Physical Layer Design for a Spread Spectrum Wireless LAN

by

Guoliang Li

Thesis submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

IN

ELECTRICAL ENGINEERING

APPROVED:

Dennis G. Sweeney, Chairman

Scott F. Midkiff

Brian D. Woerner

September 12, 1996
Blacksburg, Virginia

Keywords: Wireless LAN, Spread Spectrum, Radio transceiver, PN code

Physical Layer Design for a Spread Spectrum Wireless LAN

by

Guoliang Li

Committee Chairman: Dennis G. Sweeney

Bradley Department of Electrical Engineering

(ABSTRACT)

A wireless local area network (LAN) system is proposed to provide mobility for existing data communication services. This thesis presents a physical layer design for a direct sequence spread spectrum ISM band radio LAN system. This radio system employs spread spectrum communication technology and a differential binary phase shift keying/quadrature phase shift keying (BPSK/QPSK) non-coherent receiver to overcome the adverse indoor wireless environment. Moreover, a variable data rate transmission technique is used to dynamically configure the spread spectrum system according to channel performance. This physical layer incorporates the Zilog Z2000 Evaluation Board performing direct sequence spread spectrum processing, a Grayson 900 MHz radio receiver and a transmitter module which was designed and built at Virginia Tech. The transmitted spectrum occupies a 4 MHz bandwidth in the 900 MHz ISM band and this system supports a data rate of up to 363 Kbits/sec. The spread spectrum system design along with detailed descriptions of hardware and control software development are presented.

ACKNOWLEDGEMENTS

I would like to express my deep appreciation to my advisor, Professor Dennis G. Sweeney for his introducing me to spread spectrum communications and hardware design. Without his guidance, suggestions and encouragement, I could not have completed this work. I am also very grateful for his patience in correcting and commenting on my thesis. My sincere thanks also go to my committee members: Professor Scott F. Midkiff and Professor Brian D. Woerner for their ideas, suggestions on my work and their comments on this thesis.

I want to thank Barry Mullins for his terminal program support and also Todd Fleming for their advice on the software development.

It is also a pleasure to acknowledge Gunadi Gunawan who sponsored me for further study in United States and has always encouraged me during these past two years. I would also like to thank MingZhi Yao for her spiritual support from the beginning of this research work.

Finally, I wish to thank my parents and my brothers for their understanding, encouragement and love during my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Today, communication has increasing influence on our daily life. Wireless data communication services allow people to access the data network without a physical connection.

IEEE 802.11 is being proposed to provide wireless connection for local area network (LAN) using spread spectrum techniques in the Industrial, Scientific, Medical (ISM) bands and baseband infrared techniques. In this project, we want to design and build a real wireless LAN system operating at 902-928 MHz of the ISM bands. The physical layer of this network has been designed to perform direct sequence spread spectrum processing and RF transceiving. A variable data transmission strategy is employed in this system so that we are able to configure the physical layer dynamically based on the medium access control (MAC) layer's control frame.

In this thesis, the physical layer subsystem was designed and hardware and software supports were built based on the above requirements. We employ the Zilog Z2000 Evaluation Board to perform spread spectrum processing and a Grayson 900 MHz receiver for radio signal receiving. The radio transmitter was designed and built in the Center for Wireless Telecommunications, Virginia Tech.

In the next chapter, we want to introduce spread spectrum operation and the concept of wireless LAN. Chapter 3 describes the spread spectrum system design. In Chapter 4, we

investigate the operation of the Z2000 Evaluation Board. Mathematical representations of each functional block in the Z2000 spread spectrum burst processor are also provided. A simulation of the Z2000 was conducted and the results are shown. In Chapter 5, we present the radio transmitter design. Chapter 6 briefly describes the software interface between the MAC layer and the physical layer, the control software for the spread spectrum processing and the radio transceiver.

Chapter 2

Spread Spectrum and Wireless LANs

2.1 Introduction

Spread spectrum communication technology has been used extensively in a wide variety of military applications since the 1940's because of its inherent military advantages: immunity to intentional jamming interference, the capability of hiding a spread signal from eavesdroppers, security and high-resolution ranging [Sim 85]. In the last decade, the wireless communication market experienced significant growth, especially in cellular telephony, personal communications systems (PCS) and wireless local area networks (WLANs). Increasingly, researchers have been interested in the issue of spread spectrum because of its potential technical advantages including higher spectrum utilization and resistance to multipath. Currently, spread spectrum techniques have been used in satellite communications, position location systems and telemetry systems. These techniques are also emerging in personal communications systems and wireless information network applications.

In 1985, the Federal Communications Commission (FCC) released three unregulated ISM bands to encourage the development of spread spectrum technology. The frequencies of

these unlicensed spectrum are 902 to 928 MHz, 2400 to 2483.5 MHz and 5725 to 5850 MHz. Part 15.247 of FCC rules governing these frequencies bands requires that the transmitters use spread spectrum modulation with a transmitted power less than 1 W and the processing gain of the spread spectrum system be larger than 10 (we will discuss the processing gain in the following sections). The drawback of this unlicensed operation is that one must be able to operate with interference from other users in the same bands.

A direct sequence spread spectrum wireless LAN (WLAN) system is proposed in this paper. This system is designed to be compatible with IEEE 802.11, which is a standard being developed using direct-sequence and frequency-hopping techniques in the 2.4 GHz ISM band and baseband infrared (IR) technique [Dra 94]. The main difference in this work from IEEE 802.11 is that we use the 902-928 ISM band for radio frequency (RF) transmission. It is intended to provide signal coverage up to 500-800 feet. One of the benefits of using spread spectrum is that the transmission capability of the wireless channel can be maximized by using a variable data rate transmission technique. This technique reduces the data rate when the wireless channel is degraded to ensure reliable data reception, and increases the data rate when the channel is clear to fully use the channel capacity. By dynamically changing the data rate according to the channel performance, this technique helps to maintain the link during the adverse conditions. It reduces the necessity of link re-initialization by reducing dropout and it also maximizes the overall system throughput. This technique will be described further in Chapter 3.

2.2 Principles of Spread Spectrum

2.2.1 Spread Spectrum and the PN Code

Generally, a spread spectrum signal is generated by modulating a signal so that the resultant transmitted signal has a bandwidth much larger than the original signal bandwidth. This is shown in Figure 2.1.

Two requirements need to be met in a spread spectrum system. First, the modulated

signal occupies a much wider spectrum than the original message signal. Second, the modulated signal appears to be random and uncorrelated to the original data although generated from the original data [Buc 92]. A key parameter of the spread spectrum system is the bandwidth expansion factor or the processing gain. As we see in Figure 2.1, the transmitted power is spread over a bandwidth N times wider than the information symbol rate. Thus, without changing the signal power, the power spectral density (PSD) of the signal would be N times lower than it would be in non-spread transmission and the signal is less likely to be detected. Obviously, this property makes spread spectrum attractive for military applications in which the signal must be hidden from eavesdroppers.

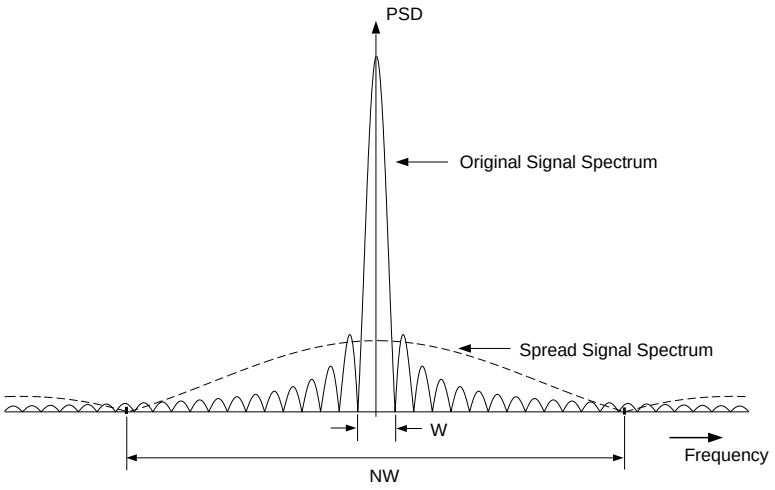

Figure 2.1: Power Spectral Density (PSD) of Spread Spectrum

The processing gain N is defined by:

$$processing\ \ gain = N = \frac{transmitted\ \ signal\ \ bandwidth}{information\ \ bandwidth} \tag{2.1}$$

which specifies to what extent the original signal is spread. Practically, N is an integer with a typical value of $10\ log_{10}N = 10$ to $30\ dB$.

The pattern used in the spread spectrum system to spread the information spectrum

is called the spreading pattern or spreading code. This spreading code is generated in a deterministic way but should appear to be random or noise-like. Hence, it is usually called the pseudo-random or pseudo-noise (PN) code. In order for the spread spectrum receiver to reliably despread the signal, a common requirement of the PN code is good autocorrelation performance. There may be additional requirements in some applications. For example, in a Code Division Multiple Access (CDMA) system, a small cross-correlation value with multiple PN codes is desired and in our wireless LAN system, a good partial-correlation is also needed. We will address the issue of the PN code in more detail in the following sections.

Spread spectrum systems are classified by the ways that the original data is modulated by the PN code. The most commonly employed spread spectrum techniques are the following:

Direct Sequence Spread Spectrum (DSSS)

Frequency Hopping Spread Spectrum (FHSS)

Hybrid Direct-Sequence and Frequency-Hopping Spread Spectrum

Time Hopping Spread Spectrum

Chirp Spread Spectrum

In wireless local area network applications, direct-sequence and frequency-hopping methods are used extensively and are discussed in this chapter.

2.2.2 Two Widely Used Spread Spectrum Systems

Direct Sequence Spread Spectrum

Direct sequence spread spectrum (DSSS) involves modulo-2 addition of the information signal to a sequence of higher frequency. This process is shown in Figure 2.2, where $d(t)$ is the information signal and $c(t)$ is the PN chip sequence signal with a higher frequency. $T(t)$ is the direct sequence spread spectrum signal by adding, modulo-2, $c(t)$ and $d(t)$. The last graph of Figure 2.2 shows that the spread spectrum signal is despread back to

the information signal $d(t)$ by adding, modulo-2, $T(t)$ with a replica of PN chip sequence signal $c(t)$. A typical direct sequence spread spectrum transmitter and receiver is shown in Figure 2.3.

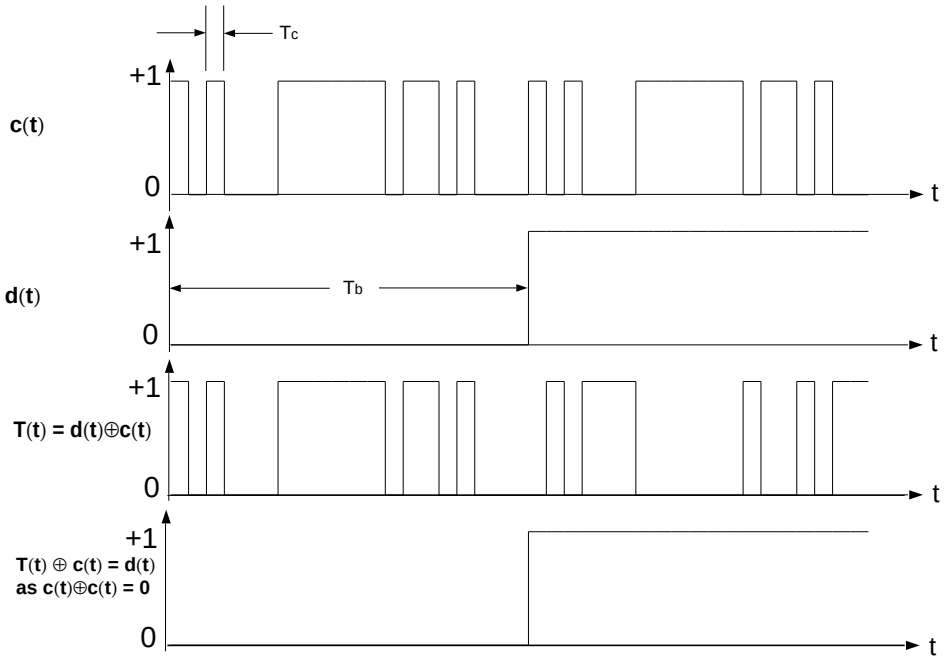

Figure 2.2: Direct Sequence Spread Spectrum: Spreading and Despreading

The narrow pulses with duration T_c are referred to as chips. The bandwidth expansion factor or processing gain is determined by $N = T_b \ / \ T_c$, where T_b is the information symbol duration. The autocorrelation function of $c(t)$ is given by $R(t)$:

$$R(t) = \int_{-\infty}^{+\infty} c(\tau) \cdot c(t-\tau) d\tau \tag{2.2}$$

$S(\omega)$, the power spectral density (PSD) of $c(t)$, is the Fourier transform of $R(t)$:

$$S(\omega) \quad = \quad \mathcal{F}[R(t)]$$

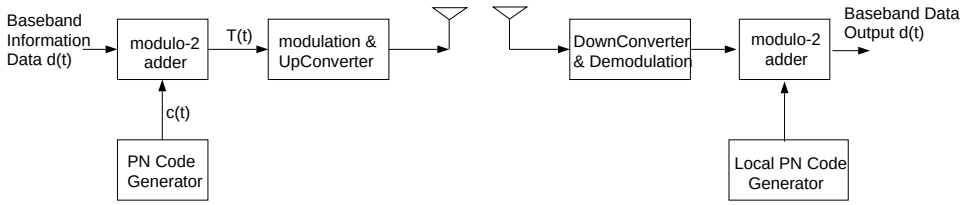

Figure 2.3: A Typical Direct Sequence Spread Spectrum Transmitter and Receiver

$$R(t) \quad = \quad \mathcal{F}^{-1}[\mathcal{S}(\omega)] \tag{2.3}$$

To spread the spectrum equally, a relatively constant power spectral density for $c(t)$ is desired. Ideally, the autocorrelation function $R(t)$, which is the reverse Fourier transform of the power spectral density $S(\omega)$, is an impulse function $\delta(t)$ (see Figure 2.4). Practically, $R(t)$ has a peak value when $t = 0$ and is relatively small when $t \neq 0$. The larger the peak value, the more signal-to-noise ratio (SNR) margin the receiver allows and the more reliable the acquisition and reception of the data.

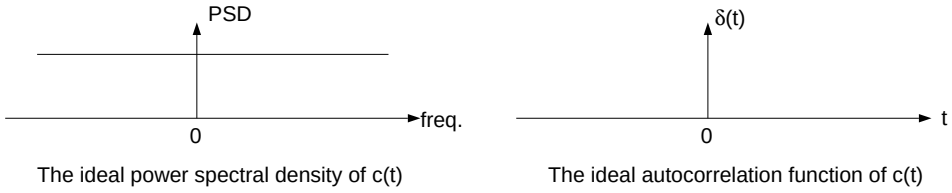

Figure 2.4: The Ideal PSD and Autocorrelation Function of c(t)

Frequency Hopping Spread Spectrum

Frequency hopping spread spectrum (FHSS) is similar to DSSS in the sense of spreading the signal energy over a wider bandwidth than the information bandwidth. The difference is that the available channel bandwidth is divided into a large number of continuous frequency slots and a PN generator "drives" a digital frequency synthesizer to hop among the

center frequencies of these available frequency slots. The hopping pattern of the frequency synthesizer is determined by the output of the PN generator (Figure 2.5).

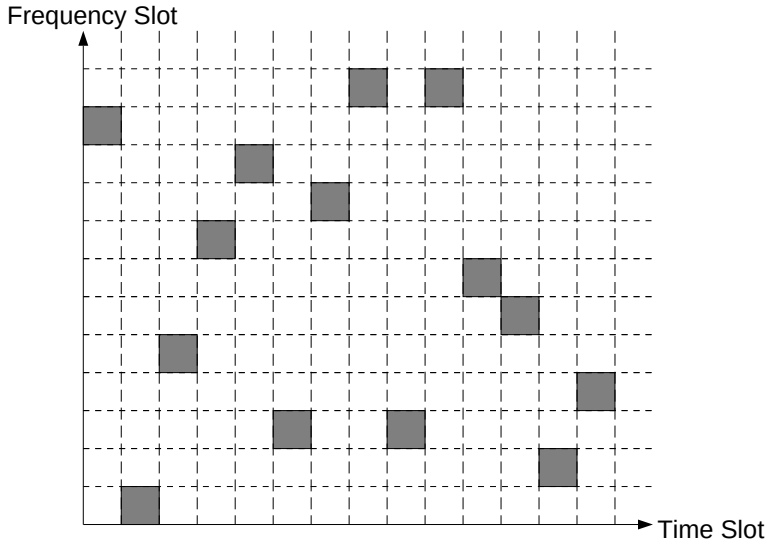

Figure 2.5: Frequency Hopping in the Time and Frequency Domains

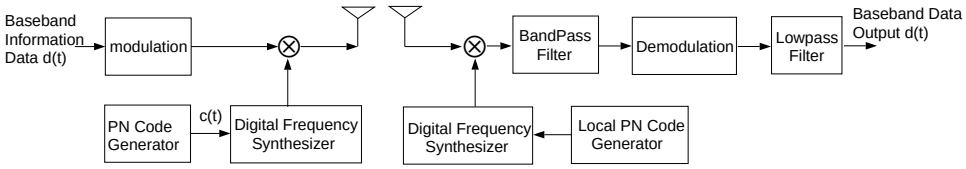

Figure 2.6: A Typical Frequency Hopping Spread Spectrum Transmitter and Receiver

Thus, a FHSS system produces a spreading effect by pseudo-randomly hopping the RF carrier frequency in the available RF band. Here, the processing gain of the FHSS system is given by the ratio of the spread signal bandwidth to the information bandwidth.

2.2.3 Synchronization Requirement in the Spread Spectrum System

In a spread spectrum system, the generated PN code at the receiver side must be aligned or synchronized to the received PN sequence, otherwise, the PN code misalignment will prevent effective despreading. Synchronization is usually accomplished by two processes: an acquisition of the initial PN code alignment followed by a tracking process to eliminate a possible new phase shift introduced to the received signal during the signal reception process.

The synchronization is important in a spread spectrum system. Without synchronization, the spread spectrum will appear as noise at the receiver will is unable to perform despreading. Hence, synchronization performance will greatly affect the overall system data reception capability. We will focus on this topic in Chapter 3.

2.3 Wireless LAN using Direct Sequence Spread Spectrum

2.3.1 Wireless LAN Systems

Wireless network products are targeted primarily for LAN coverage and data rates above 1 Mb/s. They provide mobility to wired counterparts through the wireless channel.

Generally, wireless networks come in two forms. The simple one is a base station LAN system. It is a LAN with wireless transceivers and a central base station which controls the communication access of all the mobile stations. A multi-access reservation mechanism can be used to send a short access reservation packet to reserve longer non-contending slots for actual data packets. This scenario is similar to the communication between a satellite and earth stations. Another form is a stand-alone wireless system, a LAN architecture with integrated wireless transceivers which support communication directly with each station. The wireless media is shared without a central base station or access point. In this case, access is controlled through a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol. In CSMA/CA, the nodes monitor the channel transmission activity and transmit in

sequence. A node first listens to the channel to detect a quiescent period and then continues to wait for a small amount of time. If there is no transmission occurs during this period of time, the station starts transmission. The amount of waiting time depends on where the node is in the network sequence. As each node has a different waiting period, the priority of transmission is different. The node with a longer waiting period is allowed to transmit only after the transmissions of nodes with shorter waiting periods, thus no collision will occur [Ber 92]. Some additional access protocol to maintain bandwidth or channel allocation might be included to improve the overall system performance in a wireless environment.

These two wireless network systems can be integrated with a wired network if the coordination problem is solved. This is because the transmission rates might be quite different. In this thesis, we consider a stand-alone wireless LAN based on the proposed IEEE 802.11 standard.

Three different technologies are currently in use to provide wireless connections for WLAN systems: infrared (IR) systems, licensed cellular systems operating at 18-19 GHz and unlicensed spread spectrum systems operating in the ISM bands.

IR LANs are of two distinct types: line-of-sight (LOS) direct beam infrared (DBIR) or diffused infrared (DFIR). As their signals do not penetrate walls, they are easy to install where signal confinement within a walled room is desired. The disadvantages of IR WLANs are that they suffer interference from the sun and other light sources. Propagation range is limited relative to radio WLANs and they are sensitive to shadowing.

DFIR LANs do not require a direct LOS path between the transmitter and the receiver, and therefore, they are less susceptible to signal blockage and shadowing. However, due to the scattered signal power and the resulting multipath fading, DFIR can provide only moderate data rate and coverage.

DBIR LANs use a focused beam to improve receiver signal-to-noise ratios and reduce multipath propagation. This allows for higher data rate and longer range, but they are best suited only for fixed terminals because of the need for alignment.

Signal propagation in the 18-19 GHz microwave frequency band is significantly con-

strained by dense construction materials such as concrete and steel. Signal transmission can be satisfactory throughout an open office area, but sufficiently confined by the floors and walls to permit frequency reuse by other systems. The disadvantage is that the radio transmitter and receiver at microwave bands are rather specialized and expensive.

As the signal in the ISM bands can penetrate buildings, this system provides a larger coverage than IR and microwave LANs. The inherent advantages of spread spectrum allow several spread spectrum LAN systems to operate in the same area. Also, the unlicensed operation and the variable data rate transmission technique make it attractive. The drawback is that the spread spectrum LAN must overcome current and future possible interference from other users of the same bands. The inherent advantages of direct sequence spread spectrum will be presented in the next section. The variable data rate transmission technique will be discussed in the next chapter.

2.3.2 The Benefits of Using Direct Sequence Spread Spectrum

Resistance to multipath fading and interference are the two inherent reasons for the use of spread spectrum in wireless LAN.

Resistance to multipath fading is obtained from the pseudo-random property of spreading code. The receiver for a spread spectrum signal is designed to despread the received spread signal with the desired PN code sequence. From propagation theory, we know that the receiver will receive time-shifted signals via several paths. The sum of these time-shifted signals is the input to the correlation operation with the locally generated PN code. As discussed before, the autocorrelation function of the PN code is impulsive. Therefore, the correlation with the time-shifted multipath components will be approximately zero as long as the time delays of multipath components are greater than one chip duration. At the output of the despreader, we see only the correlation of transmitted data sequence with itself and the correlation with additive white Gaussian noise (AWGN). Practically, the autocorrelation is not identically zero and there will still be some multipath interference, but it is

greatly attenuated. When the bandwidth of transmitted signal is increased, the chip duration becomes correspondingly smaller and more multipaths can be resolved. Anti-multipath capability is a significant advantage when we realize that the propagation in wireless LAN environment is strongly affected by the construction materials and the building type, especially when the transmitter and receiver operate in an obstructed channel [Rap 95].

Let us look at the model of the spread spectrum receiver in Figure 2.7 and study the signal-to-noise ratio (SNR) performance of the system.

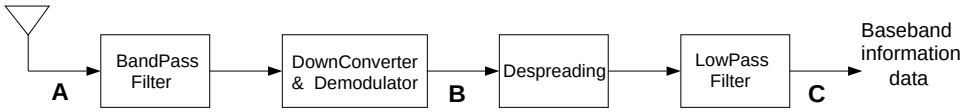

Figure 2.7: The Model of Spread Spectrum Receiver

Assume that the noise at the antenna input is thermal noise only. The definition of SNR is the received power divided by the white noise power which is contained in a bandwidth equal to the signal bandwidth [Cou 90]. The processing gain is defined as the ratio to the SNR out of the low-pass filter divided by the SNR into the despreading process:

$$processing\ \ gain = \frac{SNR\ \ at\ \ C}{SNR\ \ at\ \ B} = \frac{N_B}{N_C} = \frac{\frac{N_o}{2} \cdot 2NB}{\frac{N_o}{2} \cdot 2B} = N \tag{2.4}$$

where N_B is the noise power at B and N_C is the noise power at C. As the signal bandwidth at A is the same as that at B ($2NB$), the processing gain can also be the SNR at C divided by SNR at A.

