IMPLEMENTATION OF STANAG 4285 HF MODEM SOFTWARE ON TMS320C54X DIGITAL SIGNAL PROCESSOR

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY
ERHAN ÖRÜMLÜ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2004

Approval of the Graduate School of Natural and Applied Sciences

_____

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. Mübeccel Demirekler
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Buyurman Baykal
Supervisor

Examining Committee Members

Assoc. Prof. Dr. T. Engin Tuncer (METU,EEE) _____

Prof. Dr. Buyurman Baykal (METU,EEE) _____

Asst. Prof. Dr. Tolga Çiloğlu (METU,EEE) _____

Asst. Prof. Dr. A. Özgür Yılmaz (METU,EEE) _____

Mert Sungur (TÜBİTAK) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Erhan Örümlü

Signature :

ABSTRACT

IMPLEMENTATION OF STANAG 4285 HF MODEM SOFTWARE
ON TMS320C54X DIGITAL SIGNAL PROCESSOR

Örümlü, Erhan

M.Sc., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Buyurman Baykal

August 2004, 84 pages

In this research, STANAG 4285 HF modem software is implemented on TMS320C54x fixed point digital signal processor. The software is optimized in order to meet real-time operation requirements. A fractionally spaced least mean square (LMS) decision feedback equalizer (DFE) is employed for the receiver. In order to improve the convergence of the LMS algorithm a multipass technique is utilized. Based on Watterson's model, an HF channel simulator is employed for evaluating the performance of the modem. The simulation results show that the convergence of the LMS algorithm is improved by using multipass technique. It is also shown that the software meets the real-time operation requirements.

Keywords: STANAG 4285, HF Modem, TMS320C54x, DFE, Watterson's Model

ÖZ

STANAG 4285 HF MODEM YAZILIMININ TMS320C54X
SAYISAL İŞARET İŞLEMCİSİ ÜZERİNDE GERÇEKLENMESİ

Örümlü, Erhan

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Buyurman Baykal

Ağustos 2004, 84 sayfa

Bu araştırmada, STANAG 4285 HF modem yazılımı TMS320C54x sabit noktalı sayısal işaret işlemcisi üzerinde gerçeklendi. Yazılım, gerçek zamanlı işlem gereksinimlerini karşılaması için optimum hale getirildi. Alıcı için kesir aralıklı en küçük ortalama kare (LMS) karar geri beslemeli dengeleyici (DFE) kullanıldı. LMS algoritmasının yakınsamasını iyileştirmek için çoklu geçiş tekniğinden yararlanıldı. Modem yazılımını test etmek için Watterson modeline dayanan HF kanal benzeticisi kullanıldı. Benzetim sonuçları LMS algoritmasının yakınsamasının çoklu geçiş tekniğinin kullanılarak iyileştirildiğini gösterdi. Yazılımın gerçek zamanlı işlem gereksinimlerini karşıladığı gösterildi.

Anahtar Kelimeler: STANAG 4285, HF Modem, TMS320C54x, karar geri beslemeli dengeleyici, Watterson Modeli

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my advisor, Prof. Dr. Buyurman Baykal for his valuable guidance and encouragement throughout this research.

I would like to give special thanks to Fatih Bayramoğlu, İzzet Özçelik and Özgür Ertuğ for their valuable help during the wireless modem project.

I am also grateful to Tansu Filik and Yılmaz Kalkan for their support at the last phases of the research.

I would like to express my deep gratitude to my family and all who have encouraged and helped me at the different stages of this work.

Finally, I would like to thank TÜBİTAK UEKAE for the support offered in this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The high frequency (HF) spectrum, extending from 2 to 30 MHz, has been used for many years for beyond-light-of sight communications. In the early part of the 20^{th} century HF radio was the principal form of the long distance communications without cables. Nevertheless, in the early 1970s the commencement of the satellite communications ceased the research and development of HF systems [5]. Many users believed that HF communications would become obsolescent, since satellite communications offered higher data rates and reliable service. However, for many applications HF remained the primary means of beyond-line-of-sight communications, mainly due to its lower cost and low power requirements.

The beyond-line-of-sight capability of HF communications is achieved by the reflection of the radio waves from ionized particles in the ionosphere. Unfortunately, the ionosphere is a very challenging physical channel and causes impairments such as multipath propagation and fading. Moreover, solar induced ionospheric disturbances may even lead to the interruption of the communication.

The advances in digital technology have made possible the implementation of the designs of greater complexity in order to tackle the inherent problems encountered in HF transmission and hence the demand for HF modems using 3 KHz voiceband channel has increased. For military short wave communication systems two waveforms have been standardized, STANAG 4285 and MIL-STD-

188-110A [1], [2]. Today, modems for these waveforms are available in the market.

The subject of this thesis is the implementation of STANAG 4285 modem software on TMS320C54x ('C54x) fixed-point digital signal processor (DSP). Although the transmitter part is standard, a variety of algorithms can be employed for the receiver. However, the computational requirements of the algorithms should not exceed the capacity of the processor. Therefore, optimization of codes turns out to be an important issue besides the performance of the receiver.

In order to evaluate the performance of the HF modems without on-the-air experiments a mathematically tractable model representing the HF channel should be used. Most of the HF channel simulators employ Watterson's channel model [3] that is also recommended in STANAG 4285. Chapter 2 deals with the characteristics of the HF medium and traditional HF modems. Moreover, the STANAG 4285 waveform is described in detail.

Chapter 3 is concerned with the implementation details. The Tiger board including TMS320C5410 ('C5410) DSP is selected as the implementation platform. After a brief explanation of features of 'C5410 processor, the fixed-point representation and its numerical effects are discussed. Since adaptive algorithms are affected significantly by finite precision, the least mean square (LMS) algorithm and numerical effects are described also. In the implementation, LMS algorithm is used in order to adapt the filter coefficients of the decision feedback equalizer (DFE). At the end of the chapter also a review of the decision feedback equalizer is given.

The transmitter and receiver programs implemented on 'C5410 are discussed in Chapter 4 and Chapter 5, respectively. Both algorithms and their resource consumptions are described in detail.

Chapter 6 describes the simulation environment and illustrates the performance of the HF modem under various channel conditions.

Finally, Chapter 7 discusses the performance of the HF modem implemented on 'C5410 and the results of the code optimization techniques used for minimizing the load on the processor.

CHAPTER 2

HF CHANNEL AND HF MODEMS

2.1 Introduction

High frequency (HF) portion of the spectrum, allocated to the 2-30 MHz band, has been of great interest for many years for long-distance communications. At these frequencies the ionosphere can be used to reflect and refract radio waves enabling beyond line of sight communications.

The propagation conditions through the ionosphere may change significantly. The geographical location, time of the day, season and the solar activity can affect the conditions. Moreover, the signals passed through the HF channel face some distortions, such as multipath propagation and fading. In order to develop robust receiver structures for HF communications the behavior of the ionosphere and channel characteristics should be comprehended.

The first section of this chapter deals with the HF channel characteristics. Moreover, the physical structure of the ionosphere and Watterson's HF channel model are described. In the second section the types of HF modems are discussed and STANAG 4285 waveform is described in detail.

2.2 HF Channel

2.2.1 Ionosphere

The ionosphere is an upper atmospheric layer above the Earth's surface, extending from 50 km to 400 km altitude where electromagnetic radiation from the sun leads to ionization of the neutral gases. For reasons related to the historical development of ionospheric research, the ionosphere is divided into three regions designated D, E and F, respectively, in order of increasing altitude. While the E- and F- regions act mainly as radio wave reflectors, enabling long-range propagation, the D-region acts principally as an absorber and causes signal attenuation in the HF range [5]. The electron density of the ionospheric regions is given in Figure 2.1.

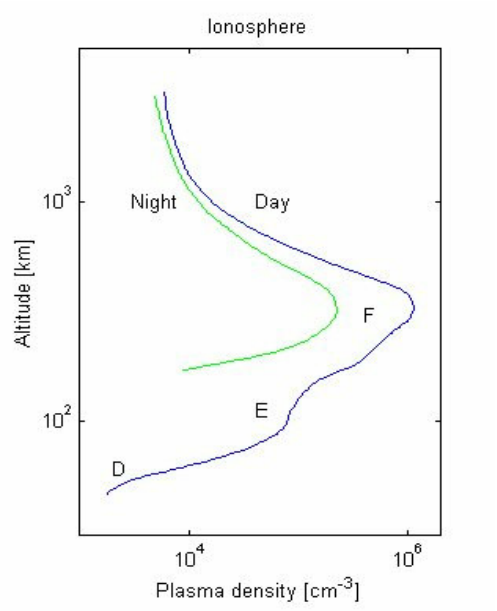

Figure 2.1: Electron densities of the ionospheric regions

The portion below 90 km of the ionosphere is known as the D-region [6]. The D-region electron density exhibits large diurnal variations. It has a maximum value shortly after midday and a very small value at night. The D-region, because of its relatively higher concentration of neutral particles and heavy ions, absorbs energy from a passing wave.

The altitude range from 90 km to 160 km constitutes the E-region. The E-region can support long distance communication up to 2000 km using frequencies as high as 20 MHz.

The F-region extends upwards from 160 km and is divided into the F1 and F2 layers [5]. The F1 layer exists only during daylight and it merges with the F2 layer at nighttime. Usually, waves that penetrate the E-region also penetrate the F1 layer and are reflected by the F2 layer. While F1 layer introduces additional absorption commonly, the F2 layer is the principal reflecting region for long distance HF communication up to 4000 km or more.

There are several modes of propagation for HF transmission. For example, a single reflection from the F-region is known as 1F mode or a double reflection from the E-region is known as 2E mode. Figure 2.2 shows some of the propagation modes. In practice, the electron densities in the ionosphere are continuous and the path of the ray will likewise be a continuous curve.

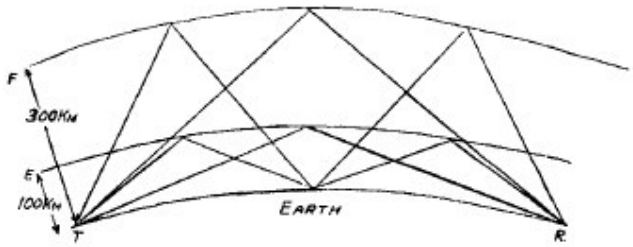

Figure 2.2: Examples of propagation paths of different modes

In general, each mode contains four components, known as ordinary and extraordinary magnetoionic components of both the low and high rays. While low rays are reflected from the boundary of the layers, high rays penetrate the layer, travel through it and then are reflected back to the Earth. Moreover, these rays are split in two components because of the Earth's magnetic field, ordinary and extraordinary waves. The existence of these components should be taken into account when modeling the HF channel.

2.2.2 HF Channel Characterization

There are many factors that affect the physical characteristics of the ionosphere. Since the ionosphere is not a static medium, the HF channel has a time-variant impulse response. If the same signal is transmitted at two widely separated time instants, the two received signals will be different and these time-varying responses are treated in statistical terms [4].

HF communication is mainly affected by multipath propagation. The transmitted signal travels over several modes or paths to the receiver, through single or multiple reflections from the E- and F- layers of the ionosphere. Since the time taken by each path is different, the signal at the receiver consists of several multipath components that are spread in time over an interval of up to several milliseconds. This phenomenon is known as multipath propagation and it is one of the major sources of the distortion in HF communications.

Another distortion type for HF communication is signal fading due to the time-variant multipath characteristics of the channel. In general, the transmitted signal is represented as

$$s(t) = \mathrm{Re}\left[s_l(t)e^{j2\pi f_c t}\right] \tag{2.1}$$

where $s_l(t)$ is the equivalent lowpass signal of s(t). Assuming that there are multiple discrete propagation paths, each path has a delay and an attenuation factor [4]. Then, the received bandpass signal can be expressed as

$$x(t) = \sum_n \alpha_n(t)s(t-\tau_n(t)) \tag{2.2}$$

where $\alpha_n(t)$ and $\tau_n(t)$ denote the attenuation factor and the propagation delay for the *n*th path, respectively. Substituting (2.1) into (2.2) and taking the equivalent lowpass of (2.2) yields

$$r_l(t) = \sum_n \alpha_n(t)e^{-j2\pi f_c \tau_n(t)}s_l(t-\tau_n(t)) \tag{2.3}$$

Using (2.3), the equivalent lowpass channel can be expressed as

$$c(\tau;t) = \sum_n \alpha_n(t)e^{-j2\pi f_c \tau_n(t)}\delta(t-\tau_n(t)) \tag{2.4}$$

Considering the transmission of an unmodulated carrier at frequency f_c, i.e. $s_l(t)$ is one for all t, the received signal becomes

$$r_l(t) = \sum_n \alpha_n(t) e^{-j2\pi f_c \tau_n(t)} = \sum_n \alpha_n(t) e^{-j\theta_n(t)} \quad (2.5)$$

Equation (2.5) shows that the received signal is composed of the sum of time-variant phasors having amplitudes $\alpha_n(t)$ and phases $\theta_n(t)$. The attenuation factors, $\alpha_n(t)$, change very slowly and do not cause a significant change in the received signal [4]. The fading distortion is mainly due to the random time variations in the phases $\theta_n(t)$. Even a small change in the delay $\tau_n(t)$ as $1/f_c$ can change $\theta_n(t)$ by 2π radian. The randomly time variant phases may lead to addition of phasors $\alpha_n(t)e^{-j\theta n(t)}$ destructively or constructively. These amplitude variations in the received signal lead to signal fading.

The changes in the delays $\tau_n(t)$ on different paths are expected to be at different rates and in a random manner. Therefore, assuming a large number of paths and using central limit theorem, the channel impulse response $c(\tau;t)$ can be modeled as a zero-mean complex-valued Gaussian process. In this case, the envelope $|c(\tau;t)|$ for any t becomes Rayleigh distributed and the channel is called Rayleigh fading channel. Most of the HF modem designs assume this type of channel model. Another probability distribution function used to model the envelope of the channel is Rice distribution that is employed when there are fixed signal reflectors in the medium, thus when a model with zero mean becomes impossible.

In addition to multipath propagation and fading, another multiplicative distortion is the frequency shift. In general, the average heights of the ionospheric layers are increasing or decreasing with time, introducing different Doppler shifts on each of the multipath components.

2.2.3 Watterson HF Channel Model

In order to determine the performance of the HF modems in the laboratory environment, many simulators have been developed. The most popular HF channel simulator model is the “Watterson Model” which was developed in 1970 [3].

As mentioned in [3], if the consideration is restricted to band-limited channels (say, 10kHz) and sufficiently short periods of time such as 10 minutes, most of the HF ionospheric channels can be adequately represented by a stationary model. Moreover, in majority of channels, propagation is over a limited number of relatively discrete modes. Using these two characteristics, the stationary channel model illustrated in Figure 2.3 is proposed by Watterson.

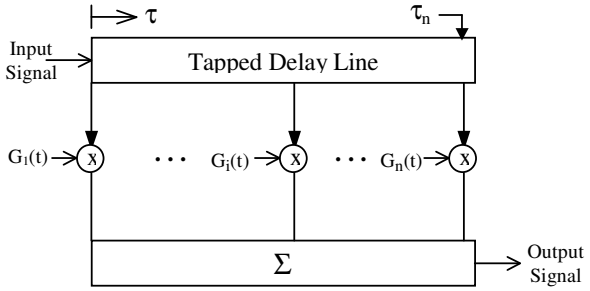

Figure 2.3: Watterson Channel Model

The input signal to the channel feeds an ideal delay line and is delivered at a limited number of paths with adjustable delays. Then, each delayed signal is modulated in amplitude and phase by a baseband tap-gain function $G_i(t)$, and all modulated signals are summed to form the output signal.

Moreover, the Watterson model involves three hypotheses for the statistical characteristics of the tap-gain functions:

i) Each tap-gain function is a complex Gaussian process that produces Rayleigh fading.
ii) Tap-gain functions are independent.
iii) Each tap-gain function has a spectrum that in general is the sum of two Gaussian functions of frequency, one for each magnetoionic component.

After measurements and analyses, Watterson et al. concluded that the proposed stationary channel model were valid [3] with some limitations on the bandwidth.

The validity was confirmed over a bandwidth of 2.5 kHz at the nighttime and 12 kHz at the daytime.

Although the Watterson model suffers some limitations, it has been widely used in order to compare the performances of HF modems. This model is also recommended in STANAG 4285 [1]. Besides multiplicative HF channel distortions the recommendation includes the addition of white Gaussian noise.

2.3 HF Modems

In the following sections, the HF modem types are presented and the required characteristics for STANAG 4285 HF modem is described.

2.3.1 Modem Types for HF Short Wave Communications

According to the number of signaling tones employed there are two types of modems, known as multi-tone and single-tone modems.

Multi-tone modems employ a symbol period, which is much longer than the maximum expected multipath delay in order to minimize the effects of inter-symbol interference (ISI) [22]. Hence, the inter-symbol distortion takes place only on the leading and trailing edges of the symbol. Using this fact, the receiver ignores the beginning of the symbol and operates on the undistorted part. The desired data rate is maintained by transmitting multiple signaling tones in parallel.

MIL-STD-118-110A standard [2] describes two multi-tone modems. One of them transmits 16 data tones employing differential QPSK to achieve a data rate of 2400 bps. Moreover, an unmodulated tone is added in order to correct frequency offset errors. The other multi-tone modem of MIL-STD-118-110A standard is a 39-tone modem, which has forward error control, whereas the 16-tone modem is uncoded.

