CHAPTER 7

SPREAD-SPECTRUM MODULATION

This chapter introduces a modulation technique called spread-spectrum modulation, which is radically different from the modulation techniques that are covered in preceding chapters. In spread-spectrum modulation, channel bandwidth and transmit power are sacrificed for the sake of secure communications.

Specifically, we cover the following topics:

- *Spreading sequences in the form of pseudo-noise sequences, their properties, and methods of generation.*
- *The basic notion of spread-spectrum modulation.*
- *The two commonly used types of spread-spectrum modulation: direct sequence and frequency hopping.*

The material presented in this chapter is basic to wireless communications using code-division multiple access, which is covered in Chapter 8.

7.1 *Introduction*

A major issue of concern in the study of digital communications as considered in Chapters 4, 5, and 6 is that of providing for the efficient use of bandwidth and power. Notwithstanding the importance of these two primary communication resources, there are situations where it is necessary to sacrifice this efficiency in order to meet certain other design objectives. For example, the system may be required to provide a form of ***secure*** communication in a ***hostile*** environment such that the transmitted signal is not easily detected or recognized by unwanted listeners. This requirement is catered to by a class of signaling techniques known collectively as ***spread-spectrum modulation.***

The primary advantage of a spread-spectrum communication system is its ability to reject ***interference*** whether it be the ***unintentional*** interference by another user simultaneously attempting to transmit through the channel, or the ***intentional*** interference by a hostile transmitter attempting to jam the transmission.

The definition of spread-spectrum modulation[1] may be stated in two parts:

1. Spread spectrum is a means of transmission in which the data sequence occupies a bandwidth in excess of the minimum bandwidth necessary to send it.
2. The spectrum spreading is accomplished before transmission through the use of a code that is independent of the data sequence. The same code is used in the receiver

(operating in synchronism with the transmitter) to despread the received signal so that the original data sequence may be recovered.

Although standard modulation techniques such as frequency modulation and pulse-code modulation do satisfy part 1 of this definition, they are not spread-spectrum techniques because they do not satisfy part 2 of the definition.

Spread-spectrum modulation was originally developed for military applications, where resistance to jamming (interference) is of major concern. However, there are civilian applications that also benefit from the unique characteristics of spread-spectrum modulation. For example, it can be used to provide ***multipath rejection*** in a ground-based mobile radio environment. Yet another application is in ***multiple-access*** communications in which a number of independent users are required to share a common channel without an external synchronizing mechanism; here, for example, we may mention a ground-based radio environment involving mobile vehicles that must communicate with a central station. More is said about this latter application in Chapter 8.

In this chapter, we discuss principles of spread-spectrum modulation, with emphasis on direct-sequence and frequency-hopping techniques. In a ***direct-sequence spread-spectrum*** technique, two stages of modulation are used. First, the incoming data sequence is used to modulate a wideband code. This code transforms the narrowband data sequence into a noiselike wideband signal. The resulting wideband signal undergoes a second modulation using a phase-shift keying technique. In a ***frequency-hop spread-spectrum*** technique, on the other hand, the spectrum of a data-modulated carrier is widened by changing the carrier frequency in a pseudo-random manner. For their operation, both of these techniques rely on the availability of a noiselike spreading code called a ***pseudo-random*** or ***pseudo-noise sequence***. Since such a sequence is basic to the operation of spread-spectrum modulation, it is logical that we begin our study by describing the generation and properties of pseudo-noise sequences.

7.2 Pseudo-Noise Sequences

A ***pseudo-noise (PN) sequence*** is a periodic binary sequence with a noiselike waveform that is usually generated by means of a ***feedback shift register***, a general block diagram of which is shown in Figure 7.1. A feedback shift register consists of an ordinary ***shift register*** made up of m flip-flops (two-state memory stages) and a ***logic circuit*** that are interconnected to form a multiloop ***feedback*** circuit. The flip-flops in the shift register are regulated by a single timing ***clock***. At each pulse (tick) of the clock, the ***state*** of each flip-flop is shifted to the next one down the line. With each clock pulse the logic circuit computes a

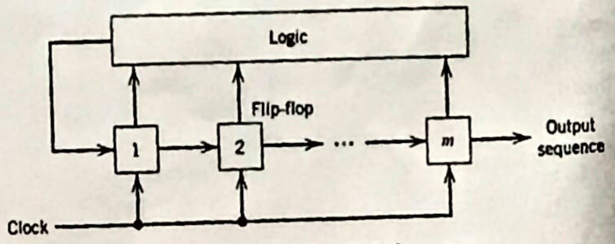


FIGURE 7.1 Feedback shift register.