We know from Section 2.2.1 that the spreading process reduces the power spectral density (PSD) of transmitted signal by N at the price of occupying N times more bandwidth. If the PSD is sufficiently low, it is possible for the spread spectrum signal to be transmitted along with other narrow-band systems without greatly interfering with them. However, as the bandwidth of the bandpass filter of Figure 2.7 is N times wider than the non-spread signal and when the signal power remains the same, the SNR at receiver input is $1/N$ of that

of the system without the spreading. The SNR at received baseband output remains the same regardless of whether or not the spreading-despreading process is introduced. Thus, from the SNR point of view, spread spectrum offers no better performance if the transmitted power is kept the same. This phenomenon will be explained more clearly in terms of E_b/N_o in the next chapter.

Resistance to narrow-band interference is achieved by the processing gain. This kind of interference may come from intentional jamming or from already-existing narrow-band systems. At the receiver, the desired spectrum is despread back to the information bandwidth while the narrow-band interference is spread to modulation bandwidth. The PSD of the interference is reduced by the processing gain. The following low-pass filter has a cut-off frequency which matches the information bandwidth, thus only a fraction $1/N$ of interference power will reach the low-pass filter output. The larger the processing gain, the more the resistance to interference.

Chapter 3

Direct Sequence Spread Spectrum System Design

3.1 Variable Data Rate Transmission Technique

In the overall spread spectrum system, the baseband data $d(t)$ is multiplied twice by the spreading-despreading PN sequence $c(t)$. Since $c^2(t) = 1$, there is no effect on the received despread output signal. Thermal noise, modeled as an additive white Gaussian noise (AWGN), is introduced in the receiver by the low-noise amplifier (LNA) and the down-converter. This noise bandwidth is as wide as that of the spread-spectrum signal. During the despreading process, the noise is multiplied by the despreading PN sequence $c(t)$. Multiplication reverses the polarity of the noise waveform but has no impact on the power spectral density or the probability density function of AWGN. We come to the conclusion that the spreading-despreading operation does not affect the signal energy and does not affect the spectral and probability density function of the noise. For this reason, the overall bit error rate (BER) or the probability of error, P_e, in an AWGN channel, is the same as the BER performance of the modulated and demodulated system, without spread spectrum. In our case, if we assume that non-coherent differential phase shift keying (DPSK)

modulation is used, then the BER performance of this spread spectrum system is found to be approximated by [Pro 95]:

$$BER = P_e = \frac{1}{2} e^{\frac{-E_b}{N_o}} \tag{3.1}$$

where E_b is the average energy of a received bit and N_o is the noise power density.

In this wireless LAN application, we use a symbol synchronous PN modulation technique. The PN code repeats itself once per symbol and is aligned with the symbol transitions, as shown in Figure 3.1.

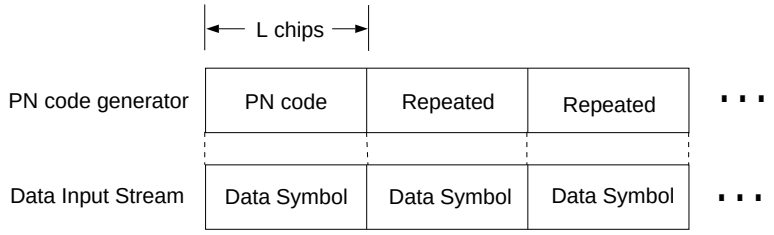

Figure 3.1: Symbol Synchronization

Assuming that the PN code length is L chips, E_b can be given by the energy of each PN chip E_c times L:

$$E_b = L \cdot E_c \tag{3.2}$$

hence, the BER or P_e is also given by:

$$BER = P_e = \frac{1}{2} e^{\frac{-E_c \cdot L}{N_o}} \tag{3.3}$$

If we decrease the input data rate (lengthen the symbol period) without changing the PN chip rate (limited by RF receiver automatic gain control bandwidth) as shown in Figure 3.2, the bandwidth of the spread signal is unchanged as the PN chip rate remains the same. In addition, the energy of each PN chip E_c is unchanged. Consequently, L is increased and,

thus, the processing gain and the energy per bit E_b are increased. It can be seen from Equation 3.3 that as LE_c/N_o increases, the BER will decrease.

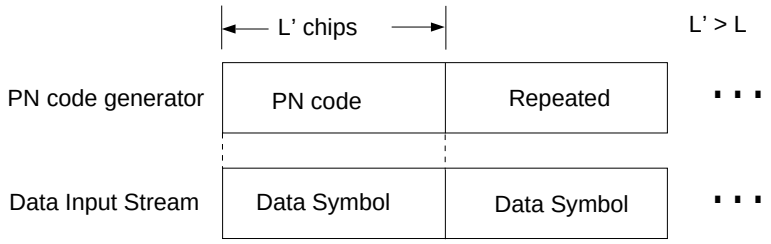

Figure 3.2: Decreasing the Data Rate

Therefore, we find that by increasing the length of the PN code for a given chip rate, we have better BER performance at the cost of a lower data rate. From this mechanism, we come to our variable data rate transmission technique.

1. When the wireless channel is degraded and BER is high, we increase the PN code length and slow down the data transmission rate to compensate for channel deterioration.

2. When the channel is clear, we use a short PN code and increase the data transmission rate to increase the throughput.

By dynamically changing the transmission rate according to the channel performance, we have the following benefits.

1. A disconnected link is less likely to occur and data transmission remains continuous, but perhaps at a lower rate.

2. The channel transmission capacity is fully used in both the bad channel and good channel cases.

The theory that longer PN codes have a better BER performance for a given chip rate leads to the second application in our system. We may use a long PN code for the preamble and a short one for the information data. Thus, by this assignment, we improve the burst acquisition reliability and consequently, increase the packet throughput (or overall data throughput) of the system.

The variable data rate transmission technique is useful in the wireless environment as the physical channel is always impacted by the movement of transceivers from one location to another, by a changing physical environment, or by interference from outside signals.

From the above discussion, we obtain the formula for calculating the channel data rate as follows:

$$channel\ data\ rate\ = PN\ chip\ rate/PN\ code\ length \qquad (BPSK) \tag{3.4}$$

$$channel\ data\ rate\ = 2 \times (PN\ chip\ rate/PN\ code\ length) \quad (QPSK) \tag{3.5}$$

3.2 PN-Despread Techniques

3.2.1 Symbol Acquisition and its Impact on System Performance

In a spread spectrum system, synchronization is a common requirement and is usually combined with the PN-despreading process: acquiring initial alignment of the transmitted and local codes within a time-frequency uncertain region (acquisition) and, secondly, reducing and maintaining the alignment error as much as possible (tracking). Typically, the acquisition process is realized by continuously monitoring some function of despread output signal. If this function of the output signal level exceeds some specified threshold, then an initial acquisition is declared followed by a transition to fine-alignment tracking process. There are two basic approaches to the direct-sequence spread spectrum despreading function: the PN matched filter and the sliding correlator.

With the bursty data transmission in LAN applications, a fast-acquisition with high detection probability PN-despreading process is desirable to minimize the physical-level processing time. This improves the system performance in terms of overall system throughput. The transmitter should have a preamble that is long enough for the receiver to perform acquisition. Thus, the longer the average acquisition time, the longer the preamble required and the greater the physical layer overhead. This results in reduced overall throughput.

Secondly, since the radio environment in which the wireless LANs operate is usually adverse, the transmitted signal is distorted by multipath fading, collapsed with thermal noise, or interfered by other signals. This distorted signal may cause the receiver to make incorrect decisions, declaring an acquisition while missing the correct position, or just missing the detection. In both cases the packet will be declared lost by the receiver after a time-out period and the transmitter has to retransmit the packet while other users have to wait. Such packet loss will not only decrease the overall system throughput, but will also increase the mean transmission time of packet and, hence, a longer delay can be expected.

Thus, mean acquisition time, detection probability, and false alarm probability are important criteria for analyzing the performances of PN matched filters and sliding correlators.

3.2.2 Sliding Correlator

Figure 3.3 shows a sliding correlator. The operation of this correlator is as follows. The signal after the multiplication operation is simply integrated over a fixed period of time (the integration period). At the end of this time period, a threshold detector is used to decide whether the correlator output signal corresponds to an in-phase condition or out-of-phase condition, and hence whether or not initial synchronization can be declared. If the threshold is not exceeded, the locally generated PN code is delayed by half of a chip period and this process repeats until an initial synchronization has been achieved. The integration period here is a key parameter for which each system design must trade-off based on its peculiar performance requirements. A short integration period integration results in a low probability of detecting the in-phase signal P_d, and a high probability of false alarm detections P_f. A long integration period, on the other hand, improves the reliability of a correct decision, but the time taken to dismiss each out-of-phase conditions is longer and, consequently, the mean acquisition time is longer.

3.2.3 PN Matched Filter

A matched filter is a linear filter which maximizes the output signal-to-noise ratio, given

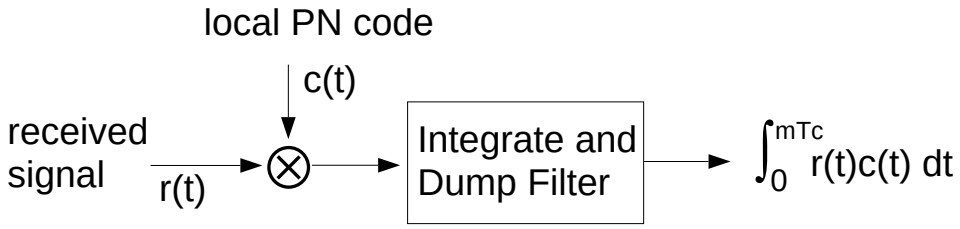

Figure 3.3: A Sliding Correlator

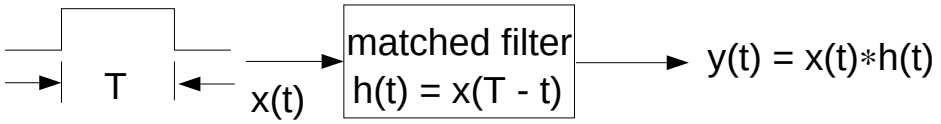

Figure 3.4: A Matched Filter

an input signal waveform [Cou 90]. When the input noise is white Gaussian, the impulse response of the matched filter becomes:

$$h\left(t\right) = x\left(T - t\right) \tag{3.6}$$

where $x(t)$ is the input signal with interval from 0 to T. The output of the matched filter is given by

$$y\left(t\right) = x(t) \star h(t) = \int_0^t x\left(\tau\right) h\left(t - \tau\right) d\tau = \int_0^t x\left(\tau\right) x\left(T - t + \tau\right) d\tau \tag{3.7}$$

It can be shown that $y\left(t\right)$ is the autocorrelation function of $x\left(t\right)$. $y(t)$ reaches its peak value when it is sampled at time $t = T$:

$$y(T) = \int_0^T x(\tau)x(t)d\tau = \int_0^T x^2(\tau)d\tau \tag{3.8}$$

The impulse response of this filter is said to match the signal $x(t)$.

A transversal digital matched filter is usually used for PN despreading in a spread spectrum system. The structure of this transversal digital matched filter is shown in Figure 3.5.

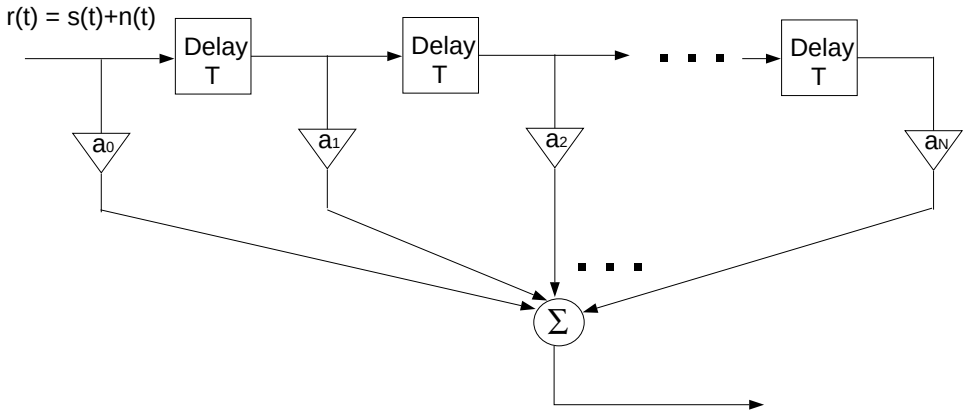

Figure 3.5: A Transversal Matched Filter

The set of transversal matched filter coefficients $a' = [a_1 \;\; a_2 \;\; a_3 \cdots a_N]$ are found in [Cou 90] to be:

$$a = R^{-1} \cdot s \tag{3.9}$$

where s is the known signal vector $s' = [s_1 \;\; s_2 \;\; s_3 \cdots s_N]$ and R is the known autocorrelation matrix for the input noise:

$$\mathrm{R} = \begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1N} \\ r_{21} & r_{22} & \ldots & r_{2N} \\ . & . & \ldots & . \\ . & . & \ldots & . \\ r_{N1} & r_{N2} & \ldots & r_{NN} \end{bmatrix}$$

If we assume that the input noise is white Gaussian noise, then

$$r_{mn} = \begin{cases} 0, & m \neq n \\ \sigma_0^2, & \mathrm{m = n} \end{cases} \tag{3.10}$$

where σ_0^2 is the noise power. Thus,

$$R = \sigma_0^2 \cdot diag\,(N) \tag{3.11}$$

and

$$a = R^{-1} \cdot s = \left(\sigma_0^2 \cdot diag\,(N)\right)^{-1} \cdot s = \sigma_0^{-2} \cdot diag\,(N) \cdot s = \sigma_0^{-2} \cdot s \tag{3.12}$$

Therefore, if the transversal matched filter's coefficient vector is proportional to the incoming signal vector, the instantaneous output signal-to-noise ratio is maximized. In practice, σ_0^2 affects only the gain of the transversal filter, and a is simply set to be s. The PN matched filter based on this derivation of transversal matched filter is depicted in Figure 3.6.

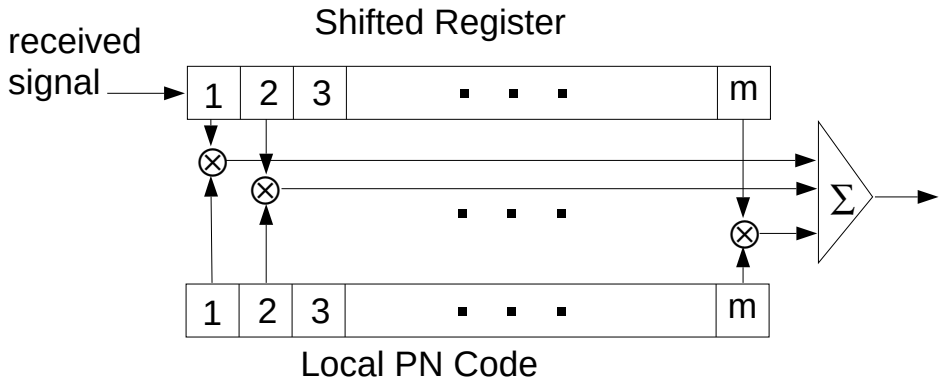

Figure 3.6: A PN Matched Filter

3.2.4 Mean Acquisition Time Analysis

Generally, there are two analytical methods to derive the mean acquisition time: a discrete-time Markov chain model and a time-domain combinational technique. The former models the acquisition process as a Markov statistical process with a state transition diagram. The transition probabilities of each state transition are calculated. Using appropriate Markov

chain properties and signal flow-graph connection reduction techniques, the moment generating function of the acquisition process can be derived and the first-and second-order statistics of acquisition time can be determined from this moment generating function. The Markov chain model, which is a systematic method, requires a background in statistical processes, but it can be modified to model some complicated search strategies. A detailed discussion of this method is presented in [Pol 84].

The time-domain combinational technique provides more insight and understanding of the acquisition process [Jib 91]. Assume a single-dwell serial search acquisition system. Single-dwell means that one fixed integration period is applied to generate decision statistics for every threshold comparison. The dwell-time or integration period is defined as τ_d. Let P_f denote the probability of false alarm (the error made by the threshold detector due to noise). We assume that the following verification mode will detect the false alarm with probability of 1 and that the penalty time of false alarm is $k\tau_d$, where k is an integer. The elapsed time between test cells is now τ_d if no false alarm happens (with probability of $1 - P_f$) and, $(k+1)\,\tau_d$ when false alarm happens (with probability of P_f). Thus, the average time spent per test is

$$\overline{T}_o = \tau_d\,(1 - P_f) + (k+1)\,\tau_d P_f = (kP_f + 1)\,\tau_d \tag{3.13}$$

Secondly, we consider the missed detection probability. Let q denote the total number of search cells and P_d the probability of detecting correct phase position. If a missed detection happens, another q tests should follow before the next possible correct detect. If we assume the time uncertainty region is uniformly distributed from 0 to $q-1$ with the mean value of $\frac{q-1}{2}$, then the average acquisition time $\overline{T}_{acq}$ becomes:

$$\begin{aligned}
\overline{T}_{acq} &= \frac{q-1}{2}\overline{T}_o + q\overline{T}_o\,(1 - P_d)\,P_d + 2q\overline{T}_o\,(1 - P_d)^2\,P_d + 3q\overline{T}_o\,(1 - P_d)^3\,P_d + \cdots \\
&= \frac{q-1}{2}\overline{T}_o + q\overline{T}_o P_d \sum_{k=1}^{\infty} k\,(1 - P_d)^k
\end{aligned}$$

$$\begin{aligned} &= \frac{q-1}{2}\overline{T}_o + q\overline{T}_o\frac{(1-P_d)}{P_d} \\ &= q\overline{T}_o\frac{(2-P_d)}{2P_d} - \frac{\overline{T}_o}{2} \end{aligned} \tag{3.14}$$

If q is a large integer, then

$$\begin{aligned} \overline{T}_{acq} &= q\frac{2-P_d}{2P_d}\overline{T}_o \\ &= q\frac{2-P_d}{2P_d}\cdot(kP_f+1)\,\tau_d \end{aligned} \tag{3.15}$$

Basically, the difference in acquisition time between a sliding correlator and a PN matched filter is the dwell time τ_d. In the PN matched filter, τ_d is half chip duration, while in the sliding correlator, τ_d is a multiple of the chip duration (10 to 100) ensuring that the decision is based on enough received information. The minimum integration period depends solely on the length of the PN-sequence used. [Dav 78] shows that the minimum value must be n for maximum-length code with code length $2^n - 1$. For non-maximum length code operation with a short code length, the integration period is typically chosen to be the whole code length. As we see, the PN matched filter has a short acquisition time compared to the sliding correlator. The presence of data modulation, frequency offsets and Doppler effect place a lower limit on the integration period and the sliding correlator is better in the low signal-to-noise ratio situation as a longer correlation period can be used.

In this project, we chose the Zilog Z2000 chip [Adv 94] (see Chapter 4) for direct sequence spread spectrum processing. A transversal PN matched filter is incorporated in this chip to perform the preamble and data despreading. As its mean acquisition time is much smaller than that of the sliding correlator, the PN matched filter satisfies the fast acquisition requirement of our wireless LAN application.

3.3 Spreading Code Design

3.3.1 The Despreading Process and Partial Correlation

In wireless LAN applications, data transmission is bursty and the receiver is not synchronized to the other transmitters when they are not transmitting. This means that the local PN code generator has no information about when the signal comes. Therefore, code acquisition is needed at the receiver each time a burst arrives, otherwise, the PN code despreader will not have the timing information to correctly despread the signal. At least one preamble symbol is required precede the information symbols. After the receiver detects this preamble symbol (acquisition), it generates the exact timing signal, based on the PN chip rate, for the PN despreader to extract the information symbols.

Let us study the general model of the despreading operation, as shown in Figure 3.7.

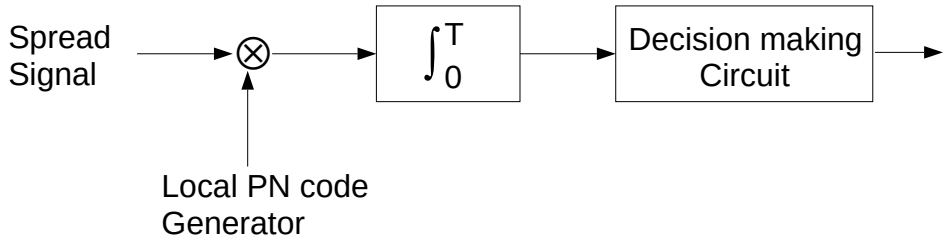

Figure 3.7: A General Model of Despreading Process

As mentioned before, the receiver has to detect the preamble symbol, but before the whole preamble PN code has been received and shifted chip-by-chip into the PN despreader, there is only a partial overlap between the received and the local PN code. This is called the partial correlation which results in a noisy input for the decision circuit of the acquisition process [Yan 95]. Thus, we want this partial correlation of preamble PN code to be as small as possible compared to its the autocorrelation peak. Moreover, this partial correlation is proportional to the magnitude of the signal, and it can not be eliminated by increasing the signal power. We should be careful to choose a PN code with a good autocorrelation

function and, also, whose partial correlation is considerably below its autocorrelation peak. By doing this, we provide a large threshold margin for the acquisition and symbol decision circuit and will have small false-alarm and missed-detect probabilities.

Several kinds of PN codes, such as Walsh codes, M-sequence codes, Gold codes and Barker codes have been proposed for different applications of spread spectrum. In the next section, we describe the M-sequence and the Barker codes. These two codes have good periodic autocorrelation properties and their partial correlation is well below the autocorrelation peak. In Section 3.3.3, three codes proposed by Zilog and Utilicom are compared in terms of their correlation performance.

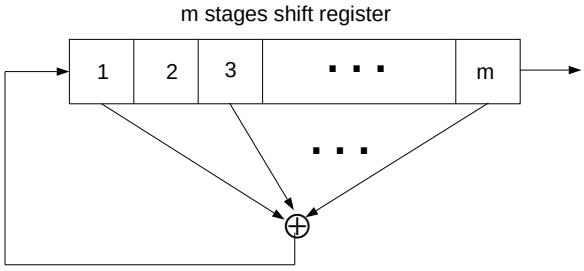

Figure 3.8: M-Sequence PN Code Generator

3.3.2 M-Sequences and Barker Codes

M-sequences, also called maximum-length shift-register codes, are usually generated by the mean of an m-stage digital shifted register with feedback as shown in Figure 3.8. The output sequence is periodic with length $n = 2^m - 1$. The most important property of this maximum length PN-code is that its periodic autocorrelation function is impulse-like: n when the time shift is zero and -1 for all other shifts. This implies that the power spectrum is nearly white and, hence, the sequence resembles white noise which is the desired property for the spread spectrum system. However, the partial correlation function of an m-sequence code is not as good as that of a Barker code.

Barker codes are characterized by a partial correlation function that is negative or zero at all time shifts except at the zero time shift. For this reason, Barker codes are ideal for preamble acquisition. Besides, theirs periodic autocorrelation function is also impulse-like, just the same as for M-sequence codes. Therefore, Barker codes are also ideal for information data detection. Unfortunately, Barker codes only exist for code lengths of L = 2, 3, 4, 5, 7, 11, 13 [Rao 88] and thus offer limited processing gain (the maximum is only $10 \log 13 = 11.1$ dB).

3.3.3 Correlation Function Comparison

In this project, we use the Zilog Z2000 Evaluation Board [The 94] to perform spread spectrum processing. Zilog provides an emulation program to demonstrate the operation of this board. In this program, an 11-bit Barker code, one 64-bit code, and another 63-bit code are suggested. We have determined the partial correlation functions and periodic autocorrelation functions for each code, as shown in Figures 3.9, 3.10 and 3.11.