In single-tone modems the symbol period is short compared to multipath spread and hence, ISI extends over several symbols. To overcome this channel distortion a training sequence is repetitively inserted into the transmitted data and an equalizer is employed at the receiver. The receiver makes use of this repeated

training sequence to obtain synchronization and correct the frequency offset error. STANAG 4285 is an example for single-tone modems.

The standard HF channels have the same bandwidth as the telephone channels. However, the HF channel has major impairments compared to telephone one. For example, the two channels have different types of fading. A telephone channel has flat fading whereas HF channels have frequency-selective fading, generally. For a single-tone HF modem training sequences are repeteadly inserted into the transmitted data which reduces the efficiency. Therefore, as the telephone modems can work up to 33.6 kbits/s, fast data rates for standardized military single-tone HF modems are only 2.4kbits/s.

2.3.2 STANAG 4285 Waveform

STANAG 4285 waveform is standardized for military short wave communication systems. In this section the required characteristics of the STANAG 4285 waveform is presented to ensure interoperability between modems transmitting data over HF radio links [1].

2.3.2.1 Modulation

The modulation technique is chosen to be phase shift keying of a sub-carrier of 1800 Hz with a modulation speed of 2400 bauds. Table 2.1 presents the values that the phase shift of the modulated signal may take relative to the unmodulated reference. Each phase shift value indicates a symbol number, as shown in the table.

Table 2.1: Phase Shift Values

Symbol Number	**Phase**
0	0
1	$\pi / 4$
2	$\pi / 2$
3	$3\pi / 4$
4	π
5	$5\pi / 4$
6	$3\pi / 2$
7	$7\pi / 4$

The complex number $\exp(jn\pi/4)$ is linked with the symbol number n. Figure 2.4 shows the symbols on the constellation diagram.

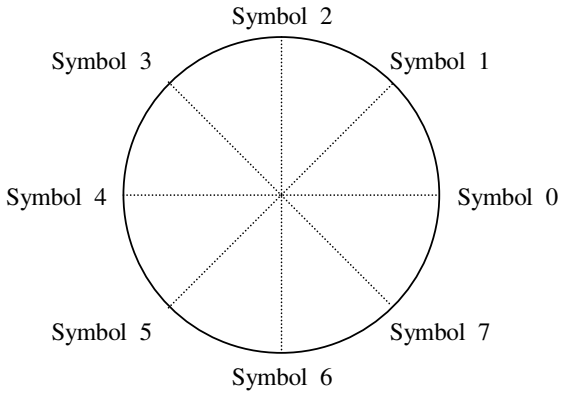

Figure 2.4: Phase State Encoding

2.3.2.2 Transcoding

Transcoding process is the operation of linking a symbol to be transmitted to a group of bits. The assignment is Gray coding in which the adjacent symbols differ only by one binary digit, so that the most likely errors caused by noise will result in a single bit error. 2-PSK, 4-PSK, and 8-PSK modulation techniques are employed for 1200 bps, 2400 bps, 3600 bps uncoded data rates, respectively.

- For 1200 bps uncoded data rate transcoding is achieved by linking one symbol to one bit according to the rule indicated in Table 2.2.

Table 2.2: Transcoding for 2-PSK

Bit	Symbol
0	0
1	4

- For 2400 bps uncoded data rate transcoding is achieved by linking one symbol to a set of two consecutive bits according to the rule indicated in Table 2.3.

Table 2.3: Transcoding for 4-PSK

Dibit	Symbol
0 0	0
0 1	2
1 0	6
1 1	4

Oldest bit Most recent bit

- For 3600 bps uncoded data rate transcoding is achieved by linking a symbol to a set of three consecutive bits according to the rule indicated in Table 2.4.

Table 2.4: Transcoding for 8-PSK

Tribit	Symbol
0 0 0	1
0 0 1	0
0 1 0	2
0 1 1	3
1 0 0	6
1 0 1	7
1 1 0	5
1 1 1	4

Oldest bit Most recent bit

2.3.2.3 Frame Structure

In STANAG 4285 waveform the symbols to be transmitted are structured in recurrent frames 106.6 ms in length. Figure 2.5 shows the frame structure.

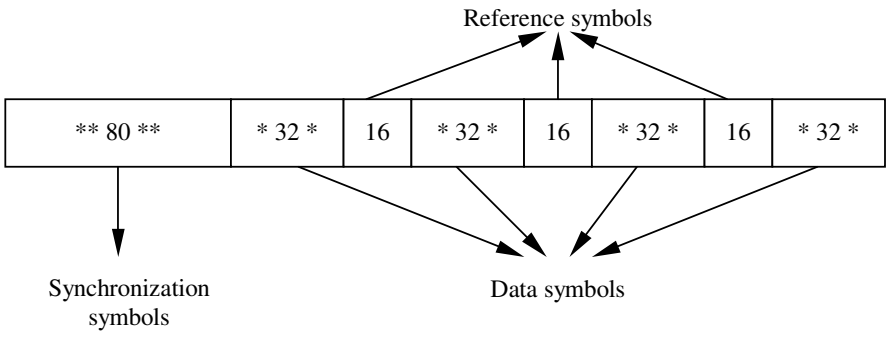

Figure 2.5: Frame structure

Every frame consists of 256 symbols. There are totally 80 synchronization, 48 reference, and 128 data symbols. The data symbols are broken into four 32-symbol length blocks and the reference symbols are broken into three 16-symbol length blocks. The number of data bits transmitted per frame is 128 at 1200 bps, 256 at 2400 bps, and 384 at 3600 bps uncoded transmission rate.

The synchronization and reference symbols are all known by the receiver. The modem uses the synchronization sequence for equalizer training, frequency shift correction and in order to detect the presence of the signal.

2.3.2.4 Synchronization Sequence Generator

The synchronization sequence consists of 80 symbols and is transmitted recurrently every 106.6 ms. This sequence uses 2-PSK modulation, whatever the data rate may be.

The generator polynomial for the synchronization sequence is $(x^5 + x^2 + 1)$. Figure 2.6 shows the generator structure.

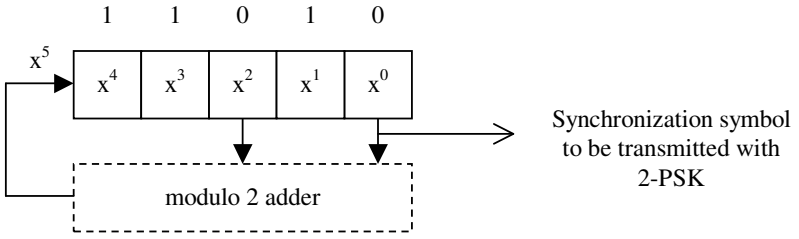

Figure 2.6: Synchronization Sequence Generator

At the beginning of every frame the generator is set to the initial value (1 1 0 1 0). The first synchronization symbol is identical to the least significant bit of the initial value (0). The remaining 79 symbols are obtained by applying the clock 79 times.

The sequence obtained by the generator is identical to a pseudo random sequence of length 31. This pseudo random sequence is repeated periodically

within the 80-symbol window. Consequently, the synchronization sequence consists of 2 periods of length 31 plus the first 18 symbols of another period.

2.3.2.5 Data Block Scrambling

The scrambling operation is performed only on data and reference symbols, which constitute data block. This operation consists of modulo 8 addition of the data and reference symbol number to the scrambling symbol number, which leads to complex multiplication of the data and reference symbol by the scrambling symbol.

The scrambling symbol generator is shown in Figure 2.7. The generator polynomial is $(x^5 + x^2 + 1)$.

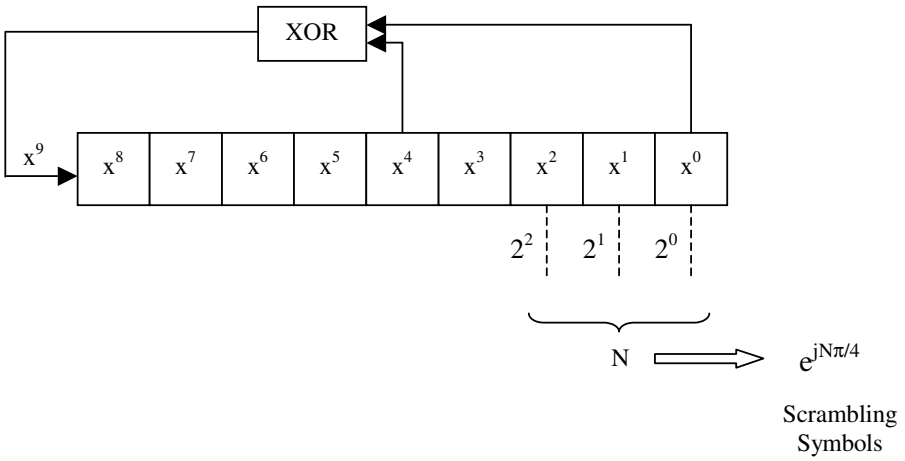

Figure 2.7: Scrambling Sequence Generator

The generator is initialized to 1 at the start of the each frame. Then, the symbols are derived from the triplet consisting of the last three bits in the PN register according to the following relationship:

Scrambling symbol $B_k = \exp(jn\pi/4)$

, where $n = 4x^2 + 2x^1 + x^0$ and x^2, x^1, x^0 may be 0 or 1.

The generation from one symbol to the next is achieved by successive shifting of the PN register by three positions. As shown in Figure 2.7, the scrambling symbols are independent from data and reference symbols.

The scrambling sequence is composed of 176 symbols and is repeated every 106.6 ms. Whatever the data rate may be (1200, 2400, or 3600 bps), data scrambling by an eight-phase-state sequence makes it possible to create an eight-phase-state data block.

2.3.2.6 Error Correction Coding

As the basis for error correction coding for all of the data rates, a constraint length 7, rate ½ convolutional encoder is used. Figure 2.8 shows the structure of the convolutional encoder.

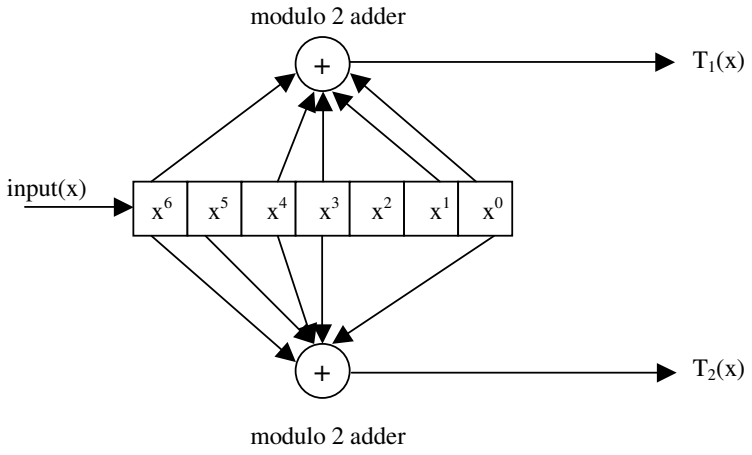

Figure 2.8: Convolutional Encoder

As shown in the figure, for each bit input to the encoder, two bits are taken as the output, the upper output bit $T_1(x)$ being taken first. The generator polynomials for $T_1(x)$ and $T_2(x)$ are as follows:

For T_1: $(x^6 + x^4 + x^3 + x^1 + x^0)$

For T_2: $(x^6 + x^5 + x^4 + x^3 + x^0)$

Uncoded data rates for STANAG-4285 waveform are 3600 bps, 2400 bps, and 1200 bps. Using convolutional encoding technique 2400 bps, 1200 bps, 600 bps, 300 bps, 150 bps, and 75 bps coded data rates can be obtained.

For 2400 bps coded data rate, 256 data bits are transmitted in every frame. Firstly, these data bits are coded by the convolutional encoder, resulting in 512 coded bits. Then, the puncturing operation, which is used only in 2400 bps coded data rate, is performed at the output of the interleaver. The puncturing technique, i.e., skipping every fourth bit at the interleaver output, reduces the number of bits to 384. Lastly, 128 data symbols are obtained by using 8-PSK modulation.

For 1200 bps coded data rate, 128 data bits are transmitted in every frame. The convolutional encoder forms a coded bit stream of length 256. The number of bits is the same at the input and at the output of the interleaver for 1200 bps coded data rate. Lastly, using 4-PSK modulation technique 128 data symbols are formed for the frame.

The number of data bits transmitted every frame is 64 for 600 bps coded data rate. At the output of the convolutional encoder 128 coded bits are obtained. Then, these coded bits are interleaved. Lastly, 2-PSK modulation technique is performed in order to obtain 128 data symbols for the frame.

For 300, 150, and 75 bps coded data rates, 32, 16, and 8 data bits are transmitted in every frame, respectively. The modulation technique used in these low data rates is 2-PSK as in 600 bps. In order to get 128 data symbols for every frame, the number of bits at the output of the convolutional encoder should be 128. This coded bit stream is generated by repeating the pairs of output bits the appropriate number of times. It must be noted that the bits are repeated in pairs rather than the repetitions of $T_1(x)$, followed by the repetitions of the second, $T_2(x)$. Table 2.5 summarizes the basic formats for error correction coding at each of the data rates.

Table 2.5: Basic Formats For Error Correction Coding

Coded Data Rate	Waveform Format Used	Effective Code Rate	Method For Achieving That Code Rate
2400 bps	8 Phase (3600 bps)	2 / 3	Rate 1/2 Punctured to Rate 2/3
1200 bps	4 Phase (2400 bps)	1 / 2	Unmodified Rate 1/2 Code
600 bps	2 Phase (1200 bps)	1 / 2	Unmodified Rate 1/2 Code
300 bps	2 Phase (1200 bps)	1 / 4	Rate 1/2 Code Repeated 2 Times
150 bps	2 Phase (1200 bps)	1 / 8	Rate 1/2 Code Repeated 4 Times
75 bps	2 Phase (1200 bps)	1 / 16	Rate 1/2 Code Repeated 8 Times

2.3.2.7 Interleaver and Deinterleaver

The interleaving technique used for STANAG-4285 waveform is a slight modification of a normal convolutional interleaver. Figure 2.9 shows a conceptual representation of a convolutional interleaver. For a normal implementation, coded bits are shifted into interleaver shift registers (rows) on the left side of the figure. On each new bit, the commutator switches into the next lower row. Each row has k more bits of storage than the preceding one above it. Bits are extracted through the output commutator in a similar way and transmitted. In the receiver, deinterleaving is performed by a similar operation. The only difference is that each shift register in the deinterleaver has k fewer bits of storage than the preceding one above it. For STANAG-4285 waveform, this normal interleaving technique is modified by making the commutators at the input of the interleaver and at the output of the deinterleaver cycle through all of the rows in a nonsequential pattern.

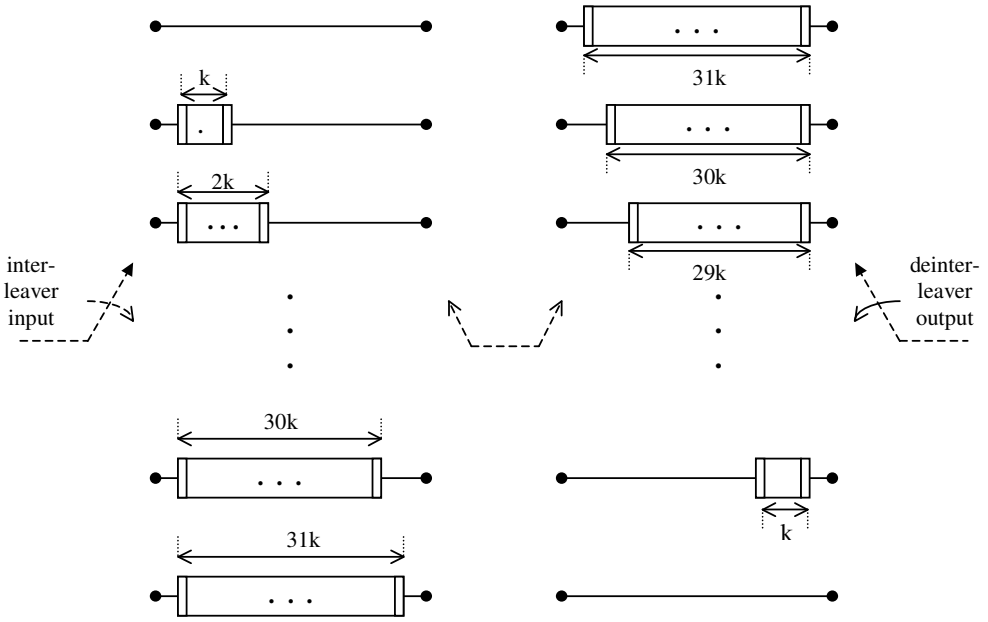

Figure 2.9: Convolutional Interleaver and Deinterleaver

According to the different delay increment k for each successive row, there are two types of interleavers used in STANAG-4285 waveform. The total interleaving delay is 10.24 seconds for the long interleaver, whereas 0.853 seconds for the short interleaver. Number of rows of both interleavers is 32 for all data rates. Table 2.6 shows the specific parameters of the interleaver to be utilized.

Table 2.6: Interleaver Parameters

	Delay Increment "k" For Each Successive Row	
Data Rates	Long Interleaver	Short Interleaver
2400 bps	48	4
1200 bps	24	2
600 – 75 bps	12	1

The commutator row sequence for outputting bits from the interleaver at all data rates other than 2400 bps is as follows:

This is the normal sequence for one complete cycle of 32 output bits.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31

For the 2400 bps coded data rate, 512 bits are taken from the encoder as the input of the interleaver in every frame. However, 384 bits must be taken out of the interleaver instead of 512 bits. Therefore, at the output of the interleaver every fourth row is skipped when taking bits out and this process is called puncturing. The commutator row sequence for outputting bits from the interleaver for the 8-PSK modulation used for the 2400 bps coded data rate is as follows:

This is one complete cycle of 24 output bits and is repeated for every eight successive 8-PSK symbols.

