Cleveland State University
Department of Electrical Engineering

A Small Terminal for Satellite Communication Systems

by
Fuqin Xiong
Dong Wu
Min Jin

Department of Electrical Engineering
Cleveland State University
Cleveland, Ohio 44115

submitted to

NASA Lewis Research Center

Grant NCC3-201 Final Report, May 8, 1994

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

This is a final report for the design activities supported by the NASA grant NCC3-201. The goal of this project is to design and develop a small portable terminal system for satellite communications. A multi-scheme, multi-rate modulator/demodulator (MODEM) and a convolutional-Viterbi coder/decoder (CODEC) are the main parts of this system.

Recent technological improvements are leading towards low-cost satellite communication systems that can be applied to rural communications worldwide[1][2]. Advance in signal processing and error-correction techniques allow more efficient use of the space segment by locating the sophisticated processing equipment on board the satellite[3][14]. Combined with the trend of higher power and higher frequency satellites this results in simple and inexpensive ground terminal architecture, making VSAT technology more attractive. The Advanced Communications Technology Satellite (ACTS) is certainly no exception to this general trend.

ACTS operating at Ka band incorporates most of these technological advances, Namely, higher power, higher frequency, frequency and spatial reuse using spot beams and polarization. These capabilities and facts the ACST uses beam hopping makes the development of small portable terminals very attractive to service to low population density areas, remote locations, as well as the areas where traffic is spread geographically. Further, the efficiency and flexibility of a beam-hopping satellite system serving small and economical earth stations would also benefit developing nations.

This project is a part of designing and realizing this kind of small and economical earth stations. The research activities involved in this project include:

(1). Design of a programmable Modulator/Demodulator which can provide multiple bit rates, multiple modulation and demodulation schemes.

(2). Design of a code rate 1/2, constaint length 7 Convolutional coder/Viterbi decoder which can provide a low cost, high performance solution for FEC (Forward Error Correction) system requirement.

(3). Design of a control system for this small terminal system with one microcontroller. It can write the control code into the internal registers of the VLSI chips for proper system configuration and control. Also several required clocks are produced using this microcontroller. Another microcontroller is used to realize digital Phase Lock Loop (PLL) for carrier recovery for QPSK, OQPSK, and BPSK.

(4). Translation of this design into a prototype which was built using wire wrapping method.

(5). Debugging and trouble-shooting both hardware and software of this prototype system.

(6). Testing the transmitter and the receiver.

The MODEM can provide four kinds of modulation schemes: BPSK, QPSK, OQPSK, and MSK. Direct Digital Synthesizer (DDS) is used to generate modulation signals. Bit synchronizer/PSK demodulator (STEL-2110A) and digital Phase Lock Loop (PLL) are used to provide clock and carrier synchronization for BPSK, QPSK and OQPSK. A new type of low cost, easily realized MSK demodulator is presented for MSK demodulation. Five kinds of low bit rates ranging frcm 1,200 bps to 19,200 bps are employed in each kind of modulation scheme. Four different modulation schemes and five different bit rates provides us twenty different communication mode combinations. That which combination is well suited for low bit rate satellite communications will be tested through field experiments using ACTS launched by NASA in October, 1993. This is also the final purpose of this project.

The QUALCOMM Q0256 convolutional coder/Viterbi decoder VLSI chip is selected to provide low bit rate, high volume communications. Rate 1/2 and constraint length 7 convolutional coding scheme is selected. 3-bit soft-decision encoder data greatly improve the BER performance of the whole system. Only about 5.2 dB E_b/N_0 is required for 1E-6 BER performance.

Two 80C32 microcontrollers in the INTEL MCS-51 family are used in the control system and the digital PLL. Over 2,500 lines software are developed for the proper system operation, control and digital PLL calculation. An ICE-51FX emulator is used for the software developing. Selections of modulation schemes and bit rates can be done easily by switches. The control system also provides a master clock to the vocoder - the stage before the MODEM and CODEC.

Most of the system has been successfully built according to the system design requirements. The measured power spectral densities of modulated signals, BPSK, QPSK, OQPSK and MSK, under five different data rates agreed with the theoretical predictions very well. 0 BER performance was realized when signals passed through an idea channel. Realizing carrier synchronization for BPSK, QPSK and OQPSK in low bit rate situation and finding an MSK demodulation scheme suitable for low cost, small terminal are the key points in the system design and realization. Having successfully solved these problems with innovation offers several unique features to this system.

There are also some problems left. The system is not working at bit rate of 19.2 kbps for BPSK because the microcontroller which we employ doesn't have the function to generate 50% duty cycle clock. Thus, in our design, we generate a double frequency clock, then let it pass through a frequency divider to generate the required frequency 50% clock. For 19.2 kbps, we have to generate a 76.8 kbps clock for BPSK encoder use, but it is not possible for microcontroller 80C32. We can use a microcontroller which can directly generate 50% duty cycle clock or a microcontroller which can generate a 76.8 kbps clock to solve this problem. This is not a big problem.

Another problem is with MSK demodulation, only 2.4 kbps and 4.8 kbps bit rate can be successfully demodulated, but they are not robust enough. The key point here is to find or built a noncoherent robust FSK demodulator. NE564 which we used in our system is what we can find to most suit for our system, but it still can not give us satisfactory results.

Overall, we have meet many problem in our system realization and we have solve most of them. For QPSK, OQPSK and BPSK, the MODEM/CODEC can successfully operate at bit rate 1.2 kbps, 2.4 kbps, 4.8 kbps and 9.6 kbps. For MSK, transmitter is well working. there is a problem with the demodulator under some bit rates.

Chapter 2

MODEM/CODEC THEORY

2.1 MODEM Theory

For satellite communication, due to the nonlinear amplification of the TWTA and limited bandwidth allocation, the most efficient MODEM technique is the PSK with coherent detection. It has the desirable characteristic that the transmitted signal has a constant envelope with the information in the carrier phase transitions. Thus it is the least susceptible to the nonlinear amplification. It also has a higher bandwidth efficiency than the FSK even though the FSK is a constant envelope modulation too.

BPSK (Binary Phase Shift Keying), QPSK (Quadrature Phase Shift Keying) and OQPSK (Offset QPSK) are most often used PSK variations for satellite modems [4]-[8]. MSK (Minimum Shift Keying), which can be considered as an OQPSK with sinusoidal pulse shaping, was developed in recent years [9][11][12].

2.1.1 BPSK (Binary Phase Shift Keying)

BPSK is a binary signaling scheme where the phase of the carrier changes between two values separated by 180° with each new binary digit. Hence, two signals $s_1(t)$ and $s_2(t)$ are employed to represent the binary digits 1 and 0, as follows

$$s_1(t) = A\cos(2\pi f_c t + \theta)\ , \qquad (k-1)T < t < kT \tag{2.1}$$

$$s_2(t) = A\cos(2\pi f_c t + \theta + \pi) = -A\cos(2\pi f_c t + \theta)\ , \qquad (k-1)T < t < kT \quad (2.2)$$

or simply as

$$s(t) = d_k(t) A\cos(2\pi f_c t + \theta) \qquad (2.3)$$

where f_c is the carrier frequency, θ is the initial phase of the carrier, A is the carrier amplitude, T is the bit duration, and $d_k(t)$ is the binary data stream, $d_k(t) = \{d_0, d_1, d_2, \cdots\cdots\}$, consisting of bipolar pulses; that is, the values of d_k are +1 or -1, representing binary one and zero, respectively. The power spectral density of BPSK is shown in Figure 2.7.

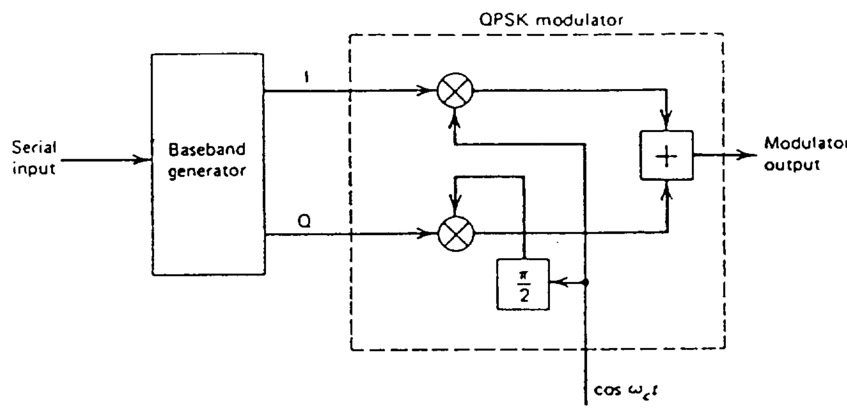


Figure 2.1: Generalized quadrature modulator.

Fiqure 2.1 shows the generalized quadrature modulator which is applicable for BPSK, QPSK and OQPSK. For BPSK the baseband generator is not needed, and only the upper half of the modulator is required.

Figure 2.2 shows the generalized quadrature demodulator which is applicable for BPSK, QPSK and OQPSK. BPSK does not need the lower half of the circuit and combiner. The input signal is $d_k(t) A\cos 2\pi f_c t$. The carrier recovery circuit detects and regenerates a carrier signal that is both frequency and phase coherent with the original transmit carrier. The output of the mixer is the product of the two inputs (the BPSK signal and the recovered carrier). The low-pass filter (LPF) separates the recovered binary data from the complex demodulated spectrum. The demodulation process is as follows:

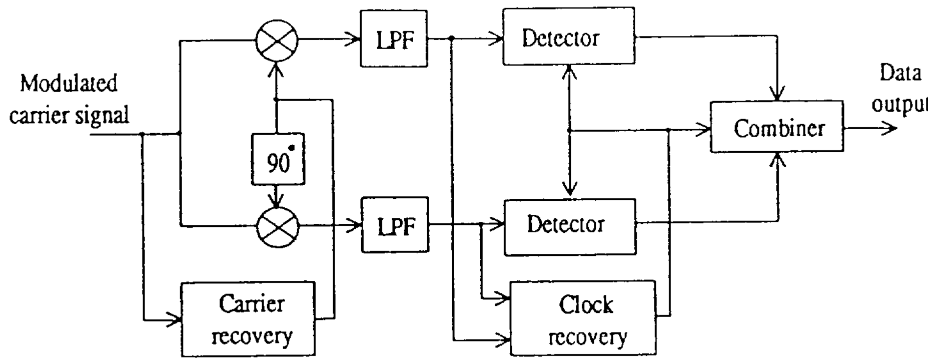


Figure 2.2: Generalized quadrature demodulator.

$$\begin{aligned} Multiplier\ output &= [d_k(t) A \cos 2\pi f_c t]\,(\cos 2\pi f_c t) \\ &= \tfrac{A}{2} d_k(t) + \tfrac{A}{2} d_k(t) \cos 2(2\pi f_c) t \end{aligned} \tag{2.4}$$

After low-pass filter, the second component is filtered out. So we obtain

$$Data\ output = \frac{A}{2} d_k(t)$$

which is proportional to the original data stream we have transmitted.

2.1.2 QPSK (Quadrature Phase Shift Keying)

QPSK is an M-ary encoding technique where M=4. Figure 2.3 illustrates the partitioning of a typical data stream for QPSK. Figure 2.3(a) shows the original data stream $d_k(t) = \{d_0, d_1, d_2, d_3, \cdots\cdots\}$ consisting of bipolar pulses. This data stream is divided into two bit streams: (1) the in-phase stream $d_I(t)$ for I channel, (2) the quadrature stream $d_Q(t)$ for Q channel. This is illustrated in Figure 2.3(b).

$$\begin{aligned} d_I(t) &= \{d_0, d_2, d_4, \cdots\cdots\} \qquad (even) \\ d_Q(t) &= \{d_1, d_3, d_5, \cdots\cdots\} \qquad (odd) \end{aligned} \tag{2.5}$$

Note that $d_I(t)$ and $d_Q(t)$ have half the bit rate of $d_k(t)$. A convenient orthogonal realization of a QPSK waveform, $s(t)$, is achieved by modulating the in-phase and quadrature data streams onto a cosine and a sine carriers, as follows:

$$s(t) = \frac{A}{\sqrt{2}} d_I(t) \cos 2\pi f_c t + \frac{A}{\sqrt{2}} d_Q(t) \sin 2\pi f_c t \tag{2.6}$$

Using the trigonometric identities, Equation (2.6) can also be written as

$$s(t) = A\cos\left[2\pi f_c t + \theta(t)\right] \tag{2.7}$$

The value of $\theta(t)$ will correspond to one of the four possible combinations of $d_I(t)$ and $d_Q(t)$ in Equation (2.6). These values are: $\theta(t) = \pm 45°$, or $\pm 135°$.

The power spectral density for QPSK is given by [4]

$$G(f) = 2PT\left(\frac{\sin 2\pi fT}{2\pi fT}\right)^2, \tag{2.8}$$

where P is the average power in the modulated waveform, as shown in Figure 2.7.

The block diagram of a QPSK modulator is shown in the Figure 2.1. The baseband generator is a serial to parallel converter that is used to split data stream $d_k(t)$ into $d_I(t)$ and $d_Q(t)$. The in-phase stream $d_I(t)$ modulates the cosine function. This produces a BPSK waveform. Similarly, the quadrature stream $d_Q(t)$ modulates the sine function, yielding a BPSK waveform orthogonal to the cosine function. The summation of these two orthogonal components of the carrier yields the QPSK waveform.

The block diagram of a QPSK receiver is shown in Figure 2.2. The input signal is directed to the I channel, Q channel and the carrier recovery circuit. The detector here is a integrate-dump circuit (or Matched filter). The QPSK signal is demodulated in the I and Q channels, which generate the original I and Q data streams.

The incoming QPSK signal can be seen from Equation (2.6) as

$$s(t) = Ad_I(t)\cos 2\pi f_c t + Ad_Q(t)\sin 2\pi f_c t \tag{2.9}$$

For I channel, recovered carrier is $\cos 2\pi f_c t$, so the output of I channel is

$$\begin{aligned} I_{out} &= \left[Ad_I(t)\cos 2\pi f_c t + Ad_Q(t)\sin 2\pi f_c t\right]\cos 2\pi f_c t \\ &= \tfrac{A}{2}d_I(t) + \tfrac{A}{2}d_I(t)\cos 2(2\pi f_c)t + \tfrac{A}{2}d_Q(t)\sin 2(2\pi f_c)t \end{aligned} \tag{2.10}$$

after LPF, second and third components are filtered out. So

$$I_{out} = \frac{A}{2}d_I(t).$$

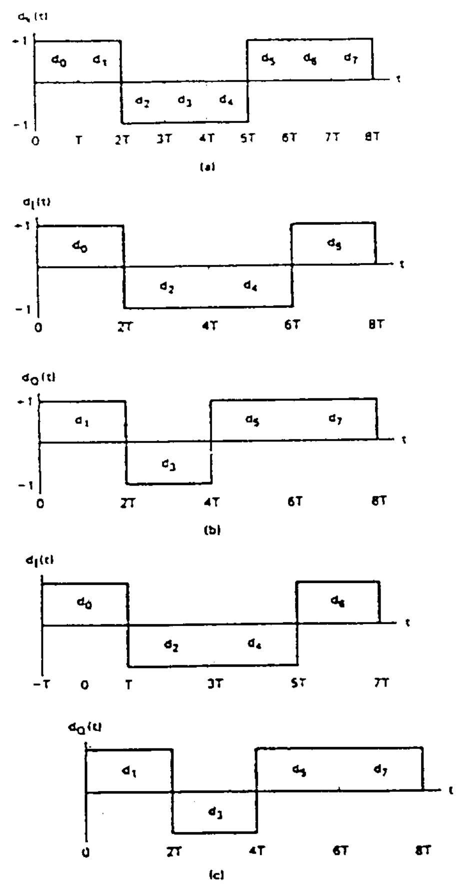


Figure 2.3: (a) input data stream, (b) QPSK data stream, (c) OQPSK data stream.

For Q channel, recovered carrier is $\sin 2\pi f_c t$, so the output of Q channel is

$$\begin{aligned} Q_{out} &= [Ad_I(t)\cos 2\pi f_c t + Ad_Q(t)\sin 2\pi f_c t]\sin 2\pi f_c t \\ &= \tfrac{A}{2}d_I(t)\sin 2(2\pi f_c)t + \tfrac{A}{2}d_Q(t) - \tfrac{A}{2}d_Q(t)\cos 2(2\pi f_c)t \end{aligned} \tag{2.11}$$

after LPF, first and third components are filtered out. So

$$Q_{out} = \frac{A}{2}d_Q(t).$$

The output of I and Q channels are fed to the bit combining circuit, where they are converted from parallel I and Q data channels to a single binary output data stream $d_K(t)$.

2.1.3 OQPSK (Offset QPSK)

OQPSk signaling can also be represented by Equations (2.6) or (2.7); the difference between the two modulation schemes, QPSK and OQPSK, is only in the alignment of the two baseband waveforms. In QPSK, the odd and even pulse streams are both synchronously aligned. In OQPSK, there is the same data stream partitioning and orthogonal transmission; the difference is that the timing of the pulse stream $d_I(t)$ and $d_Q(t)$ is shifted such that the alignment of the two streams is offset by T. Figure 2.3(c) illustrates this offset.

In QPSK, due to the alignment of $d_I(t)$ and $d_Q(t)$, the phase change of the carrier during any 2T interval can be any one of the four phases 0°, ±90° and 180°. Figure 2.4(a) shows a typical QPSK waveform for the sample sequence $d_I(t)$ and $d_Q(t)$ shown in Figure 2.3(b).

If a QPSK modulated signal undergoes filtering to reduce the spectral sidelobes, the resulting waveform will not longer have a constant envelope and in fact the occasional 180° phase shifts will cause the envelope to go to zero momentarily. When these signals are used in satellite channels employing highly nonlinear amplifiers, the constant envelope will tend to be restored. However, at the same time, all of the undesirable frequency side-lobes, which can interfere with nearby channels and other communication systems, are also restored.

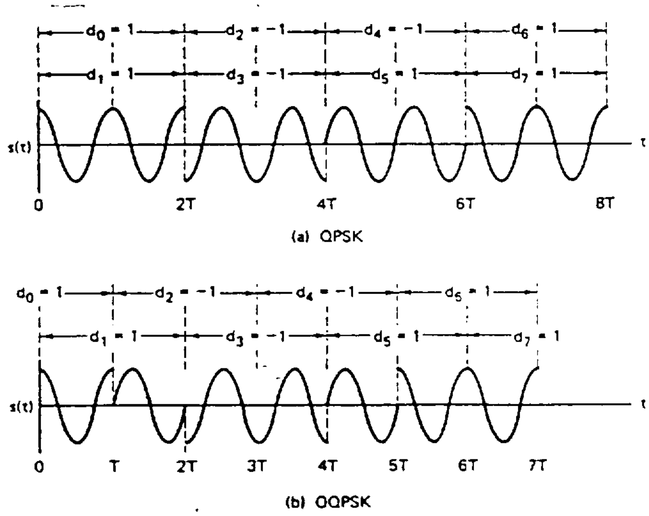


Figure 2.4: (a) QPSK and (b) OQPSK waveforms.

In OQPSK, the pulse streams $d_I(t)$ and $d_Q(t)$ are staggered and thus do not change states simultaneously. The possibility of the carrier changing phase by 180° is eliminated, since only one component can make a transition at one time. Changes are limited to 0 and ±90° every T seconds. Fiqure 2.4(b) shows a typical OQPSK waveform for the sample sequence in Figure 2.3(c). When an OQPSK signal undergoes bandlimiting, the resulting intersymbol interference causes the envelope to droop slightly in the region of ±90° phase transition, but since the phase transitions of 180° have been avoided in OQPSK, the envelope will not go to zero as it does with QPSK.

OQPSK can be used the same block diagram of Figure 2.1 and Figure 2.2 to be accomplished. The baseband generator of Figure 2.1 consists of a serial to parallel converter followed by a Q channel delay of T, and a delay of T in Figure 2.2 is needed after the detector in the I channel. Furthermore, the power spectral density of OQPSK is identical to that of QPSK.

2.1.4 MSK (Minimum Shift Keying)

MSK can be though of as a special case of OQPSK with sinusoidal pulse weighting [9][11][12]. Consider the OQPSK signal, with the bit streams offset as shown in Figure 2.3(c). If sinusoidal pulses are employed instead of rectangular shapes, the modified signal can be defined as MSK and equals

$$s(t) = d_I(t)\cos(\frac{\pi t}{2T})\cos 2\pi f_c t + d_Q(t)\sin(\frac{\pi t}{2T})\sin 2\pi f_c t \tag{2.12}$$

Figure 2.5 shows the various components of the MSK signal defined by Equation (2.12).The waveform in Figure 2.5(e) can be better understood if we use a trigonometric identity to rewrite Equation (2.12) as

$$s(t) = \cos(2\pi f_c t + b_k(t)\frac{\pi t}{2T} + \phi_k) \tag{2.13}$$

where

$$b_k(t) = -d_I(t)d_Q(t) \tag{2.14}$$

and ϕ_k is the initial phase.

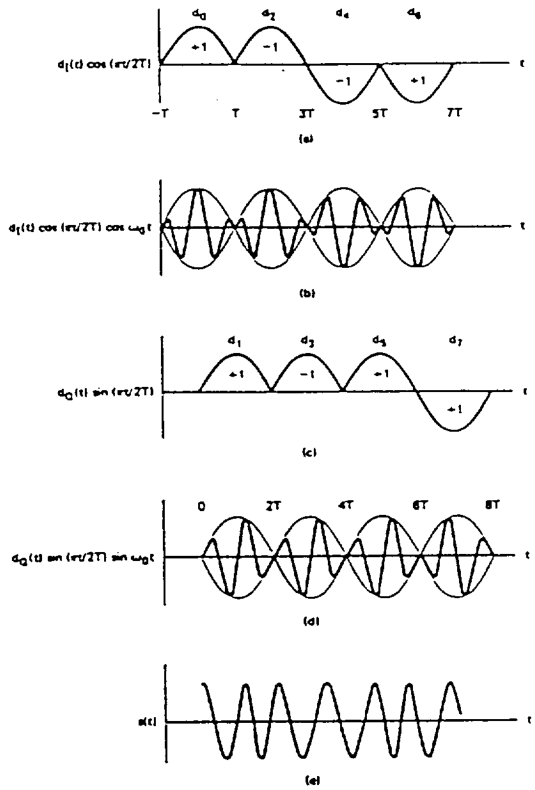


Figure 2.5: MSK waveforms.

From Figure 2.5 and Equation (2.13), we deduce the following properties of MSK:

(1) the waveform $s(t)$ has constant envelope;

(2) there is phase continuity in the RF carrier at the bit transitions;

(3) the waveform $s(t)$ can be regarded as an FSK waveform with signaling frequencies:

$$f_{c+} = f_c + \frac{1}{4T} \quad ; \quad f_{c-} = f_c - \frac{1}{4T}$$

Therefore, the minimum tone separation requires for MSK modulation is

$$\Delta f = f_{c+} - f_{c-} = \frac{1}{2T} \tag{2.15}$$

which is equal to half the bit rate. Notice that the required tone spacing for MSK is one-half the spacing, $\frac{1}{T}$, required for the noncoherent detection of FSK signals.

The modulation and demodulation block diagrams are shown in Figure 2.6. In modulation, the serial data stream $d_k(t)$ is converted into its even and odd bit

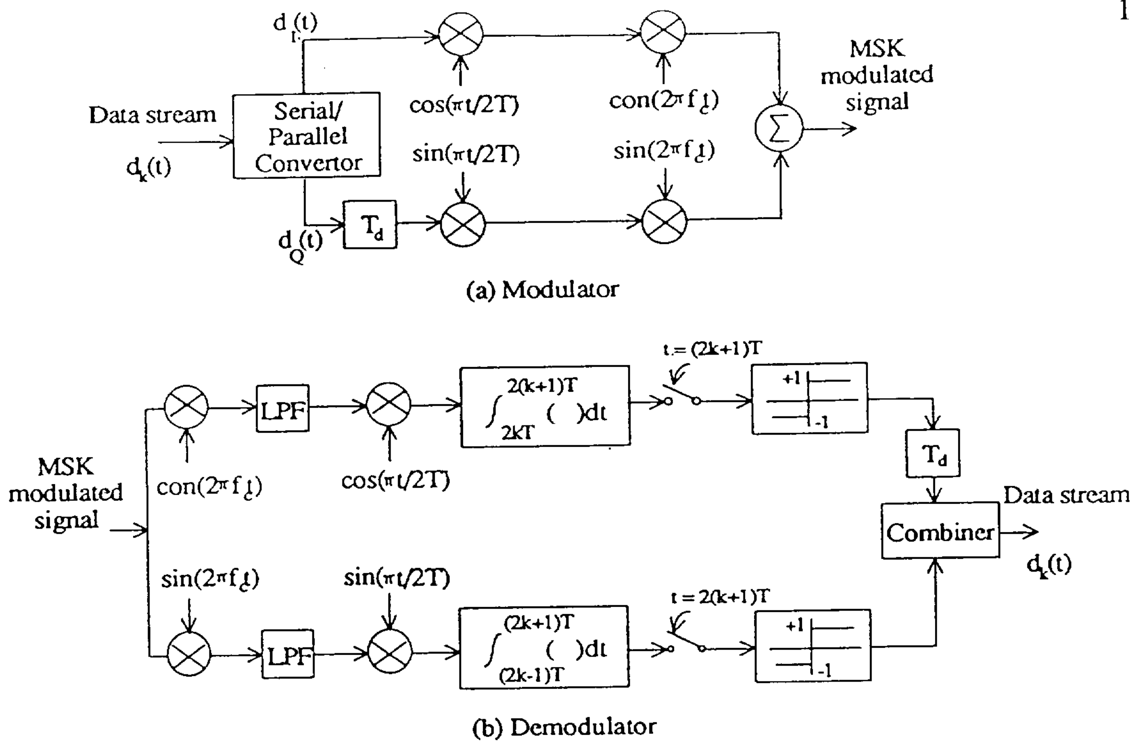


Figure 2.6: Block diagrams of MSK (a) modulator, (b) demodulator.

streams, $d_I(t)$ and $d_Q(t)$, which are staggered $\frac{1}{2}$ symbol. Each symbol of $d_I(t)$ and $d_Q(t)$ is then weighted by a sinusoid signal. If the symbol weighting function $\cos(\frac{\pi t}{2T})$ and $\sin(\frac{\pi t}{2T})$ are replaced by rectangular shaping functions, MSK becomes Offset QPSK. Without staggered by $\frac{1}{2}$ symbol and sinusoidal weighting, QPSK results.

Because MSK is a quadrature-multiplexed modulation scheme, it can be optimally detected by coherently demodulating its in-phase and quadrature components separately, as shown in Figure 2.6(b).

The power spectral density $G(f)$ for MSK is given by[4]

$$G(f) = \frac{16PT}{\pi^2}\left(\frac{\cos 2\pi fT}{1-16f^2T^2}\right)^2 \tag{2.16}$$

and shown in Figure 2.7.

The normalized power spectral density (P=1W) for BPSK, QPSK, OQPSK and MSK are sketched in Figure 2.7[10]. The one which has wider main-lobe has less bandwidth efficiency. The one which has higher side-lobes is more susceptible to nonlinearity. Even though QPSK and OQPSK have same power spectral density, OQPSK

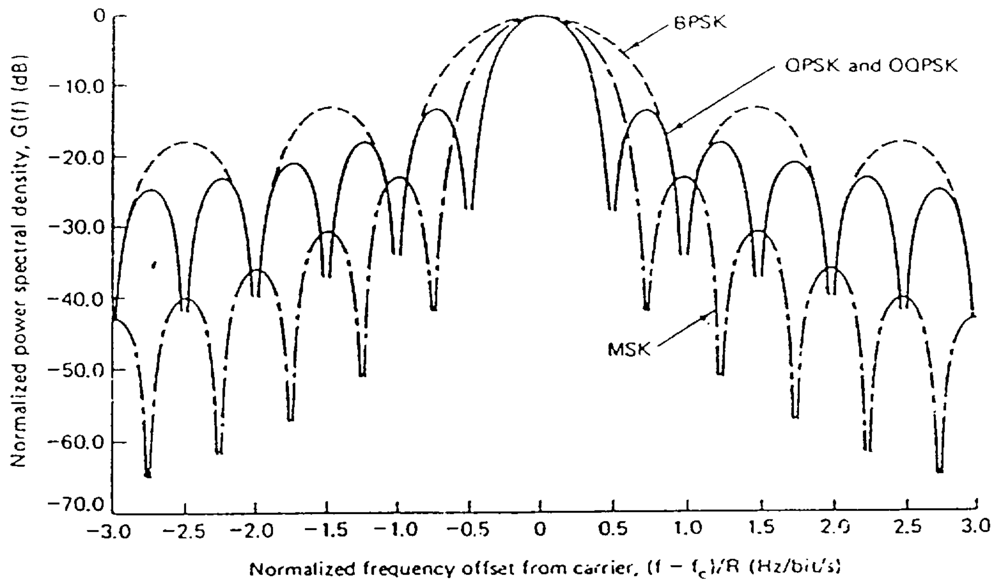


Figure 2.7: Normalized power spectral densities for BPSK, QPSK,OQPSK and MSK.

has better immunity to nonlinearity since it does not have 180° phase transitions like QPSK does.

It is seen from Figure 2.7 that the main-lobe bandwidth (null-to-null bandwidth) of these modulations are different (T is the bit duration):

For BPSK, $\quad BW_{BPSK} = 2.0/T$

For MSK, $\quad BW_{MSK} = 1.5/T$

For QPSK and OQPSK, $\quad BW_{QPSK,OQPSK} = 1.0/T$

As we can see, the BPSK has poorest bandwidth efficiency and immunity. The MSK has lower side-lobes than QPSK or OQPSK. This is a consequence of multiplying the data stream with a sinusoid, yielding more gradual phase transitions. The more gradual the transition, the faster the spectral tails drop to zero. MSK has the best immunity, moderate bandwidth efficiency.

All of them have almost the same bit error rate (P_b or BER) at same signal to noise ratio. That is, for coherent detection[4],

$$P_b = Q\left[\sqrt{\frac{2E_b}{N_0}}\right] \tag{2.17}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-y^2/2}dy$, E_b is the bit energy, and $\frac{N_0}{2}$ is the double-sided noise power spectral density at the receiver input.

2.2 CODEC Theory

2.2.1 Convolutional Encoder

Convolutional codes have been studied and used for forward error correction (FEC) in digital communication systems since the 1950's. A convolutional code maps a number (n) of information bits into a number (m) of single-bit codewords to be transmitted over the channel, where m>n. The ratio of n/m is referred to the code rate.

The transformation from information bits to codewords for transmission is accomplished by a time convolution of the information data with a finite-memory windowing function commonly referred to as a generating function. In the case of the rate 1/2 code, two generating functions G0 and G1 are convolved with the information data stream such that each time a new information data bit is considered, the G0 and G1 generating functions create one output bit or codeword, respectively.

The length of the finite memory of the convolutional generating function is the constraint length of the code. Figure 2.8 shows the generating functions of the rate 1/2 and 1/3 codes implemented by the Q0256 convolutional encoder. As the diagram shows, the memory length of the encoder is that six previous bits plus the current input bit; thus, this is a constraint length seven code, commonly denoted as k=7. The generating functions of the convolutional code are identified by denoting the "taps" of each convoluting function. For the rate 1/2, k=7 code shown in Figure 2.8, the generating functions are denoted as

$$G0 = 1111001 \quad (binary) \qquad or \qquad G0 = 171 \quad (octal)$$

and

$$G1 = 1011011 \quad (binary) \qquad or \qquad G1 = 133 \quad (octal)$$

This code provides the best error correcting performance of all rate 1/2, k=7 codes[13][17].

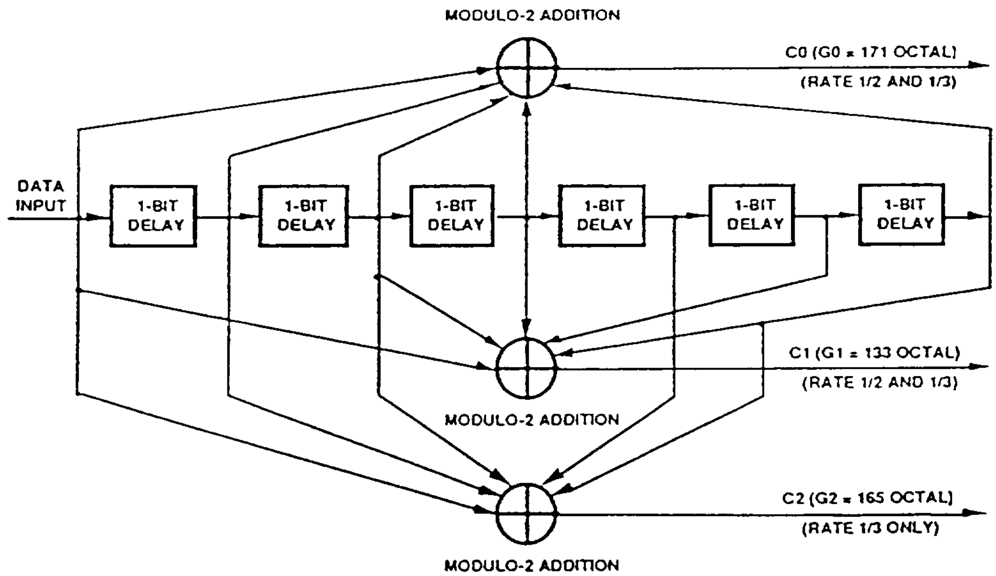


Figure 2.8: Constraint length seven (k=7) convolutional encoder.

2.2.2 Viterbi Decoder

While the implementation of a convolutional encoder is quite straightforward as shown in the previous section, the decoding of such a coded data stream at the receiving node is quite complex. In the late 1960's , Dr. A. J. Viterbi described a maximum likelihood decoding technique which greatly reduced the circuit sophistication of previous approaches.

Viterbi decoding consists fundamentally of three processes[17]. The first step in the decoder process is to generate of a set of correlation measurements, known as branch metrics, for each m grouping of codewords input from the communication channel (where m is 2 for rate 1/2 codes). These branch metric values indicate the correlation between the received codewords and the 2^m possible codeword combinations.