A comparison of these figures shows that the Barker code has ideal partial correlation and periodic autocorrelation functions. From the periodic autocorrelation function, we find that the 63-bit code is actually a M-sequence code. Although its periodic autocorrelation is impulse-like as desired, its partial correlation will introduce noise into the acquisition circuit. Thus, we conclude that the Barker code is optimum for preamble and data spreading, but is processing-gain-limited, and that the M-sequence code is optimum only for data spreading. Considering that data may be transmitted in a noisy environment and that we want to use a longer PN code with larger processing gain in such circumstance to improve the acquisition probability, the M-sequence code is considered to be a sub-optimum solution for the preamble spreading code. The other 64-bit code is shown to have poor correlation properties and will greatly degrade system performance.

In our variable data rate transmission technique, we will use the same PN spreading code for preamble and data. The 11-bit Barker code is chosen for the high data rate and this 63-bit M-sequence code for the low data rate.

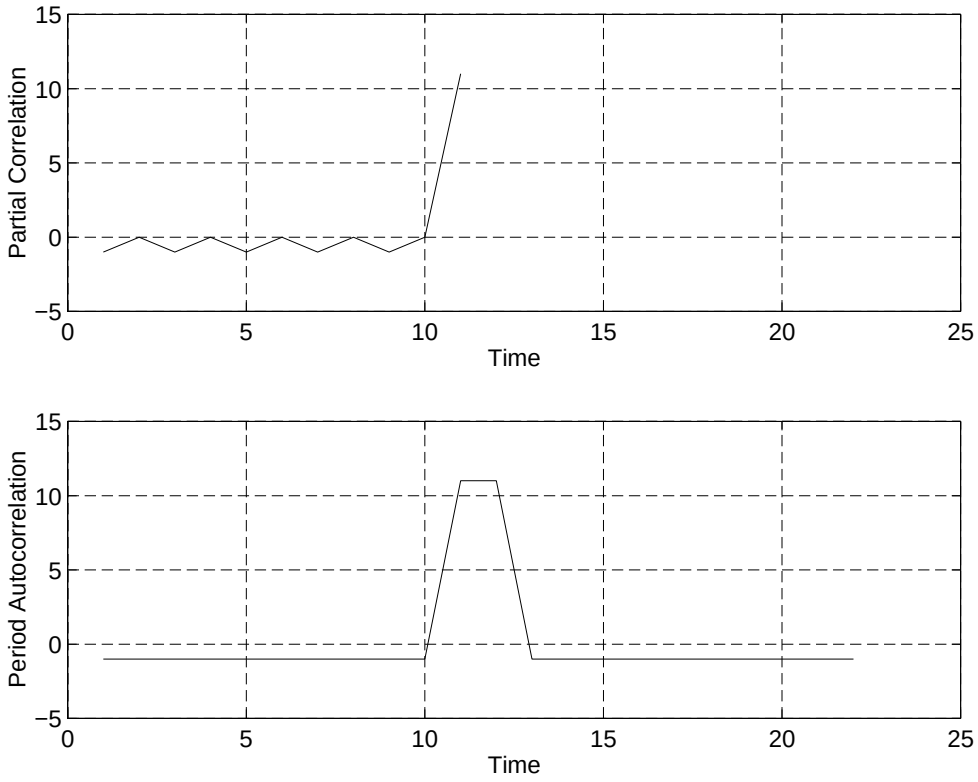

Figure 3.9: 11-Bit Barker Code Correlation Function

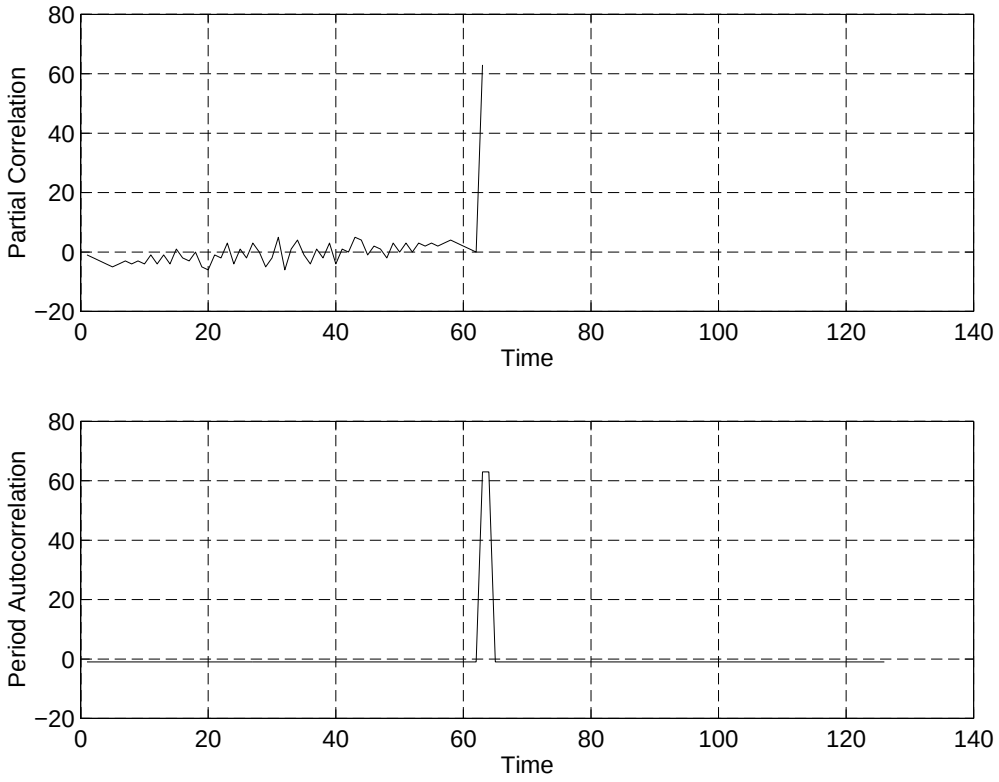

Figure 3.10: 63-Bit Code Correlation Function

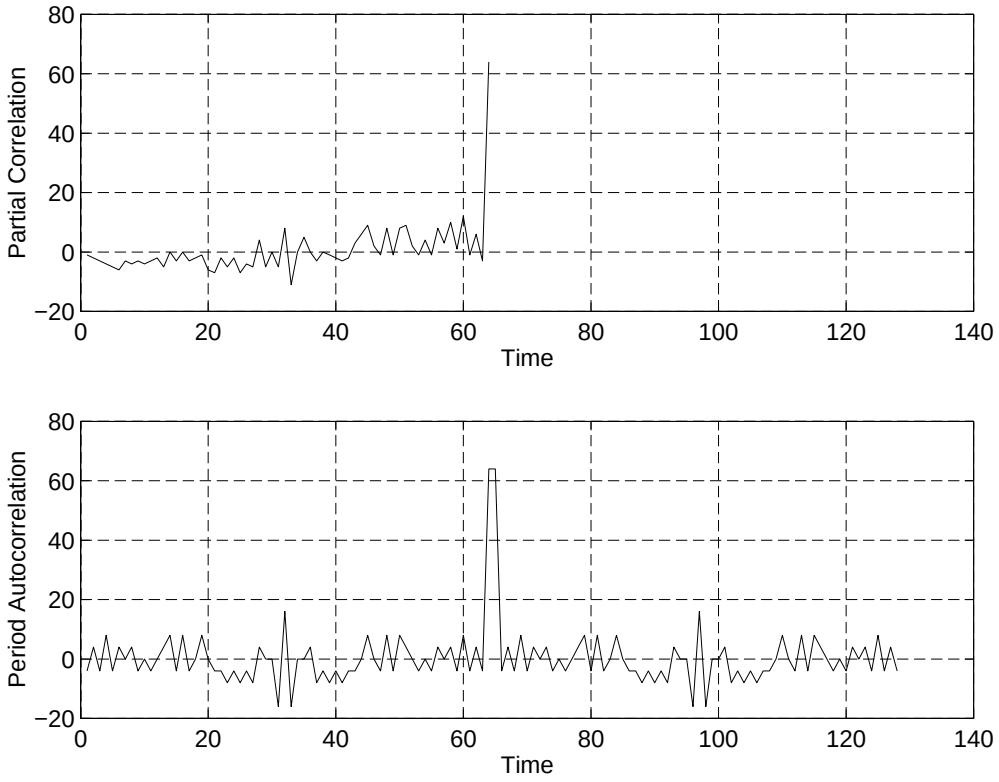

Figure 3.11: 64-Bit Code Correlation Function

Chapter 4

Operation of the Zilog Z2000 Chip

4.1 Overview

The Z2000 [The 94] is a 100-pin spread spectrum digital processor manufactured by Zilog. It combines a direct sequence spread spectrum transmitter and a receiver together in A single chip. On the Zilog Z2000 Evaluation Board, the Z2000 works with Zilog's Z80182 intelligent peripheral controller. With microcode residing in its random access memory (RAM) space, the controller carries out the system configuration (parameters setting) for

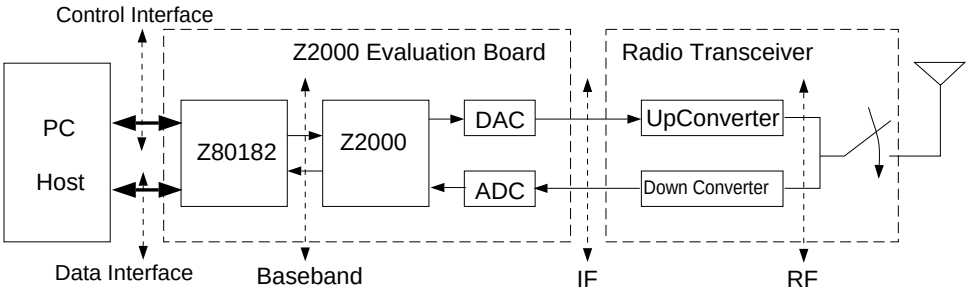

Figure 4.1: The Overall Transceiver System

the Z2000 chip and data buffering and packetizing for communications between the host and the Z2000 transceiver (Figure 4.1). We will focus on the Z2000 chip in this chapter. The

overall system hardware and software issues will be addressed in Chapter 5 and Chapter 6, respectively.

The transmitter in the Z2000 chip incorporates a PN spreader and a DPSK modulator while the receiver integrates a digital down converter, a PN matched filter (despreader), and a DPSK demodulator. The block diagram is shown in Figure 4.2. The functionality of each block is programmable by the Z80182 controller through the Z2000 control registers. We will discuss each functional block of the Z2000 in the following sections.

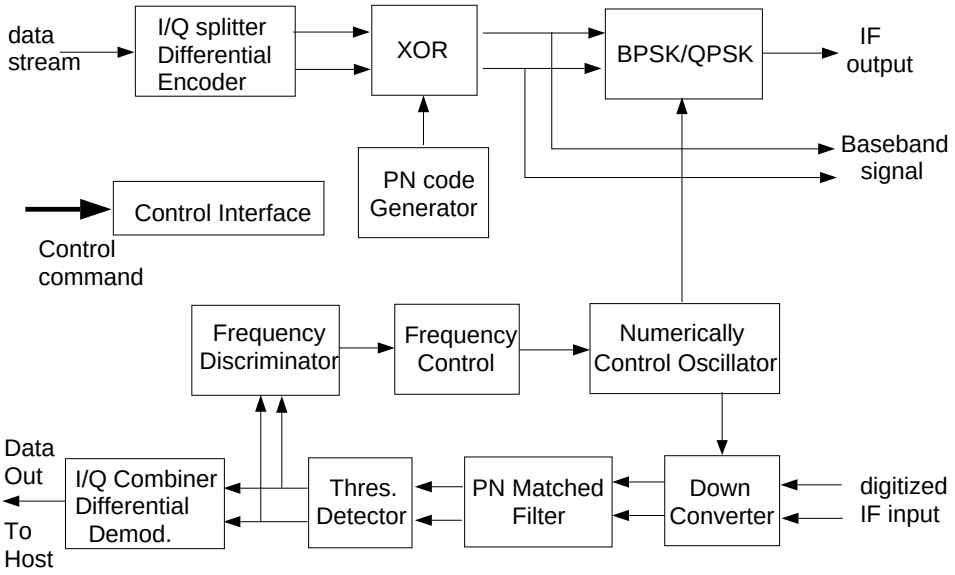

Figure 4.2: The Structure of the Z2000

4.2 Transmitter

The transmitter incorporates a differential encoder, a PN spreader and a BPSK modulator with a carrier frequency derived from a numerically controlled oscillator. The block diagram is shown in Figure 4.3.

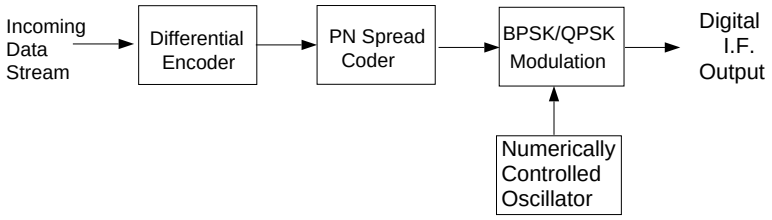

Figure 4.3: The Z2000 Transmitter

4.2.1 Differential Encoder

In the differential encoder, the polarity of the data is presented in the phase change between consecutive bits. The differential decoder can correctly recover the original data as long as the phase difference between successive symbols is maintained after the transmission. The absolute phase of each symbol is not required for correct decoding. Hence, the differential modulation allows a phase ambiguity that may be introduced by some carrier recovery circuits.

For BPSK operation, the differential encoder compares the input bit $in(k)$ with the previous output bit $out(k-1)$ by a logical exclusive-OR operation:

$$out(k) = in(k) \oplus out(k-1) = \overline{in(k)} \cdot out(k-1) + in(k) \cdot \overline{out(k-1)} \tag{4.1}$$

where $\oplus$ represent the exclusive-OR operation or modulo-2 addition. This is illustrated in Figure 4.4. An arbitrary reference binary digit is assumed for the initial bit of the output sequence.

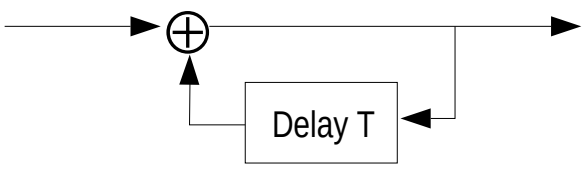

Figure 4.4: The Differential Encoder

For QPSK operation, the differential encoder is more complicated. There are four possible states in each of the inputs and outputs of the differential encoder. The 16 combinations are shown in Table 4.1.

Table 4.1: The 16 Combinations for QPSK Differential Encoder

New input	Previously Encoded $Out(I,Q)_{k-1}$							
$In(I,Q)_k$	0 0		0 1		1 1		1 0	
0 0	0	0	0	1	1	1	1	0
0 1	0	1	1	1	1	0	0	0
1 1	1	1	1	0	0	0	0	1
1 0	1	0	0	0	0	1	1	1

By a logical truth table transformation, the encoded output can be expressed in terms of the input states and previous states as follows:

$$\begin{aligned} Out(I,k) &= A\overline{C}\,\overline{D} + B\overline{C}D + \overline{A}CD + \overline{B}C\overline{D} \\ Out(Q,k) &= B\overline{C}\,\overline{D} + \overline{A}\,\overline{C}D + \overline{B}CD + AC\overline{D} \end{aligned} \tag{4.2}$$

if

$$\begin{aligned} A &= In(I,k); \qquad B = In(Q,k); \\ C &= Out(I,k-1); \quad D = Out(I,k-1). \end{aligned} \tag{4.3}$$

For ease of understanding, we can consider the signal constellation of QPSK signal(I,Q). Note that here the constellation is Gray encoded, so that the most likely errors caused by noise will result in a single bit error in the k-bit symbol.

It is easy to find that

$$Ang[out(k)] = U\{Ang[in(k)] + Ang[out(k-1)]\} \tag{4.4}$$

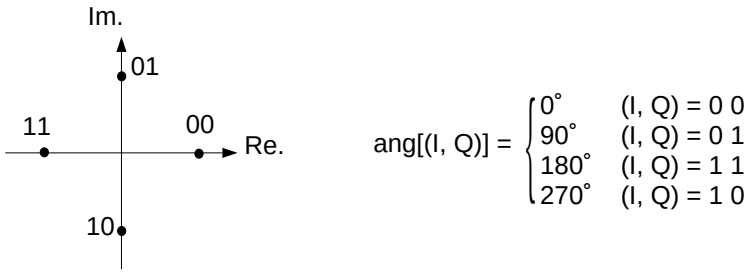

Figure 4.5: The Signal Constellation of QPSK

where $Ang(\cdot)$ is the angle of the signal and $U\{\cdot\}$ is an unwrapped phase function (modulo-360^o operation). This explanation will help us to understand the differential decoder process. Actually, the decoded output (I, Q) can be found from its constellation, whose angle value is calculated by:

$$Ang[decoded\ \ out(k)] = U\{Ang[in(k)] - Ang[in(k-1)]\} \tag{4.5}$$

4.2.2 PN Spreader

As we said earlier, the Z2000 uses symbol-synchronous PN modulation in data transmission and reception. This means that every data symbol (I and Q symbols in QPSK mode) coming out of the differential encoder is XOR-ed with one complete PN code sequence.

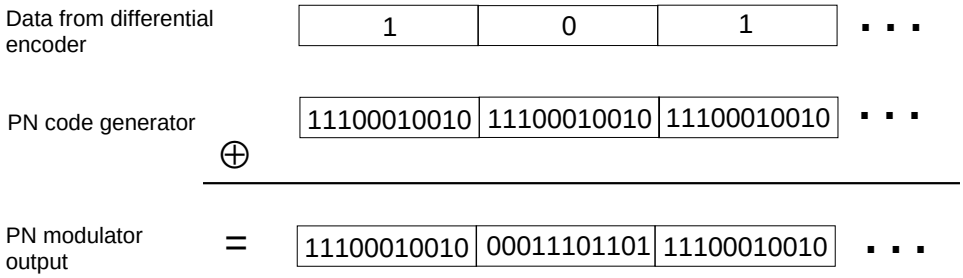

Figure 4.6: The Operation of PN Modulation

Thus, we see that the data rate is increased by the length of PN code and the spectrum is spread after PN modulation by the same number. The Z2000 allows two independent PN codes to be employed: one for preamble and another for information data. Moreover, the lengths of these two PN codes may be different. However, the length of both have an upper bound of 64 chips due to hardware limits and a lower limit of 10 chips because of the FCC Part 15 requirement of 10 dB minimum processing gain.

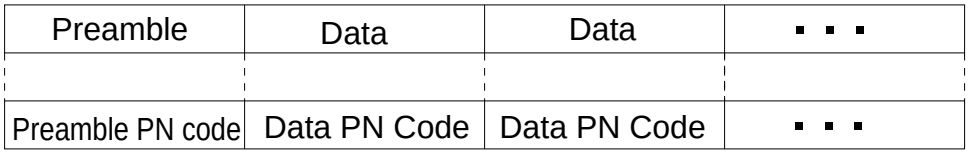

Figure 4.7: The Alignment of PN Code with Preamble and Information Data

It is suggested that a longer PN code be used for the preamble. This takes advantage of the higher processing gain and it will improve burst acquisition performance. The lengths and their coefficients of both PN codes are programmed by Z80182 control registers.

4.2.3 BPSK/QPSK Modulator

The BPSK/QPSK modulator implemented in the Z2000 is a fully digital modulator. The spread signal is translated to a digitized immediate frequency (IF) modulated signal by multiplying it with a digitized (discrete and quantized) sine and/or cosine waveform from the numerically controlled oscillator (NCO). This output signal is then fed into an external digital-to-analog converter (DAC) and the analog output can be up-converted to RF for transmission in the radio channel. The NCO output is also used in the Z2000 receiver to translate the received IF to baseband. If the receiver's IF is chosen to be different than the transmitter's IF, then an external BPSK/QPSK modulator for the transmitter is required. It uses the PN spread signal (I and/or Q channel) from the PN spreader directly and an analog carrier signal as its two inputs. (This is shown in detail in Chapter 5). With an

external modulator, the NCO frequency can be chosen by only considering the operation of the receiver. (See Section 4.3.2 for how to choose the NCO frequency.)

The NCO frequency is programmed in a 32-bit Frequency Control Register and is clocked by the receiver's clock signal RXIFCLK. To avoid destructive aliasing, the NCO frequency may not be programmed above 50% of RXIFCLK. Negative frequencies can be realized by the two's complement operation. The formula for calculating this register value is given in [Z20 94].

4.3 Receiver

The sampled and digitized received IF signal is first down-converted to baseband and then PN-despread with a PN matched filter. A symbol tracking processor incorporates a threshold detector to detect a correlation peak once per symbol and a "flywheel circuit" to track the symbol. The block diagram of the receiver is shown in Figure 4.8.

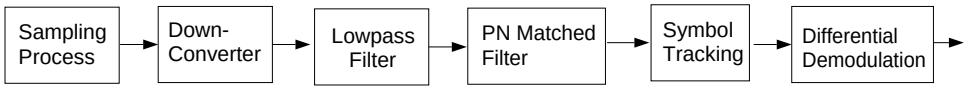

Figure 4.8: The Z2000 Receiver

4.3.1 Sampling Process

The analog IF signal is first converted to a digital IF signal by on-board analog-to-digital converter (ADC) clocked at RXIFCLK. The frequency of RXIFCLK is provided by an on-board crystal oscillator with a maximum value of 45.056 MHz. We use this maximum value in our application. The sampling process introduces a spectrum image; the fundamental and harmonics of the sampling frequency (RXIFCLK) result in images of the input signal spectrum centered at the multiples of the sampling frequency (Figure 4.9). In other words, copies of the analog signal spectrum are shifted by integer multiples of the sampling fre-

quency and then superimposed to produce a periodic signal spectrum. By this mechanism, a band-limited signal with a frequency higher than the sampling rate can still be sampled without distortion as long as the bandwidth of the input signal satisfies the Nyquist sampling theorem, $BW < \frac{1}{2}RXIFCLK$, and the frequency is supported by the external ADCs. Note that the BW here is referred to the single-side bandwidth of the baseband signal.

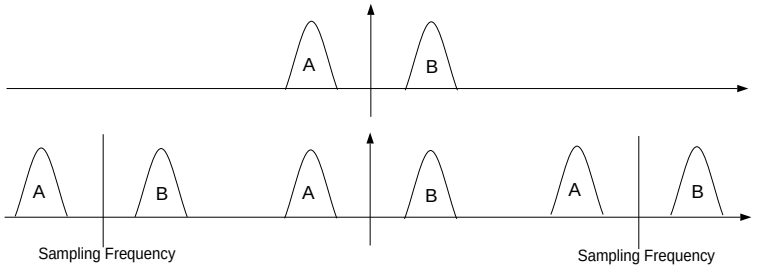

Figure 4.9: Spectrum Aliasing after Direct IF Sampling of a BPSK Signal

4.3.2 Down Converter

The numerically controlled oscillator generates a complex (sine and cosine) signal. It moves the center frequency of signal spectrum replica, which is the closest to zero frequency, to zero frequency. This can be done in two ways: a positive NCO frequency moves the left spectrum replica (A in Figure 4.9) to zero frequency and a negative NCO frequency moves the right one (B in Figure 4.9) to zero frequency.

Here, we need to talk about two sampling methods that Z2000 provides: direct IF sampling and quadrature sampling. In direct IF sampling mode, only one ADC is used with the other one disabled or held to zero (see Figure 4.10).