0, 1, 2, 4, 5, 6, 8, 9, 10, 12, 13, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26, 28, 29, 30

The commutator row sequence for inputting bits in the interleaver is as follows:

This sequence is generated by taking modulo 32 results of the multiplication of each number in the normal sequence by 9.

0, 9, 18, 27, 4, 13, 22, 31, 8, 17, 26, 3, 12, 21, 30, 7, 16, 25, 2, 11, 20, 29, 6, 15, 24, 1, 10, 19, 28, 5, 14, 23

The interleaver and deinterleaver are synchronized when the two center commutators shown in Figure 2.9 are synchronized, i.e., a bit taken from the i^{th} row of the interleaver is sent to the i^{th} row of the deinterleaver.

2.3.2.8 Initialization And Message Protocol

There are some initializations and message formatting to be used in STANAG 4285 when operating with the coding and interleaving. The shift registers for the interleaver and encoder should be initialized. Moreover, in order to indicate the

beginning and end of the message bits some message pattern should be inserted into the bit stream sent to the encoder. These rules can be stated as follows:

a) The encoder shift-register and the interleaver shift-registers should be set to all zeroes before the start of any message.

b) A unique 32-bit start of message pattern (SOM) is inserted into the bit stream sent to the encoder before the first bit of a message. The 32-bit word used for the SOM is as follows in hexadecimal format:

SOM = 03873C3C (the left most bit is the first bit sent)

c) A unique 32-bit end of message pattern (EOM) is appended after the last bit of the message. The 32-bit word used for the EOM is as follows in hexadecimal format:

EOM = 4B65A5B2 (the left most bit is the first bit sent)

d) A string of zeroes, equal in length to the interleaver delay plus 102, is appended to the EOM. The transmission of bits is terminated only after the last of these zeroes is input to the encoder. Using these zeroes, the contents of the shift registers of the interleaver and coder are set to all zeroes, getting ready for the next message.

e) The number of flush zeroes is different for each of the data rates and type of the interleaver as shown in Table 2.7.

Table 2.7: The Number Of Flush Zeroes

Data Rate	**Number Of Flush Zeroes**	
	Long Interleaver	Short Interleaver
2400 bps	24678	2150
1200 bps	12390	1126
600 bps	6246	614
300 bps	3174	358
150 bps	1638	230
75 bps	870	166

After insertion of SOM, EOM, and flush zeroes the bit stream to be transmitted takes the form as shown in Figure 2.10. The bits are sent to the encoder in the order stated as in the figure.

Figure 2.10: Data bits to be transmitted

CHAPTER 3

IMPLEMENTATION DETAILS

3.1 Introduction

The transmitter and the receiver programs of the STANAG 4285 HF modem are implemented on TIGER 5410/PC development board that features the Texas Instruments TMS320C5410 fixed point DSP processor. Both fixed point C and TMS320C54x Assembly codes are utilized throughout the development of the system. More information about the features of TIGER 5410/PC board can be found in Appendix A.

According to the type of the arithmetic there are two types of DSP processors, namely fixed point and floating point. Although floating point arithmetic provides much greater dynamic range, its implementation is generally slower and more expensive compared to fixed point implementation. Moreover, fixed point DSP processors require less power than their floating point counterparts.

The features of the TMS320C5410 DSP are described briefly in Section 3.2. Section 3.3 discusses the fixed point arithmetic and its numerical effects. In particular, the least mean square (LMS) algorithm and finite precision effects are discussed also. In the implementation, the LMS algorithm is employed for adapting the filter coefficients of the decision feedback equalizer (DFE) at the receiver. A review for DFE is given in Section 3.4.

3.2 TMS320C5410 Fixed Point DSP Processor

TMS320C5410 (’C5410) is a member of TMS320C54x (’C54x) 16-bit fixed point processor family [7]. It has an execution time of 10 ns for single cycle fixed point instruction and on-chip memory space of 64K words. Both data and program memory segments are composed of 16-bit words.

Although the word length is 16 bits, the arithmetic logic unit and accumulators of ’C54x are 40 bits wide, enabling extended precision. This feature is useful especially for filtering operations.

The memory maps for ’C5410 is given in Figure 3.1. The on-chip memory can be mapped into both data and program memory spaces by switching to overlay mode. In the overlay mode, on-chip RAM is mapped into both memory spaces as shown in Figure 3.1 [7].

Operating from on-chip memory has several advantages. Firstly, since no wait states are required, higher performance is achieved by using on-chip memory. Secondly, it requires lower power and cost than external memory. Therefore, it is worth the effort to use the memory efficiently.

Another critical point, which should be taken into account, is the execution time. As indicated before, a single cycle instruction has an execution time of 10 ns for ’C5410. Considering STANAG 4285, where frames to be transmitted are 106.6 ms in length, the maximum number of cycles available to process one frame becomes 10.6×10^6, namely the total allowable cycles spent by transmitter and receiver programs for one frame.

’C54x Assembly language has some application-specific instructions and addressing modes that provide efficiency in both speed and memory. In order to take full advantage of the specific features provided by ’C54x, the assembly language should be utilized for time-critical functions. Therefore, in the implementation most of the functions of the transmitter and receiver program are written in assembly language.

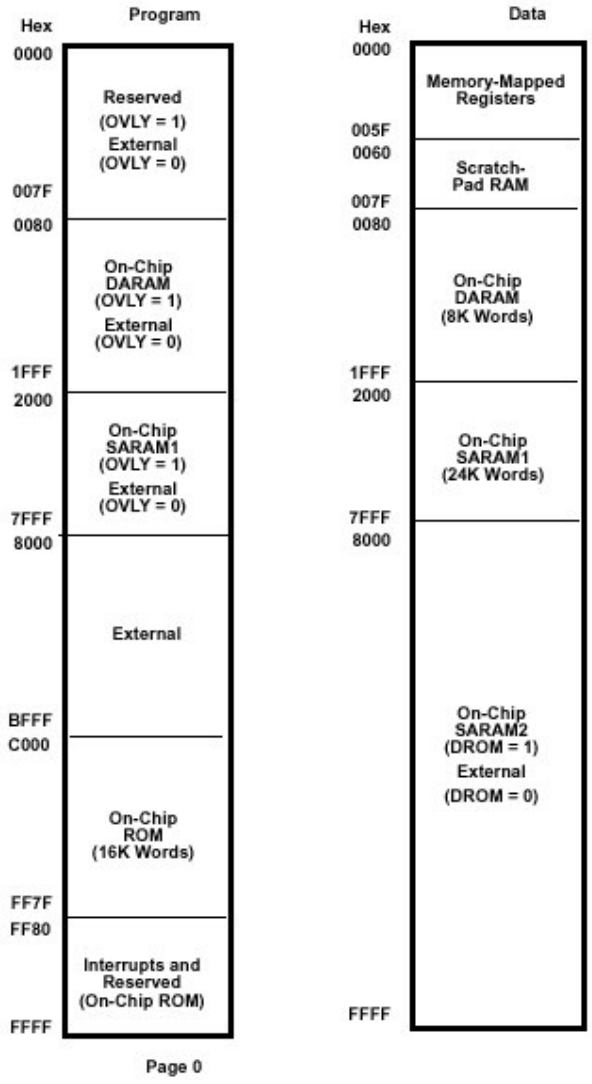

Figure 3.1: Memory maps of TMS320C5410

3.3 Fixed Point Implementation

In contrast to floating point format, fixed point implementation forces the user to understand the fixed point representation. Scaling, overflow handling, rounding are some examples of issues that should be taken into account when using fixed point format. Fixed point representation and its finite precision effects are discussed in the following subsections.

3.3.1 Fixed Point Representation

In fixed point implementation, signed or unsigned integers are used to represent fractional numbers. The range of numbers to be represented is selected by the use of a radix point within an integer. Figure 3.2 gives an example of a fixed point number with a length of 16 bits. The radix point marks the location where bits to the right represent a fractional base-2 addition to the number on the left of the radix point. The first bit to the right of the radix point represents 2^{-1}, the second 2^{-2}, and so on. For the example in Figure 3.2 the number represented is $2^{-1}+2^{-2}+2^{-3}$ yielding to 0.875. This representation is also called Q-point format representation. For this example, since there are 15 bits representing the fractional part, the number is said to be in Q15 format.

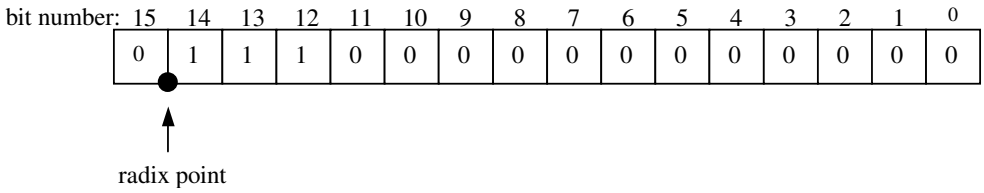

Figure 3.2: An example of a fixed point number

For Q15 format, the smallest non-zero magnitude, i.e. the resolution, is 2^{-15}. For smaller Q-point formats, such as Q14 and Q13, the resolution increases and hence

the accuracy decreases. Therefore, the maximum possible Q-point format should be used in order to obtain closer values to results of infinite precision.

The main difficulty in fixed point implementation arises from manipulation of numbers with different Q-point formats. Handling such numbers requires bit-wise shifting operation that increases execution time. Also, in order to prevent overflows the results of arithmetic operations should be scaled in some cases. As a consequence, it is necessary to make certain adjustments to the numbers when using fixed point format.

3.3.2 Quantization Errors

In a digital implementation of a filtering operation, quantization errors arise from analog to digital conversion and finite word-length arithmetic [8].

For a uniform quantization process with a step size Δ, which is sufficiently small, the quantization error can be assumed as uniformly distributed over the range $-\Delta/2$ to $\Delta/2$ [11]. Then, the variance of the quantization error is given by

$$\sigma^2 = \frac{\Delta^2}{12} \tag{3.1}$$

If each quantizing level is represented by B bits plus sign bit, the quantizer step size becomes 2^{-B}. Therefore, the variance of the quantization error becomes

$$\sigma^2 = \frac{2^{-2B}}{12} \tag{3.2}$$

In the implementation of STANAG 4285 HF modem, the input data and filter coefficients are represented generally by 15 bits plus sign bit. Hence, the variance of error has a value of $2^{-30}/12$.

Another source of the quantization error is finite word-length arithmetic. If no overflows take place, additions do not introduce errors. However, each multiplication introduces an error after the product is quantized. These errors may cause the performance of the implementation of the algorithms deviate from their

theoretical values. Especially, adaptive filtering algorithms are sensitive to finite word-length arithmetic.

As indicated before, the accumulators of 'C54x are 40 bits wide. Therefore, in a filtering operation it is not an advisable choice to quantize the product after each multiplication. Instead, the quantization process can be performed after accumulation of products.

3.3.3 LMS Algorithm and Finite Precision Effects

The least mean square (LMS) algorithm is a member of the family of the stochastic gradient algorithms [8]. The algorithm is used to adapt the weights of an FIR filter ($\mathbf{w}(n)$) to minimize the mean square error between the filter's output ($y(n)$)and the desired signal ($d(n)$). Representing the input vector as $\mathbf{u}(n)$ the LMS algorithm can be summarized as follows:

$$y(n) = \mathbf{w}^H(n)\mathbf{u}(n) \tag{3.3}$$

$$e(n) = d(n) - y(n) \tag{3.4}$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\mathbf{u}(n)e^*(n) \tag{3.5}$$

where $e(n)$ denotes the estimation error at *n*th time instant and μ represents the step size parameter.

Assuming that the input data $\mathbf{u}(n)$ and the weights $\mathbf{w}(n)$ are statistically independent, the LMS algorithm converges in the mean square if and only if the step size parameter μ satisfies the condition [8]

$$0 < \mu < \frac{2}{\lambda_{max}} \tag{3.6}$$

where λ_{max} is the largest eigenvalue of the correlation matrix $\mathbf{R}$ ($\mathbf{R}$=E{$\mathbf{u}(n)\mathbf{u}^H(n)$}).

The value of step size parameter also affects the steady state excess mean square error $\xi_{ex}(\infty)$. Using the independence assumption of $\mathbf{u}(n)$ and $\mathbf{w}(n)$ it can be shown that the excess error is proportional to the step size μ [12]

$$\xi_{ex}(\infty) \approx \frac{1}{2}\mu\xi_{min}\mathrm{tr}(\mathbf{R}) \tag{3.7}$$

where tr(**R**) denotes the trace of **R**. As a result, selection of a smaller step size decreases the excess mean square error.

On the other hand, the convergence rate of the LMS algorithm is inversely proportional to the step size value and directly proportional to the eigenvalue spread of **R**, which is defined as

$$\chi = \frac{\lambda_{max}}{\lambda_{min}} \tag{3.7}$$

Therefore, if the eigenvalue spread of **R** is wide or the step size parameter has a small value then it takes a long time for the LMS algorithm to converge. Consequently, the selection of the step size parameter plays an important role in both convergence rate and excess error value.

Finite precision has some significant effects on LMS algorithm. In [9] it is shown that the total output mean squared error (MSE) has a term due to the error $\Delta\mathbf{w}(n)$ in the quantized tap-weight vector $\mathbf{w}_q(n)$. Moreover, this contribution to the total error is inversely proportional to the step size parameter μ. Therefore, in practice the step size parameter can only be decreased to a level at which the effects of quantization errors in the tap weights are not significant [8].

Another phenomenon in finite precision LMS algorithm is known as stalling [9], [13]. This phenomenon occurs when the update term in the algorithm recursion becomes smaller in magnitude than the least significant bit of the tap weight. Therefore, reduction of step size parameter value μ beyond a certain value increases the steady state MSE due to the stalling phenomenon. Another result obtained in [9] and [13] is that the steady state MSE approaches the performance of the analog LMS algorithm as the number of bits in the digital implementation approaches infinity.

In some cases the tap weights in the LMS algorithm may attain large values although the inputs, disturbances, and estimation errors remain bounded [14]. This unbounded behavior of parameters is called as parameter drift, which may be viewed as a hidden form of instability [8].

3.4 Review of the Decision Feedback Equalizer (DFE)

Because inter-symbol interference (ISI) can be quite severe at times for HF transmission, the nonlinear equalizer DFE, which was first introduced by Austin in 1967 [15], is to be preferred over the linear equalizer. The linear equalizer only passes the received symbols through a single transversal filter, but the DFE also uses previous decisions.

In order to understand the idea of the DFE, consider a baseband channel with impulse response sequence $\{h_n\}$. In the absence of noise, the response of the channel to an input sequence $\{x_n\}$ is given by

$$y_n = \sum_k h_k x_{n-k} = h_0 x_n + \sum_{k<0} h_k x_{n-k} + \sum_{k>0} h_k x_{n-k} \tag{3.8}$$

The second and third terms in (3.8) are due to the precursors and postcursors of the channel impulse response, respectively. The DFE is composed of feedforward and feedback sections as shown in Figure 3.3. The feedforward section is identical to the linear equalizer and the feedback section is used to remove the ISI caused by the previously detected symbols, i.e. due to postcursors of the channel impulse response. However, decision errors lead to residual ISI which may increase the probability of error of future decisions. Therefore, error propagation may reduce the effectiveness of DFE. Analyses of the error propagation for DFE can be found in [16], [17], [18].

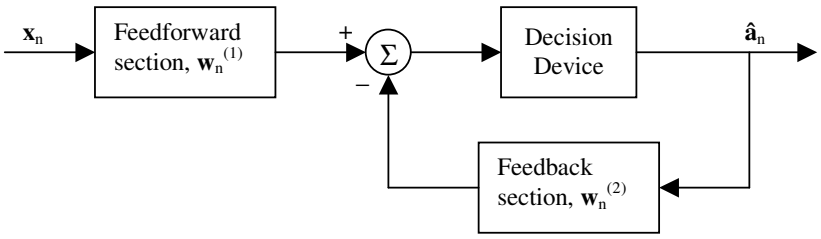

Figure 3.3: DFE structure

It is known that the fractionally spaced equalizers (FSE) have an advantage in that they are less sensitive to timing phase [19]. Noting the performance of DFE in a multipath fading environment, the fractionally spaced DFE structure was first suggested by [20] and [19]. The structure of a fractionally spaced DFE, which has tap spacings at half the symbol interval on the forward taps, is given in Figure 3.4.