The Viterbi decoder determines the state of the 7-bit memory at the encoder using a maximum likelihood technique. Once the value of the encoder memory is determined, the original information is known, since the encoder memory is simply the information that has been stored in the memory. To determine the encoder state, the second step in the Viterbi algorithm generates a set of 2^{k-1} (where k is

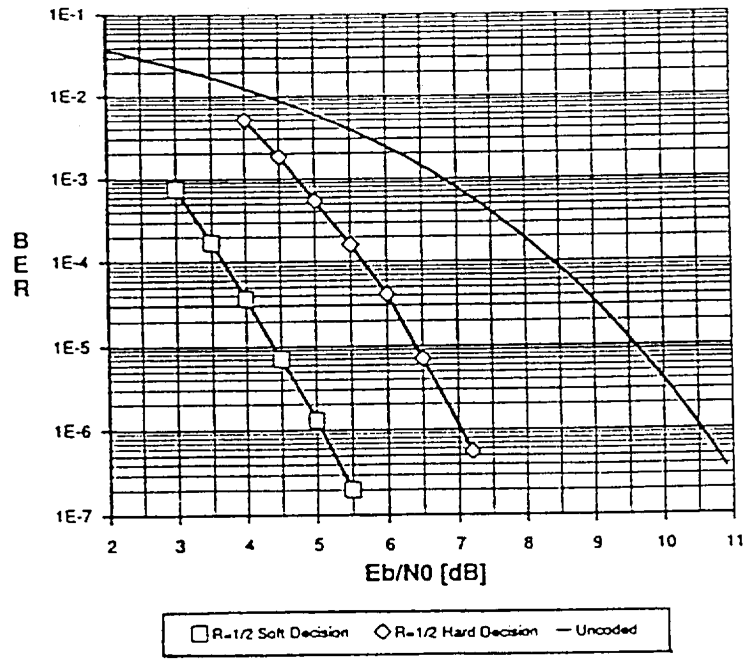


Figure 2.9: Q0256 coding performance.

the constraint length, i.e., k=7 for the Q0256 algorithms) state metrics which are measurements of the occurrence probability for each of the 2^{k-1} possible states as to the probable path taken to arrive at that particular state. These binary decision are stored in a path memory.

Step three computes the decoded output data. To do this, the path from the current state to some point in the finite past is traced back by chaining the binary decisions stored in the path memory during step 2 from state to state. The effects caused by noise to the one and only correct results are mitigated as the paths within the chainback memory converge after some history. The greater the depth of the chainback process the more likely that the final decoded result is error free. As a result, higher code rates and constraint lengths require longer chainback depth for best performance. The chainback memory in the Viterbi decoder traces the history of the previous states to arrive at the most probable state of the encoder in the past, and thus determine the transmitted data.

The Q0256 provides coding gain of 5.2 dB for rate 1/2 at 10^{-5} BER shown in Figure 2.9[17].

Chapter 3

SYSTEM DESIGN

The goal of this design and development is to produce a programmable digital coder/decoder and modulator/demodulator to provide flexible data rates and multiple modulation/demodulation modes. The approach adopted is to use current VLSI chips from ASIC manufacturers such as Qualcomm and Stanford Telecomm to simplify needed circuits and get good performance. These VLSI chips can handle wide range of data rates and are programmable via a microcontroller through assembly language code.

In this chapter we will discuss some assumptions and constraints, such as system specifications, system interfaces, selection of IF frequency and system structure.

3.1 System Specifications

(1). Coder/Decoder (CODEC)

Code Rate: 1/2

Constraint Length: K= 7

CODEC Scheme: convolutional encoder and Viterbi decoder;
differential encoder and differential decoder.

(2). Modulation/Demodulation (MODEM)

Data Rate: 1,200 bps, 2,400 bps, 4,800 bps, 9,600 bps, 19,200 bps.

MODEM Schemes: BPSK, QPSK, OQPSK, MSK.

(3). Control

One microcontroller (Intel 80C32) controls whole system. It can write or read control registers of the VLSI chips, communicate with the console display and provide other control signals. Another microcontroller is used in a digital Phase Lock Loop (PLL) for proper control and calculation.

(4). Flexibility

The MODEM/CODEC must be easily switched from one modulation mode to another and from one data rate to another.

(5). Full-Duplex Operation Capability

The terminal can be used as transmitter and receiver simultaneously.

3.2 System Interfaces

(1). Transmitter Interfaces

Tne input of the transmitter interfaces with a Vocoder which can provide several speech compression modes. The coming data is in serial format and is fed to the input of the encoder. After encoder the data rate is double, and becomes 2,400 bps, 4,800 bps 9,600 bps 19,200 bps and 38,400 bps. A master clock is created by the transmitter to synchronize the input data from the vocoder.

The output of the transmitter interfaces with a upconverter. Figure 3.1(a) shows the transmitter interfaces. The intermediate frequency (IF) signal has following specifications:

(a). IF signal frequency: f_{IF} = 4.8MHz.

(b). IF signal power: P_{IF} = 0~20dBm.

(c). Maximum IF bandwidth = 100KHz.

(2). Reveiver Interfaces

The input of the receiver interfaces with two input resources: one is the downconverter signal (RF input), another is the local oscillator signal (LO input).

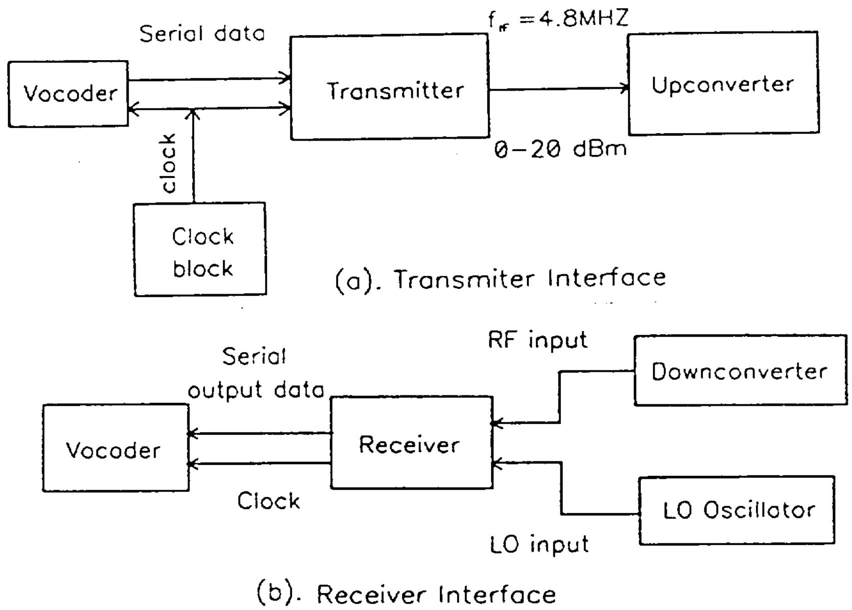


Figure 3.1: System interfaces.

The LO frequency signal mixed with RF frequency signal gives the IF frequency signal. These signals have following specifications:

(a). IF signal frequency: $f_{IF} = f_{RF} - f_{LO} = 4.8$MHz.

(b). Maximum IF bandwidth = 100KHz.

(c). LO signal frequency: f_{LO} = 900~1600MHz with a 1.25MHz step.
LO signal power: P_{LO} = 0~ +5dBm.

(d). RF signal power: P_{RF} = -30dBm±10dBm.
RF signal frequency: Depending on the downconverter.

The outputs of the receiver are the serial output data and its clock. Figure 3.1(b) shows the receiver interfaces.

3.3 IF Frequency

The selection of IF frequency f_{IF} must satisfy the following two conditions.

(1). In order to make the phase continuous at bit transitions in MSK, the IF frequency f_{IF} (or carrier frequency f_c) should be chosen such that f_{IF} is integral multiple of 1/4T, one-fourth the bit rate[9].

(2). The demodulator's analog signal processing section is running at IF frequency f_{IF}. If it is much higher than necessary, as a result, board layout becomes more critical, circuitry becomes very sensitive to component variations and stray reactances, and the performance parameters of many devices are pushed to the limit.

In our design we chose that the IF frequency f_{IF} equals to 4.8MHz. Obviously, it is the integral multiple of 1/4T and not too high so that we can handle easily.

3.4 System Structure

(1). Transmitter Section

Figure 3.2(a) shows the block diagram of the transmitter section. The input signal of transmitter comes from previous stage Vocoder. After convolutional encoding in the chip Q0256 (Qualcomm Inc.), encoded signals are sent to the DDS chip Q2334 (Qualcomm Inc.) through I channel and Q channel control blocks. Our design uses the DDS to generate four different modulated signals instead of using traditional analog-generated method. DDS-generated quadrature signals have significant advantages over analog-generated quadrature signals. These include accurate 90 degrees phase shift and amplitude balance over a wide bandwidth, as well as minimal temperature and aging effect. The digital modulated signals output from chip Q2334 and go through two D/A converters. Then I and Q channel signals are combined together and go to a bandpass filter and are finally sent to the upconverter.

(2). Receiver Section

Figure 3.2(b) and (c) show the block diagrams of the receiver section. Figure 3.2(b) is for BPSK, QPSK and OQPSK. After mixer the IF signal goes through an IF amplifier and gains 30~50 dB. This signal is directly fed to I and Q channels and demodulated. Output signals of the multipliers are filtered out high frequency components and further amplified. Then an A/D converter converts analog signals to 6-bit digital signals which are sent to chip STEL-2110 (Stanford Telecom). The chip STEL-2110 has three fundamental functions: (1) The bit synchronizer produces the clock signals to drive the entire circuit as well as the sampling of the incoming signals. (2) The optimally integrated I and Q signals are used to derive a feedback signal to control a digital Phase Lock Loop (PLL) circuit for carrier tracking. This signal is connected to a microcontroller which is used for calculations and control needed by the digital PLL. (3) Integrated I and Q channel signals are also provided in soft-decision output format which is used to facilitate the inclusion of Forward Error Correction (FEC) using convolutional coding and Viterbi decoding in the system.

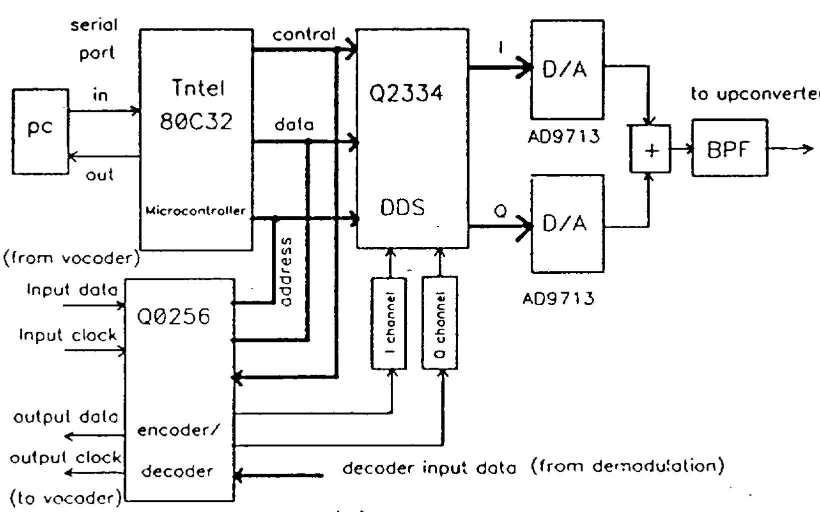
Figure 3.2(c) shows the block diagram of MSK. An easier realized, low cost MSK demodulator is used.

(3). Control Section

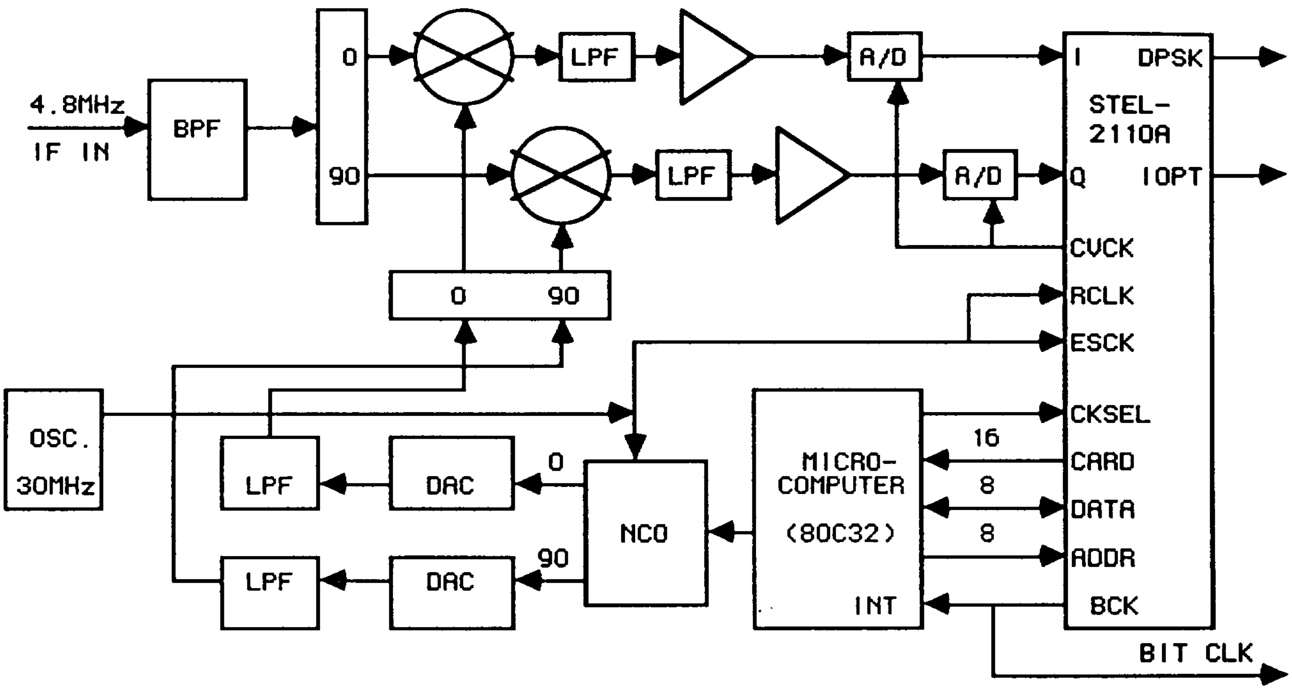
The control circuit section can offer three main features in our design. (1). It can communicate with two input switchs which is used for the selections of modulation/demodulation modes and bit rates. (2). It produces all kinds of clocks for properly system operation and coding/decoding. (3) It can be used to write into or read from the control registers of the VLSI chips used in this system for proper configuration. Besides these, the control circuit also provides a lot of control signals to whole system.

An important task in this project is to develop the control code to write into or read from the internal registers of the VLSI chips for proper configuration and system control.

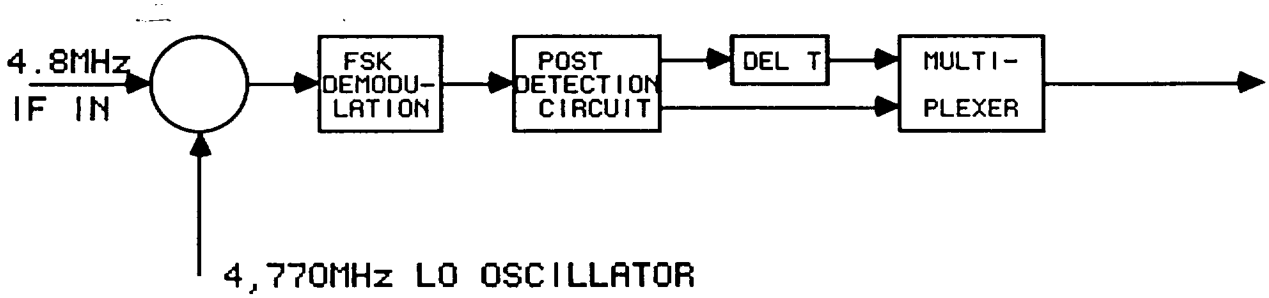
The control software is required to store in an EPROM at the final step.



(a). Transmitter



(b). Demodulator-BPSK, QPSK, OQPSK



(c). Receiver-MSK

Figure 3.2: System structures.

Chapter 4

TRANSMITTER DESIGN

In this chapter we will discuss the design activities which include the modulator design, encoder/decoder, D/A converter and other related designs. We put the decoder design in this chapter just for easy description.

4.1 Modulator Design

The Qualcomm Q2334 Direct Digital Synthesizer (DDS) is used for the modulator to support a wide range of modulation types including BPSK, QPSK,OQPSK and MSK[15][16]. This technique provides fine frequency resolution and phase control, a broad bandwidth of operation, fast frequency switching, good spurious and phase noise performance, and the small size and power consumption.

DDS quadrature signals have significant advantages over analog quadrature signals, including excellent 90 degrees phase shift and amplitude balance over a wide bandwidth, as well as minimal temperature and aging effects. When DDS interfaces with a microprocessor, the intent is to make the modulator sufficiently versatile to allow easy modifications and upgrades.

4.1.1 Q2334 Direct Digital Synthesizer

The Qualcomm Q2334 contains two independent DDS functions controlled from a single microprocessor interface. This interface provides the control for the phase and frequency of the generated sine waves as well as controlling the operating mode of the device. Figure 4.1 shows the internal structure of the Q2334. The value stored in phase increment register A or B is added to the value in the phase accumulator once during each clock period of the reference frequency. The resulting phase value (from 0 to 2π) is converted to a digitized sine wave value by the sine lookup function and this digital value is output from the DDS device.

The DDS is able to generate frequencies from 0 Hz to 1/2 the reference frequency. However, the practical upper limit of the output frequency is about 40% of the reference frequency. To output a particular frequency, the associated phase increment value $\triangle\Phi$ must be loaded into the phase increment registers A or B. The generated frequency F_G and reference frequency (system clock) F_S are related to the phase increment value $\triangle\Phi$ by the following equation:

$$F_G = \frac{F_S \times \triangle\Phi}{2^{32}} \tag{4.1}$$

The frequency resolution is determined by

$$Frequency\ Resolution = \frac{F_S}{2^{32}} \tag{4.2}$$

when F_S=30 MHz, we have the Frequency Resolution = 0.007 Hz.

Table 4.1 gives the register address map for the Q2334.

The Q2334 DDS provides the following modulation features:

(1). External Phase Modulation

External phase modulation operates as an absolute phase adjustment technique. When using this mode the phase increment value for the unmodulated input is written into PIRA. The External Phase Modulation Enable (EPME) bit in the SMC register is set to logic 1 to enable this mode. The phase offset determined by the PM EXT BITs is latched into the DDS function each time the signal PM CLK is asserted. This PM EXT BIT setting causes a phase offset in 45 degrees increments

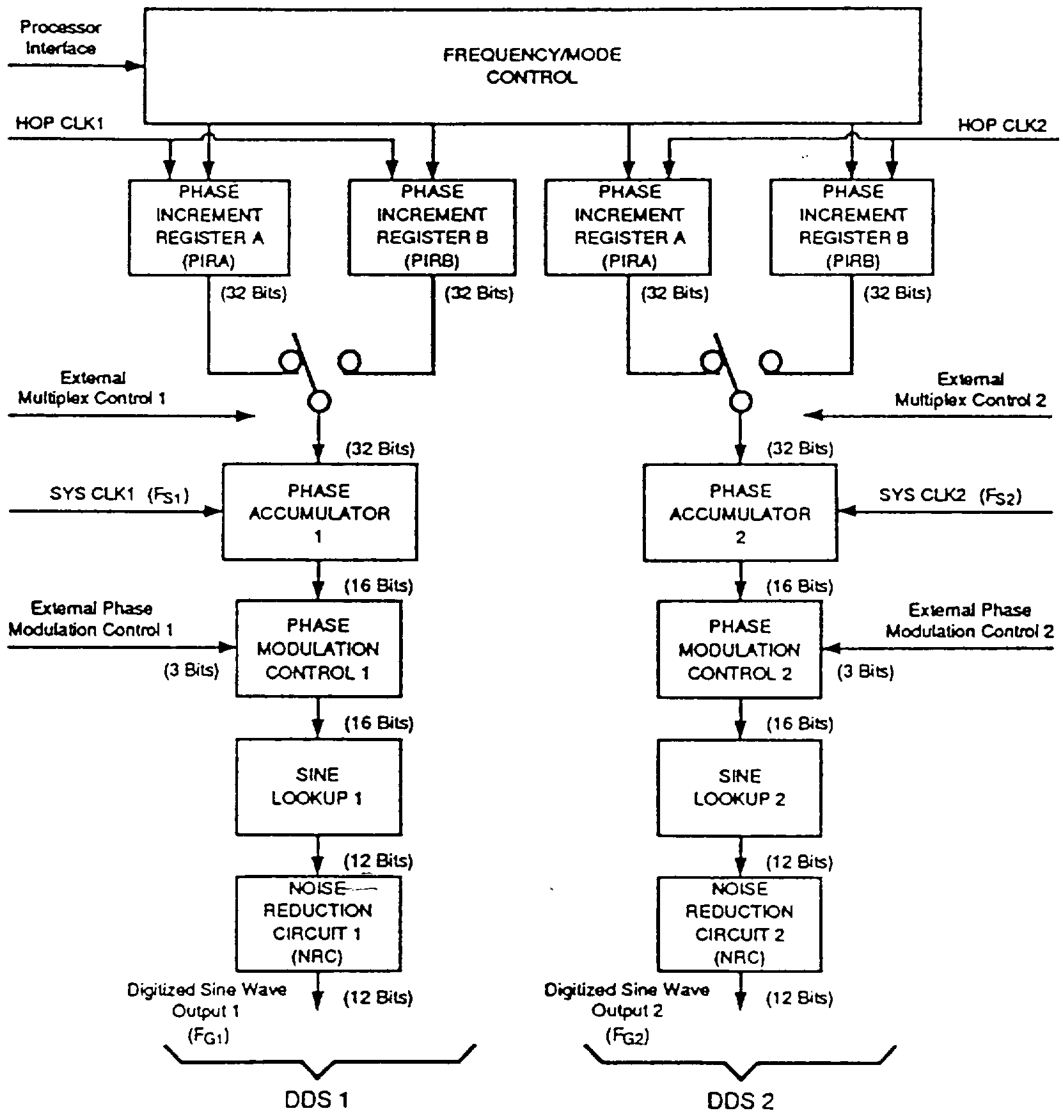


Figure 4.1: Q2334 DDS blok diagram.

DDS1 REGISTER ADDRESS	DDS2 REGISTER ADDRESS	FUNCTION
00H	10H	Phase Increment A (PIRA) bits 0-7 (LSB)
01H	11H	Phase Increment A (PIRA) bits 8-15
02H	12H	Phase Increment A (PIRA) bits 16-23
03H	13H	Phase Increment A (PIRA) bits 24-31 (MSB)
04H	14H	Phase Increment B (PIRB) bits 0-7 (LSB)
05H	15H	Phase Increment B (PIRB) bits 8-15
06H	16H	Phase Increment B (PIRB) bits 6-23
07H	17H	Phase Increment B (PIRB) bits 24-31 (MSB)
08H	18H	Synchronous Mode Control (SMC)
09H	19H	Reserved
0AH	1AH	Asynchronous Mode Control (AMC)
0BH	1BH	Reserved
0CH	1CH	Accumulator Reset Register (ARR)
0DH	1DH	Reserved
0EH	1EH	Asynchronous Hop Clock (AHC)
0FH	1FH	Reserved

Table 4.1: Q2334 interface register address map.

as indicated in Table 4.2 without affecting the operation of the phase accumulator. Using this method we designed the BPSK, QPSK and OQPSK.

(2). Frequency Modulation

Frequency modulation is achieved by using the frequency multiplexer function that selects which PIR register (A or B) is used for accumulation in the phase accumulator function. External Multiplexer Enable (EME) bit in the SMC register is set to logic 1 to enable this mode. The signal EXT MUX controls the selection

PM EXT BIT2	PM EXT BIT1	PM EXT BIT0	ABSOLUTE PHASE OFFSET (degrees)
0	0	0	0
0	0	1	45
0	1	0	90
0	1	1	135
1	0	0	180
1	0	1	225
1	1	0	270
1	1	1	315

Table 4.2: External phase modulation offset setting.

of the value stored in either PIRA or PIRB and the signal MUX CLK enables the selection made by the EXT MUX signal. The selection made by the EXT MUX signal is synchronously activated on the rising edge of the MUX CLK signal. Using this method we designed the MSK.

(3). Internal Modulation

Internal modulation requires use of the processor interface. By storing the synthesizer frequency (basic frequency without phase modulation) in the PIRA and modifying only the most 8 significant bits of the PIRB register, we can obtain a modulator up to 256 states phase modulation.

4.1.2 Modulator Design

(1). BPSK Modulator design

In this design, the external phase modulation mode and only one half of the DDS (DDS1) are used. The design steps are as following:

(a). The EPME bit in the SMC1 register (08H) is set to logic 1 to enable the external phase modulation mode. At same time, we need to disable the second part of the DDS (DDS2).

(b). The phase increment value $\Delta\Phi$ is loaded into the PIRA1 of the DDS1 to generate a unmodulated sine wave whose frequency is 4.8 MHz. According to Equation (4.1), and F_S = 30 MHz and F_G = 4.8 MHZ;

$$\Delta\Phi = \frac{4.8 \times 10^6}{30 \times 10^6} \times 2^{32} = 28F5C28F \quad (Hex)$$

We wrote this value $\Delta\Phi$ to the register PIRA1 of the DDS1.

(c). From Equation (2.3) the BPSK modulated signal $s(t)$ can be represented by

$$s(t) = d_I(t)\cos(2\pi f_c t)$$

where $d_I(t)$ is the input data stream. When:

$d_I(t) = 1$ (high): $\qquad s(t) = \cos(2\pi f_c t) = \sin(2\pi f_c t + 90°)$

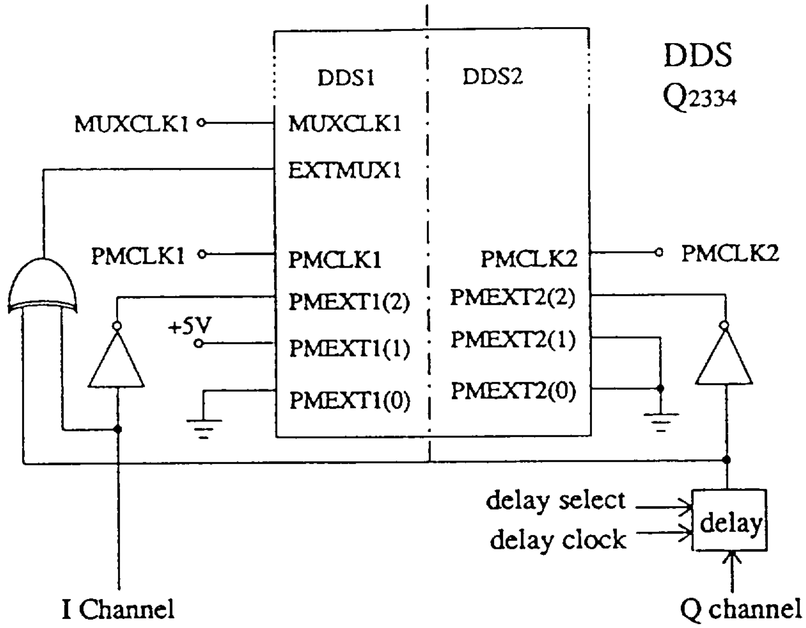


Figure 4.2: The modulator circuit for BPSK, QPSK, OQPSK and MSK.

$d_I(t) = 0$ (low): $\quad s(t) = \cos(2\pi f_c t + 180°) = \sin(2\pi f_c t + 270°)$

According to the Table 4.2 we can see that when setting PM1 EXT BIT 0 = 0 and PM1 EXT BIT 1 = 1 the phase offset of the unmodulated sine wave will only depend on the value of PM1 EXT BIT 2, so that

PM1 EXT BIT 2 = 0: phase offset = 90°

PM1 EXT BIT 2 = 1: phase offset = 270°

Figure 4.2 shows the BPSK modulator circuit. There needs an inverter at pin PM EXT BIT 2 to make I channel coming signal satisfy the phase transition of the BPSK signal.

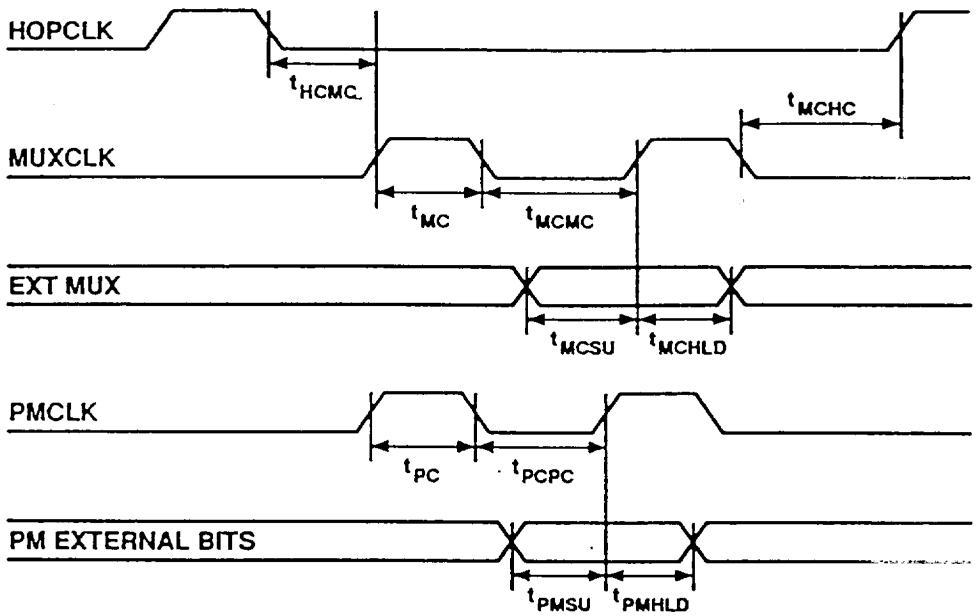
Figure 4.3 shows the external control timing. The signal PM CLK comes from the microcontroller and is used to control the data rate.

(2). QPSK and OQPSK Modulators Design

In QPSK design the external phase modulation mode and two parts of the DDS are used. The design steps are as following:

(a). The EPME bits in SMC1 (08H) and SMC2 (18H) registers are set to logic 1 to enable the external phase modulation mode.

(b). The phase increment value $\Delta\Phi$ is loaded into phase increment registers



SIGNAL	DESCRIPTION
t_{HCMC}	HOP CLK falling to MUX CLK rising
t_{MC}	MUX CLK high period
t_{MCMC}	MUX CLK low period
t_{MCHC}	MUX CLK falling to HOP CLK rising
t_{MCHLD}	EXT MUX setup to MUX CLK
t_{MCSU}	EXT MUX hold after MUX CLK
t_{PC}	PM CLK high period
t_{PCPC}	PM CLK low period
t_{PMSU}	PM data setup to PM CLK
t_{PMHLD}	PM data hold after PM CLK

Figure 4.3: External control timing.

PIRA1 and PIRA2 of the DDS to generate unmodulated sine waves whose frequencies are equal to 4.8 MHz. According to equation (4.1)

$$\Delta\Phi = \frac{F_G}{F_S} \times 2^{32} = 28F5C28F \qquad (Hex)$$

where $F_G = 30MHz$, and $F_S = 4.8MHz$.

(c). From Equation (2.6) the QPSK modulated signal $s(t)$ can be written by:

$$s(t) = d_I(t)\cos(2\pi f_c t) + d_Q(t)\sin(2\pi f_c t)$$

Because the QPSK signal can be represented as the summation of the orthogonal BPSK signals, we can design I channel and Q channel separately.

I channel: It is identical with the BPSK design. we can use the same configuration and circuit to accomplish the I channel function of QPSK.

Q channel: when

$d_Q(t) = 1$ (high): $\sin(2\pi f_c t)$

$d_Q(t) = 0$ (low): $\sin(2\pi f_c t + 180°)$

According to the Table 4.2 we can see that when setting PM2 EXT BIT 0 = 0 and PM2 EXT BIT 1 = 0, the phase offset of the unmodulated sine wave of Q channel will only rely on the value of PM2 EXT BIT 2. So we have

PM2 EXT BIT 2 = 0 (low): phase offset = 0

PM2 EXT BIT 2 = 1 (high): phase offset = 180°

Figure 4.2 shows the QPSK modulator circuit. QPSK has the same external control timing as BPSK shown in Figure 4.3.

OQPSK modulator is identical with QPSK modulator except there is a delay of T, the one half symbol during, at Q channel input signal. A dual D Flip-Flop (74LS74A) is used to design this delay function.

(3). MSK Modulator Design

In MSK design the frequency multiplexer function and only one half of the DDS (DDS1) are used. The following is the design steps:

(a). The EME bit in SMC1 register (08H) is set to logic 1 to enable the external multiplex control. Meanwhile, we need to disable the second part of the DDS.

(b). There are two phase increment values $\triangle\Phi_+$ and $\triangle\Phi_-$ which must be loaded into phase increment registers PIRA1 and PIRB1 of the DDS1 in MSK.

$$\begin{aligned} \triangle\Phi_+ &= \tfrac{f_+}{F_S} \times 2^{32} \\ \triangle\Phi_- &= \tfrac{f_-}{F_S} \times 2^{32} \end{aligned} \tag{4.3}$$

where $f_+ = f_c + \frac{1}{4T}$; $f_- = f_c - \frac{1}{4T}$ and $\frac{1}{T}$ is the channel data rate which is double input data rate because there is an encoder with 1/2 code rate in the system.

With different data rate the $\triangle\Phi_+$ and $\triangle\Phi_-$ are different. For example,

If $\frac{1}{T}$ = 19200 bps (input data rate = 9600 bps)

$$\begin{aligned} f_+ &= f_c + \tfrac{1}{4T} = 4804800\ Hz \\ f_- &= f_c - \tfrac{1}{4T} = 4795200\ Hz \end{aligned}$$

and

$$\begin{aligned} \triangle\Phi_+ &= \tfrac{4804800}{30000000} \times 2^{32} = 29003EEA \quad (Hex) \\ \triangle\Phi_- &= \tfrac{4795200}{30000000} \times 2^{32} = 28EB4635 \quad (Hex) \end{aligned}$$

If $\frac{1}{T}$ = 9600 bps (input data rate = 4800 bps)

$$\begin{aligned} f_+ &= f_c + \tfrac{1}{4T} = 4802400\ Hz \\ f_- &= f_c - \tfrac{1}{4T} = 4797600\ Hz \end{aligned}$$

and

$$\begin{aligned} \triangle\Phi_+ &= \tfrac{4802400}{30000000} \times 2^{32} = 28FB00BD \quad (Hex) \\ \triangle\Phi_- &= \tfrac{4797600}{30000000} \times 2^{32} = 28F08463 \quad (Hex) \end{aligned}$$

(c). From Equation (2.13) the MSK modulated signal $s(t)$ can be written by

$$s(t) = \cos\left(2\pi f_c t + b_k(t)\frac{\pi t}{2T} + \Phi_k\right)$$

where $b_k(t) = -d_I(t)d_Q(t)$, and Φ_k is an initial phase. From this Equation, we have:

If $d_I(t)$ and $d_Q(t)$ are opposite, $b_k(t)$ =1 (high)

If $d_I(t)$ and $d_Q(t)$ are same, $b_k(t) = -1$ (low)

The relationship between $d_I(t)$ and $d_Q(t)$ is the exclusive-OR function. Practically, we use an exclusive-OR gate to realize this function.