Figure 4.9 and 4.11 show the BPSK and QPSK spectrum after the direct IF sampling. To clearly show the I and Q channels' spectrum in the same graph, we use the different spectral heights to represent the I and Q channel's spectrum. They must be the same height in the reality if the I and Q channels of a QPSK signal are balanced. The phase is shown

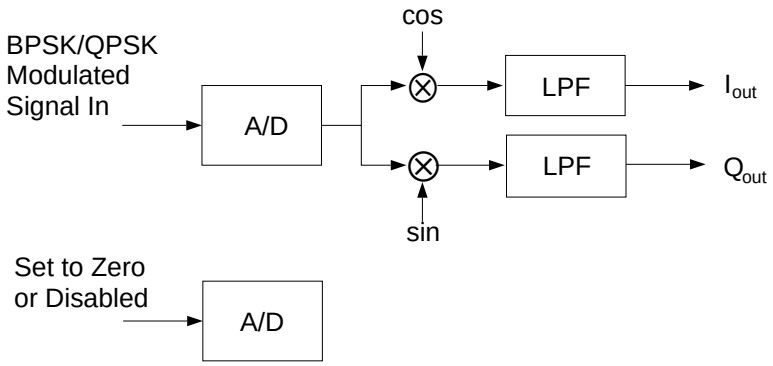

Figure 4.10: The Operation of Direct IF Sampling

by $\pm j\pi$.

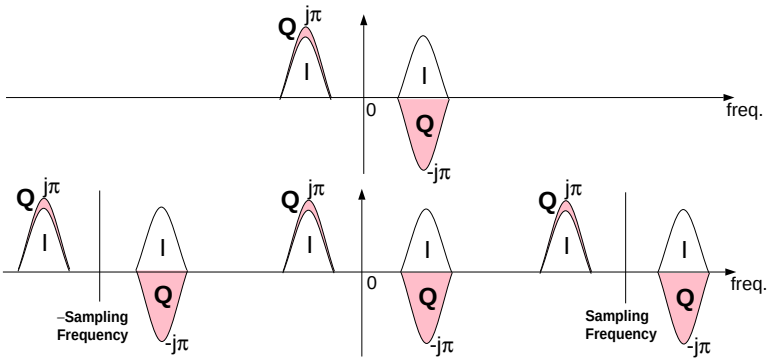

Figure 4.11: Direct IF Sampling of a QPSK Signal

As we observe from Figure 4.9, it does not matter for a BPSK signal which way the spectrum is moved by the complex NCO. However, if the signal is QPSK modulated, an incorrect sign of the NCO will move a spectral inverted Q signal to the zero frequency. This will cause a 180^o phase inversion to the Q signal. In this case, the sign of the NCO frequency is determined by high-side or low-side conversion. High-side conversion means that an IF signal is down-converted by a local oscillator whose frequency is higher than the signal center frequency. As we see in Figure 4.12, a negative NCO frequency should

be applied. On another hand, the low-side conversion should use an NCO with positive frequency. If the opposite sign is chosen, the Z2000 provides a second chance to recover the inverted Q channel signal in the Output Control Processor. As we know, in a QPSK signal, the I channel leads the Q channel by 90^o. The inverted Q channel spectrum introduces an inversion to the Q channel and this can be corrected by reversing the output sequence to Q first followed by I. This sequence reversing operation can be accomplished by a setting on the Output Processor Control Register.

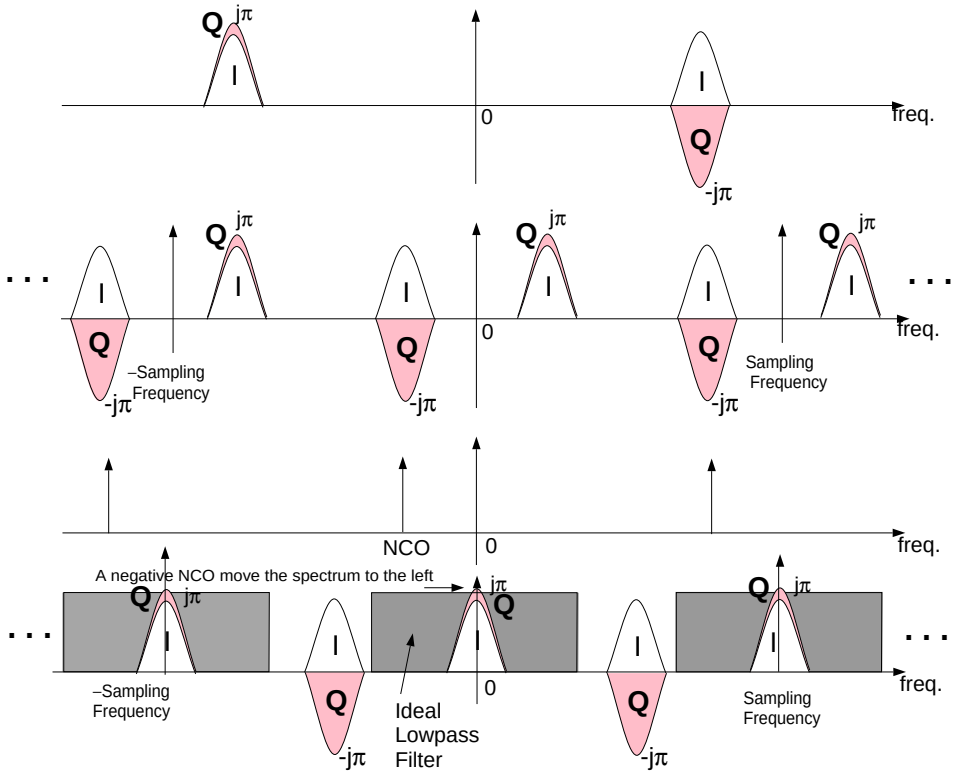

Figure 4.12: High Conversion With Direct IF Sampling

In quadrature sampling mode, two ADCs are used simultaneously. Figure 4.13 shows the operation of the quadrature sampling method. Equation 4.6 shows the mathematical formula for this down-conversion operation.

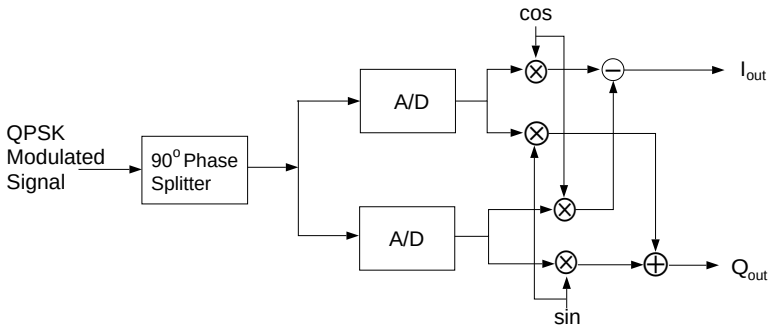

Figure 4.13: The Operation of Quadrature Sampling

$$\begin{aligned}(I_{in} + Q_{in})(cos\omega t + jsin\omega t) &= (I_{in}cos\omega t - Q_{in}sin\omega t) + j(I_{in}sin\omega t + Q_{in}cos\omega t) \\ &= I_{out} + jQ_{out} \end{aligned} \tag{4.6}$$

where ω is the angle frequency of the NCO output.

This sampling method is equivalent to when the input signal is complex and is thus only defined in the positive frequency domain. The two ADCs converting the real I and Q channel signals can be thought of an ADC converting a complex $I + jQ$ signal. Thus, this sampling does not introduce an undesired alias in the primary Nyquist region. This is shown in Figure 4.14. A negative NCO frequency will correctly down-convert the spectrum to baseband.

4.3.3 Low-Pass Filter

A digital low-pass filter is required with the direct IF sampling to remove the other undesired spectrum in the primary Nyquist region. While, in quadrature sampling, as we see in Figure 4.14, this filter is unnecessary. In the digital domain, an integrate and dump filter is usually used to perform this low-pass filtering process. In our application, this integrate and dump filter not only low-pass filters the undesired spectrum in the primary Nyquist

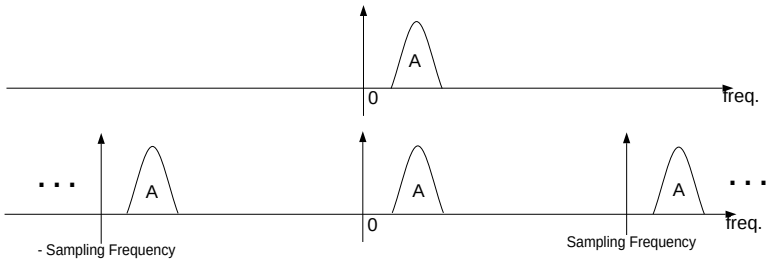

Figure 4.14: Quadrature Sampling

region, but also decreases the sampling rate. This reduces the computational complexity of the following operations. Due to this, it is usually called a decimator filter.

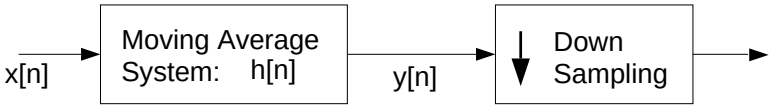

Figure 4.15: Integrate and Dump Filter

The low-pass function in this filter is implemented by a moving average system (Figure 4.15), which can be characterized by the impulse response $h[n]$ as follows:

$$y[n] = \sum_{k=1}^{M} x[n-k] = x[n] * \sum_{k=1}^{M} \delta[n-k] = x[n] * h[n] \tag{4.7}$$

where

$$h[n] = \sum_{k=1}^{M} \delta[n-k] \tag{4.8}$$

The frequency response of $h[n]$ is given by:

$$\begin{aligned} H(e^{j\omega}) &= \mathcal{F}[h[n]] = \int_{-\infty}^{+\infty} h[n] \cdot e^{j\omega} d\omega \\ &= \sum_{k=1}^{M} e^{-j\omega k} = \frac{e^{-j\omega}(1 - e^{-j\omega M})}{1 - e^{-j\omega}} \end{aligned}$$

$$
\begin{aligned}
&= e^{-j\omega} \cdot e^{-j\omega\frac{M}{2}} \frac{e^{j\omega\frac{M}{2}} - e^{-j\omega\frac{M}{2}}}{e^{\frac{-j\omega}{2}}(e^{j\frac{\omega}{2}} - e^{-j\frac{\omega}{2}})} \\
&= e^{-j\omega\frac{M+1}{2}} \cdot \frac{sin(\frac{\omega M}{2})}{sin(\frac{\omega}{2})}
\end{aligned} \tag{4.9}
$$

A plot of the magnitude response of the moving average system with $M = 11$ is shown in Figure 4.16 as an example. As we see, $H(e^{j\omega})$ is periodic with a normalized period of 2π as the sampling process in the discrete-time system introduces the spectrum aliasing.

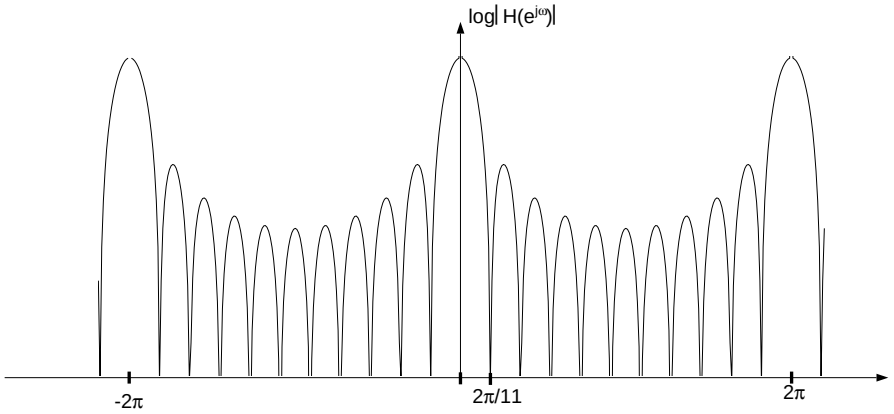

Figure 4.16: The Magnitude Response of the Moving Average System

In our application, the receiver outputs a BPSK/QPSK modulated signal, centered at 20 MHz with a bandwidth of 4 MHz. As the PN matched filter requires two samples per chip, the decimation ratio is given by

$$
M = \frac{sample\ rate}{2 \times\ single\ \ side\ \ bandwidth} = \frac{45.056MHz}{2 \times 2MHz} = 11 \tag{4.10}
$$

The undesired spectrum is 20MHz away from the zero frequency and its frequency, normalized by sampling rate RXIFCLK of 45.056MHz, is

$$
\frac{20MHz}{45.056MHz} = 0.44 \tag{4.11}
$$

We find in the graph (Appendix A) that the alias is attenuated more than 20 dB. Theoretically, we want to choose an IF frequency as high as possible compared to the sampling rate so that the undesired alias will appear far away from the zero frequency. This allows the low-pass filter to, as much as possible, eliminate the alias in the primary Nyquist region. We also desire the sampling rate to be as high as possible compared to the incoming signal bandwidth in order to maximize the decimation ratio M and to increase filter attenuation. Practically, the sampling rate is limited by the highest operating frequency of the Z2000 Evaluation Board. The signal bandwidth should be traded off with the IF center frequency to provide the minimum spectrum aliasing after the integrate and dump filter. The signal aliasing distortion can not be eliminated or decreased by increasing signal power as the distortion is proportional to the signal strength, and it can only be reduced by careful early system planning considering the system performance and capacity.

The output magnitude of the accumulation process depends on the number of samples accumulated in the sum operation, which is determined by the decimation ratio M. We show this in the following example.

In this project, the analog input to the ADC is 0.8 V_{pp} with 1V DC bias voltage added by the Z2000 Evaluation Board. The reference voltage of the ADC is 2V (unipolar positive operation with range from 0 to 2V) and the ADC uses two's complement representation. Thus, the maximum digital output of the ADC is $\frac{0.8V}{2V} \times 128\ LRL = 51\ LRL$, where LRL stands for lowest resolvable signal level after the quantization. Now, the digitized signal is multiplied by $\pm 128\ \ LRL$ NCO signal, summed over 11 samples and then divided by 256 (the 8 least significant bits are discarded). The peak magnitude of the output from the integrate and dump filter is given by:

$$\frac{51\ LRL \times 128 \times 11}{256} = 280\ LRL \tag{4.12}$$

The PN matched filter allows only a 3 bit offset two's complement word with an effective range of $\pm 3.5\ LRL$ (from $-3.5\ LRL$ to $+3.5\ LRL$). The output must be scaled for the PN matched filter. In the Z2000, this is referred to as a "*viewport*" and it is effectively a gain

controller.

The " viewport" should be set to 07, then the PN matched filter input is given by

$$\frac{280}{2^7} = 2.2\ LRL < 3.5\ LRL \tag{4.13}$$

4.3.4 PN Matched Filter

The PN matched filter is designed to operate with two signal samples per chip to allow the system to sample the incoming signal asynchronously with respect to the PN spread rate. This requires that the integrate and dump filter output rate must be twice the PN chip rate. Also, to avoid spectrum aliasing, the output rate of the integrate and dump filter must be less than or equal to one half the frequency of RXIFCLK. The number of samples summed by the integrate and dump filter is the number of RXIFCLK samples over half a PN chip duration.

A front-end processor operating at the front of the matched filter averages the data stream from the integrate and dump filter over each chip period by adding each incoming sample to the previous one.

The PN matched filter calculates the cross-correlation between the incoming signal and the locally stored PN code coefficients twice per chip. The FEP and PN matched filter is shown in Figure 4.17.

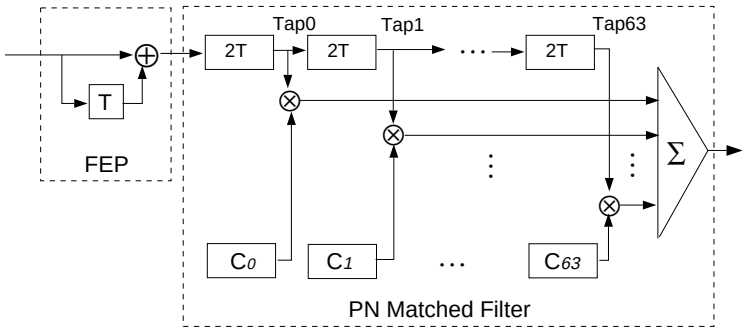

Figure 4.17: The Operation of PN Matched Filter

The matched filter output is

$$output = \sum_{n=0}^{63} Data(n \cdot 2T) * c(n \cdot 2T) \tag{4.14}$$

Note that 1/T is the baseband sampling rate, which is twice the chip rate. In other words, T is half the chip duration. For the proper operation of the PN matched filter, a delay time of 2T (one chip duration) is required for each of the 64 delay lines.

The cross-correlation value is compared against a programmable threshold each symbol period. When the threshold is exceeded, symbol detection is declared and a symbol clock pulse is generated. To increase the probability of detection and reduce the false alarm rate, a "flywheel circuit" is inserted in the symbol tracking processor. This "flywheel circuit" is designed to compare the cross-correlation value one symbol period after the last correctly detected one within a window of $\pm T$. Therefore, in one symbol duration, only three of the PN matched filter outputs are compared against the threshold. One is the sample exactly one symbol period after the last correct detect, plus or minus one sample. If none of these three samples exceeds the threshold, a symbol detect signal will be inserted exactly one symbol period after the last valid detect. The operation of this circuit assumes a constant symbol rate and uses the knowledge of the symbol rate to predict the time of next symbol with a precision of 1 sample (half the chip duration). The PN matched filter is designed to operate at twice the chip rate such that it provides a time accuracy within 0.5 chip duration during the tracking operation.

In the Z2000, the PN matched filter is able to switch between the acquisition mode and the data despreading mode. In other words, after acquisition is declared by using the preamble PN code coefficients to despread the first symbol of the burst (preamble), the filter is configured to automatically switch to another set of PN code coefficients for data code spreading. The output of the PN matched filter is 10 bits and another viewport control is implemented to prevent the input signal into the differential demodulation stage from exceeding $\pm 128LRL$ (8 bits). For PN code lengths of 11 and 63, Table 4.2 shows how to set this viewport register:

Table 4.2: PN Matched Filter Viewport Setting

PN code length M	level at output of PN matched filter (LRL)	view-port register setting	view-port ratio	input level into Diff. Demo. (LRL)
11	38.5	00	1	38.5
63	220.5	01	2	110.25

4.3.5 Differential Decoder

The data is encoded as the phase difference of two consecutive symbols in differential modulation, so the objective of the differential decoder is to detect this phase difference. Any possible constant phase shift added to the whole symbol stream has no effect on the correct decoding. An easy way to compute the phase difference is conjugate multiplication. Let $r(k), r(k-1)$ denote the received baseband signal at time interval k and a previous time interval $k-1$, respectively. To show the operation clearly, we represent these two signals in amplitude-angle form as follows:

$$\begin{aligned} r(k) &= A(k) \cdot e^{j\phi(k)} \\ r(k-1) &= A(k-1) \cdot e^{j\phi(k-1)} \end{aligned} \tag{4.15}$$

where $A(\cdot), \phi(\cdot)$ are the amplitude and angle components, respectively.

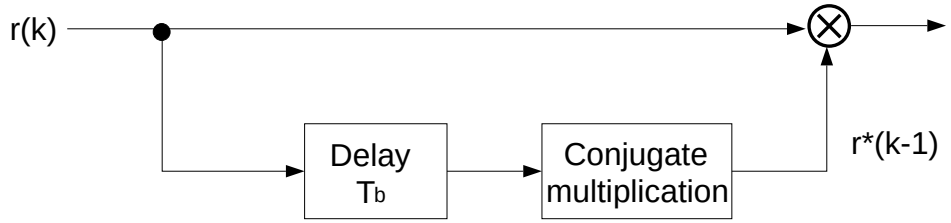

Figure 4.18: The Operation of Conjugate Multiplication

Then by conjugate multiplication:

$$\begin{aligned} c(k) &= r(k) \cdot r^*(k-1) \\ &= [A(k) \cdot e^{j\phi(k)}][A(k-1) \cdot e^{-j\phi(k-1)}] \\ &= A \cdot e^{j[\phi(k)-\phi(k-1)]} \end{aligned} \tag{4.16}$$

where $A = A(k) \cdot A(k-1)$ is a constant. Thus, the phase difference is indicated by the angle of $c(k)$.

For BPSK modulation, the only two possible phase differences are 0^o and 180^o. So, the sign of the real part of $c(k)$ has the information for the phase difference. As $r(k)$ is a real signal, the conjugate operation can be omitted.

$$\begin{aligned} Re[c(k)] &= Acos[\phi(k) - \phi(k-1)] \\ sign\{Re[c(k)]\} &= \begin{cases} 1, & \phi(k) - \phi(k-1) = 0^o \\ -1, & \phi(k) - \phi(k-1) = 180^o \end{cases} \end{aligned} \tag{4.17}$$

For QPSK modulation, there are four possible phase differences: 0^o, 90^o, 180^o and 270^o. We still can decode the data from the phase of $c(k)$. In practice, the Z2000 introduces a $+45^o$ or -45^o phase shift before the complex conjugation as shown in Figure 4.19.

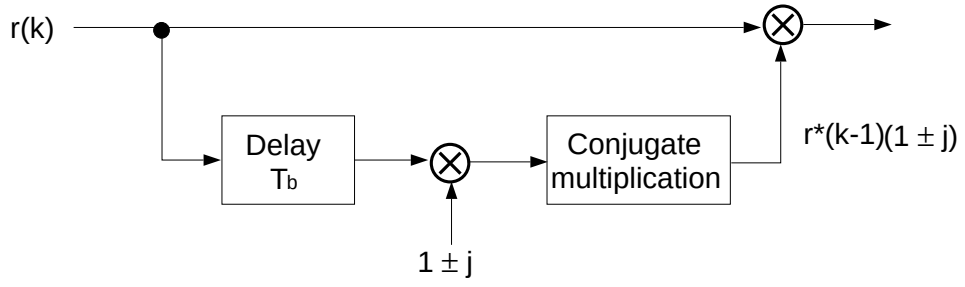

Figure 4.19: The Conjugate Multiplication for DQPSK

The $c(k)$ is thus expressed as

$$c(k) = \sqrt{2}A \cdot e^{j[\phi(k)-\phi(k-1)\pm 45^o]} \tag{4.18}$$

The signal constellation of $c(k)$ is rotated clockwise or counterclockwise, depending on whether $+45^o$ or -45^o is introduced (Figure 4.20). Now, the original data can be easily decoded by the signs of the real and imaginary components of $c(t)$ (dot or cross products, respectively, used in the Z2000 data sheet).

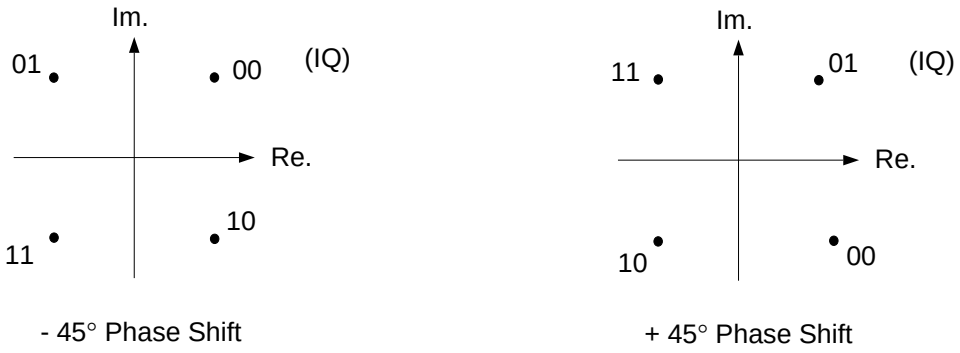

Figure 4.20: The Conjugate Multiplication for DQPSK with Phase Shift

4.4 Simulation of the Z2000

The purpose of this simulation is to verify our understanding of the operation of the Z2000 and to predict the performance of the system. Although the Z2000 is a digital processing device and a computer simulation seems to be easy, not every function and its realization in this chip has been well documented and a software simulation is a convenient way to check our perception.