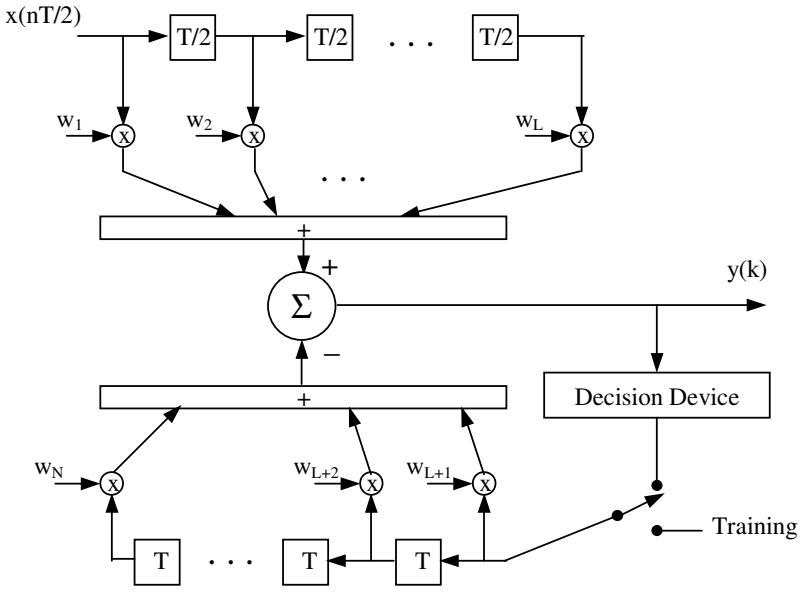

Figure 3.4: Fractionally spaced DFE structure

Since LMS algorithm is employed for adapting the filter coefficients of DFE in the implementation, finite precision also affects the performance of DFE.

CHAPTER 4

TRANSMITTER PROGRAM

4.1 Introduction

The transmitter program initializes transmitter parameters and registers; takes the data bits from an input bit file and after operations according to the message protocol it produces frames. Then, the frames are written to an output file in binary format. Both C and ‘C54x Assembly programming languages were employed while developing the transmitter functions. The block diagram of the transmitter is given in Figure 4.1.

During the formation of the frames, firstly, the data bits are coded through the convolutional encoder in the transmitter. Then, the coded bits are sent to the interleaver as input. Using the corresponding modulation technique for selected data rate, the data symbols are formed from the interleaved data symbols. After that, the synchronization, data, and reference symbols are arranged in order to constitute the frame structure shown in Figure 2.5. The next operation performed is data block scrambling. Only data and reference symbols, that constitute the data block, are scrambled. While synchronization sequence is transmitted in 2-PSK format, the data block forms eight-phase-state sequence because of the scrambling process. After scrambling, transmission filtering and transposition to 1800Hz processes are performed by the transmitter. The flowchart of the transmitter operations is given in Figure 4.2.

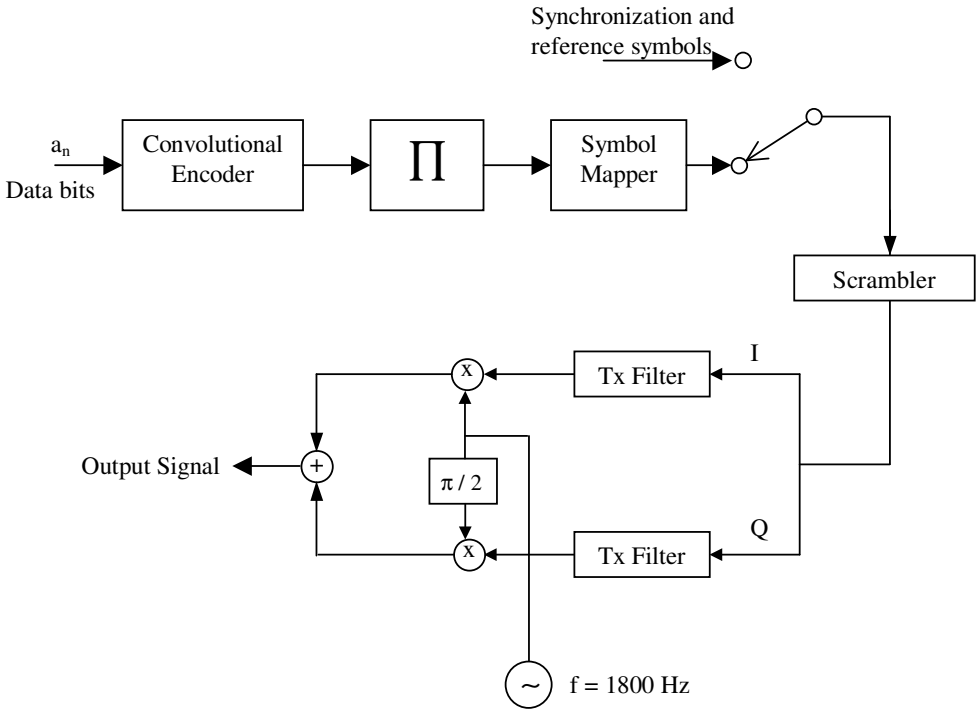

Figure 4.1: Block diagram of the transmitter

(Π denotes interleaver)

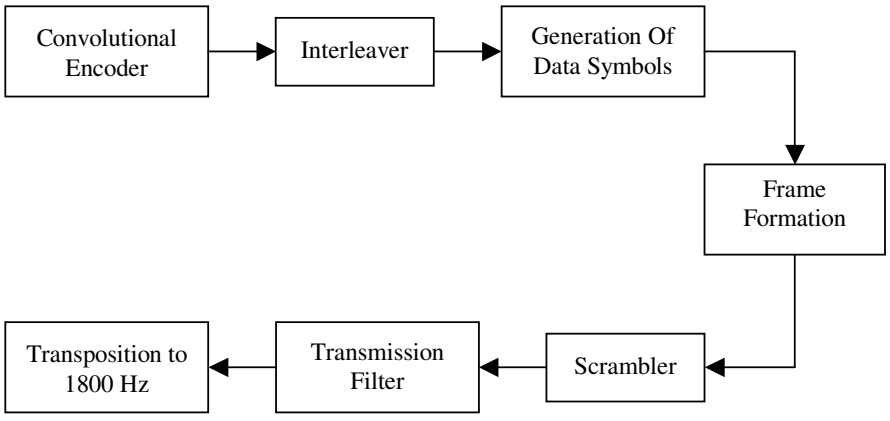

Figure 4.2: Flowchart of the transmitter operations

4.2 Implementation of the Transmitter Program

There are mainly three processes performed by the transmitter program. Firstly, at the beginning of a message transmission, the transmitter parameters and registers are initialized by the program. Secondly, start of message pattern (SOM), end of message pattern (EOM), and flush zeroes are inserted into the message bits in order to conform to the message protocol. Lastly, the program forms frames using the required characteristics for STANAG 4285 waveform. The main function of the transmitter program is the "trans_main" function. It performs initializations and message protocol; and calls the "transmitter" function for the formation of frames.

The transmitter program includes many functional blocks. The following subsections present these functional blocks in terms of algorithms utilized, their memory usage and cycle counts per frame. The CPU cycles given are measured when both data and program memory spaces are allocated to on-chip memory. Obviously, using external memory will increase the number of cycles spent.

4.2.1 Initializations

When the transmitter program starts to execute, first operation performed by the trans_main function is initialization. After the user selects the coded data rate and the interleaver type, the transmitter parameters are initialized by the function. It sets the new message flag to 1 and the end of message flag to 0, meaning that the trans_main function is called for the first time and the start of the message pattern (SOM) should be transmitted before message bits. Moreover, arrays and variables, such as concerning the modulation type, the number of bits transmitted per frame (given in Table 4.1) are initialized according to the selected data rate. Then, the shift register of the convolutional encoder is filled with all zeroes. Lastly, the initialization of the interleaver is performed setting the shift registers to all zeroes.

The cycle counts for the initialization process are measured as 11466 with long interleaver and 6269 with short interleaver. The difference is caused by the number

of shift registers to be set to zeroes. The program memory allocated for initialization is 133 words.

Table 4.1: Number Of Bits Transmitted Per Frame

Coded Data Rate	Bits per Frame
2400 bps	256
1200 bps	128
600 bps	64
300 bps	32
150 bps	16
75 bps	8

4.2.2 Message Protocol

As stated in Section 2.3.2.8, the transmitter inserts a unique 32-bit start of message pattern (SOM) into the beginning of the message bits and appends a unique 32-bit end of message pattern (EOM) after the message bits. These message patterns are required for the receiver to detect the start and end of message bits. Moreover, flush zeros should be appended after the EOM, in order to fill the encoder shift register and interleaver shift registers with zeroes.

The trans_main function checks the start of message and end of message flags to transmit SOM, message bits, EOM, and flush zeroes in the correct order. The algorithm used for the message protocol is given in Figure 4.3.

Firstly, the start of message flag is checked by the function. If it is 1, namely, if no message bits have been transmitted yet, 32-bit SOM pattern is to be transmitted. However, this process exhibits differences between the different data rates. As given in Table 4.1, the number of bits transmitted per frame is less or equal to 32 for data rates less than 600 bps. Therefore, 32-bit SOM is transmitted in 4 frames for 75 bps, in 2 frames for 150 bps and in one frame for 300 bps. However, since the number of bits transmitted per frame is greater than 32 for other data rates, also

message bits are transmitted in the first frame. In that case, the transmitter appends 224 message bits for 2400 bps, 96 message bits for 1200 bps and 32 message bits for 600 bps at the end of 32-bit SOM. After completion of transmitting SOM pattern, the start of message flag is set to 0.

The transmitter takes the message bits from the input bit file on PC and checks if the end of file is reached or not. The end of file flag is initially set to 0. However, if the end of file is reached, the transmitter sets the flag to 1. While the end of message flag is 0, the transmission of message bits continues. When it is 1, the transmitter appends 32-bit EOM pattern and flush zeroes at the end of message bits. After completion of transmitting last flush zero, the program terminates.

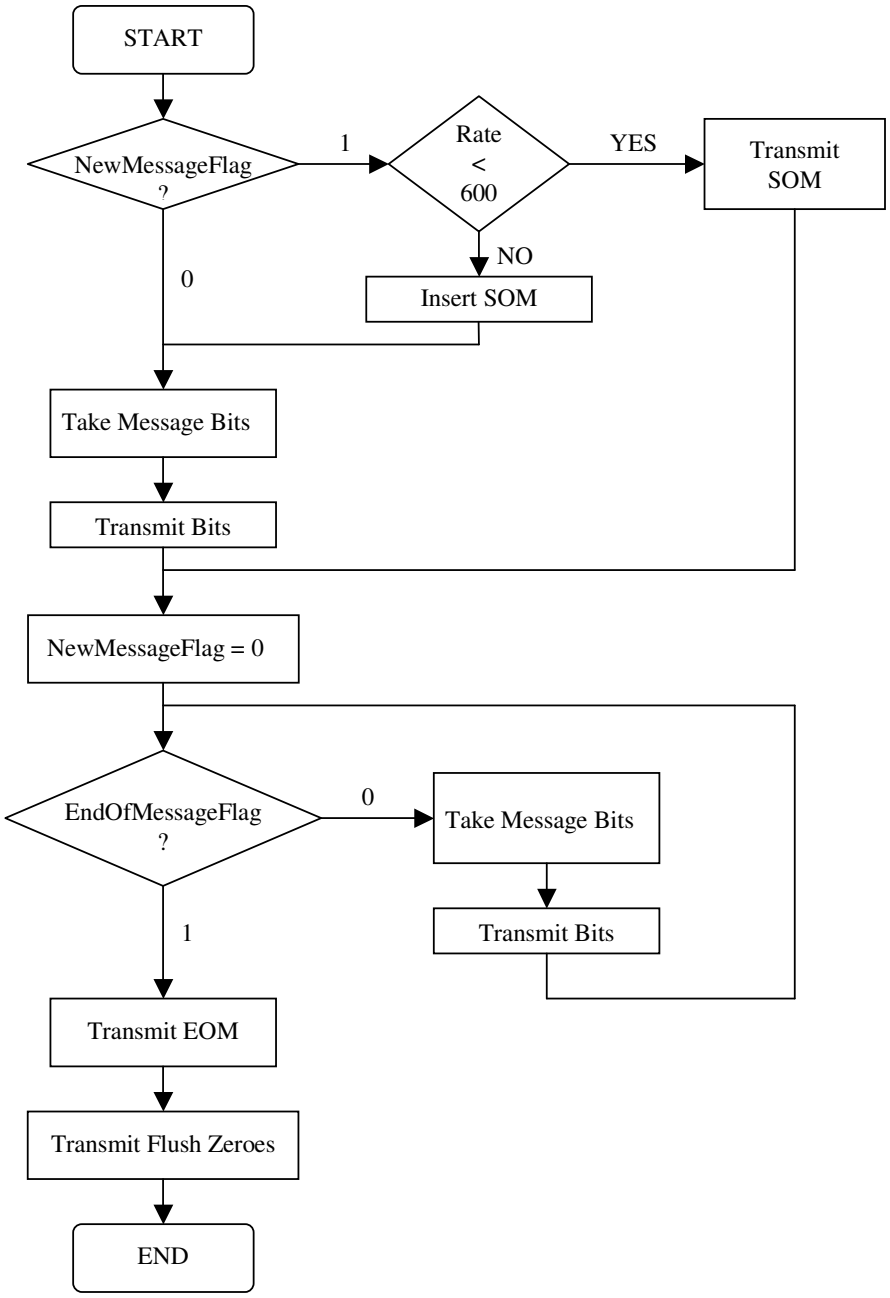

Figure 4.3: Flowchart of the Transmission Control

4.2.3 Convolutional Encoder

The SOM, message, EOM, and flush zero bits constitute the data bits that should be processed and transmitted as a sequence of frames. The trans_main function calls the transmitter function in order to process data bits and form frames. The first operation carried out by transmitter function is encoding. As stated in Section 2.3.2.6, the convolutional encoder used for STANAG 4285 has a rate of 1/2 and constraint length of 7. The number of input and output bits of the encoder is given in Table 4.2.

Table 4.2: Number of input and output bits of the encoder

Coded Data Rate	No. Of Input Bits	No. Of Output Bits
2400 bps	256	512
1200 bps	128	256
600 bps	64	128
300 bps	32	64
150 bps	16	32
75 bps	8	16

For every input bit I(x) the encoder outputs two bits, $T_1(x)$ and $T_2(x)$. For data rates 300, 150, and 75 bps, the output bit pairs are repeated 2, 4, and 8 times, respectively, in order to obtain 128 bits for 2-PSK modulation.

Since the number of input bits to the encoder is different for data rates, the execution speed of the encoding process exhibits differences between them as shown in Table 4.3. It requires 5944 cycles for 2400 bps and as the data rate decreases the cycle counts decreases also. The program memory requirement of encoding process is 330 words.

Table 4.3: Cycle counts for encoding process

Data Rate	Cycle Counts
2400 bps	5944
1200 bps	3000
600 bps	1528
300 bps	920
150 bps	616
75 bps	464

4.2.4 Interleaver

There are two interleaving options for STANAG 4285, long and short interleaving. As indicated in Section 2.3.2.7, the total interleaving delays are 10.24 and 0.853 seconds for long and short interleavers, respectively.

The delay increment between successive shift registers is the largest for 2400 bps data rate with long interleaver as shown in Table 2.6. For that data rate, the total number of delay bits in the long interleaver is 23808. If every bit of a memory word were used for shift registers, then a data memory of 1488 words would be reserved for long interleaver:

$$1 \text{ word} = 16 \text{ bits} \tag{4.1}$$

$$23808 / 16 = 1488 \text{ words} \tag{4.2}$$

Since the delay increment is 48 for 2400 bps, then every row would have 3 more words than the preceding one above it. However, considering other data rates with long interleaver, the values of the delay increment are 24 and 12. Therefore, the most convenient way to reduce the complexity is to reserve only 12 bits of a 16-bit memory word for shift registers. With this choice, the difference in successive rows of long interleaver becomes 4 words of memory for 2400 bps, 2 words for 1200 bps and 1 word for other data rates. In this case, a data memory of 1984 words should be reserved for long interleaver:

$$23808 / 12 = 1984 \text{ words} \quad (4.3)$$

The values of delay increment are much smaller for short interleaver compared with the long one as shown in Table 2.6. Maximum value of delay increment is 4 and in this case, the total number of delay bits becomes 1984. Since this value is equal to the number of words used for long interleaver, every word of memory can be reserved for a delay bit. Therefore, total memory of 1984 words is required for short interleaver as for the long one. Consequently, both types can use the same data memory.

The speed of the interleaving process plays an important role for the overall speed of the transmitter. In order to reduce the processing time, the interleaver employs circular addressing mode. The number of cycles spent by interleaving process is given in Table 4.4.

Table 4.4: Cycle counts for interleaving process

	Cycle Counts	
Data Rates	Long Interleaver	Short Interleaver
2400 bps	22851	17378
1200 bps	10256	7223
600 – 75 bps	5196	3679

Since every delay bit of a shift register utilizes a whole word in short interleaver, there is no need to shift the words. Instead, overwriting by circular addressing is adequate in order to write into or read from the interleaver. However, 12 delay bits employ a word of memory in long interleaver and hence, shifting operation is needed. Therefore, two types of interleavers exhibit differences in cycle counts for the same data rates.

The long and short interleavers require 110 and 141 words of program memory, respectively. The algorithms utilized lead to this difference.

4.2.6 Generation of Data Symbols

The first operation in which fixed point numbers are used is the generation of data symbols. The transmitter employs 8-PSK, 4-PSK, and 2-PSK modulation techniques for 2400 bps, 1200 bps, and other coded data rates, respectively. The interleaved data bits are assigned to PSK symbols using the transcoding process as explained in Section 2.3.2.2. For each data rate 128 data symbols are generated in a frame, totally.

The PSK symbols are stored in Q15 format. As seen in Figure 2.4, the real part of Symbol-0 and the imaginary part of Symbol-2 are 1. Therefore, saturation occurs when storing these symbols in Q15 format.

Table 4.5 gives the information about the number of cycles spent for generation of the data symbols. The number of input bits is 384, 256, and 128 for 2400 bps, 1200 bps, and other coded data rates, respectively. Therefore, the number of cycles spent for the operation is different for different data rates. Moreover, the program memory allocated for the process is 65 words.