When the external multiplex control is enable we have the following features in Q2334 DDS. If the EXT MUX signal is high when the MUT CLK is asserted the phase accumulator accumulates phase increments from the PIRB register. If the EXT MUX signal is low when the MUX CLK is asserted the phase accumulator accumulates phase increments from the PIRA register. Changing the value of the EXT MUX input therefore causes the alternation between the frequency controlled by the PIRA and the frequency controlled by the PIRB. In this way, we have to write the value of $\triangle\Phi_+$ into register PIRB1 to generate the first frequency and the value of $\triangle\Phi_-$ into register PIRA1 to generate the second frequency, respectively. The output of the exclusive-OR gate is connected to the pin EXTMUX1. Following above rules, we have:

$d_I(t)$ and $d_Q(t)$ are opposite $\Rightarrow b_k(t) = 1 \Rightarrow EXTMUX1 = high \Rightarrow PIRB1$

$d_I(t)$ and $d_Q(t)$ are same $\Rightarrow b_k(t) = 0 \Rightarrow EXTMUX1 = low \Rightarrow PIRA1$

Figure 4.2 shows this design and Figure 4.3 provides the external control timing. The signal MUX CLK comes from the microcontroller and is used to control the data rate.

4.2 Encoder/Decoder Design

The Qualcomm Q0256 is used as encoder and decoder in our design[17]. The Q0256 provides:

(a). On-chip convolutional encoder/Viterbi decoder, differential encoder/decoder, and V.35 data scrambler/descrambler.

(b). Processing data at one of four selectable code rates (1/2, 1/3, 3/4 and 7/8).

(c). Built-in synchronization capability for BPSK, QPSK and OQPSK modems and operating with either 1 bit hard-decision or 3-bit soft-decision.

(d). Two powerful techniques for monitoring synchronization status as well as performing channel bit error rate measurement.

(e). 5.2 dB coding gain (rate 1/2) at 10^{-5} BER.

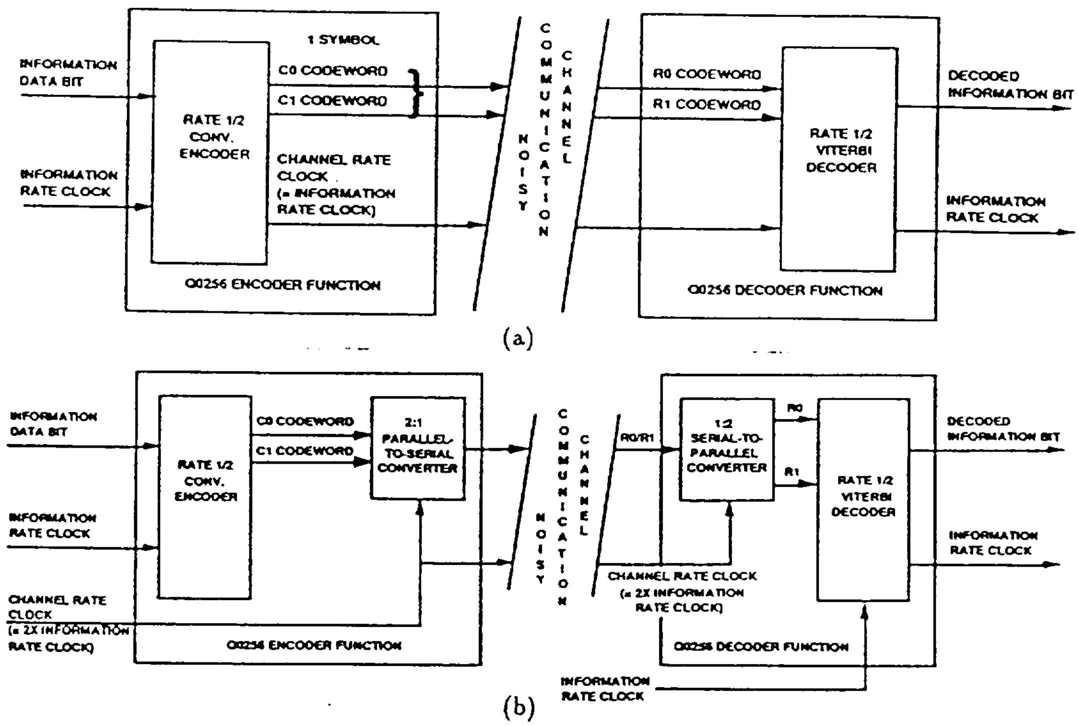


Figure 4.4: (a). Parallel data mode, (b). Serial data mode.

4.2.1 Parallel and Serial Data Modes

The Q0256 provides two kinds of data modes: "parallel" and "serial", as shown in Figure 4.4.

The Q0256 encoder produces two encoded bits with code rate 1/2 for each information input bit. When operating in the parallel data mode these two output bits are presented at C0 and C1 output pins during each period of the channel rate clock (ENCOUTCLK). In this case, the ENCOUTCLK frequency should be the same as the frequency of information rate clock (ENCINCLK). When operating with serial data mode all encoded bits are provided on the single output pin C0 at the period of the ENOUTCLK signal. In this mode, the ENOUTCLK frequency should be twice the ENCINCLK frequency.

The Q0256 decoder inputs data in either serial or parallel mode. When operating in the parallel data mode with code rate 1/2, two input codewords are provided in the R0 and R1 input pins during each period of the DECINCLK. When operating in the serial mode, the decoder inputs all encoded data using only the R0 input pin. The relationship of the DECINCLK to DECOUTCLK frequencies is the reciprocal of the relationship of the encoder ENCINCLK to ENCOUTCLK frequencies.

Design rules (code rate 1/2):

(1). Parallel mode (QPSK, OQPSK and MSK)

ENCINCLK = ENCOUTCLK

DECINCLK = DECOUTCLK

(2). Serial mode (BPSK)

2×ENCINCLK = ENCOUTCLK

$\frac{1}{2}$×DECINCLK = DECOUTCLK

4.2.2 Synchronization Status Monitor Design

The Q0256 can automatically synchronize incoming data streams to the Viterbi decoder circuit. The synchronization technique is a two-step process.

(1). Detect: The decoder quality state is constantly monitored by using the "state metric normalization rate" circuit. The designer programs an "in-sync/out-of-sync" threshold for this internal circuit. The success or failure of this test for each test period is indicated on output pins 53 (INSYNC) and 52 (OUTOFSYNC).

(2). Correct: The OUTOFSYNC output pin can be directly connected to the SYNCCHNG input pin. This provides a feedback path between the synchronization monitor and the synchronization correction circuit. The effects of the out-of-sync condition can be compensated for either by a timing re-alignment or by permutation of the decoder input data.

The normalization circuit consists of two counters:

- T counter measures the number of decoded bits.
- N counter measures the number of state metric normalizations.

(1). Design Rules:

(a). The actual number of decoded bits in the normalization teat period is

$$T \times 256 \tag{4.4}$$

(b). The actual number of normalizations allowed is

$$(N-1) \times 8 + 4 \tag{4.5}$$

(c). The normalization rate threshold is

$$\frac{(N-1) \times 8 + 4}{T \times 256} \tag{4.6}$$

where T and N are the two's complement values of the 8-bit numbers loaded into the T and N counters.

(2). Conditions:

(a). When operating with rate 1/2 coding, a normalization rate threshold of about 10% will reliably detect a loss of synchronization.

(b). The normalization measurement should detect at least 20-30 normalizations before declaring a loss of synchronization.

(3). Design:

(a). Select the number of normalizations to be detected to be approximately 50. So we set:

$$N = 7$$

Because

$$(N-1) \times 8 + 4 = (7-1) \times 8 + 4 = 52$$

the binary value which is loaded into N counter is F9 (Hex).

(b). Because the value for the T counter must be approximately ten times the value in the N counter, we set:

$$T = 2$$

Because

$$T \times 256 = 2 \times 256 = 512$$

the binary value which is loaded into T counter is FE (Hex).

(c). The normalization rate threshold is

$$\frac{52}{512} \times 100\% = 10.2\%$$

It is satisfied the condition (a).

(4). Experiment results:

Through many times of experiments, We found that for different modulation modes and different bit rates, the values in N counter and T counter vary from the theoretical values for best system performance. These values are given in the software-MDCOB. Also, the procedure of loading values into the internal registers of Q0256 is fixed for proper system operation. The software MDCOB is in consistance with this fixed procedure.

4.2.3 Monitoring Channel Bit Error Rate (BER)

The on-chip BER monitor circuit consists of two accumulators acting as counters. One accumulator counts decoder input codewords. Another accumulator counts codeword errors detected by the on-chip re-encode and compare circuit. The design steps are as following:

(1). Set BER measurement period register. The loaded value is the two's complement 24-bit binary value and is multiplied by 1000 to give the actual number of codewords to be monitored. For example, if the actual number is 10^7, the two's complement binary value FFD8F0 (Hex) of 10^4 is loaded into addresses 0CH, 0BH and 0AH.

(2). When the BER measurement period is completed, the signal BERDONE (pin 50) goes to high for two periods of DECOUTCLK. It can be used as an interrupt status bit to microprocessor. The actual measured bit error count is found from the following formula:

$$Actual\ Error\ count = (register\ value - 1) \times 8 \tag{4.7}$$

(3). The actual symbol BER is

$$BER = \frac{the\ measured\ error\ quantity}{the\ number\ of\ codewords\ in\ the\ test} \tag{4.8}$$

4.2.4 The Other Considerations

(1). The 3-bit soft-decision values can be fed to the Q0256 decoder inputs in either sign-magnitude or offset-binary notation. The selection of the input format is made via the microprocessor interface. We used offset binary format in our design.

(2). Enable the on-chip differential encoder/decoder, and data scrambling/descramblin circuits. This is also made via the microprocessor interface.

(3). The Q0256 processor interface has 4 read registers and 21 write registers. Carefully setting these registers we can configure the different operating modes for encoder and decoder.

(4). In our design, there is a mechanism that can interchange the demodulated I-channel data and Q-channel data before these data go into the decoder. This is because we found that these data need be interchanged for some modulation modes for proper operation.

4.3 D / A Converter

We selected AD9713 (Analog Devices) as the digital to analog converter[19]. The AD9713 has the following features:

1). 12-bit resolutions

2). TTL-compatible

3). Fast setting

4). 80 MSPS update rate

5). Low power consumption

Figure 4.5 shows the actual circuit that we used in our design for D/A converter.

(1). Setting the Reference

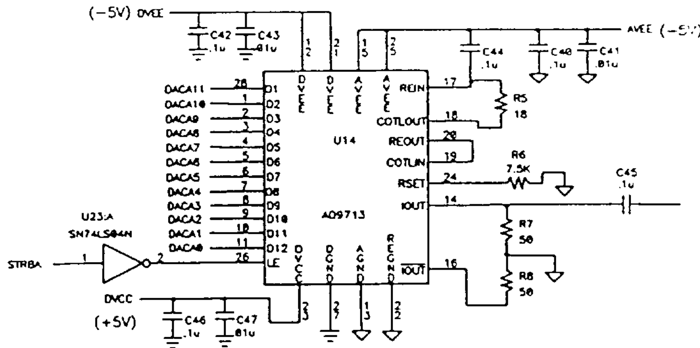


Figure 4.5: D / A converter circuit.

We used the internal reference that allows operation with a minimum of external components in our design. When using the internal reference:

a). PEOUT (pin 20) should be connected to COTLIN (pin 19);

b). COTLOUT (pin 18) should be connected to REIN (pin 17) through as 18 Ω resistor;

c). A 0.1 uF capacitor from pin 17 to -Vs (pin 15) improves setting by decoupling switching noise from the current sink base line;

d). R_{SET} (pin 24) should be connected to ground through a 7.8 KΩ resister. This determines the Full-scale current out.

(2). Outputs

The switch network controls complementary current outputs I_{out} and $\overline{I_{out}}$. The current output can be converted to a voltage output by resistive loading as shows in Figure 4.5. Both I_{out} and $\overline{I_{out}}$ should be loaded equally for best overall performance.

Full-scale output current $I_{out(FS)}$ is determined by

$$I_{out(FS)} = \frac{Reference\ Voltage}{R_{SET}} \times 128 \qquad (4.9)$$

The internal reference is nominally -1.26 V with a tolerance of ± 10%, and R_{SET} = 7.5 KΩ, so

$$I_{out(FS)} = -20.48 \ \ mA$$

The voltage which is developed is the product of the output current and the value of the load resistor.

$$V_{out(FS)} = -20.48 \times 10^{-3} \times 50 = -1.024 \;\; V$$

The voltage swing will be from 0 to -1.024 V across 50 Ω resistor.

(3). Power and Grounding

Maintaining low noise on power supplied and ground is critical for obtaining optimum results with the AD9713. We separate the analog ground plane with digital ground plane, and also isolate digital power supply with analog power supply. Figure 4.6 shows this effort.

4.4 Lowpass Filter

Since this is a sampled data system, spectral components will be generated at all the frequencies $nf_{clk} \pm f_c$, where n is an integer. $n = 0$ gives the carrier frequency, and the frequencies given by all other values of n are above the Nyquist frequency (half the sampling frequency). The design is made on the board for an anti-aliasing (low pass) filter at the output of the combiner to attenuate these spurious signals. This filter has up to 3 sections (7th. order), and is a pole and zero type (cauer). Cauer (elliptic) filters are recommended because of their superior characteristics[20].

The maximum cutoff frequency is given by the equation:

$$f_{out} = \frac{f_{out}}{1 + tr} \tag{4.10}$$

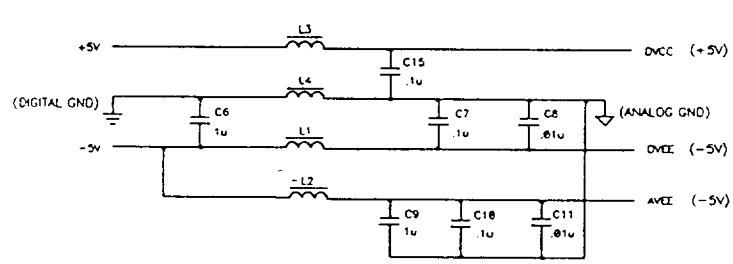


Figure 4.6: Power supply.

where tr is the transition ratio of the filter. Assuming that about 1 dB of ripple is allowable in the passband and 55-60 dB of attenuation is required in the stop-band, a 7 pole (3 sections) filter have a tr of about 1.17, and the maximum value of f_{out} is

$$f_{out} = \frac{f_{clk}}{1 + 1.17} = \frac{30 \times 10^6}{2.17} = 13.9 \ \ MHz$$

where $f_{clk} = 30 \ MHz$. Figure 4.7 (a) shows the lowpass filter circuit (50 Ω impedance) and (b) is the frequency response of this lowpass filter.

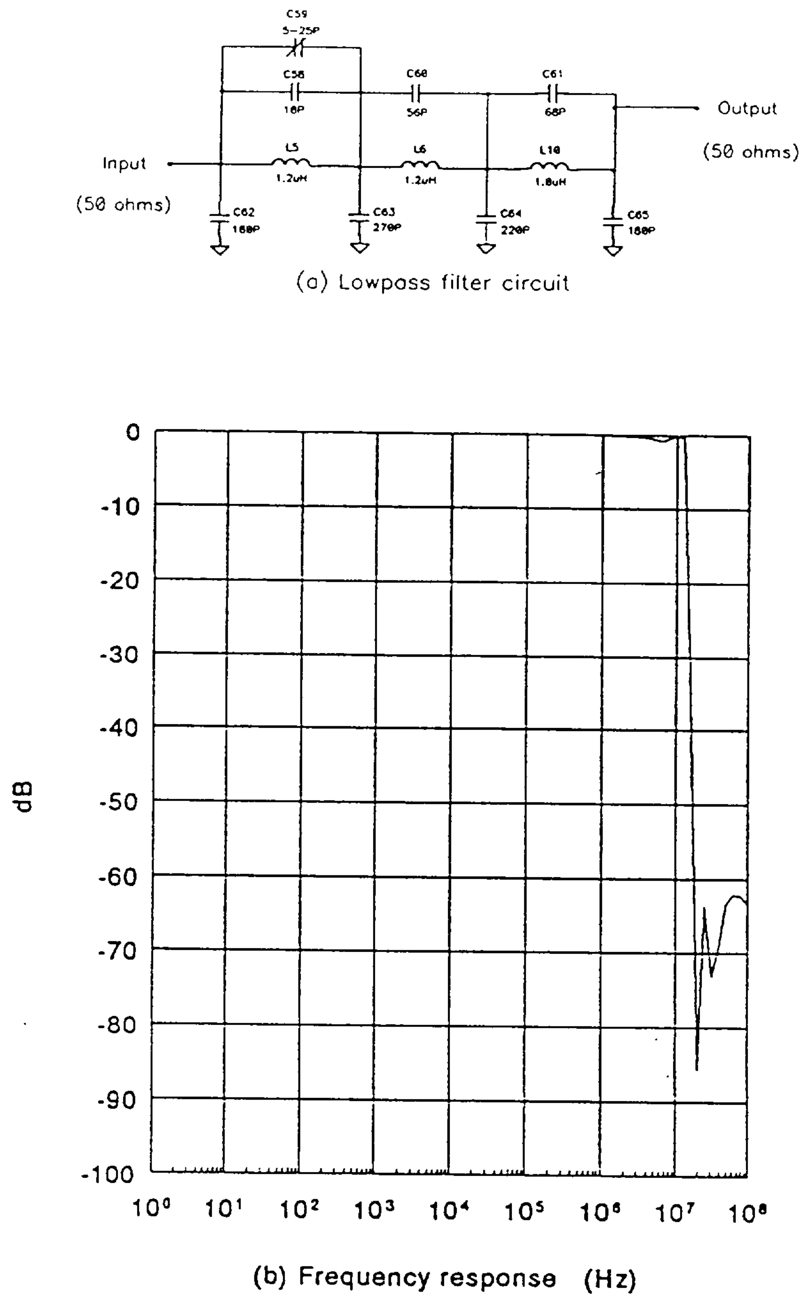


Figure 4.7: (a) lowpass filter circuit, (b) frequency response.

Chapter 5

RECEIVER DESIGN

In this chapter, we will discuss two kinds of demodulator design, one for BPSK, QPSK and OQPSK, and another for MSK. Figure 3.2(b) and (c) show these two demodulator block diagrams.

5.1 IF Amplifier and Bandpass Filter

Mainly using the MOTOROLA MC1350 chip, this circuit combines IF amplifier and bandpass filter (BPF) together. MC1350 is a monolithic IF amplifier chip featuring wide range AGC, nearly constant input and output admittances over the entire AGC range and low reverse transfer admittance[27]. Operating at required center frequency 4.8 MHz and 3-dB bandwidth 100 KHz, this circuit can realize power gain about 45 dB (with input = 0.01 v) and has low noise and linear amplifying features. Figure 5.1 shows this circuit.

There are several points which should be noted:

(1). There are two self-resonant loops in this circuit (C_{67}, C_{68}, L_7 and C_{71}, T_1 with C_{68}' and C_{71}' micro-adjustment). Theoretically, it should be better to make both of these two loops resonant at center frequency. In practice, it may be discovered C_{71} and the choke T_1 are more sensitive to the center frequency while C_{67}, C_{68} and

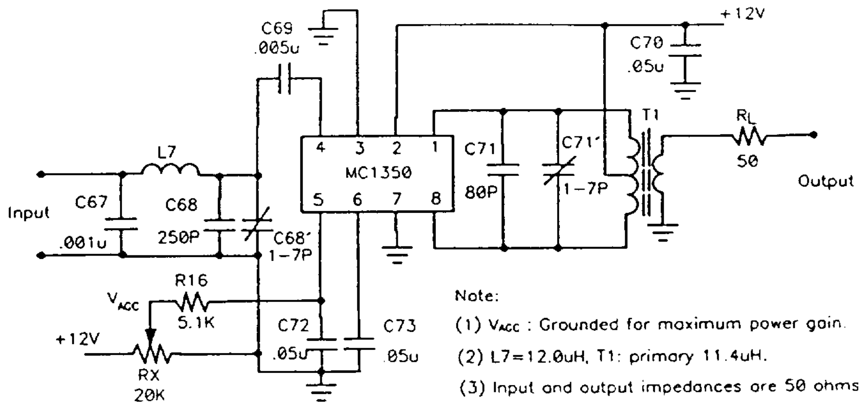


Figure 5.1: IF amplifer and BPF.

L_7 are more related with the range of 3-dB bandwidth. When this circuit is running properly, only the loop of C_{71} and T_1 resonates at center frequency (4.8 MHz). And increasing the value of C_{67} or C_{68}, we will have narrower bandwidth.

(2). T_1 is wound with #36 AWG. It's primary winding is about 10-12 turns and secondary winding about 2-3 turns. The number of primary winding turns determines the center frequency while the number of secondary winding turns is related with power gain.

(3). All the system parameters are measured with V_{AGC} grounded for maximum power gain. The value of V_{AGC} has little influence on the center frequency and 3-dB bandwidth.

The frequency response curve of this circuit is shown in Figure 5.2 (V_{AGC} grounded).

5.2 Multiplier, LPF and Amplifier

The chip XR-2208 (EXAR) is used for the purposes of multiplier, LPF and amplifier as shown in Figure 5.3. The XR-2208 contains a four quadrant multiplier and an independent Op Amp[30]. The main features are maximum versatility, excellent linearity and wide bandwidth.

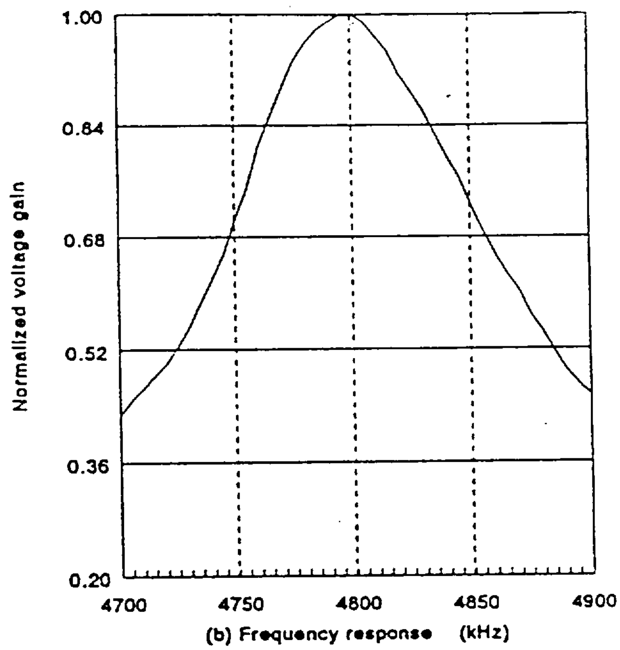


Figure 5.2: The frequency response curve of the IF amplifier and BPF.

5.2.1 Multiplier

The multiplier section of the XR-2208 can be used as a synchronous detector. There are two input signals:

(1) IF input signal V_X: IF frequency = 4.8 MHz, input power level = 0 dB (approximately).

(2) Reference signal V_Y: a square wave coming from the carrier recovery

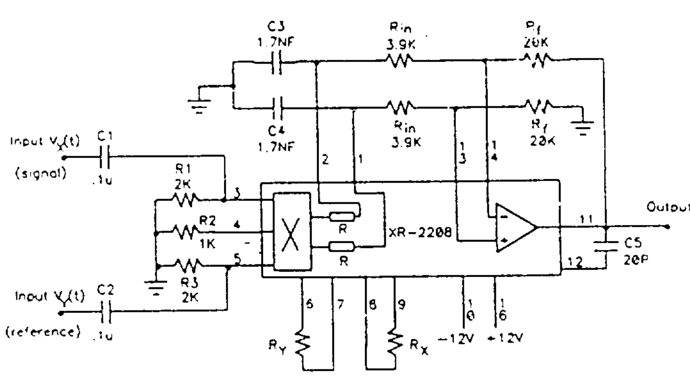


Figure 5.3: Multiplier, LPF and amplifier.

circuit.

The differential output voltage V_d , across the pins 1 and 2, is proportional to the product of voltages V_X and V_Y applied to the inputs. The V_d can be expressed as

$$V_d \approx \left(\frac{25}{R_X R_Y}\right)(V_X V_Y) \tag{5.1}$$

where all voltages are in volts and resistors are in KΩ. R_X and R_Y are the gain control resistors for X and Y sections of the multiplier. From above, the gain constant of the multiplier section K_m can be expressed as

$$K_m = \frac{25}{R_X R_Y} \tag{5.2}$$

If $R_X = 0.4$ KΩ and $R_Y = 3.6$ KΩ, we have $k_m = 6.25$. The resistors R_X and R_Y are selected so that the circuit can never become saturated.

Now, consider an IF input signal as

$$V_X(t) = V_X \sin(\omega_i t + \theta_i) \tag{5.3}$$

and the square wave reference signal from the carrier recovery circuit is

$$V_Y(t) = V_Y \sum_{n=0}^{\infty} \frac{4}{\pi(2n+1)} \sin\left[(2n+1)\omega_r t\right] \tag{5.4}$$

where ω_i is the IF frequency, ω_r is the reference signal frequency and θ_i is the phase in relation to the reference signal.

Multiplying these two terms, using the appropriate trigonometric relationships, gives:

$$\begin{aligned} V_d(t) = \tfrac{2K_m}{\pi}\Big[& \textstyle\sum_{n=0}^{\infty} \frac{V_X V_Y}{(2n+1)} \cos[(2n+1)\omega_r t - \omega_i t - \theta_i] \\ & - \textstyle\sum_{n=0}^{\infty} \frac{V_X V_Y}{(2n+1)} \cos[(2n+1)\omega_r t + \omega_i t + \theta_i] \Big] \end{aligned} \tag{5.5}$$

If ω_r is close to ω_i, the first term (n=0) has a low difference frequency component. As ω_r is driven closer to ω_i this difference becomes smaller until $\omega_r = \omega_i$ and lock is achieved. The first term then becomes:

$$V_d(t) = \frac{2K_m V_X V_Y}{\pi} \cos\theta_i$$

Other terms are of high frequencies and are rejected by the loowpass filter.

5.2.2 LPF and Amplifier

The equivalent circuit for LPF and amplifier is shown in Figure 5.4.

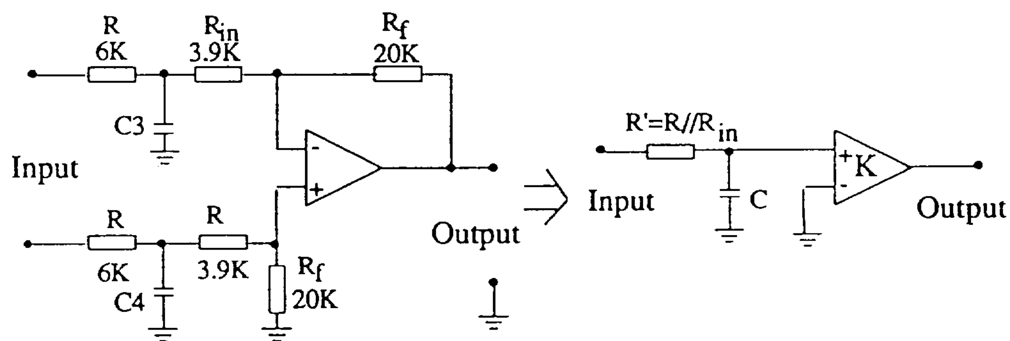


Figure 5.4: The equivalent circuit for LPF and amplifier.

where R is the internal resistor of the XR-2208, gain K for the amplifier

$$K = -\frac{R_f}{R + R_{in}} \approx -2$$

and

$$R' = \frac{R \times R_{in}}{R + R_{in}} = 2.36\ K\Omega$$

Let @-3dB cutoff frequency f_{cut} of the lowpass filter be 40 KHz (>38.4KHz), so

$$C = \frac{1}{2\pi f_{cut} R'} = 1700\ PF$$

We chose 1500 PF~1800 PF for C_3 and C_4. If needed, the amplifier gain can be increased by changing the resistors R_{in} and R_f.

5.3 Analog to Digital Converter

After synchronous detection stage, the analog signals from I and Q channels should be converted to digital signals, which are then fed to the STEL-2110 chip for further processing. We have known that the maximum inputs from I and Q channels are $\pm$ 0.5 V_{p-p} bipolar analog signals, and outputs of STEL-2110 should be digital signals with offset binary format. The AD9058 (Analog Devices) is selected for the

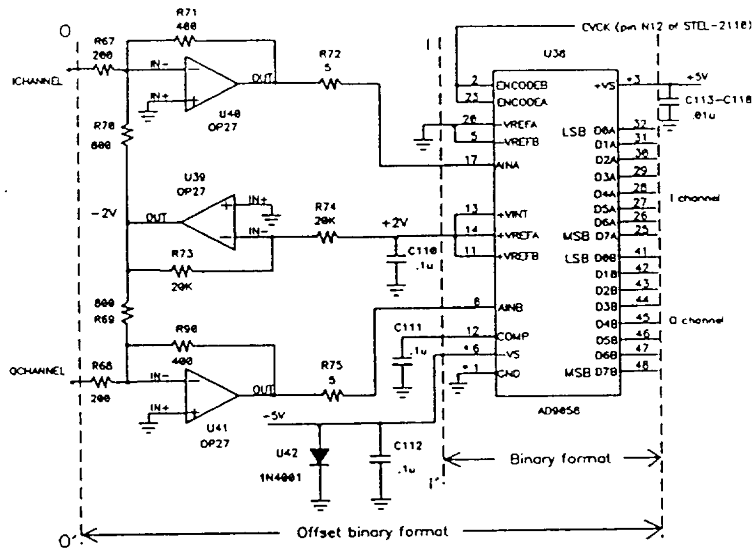


Figure 5.5: The analog to digital converter.

purpose of converting the analog signal to the digital signal[19]. It combines two independent high performance 8-bit analog-to-digital converters (ADCs) on a single monolithic IC. Analog input range is established by the voltages applied at the voltage reference inputs (+V_{ref} and -V_{ref}). The AD9058 can operate from 0V to 2V using the internal voltage reference, or anywhere between -1V to +2V using external reference.

Figure 5.5 shows the analog-to-digital converter circuit. In our design, the internal voltage reference was used for reducing the number of external components. The input range of the ADCs are positive unipolar in this configuration, ranging from 0V to +2V. The bipolar input signals are buffered, amplified and offset into the proper input range of the ADC using three low distortion amplifiers such as OP27.

We have: $amplifier\ gain = 2$ and $offset\ voltage = 1V$.

In this case, the bipolar ±0.5V input signals are changed into the unipolar positive 0V to +2V input signals. The output signal format is binary format (unipolar) with respect to the input signals I—I' and is offset binary format (bipolar) with respect to the input signals O—O'.

The sampling clock comes from the pin N12 (CVCK) of the STEL-2110chip. The diode between ground and -V_S is normally reverse biased and is used to prevent latch-up.

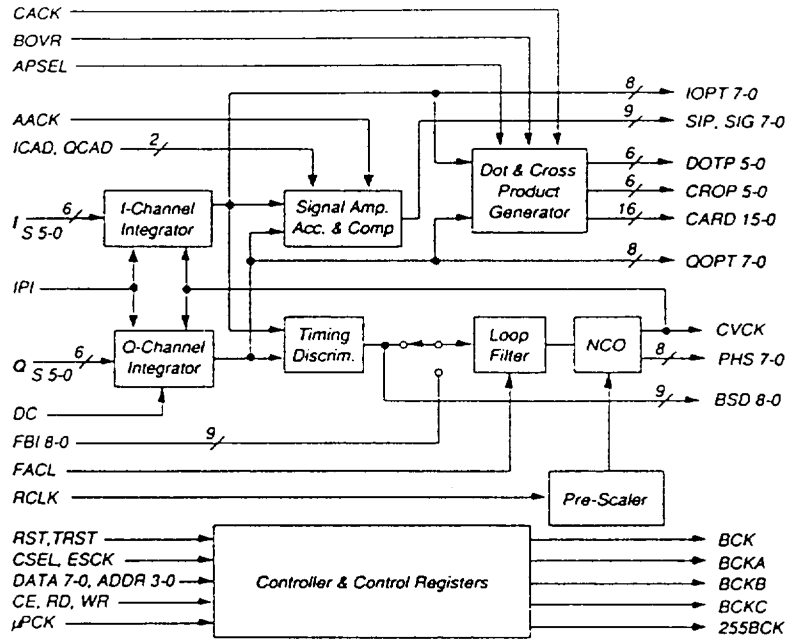


Figure 5.6: Block diagram of the STEL-2110.

5.4 STEL-2110

5.4.1 STEL-2110 Bit Synchronizer/PSK Demodulator

Figure 5.6 shows the block diagram of the STEL-2110. The STEL-2110 has three fundamental functions: (a) Bit timing recovery circuit, (b) Carrier tracking feedback, (c) Integrated I and Q channel output signals[21][22].

(1) Bit timing recovery circuit

The bit synchronizer portion of the STEL-2110 is a digital phase locked loop which operates by integrating the input signals in both the I and Q channels over one symbol period. This is done 3 times: in addition to the nominally "on time" integration, "quarter period early" and "quarter period late" integrations are also carried out. The difference between the early and late integrations gives an indication of the timing error, since the averaged difference will be zero when the timing is correct. This signal is passed into the loop filter which is effectively a second order filter and then used to drive a numerically controlled oscillator (NCO) which produces the clock signals to drive the entire circuit as well as sampling the incoming signals.

The NCO has 28-bit frequency resolution and is preset to the nominal symbol frequency at the start-up. The preset data is a 24-bit word and is loaded into the 24 MSBs of the 28-bit accumulator. The data from the loop filter is added to this word so that the data from the loop filter modifies the 23 LSBs of the 28-bit phase accumulator.

In our design, this circuit provides the bit timing for all four demodulations.

(2) Carrie tracking feedback circuit

The punctual I and Q signals from the I and Q channel integrator circuits are processed in the carrier discriminator circuit. The dot and cross produce of the I and Q signals are first formed, where:

$$Dot\ product = I_n \times I_{n-1} + Q_n \times Q_{n-1} \tag{5.6}$$

and

$$Cross\ product = I_n \times Q_{n-1} - Q_n \times I_{n-1} \tag{5.7}$$

both signals are used to form the carrier discriminator functions.

In the automatic frequency control (AFC) mode, this is

for BPSK data:

-Sign (Dot) $\times$ Cross

for QPSK data:

-Sign (Dot) $\times$ Cross,   if | Dot | > | Cross |

Sign (Cross) $\times$ Dot,   else

In the phase locked loop (PLL) mode, this is

for BPSK data:

-Sign (I) $\times$ Q

for QPSK data:

-Sign (I_{rot}) $\times$ Q_{rot},   if | I_{rot} | > | Q_{rot} |

Sign (Q_{rot}) $\times$ I_{rot},   else

where I_{rot} and Q_{rot} are the I and Q vectors rotated by 45°.

In our design, the PLL mode is selected since it is intended for coherent demodulation in continuous carrier systems. This function is integrated under the control of the carrier accumulate clock (CACK) to form the discriminator output, which is available on the 16-bit bus. The 12-bit of them is connected to a digital to analog converter to drive a loop filter which in turn drives the frequency control of the local oscillator. This is the method we used to design the carrier recovery circuit for BPSK, QPSK and OQPSK. For MSK, we used the self-synchronization feature of MSK to generate the reference carrier signals.