4.4.1 Simulation Techniques

A simulation model based on MATLAB [Mat 92] is described as shown in Figure 4.21. MATLAB was chosen due to its rich collection of functions.

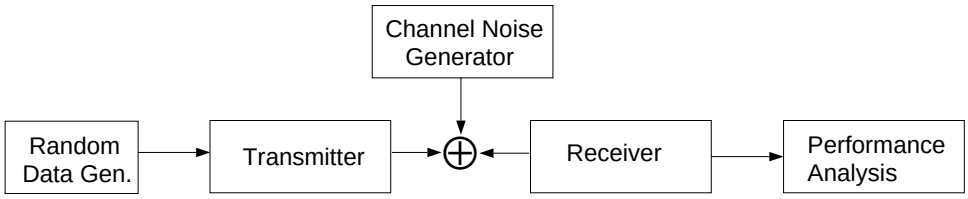

Figure 4.21: Simulation Model of the Z2000

Transmitter and the receiver are each partitioned into several functional blocks according to the operation of the Z2000. The signals are represented in complex envelope format and are sequentially processed from input to output. The signal quantization process is performed before and after each stage of simulation with the quantization bit number set according to the specification of the Z2000.

4.4.2 The Simulation Model

This simulation model is based on a baseband signal simulation. We do not include the BPSK/QPSK modulation and demodulation processes as the bandpass waveform can be sufficiently represented by baseband waveform samples. If later on, we need to study the quantization effect or other issues of the modulation and demodulation processes, we can add these blocks to our simulation models. Since the code is written in block format, the integration will not take great effort as long as the input and output interfacing is matched. The block diagram is shown in Figure 4.22. It is almost the same as the Z2000 internal block diagram except that the modulation and demodulation blocks are removed. The more detailed descriptions of the simulation environment, the realization of these blocks and also the commented code written in MATLAB are included in Appendix B.

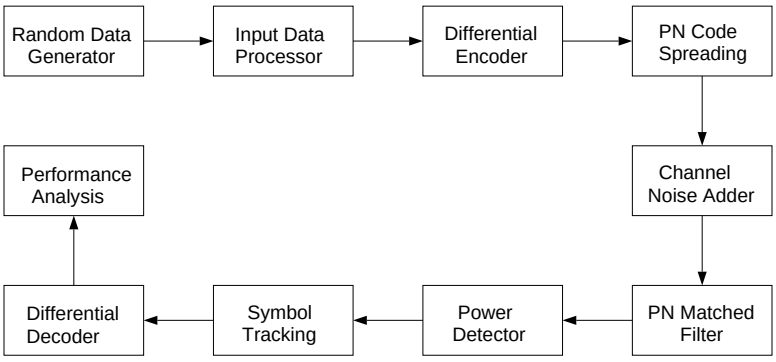

Figure 4.22: Simulation Block Diagram

In the current stage of simulation, we have not included a model for the frequency tracking loop. Thus, we assume that there is no phase offset between the transmitted and received baseband signal.

4.4.3 Calculation of E_b/N_o and Noise Variance

The energy per bit E_b is calculated as the product of average signal power P and bit duration T_b. As the channel signal is assumed to be a polar waveform with an amplitude of 1 volt, the average signal power is thus 1 watt. Hence, the energy per bit is expressed as:

$$E_b = P \cdot T_b = \frac{1}{R_b} \tag{4.19}$$

where R_b is the bit rate.

The noise power in the channel is the product of the two-side noise figure $N_o/2$ and the bandwidth of chip sequence (we only consider the positive frequency domain):

$$P_{noise} = \frac{N_o}{2} \cdot BW_{chip} = \frac{N_o}{2} \cdot (NR_b) \tag{4.20}$$

where N is the processing gain. From Equation 4.19 and 4.20, we find the relationship between noise power P_{noise} and E_b/N_o as follows:

$$P_{noise} = \frac{N_o}{E_b}\frac{NR_b}{2} \cdot E_b = \frac{1}{E_b/N_o}\frac{NR_b}{2} \cdot \frac{1}{R_b} = \frac{N}{2} \cdot \frac{1}{E_b/N_o} \tag{4.21}$$

Actually, the noise power equals the noise variance as the mean value of the white noise is zero.

4.4.4 Simulation Results and Analysis

Simulations were carried out to study the bit error rate (BER) performance of the Z2000. The quantization effect and the symbol tracking performance are obtained from the simulation results.

As we know, this spread spectrum system employs the Differential Phase Shift Keying (DPSK) technique. The BER performance of this spread spectrum system is the same as that of DPSK without spread spectrum. This conclusion comes from Section 3.1. Thus in our non-coherent DPSK spread spectrum simulation, the theoretical BER performance is given by:

$$BER = P_e = \frac{1}{2}e^{\frac{-E_b}{N_o}} \tag{4.22}$$

A plot of Equation 4.22 is shown in Figure 4.23. The simulations were conducted with E_b/N_o from -4 dB to 9 dB with a step size of 1 dB. White Gaussian noise is added to the each baseband sample with a noise variance calculated by Equation 4.20. The 11-bit Barker code is applied to each simulation.

In the first simulation, the quantization processes were removed before and after each simulation block and the symbol tracking process suggested by Zilog was disabled. The symbol output is the power detector output value when the local PN code and received PN code is aligned. In other words, detections happen at the symbol boundaries. For each simulation, 5000 data bits for E_b/N_o below or equal 7 dB and 10000 data bits for E_b/N_o above 7 dB (to improve the simulation accuracy when the BER is considerably small) are generated by the random data generator. This whole process iterates 10 times for each

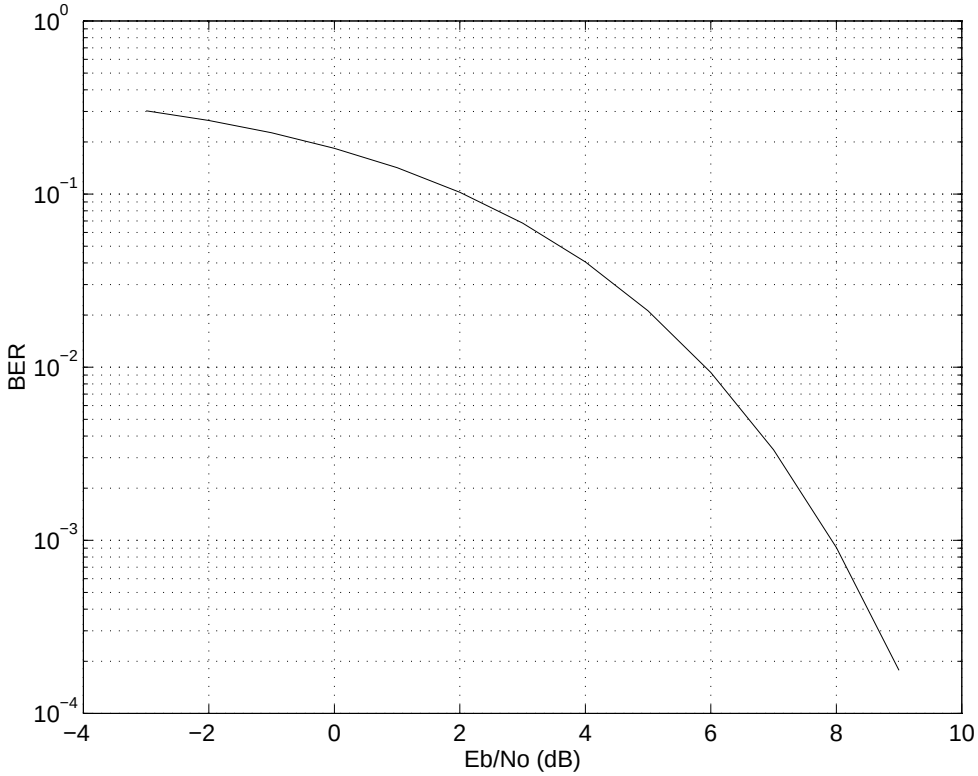

Figure 4.23: Theoretical BER Performance of Non-coherent DPSK

E_b/N_o value and the BER is averaged over these 10 iterations. Figure 4.24 shows this result. Careful observation shows that the BER performance is close to the theoretical one. Moreover, if we reverse the channel signal's polarity, the BER performance remains the same because the DPSK demodulation operation is not affected by any constant phase shift to the received signal. All these results imply that the simulation model is created correctly.

A second simulation was designed to study the quantization effect of this digital processing chip. The quantization is realized in simulation by truncating or rounding the value to its closest quantization level. The number of the quantization levels is determined by the bit number of each signal in the Z2000. The biggest quantization effect is thought to be from quantizing the PN matched filter input signal which is rounded to 3 bits in the chip. The same number of data bits and iterations were carried out as in the first series of simulations. Figure 4.24 shows this simulation result. Interestingly, the resulting BER performance is also close to the theoretical one. This means that the quantization effect can be ignored based on the current simulation model. One explanation is that white Gaussian noise, even at E_b/N_o equal to 9 dB, is much greater than the quantization noise so that the quantization noise can be neglected.

A third simulation was done to study the performance of the symbol tracking scheme suggested by Zilog. The simulation is carried out without quantization and the flywheel tracking strategy is applied in the symbol tracking block. The number of data bits and iterations are the same as before. As we know, the flywheel strategy is based on the knowledge of a previous correct detection. While the Zilog data sheet does not have a description of how the acquisition is performed, we assume an ideal situation where the first detection always happens at the correct time. By doing this, we study the flywheel strategy's impact only on data reception. This simulation shows that the BER performance is greatly affected by the symbol tracking process. Figure 4.25 shows that the flywheel tracking process will degrade the system BER performance by at least 3 dB when E_b/N_o is large.

The threshold settings will greatly also affect the BER performance. Figures 4.26 and

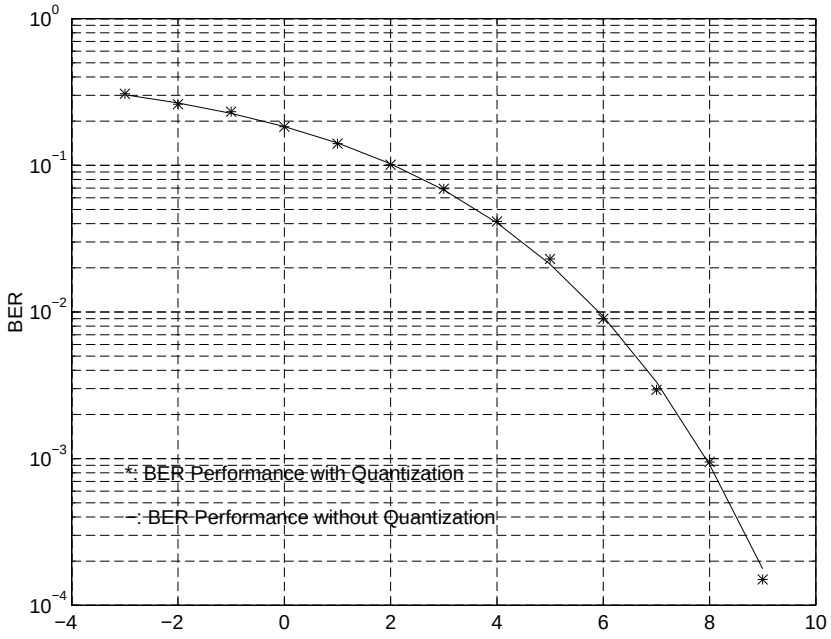

Figure 4.24: Quantization Effect

4.27 show how the BER performance is affected. If the white noise is small enough, the BER decreases when the threshold increases. This means that when a higher threshold is set, the detections will be less likely to exceed this threshold. Thus, the flywheel strategy is more likely to use the previous correct detection timing information instead of the largest noisy signal's timing to track the symbol.

We should remember that we assumed that the first detection is always at the right time. If this condition is not satisfied, we should expect increased deterioration in the BER performance. Another situation we need to consider is that we assume the ideal frequency and phase tracking in this simulation. However, this is not true for the real situation. The flywheel strategy is designed for the real receiver when there are phase and frequency offsets between transmitter and receiver. In this case, we expect the flywheel strategy will help to correct symbol tracking.

4.4.5 Future Work

As stated previously, the simulation was done to verify our understanding of Z2000's operation and it only considers for a limited system configuration. Some additional functions should be added to provide a more sophisticated performance analysis.

1. QPSK modulation. The data stream is divided into an I and Q channels at the Input Processor and converged back to one stream at the Output Processor. This will affect the differential encoder and decoder. This function and the realization of DQPSK has already been discussed in Section 4.2 and 4.3.

2. Preamble acquisition. In the current simulation model, we assume there is no phase difference between the carrier of transmitter and the local NCO frequency. In other words, it is a coherent system. If this assumption is not valid, then one preamble symbol is required to precede the information symbol for the PN matched filter to perform acquisition. As the PN code for this preamble could be different from the PN code used for information symbols, the PN matched filter is required to automatically switch to another set of PN code immediately after the acquisition is declared. If this is the case, then an automatic

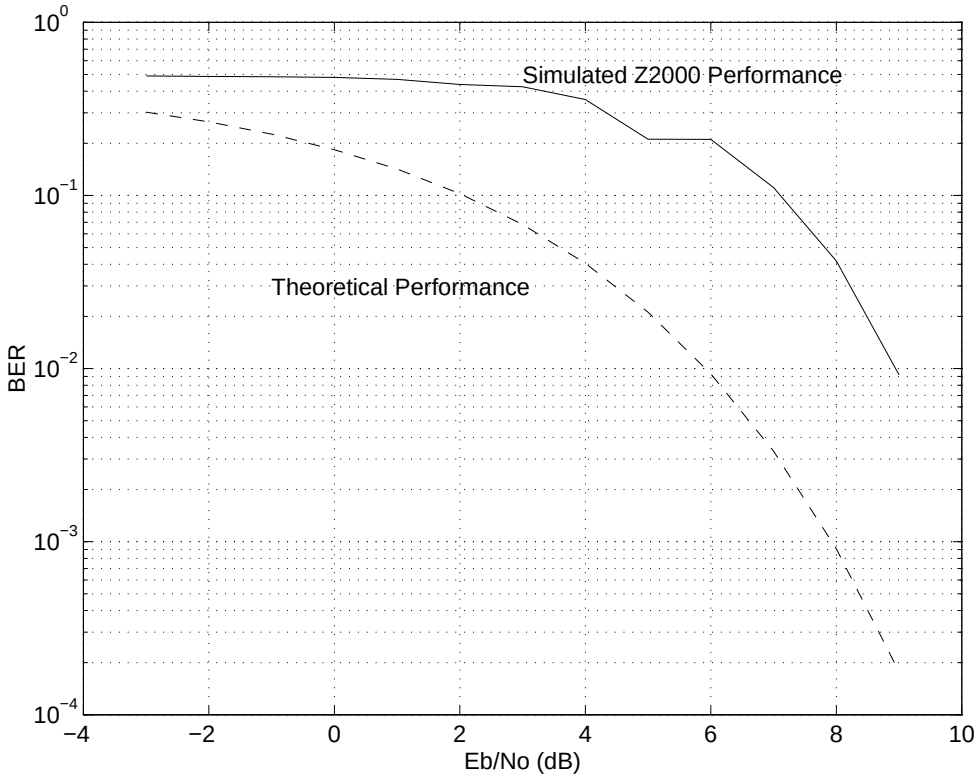

Figure 4.25: BER Performance with Symbol Tracking (Threshold = 5)

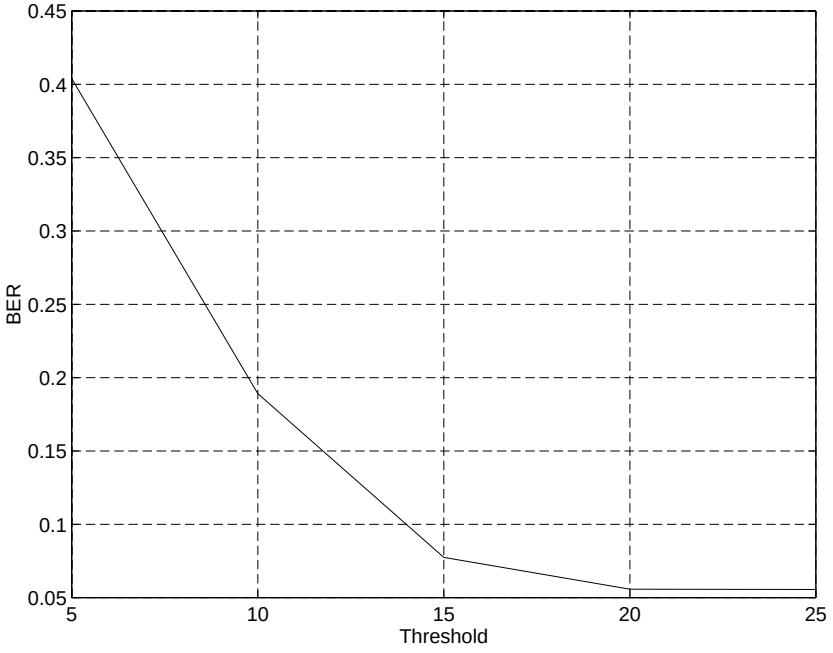

Figure 4.26: BER Performance with Symbol Tracking ($E_b/N_o = 4$ dB)

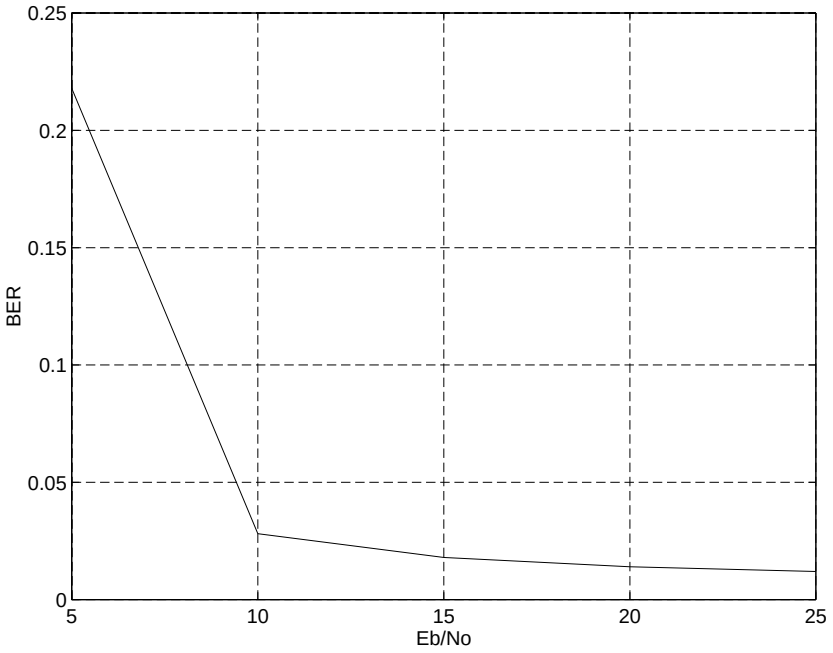

Figure 4.27: BER Performance with Symbol Tracking ($E_b/N_o = 6$ dB)

frequency control (AFC) loop should be added to perform frequency tracking to maintain synchronization.

3. Missed-detection counter. This counter records the number of times that a threshold is not exceeded at the expected time per burst. The threshold for this counter can be set in the control registers. If this threshold is exceeded during a burst, it means that a PN code misalignment occurs or the AFC is tracking the incorrect frequency. In this case, the receiver should be reset to acquisition mode.

4. Automatic frequency control. The AFC loop should synthesize the NCO output to the received carrier when the initial frequency uncertainty is within the pull-in range of the AFC loop. Only a small phase shift, relative to the phase difference carrying information (180^o for BPSK and 90^o for QPSK), is allowed if the AFC is tracking the correct frequency.

Chapter 5

RF Hardware Design

5.1 RF Module Description

The RF module is a half-duplex radio transceiver designed to interface with the Zilog Z2000 Evaluation Board. The RF frequency is 915 MHz which is in the middle of the 902-928 MHZ ISM band. An additional stage of up-conversion can be implemented to move the 915 MHz spectrum to a higher frequency, such as the 2.4 GHz and 5.7 GHz ISM bands. A corresponding down-converter can be employed in the receiver. The RF module, including a transmitter and a receiver, is shown in Figure 5.1.

The transmitter is of conventional design. The transmitter input is the baseband spread signals TXI and TXQ from the Zilog Z2000 Evaluation Board. These digital signals are first low-pass filtered to reduce the modulation side lobes and then BPSK/QPSK modulated onto a 170 MHz carrier generated by the Colpitts crystal oscillator [Rar 83]. After a current controlled variable attenuator and an amplifier, the modulated signal is fed into a 170MHz helical bandpass filter for the further elimination of side lobes. Another amplifier stage increases the power level of this band-limited signal to 0 dBm. This signal is then mixed with a 745 MHz carrier from a frequency synthesizer. The sum product of 915 MHz passes through a bandpass filter centered at 915 MHz to an RF amplifier and then onto antenna

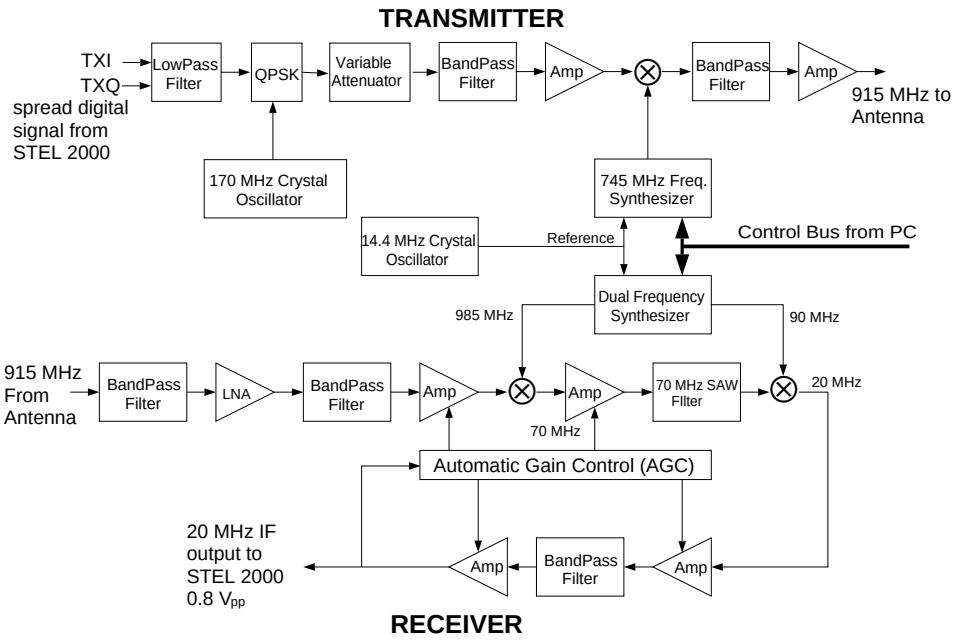

Figure 5.1: RF Module Architecture

for transmission. The frequency synthesizer can be controlled to generate a signal at other frequencies to up-convert the 170 MHz spectrum to any frequency in the ISM band as long as the bandpass filters and the RF amplifiers have the appropriate bandwidth. Note that the signal bandwidth should not exceed the ISM band boundaries.

The Grayson receiver was manufactured by Grayson Electronics in Lynchburg, VA. It was designed for a similar applications and it is being used for our wireless LAN project. It uses two stages of down-conversion, from 915 MHz to 70 MHz and then to 20 MHz. A dual frequency synthesizer is implemented in the receiver to generate two local oscillator frequencies, one at 985 MHz and another at 90MHz. The 20 MHz IF output is fed into the automatic gain control (AGC) circuit to generate the gain control signals for the four amplifiers, which are located before and after the two mixers. This AGC is adjusted such that the receiver outputs an IF level of 0.8 V_{pp}. This output can be fed directly into the Z2000 Evaluation Board in-phase channel input for direct IF sampling. For QPSK

modulation, the 20 MHz modulated signal can be 90^o split into two separate channels (in-phase and quadrature-phase) and then fed into two ADCs to perform the quadrature sampling.