Table 4.5: Cycle counts for generation of data symbols

Coded Data Rate	Cycle Counts
2400 bps	4559
1200 bps	3918
600-75 bps	3277

4.2.7 Frame Formation

Every frame in STANAG 4285 consists of synchronization, reference and data symbols. After generation of data symbols, all of the symbols should be grouped and arranged according to the frame structure as explained in Section 2.3.2.3.

The number of CPU cycles required for this process is 937 and the program memory space allocated is 106 words.

4.2.8 Scrambling

The scrambling process is performed only on data and reference symbols. The only operation performed is complex multiplication of data and reference symbols by scrambling symbols. Since there is no possibility of saturation, the outputs of scrambling process are stored in Q15 format.

Figure 4.4 shows the scrambling operation. As indicated in 2.3.2.5, the scrambling sequence consists of 8-PSKsymbols, which are independent of the data and reference symbols and are repeated each frame. After scrambling operation, the data and reference symbols become 8-PSK, whereas the synchronization sequence is transmitted as 2-PSK.

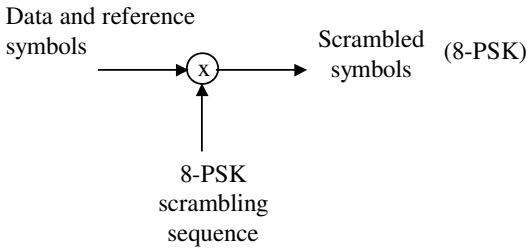

Figure 4.4: Scrambling operation

The scrambling process requires 1901 CPU cycles and 33 words of program memory.

4.2.9 Transmission Filtering

For the ideal channel case, the transmission and reception filters, $h_T(t)$ and $h_R(t)$ respectively, should be matched in order to maximize SNR in the receiver. Moreover, they should be so chosen that an intersymbol interference (ISI) free transmission is achieved. ISI free transmission can be accomplished if overall signal pulse shape at the receiver matched filter output satisfies the Nyquist

criterion. A widely used pulse in practice with desirable spectral properties is the raised cosine pulse [4]. The raised cosine spectrum is given as

$$G(f)=\begin{cases} T & \cdots\cdots 0\le |f|\le \dfrac{1-\alpha}{2T} \\ \dfrac{T}{2}\left\{1+\cos\left[\dfrac{\pi T}{\alpha}\left(|f|-\dfrac{1-\alpha}{2T}\right)\right]\right\} & \cdots\cdots \dfrac{1-\alpha}{2T}\le |f|\le \dfrac{1+\alpha}{2T} \\ 0 & \cdots\cdots \text{otherwise} \end{cases} \tag{4.4}$$

where α is called the rolloff factor, and it is in the range $0 \le \alpha \le 1$. The expression of the overall pulse shape in time domain is

$$g(t) = h_T(t) * h_R(t) \tag{4.5}$$

where "$*$" denotes convolution. Since the reception filter is matched to the transmission filter, their impulse responses satisfy the relation

$$h_R(t) = h_T^*(-t) \tag{4.6}$$

where "*" denotes complex conjugation. Therefore, the spectral shape of the overall pulse can be simplified as

$$G(f) = H_T(f)H_R(f) = H_T(f)H_T^*(f) = |H_T(f)|^2 \tag{4.7}$$

by using(4.6). As a result, the responses of the transmission and reception filters satisfy the relation

$$H_T(f) = H_R(f) = \sqrt{|G(f)|} \tag{4.8}$$

The pulse shape used for transmission and reception filters in STANAG 4285 is a root-raised cosine pulse shape with a rolloff factor of 0.2. The pulse is sampled at N = 4 samples/symbol and is truncated to span 10 symbols. Therefore, the filtering operation leads to 1024 samples per frame at the output. Time characteristic of this pulse is given in Figure 4.5.

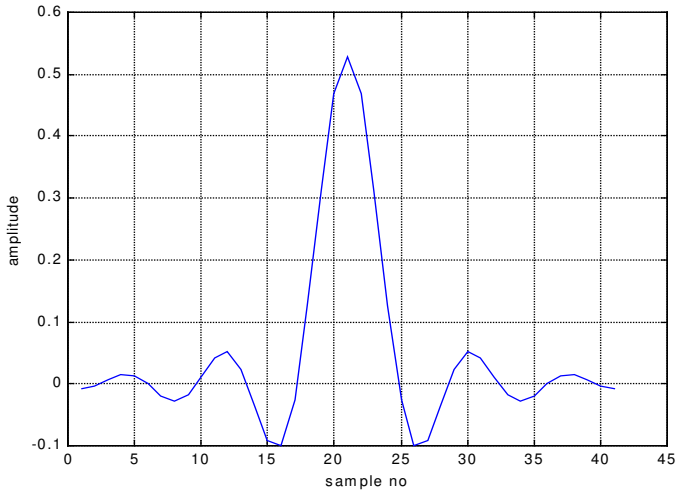

Figure 4.5: Impulse response of transmission filter

The filter coefficients and symbols are all stored in Q15 format. Since there is no probability of saturation, the output of the filter is also stored in Q15.

The filtering operation requires 125173 cycles of execution time. This number may seem to be a bit large, but taking into account the filter length and the number of samples to be filtered it can be considered as moderate. Moreover, the program memory space allocated for filtering operation is 69 words.

4.2.10 Transposition to Intermediate Frequency at 1800 Hz

Transposition to intermediate frequency is the last operation performed by the transmitter program. This process includes multiplication of in-phase and quadrature phase channel signals with cosine and sine signals at 1800 Hz respectively, and addition of multiplication results. Figure 4.6 shows the transposition to intermediate frequency (IF).

Using the relationship between the sample rate 4/T (9600 Hz) and the intermediate frequency 1800 Hz, a look-up table of length 16 is required for both sine and cosine signals in Q15 format. The output of the modulation process is also stored in Q14 format.

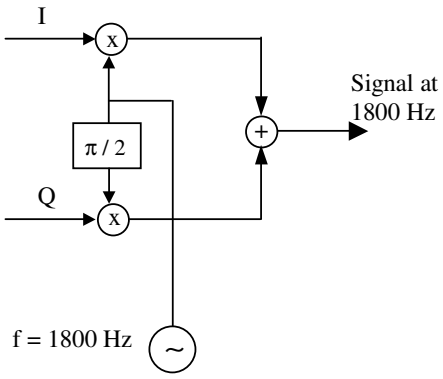

Figure 4.6: Block diagram of transposition to IF frequency

The number of CPU cycles used by the process is 5228 and the program memory space requirement is 46.

4.3 Resource Requirements of The Transmitter Program

The transmitter program implemented on ’5410 DSP processor requires 3.87 Kwords of data and 3.37 Kwords of program memory space, totally. The data memory is mainly occupied by the interleaver with 1984 words. Constant arrays, such as transmission filter coefficients, synchronization and scrambling symbols use the remaining part of the data memory. Considering program memory space, message protocol routine uses most of the memory with 1.5 Kwords.

The execution time of the transmitter program varies with data rates. Since the number of bits to be transmitted in 2400 bps coded data rate is larger than the others, the CPU cycles required is the largest one for this data rate. Table 4.6 shows maximum CPU cycles used for different data rates.

Table 4.6: Cycle counts for transmitter program

Data Rate	Cycles
2400 bps	169624
1200 bps	153437
600 bps	146262
300 bps	145685
150 bps	145364
75 bps	145205

As shown in Table 4.6 the maximum number of cycles required for transmitter program is 169624 per frame. Since every frame has a length of 106.6 ms, approximately 10.6×10^6 cycles is available for 'C5410 in order to process the frame. As a consequence, the transmitter program uses maximum 1.6% of the time available for processing the frame.

CHAPTER 5

RECEIVER PROGRAM

5.1 Introduction

The STANAG 4285 waveform is suitable for reception using a variety of receive algorithms. In this implementation, the receiver structure is based on a fractionally spaced LMS DFE. In order to improve the convergence of the LMS algorithm a multipass technique is employed for training the equalizer.

The block diagram of the receiver is given in Figure 5.1. The following sections present the implementation details of the receiver program.

5.2 Implementation of the Receiver Program

The receiver program mainly consists of two functions. The first function, 'rec_main', performs initializations, reads 1024 samples from the input file where the sample rate is 4/T (9600Hz) and then calls the 'receiver' function, which demodulates the signal and decodes bits. The flowchart of the 'receiver' function is given in Figure 5.2 There are two modes of the receiver program, acquisition and tracking. Firstly, the receiver is set to acquisition mode and seeks to detect the synchronization sequence within the received samples. If the synchronization is obtained, then the receiver switches to tracking mode.

The following subsections describe the functional blocks of the receiver program with the resource requirements for ’C54x.

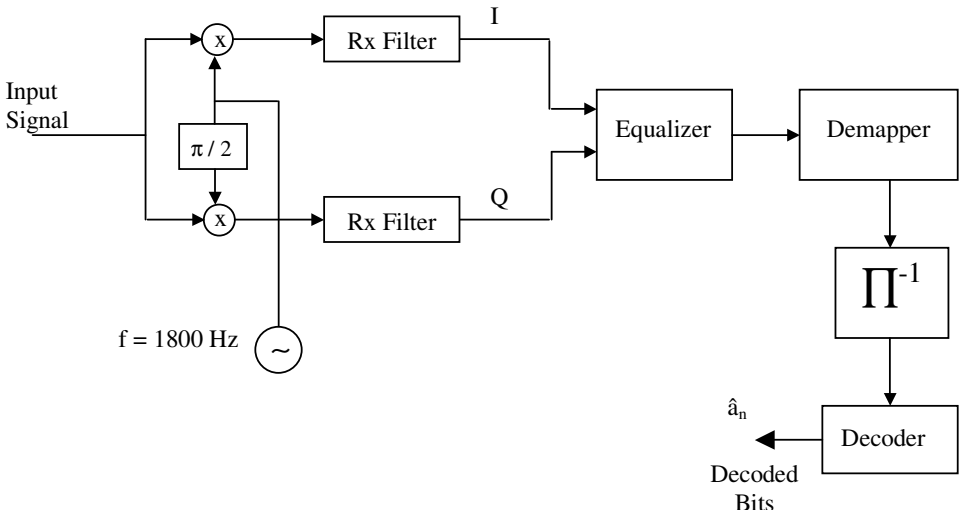

Figure 5.1: Block diagram of the receiver

(Π^{-1} denotes deinterleaver)

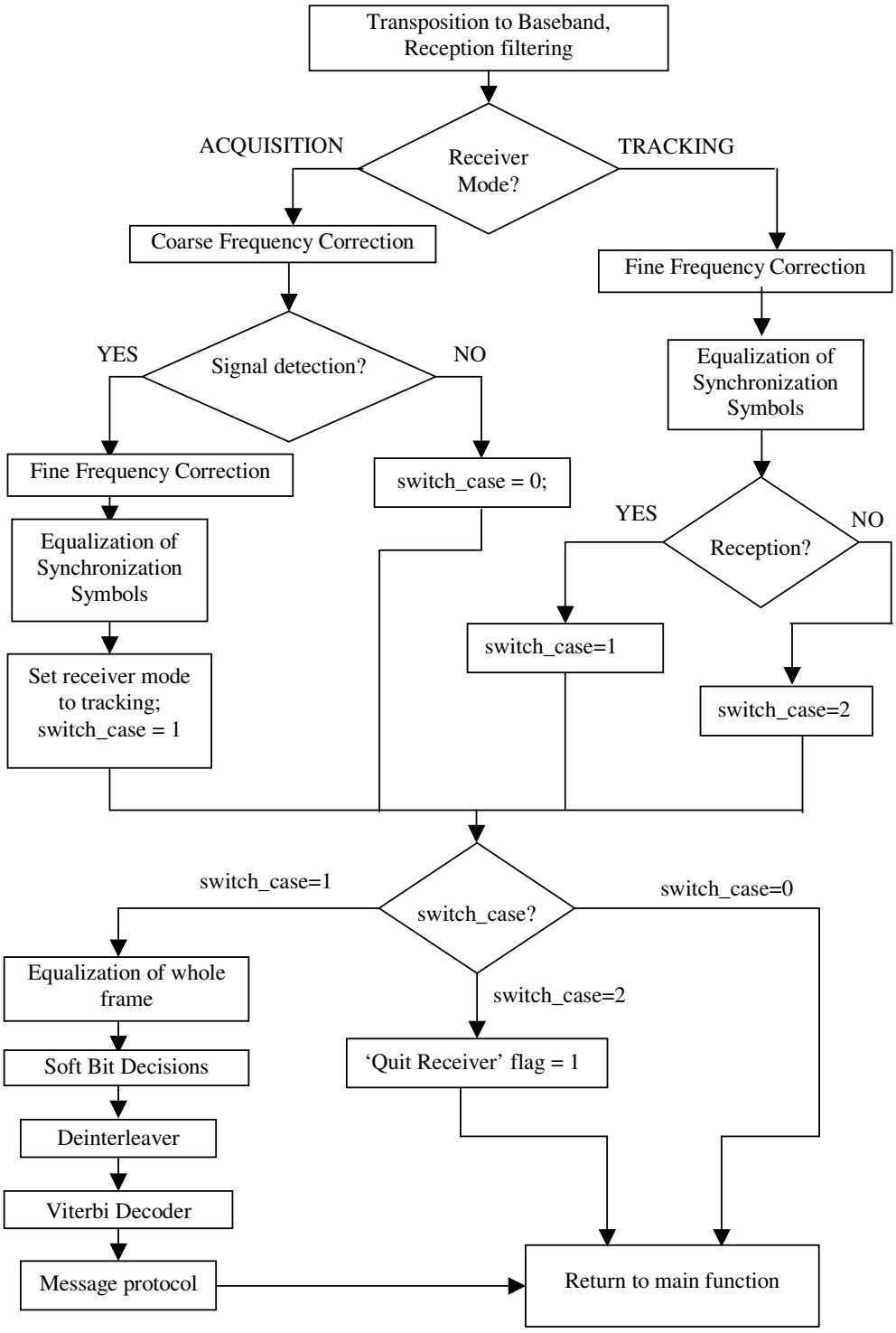

Figure 5.2: Flowchart of the receiver function

5.2.1 Initializations

The first operation performed by the receiver program is initialization. After the user selects the coded data rate and the interleaver type, the receiver parameters are initialized firstly. Main operations performed can be summarized as follows:

- The receiver mode is set as acquisition, i.e. the synchronization sequence has not been detected yet. The other receiver mode is tracking.
- 'Start of message' flag is set to 0, i.e. the SOM sequence has not been detected within the decoded bits yet. When SOM is detected, this flag will be set as 1.
- 'End of message' flag is set to 0, i.e. the EOM sequence has not been detected yet.
- 'Quit receiver' flag is set to 0. When its value becomes 1, the receiver program terminates.
- 'Viterbi' flag is set to 0. Viterbi decoding process is not performed unless this flag is set to 1.
- 'Output data' flag is set to 0. The receiver waits for the detection of the SOM sequence in order to output the decoded bits.

In addition to receiver parameters, the program initializes the data memory used by the deinterleaver, equalizer and Viterbi decoder. The program memory allocated by the initialization routine is 350 words and 26000 cycles are required for the process.

5.2.2 Transposition to Baseband and Reception Filtering

Transposition to baseband is the first operation performed on the received samples regardless of the receiver mode. This process includes the multiplication of the received signal with cosine and sine signals at 1800 Hz and reception filtering. The reception filter is the same of the transmission filter explained in Section 4.2.9.

The input to this process is in Q15 format, but the output is stored in Q14 format in order to prevent the saturation. The operation spends 118473 cycles, mainly due to reception filtering. Moreover, it requires a program memory of 118 words.

5.2.3 Signal Detection

In order to process received data and decode bits the receiver should firstly detect the frame out of the received samples. Every frame has the synchronization sequence of length 80 symbols as a first block. Therefore, this sequence is used for the detection of the signal.

The synchronization sequence consists of 2 periods of a 31-symbol pseudo random (PN) sequence and first 18 symbols of that sequence. The signal detection algorithm is based on this periodicity [1].

The first operation in signal detection is correlating the sequence of received input samples g(i) with the 31-symbol length reference PN sequence $x_{pnref}(i)$:

$$v(i) = g(i) \circ x_{pnref}(i) \tag{5.1}$$

where "$\circ$" denotes continuous correlation and v(i) represents the output signal of the correlator. This process is performed at sample rate (4/T).

After obtaining the correlator output sequence, the energy E(i) over v(i) is calculated within a sliding window using the following equation:

$$E(i) = \sum_{k=0}^{NH} \left| v(i+k) \right|^2 \tag{5.2}$$

where NH is the maximum length expected for the channel impulse response.

Because of the periodicity of the synchronization sequence, E(i) will also be periodic [1]. The maximums of E(i) are obtained for two periods of the transmitted PN sequence, and the minimum is obtained in the middle between two periods [1]:

$$E_{peak1}(i) = E(i)$$
$$E_{peak2}(i) = E(i+31*4) \tag{5.3}$$

$E_{middle}(i) = E(i+31*4/2)$

Lastly, the ratios $E_{peak1}(i)/ E_{middle}(i)$ and $E_{peak2}(i)/ E_{middle}(i)$ are compared with a threshold. If both ratios are grater than the threshold then the synchronization is achieved.

The algorithm block diagram for signal detection is given in Figure 5.3.