(3) The I and Q channel output signals

The punctually integrated I and Q channel signals are provided as the outputs in 8-bit soft-decision format which is used to facilitate the inclusion of forward error correction using Viterbi deciding in the system. The 8-bit integrated outputs are represented in offset two's complement format.

5.4.2 Design with STEL-2110

(1) Bit rate control

Addresses 12_H, 13_H and 14_H are the bit rate control register. The 28-bit NCO is programmed with these 3 bytes, which are loaded into the 24 MSBs of the 28-bit phase increment register. Bit 7 of address 12_H is the MSB, and bit 0 of address 14_H is the LSB. The formula for N_r, the number programmed in the NCO, is as follows:

$$A_r = R_s \times S_s \times A_i \tag{5.8}$$

and

$$N_r = \frac{A_r}{f_c} \times 2^{24} \tag{5.9}$$

where:

N_r: 24 bit number which establishes the nominal A/D sample rate.

f_c: NCO input clock frequency $f_c = \frac{reference\ clock}{C_s}$ (after scaling the input clock by C_s= 16).

A_r: A/D converter clock rate.

R_s: symbol rate of PSK information to be demodulated.

S_s: number of accumulated samples per symbol.

A_i: number of front end accumulator.

We set:

$f_c = \frac{30\,MHz}{16} = 1875000\ Hz$

$S_s = 4$

$A_i = 1$

For QPSK, OQPSK and MSK (parallel mode)

R_s =1200, 2400, 4800, 9600, 19200 bps

For BPSK (serial mode)

R_s =2400, 4800, 9600, 19200, 38400 bps

The N_r has the different value with the different symbol rates. In our design, for example, if R_s =9600 bps

$$A_r = 9600 \times 4 \times 1 = 38400$$

and

$$N_r = \frac{38400}{1875000} \times 2^{24} = 053E2D \quad (Hex)$$

The NCO is not double buffered and will immediately switch to the newly programmed frequency after any of the bytes are changed. Do not set the 24-bit data to be 000000_H at any time as this will set the NCO output frequency to zero, causing the entire chip to freeze up, requiring to restart the chip.

(2) Loop gain control

Address 11_H is the loop gain control register. Figure 5.7 is a simplified block diagram of the bit timing feedback loop system. The value of the loop constant K is a function of the chip setup parameters and signal input conditions. The parameters K_1 and K_2 are controlled by the loop gain control register (11_H). For the case, when an AGC controls the level of the input signal the formula for K can be used as following

$$K = \frac{A \times A_i \times S_s \times T_d \times P_a^2}{8 \times N_r \times b} \tag{5.10}$$

where:

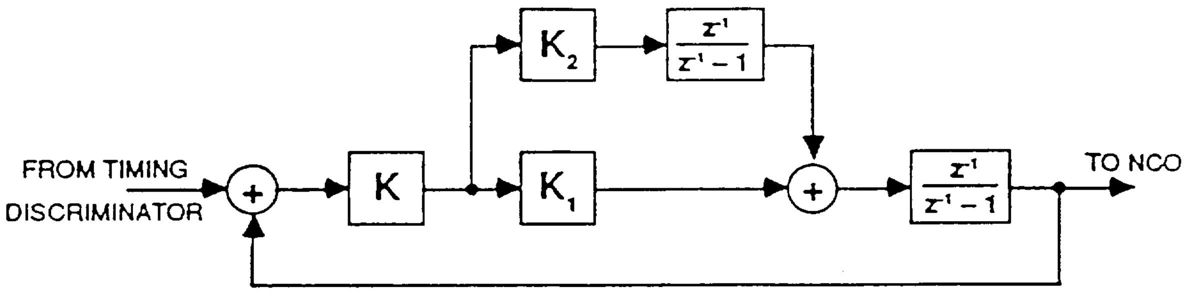


Figure 5.7: Block diagram of the bit timing feedback loop system.

A is defined to be the magnitude of the digitized signal into the bit synchronizer, and this strictly assumes a noiseless input.

T_d is the transition density of data. If the phase changed at every symbol transition, then T_d would be 1.0. In normal operation for a BPSK signal, $T_d = 0.5$; for a QPSK signal, $T_d = 0.75$.

P_a is the number of discriminator accumulations.

b is the scaling factor used in the pre-accumulator block, so that

b = 1 when A_i =1

b = 2 when A_i =2 or 4

b = 4 when A_i =8 or 16

Figure 5.8 is a graph illustrating a loop which was designed for a bandwidth (normalized to the data rate) of 1 % with K=1. As K increases, the loop gain increases eventually resulting in loop instability. As K is reduced, the loop gain approaches a value of about 0.2 %, but stability is maintained. The best region to operate the feedback loop, from the standpoint of transition response characteristics, is in the center, where bandwidth and gain are almost linearly related.

The loop bandwidth B_L can be determined from the following Table 5.1 and controlled by varying K_1 and K_2, given K as computed in equation (5.10).

In our design, with the following parameters:

A = 32 (with 6-bit input) $T_d = 0.75$ (QPSK, OQPSK and MSK)

$A_i = 1$ $T_d = 0.5$ (BPSK)

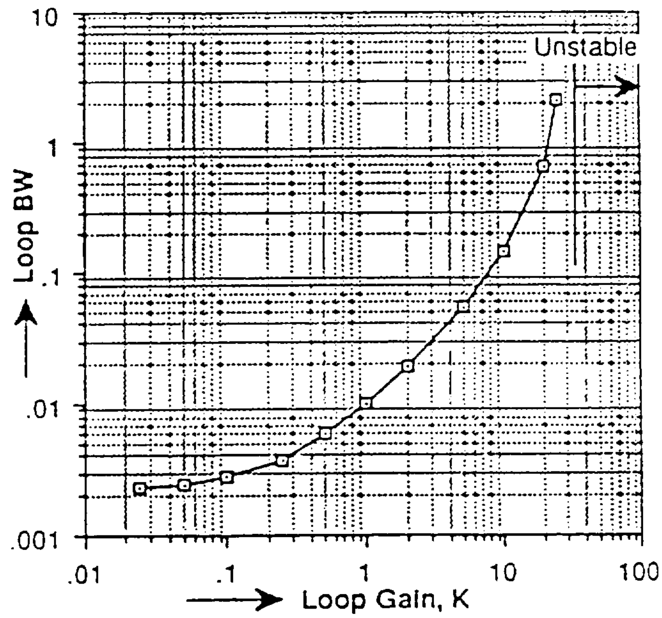


Figure 5.8: The performance related with loop gain and bandwidth.

K×K1	K×K2	B_L
1×2^{-3}	1×2^{-7}	0.05
1×2^{-4}	1×2^{-9}	0.025
1×2^{-5}	1×2^{-10}	0.012
1×2^{-6}	1×2^{-12}	0.006
1×2^{-7}	1×2^{-14}	0.003
1×2^{-8}	1×2^{-16}	0.0015
1×2^{-9}	1×2^{-18}	0.0008

Table 5.1: The loop bandwidth with K1 and K2.

Bit 3	Bit 2	Bit 1	Bit 0	K_1
0	0	0	0	2^0
0	0	0	1	2^1
0	0	1	0	2^2
0	0	1	1	2^3
0	1	0	0	2^4
0	1	0	1	2^5
0	1	1	0	2^6
0	1	1	1	2^7*
1	0	0	0	2^8
1	0	0	1	2^9
1	0	1	0	2^{10}
1	0	1	1	2^{11}
1	1	0	0	2^{12}
1	1	0	1	2^{13}
1	1	1	0	2^{14}
1	1	1	1	2^{15}

Bit 7	Bit 6	Bit 5	Bit 4	K_2
0	0	0	0	2^{-4}
0	0	0	1	2^{-3}
0	0	1	0	2^{-2}
0	0	1	1	2^{-1}
0	1	0	0	2^0
0	1	0	1	2^1
0	1	1	0	2^2
0	1	1	1	2^3
1	0	0	0	2^4
1	0	0	1	2^5
1	0	1	0	2^6
1	0	1	1	2^7*
1	1	0	0	2^8
1	1	0	1	2^9
1	1	1	0	2^{10}
1	1	1	1	2^{11}

* Default values

Table 5.2: (a) The control factor K1, (b) The control factor K2.

$S_s = 4$ $\qquad\qquad$ b = 1 when $A_i = 1$

$P_a = 1$

The N_r has different values with different symbol rates R_s. If $R_s = 9600$ bps for QPSK, we have $K = 2^{-15}$.

By setting $K_1 = 2^{10}$ $(K \times K_1 = 2^{-5})$ and $K_2 = 2^5$ $(K \times K_2 = 2^{-10})$, the loop bandwidth B_L is determined from Table 5.1 to be approximately 1 % of the symbol rate. Then from Table 5.2, we can determine what value will be programmed into the loop gain control register. For above setting

$$K_1 = 2^{10} \Rightarrow 1010 \quad or \quad A\,(Hex)$$
$$K_2 = 2^5 \Rightarrow 1001 \quad or \quad 9\,(Hex)$$

This value $9A_H$ can be programmed into address 11_H.

(3) Start up the chip STEL-2110

It is necessary to connect RCLK (reference clock, pin N9) to ESCK (external sample clock, pin M12) and to set CKSEL (select clock, pin M1) low while loading

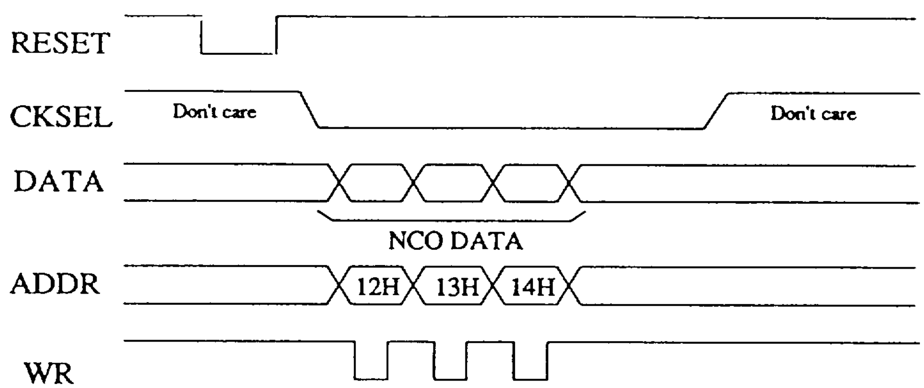


Figure 5.9: Start up the chip STEL-2110.

the NCO data, after power up or after any time the chip is reset with the reset pin RST, and then CKSEL can be set high as shown in Figure 5.9.

(4) Other considerations

(a) 255BCK signal (pin N11) is connected directly to CACK (pin D6) and AACK (pin R1). This signal controls the number of integrations used in the carrier tracking discriminator function and the number of integrations used in the lock detection function.

(b) BCK (pin P8) signal is synchronized to the input signal timing and is nominally a square wave. It is used as the output data clock. The falling edge of BCK corresponds to the beginning of a new symbol data period, so that the signal is low during the first half of the symbol.

(c) DC (pin H12) is connected to a control line which comes from the microcontroller. A high level on this pin causes the Q channel inputs to be delayed by one half of a symbol period. This control is intended to enable the chip to be used for OQPSK and MSK signals.

(d) APSEL (pin C14):

APSEL = 1 (high) for noncoherent AFC to perform carrier tracking

APSEL = 0 (low) for coherent PLL to perform carrier tracking

(e) QCDD (pin E3) and BOVR (pin D14) are connected to a control line which comes from the microcontroller.

QCDD and BOVR = 1 (high) for BPSK

QCDD and BOVR = 0 (low) for QPSK, OQPSK and MSK

This setup can improve carrier tracking and the timing discrimination performances.

5.5 Carrier Recovery Circuit for QPSK, OQPSK and BPSK

We use digital Phase Lock Loop circuit to realize carrier recovery for QPSK, OQPSK, and BPSK. The block diagram of this carrier recovery circuit is shown in Figure 3.2.

In the system shown the microcontroller integrats the $\mathbf{CARD}_{15-0}$ signal to produce the data to program the NCO (in our design, we use DDS to act as that) which generates the local oscillator signals with variable frequencies. The **BCK** signal can be used to interrupt the microcontroller at regular intervals to do this at our low bit rate.

5.5.1 Digital Phase Lock Loop

We use second order digial Phase Lock Loop (PLL) to realize carrier recovery. Second oder PLL can make both its output signal frequency and phase be locked with the input signal. There are two loop gains in this second order PLL. The blobk diagram for this PLL is shown in Figure 5.10.

The block $\frac{z^{-1}}{1-z^{-1}}$ is just the delay associated with one period in the sampling rate. Integrating the input signal $\mathbf{CARD}_{15-0}$ is also just the addition in digital signal processing program. The values of the two gains in our digital PLL is much difficult to calculate. We have successfully set these two values by experience for properly system operation. The values of these two gains can determine the locking range, locking speed, and locking stabability. A lot of strength has been put on the correct selection of thses two parameters.

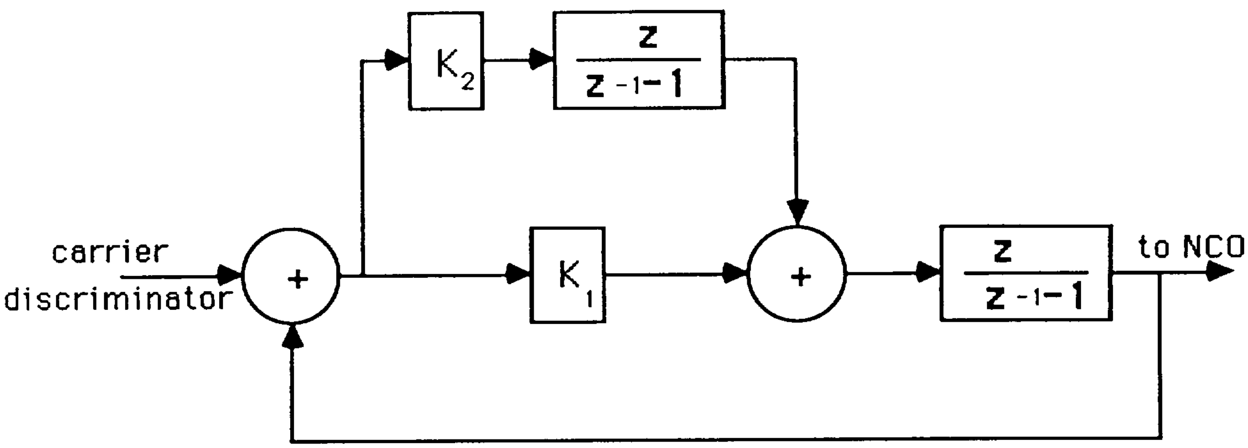


Figure 5.10: Digital Phase Lock Loop.

DDS Q2334 is selected as the NCO. It can produce two perfect orthogonal signals which are required. The frequencies and phases of these two signals are controlled by the microcontroller according to the calculated results of our software loop filter (second order). After D/A converter, two analog carrier reference signals are with same frequencies and 90° difference phase. Note the phases of these two reference signals are 45° different from that of our demodulator input signal, one is 45° ahead and one is 45° later.

Until now, the locking range of this digital Phase Lock Loop (PLL) is not very wide. There is about 50 Hz range for 9.6 kbps bit rate and about 16 Hz range for 1.2 kbps bit rate. The locking range is mainly related to the bit rate. Because in our project, the bit rates are low so that the locking range are narrow. We have tried many ways to improve locking ranges and the above results are what we can get until now. An automatic frequency control (AFC) is needed in RF stage to reduce frequency variation in the IF signal.

Because of this narrow locking range, the carrier recovery reference frequency must be adjusted for different receivers. This is because of the frequency differences among the crystals, although the difference is relatively small compared with their own running frequency.

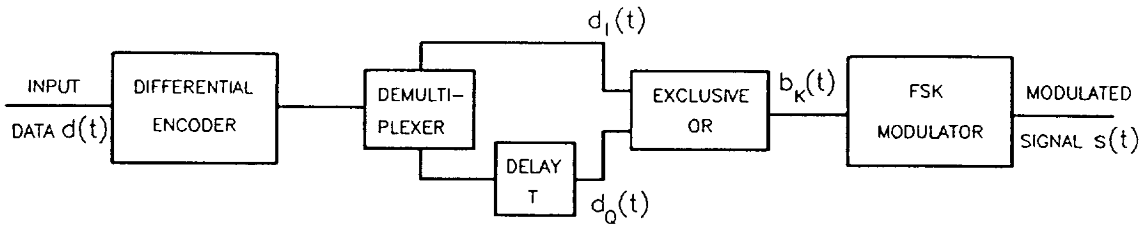


Figure 5.11: MSK modulation scheme.

The realization of this carrier recovery circuit is one of the most difficult tasks in this project.

5.6 Msk Demodulator

The circuits for coherent MSK demodulation is given in Chapter 2.1.4. This demodulation scheme keeps the good bit error rate performance of MSK, but the circuits employed are hard to implement, expecially in the case of low bit rate data transmission. For example, in coherent MSK demodulation, the two frequency tones of Sunde's FSK (square of MSK signal) are so close that it is very difficult to realize the needed bandpass filter (BPF). An easier noncoherent MSK demodulation scheme is developed. Compared with the coherent MSK demodulation, the noncoherent MSK demodulation scheme preserves the good MSK attributes of continuous phase and spectral efficiency, dramatically simplifies the demodulator at some expense of BER performance degradation.

MSK can be viewed either as an OQPSK signal with sinusoidal pulse weighting or as a continuous phase FSK (CPFSK) signal with a frequency separation equal to one-half the bit-rate [32]. An MSK modulation scheme in the form of CPFSK signal with modulation index equal to 0.5 is shown in Figure 5.11.

That MSK can be viewed as CPFSK also gives us the idea to demodulate MSK signal noncoherently. After that, a post-detection circuit must be added to restore original baseband signal. The related block diagram is show in Figure 5.12.

The operation principle of this part is based on the following observation:

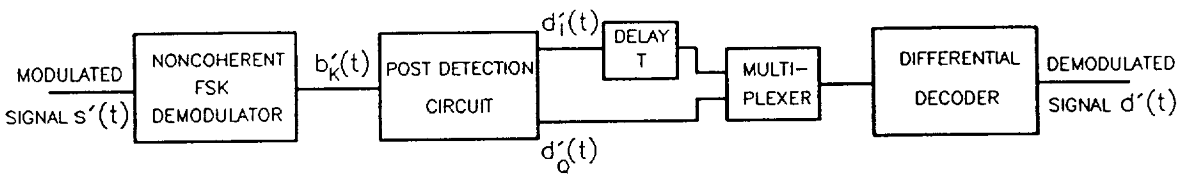


Figure 5.12: MSK demodulation scheme.

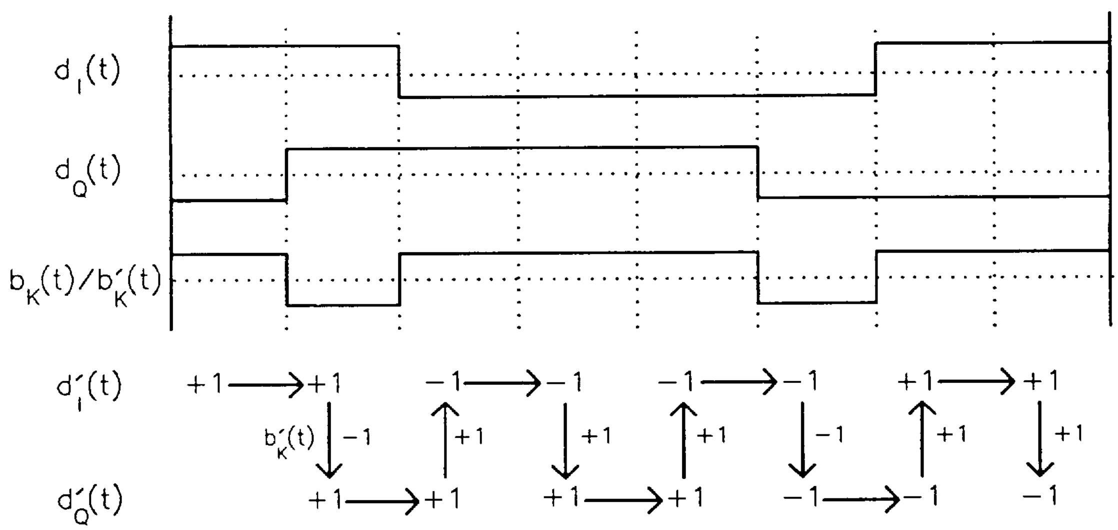


Figure 5.13: An example showing MSK baseband waveforms relations.

suppose, in first bit period, $d'_I(t)$ is arbitrarily decided (we can do this because differential coder is used). In the second bit period, $d'_I(t)$ still keeps its value in the first bir period. According to the relationship between $b'_K(t)$ and $d'_I(t)$ and $d'_Q(t)$ (e.g., $b'_K(t)$=-$d'_I(t)d'_Q(t)$), the value of $d'_Q(t)$ in the second bit period can be decided from the known values of $b'_K(t)$ and $d'_I(t)$. Similarly, in the third bit period, $d'_Q(t)$ keeps its value in the second bit period and then $d'_I(t)$ can be decided. The decision rule for the following bit periods is the same. An example for this is shown in Figure 5.13.

A simple circuit for this post-decision part according to above decision rule is shown in Figure 5.14.

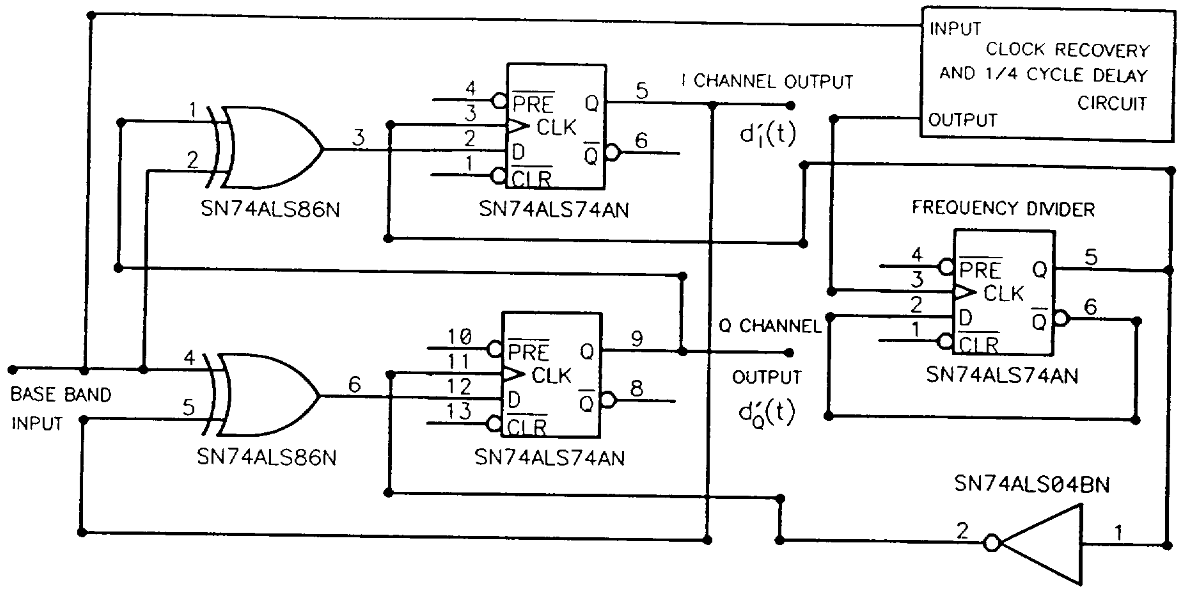


Figure 5.14: MSK post-detection circuit.

Compared with coherent and differentially coherent MSK demodulation, the bit error rate performance of the noncoherent MSK demodulation developed here is about 4 dB worse than the former and about 2 dB worse than the later [31]. But, in low bit rate communications systems, where the transmitted bit energy is relatively much larger than that in high bit rate systems, this loss in power efficiency is tolerable. In the meantime, the other attributes of MSK, such as constant envelope with continuous phase, good spectral efficiency are perserved in this easily realized demodulator because they do not rely on which demodulation scheme is used.

In our project, the key point to use this scheme is to findout good noncoherent FSK demodulator. There are several such kind of VLSI demodulator available, but only NE564 is barely suitable for our application. Because the bit rate is too low in our application, the frequency deviation is very small. Thus, the output signal is very weak. To solve this problem, we first decrease the IF frequency of the received modulated signal from 4.8 MHz to 30 KHz by multiplexing this IF frequency by a local oscillator which has a 4770 KHz frequency. This local oscillator can be replaced by the DDS in the receiver because at this time it is free from the carrier recovery circuit for QPSK, etc..

Using this method, we have successfully get the final demodulated and decoded correct result for bit rate 2.4 kbps and 4.8 kbps. But a problem exists. The MSK domodulator is not robust enough that each time when the system operates, we get to trim the related capacitor to get a correct result. For the other three bit rates, we still can not succeed. Also, because this is an analog circuit instead of a digital one, when we change bit rate, we have to change many related capacitors for proper system operation. It would be much better if we can use a digital circuit to demodulate MSK signal, but VLSI chips for such kind of digital circuit are not available in the market yet.

Chapter 6

CONTROL CIRCUITS DESIGN

In this chapter we will focus our attention on: (1) serial port communication; (2) the variable bit rate control; (3) configurations and control. But first we will introduce Intel 80C32 microcontroller briefly[18].

6.1 Intel 80C32 Microcontroller

In our design an Intel 80C32 microcontroller controls the whole system using a monitor program contained in EPROM on the board. Figure 6.1 shows the architectural block diagram of 80C32 microcontroller.

The major features of 80C32 are:

1) 8-bit CPU

2) 32 I/O lines (4 I/O ports)

3) Three 16-bit timer/counter

4) Six-source interrupts with two priority levels

5) Full duplex serial port

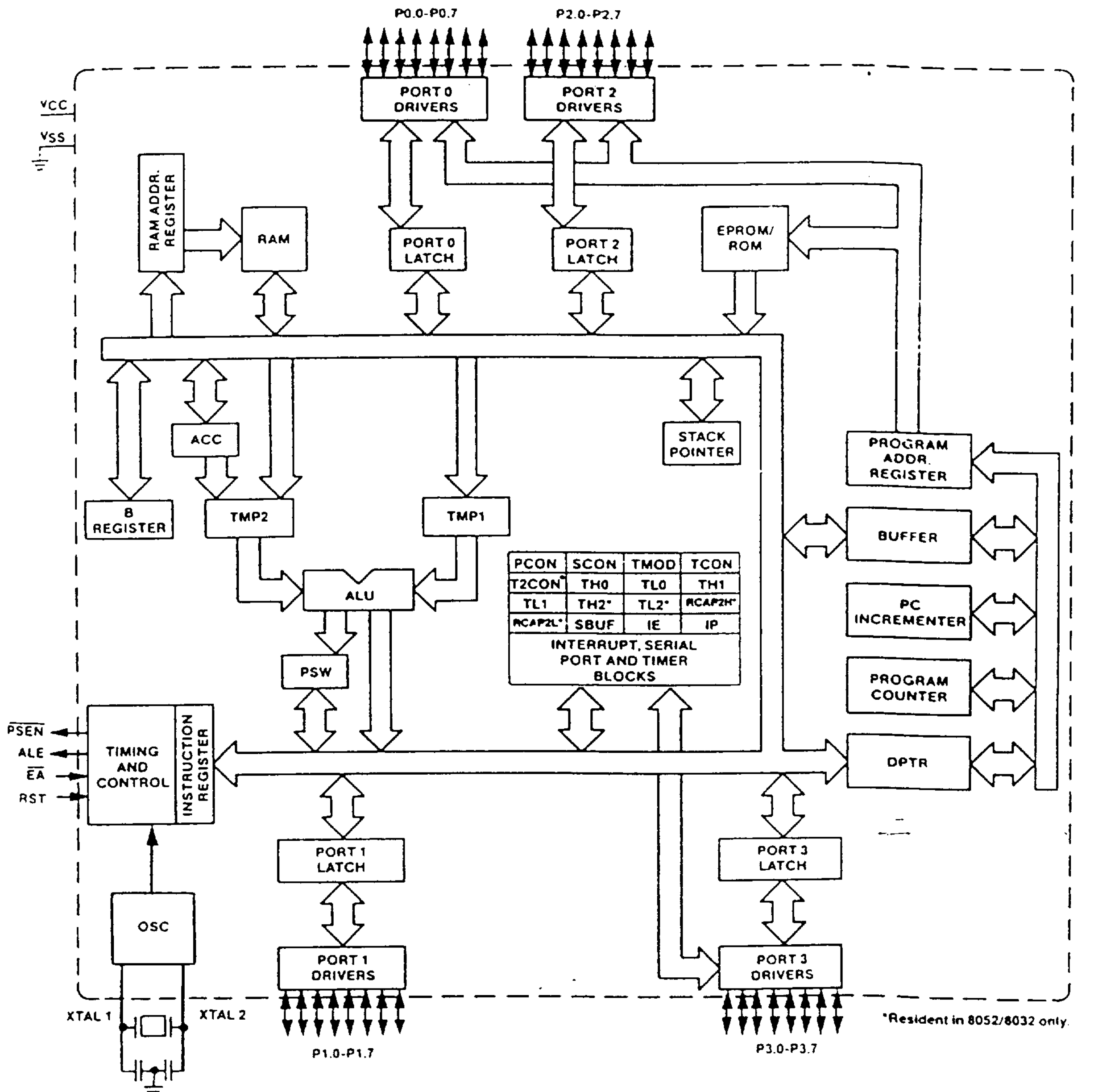


Figure 6.1: Intel 80C32 architectural block diagram.

6.2 Variable Bit Rate Control

The Timer 2 is employed to realize the variable bit rate control in modulating process. It is configured in auto-reloaded mode. The data rate R which equals to the interrupt control rate is determined by the Timer 2 overflow rate.

$$R = (bits\ of\ a\ symbol) \times \frac{Oscillator\ Frequency}{12 \times [65536 - (RCAP2)]} \tag{6.1}$$

Where RCAP2 is the register pair (RCAP2H, RCAP2L) for the Timer 2 "auto-reloaded mode".

Because there is a delay of T period in the Q channel of OQPSK and MSK, the phase transition occurs every T seconds, not 2T seconds like QPSK does. This can be seen from Figure 2.4. The OQPSK and MSK have the same formula to calculate the value of RCAP2.

For BPSK, OQPSK and MSK

$$R = \frac{Oscillator\ Frequency}{12 \times [65536 - (RCAP2)]} \tag{6.2}$$

For QPSK

$$R = 2 \times \frac{OscillatorFrequency}{12 \times [65536 - (RCAP2)]} \tag{6.3}$$

For example, if R=19200 bps (input data rate=9600 bps):

For BPSK, OQPSK and MSK

$$19200 = \frac{11.0592 \times 10^6}{12 \times [65536 - (RCAP2)]}$$

get

$$RCAP2 = 65536 - \frac{11.0592 \times 10^6}{12 \times 19200} = FFD0 \quad (Hex)$$

For QPSK

$$19200 = 2 \times \frac{11.0592 \times 10^6}{12 \times [65536 - (RCAP2)]}$$

get

$$RCAP2 = 65536 - \frac{2 \times 11.0592 \times 10^6}{12 \times 19200} = FFA0 \quad (Hex)$$

With different the data rate R and modulation schemes there are different values of RCAP2 which must be preset by software.

In our design we have developed an interrupt service routine T2_INT. It is used to precisely time writes to the DDS Q2334 chip to run the various modulations. Wherever the service routine T2_INT is entered, two control signals are generated. One is used to control PM1 CLK, PM2 CLK and MUX CLK pins of the Q2334. Another is used as the delay clock for the delay circuit of OQPSK and MSK. Following is the design steps:

(1) Enable Timer 2 interrupt with the highest priority.

(2) Loade the appropriate value into register pair RCAP2 in Timer 2 with different data and modulation schemes.

(3) As long as Timer 2 is overflowed, when running the system, it set up the interrupt flag automatically.

(4) When CPU responses this requirement, it goes to the interrupt service routine T2_INT. The routine first clears the Timer 2 interrupt flag, and then generates two control signals.

6.3 Configuration and Control

Once a particular modulation/demodulation scheme is chosen the control software goes into the subroutines to configure that scheme.

6.3.1 BPSK

(1) Set up the registers of the Q2334

(a) Calculate what will be put into Q2334 registers for carrier frequency, and load the results into phase increment registers PIRA1 (00H-03H) of the DDS1.

(b) Set up SMC1 register (0BH) of the DDS1 to enable the external phase modulation function.

(c) Set up AMC1 register (0AH) of the DDS1 to enable the on-chip Noise Reduction Circuit (NRC) function, determine the D/A converter output bits function and output data format.

(d) Clear the registers of the second part of the DDS and the accumulators.

(2) Set up the registers of the Q0256

Encoder Function:

(a) Set up the encoder control register 1 (06H) to select encoder in serial data output mode and code rate 1/2.

(b) Set up the encoder control register 2 (07H) to enable differential encoder and the V.35 data scrambler.

Decoder Function:

(a) Set up the decoder control register 1 (02H) to select the decoder in serial data input mode and code rate 1/2.

(b) Set up the decoder control register 2 (03H) to accept offset-binary format data mode in soft-decision input at R0, the differential decoder and V.35 data descrambler.

(c) Set up the decoder control register 3 (04H) to choose the Viterbi decoder algorithm use a minimum chainback path depth of 96 states or 48 states.

(d) Set up T register (08H) and N register (09H).

(e) Set up BER period input registers (0AH-0CH).

(f) Set up the reserved write registers 15H and 16H to 0.

(3) Set up the registers of the STEL-2110

(a) Set up the bit rate control registers (12H-14H) to establish the nominal A/D sample rate according to different the data rate.

(b) Set up the loop gain control register (11H) to select the proper K_1 and K_2 which relate loop gain to loop bandwidth.

(c) Set up timing control register (01H) to select the number of samples per symbol, control factor P_a, the clock frequency pre-scale factor and pre-accumulation control factor.

(d) Set up mode control register (17H) to store various control parameters as shown: input data format control, loop filter input control, coherent DPSK enable, freeze data control, loop offset control and software rest.

(4) Set up the control lines

6.3.2 QPSK and OQPSK

(1) Set up the registers of the Q2334

(a) Calculate what will be write into Q2334 registers for carrier frequency, and load the results into phase increment register PIRA1 (00H-03H) and PIRA2 (10H-13H).

(b) Set up SMC1 (08H) and SMC2 (18H) registers to enable the external phase modulation function.

(c) Set up AMC1 (0AH) and AMC2 (1AH) registers to enable the on-chip NRC function, determine the A/D converter output bits and output data format.

(d) Clear the accumulators.

(2) Set up the registers of the Q0256

Encoder Function:

(a) Set up the encoder control register 1 (06H) to select encoder in parallel data output mode and code rate 1/2.