The receiver output is a 20 MHz BPSK/QPSK modulated spread signal with a bandwidth of 4 MHz. The receiver bandwidth puts an upper limit for the system PN chip rate, which is 2 MHz chip/sec. Accordingly, the maximum data rate is limited to 181.8 Kbits/sec for an 11 bit PN code and 31.7 Kbits/sec for a 63 bit PN code. For QPSK modulation, these data rates are doubled.

A 14.4 MHz Colpitts crystal oscillator with an output level adjustable in the range of 50 to 500 mv provides the reference frequency for both frequency synthesizers.

For each frequency synthesizer, there is a serial programming bus connected to the Z2000 port. This bus is used to program the frequency synthesizer's registers which control the output frequency. The Z2000 port is programmed to send the control data streams in the format specified by each type of frequency synthesizer. Once the registers are set, the frequency synthesizer will have continuous output.

5.2 Power Consideration in the Transmitter Module

The block diagram of the transmitter module is shown in Figure 5.2. The detailed circuit for this module is also shown in Figures 5.3 and 5.4.

The two 7486 XOR gates in each channel perform unipolar-to-polar conversion with an output level at +5.5 dBm. A resistor divider designed for impedance matching introduces 6 dB attenuation. After a three-element Butterworth low-pass filter and a three-resistor resistive attenuation network, the signal is attenuated approximately 10 dB and arrives at the balanced modulator SBL-1 IF port at about - 4.5 dBm. A 170 MHz frequency is generated by the 2N5179 Colpitts crystal oscillator. Its output is -2 dBm. After a MAV-11 amplifier and a low-pass filter to eliminate the undesired harmonic frequencies,

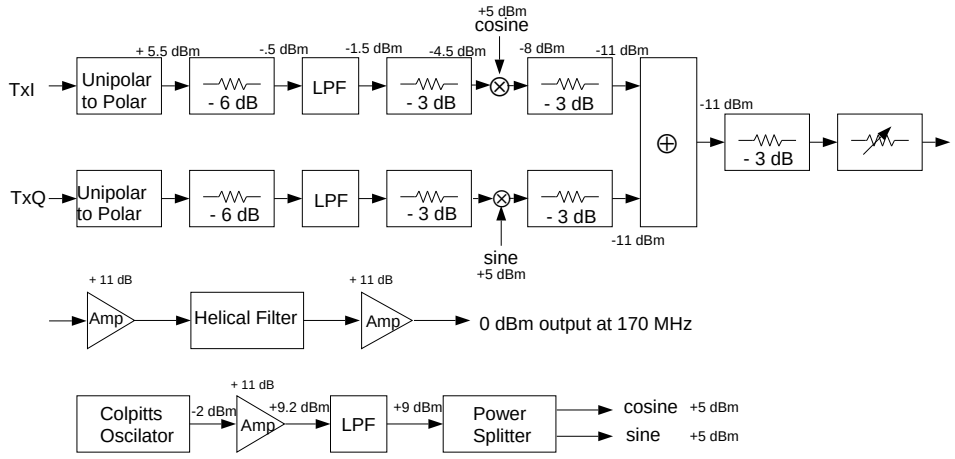

Figure 5.2: Functional Blocks of the Transmitter Module

the carrier power is above +9 dBm and is fed into PSCQ-2-180 for 90^o power splitting. This PSCQ-2-180 introduces 4 dB insertion loss. The two outputs, a +5 dBm cosine and sine signal, drive the balanced modulator SBL-1 which has a 3.5 dB conversion loss when the LO input power level is +5dBm and the frequency is 170 MHz. The RF port of SBL-1 outputs a -8 dBm signal. After a 3 dB resistive attenuator, the I and Q channel signals are combined in the PSC-2-1 combiner. The combiner output is approximately -11 dBm as the combiner's insertion loss is below 0.5 dB. Another SBL-1 device serves as a current-controlled attenuator. Through this variable attenuator and the following two amplifiers, the signal power level is adjusted and amplified to 0 dBm at 170 MHz.

5.3 Hardware Interface

The hardware interface is shown in Figure 5.5. The Transmit Exciter outputs BPSK/QPSK signal at 170 MHz as described in the last section. At the current stage of this project, an crystal controlled up-converter is used to translate the 170 MHz spectrum to 915 MHz. This crystal up-converter employs a crystal oscillator operating at 93.125 MHz and a $\times 8$

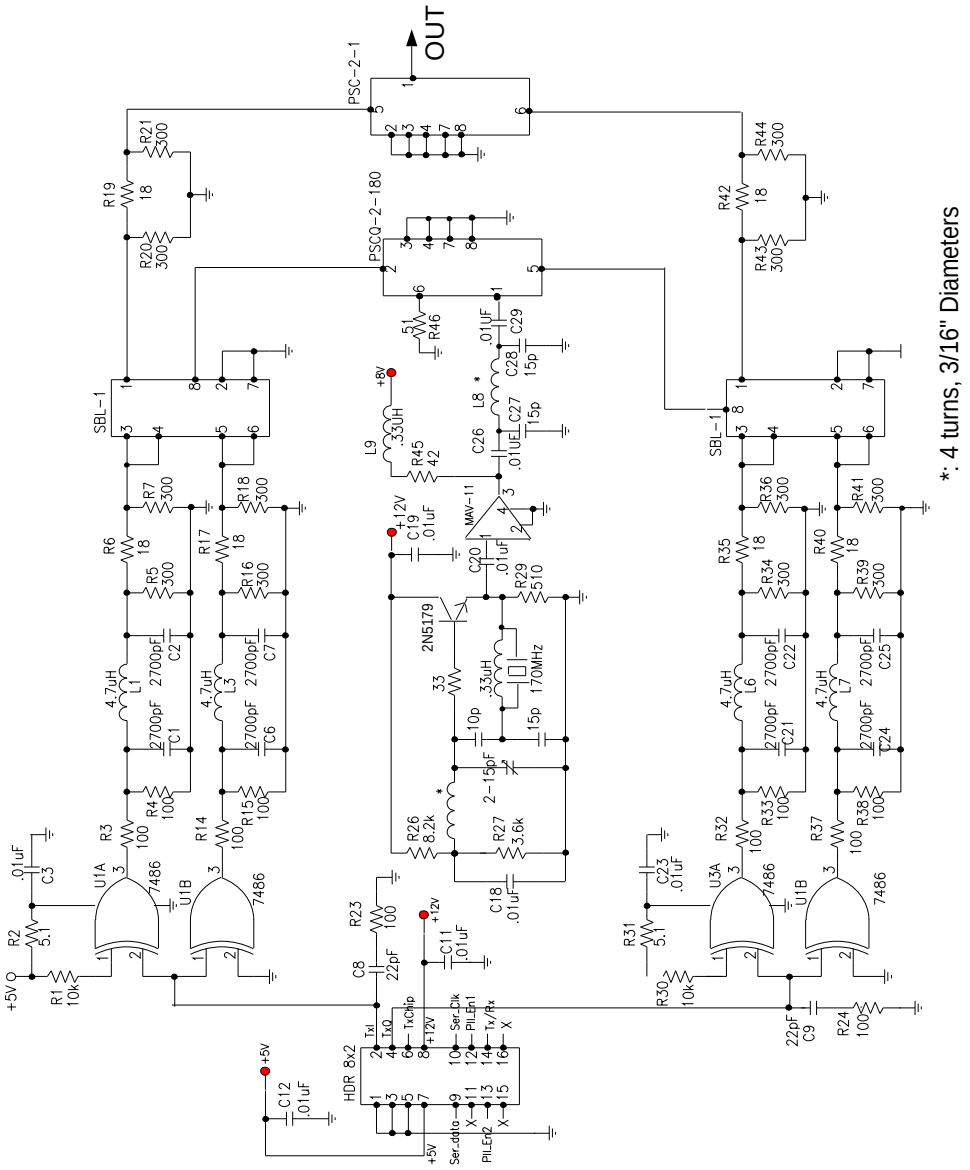

Figure 5.3: Detailed Circuit for Tx Exciter

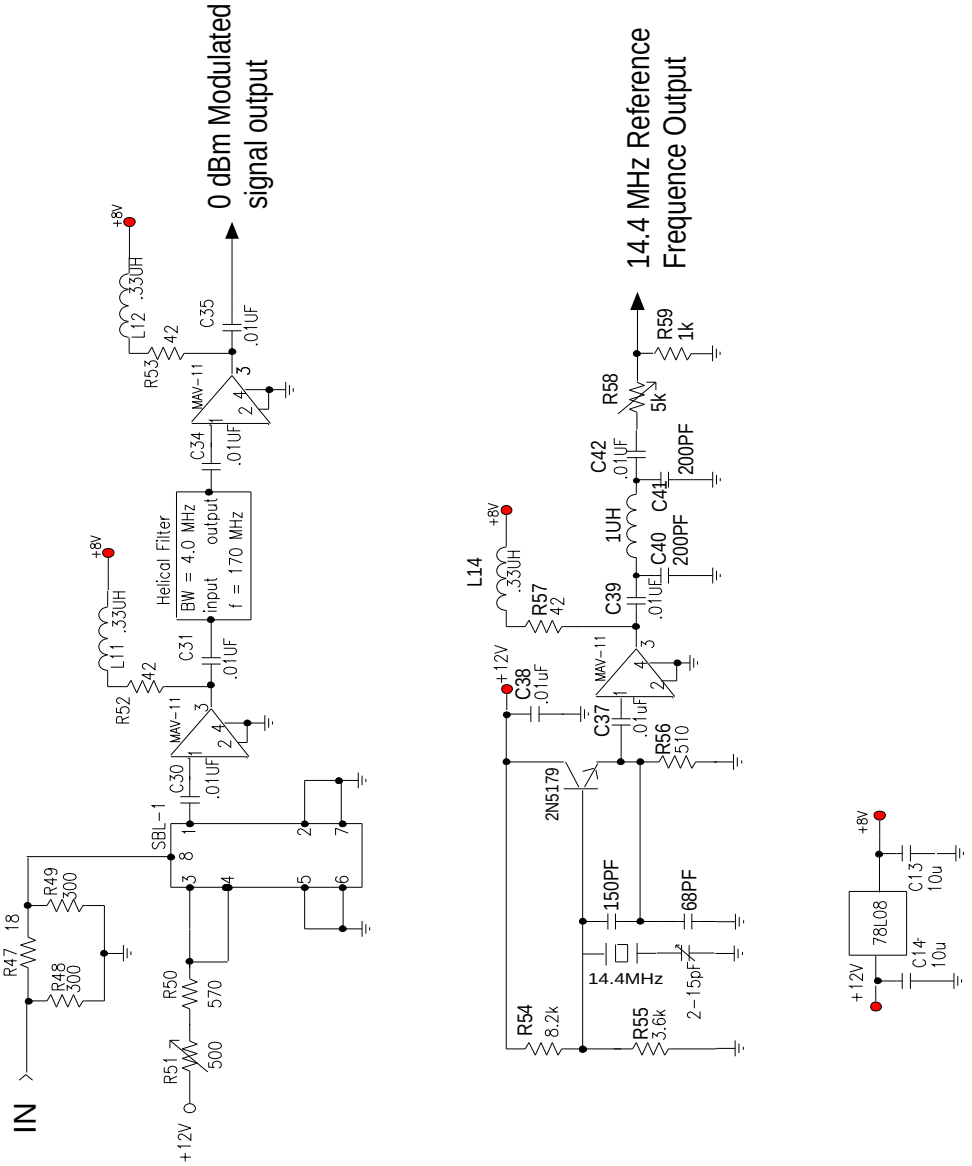

Figure 5.4: Detailed Circuit for Tx Exciter

diode multiplier and filters to generate a 745 MHz local oscillator. This signal is mixed with the 170 MHz modulated signal to generate the sum product for 915 MHz output. Another possible solution for the up-converter is to employ the Motorola MC145190/145191 Frequency Synthesizer Evaluation Kit. The output frequency of this kit is controlled through three control lines. Two of these are multiplexed with the frequency synthesizer in the receiver and the synthesizer is selected by its own enable signal. The up-converter mixer and filters work the same as the crystal up-converter we described before. The benefit of using the Frequency Synthesizer Evaluation Kit is that the frequency is adjustable so the RF spectrum can be moved in the ISM 902-928 MHz band.

All control signals are programmed by the Z2000 Evaluation Board. The exciter routes the control signals and the power supplies from the Z2000 Evaluation Board to the Grayson receiver and the up-converter in the transmitter.

5.4 Circuit Designs

5.4.1 Colpitts Oscillator Design

We use two oscillators in the RF module, one operates with a 7th overtone 170 MHz crystal and another with a fundamental 14.4 MHz crystal. The one operating at 170 MHz is shown in Figure 5.6. The AT-cut crystal may be operated on odd mechanical overtones. In this design, we use a 7th overtone 170 MHz crystal operating in serial resonance. At its serial resonance point, the crystal has minimum impedance and can be replaced by a small value resistor. It provides a path for the oscillator feedback loop [Hay 94]. However, the reactance of the crystal's parallel holder capacitance is small enough compared to the resistor that it will shunt significant current around the crystal path. To avoid this, an inductor, chosen to be resonant with the crystal parallel capacitance at the designed overtone frequency, is used across the crystal. To prevent the crystal from operating at other overtones, the LC network in this oscillator should have a relatively high Q providing enough selectivity to select only the desired overtone. Another consideration is that the Colpitts LC oscillator

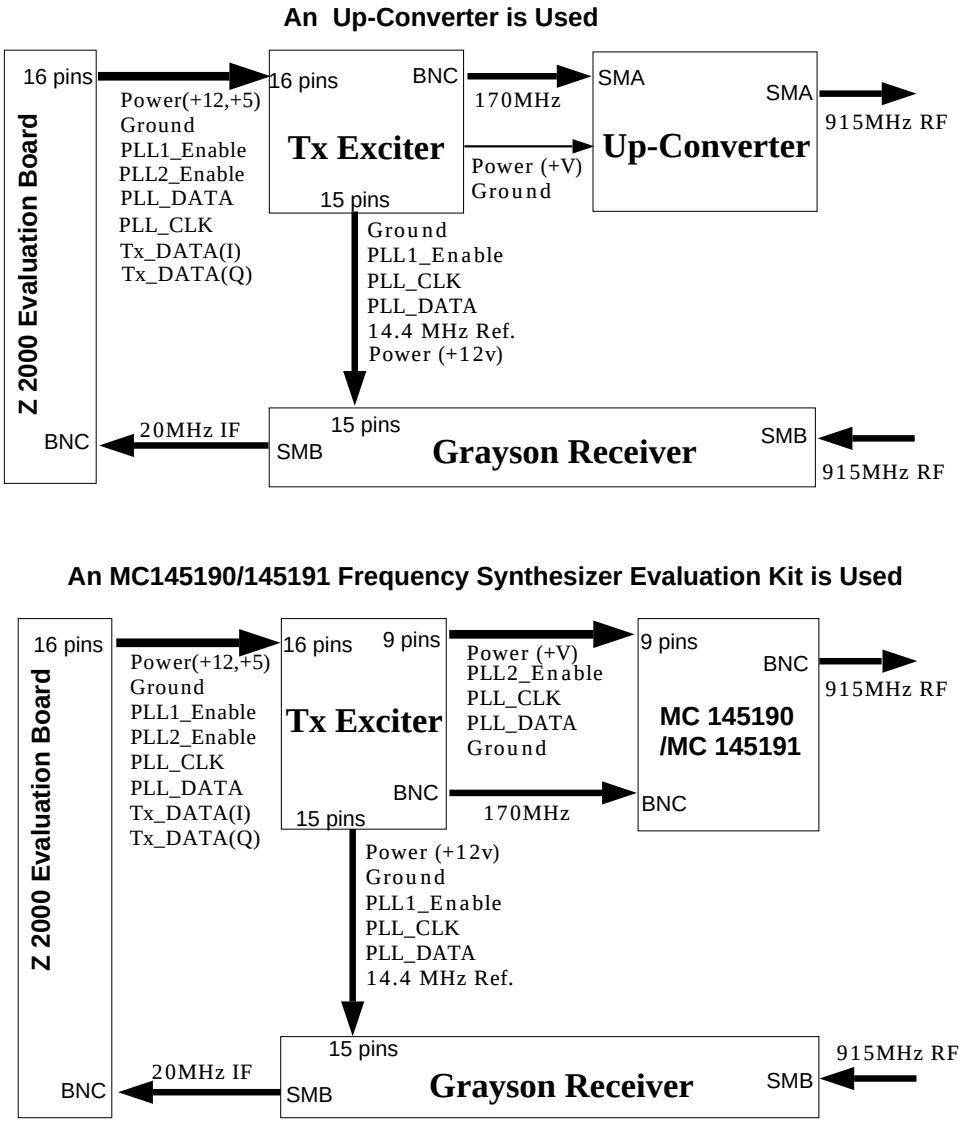

Figure 5.5: Overall System Hardware Interface

should be tuned to the frequency just below the overtone [Rar 83]. The 14.4 MHz crystal oscillator operates similarly, except that a fundamental crystal is used (Figure 5.4).

Let us consider the Colpitts LC oscillator. We assume that the crystal is short-circuited.

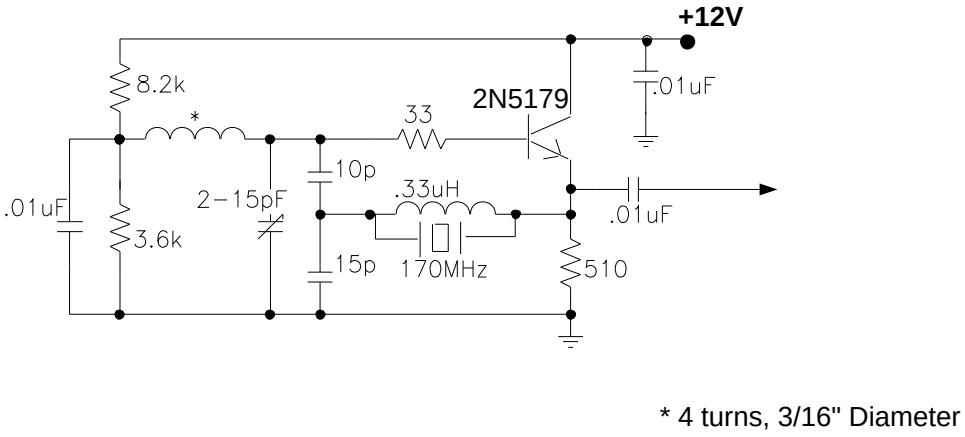

Figure 5.6: Colpitts Crystal Oscillator

The Colpitts LC oscillator can be equivalent to an emitter-follower and a capacitive divider coupling network [Got 71] (Figure 5.7).

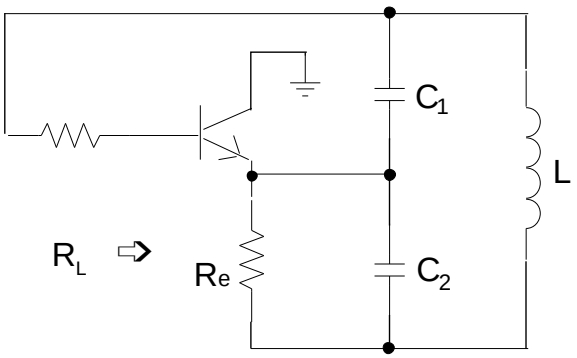

Figure 5.7: The AC Equivalent Colpitts Oscillator

The oscillator load impedance is approximately R_e. The another equivalent circuit with

an ideal transformer is shown in Figure 5.8.

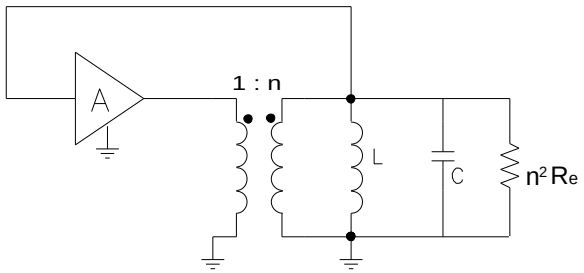

Figure 5.8: Another Equivalent Circuit

Here, the transistor is replaced by amplifier A and the turns ratio of the ideal transformer is given by:

$$n = \frac{C_1 + C_2}{C_1} \tag{5.1}$$

C is the serial capacitance of C_1 and C_2:

$$C = \frac{C_1 \cdot C_2}{C_1 + C_2} \tag{5.2}$$

L and C together contribute to the parallel resonance of this network, thus L is given by:

$$f = \frac{1}{2\pi\sqrt{LC}} \tag{5.3}$$

$$L = \frac{1}{C(2\pi f)^2} \tag{5.4}$$

Usually, in this crystal controlled oscillator, we design the LC network with Q about 10 (the reason will be discussed later). This leads to

$$2\pi f L = \frac{1}{10} R_L \tag{5.5}$$

From Equations 5.2, 5.4 and 5.5, we obtain

$$\frac{1}{2\pi f C_2} = \frac{R_e}{5} \tag{5.6}$$

In our design, we use $C_1 = 10$ pF, $C_2 = 15$ pF. The L is calculated to be 0.14 μH and R_e should be at least 312 Ω.

Although the transistor is an emitter-follower and offers no voltage amplification and no phase shift, the transformer in the equivalent circuit provides the in-phase voltage amplifying. The total phase shift of the loop is zero and the loop gain is given by:

$$loop\ \ gain = G \cdot \frac{C_1 + C_2}{C_1} \tag{5.7}$$

where G is the voltage gain of the common collector transistor amplifier A. G is related to the current amplification parameter β and the emitter resistor R_e. Its value is less than unity. The loop gain is mainly determined by the ratio of C_1 to C_2. Adjusting one of the capacitors will enable the loop gain exceed unity and will excite the oscillator. In practice, this tuning method is replaced by fixing these two capacitors and using a variable capacitor across the inductor (Figure 5.3).

5.4.2 Amplifier Design

We choose the MAV-11 monolithic amplifier for this design. It has a wide bandwidth from DC to 1 GHz with output power up to +17 dBm. The bias configuration is simple and its operation is rather stable. For normal operation, the MAV-11 amplifier requires a DC bias +5.5V at pin 3 and draws about 60 mA current from the power supply. A typical biasing circuit is shown in Figure 5.9. The RF choke increases the AC load of the biasing branch so it does not affect the amplifier's load. The biasing resistor R is calculated as follows:

$$R = \frac{V - 5.5}{60\ mA} \tag{5.8}$$

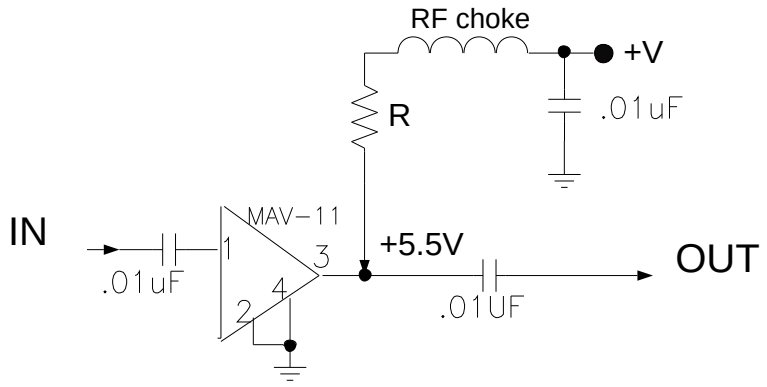

Figure 5.9: MAV-11 Amplifier

To reduce the power dissipated by the biasing resistor R, we use a +8V supply instead of +12V, then R is calculated to be 42 Ω. The power consumption is

$$P = (60\ \ mA)^2 \times 42\ \ \Omega = 0.15W = 150mW \tag{5.9}$$

A conventional resistor with 1/4 W power specification satisfies this power requirement.