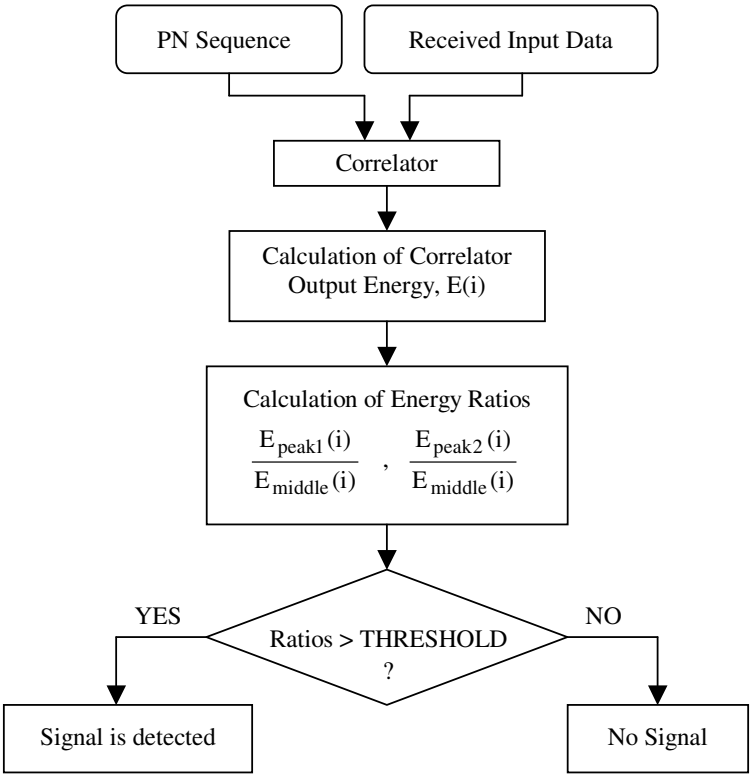

Figure 5.3: Block diagram for signal detection process

Signal detection routine requires variable number of cycles from the CPU. In the worst case, i.e. when the synchronization sequence cannot be detected within 1024 samples, the operation requires 360938 cycles. Moreover, 179 words of program memory are allocated for the routine.

5.2.4 Frequency Offset Correction

According to the required characteristics stated in [1], the modem must be capable of tolerating a frequency error of ± 75 Hz between the transmission and reception HF carriers (transmitter/receiver frequency error and Doppler shift included) and rate of frequency change up to 3.5 Hz. There are two types of frequency offset correction methods used in the implementation, coarse and fine frequency correction methods.

Coarse frequency correction method is based on spectral analysis where 2048 point Fast Fourier Transform (FFT) operation is performed on square of the input signal.

As indicated in Section 2.3.2, each frame consists of 256 symbols (d_n's) and 80 of them are real 2-PSK synchronization symbols. These symbols are passed through a transmit filter having an impulse response $g_{tr}(t)$ and transmitted signal x(t) is obtained.

$$x(t) = \sum_{n=-\infty}^{\infty} d_n g_{tr}(t - nT) \tag{5.4}$$

Then x(t) is passed through the channel and additive white Gaussian noise (AWGN), n(t), is added. For modelling the frequency shift, this signal is multiplied by $e^{j2\pi\nu t}$ where ν is the frequency offset.

$$r(t) = \left[\sum_{n=-\infty}^{\infty} d_n g_{tr}(t - nT) * h_C(t) + n(t) \right] e^{j2\pi\nu t} \tag{5.5}$$

Then, the received signal r(t) is passed through the receive filter filter ($g_r(t)$) and y(t) is obtained.

$$y(t) = \left[\sum_{n=-\infty}^{\infty} d_n g_{tr}(t - nT) * h_C(t) * g_r(t) \right] e^{j2\pi\nu t} + n(t) * g_r(t) e^{j2\pi\nu t} \tag{5.6}$$

This final form of y(t) can be simplified to the form in (5.7).

$$y(t) = \sum_{n=-\infty}^{\infty} d_n h(t - nT) e^{j2\pi\nu t} + m(t) \tag{5.7}$$

where $h(t) = g_{tr}(t)*h_C(t)*g_r(t)$ and $m(t)=n(t)*g_r(t)\, e^{j2\pi\nu t}$. Notice that m(t) is a zero mean Gaussian signal. Sampling y(t) at time instants kT/4, k=0,1,..., the signal samples are obtained.

$$y_k = \sum_{n=-\infty}^{\infty} d_n h(kT/4 - nT) e^{j2\pi\nu kT/4} + m_k \tag{5.8}$$

Then, y_k's are squared for finding the frequency offset

$$\begin{aligned} y_k^{\,2} &= \sum_{n=-\infty}^{\infty} d_n^{\,2} h^2(kT/4 - nT) e^{j2\pi\nu kT/2} + \\ &\sum_{i=-\infty}^{\infty} \sum_{m=-\infty, m\neq i}^{\infty} d_i d_m h(kT/4 - iT) h(kT/4 - mT) + z_k \end{aligned} \tag{5.9}$$

where z_k is a zero mean complex Gaussian noise denoting signalxnoise and noisexnoise components. Moreover, d_k's can be assumed to be independent, therefore expected value of the cross products are also zero. As indicated in Section 2.3.2, last 176 symbols in a frame composed of data and reference symbols are scrambled, therefore they can be assumed to be equally likely distributed over 8 complex symbols. Then, the squares of these symbols become equally likely distributed over 4 complex symbols. Hence, their mean is zero. However, first 80 symbols of a frame, i.e. the synchronization sequence, are composed of real symbols and when they are squared their values become all 1's. In conclusion, real synchronization sequence causes a large non-zero mean when squared. This DC value gives rise to a peak in Fourier transform of the squared signal which is at the origin if there is no frequency offset. According to the frequency offset term $e^{j2\pi\nu kT/2}$ in (5.9), the peak will shift to the $\pi\nu T$ radians. This peak can be detected with Fast Fourier Transform (FFT) of the squared signal. If N point FFT is used, then the frequency resolution becomes

$$\Delta f = \frac{2}{NT} \text{ Hz} \tag{5.10}$$

In the implementation, 2048-point FFT is used for the frequency offset correction and the frequency resolution is 2.34375 Hz from (5.10). Figure 5.4 shows the block diagram of the coarse frequency offset estimation.

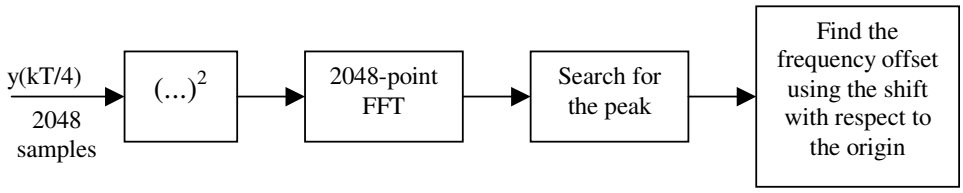

Figure 5.4: Block Diagram of the Coarse Frequency Offset Estimation

Since the frequency resolution in coarse frequency offset correction is 2.34375 Hz, a fine frequency offset correction is needed to minimize the offset error. The fine frequency offset correction method used in the implementation is based on channel estimation with LMS algorithm.

Samples of last two frames are stored in the data memory of the receiver as shown in Figure 5.5. Making use of the synchronization sequence channel coefficients are estimated for two frames by LMS algorithm. Then, frequency correction routine calculates the phases of the maximum channel taps and the difference gives the phase shift in 1024 samples due to the frequency offset.

Let the phases of the maximum channel taps be θ_n and θ_{n+1} for *n*th and (*n+1*)th frames, respectively. Considering the frequency offset given in (5.8), the phase difference of the maximum channel taps corresponds to

$$2\pi\nu_o kT/4 = \theta_{n+1} - \theta_n \qquad (5.11)$$

where k is 1024 (number of samples in a frame). Using 5.11, the frequency offset can be found.

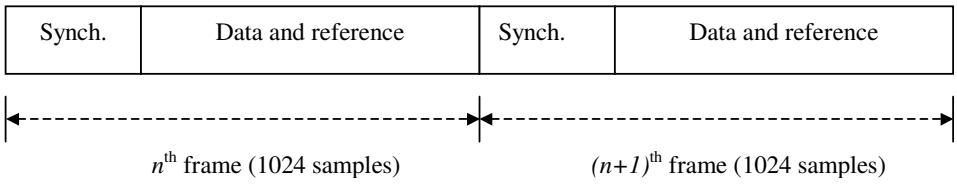

Figure 5.5: Frames stored in the data memory

The fine frequency offset correction process requires 1014 words of program memory including channel estimation routine. The process uses 667828 CPU cycles. Moreover, the coarse frequency offset correction uses 744 words of program memory and needs 538335 CPU cycles.

5.2.5 Fractionally Spaced LMS DFE with Multipass Technique

The equalizer employed in the implementation is an adaptive fractionally spaced decision feedback equalizer (FSDFE). The tap spacings of the feedforward part are at T/2 and least mean square (LMS) algorithm is utilized for the adaptation of the filter coefficients. The feedforward part spans 16 symbols and the feedback part spans 8 symbols.

Although LMS algorithm is well known for its simplicity and robustness, its convergence is slower than other adaptive algorithms. However, for fast-varying channels an adaptive algorithm possessing a fast convergence property is required. To improve the convergence of the LMS algorithm, the multipass technique is employed in the implementation.

The STANAG 4285 waveform includes synchronization and reference symbols inserted into each frame that can be used for training operation of the equalizer. Although the reference symbols are scrambled by a 8-PSK scrambling sequence, the scrambling symbols are all known by the receiver. In the multipass technique, the 80-symbol length synchronization sequence is repeatedly trained by the LMS algorithm until a pre-assigned number of iterations is reached. Denoting the iteration number index as ‘k’, the total number of iterations as ‘K’ and the symbol index as ‘n’, the multipass LMS algorithm employed in the implementation can be described as follows.

For n=1, 2, …, 80 and k=1, 2, …, K

$$\mathbf{W}_k(n) = \mathbf{W}_k(n-1) + \mu \mathbf{X}_k(n) e_k^*(n) \qquad (5.12)$$

$e_k(n)$→ error for the *n*th symbol

where $\mathbf{W}_k(n)$ and $\mathbf{X}_k(n)$ denote the weight and input vector, respectively. Then the initial weight vector for each iteration is obtained as

$$\mathbf{W}_k(1) = \mathbf{W}_{k-1}(80) \tag{5.13}$$

When the synchronization sequence part of the received samples is fed to DFE, the receiver checks the number of incorrect decisions and controls the flow of the program. The flowchart of the control process is given in Figure 5.6. When the number of decision errors is smaller than a pre-assigned threshold, then the 'error free pass' counter is incremented. However, if the number of errors exceeds the threshold, the counter is set to zero. After that, the counter value is compared to the pre-assigned maximum iteration number of the multipass operation. If the iteration number is reached, the equalization of synchronization part is ended. Otherwise, the operation is performed one more time. When equalization process fails, i.e. if many decision errors occur, 'bad frame' counter is incremented by one. The receiver program checks this counter and when a threshold value is reached it terminates.

The improvement in the performance of DFE with multipass technique is shown from the results given in Table 5.1. The table gives the number of errors detected for an uncoded test at 2400 bps with 192000 bits. The channel employed in the test has 2 independent equal-power Rayleigh paths with 0.5 Hz Doppler spread and separated by 1 ms. As shown in Table 5.1 significant improvement is achieved for high SNR values. As the noise power increases the improvement becomes negligible.

Table 5.1: Number of errors for an uncoded test with 192000 bits at 2400 bps

No. of iteration	**Number of errors within 192000 bits**			
	10 dB	15 dB	20dB	25 dB
1	8585	1751	644	569
2	8564	1643	444	355
4	8546	1635	257	214
6	8531	1616	138	100
8	8520	1603	131	42

After the multipass training, as shown in Figure 5.2, the whole frame is fed to the equalizer. The filter coefficients are also updated for reference symbols and so to follow rapid changes in the channel is enabled. The unscrambling process for data and reference symbols is also performed in the equalizer before they are fed to the decision device. This process includes only the multiplication of symbols with conjugates of the scrambling symbols.

The main problem encountered in the implementation of the LMS DFE with 16-bit length filter coefficients was the parameter drift. Except for the low signal to noise ratio (SNR) values, the filter coefficients attained arbitrary large values with slight increases in each equalization process. This unbounded behavior is especially observed for non-fading channels with high SNRs. In order to tackle this problem, double precision (2 words) is used for the filter coefficients. Obviously, the number of CPU cycles spent by the equalizer is increased because of the double precision arithmetic. While the multiplication of two 16-bit values requires only one instruction, the multiplication of a 32-bit word with a 16-bit one requires two 16-bit multiplications and an addition. Considering the filtering and coefficient update operations, the double precision has a significant effect on execution time.

The number of CPU cycles used for the equalization process depends on the maximum iteration number of the multipass technique. If the maximum iteration number is selected as 8, then the whole equalization process requires at most 3.92×10^6 CPU cycles.

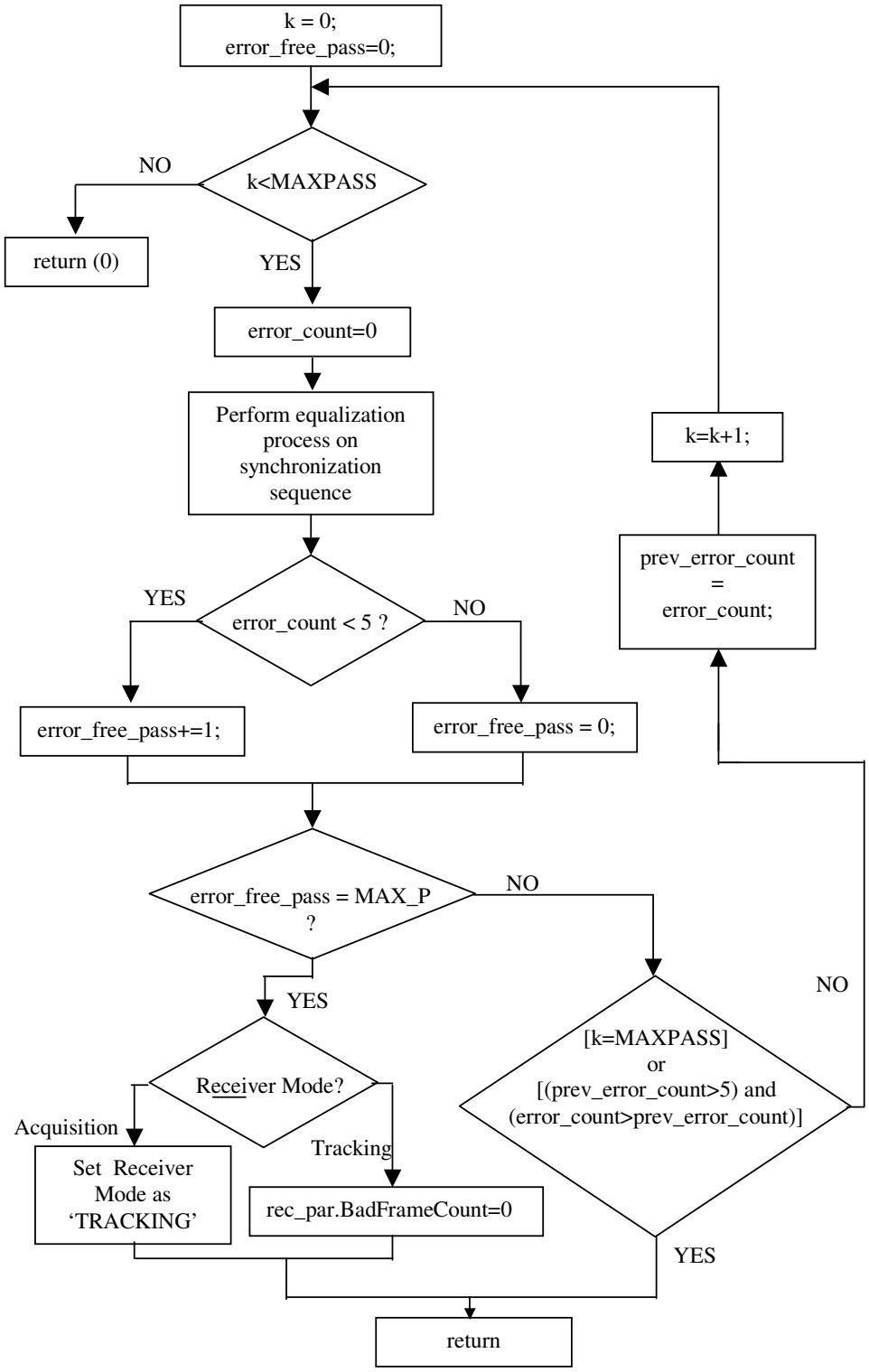

Figure 5.6: Flowchart of the multipass equalization control

5.2.6 Soft Bit Decision Computation

The Viterbi algorithm used in the receiver program operates on the soft decision values. Therefore, the data symbols should be extracted at the equalizer output and then the soft bit decisions should be computed.

Firstly, the nearest 64-PSK symbol number is calculated for each data symbol using a 64-PSK constellation diagram. Then, the soft bit decisions are obtained using the look-up tables generated for each data rate. Figure 5.7 shows the constellation diagram of 64-PSK signal.

For transmission at 2400 bps coded data rate 8-PSK modulation is used. Therefore, using 64-PSK constellation diagram every interval of two consecutive 8-PSK symbols is divided into 8 equal portions. Similarly, there are 16 portions for 1200 bps and 32 portions for other rates between two consecutive symbols. The look-up tables for soft bit decisions are generated using this fact. The soft bit decision values for different data rates are shown in Figure5.8.