(b) Set up the encoder control register 2 (07H), same as BPSK.

Decoder Function:

(a) Set up the decoder control register 1 (02H) to select the decoder in parallel data input , and code rate 1/2. Meanwhile, set bit 1 = 0 to make synchronization circuit adjust for phase ambiguities of QPSK demodulator; set bit 1 = 1 to make synchronization circuit adjust for phase ambiguities of OQPSK demodulator.

(b) Set up the decoder control register 2 (03H) to accept offset-binary format data mode in soft-decision input at R0 and R1, enable phase ambiguity automatic synchronization for QPSK and OQPSK, the differential decoder and the V.35 data descrambler.

(c) Set up the decoder control register 3 (04H), same as BPSK.

(d) Set up T register (08H) and N register (09H), same as BPSK.

(e) Set up BER period input registers (0AH-0CH), same as BPSK.

(f) Set up the reserved registers 15H and 16H to 0.

(3) Set up the registers of the STEL-2110

(a) Set up the bit rate control registers (12H-14H) to establish the nominal

A/D sample rate for QPSK and OQPSK according to different the data rate.

(b) Set up the loop gain control register (11H) to select the proper K_1 and K_2 for QPSK and OQPSK.

(c) Set up timing control register (01H), same as BPSK.

(d) Set up mode control register (17H), same as BPSK.

6.3.3 MSK

(1) Set up the registers of the Q2334

(a) Calculate what will be write into the Q2334 registers for frequencies $f_{c+} = f_c + \frac{1}{4T}$ and $f_{c-} = f_c - \frac{1}{4T}$, and load the results into phase increment registers PIRA1 (00H-03H) and PIRB1 (04H-07H) of the DDS1.

(b) Set up SMC1 (08H) register of the DDS1 to enable the external multiplex control function.

(c) Set up AMC1 (0AH) register of the DDS1 to enable the On-chip NRC-function, determine the D/A converter output bits and output data format.

(d) Clear the registers of the second part of the DDS and the accumulators.

(2) Set up the registers of the Q0256

Same as the OQPSK registers set up.

(3) Set up the registers of the STEL-2110

Same as the OQPSK registers set up.

(4) Set up control lines

6.4 Control Software

This software is designed to control the programmable small terminal for satellite communication system. It is entirely written in 8051 assembler language using Intel MCS-51 conventions. In general, high-level and system startup routines are located near the start of this program. Interrupt service routines and canned messages to the user are located at the end. Timer 1 is used to generate the serial baud rate. Timer 2 is used to time writes to the Q2334 DDS for controlling data rate.

(1) Software copy

When setting up the configurations from the console (terminal) the screen will display the contents of the registers of the three chips (Q2334, Q0256 and STEL-2110), and a menu of options for the user. Since these registers are "write only", what is actually displayed is a software copy of what was last written to the chips. The method is that we open the data block for each chips at the internal data memory of the 80C32. The space is equal to the number of registers of each chips. Whenever we write the control code into the registers of the chips, we also write the same code into the internal RAM (called for software copy). After that, when we want to know what was last written to the chips, we can simply read these control codes from their software copies.

(2) Internal and external data memory maps

Figure 6.2(a) shows the internal data memory map of the 80C32, and Figure 6.2(b) shows the external data memory map. This kind of partition of data memory can accomplish two functions:

1) use bit 5~bit 7 (3 bits) to select the chip which we want to access.

2) use bit 0~bit 4 (5 bits) to get the correct register for the selected chip.

(3) Flowcharts

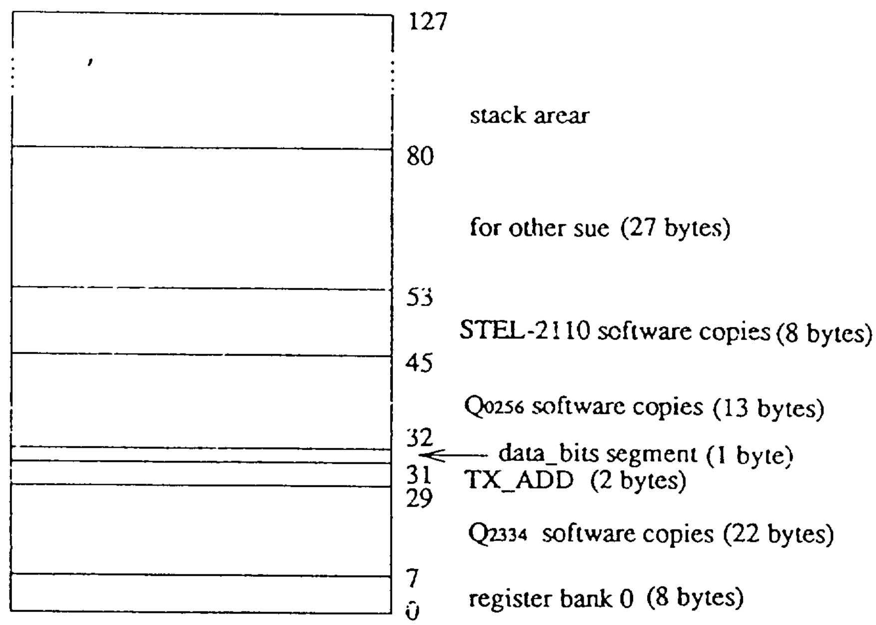
Figure 6.3(a) is the main flowchart,

Figure 6.3(b) for QPSK,

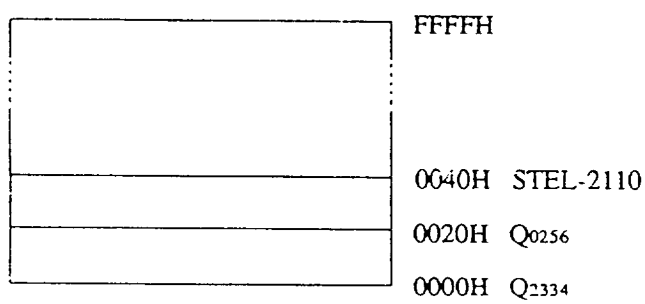
Figure 6.3(c) for OQPSK,

Figure 6.3(d) for MSK,

Figure 6.3(e) for BPSK.

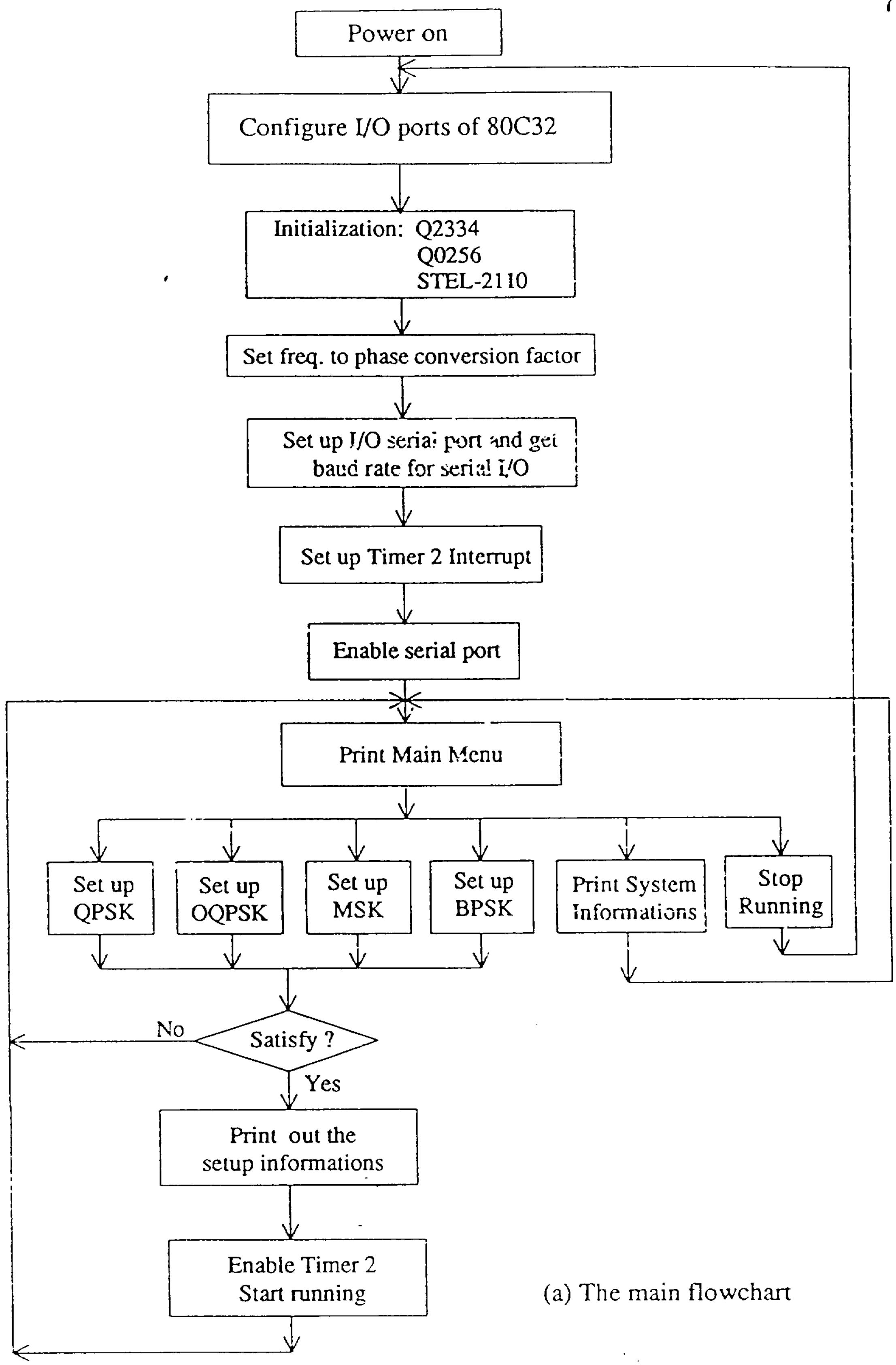


(a) Internal Memory Map

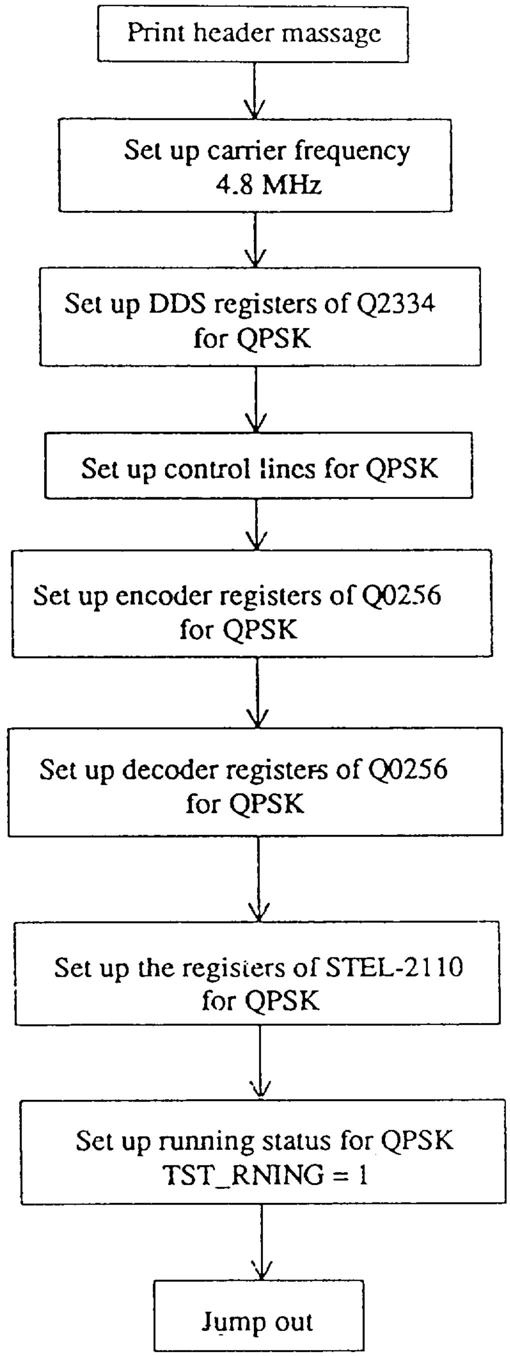


(b) External Memory Map

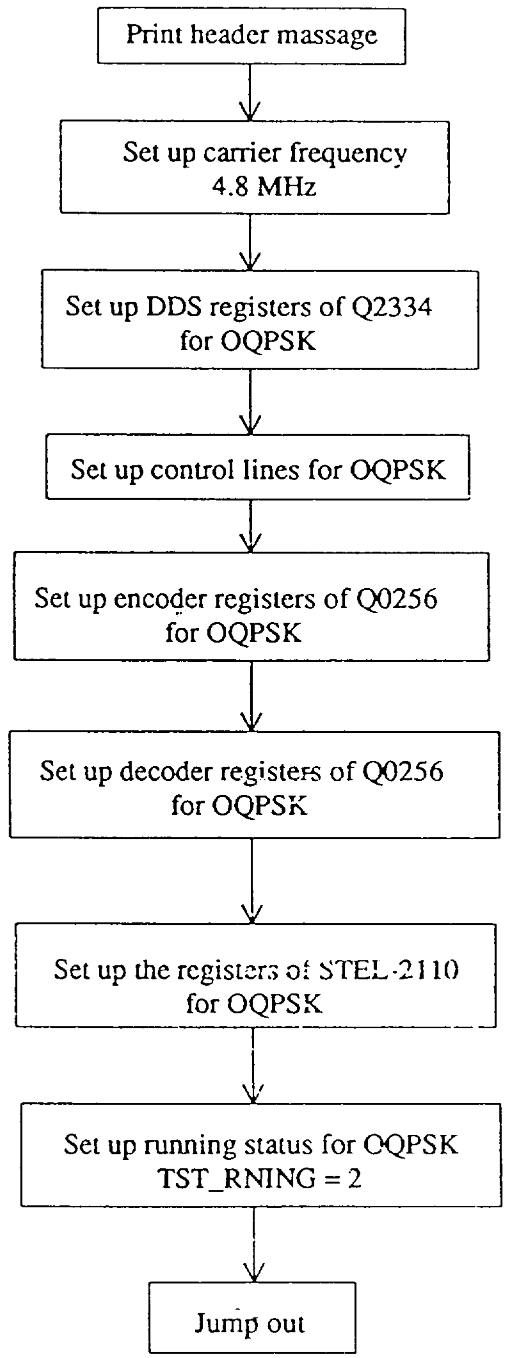
Figure 6.2: (a) Internal memory map, (b) External memory map.



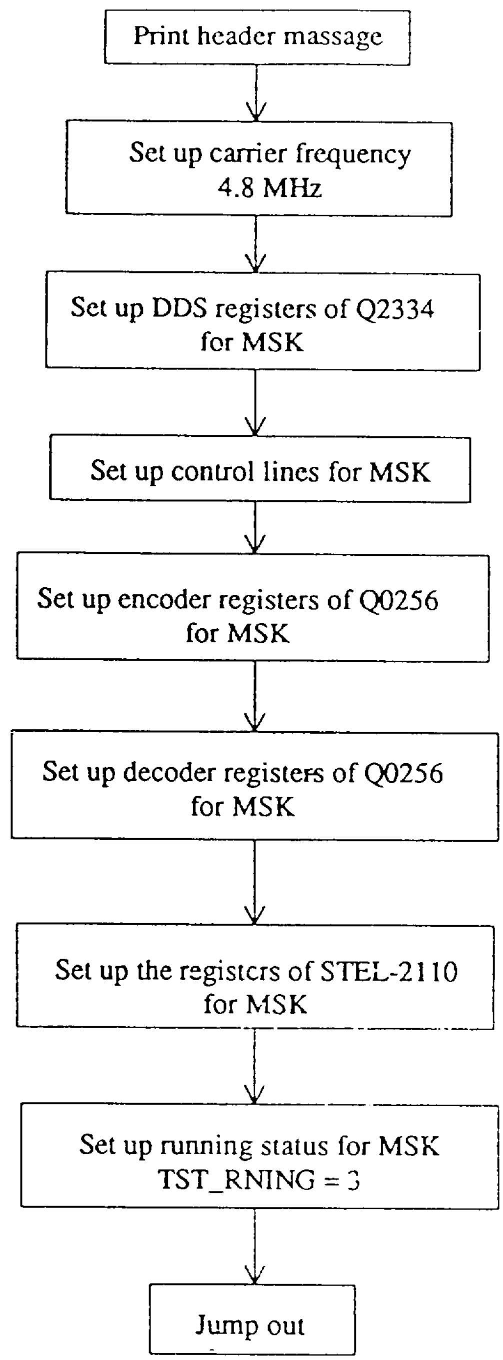
(a) The main flowchart


Print header massage
Set up carrier frequency 4.8 MHz
Set up DDS registers of Q2334 for QPSK
Set up control lines for QPSK
Set up encoder registers of Q0256 for QPSK
Set up decoder registers of Q0256 for QPSK
Set up the registers of STEL-2110 for QPSK
Set up running status for QPSK TST_RNING = 1
Jump out

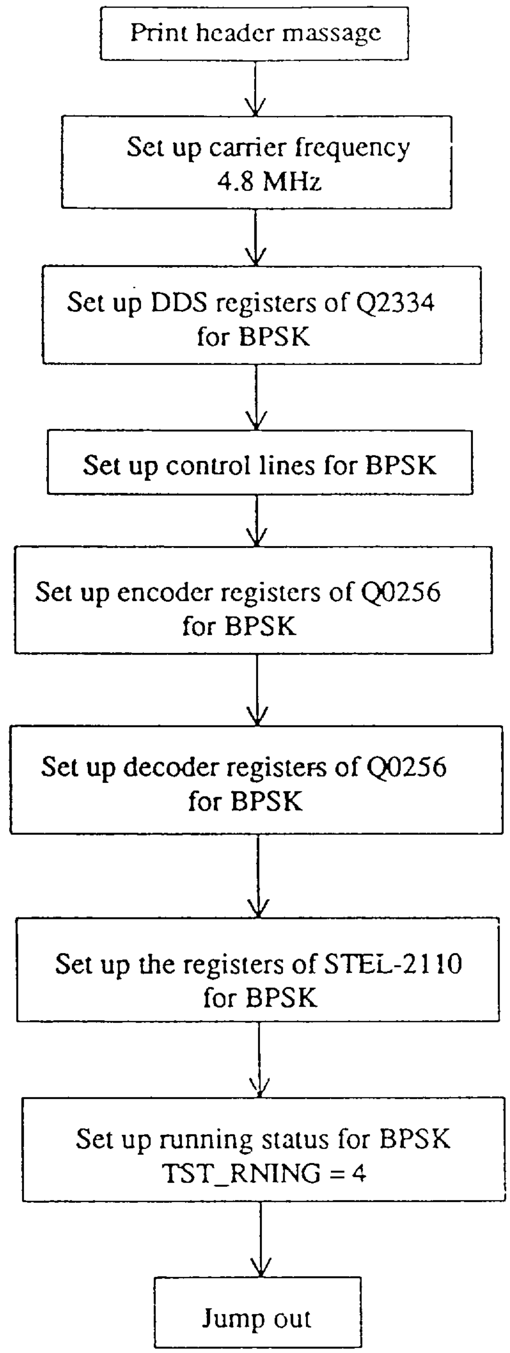
(b) The QPSK



(c) The OQPSK flowchart



(d) The MSK flowchart



(e) The BPSK flowchart

Figure 6.3: Flowchart

Chapter 7

TEST RESULTS

7.1 Power Spectral Desities Measurement

In this part, we will show test results of the transmitter section. These test results include all measured power spectral densities of modulated signals, MSK, QPSK, OQPSK and BPSK under different data rates.

The input data to the transmitter section is from the signal generator (PM5193, PHILIPS) or the data error analyzer which can generate digital data signals with different preselected rates. The output signal from the transmitter section is directly fed to the sprctrum analyzer (7L12, TEKTRONIX).

Test conditions of the spectrum analyzer.

Center Frequency:	4.8 MHz
Resolution Bandwidth:	300Hz
Video Bandwidth:	3Hz
Y Scale: Log.	10dB/div

X Scale and Frequency Span are related to modulation schemes and data rates.

(1) MSK

Figure 7.1 shows measured power spectral densities of modulated signals

MSK under different data rates. From Figure 7.1 we can clearly see that there are different X Scales (frequency/division) with different data rates. This is because with different data rates the main-lobe bandwidth (null-to-null bandwidth) of MSK is different, and equal to

$$BW_{MSK} = \frac{1.5}{T} = 1.5 \times R$$

where T is the bit duration after encoding (R=1/T). Also, we notice that spectral densities of MSK have the wider main-lobe and the lower side-lobes.

(a) Data rate = 19200 bps

The main-lobe bandwidth for this test should be

$$BW_{MSK} = 1.5 \times 19.2 \times 2 = 57.6 \quad KHz$$

Set the spectrum analyzer:

Frequency Span:	200KHz
X Scale:	20KHz/div

The power spectral density for this test is shown in Figure 7.1(a).

(b) Data rate = 9600 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 28.8\ KHz$

Set the spectrum analyzer:

Frequency Span:	100KHz
X Scale:	10KHz/div

The power spectral density for this test is shown in Figure 7.1(b).

(c) Data rate = 4800 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 14.4\ KHz$

Set the spectrum analyzer:

Frequency Span:	50KHz
X Scale:	5KHz/div

The power spectral density for this test is shown in Figure 7.1(c).

(d) Data rate = 2400 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 7.2\ KHz$

Set the spectrum analyzer:

Frequency Span: 20KHz

X Scale: 2KHz/div

The power spectral density for this test is shown in Figure 7.1(d).

(e) Data rate = 1200 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 3.6\ KHz$

Set the spectrum analyzer:

Frequency Span: 10KHz

X Scale: 1KHz/div

The power spectral density for this test is shown in Figure 7.1(e).

(2) QPSK and OQPSK

Figure 7.2 shows measured power spectral densities of modulated signals QPSK and OQPSK under different data rates. From Figure 7.2, we can see that there are different X Scales with different data rates. The main-lobe bandwidth of QPSK and OQPSK is equal to

$$BW_{QPSK} = \frac{1.0}{T} = 1.0 \times R$$

Also, we notice that spectral densities of QPSK and OQPSK have the narrower main-lobe and higher side-lobes than that of MSK.

(a) Data rate = 19200 bps

The main-lobe bandwidth for this test should be

$$BW_{MSK} = 1.0 \times 19.2 \times 2 = 38.4 \quad KHz$$

Set the spectrum analyzer:

Frequency Span: 200KHz

X Scale: 20KHz/div

The power spectral density for this test is shown in Figure 7.2(a).

(b) Data rate = 9600 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 19.2\ KHz$

Set the spectrum analyzer:

Frequency Span: 100KHz

X Scale: 10KHz/div

The power spectral density for this test is shown in Figure 7.2(b).

(c) Data rate = 4800 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 9.6\ KHz$

Set the spectrum analyzer:

Frequency Span:	50KHz
X Scale:	5KHz/div

The power spectral density for this test is shown in Figure 7.2(c).

(d) Data rate = 2400 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 4.8\ KHz$

Set the spectrum analyzer:

Frequency Span:	20KHz
X Scale:	2KHz/div

The power spectral density for this test is shown in Figure 7.2(d).

(e) Data rate = 1200 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 2.4\ KHz$

Set the spectrum analyzer:

Frequency Span:	10KHz
X Scale:	1KHz/div

The power spectral density for this test is shown in Figure 7.2(e).

(3) BPSK

Figure 7.3 shows measured power spectral densities of modulated signals BPSK under different data rates. From Figure 7.3, we can see that there are different X Scales with different data rates. The main-lobe bandwidth of BPSK is equal to

$$BW_{QPSK} = \frac{2.0}{T} = 2.0 \times R$$

Also, we notice that spectral densities of BPSK have the widest main-lobe and the highest side-lobes in the test.

(a) Data rate = 19200 bps

The main-lobe bandwidth for this test should be

$$BW_{MSK} = 2.0 \times 19.2 \times 2 = 76.8 \quad KHz$$

Set the spectrum analyzer:

Frequency Span:	200KHz
X Scale:	20KHz/div

The power spectral density for this test is shown in Figure 73(a).

(b) Data rate = 9600 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 38.4\ KHz$

Set the spectrum analyzer:

Frequency Span:	100KHz
X Scale:	10KHz/div

The power spectral density for this test is shown in Figure 7.3(b).

(c) Data rate = 4800 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 19.2\ KHz$

Set the spectrum analyzer:

Frequency Span:	50KHz
X Scale:	5KHz/div

The power spectral density for this test is shown in Figure 7.3(c).

(d) Data rate = 2400 bps

The main-lobe bandwidth for this test should be $BW_{MSK} = 9.6\ KHz$

Set the spectrum analyzer:

Frequency Span:	20KHz
X Scale:	2KHz/div

The power spectral density for this test is shown in Figure 7.3(d).

(e) Data rate = 1200 bps

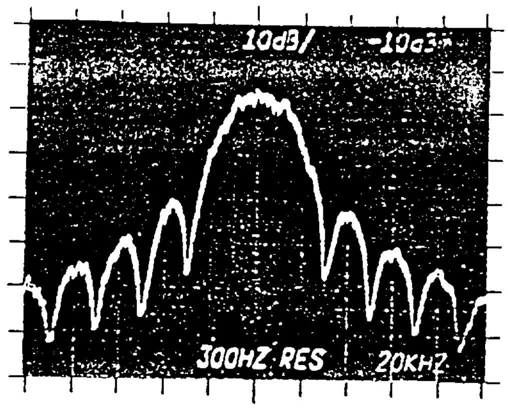
The main-lobe bandwidth for this test should be $BW_{MSK} = 4.8\ KHz$

Set the spectrum analyzer:

Frequency Span:	10KHz
X Scale:	1KHz/div

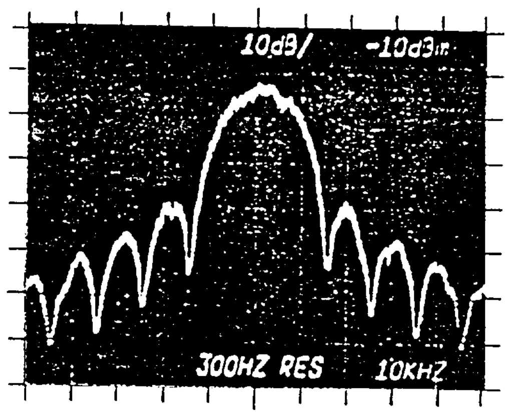
The power spectral density for this test is shown in Figure 7.3(e).

All above test results in this chapter agree with theoretical predictions very well[10]. This means that the transmitter section and control section are working very well, and satisfy reqirements of our design.



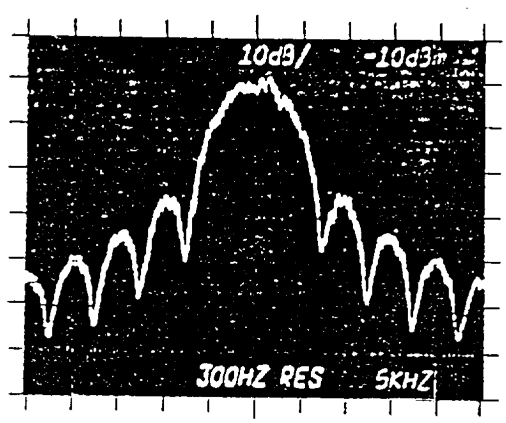
Center Freq. 4.8 MHz Freq. Span 200 KHz

(a) Data rate = 19200 bps



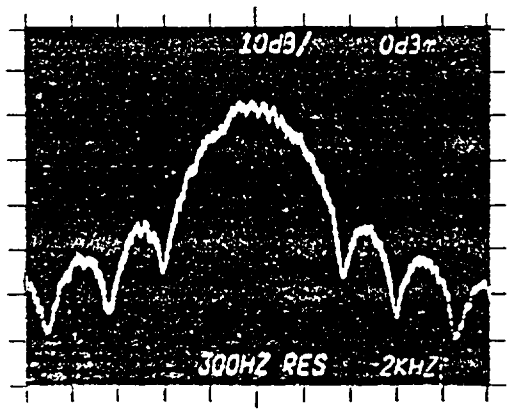
Center Freq. 4.8 MHz Freq. Span 100 KHz

(b) Data rate = 9600 bps



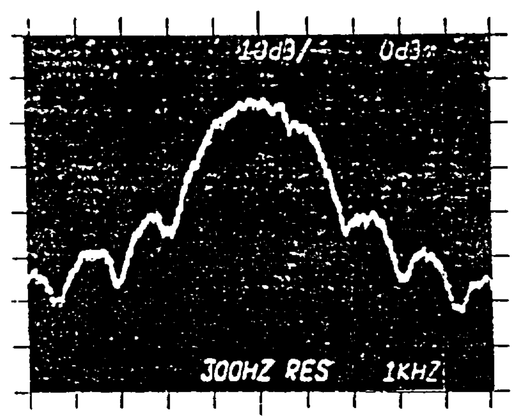
Center Freq. 4.8 MHz Freq. Span 50 KHz

(c) Data rate = 4800 bps



Center Freq. 4.8 MHz Freq. Span 20 KHz

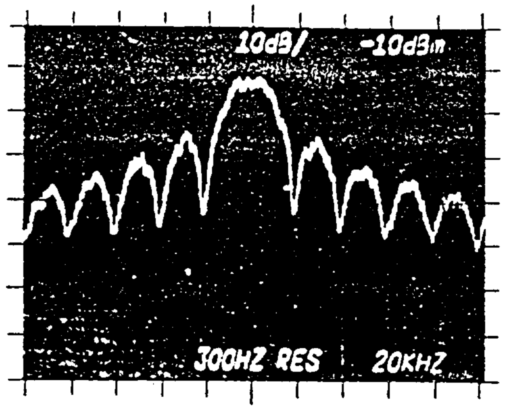
(a) Data rate = 2400 bps



Center Freq. 4.8 MHz Freq. Span 10 KHz

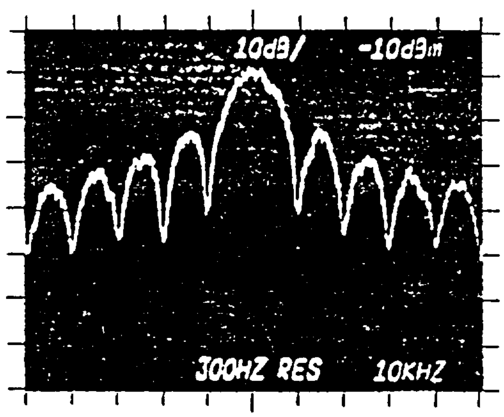
(b) Data rate = 1200 bps

Figure 7.1: Power spectral densities of MSK.



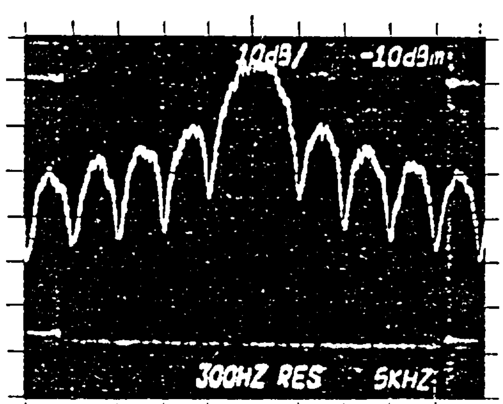
Center Freq. 4.8 MHz Freq. Span 200 KHz

(a) Data rate = 19200 bps



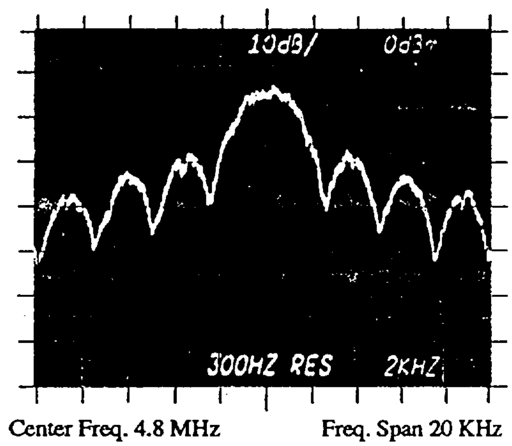
Center Freq. 4.8 MHz Freq. Span 100 KHz

(b) Data rate = 9600 bps

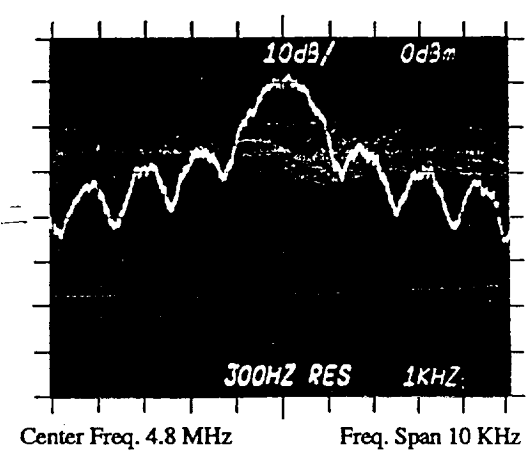


Center Freq. 4.8 MHz Freq. Span 50 KHz

(c) Data rate = 4800 bps

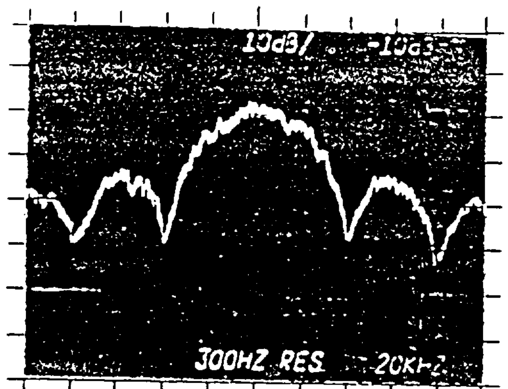


(a) Data rate = 2400 bps



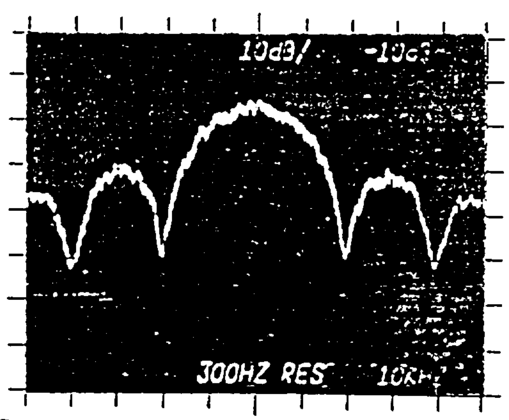
(b) Data rate = 1200 bps

Figure 7.2: Power spectral densities of QPSK and OQPSK.



