, the throughput of the modem will be enhanced.

Reference:

[1] B. Sklar, Digital Communications Fundamentals and Applications, Second Edition

[2] J.G. Proakis, Digital Communications, Fourth Edition, New York: McGraw-Hill,2001

[3] Shuh Lin, Daniel J. Costello, "Error Control Coding Fundamentals and Applications", Prentice-Hall, Inc. Engglewood Cliffs, New Jersey 1983

[4] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets-Part I:Introduction," *IEEE Communicalzons, Magazine,* vol. 25, no. 2, February 1987

[5] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets-Part II: State of the Art", *IEEE Communicalzons, Magazine,* vol. 25, no. 2, February 1987

[6] G. Ungerboeck, "Channel coding with multilevel/phase signals", IEEE Transactions on information theory,Vol. T.28, No.1 January 1982

[7] Viterbi, A. J., "Error Bonds of Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", IEEE Transaction information Theory, Vol. IT13, April 1967,PP 260-269.

[8] A. J. Viterbi, J. K. Wolf, E. Zehavi, and R. Padovani, "A pragmatic approach to trellis-coded modulation", IEEE Communication Magazine, PP.11-19, July 1989.

[9] Y. Yasuda, K. Kashusi, and Y. Hirata, "High-rate punctured convolutional codes for soft decision viterbi decoding" IEEE transaction on communications Vol. COM-32, PP 315-319, March 1984.

[10] J. K. Wolf, E. Zehavi," P^2 Codes: Pragmatic trellis codes utilizing punctured convolutional codes" IEEE Communication Magazine, PP.94-99, February 1995

[11] G. D. Forney, "Burst-Correcting codes for the classic bursty channel",IEEE transaction on communication technology, Vol. Com-19,October 1971,PP 772-781.

[12] Xilinx LogiCore RS decoder V.4.0 data sheet, http://www.xilinx.com/ipcenter/catalog/logicore/docs/rs_decoder.pdf

[13] Stephen G. Wilson, Digital Modulation and Coding, Prentice Hall, Inc., 1996.

[14] "Digital Video Broadcasting (DVB): Framing structure, Channel Coding and Modulation for Digital Satellite News Gathering (DSNG) and Other Contribution Applications by Satellite", ETSI EN 301 210, European Standard

[15] K. Gentile, The care and feeding of digital, pulse-shaping filters, April 1st 2002, http://rfdesign.com/mag/radio_care_feeding_digital/index.html

[16] L. Wanhammar, DSP integrated circuits, academic press, 1999.

[17] S.K. Mitra, Digital Signal Processing, a computer based approach, Second Edition, 2001

[18] A.V. Oppenheim, R.W. Schafer, Discrete-time signal processing, Prentice-Hall, 1999

[19] Michael D. Ross, William P. Osborne, Frank Carden and Jerry L. Stolarczyk "Pragmatic trellis coded modulation: Hardware implementation using 24-sector 8-psk", IEEE, 1992

[20] Michael D. Ross, Frank Carden and William P. Osborne "Pragmatic trellis coded modulation: A simulation using 24-sector quantized 8-psk", IEEE, 1992

[21] Eun-A Choi,Dae-lg Chang, ae-Soo kim, "Design and implementation of the 155Mbps Modem for Broadband Satellite Communication", IEEE, 2002

[22] Xilinx Vertix-II Platform FPGA Handbook, Xilinx, 2000

[23] Xilinx System Generator for DSP complete user Guide v6.1, http://www.xilinx.com/products/software/sysgen/app_docs/user_guide.htm

[24] Interleaver/De-interleaver product specification v3.0, Xilinx LogiCore, DS250 (v1.0) March 28, 2003

[25] Peter J. Ashenden,The designer's guide to VHDL, 2nd edition, Morgan Kaufmann publishers, 2002

[26] Changchun Shi, A tutorial on out Floating-point to Fixed-Point Convesion (FFC) Tool, Berkely Wireless Research Center, EECS Dep., University of California, Berkely, March 25, 2004

[27] "choosing FPGA or DSP for your application", http://hunteng.co.uk/info/fpga-or-dsp.htm

[28] FPGA and DSP, www.adaptyv.com/doc/xilink/Xilink-FPGAsandDSP.pdf

[29] "Digital Video Broadcasting (DVB): Framing structure, channel coding and modulation for 11/12 GHz satellite services", ETSI EN 300 421, European Standard

[30] S. Hamidreza Jamali, Tho Le-Ngoc, Coded-modulation technics for fading channels

[31] IEEE Computer Society (1985), *IEEE Standard for Binary Floating-Point Arithmetic*, IEEE Std 754-1985.

[32] Xiao Hu, Anh Dinh, "A novel high-speed trellis-coded modulation codec", proceeding of Canadian conference IEEE, 2002

[33] Xilinx Virtex-II data sheet, http://www.xilinx.com/bvdocs/publications/ds031.pdf

[34] Xilinx LogiCore viterbi decoder V3.0 data sheet, http://www.xilinx.com/bvdocs/ipcenter/data_sheet/viterbi.pdf

[35] Xilinx LogiCore convolutional encoder V3.0 data sheet

http://www.xilinx.com/ipcenter/catalog/logicore/docs/convolution.pdf