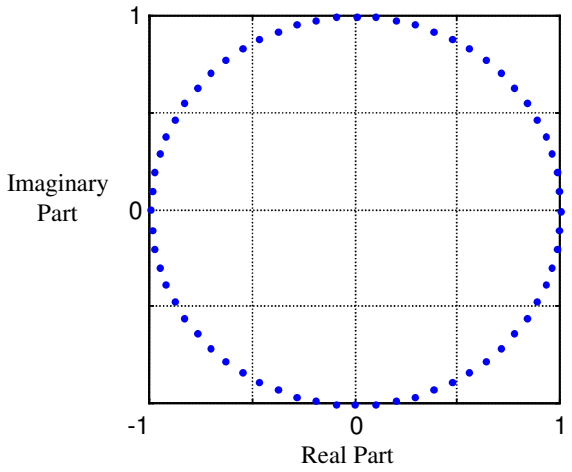

Figure 5.7: 64-PSK constellation diagram

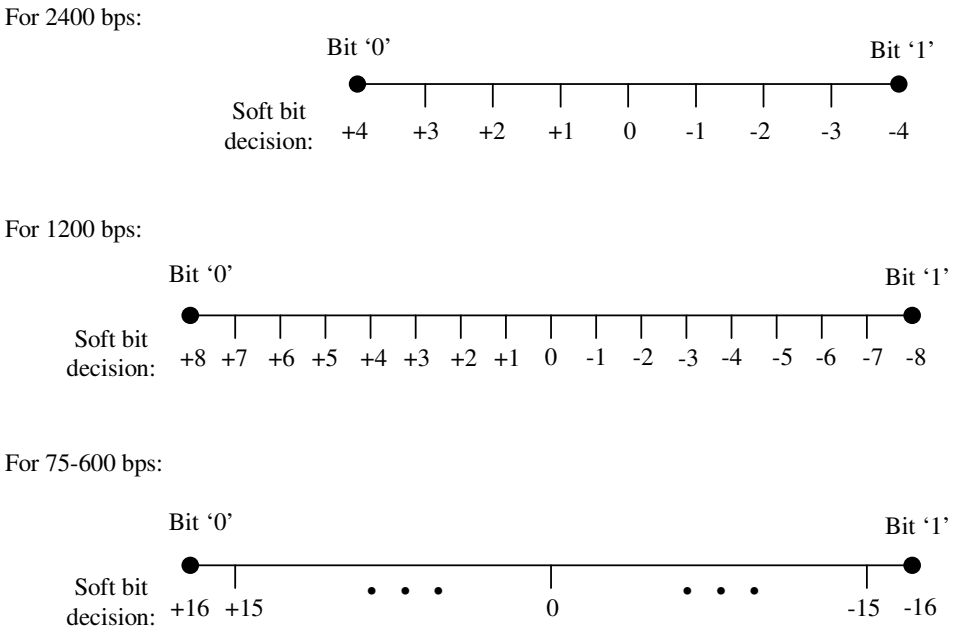

Figure 5.8: Soft bit decision values

As shown in Figure 5.8, the soft bit decision values are signed numbers and they are represented by 6 bits in the memory.

The program memory space required for soft bit decision computation process is 550 words. The number of CPU cycles used by the operation is different for different data rates due to PSK modulation employed. Table 5.2 gives the number of CPU cycles for each data rate. Maximum execution time is used for 2400 bps data rate due to 8-PSK modulation.

Table 5.2: The number of CPU cycles for soft bit decision computation

Coded Data Rate	**Cycles**
2400 bps	171182
1200 bps	168667
600-75 bps	166525

5.2.7 Deinterleaver

The next operation after soft decision computation is deinterleaving. The programming technique of the deinterleaver is very similar to one employed for the interleaver. The delay increment values for successive rows are given in Table 2.6.

While the interleaver gets hard bits as input, the soft decision inputs of the deinterleaver are 6 bits in length. Therefore, the required memory space is much grater for long deinterleaver.

In long interleaver, every word of the memory can be used by two soft bit decisions. Since there are totally 23808 soft bit decisions to be stored in the long deinterleaver for 2400 bps data rate, 11904 words of data memory should be reserved for this process. Moreover, 115 words of program memory should be allocated.

In short deinterleaver, one word is reserved for each soft bit decision. Considering 2400 bps data rate, i.e. the largest amount of memory usage, the deinterleaver requires only 1984 words of data memory. Furthermore, the number of words allocated for program memory is 175 words.

In order to decrease processing time, circular addressing mode technique is used for both long and short deinterleaver. The number of cycles used by deinterleavers for different data rates is given in Table 5.3.

Table 5.3: The number of cycles for deinterleaving process

	Cycle Counts	
Data Rates	Long Deinterleaver	Short Deinterleaver
2400 bps	14355	18875
1200 bps	7240	9504
600 – 75 bps	3679	4816

5.2.8 Viterbi Decoding

Considering the improvement in the execution speed and memory requirement of the receiver program, the programming techniques utilized for soft decision Viterbi decoder play a critical role.

The convolutional encoder in the transmitter has a constraint length of 7. Therefore, the number of possible delay states in the trellis diagram is $2^{7-1} = 64$. While the delay states represent the state of the encoder, the path states represent the output bit pairs from the encoder. Since the encoder has a rate of 1/2, there are only two paths from each delay state to another.

Viterbi algorithm consists of metric update and traceback operations. In the metric update, accumulated distances are calculated for two possible paths pointing to the delay state. The path with the minimum accumulated distance is selected as the survivor and stored for traceback operation. The other path is discarded. This computation is repeated for every delay state. In traceback operation, after five times the constraint length 7, the state with the minimum accumulated metric is selected and tracing the stored paths backward a hard output bit is obtained.

In Viterbi algorithm, most of the processing time is spent on the metric update, since all of the 64 states must be updated for every symbol interval. Throughout the development of the Viterbi function two methods are implemented for metric update operation and they exhibit considerable difference in processing time.

First method accumulates metrics for two paths one by one. Then compares them and selects the minimum one. This process is repeated for every delay state, increasing calculation time too much. Maximum number of cycles used for this method is 1962871.

Second method utilizes symmetry properties of the trellis diagram for the encoder used. Moreover, it uses specific ’C54x instructions in order to minimize metric update calculation time [10].

The path states associated with the two paths pointing to a delay state are complementary for the convolutional encoder used in STANAG 4285. A butterfly structure of the trellis diagram is given in Figure 5.9, showing the symmetry

property of path states. As shown in the figure, if one of the path state is (00), then the other path state is (11) for the paths pointing to the same delay state. For other paths not shown in the figure, the complementary path state pair may be (10) and (01).

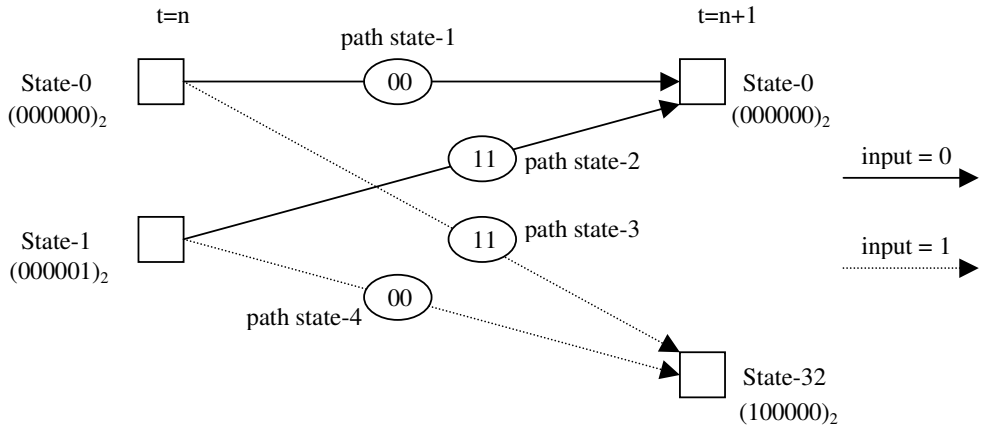

Figure 5.9: Butterfly structure of the trellis diagram

Another simplification can be made in local distance calculation. Denoting the input soft bit decision pair as SD_1 and SD_2, the local distance for path j is calculated using the Euclidean distance:

$$\text{distance}\ (j) = \sum_{n=1}^{2} [SD_n - T_n(j)]^2 \tag{5.14}$$

where $T_1(j)$ and $T_2(j)$ represent the expected inputs for path j, and they are coded as signed antipodal values, meaning that 0 corresponds to +1 and 1 corresponds to −1. Expanding (5.14) yields:

$$\text{distance}\ (j) = \sum_{n=1}^{2} \left[SD_n^{\,2} + T_n^{\,2}(j) - 2SD_n T_n(j)\right] \tag{5.15}$$

Since the distances are to be compared, the terms $\sum_{n=1}^{2} SD_n^{\,2}$ and $\sum_{n=1}^{2} T_n^{\,2}(j)$ can be eliminated because they are same for all paths. Thus, the equation reduces to:

$$\text{distance}\ (j) = -2\sum_{n=1}^{2} SD_n T_n(j) \tag{5.16}$$

In order to find minimum distance, discarding the leading –2 scalar, the maximums are searched for in the metric update. As a result, the local distance for path j is calculated using the equation:

$$\text{distance}\ (j) = \sum_{n=1}^{2} SD_n T_n \tag{5.17}$$

Since two path states are complementary entering the same delay state, the local distances for them will differ only in sign, i.e. if one of the paths has a local distance of 'x', the other will have a local distance of '-x'. Therefore, only one calculation for local distance is adequate in a butterfly structure.

In order to benefit from these results, dual add-subtract and compare-select-store instructions of 'C54x are utilized. Once the local distance is calculated, only four cycles are spent per butterfly to calculate accumulated distances and select the maximum ones for each state. As a result, a significant improvement in processing time is obtained for metric update operation. Table 5.4 gives the number of cycles used by the Viterbi decoder for different data rates. Moreover, the process requires 232 words of program memory.

Table 5.4: The number of cycles for Viterbi decoder

Coded Data Rate	Cycles
2400 bps	495800
1200 bps	226399
600-75 bps	91545

5.2.9 Message Protocol

In the message protocol of the receiver the following operations are performed:

- The receiver program searches for the SOM sequence within the decoded bits. Only after the SOM sequence is detected, the bits are written to the output file and 'start of message' flag is set to 1.
- After that, the program searches for the EOM sequence and reception terminates when it is detected.
- If the receiver continuously fails to detect the 80-symbol synchronization sequence for a time interval equal to the interleaver delay, the reception is also terminated. The number of failures in the detection of the synchronization sequence is determined by the 'bad frame' counter used in the control of the equalizer.

5.3 Resource Requirements of the Receiver Program

The receiver program implemented on 'C5410 DSP processor uses 35.18 Kwords of data and 6.98 Kwords of program memory. Since every soft decision value is represented by 6 bits, the most of the data memory space is occupied by the deinterleaver. Moreover, the receiver program uses 4 Kwords of data memory space in order to store real and imaginary parts of two frames.

The execution time of the receiver program varies with data rates and the thresholds used for multipass DFE. Table 5.5 gives the maximum numbers of CPU cycles used by the receiver program for different data rates when the maximum number of iterations for DFE is set to eight. As shown in the table, the CPU cycles required by the 2400 bps coded data rate is larger than others because of the number of bits to be decoded.

The number of CPU cycles used by the receiver program is approximately 6.42×10^6 per frame in the worst case. Since there are 10.66×10^6 cycles available in order to process a frame, the program uses 60.58% of this amount.

Table 5.5:Cycle counts for receiver program

Data Rate	Cycles
2400 bps	6421149
1200 bps	6117140
600 bps	5249448
300 bps	5175895
150 bps	5170166
75 bps	5167301

CHAPTER 6

SIMULATIONS

6.1 Introduction

In order to evaluate the performance of implemented STANAG 4285 HF modem, the Watterson's channel model is employed as the channel simulator that is also recommended in [1]. Since it is impossible to model every HF channel condition, the simulator uses three standard channels. Two of them are multipath channels with two independent Rayleigh fading paths with equal average powers and the other is single-path channel with a single Rayleigh fading path. Two multipath channels differ in time delay and Doppler spread values as shown in Table 6.1. These values are recommended in [21].

Table 6.1: Properties of multipath channels employed in the tests

Channel	Doppler Spread	Time Delay Spread
Poor	1 Hz	2 ms
Moderate	0.5 Hz	1 ms

In order to generate a Rayleigh fading path two independent Gaussian noise sources are required as shown in Figure 6.1. The white Gaussian noise signals, $n_1(t)$ and $n_2(t)$, are passed through a low-pass filter (h(t))

which determines the Doppler spread. The filter has a bandwidth of approximately one-half the Doppler spread [1]. Equation 6.1 shows the relation between the Doppler spread and the frequency response of the low-pass filter. Moreover, as indicated in [1], the shape of the low-pass filter is not critical but it should be at least a two-pole filter.

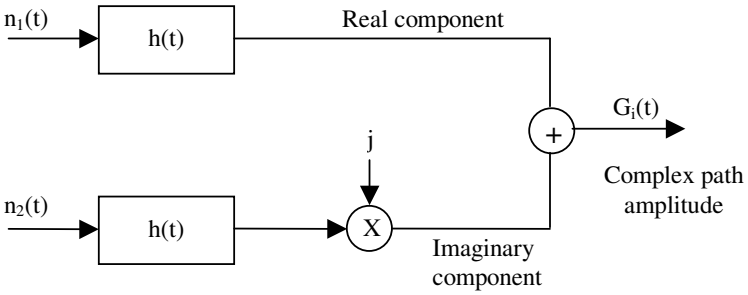

Figure 6.1:Generation of a Rayleigh Fading Path

$$\text{(Doppler Spread)}\ D_S = 2\sqrt{\frac{\int_0^{\infty} f^2 |H(f)|^2 df}{\int_0^{\infty} |H(f)|^2 df}} \qquad (6.1)$$

The tests are performed in the presence of different amounts of additive white Gaussian noise (AWGN). The simulation results are given in the following section.

6.2 Simulation Results

The following tests are implemented in the baseband and 12000 frames are used for each data rate. Since every frame is 106.6 ms in length, the test time for each bit-error measurement becomes 21.32 min. However, in order to achieve more exact values the tests should be run for extended lengths of time as recommended in [1]. For example, in order to achieve the actual bit error rate (BER) the test for moderate channel with coded and long-interleaved data should be run for 15.6 hrs [1].

There are three sets of input data for the channel used in the tests. First one is coded data with long interleaving, the second one is coded data with short interleaving, and the last one is uncoded data without interleaving. The tests are performed in the presence of AWGN and the signal to noise ratio (SNR) values are calculated as

$$\mathrm{SNR} = 10\log_{10}\left(\frac{\mathrm{average-signal-power}}{\mathrm{average-noise-power}}\right)_{\text{in 3 kHz bandwidth}} \tag{6.1}$$

Figures from 6.2 to 6.9 give the bit error rates (BER) for different channel conditions and data types. Moreover, the predicted performances of the HF modem employing error correction coding and long interleaving are tabulated in Appendix B. These performance values are suggested by [1].

The BER performance of the HF modem for poor channel with coded and long interleaved data is given in Figure 6.2. Compared to other rates, transmission at 2400 bps is less robust since 8-PSK symbols have shorter Euclidean distance than 4- and 2-PSK symbols. Moreover, puncturing at 2400 bps is another reason for degraded performance. Compared to the predicted BER values in Table B.1, the BER performance of the modem is better except for at 600 bps with 10 dB SNR and at 1200 bps with 15 dB SNR.

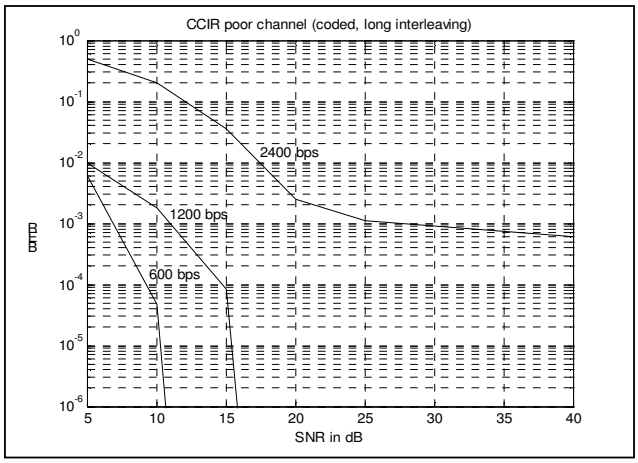

Figure 6.2: BER performance of the modem for poor channel (coded, long interleaving)

Figure 6.3 shows the performance of the HF modem for moderate channel with coded and long interleaved data. Comparing to predicted values given in Table B.2, the performance of the HF modem is worse for high SNR values. Considering the data rates, performance of 600 bps is the best as expected.