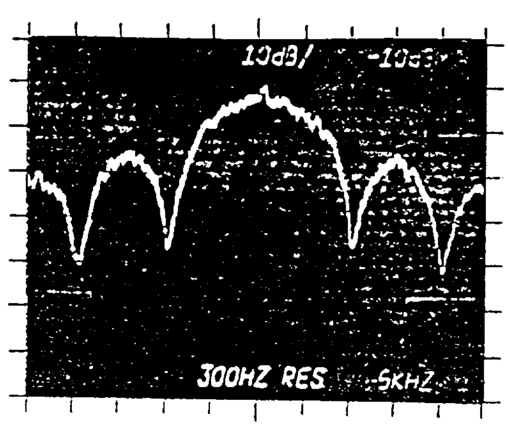
Center Freq. 4.8 MHz   Freq. Span 200 KHz

(a) Data rate = 19200 bps



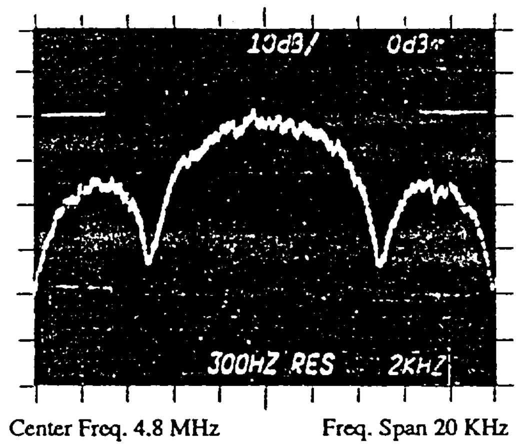
Center Freq. 4.8 MHz   Freq. Span 100 KHz

(b) Data rate = 9600 bps

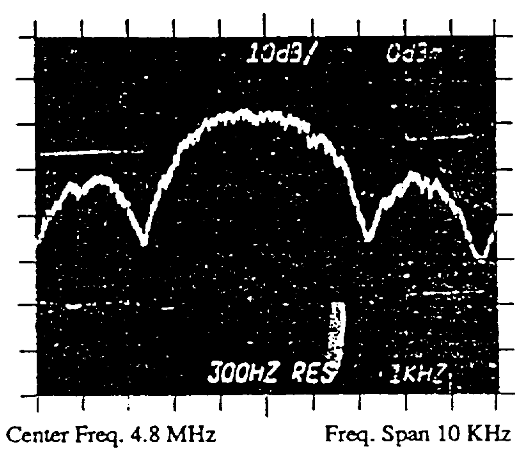


Center Freq. 4.8 MHz   Freq. Span 50 KHz

(c) Data rate = 4800 bps



(a) Data rate = 2400 bps



(b) Data rate = 1200 bps

Figure 7.3: Power spectral densities of BPSK.

For tests of the receiver section, we need several instruments to assist our BER measurement. First a attenuable noise generator with output power around -20 dBm and output frequency above 4.8 MHz is needed. Also signal combiner and power meter are required for our test. Until now, we can not find the needed noise generator. We will do this part of test in the future once we have it.

Bibliography

[1] C. Mahle et al,"Satellite Scenarios and Technology for the 1990's," *IEEE Journal on Selected Areas in Communications,* vol. SAC-5, no. 4, May 1987, pp. 556-569.

[2] J.N. Pelton and W.W. Wu,"The Challenge of 21st Century Satellite Communications: INTESAT Enters the Second Millennium," *IEEE Journal on Selected Areas in Communications,* vol.SAC-5, no. 4, May 1987, pp. 571-591.

[3] R.T. Gedney,"Assessment of the Advanced Communication Technology Satellite (ACTS) Onboard Switching and Processing System," *Proceedings of Onboard processing and Switching Conference,* NASA Lewis Research Center, Cleveland, Ohio, Nov. 1991.

[4] B. Sklar, *Digital Communications: Fundamentals and Applications,* Prentice-Hall, New Jersey, 1988.

[5] L.W. Couch II, *Digital and Analog Communication Systems,* Macmillan, London, 1990.

[6] J.G. Proakis, *Dgital Communications,* McGraw-Hill, New York, 1990.

[7] V.R. Bhargava, D. Haccoum, R. Matyas and P.P. Nuspl, *Digital Communications by Satellite,* John Wiley & Sons, New York, 1981.

[8] W. Tomasi, *Advanced Electrnic Communications Systems,* Prentice-Hall, New Jersey, 1987.

[9] S. Pasupathy,"Minimum Shift Keying: A Spectrally Efficient Modulation," *IEEE Communication Magazine,* vol. 17, July 1979, pp. 14-22.

[10] F. Amoroso,"The Bandwidth of Digital Data Signals," *IEEE Communication Magazine,* vol. 18, no. 6, Nov. 1980, pp. 13-24.

[11] R.E. Ziemer and C.R. Ryan, "Minimum-shift keyed Modem Implementations for High Data Rates," *IEEE Communications Magazine,* vol. 21, no. 7, Oct. 1983, pp. 28-37.

[12] S.A. Gronemeyer and A.L. McBride,"MSK and Offset QPSK Modulation," *IEEE Tran. on Communications,* vol.COM-24, no. 8, August 1976, pp. 809-820.

[13] S. Lin and D.J. Costello, *Error Control Coding: Fundamentals and Applications,* Prentice-Hall, New Jersey, 1983.

[14] W.W. Wu, D. Haccoun et at,"Coding for Satellite Communication," *IEEE Journal on Selected Areas in Communications,* vol. SAC-5, no. 4, May 1987, pp. 724-748.

[15] *Q2334 Dual Direct Digital Synthesizer Technical Data Sheet,* Qualcomm, Inc., 1992.

[16] *User's Guide for Q2334 DDS Evaluation Kit,* Qualcomm, Inc., 1989.

[17] *Q0256 k=7 Multi-code Rate Viterbi Decoder Technical Data Sheet,* Qualcomm, Inc.,1990.

[18] *Microcontroller Handbook,* Intel Corporation. 1991.

[19] *Data Converter Reference Manual: Volume I and II,* Ananlog Devices, 1992.

[20] A.B. Williams and F.J. Taylor, *Electronic Filter Design Handbook: LC, Active, and Digital Filters,* McGraw-Hill, New York, 1988.

[21] *The Spread Spectrum Handbook,* Stanford Telecom, Inc., 1990.

[22] *Bit Synchronizer/PSK Demodulator STEL-2110A Technical Data Sheet,* Stanford Telecom, Inc., 1992.

[23] R.E. Best, *Phase-Locked Loops: Theory, Design, and Applications,* McGraw-Hill, New York, 1984.

[24] F.M. Gardner, *Phaselock Techniques*, John Wiley &Sons, New York, 1979.

[25] *Signetics Linear Products Handbook,* Signetics Corp., 1988.

[26] *SN74LS624 Voltage-Controlled Oscillator Technical Data Sheet,* Texas Instruments,1988.

[27] *Linear and Interface Integrated Circuits Handbook,* Motorola Inc., 1989.

[28] *Fast and LS TTL Data Handbook,* Motorola Inc., 1989.

[29] D. Zeghlache,"MODEM and CODEC Board for Low Bit Date VSAT at Ka Band," Proposal to NASA Lewis Research Center, 1990.

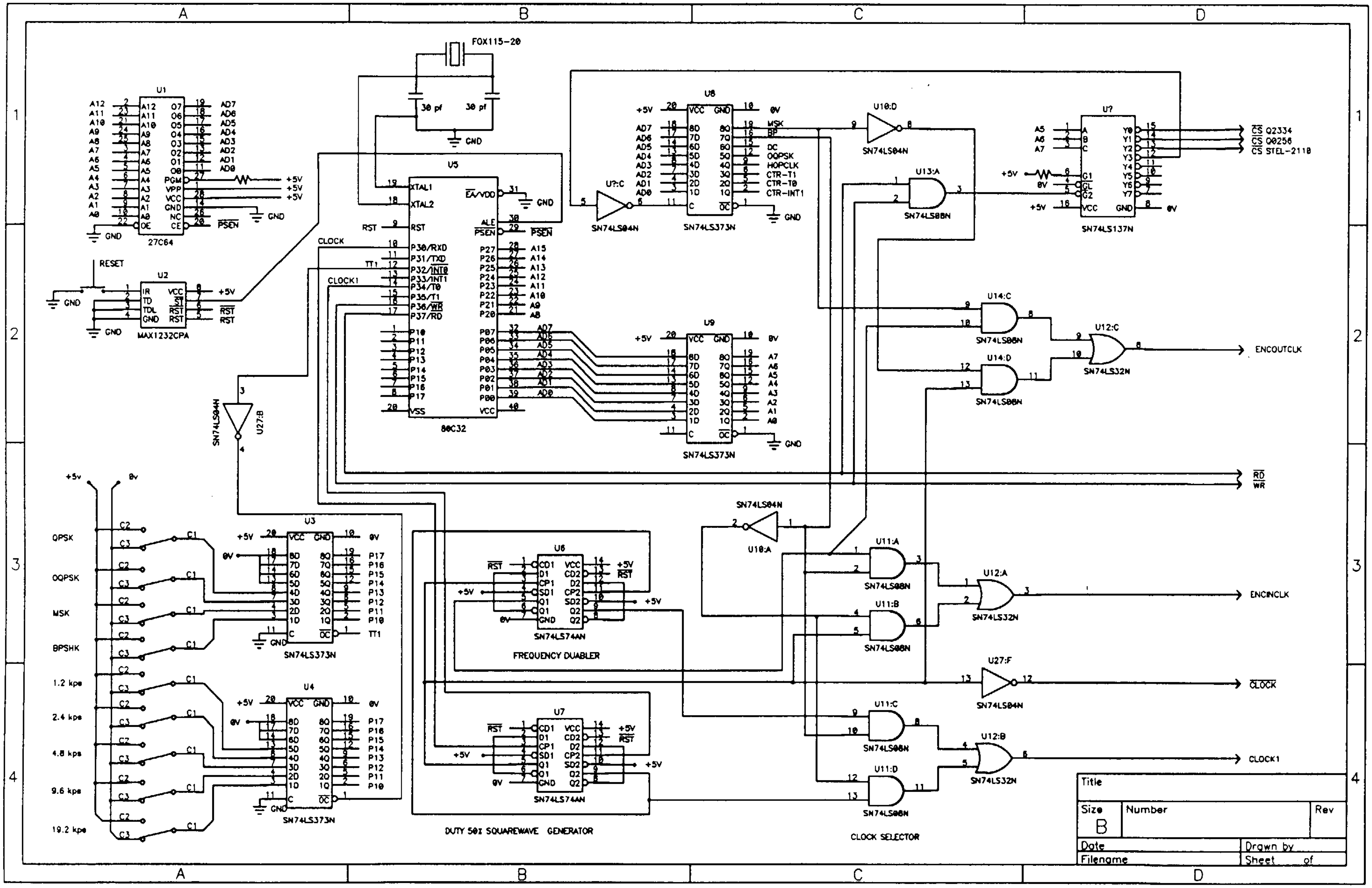
[30] *XR-2208 Operational Multiplier Technical Data Sheet,* Exar Corporation, 1992.

[31] B. Sklar, *Digital Communication, Fundamentals and Applications,* Prentic-Hall, 1988.

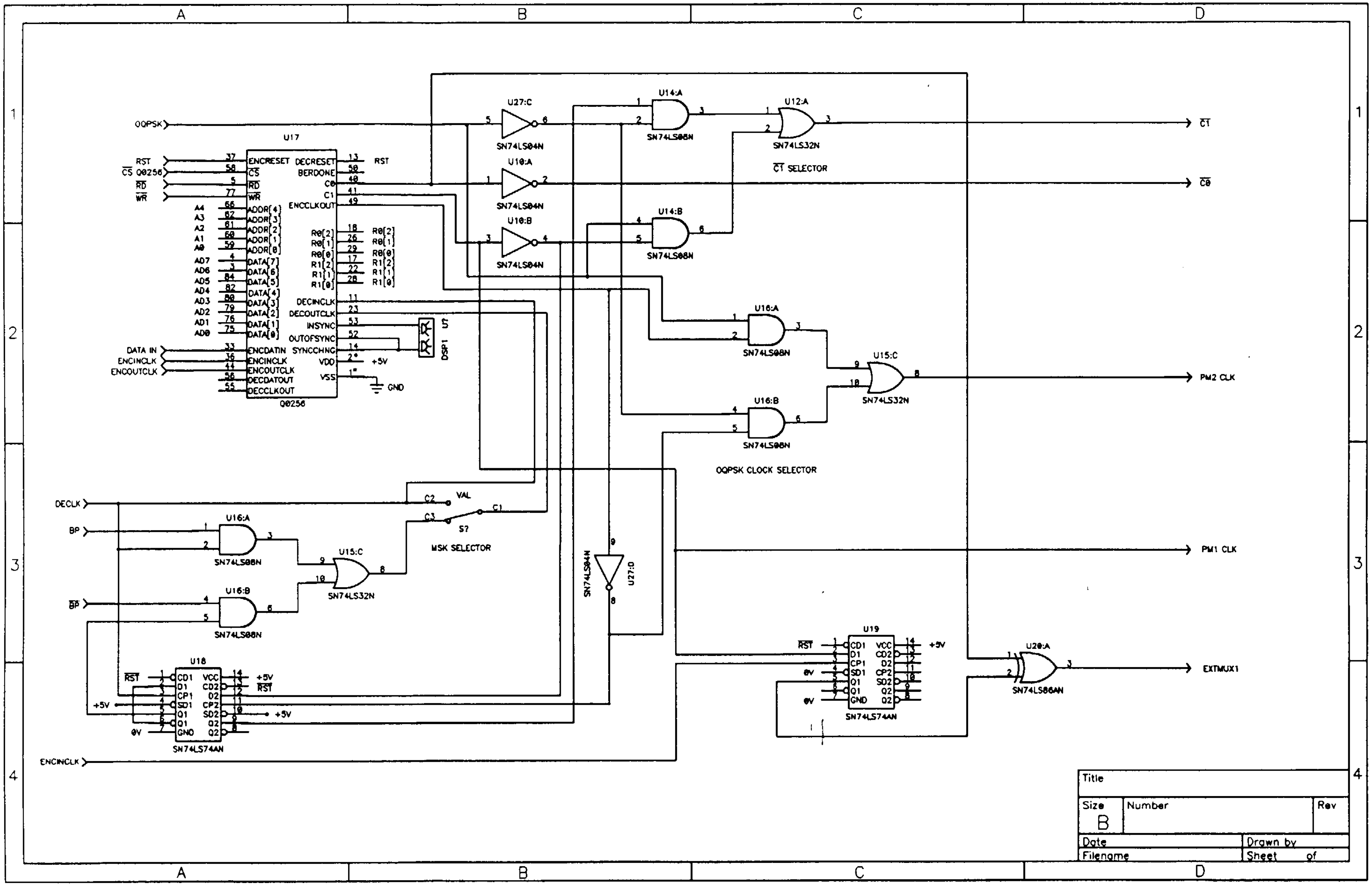
[32] S. Pasupathy, "Minimum Shift keying: A Spectrally Efficient Modulation", *IEEE Communication Mag.*, July 1979, pp. 14-22.

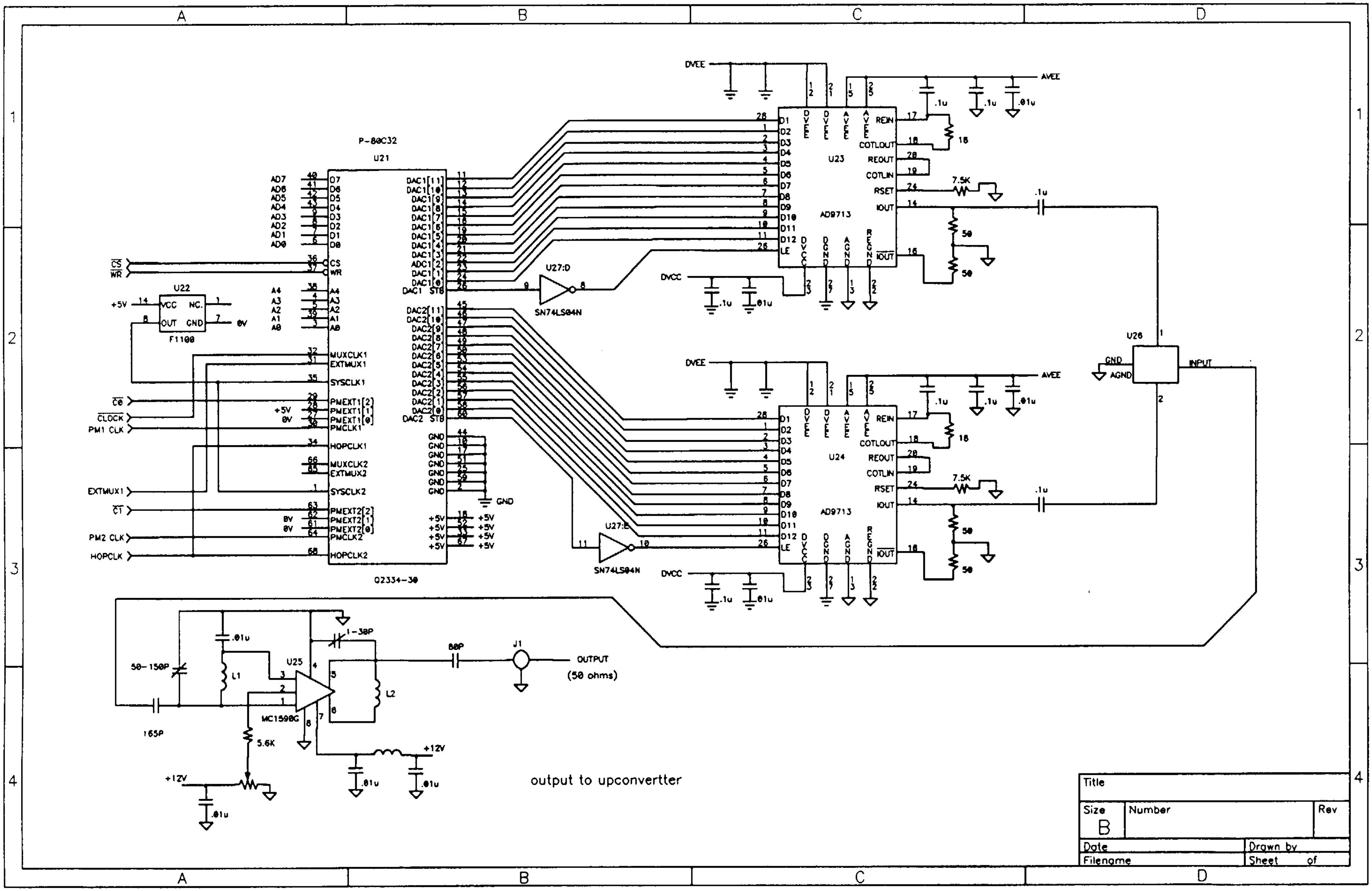
Appendix A

The Design Circuits

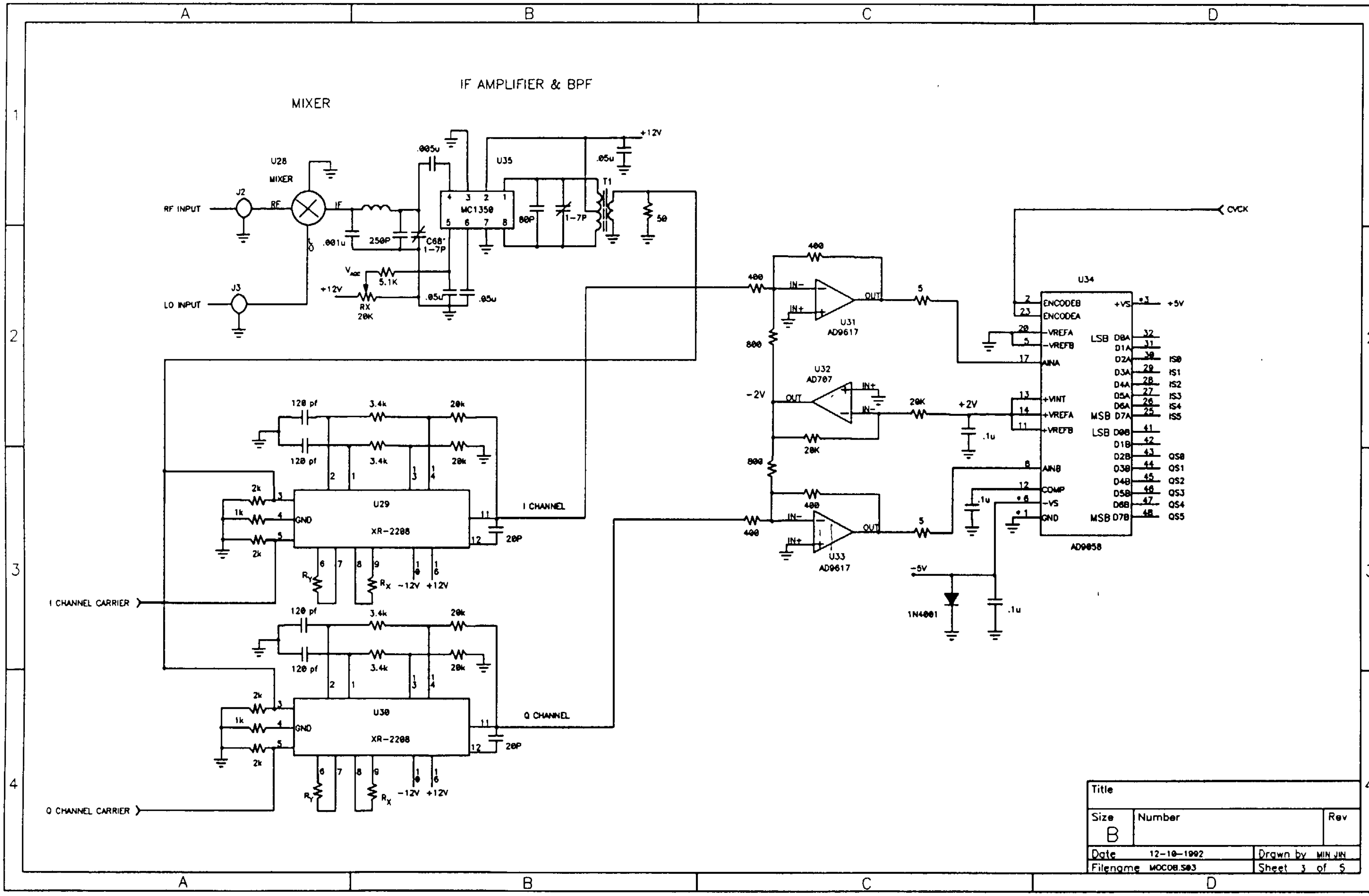


FOX115-20
U5
80C32
U1
27C64
U2
MAX1232CPA
RESET
U8
U9
SN74LS373N
U7
SN74LS137N
CS Q2334
CS Q0256
CS STEL-2110
ENCOUTCLK
RD
WR
ENCINCLK
CLOCK
CLOCK1
U3
U4
U6
SN74LS74AN
FREQUENCY DUABLER
DUTY 50% SQUAREWAVE GENERATOR
CLOCK SELECTOR
QPSK
OQPSK
MSK
BPSHK
1.2 kps
2.4 kps
4.8 kps
9.6 kps
19.2 kps
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet of

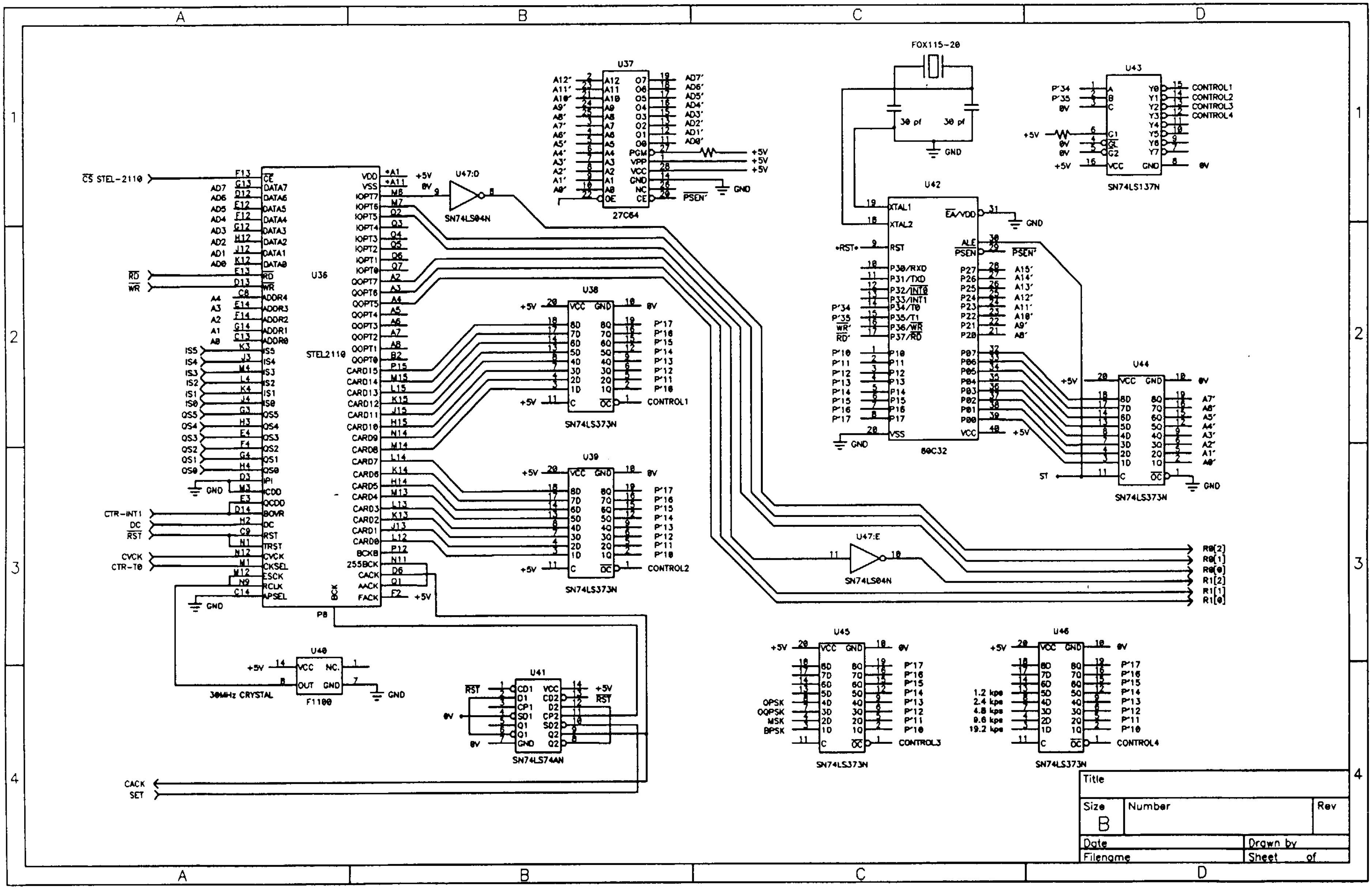

U17
Q0256
ENCRESET
DECRESET
BERDONE
ENCCLKOUT
DECINCLK
DECOUTCLK
INSYNC
OUTOFSYNC
SYNCCHNG
VDD
VSS
ENCDATIN
ENCINCLK
ENCOUTCLK
DECDATOUT
DECCLKOUT
OQPSK
RST
CS Q0256
RD
WR
DATA IN
ENCINCLK
ENCOUTCLK
+5V
GND
DSP1
U27:C
U10:A
U10:B
U27:D
SN74LS04N
U14:A
U14:B
U16:A
U16:B
SN74LS08N
U12:A
U15:C
SN74LS32N
CT SELECTOR
OQPSK CLOCK SELECTOR
CT
C0
PM2 CLK
PM1 CLK
EXTMUX1
U20:A
SN74LS86AN
DECLK
BP
MSK SELECTOR
S7
VAL
U18
U19
SN74LS74AN
0V
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet
of

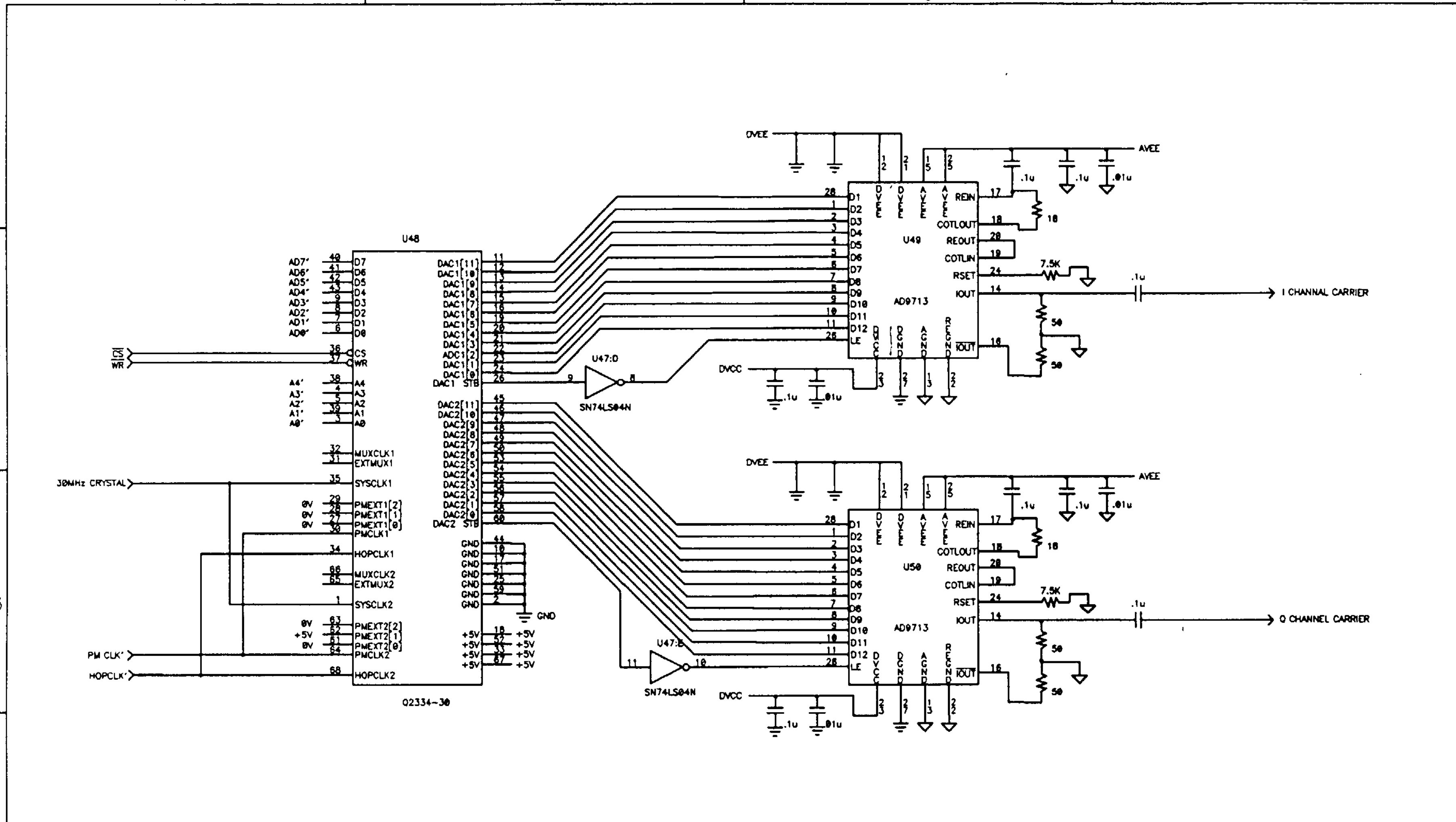


P-80C32
U21
Q2334-30
U22
F1100
U23
U24
AD9713
U26
INPUT
GND
AGND
U27:D
U27:E
SN74LS04N
DVEE
AVEE
DVCC
U25
MC1590G
L1
L2
J1
OUTPUT
(50 ohms)
50-150P
165P
1-30P
80P
5.6K
7.5K
+12V
CS
WR
C0
CLOCK
PM1 CLK
EXTMUX1
C1
PM2 CLK
HOPCLK
output to upconvertter
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet
of

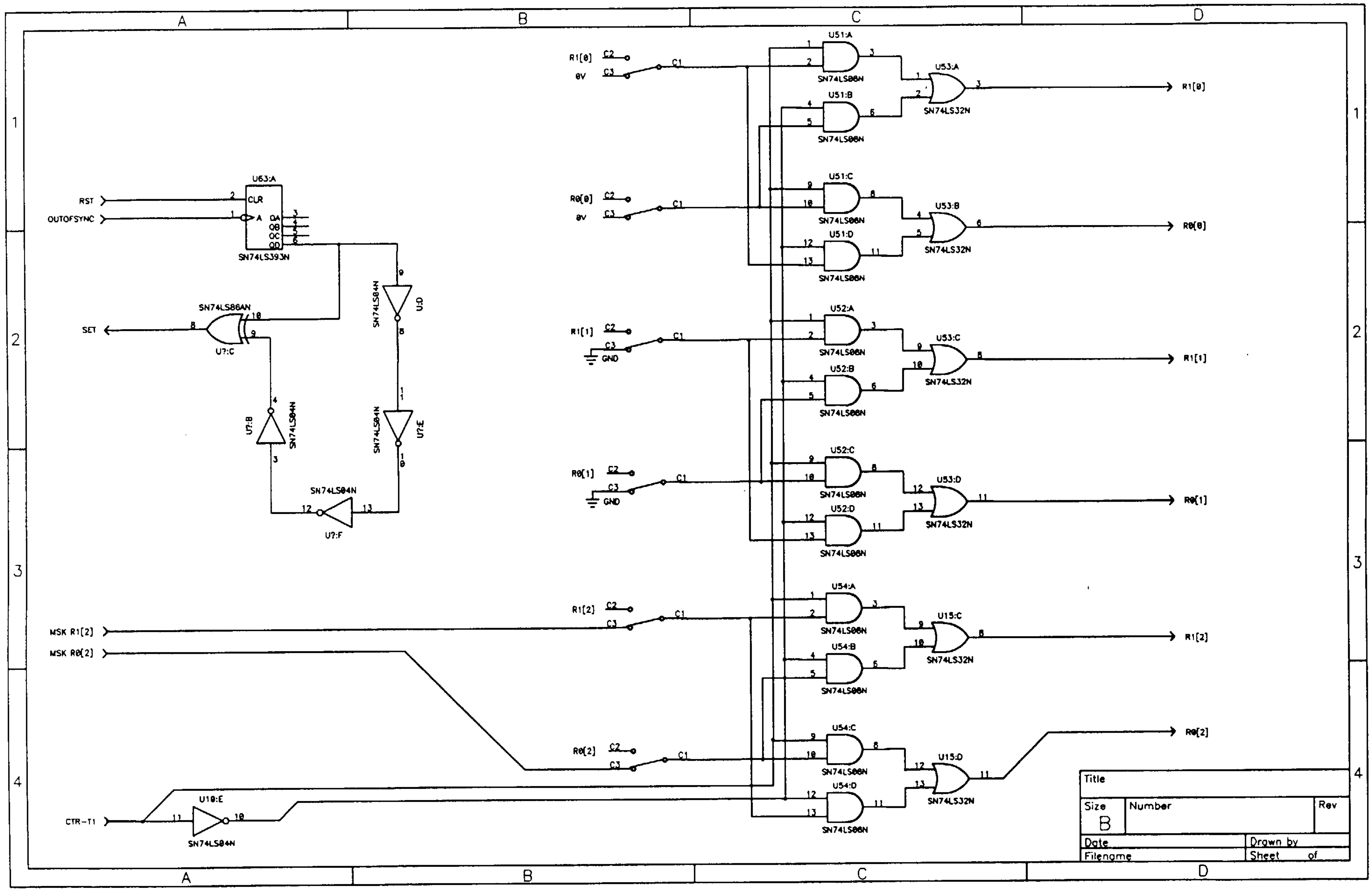


IF AMPLIFIER & BPF
MIXER
U28
MIXER
RF INPUT
J2
RF
IF
LO INPUT
J3
.001u
250P
C68
1-7P
V_{AGC}
5.1K
+12V
RX
20K
.005u
U35
MC1350
.05u
.05u
80P
1-7P
T1
50
+12V
.05u
U29
XR-2208
I CHANNEL
20P
120 pf
3.4k
20k
2k
1k
GND
R_Y
R_X
-12V
+12V
I CHANNEL CARRIER
U30
XR-2208
Q CHANNEL
Q CHANNEL CARRIER
400
IN-
IN+
OUT
U31
AD9617
800
-2V
U32
AD707
20K
+2V
.1u
U33
AD9617
5
U34
ENCODEB
ENCODEA
VREFA
VREFB
AINA
+VINT
+VREFA
+VREFB
AINB
COMP
-VS
GND
+VS
LSB D0A
D1A
D2A
D3A
D4A
D5A
D6A
MSB D7A
LSB D0B
D1B
D2B
D3B
D4B
D5B
D6B
MSB D7B
AD9058
+5V
IS0
IS1
IS2
IS3
IS4
IS5
QS0
QS1
QS2
QS3
QS4
QS5
CVCK
-5V
1N4001
.1u
Title
Size
B
Number
Rev
Date 12-10-1992
Drawn by MIN JIN
Filename MOCOB.S03
Sheet 3 of 5