5.5 Experimental Results

In this section, we show the outputs from an HP 8590 Spectrum Analyzer. These results are obtained when Utilicom's 63-bit M-sequence code is used as the PN spread code and the signal is BPSK modulated. An on-board 16 MHz crystal oscillator provides the receiver IF clock signal (RXIFCLK). The register at address 41_H defining the number of TXIFCLK cycles per chip is set to 7. Thus, from the Z2000 User's Manual, the PN chip rate is given by:

$$\frac{16\ \ MHz}{7+1} = 2MHz \tag{5.10}$$

The first graph (Figure 5.10) depicts the spectrum of the Tx Exciter input TxI, which

is the digitized spread output signal from the Z2000's PN spreader. It is a polar baseband signal spectrum with the first side lobe 13 dB lower than the main lobe.

Figure 5.11 shows the baseband signal's spectrum after the low-pass filter eliminates the undesired side lobes. The low-pass filter is a three-element Butterworth filter with a cutoff frequency at 2 MHz. The first side lobe is attenuated about 7 dB compared to Figure 5.10.

In Figure 5.12, we show the spectrum of the output from the Colpitts crystal oscillator. This signal is amplified after a MAV-11 amplifier and low-pass filtered by a three-element Butterworth filter. This filter is designed to cutoff at near 170 MHz. The graph shows that the 170 MHz frequency output is fairly clean and the second harmonic is more than 25 dB below the desired 170 MHz.

The last graph, Figure 5.13, presents the BPSK modulated signal right after the two stages of MAV-11 amplification. The output signal level is adjustable through the current-controlled attenuator. If a helical bandpass filter with center frequency at 170 MHz and a cutoff bandwidth of 4 MHz is available later, it should be operated between two stages of amplification as shown in Figure 5.3 and we will expect the side lobes be attenuated to a greater degree.

From these four graphics, we can say that this Tx Exciter works as we designed to perform the BPSK modulation at 170 MHz.

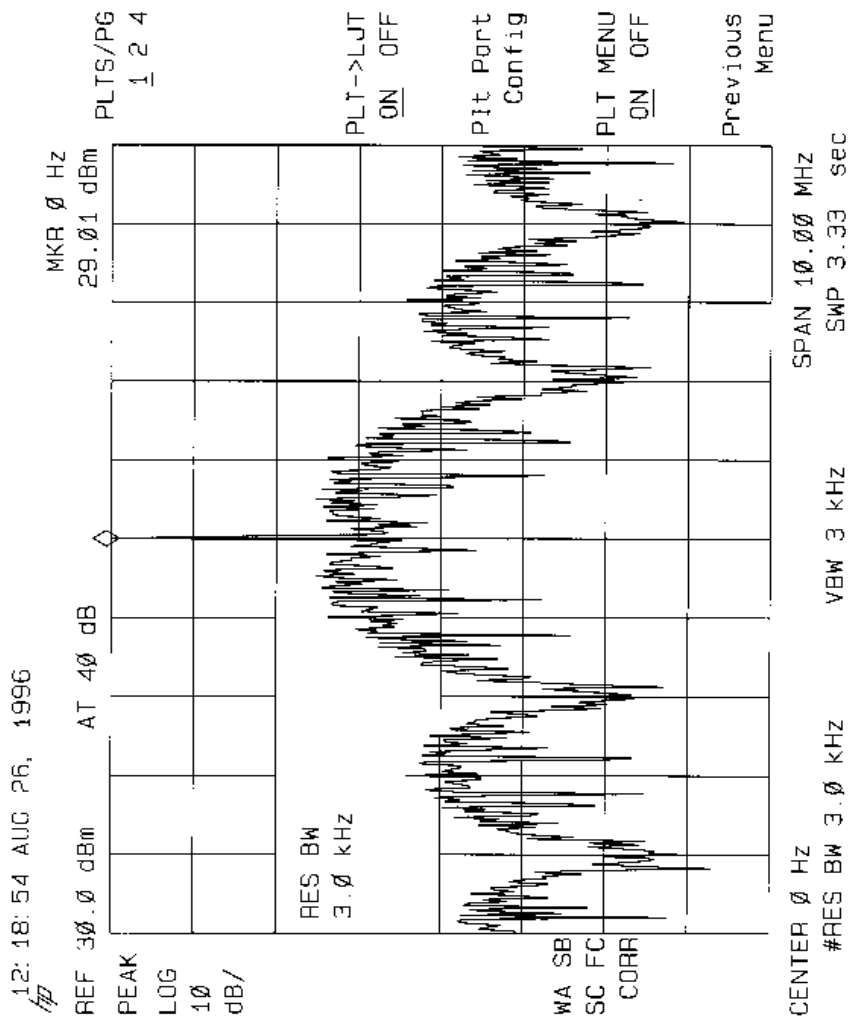

Figure 5.10: Baseband Spread Signal TxI Spectrum

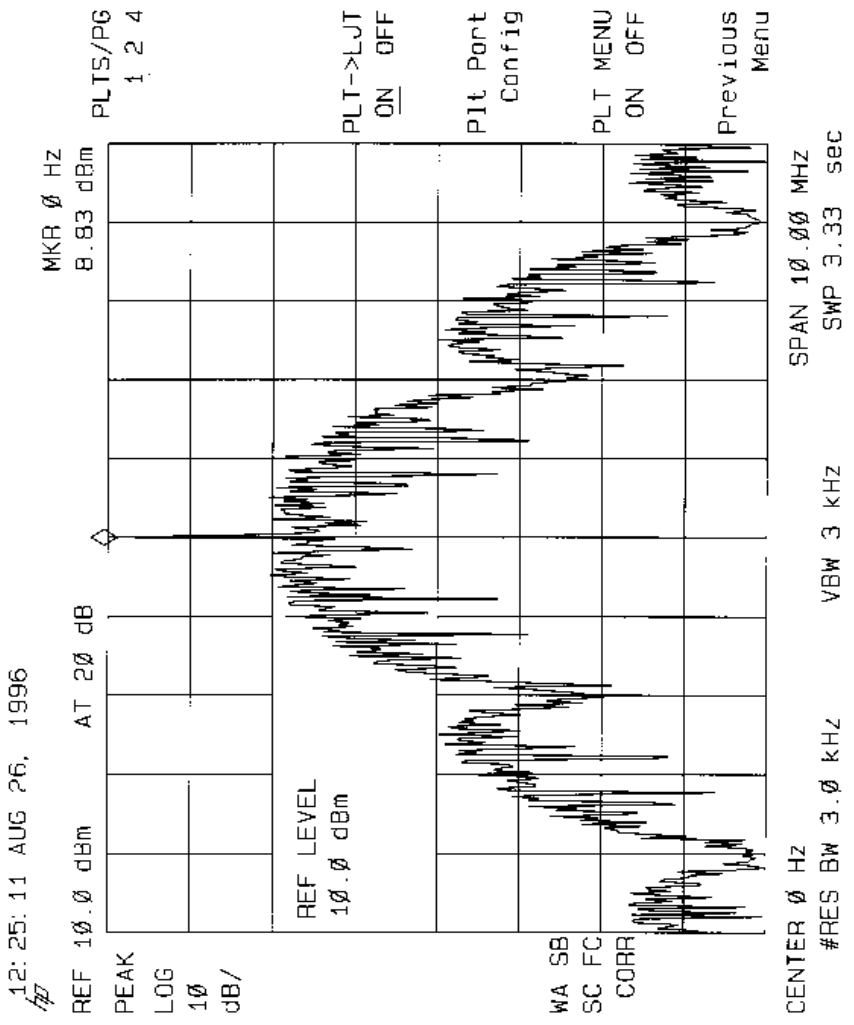

Figure 5.11: Baseband Spread Signal Spectrum After Low-pass Filter

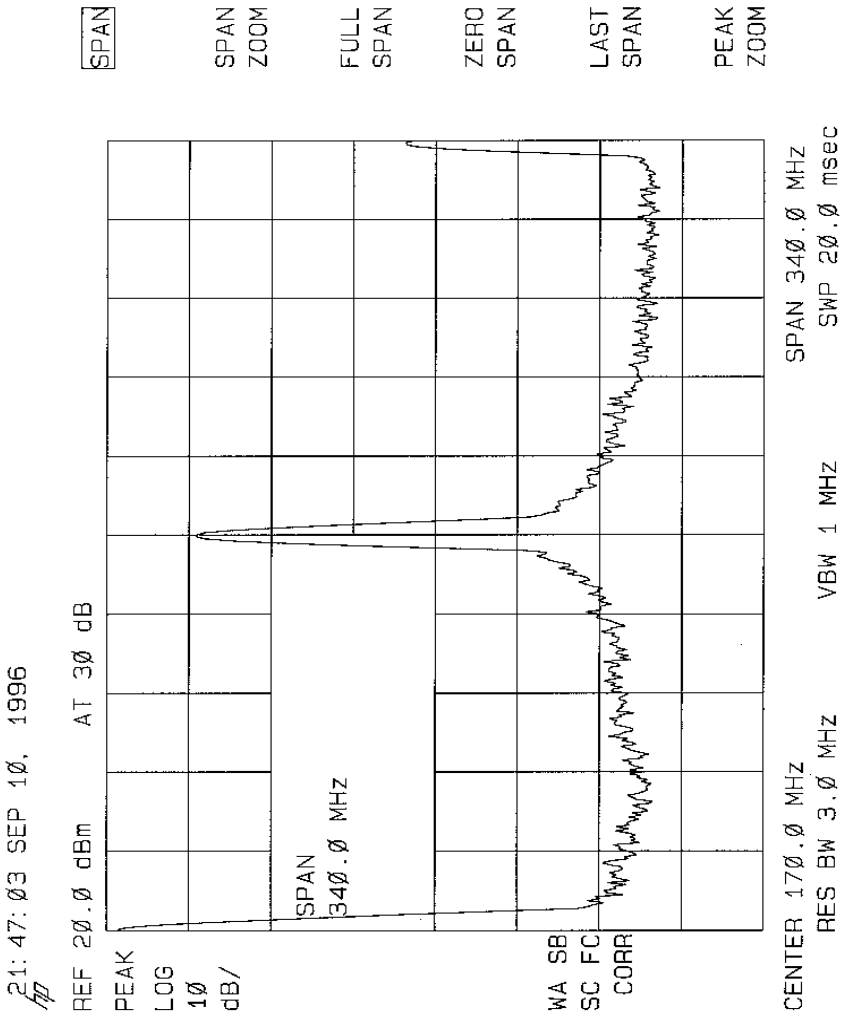

Figure 5.12: Colpitts Crystal Oscillator Output Spectrum

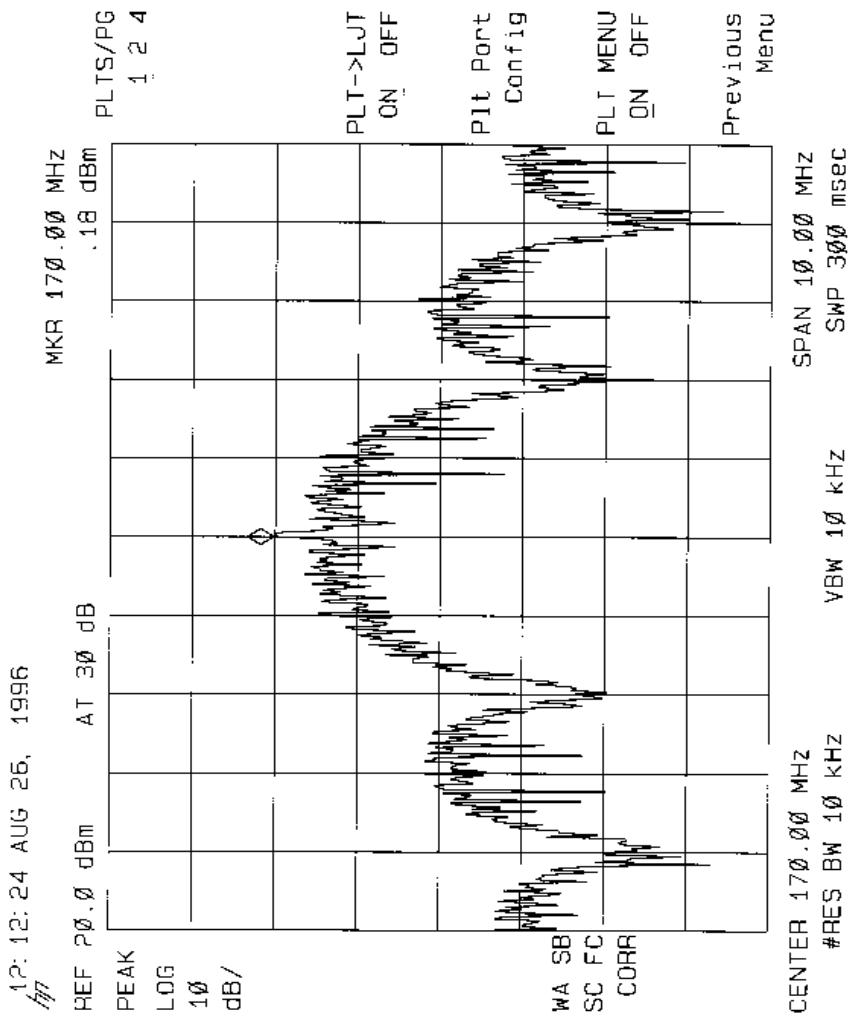

Figure 5.13: Tx Exciter Output Spectrum

Chapter 6

Software Development

6.1 Physical Layer Control Program Description

Figure 6.1 is an illustration of the overall system model on which the control software is based. The Z80182 [Z18 94] supports 128 kbytes of RAM and 64 kbytes of read-only memory (ROM). The RAM space is divided into three portions: the transmitter buffer (TxBuffer), the receiver buffer (RxBuffer) and the control program space. The locations and the sizes of these two buffers are defined by the control program and the buffer spaces are reserved during system initialization. The ROM stores the Zilog-provided Z80182 firmware which performs board level initialization after power-up or reset and then invokes the Z80182 Debug Monitor and a Hayes-compatible AT command Set Processor.

The data flow is shown in Figure 6.1. The MAC layer first sends its data frames to the assigned serial Com port connected to the personal computer (PC) bus. The Z80182 retrieves the data and then passes them to the Z80182 controller. The control program then takes the data and stores them in the transmitter buffer (TxBuffer). At the same time, the Z80182's Enhanced Serial Communications Controller (ESCC) is activated to fetch data from the transmitter buffer and then converts them into a serial bit stream. The ESCC interface then manages the transmission of this bit stream to the Z2000 spread spectrum

processor and to the RF section for on air transmission. Note that it is the MAC layer's responsibility to detect the communication channel's activity. When the MAC layer sends data to the physical layer, there is no delay in physical transmission. The buffer is designed to eliminate the data rate difference between receiving and transmitting. In the receiver direction, the received RF signal is first down-converted, despread and the data are then sent back to Z80182. The bit stream is retrieved by the ESCC channel, converted from a serial to a parallel bit stream, and stored in the receiver buffer (RxBuffer). Z80182 handles the parallel transfer of the data to the ISA bus for the PC host. The data transfer between PC host and Z2000 is fully managed by the Z80182's communication peripherals. The control program is written to configure all the necessary communication interfaces by setting the control registers and coordinating the internal micro-controller system to make it work for a high speed wireless communication system. There is additional control for the RF section to perform the RF module configuration.

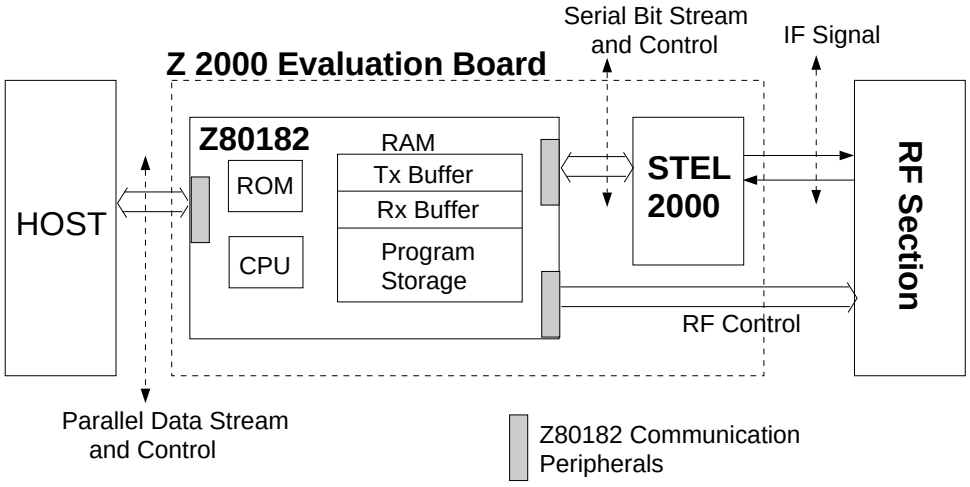

Figure 6.1: Overall Software Environment

The comprehensive description of each peripheral interface is presented in the next section. In Section 6.3, we will briefly discuss the flow chart of Zilog's emulation program.

Recommended future software development is suggested in Section 6.4.

6.2 Peripheral Interfaces

The interface between the Z80182, PC, and Z2000 is depicted in Figure 6.2. The Zilog Z80182 is an intelligent peripheral controller. It provides three 8-bit parallel input/output (I/O) Ports (Ports A, B and C) and two ESCC channels (Channels A and B).

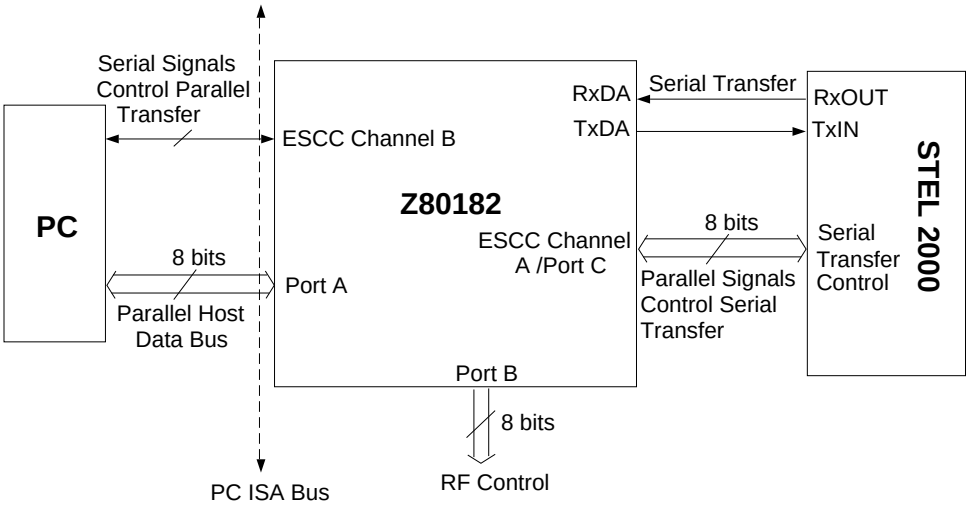

Figure 6.2: Peripheral Interfaces Configured to Plug in PC ISA Bus

PORT A: When the Z2000 Evaluation Board is plugged into the ISA bus of a PC, the software residing in the Z80182 will detect the ISA bus connection and configure this port to be the host data bus for parallel data transfer between the PC and the Z80182.

PORT B: This port is configured as a programmable I/O for the RF interface. In our application, the program sends initialization data for the two frequency synthesizers in the transmitter and the receiver and an additional RF power control signal is sent through this port.

PORT C: This port is multiplexed with ESCC channel A to provide serial data transfer

control. The transmit data are sent serially through the TxDA pin of Z80182 to TxIN pin of the Z2000 and the data are received serially through the RxDA pin of the Z80182 from RxOut pin of the Z2000 (see Figure 6.2). ESCC channel A's signals control the transceiving of these data stream through this parallel port.

ESCC Channel A: Multiplexed with Port C as described in the PORT C description.

ESCC Channel B: control the ISA bus data parallel transfer in the ISA bus mode. Otherwise, it controls the serial data transfer to and from the PC host through RS232/RS422.

6.2.1 Z80182 and PC Interface

The Z80182 communication peripheral is able to support an ISA bus or an RS232/RS422 interface performing the data exchange between the a PC host and the Z80182. The software senses the two different hardware connections and the interface is configured accordingly.

In the ISA bus case, Port A is connected to the ISA Data bus and is assigned as a parallel data port while the ISA control signal is connected to the Z80182.

In the RS232/RS422 case, the PC asynchronous COM port is connected to ESCC Channel B and data is transfered in serial format through the connection. The Channel B is configured to handle the interfacing and the data transfer. It works as a serial-to-parallel, parallel-to-serial converter and controller with data transfer rates up to 115.2 Kbps for RS232 and 4 Mbps for RS422.

In our application, we use the ISA bus to communicate with the PC. Although the Zilog provided emulation program has the capability of using RS232/RS422, our control program only focuses on the ISA bus case because it supports a higher data rate.

The ISA bus and the Z80182 provides a communication path to and from the PC. Based on this, the Z80182 AT command set processor and debug monitor can accept and interpret user commands from the PC. Control can be handed-off between the AT command set processor and the debug monitor by special commands specified in the Z2000 user manual [Z20 94]. A more detailed description of each command is also in the manual. Here we will only discuss how to load the control program to RAM space.

Loading the control program requires that the Zilog terminal emulator program tz.exe be running on the PC host. Using the load command "*L*", tz will download the specified file into RAM. The file must be in Intel Hex format [Z20 94]. After the code is loaded into RAM space, it can be executed using the "*go*" command with starting address. This starting address is determined when the Hex file is linked and is specified in the link command file. (We will discuss this further in Appendix C.)

6.2.2 Z80182 and Z2000 Interface

In this application, the least significant 6 bits of Port C (PC5-PC0) are configured to be an output/input on a bit-by-bit basis and are multiplexed with the control pins from the ESCC channel A. Serial data is communicated through ESCC Channel A TxDA and RxDA pins to TxIN and RxOut pins on the Z2000, respectively. Thus, the control signals are generated by ESCC channel A and sent out through Port C to the Z2000. All of the Z2000 control registers are written through this path. Note that these registers are not readable so the control program must keep a copy of them in the memory if it needs values later.

6.2.3 RF Control Interface

The Z2000 provides a parallel Port B for the RF section control, including frequency synthesizer initialization, power control and half-duplex switch control. In this application, there are two frequency synthesizers designed to work for the up-converter and the down-converter. The control signals should be present on the three input lines for each synthesizer immediately after power up or system reset. These three lines are PLL_Enable, Serial_Data and Serial_Clock. The synthesizer needs to first be enabled by a falling edge on PLL_Enable, then the synthesizer control register words should appear serially on the Serial_Data line and are clocked in by the Serial_Clock signal. The timing requirements for these three input signals and the method of calculating the control register words are detailed in the data sheet of each frequency synthesizer. A program written in C and compiled by the IAR compiler [Hig 94] to perform the above frequency control operations is attached in the

Appendix. If another RF section is required, this program should be modified to output other control signals using other pins of Port B. Also, this program should eventually be integrated into the initialization portion of the overall physical layer control program.

6.3 The Emulation Program's Flow Chart

The flow chart for the emulation program is shown in Figure 6.3.

In the Intro() function, the emulation program first checks the status of the ESCC Control Read registers and assigns the communication control to and from the PC host.

The Reset() function disables the Z2000's transmitter and receiver. It then clears copies of the Z2000 registers in RAM space and finally sets all of the ESCC channel Write Registers to zero.