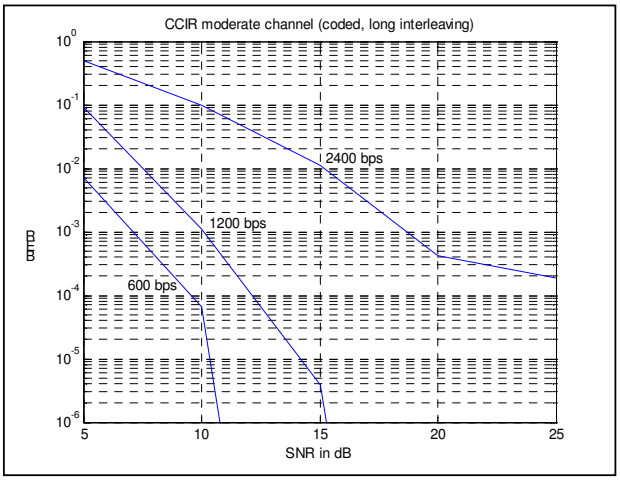

Figure 6.3: BER performance of the modem for moderate channel (coded, long interleaving)

The BER performance of the HF modem for a single Rayleigh fading path with a Doppler spread of 1 Hz is given in Figure 6.4 for coded and long interleaved data. For low SNR values the performance of the HF modem is better than the predicted one given in Table B.3. The BER performance of the modem at 600 and 1200 bps data rate for 15 dB SNR value is worse compared to predicted performance. Moreover, for high SNR values degradation in performance is observed at 2400 bps.

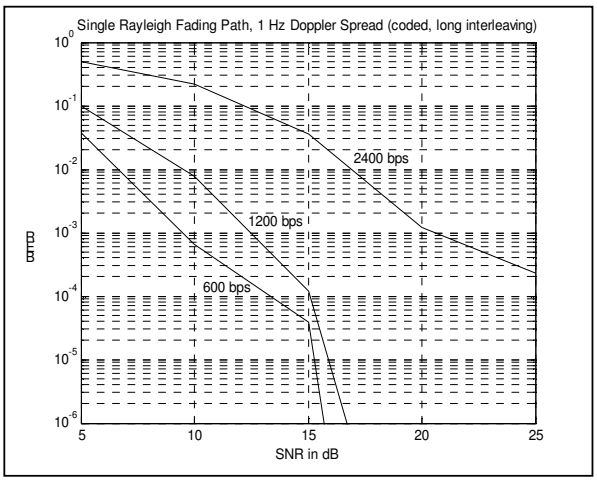

Figure 6.4: BER performance of the modem for single Rayleigh fading path (coded, long interleaving)

Figures from 6.5 to 6.7 give the BER performances of the HF modem using coded data with short interleaver. As shown in the figures, the use of short interleaving technique degrades the performance of the modem compared to long interleaving technique.

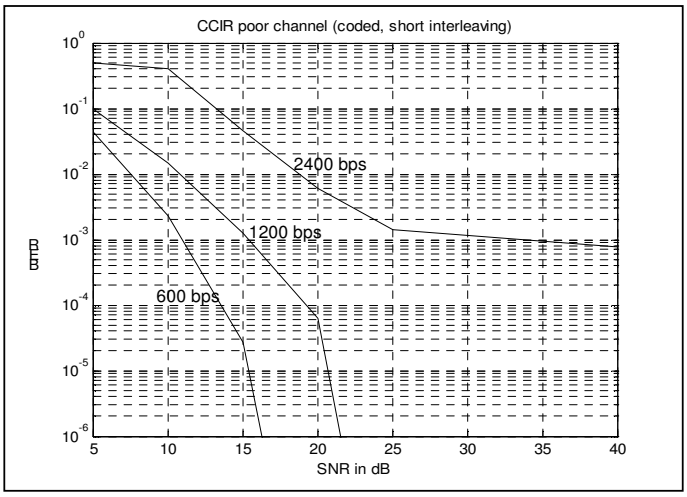

Figure 6.5: BER performance of the modem for poor channel (coded, short interleaving)

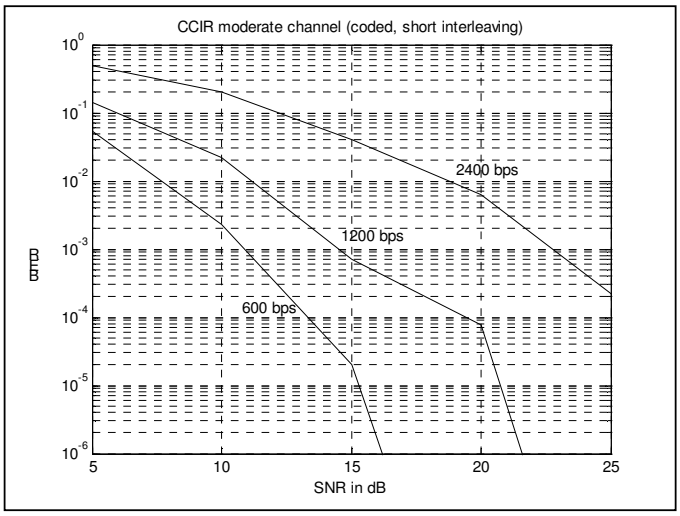

Figure 6.6: BER performance of the modem for moderate channel (coded, short interleaving)

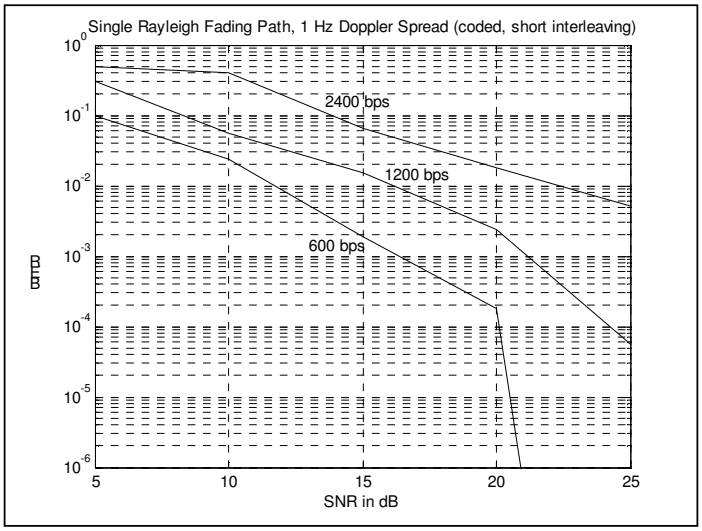

Figure 6.7: BER performance of the modem for single Rayleigh fading path (coded, short interleaving)

Figure 6.8 and 6.9 show the performance of the HF modem, which does not use error correction coding or interleaving technique. Therefore, these results may be seen as the performance of the DFE.

Figure 6.8 gives the performance of the modem for AWGN channel in absence of multipath and fading distortion. While BER values are closer for low SNR values, the difference becomes apparent for higher SNR values.

The BER performance of the modem for moderate channel with uncoded data is given in Figure 6.9.

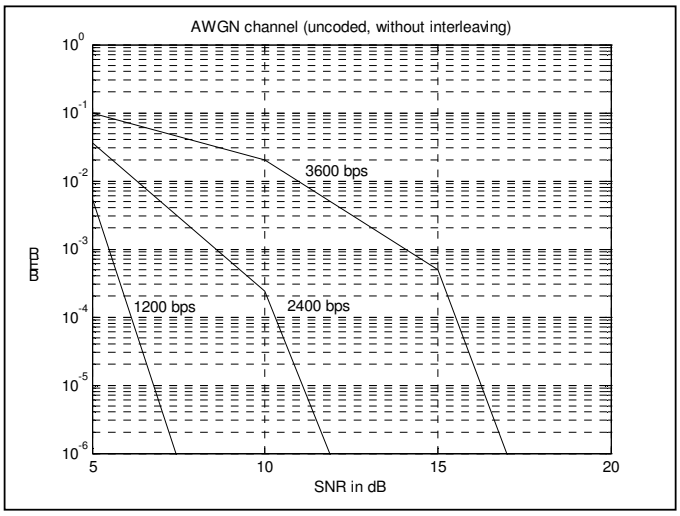

Figure 6.8: BER performance of the modem for AWGN channel (uncoded, without interleaving)

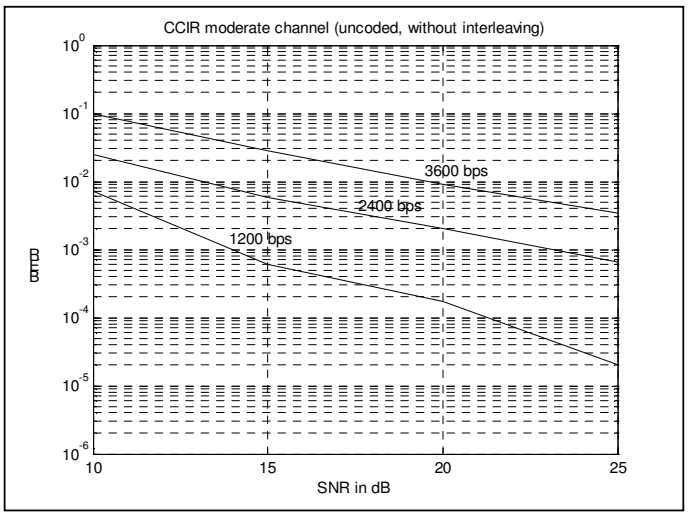

Figure 6.9: BER performance of the modem for moderate channel (uncoded, without interleaving)

Considering all results, the 8-PSK transmission is less robust since 8-PSK symbols have a shorter Euclidean distance compared to 2- or 4-PSK. The best performances are observed for 2-PSK transmission.

The simulations show that the performance of the coded transmission with long interleaving outperforms the coded transmission with short interleaving. However, the long interleaving has a disadvantage of long transmission delay.

The test results obtained for coded data and long interleaving technique are close to predicted performances given in Appendix A. However, as indicated before, the BER values shown in the figures cannot be the actual bit error rates of the HF modem. The test times to obtain confident BER values are much longer than the one used in this implementation.

CHAPTER 7

CONCLUSIONS

In this research, the STANAG 4285 HF modem software is implemented on TMS320C54x fixed point digital signal processor (DSP). The routines are tested for proper operation by using high level programs. Using code optimization techniques the software is ported on fixed point DSP. Considering the resource requirements of the HF modem software, the program is evaluated to be efficient enough for real time operation on TMS320C54x.

In order to compensate the severe inter-symbol-interference (ISI) effect of the HF channel, a fractionally spaced decision feedback equalizer is employed in the receiver. For the adaptation of equalizer filter coefficients LMS algorithm is utilized. The convergence speed of the LMS algorithm is very slow for fast-fading channels where the input correlation matrices may have wide eigenvalue spread values. Therefore, a multipass technique is used in order to improve the convergence of the LMS algorithm. The results show that the performance of the receiver employing multipass technique outperforms the one employing singlepass technique.

A significant problem is encountered when using fixed point arithmetic for LMS algorithm. When equalizer filter coefficients are represented by 16 bits, a slight increase in magnitude is observed for coefficients in each iteration resulting in divergence. This effect is especially encountered for high signal-to-noise ratio (SNR) values. In order to prevent divergence, the representation of filter coefficients is changed from 16-bit to 32-bit increasing the complexity.

The BER performance of the HF modem is evaluated by an HF channel simulator employing the Watterson's channel model. The results have shown that the performance of the modem is close to predicted one.

Although the multipass technique improves the convergence property of the LMS algorithm a whitening filter prior to the equalizer will improve the performance. Therefore, the results obtained in this implementation can be extended by using a whitening filter.

REFERENCES

[1] STANAG 4285, "Characteristics of 1200/2400/3600 Bit per Second Single Tone Modulators/Demodulators for HF Radio Links", NATO, Brussels 1989

[2] MIL-STD-118-110A, "Military Standard, Interoperability and Performance Standards for Data Modems", Department of Defense, USA 1991

[3] Watterson C.C, Juroshek J.R and Bensema W.D, "Experimental Confirmation of an HF Channel Model", IEEE Transactions on Communications Technology, Vol COM-18, No 6, Dec 1970.

[4] J. G. Proakis, "Digital Communications", New York, McGraw-Hill, 1995

[5] Nicholas Maslin, "HF Communications: A Systems Approach", New York, Plenum, 1987

[6] Bernard Goldberg, "300 kHz-30 MHz MF/HF", IEEE Transactions on Communication Technology, December 1966.

[7] TMS320C54x DSP Reference Set, "Volume 1: CPU and Peripherals", TI, April 1999

[8] Simon Haykin, "Adaptive Filter Theory", Prentice-Hall International, Third Edition,1996

[9] C. Caraiscos, B. Liu, "A Roundoff Error Analysis of the LMS Adaptive Algorithm", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. Assp-32, No. 1, February 1984

[10] H. Hendrix, "Viterbi Decoding Techniques in the TMS320C54x Family", TI, June 1996

[11] Simon Haykin, "Communication Systems", 3rd Edition, John Wiley & Sons, 1994

[12] M. H. Hayes, "Statistical Digital Signal Processing and Modeling", John Wiley & Sons, 1996

[13] R.D. Gitlin, J.E. Mazo, M.G. Taylor, “On the design of gradient algorithms for digitally implemented adaptive filters”, IEEE Transactions on Circuit Theory, vol. CT-20, pp. 125-136,Mar. 1973

[14] W.A. Sethares, D.A. Lawrence, C.R. Johnson, R.R. Bitmead, ”Parameter drift in LMS Adaptive Filters”, IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. assp-34, No. 4, August 1986

[15] M. Austin, “Decision feedback equalization for digital communication over dispersive channels”, MIT Research Laboratory of Electronics Technical Report 461, August 1967

[16] D.L. Duttweiler, J.E. Mazo, and D.G. Messerschmitt, “An upper bound to the error probability in decision feedback equalization”, IEEE Transactions on Information Theory, IT-20, pp.490-497, July 1974

[17] A. Cantoni, P. Butler, “Stability of decision feedback inverses”, IEEE Transactions on Communications, Vol. COM-24, pp.970-977, September 1976

[18] R.A. Kenedy, B.D. Anderson, R.R. Bitmead, “Tight bounds on the error probabilities of decision feedback equalizers”, IEEE Transactions on Communications, Vol. COM-35, pp. 1022-1029, October1976

[19] R.D. Gitlin, S.B. Weinstein, “Fractionally spaced equalization: an improved digital transversal equalizer”, The Bell System Technical Journal, Vol. 60, No. 2, February 1981

[20] P. Monsen, “Fading channel communications”, IEEE Communications Magazine, Jan. 1980

[21] CCIR Recommendation 520, “Use of High Frequency Ionospheric Channel Simulators”

[22] J. Pennington, “Techniques for medium-speed data transmission over HF channels”, IEE Proceedings, Vol. 136, Pt. 1, No. 1, February 1989

APPENDIX A

TIGER 5410/PC DEVELOPMENT BOARD FEATURES

A.1 TIGER 5410/PC Hardware

Digital Signal Processor:

Quantity	1
Type	TMS320VC5410 DSP
Processing Power	100 MIPS

Memory:

'C5410 SRAM (on-chip)	64 kwords
'C5410 ROM (on-chip)	16 kwords
SRAM	256 kbytes
EPROM	Up to 256 kbytes

Digital I/O Interfaces:

RS-232 Interface	16550-compatible UART
Serial Interface	Three Multi-channel Buffered Serial Ports (McBSPs)
Host Port Interface	HPI Header connects to the 'C5410 HPI

Analog I/O Interface:

I/O Interface	16-bit, dual channel, 50 kHz Connector: Line level, mini jacks Line/Microphone is jumper selectable
Telephone Interface	Analog RJ-11, FCC approved
DACs	Two,8-bit

Host Interface:

16-bit PC Host	Standard ISA bus interface with bi-directional interrupt support

Debug/Emulation:

Emulation Hardware	On-board emulation (Test Bus Controller) hardware
External Emulator	JTAG connector for external emulators

Power requirements:

Current	3.7 A @ 3.3 Vdc 0.7 A @ 5 Vdc

A.2 TIGER 5410/PC Software

Development Environment:

- Code Composer Studio environment.
- Libraries: Complete software support libraries for all on-board devices, and for communication with the host system, including loader. Standard C library.

Language Support:

- Texas Instruments C compiler, assembler, linker, and tools

A.3 Features of the TMS320VC5410

- 10-ns single-cycle fixed-point instruction execution time (100 MIPS performance)
- 40-bit Arithmetic Logic Unit (ALU)
- 64 kwords of on-chip SRAM
- 16 kwords of on-chip ROM
- 2.5V core power supply
- Three on-chip Multi-Channel Buffered Serial Ports (McBSPs)
- 8-bit enhanced host port interface (HPI8)
- Six DMA channels
- Standard 16-bit timer

APPENDIX B

PREDICTED BER PERFORMANCES

Table B.1: Predicted BER for CCIR Poor Conditions versus SNR
(for coded data with long interleaving)

SNR in 3 kHz bandwidth	**Coded Data Rate (Bits/Second)**		
	2400	1200	600
5 dB	-	-	6.63E-3
10 dB	-	1.33E-3	0
15 dB	2.13E-1	0	0
20 dB	5.65E-2	0	0
25 dB	2.53E-2	0	0
40 dB	1.14E-2	0	0

Table B.2: Predicted BER for CCIR Moderate Conditions versus SNR
(for coded data with long interleaving)

SNR in 3 kHz bandwidth	**Coded Data Rate (Bits/Second)**		
	2400	1200	600
5 dB	-	2.15E-1	6.67E-3
10 dB	-	1.78E-4	0
15 dB	1.49E-2	0	0
20 dB	9.9E-5	0	0
25 dB	0	0	0

Table B.3: Predicted BER for Single Rayleigh Fading Path versus SNR
(1 Hz Doppler Spread)

(for coded data with long interleaving)

SNR in 3 kHz bandwidth	**Coded Data Rate (Bits/Second)**		
	2400	1200	600
5 dB	-	-	2.86E-1
10 dB	0.5	3.18E-2	3.4E-4
15 dB	1.25E-1	0	0
20 dB	4.46E-4	0	0
25 dB	0	0	0