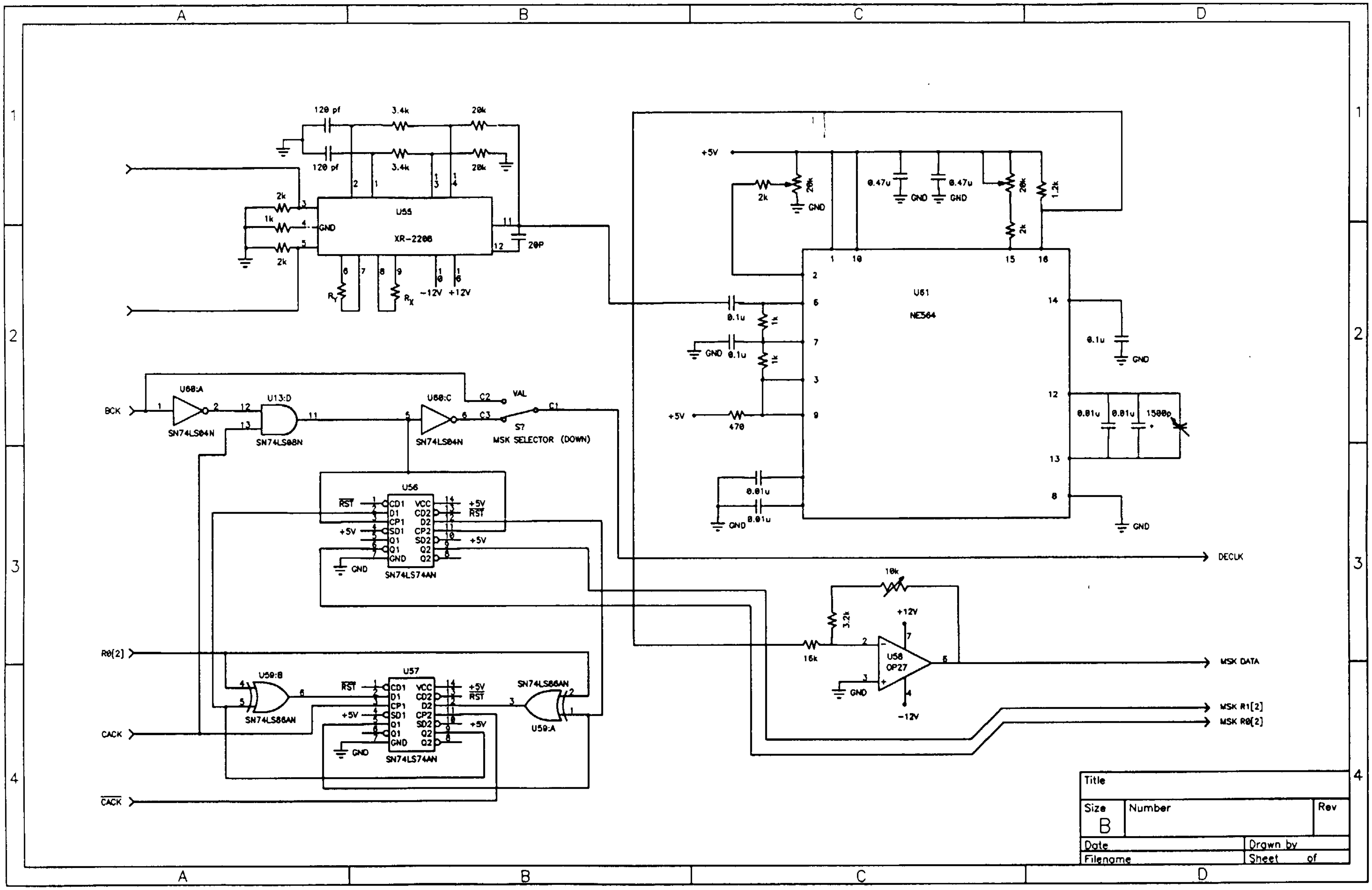

U36
STEL2110
CS STEL-2110
RD
WR
CTR-INT1
DC
RST
CVCK
CTR-T0
CACK
SET
U37
27C64
U38
SN74LS373N
CONTROL1
U39
SN74LS373N
CONTROL2
U40
F1100
30MHz CRYSTAL
U41
SN74LS74AN
U42
80C32
FOX115-20
30 pf
30 pf
U43
SN74LS137N
CONTROL1
CONTROL2
CONTROL3
CONTROL4
U44
SN74LS373N
U45
SN74LS373N
QPSK
OQPSK
MSK
BPSK
CONTROL3
U46
SN74LS373N
1.2 kps
2.4 kps
4.8 kps
9.6 kps
19.2 kps
CONTROL4
U47:D
SN74LS04N
U47:E
SN74LS04N
R0[2]
R0[1]
R0[0]
R1[2]
R1[1]
R1[0]
GND
+5V
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet of



Title		
Size B	Number	Rev
Date	Drawn by	
Filename	Sheet of	



RST
OUTOFSYNC
U63:A
CLR
A
QA
QB
QC
QD
SN74LS393N
SN74LS86AN
U7:C
SET
U:D
U7:E
U7:B
U7:F
SN74LS04N
MSK R1[2]
MSK R0[2]
CTR-T1
U10:E
R1[0]
0V
R0[0]
R1[1]
GND
R0[1]
R1[2]
R0[2]
C1
C2
C3
U51:A
U51:B
U51:C
U51:D
U52:A
U52:B
U52:C
U52:D
U54:A
U54:B
U54:C
U54:D
SN74LS08N
U53:A
U53:B
U53:C
U53:D
U15:C
U15:D
SN74LS32N
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet
of


U55
XR-2206
U61
NE564
U68:A
SN74LS04N
U13:D
SN74LS08N
U68:C
S7
MSK SELECTOR (DOWN)
VAL
U56
SN74LS74AN
U57
U59:B
SN74LS86AN
U59:A
U58
OP27
BCK
R0[2]
CACK
DECLK
MSK DATA
MSK R1[2]
MSK R0[2]
+5V
+12V
-12V
GND
Title
Size
B
Number
Rev
Date
Drawn by
Filename
Sheet
of