The ESCC initialization function Init() configures its 15 Write Registers to the Synchronous Data Link Control (SDLC) protocol and then enables the ESCC. Also, it sets the Z2000's registers for normal BPSK operation.

The transmitter and receiver's buffer space is then allocated in the RAM space. The transmitter buffer's content is set according to the user's selection. After that, the receiver buffer is reset to zero and is ready to store incoming data. This is done in the InitTxRxBuffer() function.

After this system initialization, the emulation program is ready for a user's operating command. It supports reading, writing and comparing buffers, starting data transfer from the transmitter buffer, and storing data in the receiver buffer. It can also access the Z2000 registers (reading is through copies of the registers in RAM as stated previously). The operation of the Z2000 is configured as needed by modifying these register words.

6.4 Future Work

This emulation program provides an interface for user access to all of the Z2000's registers. This is unnecessary for our control program. We only need to change some operating

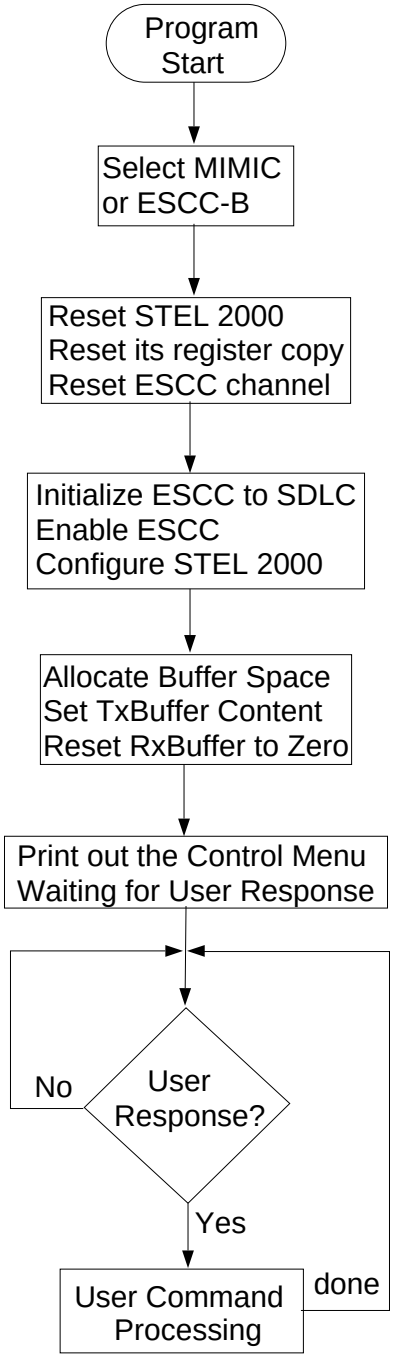

Figure 6.3: Flow Chart for the Emulation Program

parameters according to the MAC layer's configuration frames, such as the PN code or data rate. Other key parameters will be modified accordingly. There are only a few sets of parameters and thus it is much easier for the control program to manage.

The emulation program only deals with transceiving data from and to the buffer. Our next step should be to program the Z80182 to receive/send the data from/to the PC ISA bus. This portion is critical because of the high data rate and the complexity of interface control.

Finally, the RF section control program should be integrated into the initialization portion of the overall control program.

Chapter 7

Conclusions

A physical layer design for an ISM band direct sequence spread spectrum wireless LAN is presented. This physical layer directly interfaces with the MAC layer and uses the 902-928 MHz ISM band for the RF transmission. The Zilog Z2000 Evaluation Board is implemented for the spread spectrum processing. The RF module consists of the Grayson receiver module and our transmitter.

Chapter 2 introduces the concepts of the spread spectrum technology and wireless LAN systems. The operations of two widely used spread spectrum systems, direct sequence and frequency hopping systems, and the acquisition issue are briefly addressed. Several wireless LAN systems and the transmission technologies that wireless LAN employs are discussed. Finally, the interface between the physical layer and the MAC layer are defined.

In Chapter 3, we first present the variable data rate transmission technique and then we investigate how the despreading methods and the PN code affect the acquisition and the BER performance of the system. The PN matched filter despreading method is compared to the sliding correlator in the terms of average acquisition time performance and is found to be suitable for wireless LAN applications. An 11-bit Barker code and a 63-bit M-sequence code are suggested.

In Chapter 4, we talk about the operation of the Zilog Z2000 spread spectrum processor.

The operational theory and the hardware realization of each functional block of the Z2000 are described. A baseband simulation of this chip is conducted to verify our understanding and to predict the system BER performance.

Chapter 5 presents the RF hardware design issues. In this project, a transmitter exciter has been designed and built to perform BPSK/QPSK modulation and move the baseband spread spectrum to IF frequency at 170 MHz. Colpitts crystal oscillators for 170 MHz and 14.4 MHz are designed to provide the carrier frequency for the modulation and the reference frequency for the frequency synthesizers.

Chapter 6 describes the control software development. The operation of the Z2000 is controlled by the Zilog 80182 intelligent peripherals controller. The control interfaces between the PC, 80182 controller and Z2000 chip and an additional RF module control interface are defined in this chapter. Zilog's emulation source code is modified to solve the incompatibility issues of a different compiler system (IAR compiler). The work needed to implement a full physical layer control program is briefly described.

In this thesis, we studied the operation of the Z2000 Evaluation Board and translated its data sheet description into a mathematical representation. A simulation of Z2000, based on the mathematical representation, was built to study the quantization effect and the Zilog's "flywheel" symbol tracking strategy's effect on the overall BER performance. The results show that the quantization effect can be neglected and the "flywheel" strategy degrades the BER performance when there is no phase difference between the transmitter and receiver. A transmitter exciter has been designed and built. The IAR C compiler system was configured to work for the Z80182 processor and the physical layer hardware control software was modified from the Zilog's emulation program to provide reliable data frame transceiving.

References

[Ber 92] D. Bertsekas and R. Gallager, *Data Networks,* Second Edition, Prentice Hall, Inc., 1992.

[Buc 92] R. M. Buchrer and B. D. Woerner, "Teaching Spread Spectrum for Commercial wireless Communications".

[Cou 90] L. W. Couch, *Digital and Analog Communication System,* Macmillan Publishing Company, 1990.

[Adv 94] *Z2000 Spread Spectrum Transceiver Advance Information Specification,* Zilog, Inc., Cambell, California, 1994.

[Dav 78] S. Davidovici and D. L. Schilling, "Minimum Acquisition Time of a PN Sequence", *Proceedings National Telecommunications Conferences,* 1978, pp.35.6.1-35.6.4.

[Dra 94] *Draft Standard IEEE 802.11,* IEEE 1994.

[Feh 95] Kamilo Feher, *Wireless Digital Communications-Modulation & Spread Spectrum Applications,* Prentice Hall, Inc., 1995.

[Got 71] Irving M. Gottlieb, *Understanding Oscillators,* Howard W. Sams & Co., Inc., Indianapolis, Indiana, 1971.

[Hig 94] *High Performance Microprocessor Development Tools,* IAR Systems Software, Inc., San Francisco, California, September, 1994.

[Hay 94] Wes Hayward, *Introduction to Radio Frequency Design,* The American Radio Relay League, Inc., 1994.

[Hol 77] Jack K. Holmes and Chang C. Chen, "Acquisition Time Performance of PN Spread Spectrum Systems", *IEEE Transactions on Communications,* Vol. Com-25, No. 8, August 1977, pp.778-784.

References

[Hop 77] Phillip M. Hopkins, "A Unified Analysis of Pseudo-noise Synchronization by Envelope Correlation", *IEEE Transactions on Communications,* Vol. Com-25, No. 8, August 1977, pp.770-778.

[Jib 91] Waseem W.S. Jibrail and Abdul-razak J. Houmadi, "Acquisition of Direct Sequence Spread Spectrum Signals Using Sliding Correlators", *International Journal of Electronics,* Vol. 71, No. 5, 1991, pp.733-743.

[Mat 92] *MATLAB User's Guide for Unix Station,* The MathWorks, Inc., Natick, Massachusetts, August, 1992.

[Rao 88] B. V. Rao and A. A. Deshpande, "Why the Barker Sequence bit length does not exceed thirteen", *Journal of the Institution of Electronics and Telecommunication Engineers,* v.34 n. 6 Nov-Dec 1988, pp.461-462.

[Rap 84] Stephen S. Rappaport and Donald M. Grieco, "Spread-Spectrum Signal Acquisition: Methods and Technology", *IEEE Communications Magazine,* Vol.22, No.6, 1984.

[Rap 95] Theodore S. Rappaport, *Wireless Communications: Principles and Practice,* IEEE Press and Prentice Hall 1995.

[Rar 83] Benjamin Rarzen, *Design of Crystal and Other Harmonic Oscillators,* John Wiley & Sons, Inc. 1983.

[Pol 84] A. Polydoros and C. L. Weber, "A Unified approach to serial search spread-spectrum code acquisition - Part 1: General Theory", *IEEE Transactions on Communications,* Vol. Com-32, No.5, May 1984.

[Sim 85] M. K. Simon et al., *Spread Spectrum Communication,* Computer Science Press, 1985.

[The 94] *The Spread Spectrum Handbook,* Third Edition, Standford Telecommunications, Inc., September, 1994.

[Yan 95] Desmond Yan and Paul Ho, "Code Acquisition in a CDMA System based on Barker Sequence and Differential Detection", *IEEE International Symposium on Personal, Indoor and Mobile Radio Communication,* PIMRC v.1, 1995, pp. 233-236.

[Z18 94] *Z180 Family Microprocessors and Peripherals Data Book,* Zilog, Inc., Cambell, California, 1994.

[Z20 94] *Z2000 Spread Spectrum Development Kit User's Manual,* Zilog, Inc., Cambell, California, 1994.

Appendix A

Moving Average System

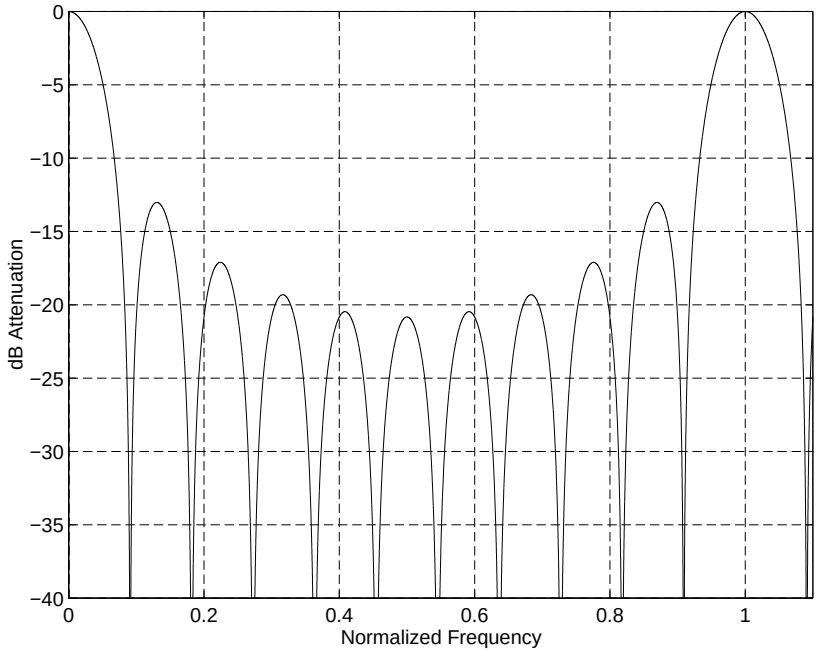

Figure A.1: Magnitude Response of the Moving Average System (N = 11)

Appendix B

Simulation Approach

B.1 Simulation in the MATLAB Environment

This spread spectrum simulation involves a significant computation and it is time-consuming and memory-consuming. It is strongly recommended to make every effort to avoid unnecessarily complicated algorithms and to follow advice from MATLAB manual [Mat 92].

1. Vectorize the simulation algorithm to eliminate "*for*" and "*while*" loops. The reason is that MATLAB 's built-in vector and matrix operations are 10 times faster than its compiler/interpreter operations.

2. Preallocate any vectors in which output results are stored. By doing so, we eliminate memory allocation each time any number of vector elements are needed and increase the efficiency of memory by using contiguous memory space.

It is also important to use the "clear" function in MATLAB to remove variables no longer in use from the work space. Otherwise, the computer system may not have enough swap space for new variables.

The simulation models for some functional blocks are straight-forward and the source code for these models is easy to understand if the reader is familiar with the operation of the functional blocks and with MATLAB. In the following, we include the code description

for some functional blocks which might not be easy for the beginner.

B.2 PN Matched Filter

```
SET FILTER INPUT = DOWN CONVERTER OUTPUT
SET THE PN MATCH FILTER' TAP COEFFICIENTS
FRONT END PROCESSOR CALCULATES THE AVERAGE OF TWO CONSECUTIVE SAMPLES
LOOP FOR EVERY SAMPLE:
      IF THE INCOMING CHIP LENGTH < PN CHIP LENGTH
 CORRELATION LENGTH = INCOMING CHIP LENGTH
      OTHERWISE
 CORRELATION LENGTH = PN CHIP LENGTH
      ENDIF
      CALCULATE THE CORRELATION VALUE
ENDLOOP
SET FILTER OUTPUT = QUANTIZATION OF CORRELATION VALUE
```

B.3 Symbol Tracking Processor

```
SET SYMBOL TRACKING PROCESSOR(STP) INPUT = POWER DETECTOR OUTPUT
SET THRESHOLD
THE FIRST DETECTION HAPPENS AT THE CORRECT LOCATION
LOCATE THIS FIRST SYMBOL DETECTION AS FLYWHEEL VALUE
LOOP FOR EVERY SYMBOL DURATION:
     FETCH THE NEXT 3 STP INPUTS CENTERED AT THE LAST FLYWHEEL VALUE
     FIND THE MAXIMUM POWER IN THESE 3 STP INPUTS
     IF THIS MAXIMUM POWER >= THRESHOLD
THIS IS A SYMBOL DETECTION
THIS MAXIMUM OUTPUT IS REGARDED AS A STP OUTPUT
```

```
LOCATE THIS SYMBOL DETECTION AS A UPDATED FLYWHEEL VALUE
     OTHERWISE
THIS MAXIMUM OUTPUT IS STILL BE REGARDED AS A STP OUTPUT
SET CURRENT FLYWHEEL VALUE = THE LAST FLYWHEEL'S VALUE
     ENDIF
ENDLOOP
```

B.4 Commented Simulation Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   This program first generates the spread spectrum signal;
%   The noise is added to each baseband sample;
%   The quantization operation is performed before and after
%   each functional block.  The flywheel strategy is applied in
%   the symbol tracking block.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;                                  % clear the memory space
Iter_No = 2;                            % Repeat the simulation
data_No = 10000;             % The data number for each simulation

for EBNO_ITER = 1 : 1,

EbNo_dB = 8;                            % Eb/No in dB
EbNo = 10 ^ (EbNo_dB / 10);             % Eb/No
barker_11 = [1 1 1 0 0 0 1 0 0 1 0];    % PN code
```

```
SampleNoPerChip = 2;                    % Two samples per chip
chip = barker_11 * 2 - 1;               % PN code in polar format
chip_rate = length(chip);               % PN code length
Noi_var = SampleNoPerChip / 2 * chip_rate / EbNo;   % Calculate the channel
            % noise variance
Throughput = zeros(1, Iter_No);         % storing the number of the correct
% received data
for Iter = 1 : Iter_No,                 % begin the simulation

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             ORIGINAL DATA GENERATOR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rand('seed',sum(100*clock));
% set the seed of the uniform generator to different value each time
data_in = rand(1, data_No + 2) > .5;
% rand() is a uniform random number generator. 1 and 0 are generated
% with the same probability 0.5

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Input Processors (INPSR)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
inpsr_out = data_in;       % The Input Processor is disabled in BPSK

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           Differential Encoder (DENCO)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = length(inpsr_out);
denco = zeros(1, N);                % preallocate the memory space
```

```
denco(1) = xor(inpsr_out(1), 1);     % The first is xor with '1'
for i = 2 : N,
    denco(i) = xor(inpsr_out(i), denco(i - 1));    % 'xor' operation
end
denco_out = [1 denco];               % The '1' is also transmitted
denco_out = 2 * denco_out - 1;       % change to polar format

clear denco inpsr_out;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                PN Spread Coder (PNCOD)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = length(denco_out);
pncod_out = zeros(1, N * chip_rate);
for i = 1 : N,
    pncod_out(chip_rate*(i-1)+1 : chip_rate*i) = denco_out(i)*chip;
end
clear denco_out;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    BPSK Modulator with Sampling Process (BPSPL)
%    The signal is generated in complex format
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = fix(length(pncod_out) * SampleNoPerChip);
bpspl_out = exp(j*(pncod_out(fix((0 : N-1)/SampleNoPerChip)+1)<0)*pi);
clear pncod_out;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%  Channel Noise Adder, AGC (CNAAA)
%  The noise is also in complex format with its
%  variance calculated from Eb/No
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = length(bpspl_out);
cnaaa_out=bpspl_out+(randn(1,N)+j*randn(1,N))*sqrt(Noi_var);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%              Matched Filter (MAFLT)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
match_in = cnaaa_out;
BitNo_Matchin = 3;
Quanlevel_Matchin = 2 ^ BitNo_Matchin;
match_in=round(match_in*Quanlevel_Matchin)/Quanlevel_Matchin;
BitNo_Matchout = 10;
Quanlevel_Matchout = 2 ^ BitNo_Matchout;
tap = fliplr(chip);             % reverse the chip
N = length(match_in);
fep_out = zeros(1, N);          % preallocate the memory space

%  front end processor average the matched filter input  %
fep_out(1) = match_in(1);
fep_out(2 : N) = (match_in(2 : N) + match_in(1 : (N - 1))) / 2;
match = zeros(1, N);            % preallocate the memory space
for i = 1 : N,
    k = fix((i - 1) / 2) + 1;
    if k > chip_rate
       k = chip_rate;   % decide if this is only partial overlay
```

```
    end
    match(i) = sum(tap(1 : k).*fep_out(i:-2:(i-2*k+2)));
end

% Quantized the matched filter output to 10 bits  %
match_out = fix(match * Quanlevel_Matchout) / Quanlevel_Matchout;
clear match match_in fep_out cnaaa_out dncov_out bpspl_out;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Power Detector (POWDE)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
BitNo_Viewport = 8;
Quanlevel_VP = 2 ^ BitNo_Viewport;
BitNo_Magnitude = 10;
Quanlevel_Magnitude = 2 ^ BitNo_Magnitude;
VP_CTRL = 0;

%  Quantized to 8 bit with a view port control  %
maflt_8 = fix(match_out/(2^VP_CTRL)*Quanlevel_VP)/Quanlevel_VP;
powde = real(match_out);     % the real value is used in BPSK case
powde_10 = fix(powde * Quanlevel_Magnitude) / Quanlevel_Magnitude;
clear match_out;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Symbol Tracking (SYMTK)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
THRESHOLD = 5;                   % define the threshold
N = round(length(powde_10) / 2 / chip_rate) - 2;
```

```
symtk_out = zeros(1, N);        % preallocate the memory space
flywheel = zeros(1, N);         % preallocate the memory space
flywheel(1) = 22;               % first one is correctly detected
Ha_win = 1;                     % specify the window size
for i = 2 : N,
    flywheel(i) = flywheel(i - 1) + 2 * chip_rate;
    A = powde_10((flywheel(i) - Ha_win) : (flywheel(i) + Ha_win));
    max_A = max(A);
    B = find(A == max_A);
    if max_A >= THRESHOLD,
       flywheel(i) = flywheel(i) + B(1) - 2;
    end
end
symtk_out = maflt_8(flywheel);
clear powde_10 maflt_8;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Differential Demodu (DDEMO)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = length(symtk_out);
test = zeros(1, N - 1);         % preallocate the memory space

% Calculate the dot product for decision circuit  %
test = symtk_out(1 : N - 1) .* conj(symtk_out(2 : N));
data_out = test < 0;            % differential decoder's decision

% calculate the throughput for this simulation  %
Throughput(Iter)=sum(data_in(1:data_No)==data_out(1:data_No));
```

```
% calculate the BER of this simulation  %
ber(EBNO_ITER,Iter) = 1 - Throughput(Iter) / data_No;

end             % end of one simulation

BER(EBNO_ITER) = 1 - sum(Throughput) / data_No / Iter_No

end             % end of the total iterations of simulations
```

Appendix C

Modifications to the IAR Compiler

- In the \iar\iccz80\getchar.c file,

  1. add an include file for both Z80 and 64180 intrinsic functions:

  ```
  #include <intrz80.h>
  ```

  2. add the following two definitions:

  ```
  #define MIMIC_DATA   0xF0    /* addr of mimic RBR fifo */
  #define MIMIC_CONT   0xF5    /* addr of mimic LSR Reg. */
  ```

  3. modify the low_level_get() function by changing the
     input/output addresses:

  ```
  char low_level_get(void) /* Read one char from I/O Hardware-dep */
  {
    while (input(MIMIC_CONT) & 0x20)
  ```

```
      ;
  return (input(MIMIC_DATA) & 0x7f);

/* return In_EOF; */            /* indicate failure */
}
```

4. save it back to the same name.

- In the \iar\iccz80\putchar.c file,

1. add a include file for both Z80 and 64180 intrinsic functions:

```
#include <intrz80.h>
```

2. add the following two definitions:

```
#define MIMIC_DATA   0xF0    /* addr of mimic RBR fifo */
#define MIMIC_CONT   0xF5    /* addr of mimic LSR Reg. */
```

3. modify the __low_level_put() function by changing the input/output addresses:

```
void __low_level_put(char c)
{
  while (input(MIMIC_CONT) & 0x01)  /* while buffer not ready */
    ;
  output(MIMIC_DATA, c);
}
```

4. save it back to the same name.

- Copy these two modified files into the \iar\lib directory.
- Compile these two files to create .r01 object modules using the following commands:

```
iccz80 putchar -ml -z9 -v1 -b
iccz80 getchar -ml -z9 -v1 -b
```

'-ml' specifies that the large memory model is used and '-z9' define the optimization level. '-v1' selects the processor type is 64180 '-b' makes the object code a library module. Note that the Z80182 processor is compatible with the 64180 instead of Z80 processor.

- Add these two modules to the run-time library module cl64180 by the following commands:

```
xlib
def-cpu z80
rep-mod putchar cl64180
rep-mod getchar cl64180
exit
```

Now, the library module cl64180 will have the modified putchar and getchar modules.

- To test the modified modules before installing them in the library, place the following lines into the rf.xcl file.

```
-A putchar
-A getchar
cl64180
```

- Also, we need to modify the rf.xcl file to allocate the memory segments:

```
-Z(CODE)CODE,RCODE,CDATA0,ZVECT,CONST,CSTR,CCSTR=4680
-Z(CODE)INTVEC=2000
-Z(DATA)DATA0,IDATA0,UDATA0,ECSTR,WCSTR,TEMP,CSTACK+1000#1FFF
```

- Finally, the last two lines of the generated rf.hex file should be deleted to provide file compatiblity.

VITA

Guoliang Li was born in China on July 12, 1971. He received his Bachelor of Science of Electrical Engineering (BSEE) from Xiamen University, China in June 1992. After that, he worked in Xiamen TV Station, China as a system engineer. In 1994, he came to the Bradley Department of Electrical Engineering at Virginia Polytechnic Institute and State University (Virginia Tech) for his M.S.E.E., with concentration in the communication area. From August 1995 to January 1996, he worked as a co-op design engineer in the Personal Communications System (PCS) Department at Bell-Northern Research (now Nortel Research) in Richardson, Texas. From February 1996 to August 1996, he was a research assistant in the Center for Wireless Telecommunications at Virginia Tech. His Master research focused on the physical layer design for a spread spectrum wireless LAN.