Appendix B

The Firmware (Software) Listing

The firmware (software) listing is attached.

```
$DATE(03/27/93)
$TITLE(MDCOB.A51, VERSION 2.0, BY DONG WU)
$OBJECT(C:\WU\MDCOB.OBJ)
$ERRORPRINT(C:\WU\MDCOB.ERR)
$PRINT(C:\WU\MDCOB.LST)
$XREF
$NOMOD51
$INCLUDE(REG52.INC)
$EJECT
;**************************************************************************
;
;  Equates and Memory-Mapped I/O Addresses
;
;**************************************************************************
;
Q2334_BASE EQU     00H    ;external RAM address of the Q2334 register #0
Q0256_BASE EQU     20H    ;external RAM address of the Q0256 register #0
S2110_BASE EQU     40H    ;external RAM address of the STEL_2110 registe
CONTROL    EQU     60H

CLOCK           BIT     P3.0
MISTAKE         BIT     P3.1
TT1             BIT     P3.2
SWITCH          BIT     P3.3    ;begin the transmission
CLOCK1          BIT     P3.4
RUN             BIT     P3.5    ;indicate the transmitter running

           USING  0       ;inform assembler that we will use reg. bank 0
$EJECT
;**************************************************************************
;
;  Data Byte Segment (Internal RAM)
;
;**************************************************************************
;

            DSEG   AT 30H ;skip a byte to leave space for the DATA_BITS segmen
t

Q2334_REGS:     DS      8
F_TO_P:         DS      8       ;freq. to phase increment conversion factor
TMP:            DS      8       ;general scratchpad area
FREQ:           DS      4       ;hold the 32-bit value of frequency
FREQ1:          DS      4       ;use for MSK
FREQ2:          DS      4       ;use for MSK
TTT:            DS      2
TST_DR:         DS      1       ;currently data rate
STACK:          DS      0       ;stack starts just above the data area
$EJECT

;***************************************************************************
*
;
;  Data Bit Segment (Internal RAM)
;
;***************************************************************************
*
;
        BSEG    AT 0            ;position DATA-BITS segment at address

INI:    DBIT    1               ;initial flag
FLAG:   DBIT    1               ;general use flag
```

```
FLAG1:  DBIT    1               ;indicates the bit rate low or high
$EJECT

;***************************************************************************
*
;
;  Define the Interrupt Vectors
;
;***************************************************************************
*
;
        CSEG                    ;select the code segment

        ORG     RESET
        LJMP    START           ;system reset

        ORG     EXTI0
        RETI                    ;external interrup 0, not used

        ORG     TIMER0
        RETI                    ;timer 0 interrupt, not used

        ORG     EXTI1
        RETI                    ;external interrupt 1, not used

        ORG     TIMER1
        RETI                    ;timer 1 interrupt, not used

        ORG     SINT
        RETI

        ORG     TIMER2
        LJMP    T2_INT          ;timer 2 interrupt
$EJECT
;***************************************************************************
;                FUNCTION: START
;
;DESCRIPTION: This is the reset routine that is entered on power-up and
;             whenever the reset butten is pushed.
;
;***************************************************************************
***
;
START:  CLR     RUN
        CLR     MISTAKE
        MOV     IE,#00H         ;ensure that all interrupt are disabled
        MOV     SP,#STACK       ;initialize the stack
        MOV     PSW,#00H        ;use reg. bank 0 throughout this program

        LCALL   INIT_Q2334      ;initialize registers of Q2334 chip
        LCALL   INIT_S2110      ;initialize registers of STEL_2110A chip

        MOV     TST_DR,#00H     ;data rate=0
        MOV     IP,#00H         ;make all interrupt low priority
        SETB    PT2             ;except timer2 it has highest priorty
        MOV     T2CON,#00H      ;set up timer2, but don't start it running

        CLR     TR2
        SETB    ET2             ;enable its interrupt
        SETB    EA

        MOV     FREQ,#00H       ;set up f=4.8 MHZ in FREQ+0--FREQ+3
```

```
        MOV     FREQ+1,#3EH     ;f=4.8MHZ=00493E00H
        MOV     FREQ+2,#49H
        MOV     FREQ+3,#00H

        MOV     F_TO_P+4,#00H   ;this is the correct freq. to phase conversion

        MOV     F_TO_P+3,#8FH   ;factor for clock=30MHZ
        MOV     F_TO_P+2,#2AH   ;i.e. (2^32 / clock)*2^24
        MOV     F_TO_P+1,#63H
        MOV     F_TO_P+0,#39H

WWW1:   NOP
        NOP
        JNB     SWITCH,WWW1

        CLR     TT1
        NOP
        NOP
        MOV     A,P1

        JZ      WWW3
        MOV     R1,#00H

WWW2:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     WWW4
        MOV     A,R0
        RR      A
        SJMP    WWW2

WWW3:   MOV     R1,#00H

WWW4:   MOV     A,R1            ;valid input, recover number of the selection
        RL      A               ;multiply by 4 for a 4-byte jump table
        RL      A
        MOV     DPTR,#JMPTBL
        JMP     @A+DPTR

JMPTBL: LJMP    M_LOOP          ;A=0, N/A
        NOP
        LJMP    QPSK_TEST       ;A=1, QPSK function
        NOP
        LJMP    OQPSK_TEST      ;A=2, OQPSK function
        NOP
        LJMP    MSK_TEST        ;A=3, MSK function
        NOP
        LJMP    BPSK_TEST       ;A=4, BPSK function
        NOP

WWW:    SETB    TR2

WWWW:   NOP
        NOP
        SJMP    WWWW
$EJECT

M_LOOP: SETB    MISTAKE
WWW5:   NOP
        NOP
        JB      SWITCH,WWW5
        CLR     MISTAKE
```

```
        LJMP    START
$EJECT

;***************************************************************************
**
;                       FUNCTION: QPSK_TEST
;DESCRIPTION: This function runs the QPSK modulator/demodulator.
;
;***************************************************************************
**
;
QPSK_TEST:
        MOV     R4,FREQ             ;set up carrier frequency
        MOV     R5,FREQ+1
        MOV     R6,FREQ+2
        MOV     R7,FREQ+3

        ;set up DDS registers of Q2334

        MOV     R0,#F_TO_P
        MOV     R1,#Q2334_REGS
        LCALL   MULT                ;calculate what will be put into Q2334 registe
rs
                                    ;and put result into software copies of #1
        MOV     R0,#Q2334_BASE
        MOV     R1,#Q2334_REGS
        MOV     R2,#4
QT1:    MOV     A,@R1
        MOVX    @R0,A               ;put the result into the #1 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,QT1

        MOV     R0,#Q2334_BASE+10H
        MOV     R1,#Q2334_REGS
        MOV     R2,#4
QT2:    MOV     A,@R1
        MOVX    @R0,A               ;put the result into the #2 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,QT2

        MOV     R0,#08H             ;set up SMC register of #1(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#18H             ;set up SMC register of #2(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#0AH             ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH             ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#1AH             ;set up AMC register of #2(Q2334) and
        MOV     R1,#0EH             ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#Q2334_BASE+0CH          ;clear accumulator #1
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1CH          ;clear accumulator #2
        MOVX    @R0,A
```

```
;set control lines

MOV     R0,#CONTROL
MOV     A,#08H
MOVX    @R0,A
MOV     A,#00H
MOVX    @R0,A

SETB    INI             ;set initial condition
CLR     FLAG            ;FLAG=0 for QPSK
LCALL   GET_DR          ;set up data rate of timer2

MOV     R0,#15H
MOV     R1,#00H         ;set up registers 15H and 16H to 0
LCALL   WR_Q0256

MOV     R0,#16H
MOV     R1,#00H
LCALL   WR_Q0256

MOV     R0,#02H         ;set up control register 1
MOV     R1,#04H
LCALL   WR_Q0256

MOV     R0,#03H         ;set up control register 2
MOV     R1,#34H
LCALL   WR_Q0256

MOV     R0,#04H         ;set up control register 3
MOV     R1,#01H
LCALL   WR_Q0256

MOV     R0,#04H
MOV     R1,#05H
LCALL   WR_Q0256

MOV     R0,#08H         ;set up Normalization T count
MOV     R1,#0FCH
LCALL   WR_Q0256

MOV     R0,#09H         ;set up N count
MOV     R1,#0F9H
LCALL   WR_Q0256

MOV     R0,#0AH         ;set up BER period LS byte
MOV     R1,#0FCH        ;BER(0CH,0BH,0AH)=0FFFF9CH for 1E+5
LCALL   WR_Q0256        ;BER(0CH,0BH,0AH)=0FFD8F0H for 1E+7

MOV     R0,#0BH         ;set up BER period CS byte
MOV     R1,#0FFH
LCALL   WR_Q0256

MOV     R0,#0CH         ;set up BER period MS byte
MOV     R1,#0FFH
LCALL   WR_Q0256

MOV     R0,#17H
MOV     R1,#00H
LCALL   WR_Q0256

MOV     R0,#18H
```

```
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#06H         ;set up control register 1
        MOV     R1,#04H
        LCALL   WR_Q0256

        MOV     R0,#06H
        MOV     R1,#06H
        LCALL   WR_Q0256

        MOV     R0,#07H         ;set up control register 2
        MOV     R1,#30H
        LCALL   WR_Q0256

        ;set up the registers of STEL_2110A chip

        MOV     R0,#CONTROL
        MOV     A,#00H
        MOVX    @R0,A

        MOV     A,TST_DR
        RL      A
        RL      A
        MOV     DPTR,#JQPSK
        JMP     @A+DPTR

JQPSK:  LJMP    OUT_QPSK                ;A=0, N/A
        NOP
        LJMP    QPSK_1200               ;data rate=1200 bps
        NOP
        LJMP    QPSK_2400               ;data rate=2400 bps
        NOP
        LJMP    QPSK_4800               ;data rate=4800 bps
        NOP
        LJMP    QPSK_9600               ;data rate=9600 bps
        NOP
        LJMP    QPSK_192                ;data rate=19200 bps
        NOP

QPSK_1200:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#00H         ;BRCR(12H,13H,14H)=00A7C6H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#0A7H
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#0C6H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#02H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=67H
        MOV     R1,#26H         ;K1=(0111), K2=(0110)
        LCALL   WR_S2110

        LJMP    OUT_QPSK
```

```
QPSK_2400:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#01H         ;BRCR(12H,13H,14H)=014F8BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#4FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#8BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#02H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=78H
        MOV     R1,#37H         ;K1=(1000), K2=(0111)
        LCALL   WR_S2110

        LJMP    OUT_QPSK

QPSK_4800:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#02H         ;BRCR(12H,13H,14H)=029F17H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#9FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#17H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#02H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=89H
        MOV     R1,#48H         ;K1=(1001), K2=(1000)
        LCALL   WR_S2110

        LJMP    OUT_QPSK

QPSK_9600:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#05H         ;BRCR(12H,13H,14H)=053E2DH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#3EH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#2DH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#02H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=9AH
        MOV     R1,#59H         ;K1=(1010), K2=(1001)
```

```
        LCALL   WR_S2110

        LJMP    OUT_QPSK

QPSK_192:
        SETB    FLAG1
        MOV     R0,#12H             ;set Bit Rate Control Register
        MOV     R1,#0AH             ;BRCR(12H,13H,14H)=0A7C5BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#7CH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#5BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#02H
        MOVX    @R0,A

        MOV     R0,#11H             ;set Loop Gain Control Register LGCR(11H)=0ABH
        MOV     R1,#6AH             ;K1=(1011), K2=(1010)
        LCALL   WR_S2110

        LJMP    OUT_QPSK

OUT_QPSK:
        MOV     R0,#10H             ;set Timing Control Register TCR(10H)=08H
        MOV     R1,#08H
        LCALL   WR_S2110

        MOV     R0,#17H             ;SET Mode Control Register MCR(17H)=81H
        MOV     R1,#81H
        LCALL   WR_S2110

        LJMP    WWW
$EJECT
;*****************************************************************************
**
;                       FUNCTION: OQPSK_TEST
;DESCRIPTION: This function runs the OQPSK modulator/demodulator.
;
;*****************************************************************************
**
;
OQPSK_TEST:
        MOV     R4,FREQ                     ;set up carrier frequency
        MOV     R5,FREQ+1
        MOV     R6,FREQ+2
        MOV     R7,FREQ+3

        ;set up DDS registers of Q2334

        MOV     R0,#F_TO_P
        MOV     R1,#Q2334_REGS
        LCALL   MULT                ;calculate what will be put into Q2334 registe
rs
                                    ;and put result into software copies of #1
        MOV     R0,#Q2334_BASE
        MOV     R1,#Q2334_REGS
        MOV     R2,#4
OQT1:   MOV     A,@R1
```

```
        MOVX    @R0,A             ;put the result into the #1 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,OQT1

        MOV     R0,#Q2334_BASE+10H
        MOV     R1,#Q2334_REGS
        MOV     R2,#4
OQT2:   MOV     A,@R1
        MOVX    @R0,A             ;put the result into the #2 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,OQT2

        MOV     R0,#08H           ;set up SMC register of #1(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#18H           ;set up SMC register of #2(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#0AH           ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH           ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#1AH           ;set up AMC register of #2(Q2334) and
        MOV     R1,#0EH           ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#Q2334_BASE+0CH        ;clear accumulator #1
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1CH        ;clear accumulator #2
        MOVX    @R0,A

        ;set control lines

        MOV     R0,#CONTROL
        MOV     A,#38H
        MOVX    @R0,A
        MOV     A,#30H
        MOVX    @R0,A

        SETB    INI               ;set initial condition
        CLR     FLAG              ;FLAG=0 for OQPSK
        LCALL   GET_DR            ;set up data rate of timer2

        ;set up encoder registers of Q0256

        MOV     R0,#15H
        MOV     R1,#00H           ;set up registers 15H and 16H to 0
        LCALL   WR_Q0256

        MOV     R0,#16H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#02H           ;set up control register 1
        MOV     R1,#06H
        LCALL   WR_Q0256

        MOV     R0,#03H           ;set up control register 2
```

```
        MOV     R1,#34H
        LCALL   WR_Q0256

        MOV     R0,#04H         ;set up control register 3
        MOV     R1,#01H
        LCALL   WR_Q0256

        MOV     R0,#04H
        MOV     R1,#05H
        LCALL   WR_Q0256

        MOV     R0,#08H         ;set up Normalization T count
        MOV     R1,#0FCH
        LCALL   WR_Q0256

        MOV     R0,#09H         ;set up N count
        MOV     R1,#0F9H
        LCALL   WR_Q0256

        MOV     R0,#0AH         ;set up BER period LS byte
        MOV     R1,#0FCH        ;BER(0CH,0BH,0AH)=0FFFF9CH for 1E+5
        LCALL   WR_Q0256        ;BER(0CH,0BH,0AH)=0FFD8F0H for 1E+7

        MOV     R0,#0BH         ;set up BER period CS byte
        MOV     R1,#0FFH
        LCALL   WR_Q0256

        MOV     R0,#0CH         ;set up BER period MS byte
        MOV     R1,#0FFH
        LCALL   WR_Q0256

        MOV     R0,#17H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#18H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#06H         ;set up control register 1
        MOV     R1,#04H
        LCALL   WR_Q0256

        MOV     R0,#06H
        MOV     R1,#06H
        LCALL   WR_Q0256

        MOV     R0,#07H         ;set up control register 2
        MOV     R1,#30H
        LCALL   WR_Q0256

        ;set up the registers of STEL_2110A chip

        MOV     R0,#CONTROL
        MOV     A,#30H
        MOVX    @R0,A

        MOV     A,TST_DR
        RL      A
        RL      A
        MOV     DPTR,#JOQPSK
        JMP     @A+DPTR
```

```
JOQPSK: LJMP    OUT_OQPSK       ;A=0, N/A
        NOP
        LJMP    OQPSK_1200      ;data rate=1200 bps
        NOP
        LJMP    OQPSK_2400      ;data rate=2400 bps
        NOP
        LJMP    OQPSK_4800      ;data rate=4800 bps
        NOP
        LJMP    OQPSK_9600      ;data rate=9600 bps
        NOP
        LJMP    OQPSK_192       ;data rate=19200 bps
        NOP

OQPSK_1200:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#00H         ;BRCR(12H,13H,14H)=00A7C6H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#0A7H
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#0C6H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#32H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=67H
        MOV     R1,#26H
        LCALL   WR_S2110

        LJMP    OUT_OQPSK

OQPSK_2400:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#01H         ;BRCR(12H,13H,14H)=014F8BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#4FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#8BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#32H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=78H
        MOV     R1,#37H
        LCALL   WR_S2110

        LJMP    OUT_OQPSK

OQPSK_4800:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#02H         ;BRCR(12H,13H,14H)=029F17H
```

```
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#9FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#17H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#32H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=89H
        MOV     R1,#48H
        LCALL   WR_S2110

        LJMP    OUT_OQPSK

OQPSK_9600:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#05H         ;BRCR(12H,13H,14H)=053E2DH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#3EH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#2DH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#32H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=9AH
        MOV     R1,#59H
        LCALL   WR_S2110

        LJMP    OUT_OQPSK

OQPSK_192:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#0AH         ;BRCR(12H,13H,14H)=0A7C5BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#7CH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#5BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#32H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=0ABH
        MOV     R1,#6AH
        LCALL   WR_S2110

        LJMP    OUT_OQPSK
```

```
OUT_OQPSK:
        MOV     R0,#10H         ;set Timing Control Register TCR(10H)=08H
        MOV     R1,#08H
        LCALL   WR_S2110

        MOV     R0,#17H         ;SET Mode Control Register MCR(17H)=81H
        MOV     R1,#81H
        LCALL   WR_S2110

        LJMP    WWW
$EJECT

;*****************************************************************************
**
;                       FUNCTION: BPSK_TEST
;DESCRIPTION: This function runs the BPSK modulator/demodulator.
;
;*****************************************************************************
**
;
BPSK_TEST:
        MOV     R4,FREQ         ;set up carrier frequency
        MOV     R5,FREQ+1
        MOV     R6,FREQ+2
        MOV     R7,FREQ+3

        ;set up DDS registers of Q2334

        MOV     R0,#F_TO_P
        MOV     R1,#Q2334_REGS
        LCALL   MULT            ;calculate what will be put into Q2334 registe
rs
                                ;and put result into software copies of #1
        MOV     R0,#Q2334_BASE
        MOV     R1,#Q2334_REGS
        MOV     R2,#4
BT1:    MOV     A,@R1
        MOVX    @R0,A           ;put the result into the #1 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,BT1

        MOV     R0,#08H         ;set up SMC register of #1(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#0AH         ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH         ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#Q2334_BASE+0CH      ;clear accumulator #1
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1CH      ;clear accumulator #2
        MOVX    @R0,A

        MOV     R0,#Q2334_BASE+10H      ;clear A register of #2
        MOV     A,#00H
        MOV     R2,#4
BCLR_A: MOVX    @R0,A
        INC     R0
        DJNZ    R2,BCLR_A
```

```
MOV     R0,#Q2334_BASE+18H      ;clear SMC register of #2
MOV     A,#00H
MOVX    @R0,A
MOV     R0,#Q2334_BASE+1AH      ;clear AMC register of #2
MOV     A,#0FH
MOVX    @R0,A

;set control lines

MOV     R0,#CONTROL
MOV     A,#4DH
MOVX    @R0,A
MOV     A,#45H
MOVX    @R0,A

SETB    INI             ;set initial condition
SETB    FLAG            ;FLAG=1 for BPSK
LCALL   GET_DR          ;set up data rate of timer2

;set up encoder registers of Q0256

MOV     R0,#15H
MOV     R1,#00H         ;set up registers 15H and 16H to 0
LCALL   WR_Q0256

MOV     R0,#16H
MOV     R1,#00H
LCALL   WR_Q0256

MOV     R0,#02H         ;set up control register 1
MOV     R1,#05H
LCALL   WR_Q0256

MOV     R0,#03H         ;set up control register 2
MOV     R1,#30H
LCALL   WR_Q0256

MOV     R0,#04H         ;set up control register 3
MOV     R1,#01H
LCALL   WR_Q0256

MOV     R0,#04H
MOV     R1,#05H
LCALL   WR_Q0256

MOV     R0,#08H         ;set up Normalization T count
MOV     R1,#0FCH
LCALL   WR_Q0256

MOV     R0,#09H         ;set up N count
MOV     R1,#0F9H
LCALL   WR_Q0256

MOV     R0,#0AH         ;set up BER period LS byte
MOV     R1,#0FCH        ;BER(0CH,0BH,0AH)=0FFFF9CH for 1E+5
LCALL   WR_Q0256        ;BER(0CH,0BH,0AH)=0FFD8F0H for 1E+7

MOV     R0,#0BH         ;set up BER period CS byte
MOV     R1,#0FFH
LCALL   WR_Q0256

MOV     R0,#0CH         ;set up BER period MS byte
```

```
        MOV     R1,#0FFH
        LCALL   WR_Q0256

        MOV     R0,#17H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#18H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#06H         ;set up control register 1
        MOV     R1,#05H
        LCALL   WR_Q0256

        MOV     R0,#06H
        MOV     R1,#07H
        LCALL   WR_Q0256

        MOV     R0,#07H         ;set up control register 2
        MOV     R1,#30H
        LCALL   WR_Q0256

        ;set up the registers of STEL_2110A chip

        MOV     R0,#CONTROL
        MOV     A,#45H
        MOVX    @R0,A

        MOV     A,TST_DR
        RL      A
        RL      A
        MOV     DPTR,#JBPSK
        JMP     @A+DPTR

JBPSK:  LJMP    OUT_BPSK        ;A=0, N/A
        NOP
        LJMP    BPSK_1200       ;data rate=1200 bps
        NOP
        LJMP    BPSK_2400       ;data rate=2400 bps
        NOP
        LJMP    BPSK_4800       ;data rate=4800 bps
        NOP
        LJMP    BPSK_9600       ;data rate=9600 bps
        NOP
        LJMP    BPSK_192        ;data rate=19200 bps
        NOP

BPSK_1200:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#01H         ;BRCR(12H,13H,14H)=014F8BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#4FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#8BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#47H
```

```
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=78H
        MOV     R1,#78H
        LCALL   WR_S2110

        LJMP    OUT_BPSK

BPSK_2400:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#02H         ;BRCR(12H,13H,14H)=029F17H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#9FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#17H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#47H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=89H
        MOV     R1,#89H
        LCALL   WR_S2110

        LJMP    OUT_BPSK

BPSK_4800:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#05H         ;BRCR(12H,13H,14H)=053E2DH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#3EH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#2DH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#47H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=9AH
        MOV     R1,#9AH
        LCALL   WR_S2110

        LJMP    OUT_BPSK

BPSK_9600:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#0AH         ;BRCR(12H,13H,14H)=0A7C5BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#7CH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#5BH
```

```
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#47H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=0ABH
        MOV     R1,#0ABH
        LCALL   WR_S2110

        LJMP    OUT_BPSK

BPSK_192:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#14H         ;BRCR(12H,13H,14H)=14F8B6H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#0F8H
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#0B6H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#47H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=0BCH
        MOV     R1,#0BCH
        LCALL   WR_S2110

        LJMP    OUT_BPSK

OUT_BPSK:
        MOV     R0,#10H         ;set Timing Control Register TCR(10H)=08H
        MOV     R1,#08H
        LCALL   WR_S2110

        MOV     R0,#17H         ;SET Mode Control Register MCR(17H)=81H
        MOV     R1,#81H
        LCALL   WR_S2110

        LJMP    WWW
$EJECT

;*****************************************************************************
**
;                    FUNCTION: MSK_TEST
;DESCRIPTION: This function runs the MSK modulator/demodulator.
;
;*****************************************************************************
**
;
MSK_TEST:
        MOV     R4,FREQ         ;set up carrier frequency
        MOV     R5,FREQ+1
        MOV     R6,FREQ+2
        MOV     R7,FREQ+3

        ;set up DDS registers of Q2334
```

```
        MOV     R0,#F_TO_P
        MOV     R1,#FREQ1           ;calculate the phase increment for 4.8 MHZ
        LCALL   MULT                ;and put result into FREQ1+0--FREQ1+3

        SETB    FLAG                ;FLAG=1 for MSK
        LCALL   GET_DR              ;get data rate and offset 1/4 data rate holds
                                    ;in 4 registers: R7,R6,R5,R4.

        MOV     R0,#F_TO_P          ;calculate the phase increment for 1/4 data ra
te
        MOV     R1,#FREQ2           ;and put result into FREQ2+0--FREQ2+3
        LCALL   MULT

        LCALL   CALC_MSK            ;calculate 2 frequencies f+ and f-, then send
to
                                    ;DDS.

        MOV     R0,#08H             ;set up SMC register of #1(Q2334)
        MOV     R1,#04H
        LCALL   WR_Q2334

        MOV     R0,#0AH             ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH             ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#Q2334_BASE+0CH          ;clear accumulator #1
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1CH          ;clear accumulator #2
        MOVX    @R0,A

        MOV     R0,#Q2334_BASE+10H          ;clear A register of #2
        MOV     A,#00H
        MOV     R2,#4
MCLR_A: MOVX    @R0,A
        INC     R0
        DJNZ    R2,MCLR_A

        MOV     R0,#Q2334_BASE+18H          ;clear SMC register of #2
        MOV     A,#00H
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1AH          ;clear AMC register of #2
        MOV     A,#0FH
        MOVX    @R0,A

        ;set control lines

        SETB    INI                 ;set initial condition

        MOV     R0,#CONTROL
        MOV     A,#0CCH
        MOVX    @R0,A
        MOV     A,#0C4H
        MOVX    @R0,A

        ;set up encoder registers of Q0256

        MOV     R0,#15H
        MOV     R1,#00H             ;set up registers 15H and 16H to 0
        LCALL   WR_Q0256

        MOV     R0,#16H
        MOV     R1,#00H
```

```
        LCALL   WR_Q0256

        MOV     R0,#02H         ;set up control register 1
        MOV     R1,#04H
        LCALL   WR_Q0256

        MOV     R0,#03H         ;set up control register 2
        MOV     R1,#34H
        LCALL   WR_Q0256

        MOV     R0,#04H         ;set up control register 3
        MOV     R1,#01H
        LCALL   WR_Q0256

        MOV     R0,#04H         ;set up control register 3
        MOV     R1,#05H
        LCALL   WR_Q0256

        MOV     R0,#08H         ;set up Normalization T count
        MOV     R1,#0FCH
        LCALL   WR_Q0256

        MOV     R0,#09H         ;set up N count
        MOV     R1,#0F9H
        LCALL   WR_Q0256

        MOV     R0,#0AH         ;set up BER period LS byte
        MOV     R1,#0FCH        ;BER(0CH,0BH,0AH)=0FFFF9CH for 1E+5
        LCALL   WR_Q0256        ;BER(0CH,0BH,0AH)=0FFD8F0H for 1E+7

        MOV     R0,#0BH                 ;set up BER period CS byte
        MOV     R1,#0FFH
        LCALL   WR_Q0256

        MOV     R0,#0CH                 ;set up BER period MS byte
        MOV     R1,#0FFH
        LCALL   WR_Q0256

        MOV     R0,#17H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#18H
        MOV     R1,#00H
        LCALL   WR_Q0256

        MOV     R0,#06H         ;set up control register 1
        MOV     R1,#04H
        LCALL   WR_Q0256

        MOV     R0,#06H
        MOV     R1,#06H
        LCALL   WR_Q0256

        MOV     R0,#07H         ;set up control register 2
        MOV     R1,#30H
        LCALL   WR_Q0256

        ;set up the registers of STEL_2110A chip

        MOV     R0,#CONTROL
        MOV     A,#0C4H
```

```
        MOVX    @R0,A

        MOV     A,TST_DR
        RL      A
        RL      A
        MOV     DPTR,#JMSK
        JMP     @A+DPTR

JMSK:   LJMP    OUT_MSK         ;A=0, N/A
        NOP
        LJMP    MSK_1200        ;data rate=1200 bps
        NOP
        LJMP    MSK_2400        ;data rate=2400 bps
        NOP
        LJMP    MSK_4800        ;data rate=4800 bps
        NOP
        LJMP    MSK_9600        ;data rate=9600 bps
        NOP
        LJMP    MSK_192         ;data rate=19200 bps
        NOP

MSK_1200:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#01H         ;BRCR(12H,13H,14H)=00A7C6H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#4FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#8BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#0C6H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=67H
        MOV     R1,#67H
        LCALL   WR_S2110

        LJMP    OUT_MSK

MSK_2400:
        CLR     FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#02H         ;BRCR(12H,13H,14H)=014F8BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#9FH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#17H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#0C6H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=78H
        MOV     R1,#89H
        LCALL   WR_S2110
```

```
        LJMP    OUT_MSK

MSK_4800:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#05H         ;BRCR(12H,13H,14H)=029F17H
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#3EH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#2DH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#0C6H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=89H
        MOV     R1,#9AH
        LCALL   WR_S2110

        LJMP    OUT_MSK

MSK_9600:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#0AH         ;BRCR(12H,13H,14H)=053E2DH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#7CH
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#5BH
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#0C6H
        MOVX    @R0,A

        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=9AH
        MOV     R1,#0ABH
        LCALL   WR_S2110

        LJMP    OUT_MSK

MSK_192:
        SETB    FLAG1
        MOV     R0,#12H         ;set Bit Rate Control Register
        MOV     R1,#14H         ;BRCR(12H,13H,14H)=0A7C5BH
        LCALL   WR_S2110
        MOV     R0,#13H
        MOV     R1,#0F8H
        LCALL   WR_S2110
        MOV     R0,#14H
        MOV     R1,#0B6H
        LCALL   WR_S2110

        MOV     R0,#CONTROL
        MOV     A,#0C6H
        MOVX    @R0,A
```

```
        MOV     R0,#11H         ;set Loop Gain Control Register LGCR(11H)=0ABH
        MOV     R1,#0BCH
        LCALL   WR_S2110

        LJMP    OUT_MSK

OUT_MSK:
        MOV     R0,#10H         ;set Timing Control Register TCR(10H)=08H
        MOV     R1,#08H
        LCALL   WR_S2110

        MOV     R0,#17H         ;SET Mode Control Register MCR(17H)=81H
        MOV     R1,#81H
        LCALL   WR_S2110

        LJMP    WWW
$EJECT

;***************************************************************************
*********
;                       FUNCTION: CALC_MSK
;DESCRIPTION:This function is called to do the computations and set up the
;           DDS chip for MSK modulation.
;       (1) FREQ1 holds the 32-bit number to send to DDS for basic carrier.
;       (2) FREQ2 holds the +/- offset that must be added/substructed from
;           the basic carrier.
;           PIRA and PIRB registers will be set for MSK.
;***************************************************************************
**
;
CALC_MSK:
        MOV     R3,#00H         ;address of PIRA register (DDS#1)
        MOV     R0,#FREQ1
        MOV     R1,#FREQ2
        MOV     R4,#4
        CLR     C
CM1:    MOV     A,@R0           ;put FREQ1-FREQ2 in PIRA register (fc-1/4T)
        SUBB    A,@R1
        XCH     A,R1
        XCH     A,R3
        XCH     A,R0
        PUSH    ACC
        PUSH    PSW
        LCALL   WR_Q2334
        POP     PSW
        POP     ACC
        XCH     A,R0
        XCH     A,R3
        XCH     A,R1
        INC     R0
        INC     R1
        INC     R3
        DJNZ    R4,CM1

        MOV     R0,#FREQ1       ;put FREQ1+FREQ2 in PIRB register (fc+1/4T)
        MOV     R1,#FREQ2
        MOV     R4,#4
        CLR     C
CM2:    MOV     A,@R0
        ADDC    A,@R1
        XCH     A,R1
```

```
        XCH     A,R3
        XCH     A,R0
        PUSH    ACC
        PUSH    PSW
        LCALL   WR_Q2334
        POP     PSW
        POP     ACC
        XCH     A,R0
        XCH     A,R3
        XCH     A,R1
        INC     R0
        INC     R1
        INC     R3
        DJNZ    R4,CM2
        RET
$EJECT
;***************************************************************************
*****
;                         FUNCTION: MULT
;DESCRIPTION: The function MULT multiplies the 4-byte number in R4-R7 by the
;             5-byte number pointed to by R0 (F_TO_P). It is assumed that the
;             product will be no longer than 7 bytes. The least significant 3
;             bytes are dropped, (corresponding to a divide by 2^24) and the
;             remaining 4-byte number is placed in the location pointed to by
;             R1.
;***************************************************************************
********
;
MULT:
        PUSH    AR0
        MOV     R0,#TMP
        MOV     R2,#07H
M1:     MOV     @R0,#00H        ;first, zero the product space (TMP0-6)
        INC     R0
        DJNZ    R2,M1

        POP     AR0             ;recover address of multiplier
        PUSH    AR1             ;save location for final result

        MOV     R1,#TMP         ;put 7-byte product in TMP0-6

        MOV     A,R4            ;first, multiply by byte #1
        MOV     R2,A
        MOV     R3,#05H
        LCALL   MULT_DIG
        INC     R1

        MOV     A,R5            ;then, multiply by byte #2
        MOV     R2,A
        MOV     R3,#05H
        LCALL   MULT_DIG
        INC     R1

        MOV     A,R6            ;then, multiply by byte #3
        MOV     R2,A
        MOV     R3,#05H
        LCALL   MULT_DIG
        INC     R1

        MOV     A,R7            ;then, multiply by byre #4
        MOV     R2,A
```

```
        MOV     R3,#04H
        LCALL   MULT_DIG

        POP     AR1                 ;recover where we want to put result (TMP3-6)
        MOV     A,TMP+3
        MOV     @R1,A
        INC     R1
        MOV     A,TMP+4
        MOV     @R1,A
        INC     R1
        MOV     A,TMP+5
        MOV     @R1,A
        INC     R1
        MOV     A,TMP+6
        MOV     @R1,A
        RET
$EJECT
;***************************************************************************
*******
;                       FUNCTION: MULT_DIG
;DESCRIPTION: The function MULT_DIG is a general purpose multiplication
;             routine. It multiplies a single byte by an arbitrarily large
;             number.
;               R0=location of multiplier,      R1=location of product
;               R2=single_byte multiplier,      R3=# of bytes to multiply
;
;***************************************************************************
*****
;
MULT_DIG:
        PUSH    AR0                 ;want to return with R0 & R1 unchanged
        PUSH    AR1
MD1:    MOV     B,R2                ;get the byte we're multiplying by
        MOV     A,@R0               ;get one byte of the number we're multiplying
        MUL     AB
        ADD     A,@R1
        MOV     @R1,A
        INC     R1
        MOV     A,B
        ADDC    A,@R1
        MOV     @R1,A
        PUSH    AR1
        MOV     A,#0

MD2:    JNC     MD3
        INC     R1
        ADDC    A,@R1
        MOV     @R1,A
        SJMP    MD2

MD3:    POP     AR1
        INC     R0
        DJNZ    R3,MD1

        POP     AR1
        POP     AR0
        RET
$EJECT
;***************************************************************************
*****
;                    FUNCTION: WR_Q2334
;DESCRIPTION: The function WR_Q2334 writes a new value to a port of Q2334
```

```
;
;                R0=port number,     R1=value
;
;***************************************************************************
******
;
WR_Q2334:
        PUSH    AR0             ;want to be able to exit with R0 unchanged
        PUSH    AR1
        MOV     A,R0
        ADD     A,#Q2334_BASE
        MOV     R0,A
        MOV     A,R1
        MOVX    @R0,A
        POP     AR1
        POP     AR0             ;recover R0
        RET
$EJECT
;***************************************************************************
*****
;                    FUNCTION: WR_Q0256
;DESCRIPTION: The function WR_Q0256 writes a new value to a port of Q0256
;             and its software copies.
;               R0=port number,          R1=value
;
;***************************************************************************
***
;
WR_Q0256:
        PUSH    AR0
        PUSH    AR1
        MOV     A,R0
        ADD     A,#Q0256_BASE
        MOV     R0,A
        MOV     A,R1
        MOVX    @R0,A
        POP     AR1
        POP     AR0
        RET
$EJECT
;***************************************************************************
*****
;                    FUNCTION: WR_S2110
;DESCRIPTION: The function WR_S2110 writes a new value to a port of STEL-2110A
;             and its software copies.
;               R0=port number,          R1=value
;
;***************************************************************************
***
;
WR_S2110:
        PUSH    AR0
        PUSH    AR1
        MOV     A,R0
        ADD     A,#S2110_BASE
        MOV     R0,A
        MOV     A,R1
        MOVX    @R0,A
        POP     AR1
        POP     AR0
        RET
```

```
$EJECT

GET_DR: SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      WWW7
        MOV     R1,#00H

WWW6:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     WWW8
        MOV     A,R0
        RR      A
        SJMP    WWW6
WWW7:   MOV     R1,#00H

WWW8:   MOV     A,R1                ;valid input, recover number of the selection
        RL      A
        RL      A
        MOV     DPTR,#JMPDR
        JMP     @A+DPTR

JMPDR:  LJMP    OUT_DR
        NOP
        LJMP    DR_1200             ;data rate=1200 bps
        NOP
        LJMP    DR_2400             ;data rate=2400 bps
        NOP
        LJMP    DR_4800             ;data rate=4800 bps
        NOP
        LJMP    DR_9600             ;data rate=9600 bps
        NOP
        LJMP    DR_192              ;data rate=19200 bps
        NOP

DR_1200:JB      FLAG,DR12_1         ;set timer2 for 1200 bps

        MOV     RCAP2L,#80H         ;QPSK (1/2 R)   (NOTE:TL2 &TH2)
        MOV     RCAP2H,#0FEH
        SJMP    DR12_2
DR12_1: MOV     RCAP2L,#40H         ;OQPSK, MSK, BPSK (R)
        MOV     RCAP2H,#0FFH
        MOV     R4,#58H
        MOV     R5,#02H
        MOV     R6,#00H
        MOV     R7,#00H
DR12_2: MOV     TST_DR,#01H         ;TST_DR=1 (1200 bps)
        LJMP    OUT_DR

DR_2400:JB      FLAG,DR24_1         ;set timer2 for 2400 bps

        MOV     RCAP2L,#40H         ;QPSK (1/2 R)
        MOV     RCAP2H,#0FFH
        SJMP    DR24_2
DR24_1: MOV     RCAP2L,#0A0H                ;OQPSK, MSK, BPSK (R)
        MOV     RCAP2H,#0FFH
        MOV     R4,#60H
```

```
        MOV     R5,#04H
        MOV     R6,#00H
        MOV     R7,#00H
DR24_2: MOV     TST_DR,#02H     ;TST_DR=2 (2400 bps)
        LJMP    OUT_DR

DR_4800:JB      FLAG,DR48_1     ;set timer2 for 4800 bps

        MOV     RCAP2L,#0A0H            ;QPSK (1/2 R)
        MOV     RCAP2H,#0FFH
        SJMP    DR48_2
DR48_1: MOV     RCAP2L,#0D0H    ;OQPSK, MSK, BPSK (R)
        MOV     RCAP2H,#0FFH
        MOV     R4,#60H
        MOV     R5,#09H
        MOV     R6,#00H
        MOV     R7,#00H
DR48_2: MOV     TST_DR,#03H     ;TST_DR=3 (4800 bps)
        LJMP    OUT_DR

DR_9600:JB      FLAG,DR96_1     ;set timer2 for 9600 bps

        MOV     RCAP2L,#0D0H    ;QPSK (1/2 R)
        MOV     RCAP2H,#0FFH
        SJMP    DR96_2
DR96_1: MOV     RCAP2L,#0E8H    ;OQPSK, MSK, BPSK (R)
        MOV     RCAP2H,#0FFH
        MOV     R4,#0C0H
        MOV     R5,#12H
        MOV     R6,#00H
        MOV     R7,#00H
DR96_2: MOV     TST_DR,#04H     ;TST_DR=4 (9600 bps)
        LJMP    OUT_DR

DR_192: JB      FLAG,DR192_1    ;set timer2 for 19200 bps

        MOV     RCAP2L,#0E8H    ;QPSK  (1/2 R)
        MOV     RCAP2H,#0FFH
        SJMP    DR192_2
DR192_1:MOV     RCAP2L,#0F4H    ;OQPSK, MSK, BPSK (R)
        MOV     RCAP2H,#0FFH
        MOV     R4,#80H
        MOV     R5,#25H
        MOV     R6,#00H
        MOV     R7,#00H
DR192_2:MOV     TST_DR,#05H     ;TST_DR=5 (19200 bps)

OUT_DR: RET
$EJECT

;*****************************************************************************
**
;                       FUNCTION: INIT_Q2334
;DESCRIPTION: This routine is called on reset to initialize all the registers
;             in the DDS chip.
;
;*****************************************************************************
**
;
INIT_Q2334:
        MOV     R0,#00H
        MOV     R1,#00H
```

```
ID1:    LCALL   WR_Q2334        ;fill #1 frequency registers with 0
        INC     R0
        CJNE    R0,#8H,ID1

        MOV     R0,#10H
ID2:    LCALL   WR_Q2334        ;fill #2 frequency registers with 0
        INC     R0
        CJNE    R0,#18H,ID2

        MOV     R0,#08H         ;clear #1, mode_ctrl1 (SMC)
        LCALL   WR_Q2334
        MOV     R0,#18H         ;clear #2, mode_ctrl2  (SMC)
        LCALL   WR_Q2334

        MOV     R1,#00H
        MOV     R0,#0AH
        LCALL   WR_Q2334        ;clear #1 AMC
        MOV     R0,#1AH
        LCALL   WR_Q2334        ;clear #2 AMC

        MOV     R0,#Q2334_BASE+0CH      ;clear #1, accumulator
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1CH      ;clear #2, accumulator
        MOVX    @R0,A

        MOV     R0,#Q2334_BASE+0EH      ;updata
        MOVX    @R0,A
        MOV     R0,#Q2334_BASE+1EH      ;updata
        MOVX    @R0,A

        RET
$EJECT
;***************************************************************************
*
;                        FUNCTION: INIT_Q0256
;DESCRIPTION: This routine is called on reset to initialize all the registers.
;
;***************************************************************************
*
;
INIT_Q0256:

        MOV     R0,#04H
        MOV     R1,#04H
        LCALL   WR_Q0256

        MOV     R0,#06H
        MOV     R1,#02H
        LCALL   WR_Q0256

        MOV     R0,#00H
        MOV     R1,#00H
ID3:    LCALL   WR_Q0256
        MOV     R1,#00H
        INC     R0
        CJNE    R0,#0DH,ID3

        RET
$EJECT
;***************************************************************************
**
;                        FUNCTION: INIT_S2110
```

```
;DESCRIPTION: This routine is called on reset to initialize all the registers
;
;*************************************************************************
**
;
INIT_S2110:
        MOV     R0,#S2110_BASE+00H      ;set Timing Control Register TCR(10H)=
01H
        MOV     A,#01H
        MOVX    @R0,A
        MOV     R0,#S2110_BASE+01H      ;set Loop Gain Control Register LGCR(1
1H)=B7H
        MOV     A,#0B7H
        MOVX    @R0,A
        MOV     R0,#S2110_BASE+02H      ;set Bit Rate Control Register
        MOV     A,#00H
        MOVX    @R0,A                   ;BRCR(12H,13H,14H)=000000H
        MOV     R0,#S2110_BASE+03H
        MOVX    @R0,A
        MOV     R0,#S2110_BASE+04H
        MOVX    @R0,A
        MOV     R0,#S2110_BASE+07H      ;SET Mode Control Register MCR(17H)=81
H
        MOVX    @R0,A
        RET
$EJECT
;*************************************************************************
***
;                     FUNCTION: T2_INT
;DESCRIPTION: This is the TIMER2 Interrupt service routine. It is used to
;             precisely time writes to the DDS (Q2334) chip to run the
;             modulations.
;              INT=1 ;for initial tim,  and  INT=0  ;for other times
;              ENCCLKOUT=1 ;for no signal,  and  ENCCLKOUT=0 ; for signal in
;
;*************************************************************************
**
;
T2_INT: CLR     TF2             ;clear timer2 interrupt flag
        SETB    RUN
        SETB    CLOCK1
        JB      FLAG1,CCC
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
```

```
        NOP
        NOP
        NOP
CCC:    CLR     CLOCK1
        SETB    CLOCK
        CLR     CLOCK
        RETI
$EJECT

        END                     ;end of the program
```

```
$DATE(02/24/93)
$TITLE(PLL.A51, VERSION 2.0, BY DONG WU)
$OBJECT(C:\WU\PLL.OBJ)
$ERRORPRINT(C:\WU\PLL.ERR)
$PRINT(C:\WU\PLL.LST)
$XREF
$NOMOD51
$INCLUDE(REG52.INC)
$EJECT
;***************************************************************************
*
;
;               Equates and Memory-Mapped I/O Addresses
;
;***************************************************************************
*
;
Q2334_BASE      EQU     00H     ;external RAM address of the Q2334 register #0

PMCLK           BIT     P3.0
MODE            BIT     P3.1
HOPCLK          BIT     P3.3
TT0             BIT     P3.4
TT1             BIT     P3.5

                USING   0       ;inform assembler that we will use reg. bank 0
$EJECT
;***************************************************************************
*
;
;               Data Byte Segment (Internal RAM)
;
;***************************************************************************
*
;
                DSEG    AT 28H

Q2334_REGS:     DS      8
DF:             DS      8
DFCA:           DS      8       ;area keeping for digital filter calculation
DFCA1:          DS      8
Q2334_REGS1:    DS      8
Q2334_REGS2:    DS      8
DATA1:          DS      1
DATA2:          DS      1
STACK:          DS      0       ;stack starts just above the data area
$EJECT
;***************************************************************************
*
;
;               Data Bit Segment (Internal RAM)
;
;***************************************************************************
*
;
                BSEG    AT 0    ;position DATA-BITS segment at address 20H

INI1:           DBIT    1       ;initial flag
INI2:           DBIT    1
$EJECT
;***************************************************************************
*
```

```
;
;               Define the Interrupt Vectors
;
;*************************************************************************
*
;
                CSEG            ;select the code segment

                ORG     RESET
                LJMP    START   ;system reset

                ORG     EXTI0
                LJMP    CAL     ;external interrup 0, used for digital filter

                ORG     TIMER0
                RETI            ;timer 0 interrupt, not used

                ORG     EXTI1
                RETI            ;external interrupt 1, not used

                ORG     TIMER1
                RETI            ;timer 1 interrupt, not used

                ORG     SINT
                RETI            ;serial port interrupt, not used

                ORG     TIMER2
                RETI            ;timer 2 interrupt, not used
$EJECT
;*************************************************************************
;                 FUNCTION: START
;
;DESCRIPTION: This is the reset routine that is entered on power-up and
;             whenever the reset butten is pushed.
;
;*************************************************************************
***
;
START:  MOV     IE,#00H         ;ensure that all interrupt are disabled
        MOV     SP,#STACK       ;initialize the stack
        MOV     PSW,#00H        ;use reg. bank 0 throughout this program
        MOV     TCON,#00H

        LCALL   INIT_Q2334      ;initialize registers of Q2334 chip

        SETB    IT0             ;external interrupt 0 edge triggered
        MOV     IP,#00H         ;make all interrupt low priority
        SETB    PX0

        JNB     MODE,PL
        SJMP    PP
PL:     LJMP    PLLL

PP:     SETB    TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DDD3
        MOV     R1,#00H
```

```
DDD2:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DDD4
        MOV     A,R0
        RR      A
        SJMP    DDD2

DDD3:   MOV     R1,#00H

DDD4:   MOV     A,R1
        RL      A
        RL      A
        MOV     DPTR,#TLQD
        JMP     @A+DPTR

TLQD:   LJMP    M_LOOP
        NOP
        LJMP    RATED1
        NOP
        LJMP    RATED2
        NOP
        LJMP    RATED3
        NOP
        LJMP    RATED4
        NOP
        LJMP    RATED5
        NOP

RATED1: MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#070H
        MOV     Q2334_REGS+2,#0B2H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2
        SJMP    COM

RATED2: MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#070H
        MOV     Q2334_REGS+2,#0B2H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2
        SJMP    COM

RATED3: MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#070H
        MOV     Q2334_REGS+2,#072H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2
        SJMP    COM

RATED4: MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#070H
        MOV     Q2334_REGS+2,#0B2H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2
        SJMP    COM

RATED5: MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#070H
        MOV     Q2334_REGS+2,#0B2H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2
```

```
        SJMP    COM

COM:    MOV     R0,#08H         ;set up SMC register of #1(Q2334) to EPM
        MOV     R1,#00H
        LCALL   WR_Q2334

        MOV     R0,#18H         ;set up SMC register of #2(Q2334) to EPM
        MOV     R1,#00H
        LCALL   WR_Q2334

        MOV     R0,#0AH         ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH         ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#1AH         ;set up AMC register of #2(Q2334) and
        MOV     R1,#0EH         ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        SETB    PMCLK
        NOP
        NOP
        CLR     PMCLK

        SETB    HOPCLK
        NOP
        NOP
        CLR     HOPCLK

        LJMP    WWW

PLLL:   MOV     Q2334_REGS,#8FH
        MOV     Q2334_REGS+1,#0DH
        MOV     Q2334_REGS+2,#0F6H
        MOV     Q2334_REGS+3,#28H
        LCALL   W2

        MOV     R0,#08H         ;set up SMC register of #1(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#18H         ;set up SMC register of #2(Q2334) to EPM
        MOV     R1,#02H
        LCALL   WR_Q2334

        MOV     R0,#0AH         ;set up AMC register of #1(Q2334) and
        MOV     R1,#0EH         ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        MOV     R0,#1AH         ;set up AMC register of #2(Q2334) and
        MOV     R1,#0EH         ;enable NRC and D/A = 12-bit
        LCALL   WR_Q2334

        SETB    PMCLK
        NOP
        NOP
        CLR     PMCLK

        SETB    HOPCLK
        NOP
        NOP
        CLR     HOPCLK
```

```
        CLR     TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DD3
        MOV     R1,#00H

DD2:    INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DD4
        MOV     A,R0
        RR      A
        SJMP    DD2

DD3:    MOV     R1,#00H

DD4:    MOV     A,R1
        RL      A
        RL      A
        MOV     DPTR,#TL
        JMP     @A+DPTR

TL:     LJMP    M_LOOP
        NOP
        LJMP    QPSKTL
        NOP
        LJMP    OQPSKTL
        NOP
        LJMP    MSKTL
        NOP
        LJMP    BPSKTL
        NOP

DD:     SETB    EA
        SETB    EX0

WWW:    NOP
        NOP
        SJMP    WWW
$EJECT

M_LOOP: LJMP    START
$EJECT

QPSKTL: SETB    TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DD31
        MOV     R1,#00H

DD21:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DD41
        MOV     A,R0
        RR      A
```

```
        SJMP    DD21

DD31:   MOV     R1,#00H

DD41:   MOV     A,R1
        RL      A
        RL      A
        MOV     DPTR,#TLQ
        JMP     @A+DPTR

TLQ:    LJMP    M_LOOP
        NOP
        LJMP    RATE11
        NOP
        LJMP    RATE21
        NOP
        LJMP    RATE31
        NOP
        LJMP    RATE41
        NOP
        LJMP    RATE51
        NOP

RATE11: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE21: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE31: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE41: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE51: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD
$EJECT

OQPSKTL:SETB    TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DD32
        MOV     R1,#00H

DD22:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DD42
        MOV     A,R0
        RR      A
        SJMP    DD22

DD32:   MOV     R1,#00H
```

```
DD42:   MOV     A,R1
        RL      A
        RL      A
        MOV     DPTR,#TLO
        JMP     @A+DPTR

TLO:    LJMP    M_LOOP
        NOP
        LJMP    RATE12
        NOP
        LJMP    RATE22
        NOP
        LJMP    RATE32
        NOP
        LJMP    RATE42
        NOP
        LJMP    RATE52
        NOP

RATE12: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE22: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE32: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE42: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE52: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD
$EJECT

MSKTL:  SETB    TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DD33
        MOV     R1,#00H

DD23:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DD43
        MOV     A,R0
        RR      A
        SJMP    DD23

DD33:   MOV     R1,#00H

DD43:   MOV     A,R1
        RL      A
```

```
        RL      A
        MOV     DPTR,#TLM
        JMP     @A+DPTR

TLM:    LJMP    M_LOOP
        NOP
        LJMP    RATE13
        NOP
        LJMP    RATE23
        NOP
        LJMP    RATE33
        NOP
        LJMP    RATE43
        NOP
        LJMP    RATE53
        NOP

RATE13: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE23: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE33: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE43: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE53: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD
$EJECT

BPSKTL: SETB    TT0
        SETB    TT1
        NOP
        NOP
        MOV     A,P1

        JZ      DD34
        MOV     R1,#00H

DD24:   INC     R1
        MOV     R0,A
        ANL     A,#01H
        JNZ     DD44
        MOV     A,R0
        RR      A
        SJMP    DD24

DD34:   MOV     R1,#00H

DD44:   MOV     A,R1
        RL      A
        RL      A
        MOV     DPTR,#TLB
        JMP     @A+DPTR
```

```
TLB:    LJMP    M_LOOP
        NOP
        LJMP    RATE14
        NOP
        LJMP    RATE24
        NOP
        LJMP    RATE34
        NOP
        LJMP    RATE44
        NOP
        LJMP    RATE54
        NOP

RATE14: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE24: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE34: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE44: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD

RATE54: MOV     DATA1,#20H
        MOV     DATA2,#10H
        LJMP    DD
$EJECT

CAL:    SETB    INI1

        PUSH    Q2334_REGS+3
        PUSH    Q2334_REGS+2
        PUSH    Q2334_REGS+1
        PUSH    Q2334_REGS

        CLR     TT0
        CLR     TT1
        NOP
        NOP
        MOV     DF+1,P1

        SETB    TT0
        CLR     TT1
        NOP
        NOP
        MOV     DF,P1

        MOV     A,DF+1
        ANL     A,#80H
        JZ      W1

        MOV     A,DF+1
        XRL     A,#0FFH
        MOV     DF+3,A
        MOV     A,DF
```

```
        XRL     A,#0FFH
        MOV     DF+2,A

        MOV     R0,#DF+2
        MOV     R1,#Q2334_REGS1
        MOV     R2,DATA1
        LCALL   MULT

        MOV     R1,#Q2334_REGS
        MOV     R0,#Q2334_REGS1
        LCALL   AD
        LCALL   W2

        POP     Q2334_REGS
        POP     Q2334_REGS+1
        POP     Q2334_REGS+2
        POP     Q2334_REGS+3

        MOV     R0,#DF+2
        MOV     R1,#Q2334_REGS2
        MOV     R2,DATA2
        LCALL   MULT1

        MOV     R1,#Q2334_REGS
        MOV     R0,#Q2334_REGS2
        LCALL   AD1
        LJMP    W3

W1:     MOV     A,DF+1
        MOV     DF+5,A
        MOV     A,DF
        MOV     DF+4,A

        MOV     R0,#DF+4
        MOV     R1,#Q2334_REGS1
        MOV     R2,DATA1
        LCALL   MULT

        MOV     R1,#Q2334_REGS
        MOV     R0,#Q2334_REGS1
        LCALL   SB
        LCALL   W2

        POP     Q2334_REGS
        POP     Q2334_REGS+1
        POP     Q2334_REGS+2
        POP     Q2334_REGS+3

        MOV     R0,#DF+4
        MOV     R1,#Q2334_REGS2
        MOV     R2,DATA2
        LCALL   MULT1

        MOV     R1,#Q2334_REGS
        MOV     R0,#Q2334_REGS2
        LCALL   SB1

W3:     SETB    HOPCLK          ;strobe HOPCLK so this setup takes affect
        NOP
        NOP
        CLR     HOPCLK
```

```
        SETB    PMCLK
        NOP
        NOP
        CLR     PMCLK

        SETB    INI2
        RETI
$EJECT

W2:     MOV     R0,#Q2334_BASE
        MOV     R1,#Q2334_REGS
        MOV     R2,#04H
QT1:    MOV     A,@R1
        MOVX    @R0,A           ;put the result into the #1 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,QT1

        MOV     R0,#Q2334_BASE+10H
        MOV     R1,#Q2334_REGS
        MOV     R2,#04H
QT2:    MOV     A,@R1
        MOVX    @R0,A           ;put the result into the #2 of Q2334 chip
        INC     R1
        INC     R0
        DJNZ    R2,QT2
        RET
$EJECT

MULT:   PUSH    AR1
        PUSH    AR2
        PUSH    AR0
        MOV     R0,#DFCA
        MOV     R2,#06H
W11:    MOV     @R0,#00H
        INC     R0
        DJNZ    R2,W11

        POP     AR0
        POP     AR2
        MOV     R1,#DFCA
        MOV     R3,#2
        LCALL   MULT_DIG

        POP     AR1
        MOV     A,DFCA+0
        MOV     @R1,A
        INC     R1
        MOV     A,DFCA+1
        MOV     @R1,A
        INC     R1
        MOV     A,DFCA+2
        MOV     @R1,A
        RET
$EJECT

MULT1:  PUSH    AR1
        PUSH    AR2
        PUSH    AR0
        MOV     R0,#DFCA1
        MOV     R2,#06H
W111:   MOV     @R0,#00H
```

```
        INC     R0
        DJNZ    R2,W111

        POP     AR0
        POP     AR2
        MOV     R1,#DFCA1
        MOV     R3,#2
        LCALL   MULT_DIG

        POP     AR1
        MOV     A,DFCA1+1
        MOV     @R1,A
        INC     R1
        MOV     A,DFCA1+2
        MOV     @R1,A
        RET
$EJECT

MULT_DIG:
        PUSH    AR0             ;want to return with R0 & R1 unchanged
        PUSH    AR1
MD1:    MOV     B,R2            ;get the byte we're multiplying by
        MOV     A,@R0           ;get one byte of the number we're multiplying
        MUL     AB
        ADD     A,@R1
        MOV     @R1,A
        INC     R1
        MOV     A,B
        ADDC    A,@R1
        MOV     @R1,A
        PUSH    AR1
        MOV     A,#0

MD2:    JNC     MD3
        INC     R1
        ADDC    A,@R1
        MOV     @R1,A
        SJMP    MD2

MD3:    POP     AR1
        INC     R0
        DJNZ    R3,MD1

        POP     AR1
        POP     AR0
        RET
$EJECT

AD:     CLR     C
        MOV     A,@R1
        ADD     A,@R0
        MOV     @R1,A

        INC     R1
        INC     R0
        MOV     A,@R1
        ADDC    A,@R0
        MOV     @R1,A

        INC     R1
        INC     R0
        MOV     A,@R1
```

```
        ADDC    A,@R0
        MOV     @R1,A
        INC     R1
        MOV     A,@R1
        ADDC    A,#00H
        MOV     @R1,A
        CLR     C
        RET
$EJECT

SB:     CLR     C
        MOV     A,@R1
        SUBB    A,@R0
        MOV     @R1,A

        INC     R1
        INC     R0
        MOV     A,@R1
        SUBB    A,@R0
        MOV     @R1,A

        INC     R1
        INC     R0
        MOV     A,@R1
        SUBB    A,@R0
        MOV     @R1,A
        INC     R1
        MOV     A,@R1
        SUBB    A,#00H
        MOV     @R1,A
        CLR     C
        RET
$EJECT

AD1:    CLR     C
        MOV     A,@R1
        ADD     A,@R0
        MOV     @R1,A
        INC     R1
        INC     R0
        MOV     A,@R1
        ADDC    A,@R0
        MOV     @R1,A
        INC     R1
        MOV     A,@R1
        ADDC    A,#00H
        MOV     @R1,A
        CLR     C
        RET
$EJECT

SB1:    CLR     C
        MOV     A,@R1
        SUBB    A,@R0
        MOV     @R1,A
        INC     R1
        INC     R0
        MOV     A,@R1
        SUBB    A,@R0
        MOV     @R1,A
        INC     R1
        MOV     A,@R1
```

```
        SUBB    A,#00H
        MOV     @R1,A
        CLR     C
        RET
$EJECT

WR_Q2334:
        PUSH    AR0             ;want to exit with R0 and R1 unchanged
        PUSH    AR1
        MOV     A,R1
        MOVX    @R0,A
        POP     AR1
        POP     AR0             ;recover R0 AND R1
        RET
$EJECT

INIT_Q2334:
        PUSH    AR0
        PUSH    AR1
        MOV     R0,#00
        MOV     R1,#00
ID1:    LCALL   WR_Q2334        ;fill #1 frequency registers with 0
        INC     R0
        CJNE    R0,#08H,ID1

        MOV     R0,#10H
        MOV     R1,#00H
ID2:    LCALL   WR_Q2334        ;fill #2 frequency registers with 0
        INC     R0
        CJNE    R0,#18H,ID2

        MOV     R1,#00H
        MOV     R0,#08H         ;clear #1, mode_ctrl1 (SMC)
        LCALL   WR_Q2334
        MOV     R1,#00H
        MOV     R0,#18H         ;clear #2, mode_ctrl2  (SMC)
        LCALL   WR_Q2334

        MOV     R1,#00H
        MOV     R0,#0AH
        LCALL   WR_Q2334        ;clear #1 AMC
        MOV     R1,#00H
        MOV     R0,#1AH
        LCALL   WR_Q2334        ;clear #2 AMC

        MOV     R1,#00H
        MOV     R0,#0CH         ;clear #1, accumulator
        LCALL   WR_Q2334
        MOV     R1,#00H
        MOV     R0,#1CH         ;clear #2, accumulator
        LCALL   WR_Q2334

        MOV     R1,#1FH
        MOV     R0,#0EH         ;updata
        LCALL   WR_Q2334
        MOV     R0,#1EH         ;updata
        LCALL   WR_Q2334

        POP     AR1
        POP     AR0
        RET
$EJECT
```

```
END                    ;end of the program
```