Boolean function of the states of the flip-flops. The result is then fed back as the input to the first flip-flop, thereby preventing the shift register from emptying. The PN sequence so generated is determined by the length m of the shift register, its initial state, and the feedback logic.

Let $s_j(k)$ denote the state of the jth flip-flop after the kth clock pulse; this state may be represented by symbol 0 or 1. The state of the shift register after the kth clock pulse is then defined by the set $\{s_1(k), s_2(k), \ldots, s_m(k)\}$, where $k \geq 0$. For the initial state, k is zero. From the definition of a shift register, we have

$$s_j(k+1) = s_{j-1}(k), \qquad \begin{cases} k \geq 0 \\ 1 \leq j \leq m \end{cases} \tag{7.1}$$

where $s_0(k)$ is the input applied to the first flip-flop after the kth clock pulse. According to the configuration described in Figure 7.1, $s_0(k)$ is a Boolean function of the individual states $s_1(k), s_2(k), \ldots, s_m(k)$. For a specified length m, this Boolean function uniquely determines the subsequent sequence of states and therefore the PN sequence produced at the output of the final flip-flop in the shift register. With a total number of m flip-flops, the number of possible states of the shift register is at most 2^m. It follows therefore that the PN sequence generated by a feedback shift register must eventually become *periodic* with a period of at most 2^m.

A feedback shift register is said to be *linear* when the feedback logic consists entirely of *modulo-2 adders*. In such a case, the *zero state* (e.g., the state for which all the flip-flops are in state 0) is *not* permitted. We say so because for a zero state, the input $s_0(k)$ produced by the feedback logic would be 0, the shift register would then continue to remain in the zero state, and the output would therefore consist entirely of 0s. Consequently, the period of a PN sequence produced by a linear feedback shift register with m flip-flops cannot exceed $2^m - 1$. When the period is exactly $2^m - 1$, the PN sequence is called a *maximal-length-sequence* or simply *m-sequence*.

▶ **EXAMPLE 7.1**

Consider the linear feedback shift register shown in Figure 7.2, involving three flip-flops. The input s_0 applied to the first flip-flop is equal to the modulo-2 sum of s_1 and s_3. It is assumed that the initial state of the shift register is 100 (reading the contents of the three flip-flops from left to right). Then, the succession of states will be as follows:

100, 110, 111, 011, 101, 010, 001, 100, ...

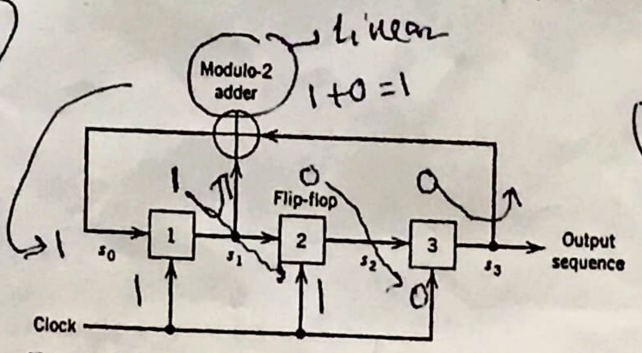
$s_3 + s_1 \rightarrow$ modulo-2 adder

[3, 1]

must be three

$\rightarrow$ circular

3



$\boxed{100 \rightarrow 110}$

$000 \rightarrow 000 \rightarrow 000$

$\boxed{2^m - 1}$

m-sequence

maximal length-sequ

FIGURE 7.2 Maximal-length sequence generator for $m = 3$.

$S_j(k+1) = S_{j-1}(k)$

The output sequence (the last position of each state of the shift register) is therefore

$$00111010\ldots$$

which repeats itself with period $2^3 - 1 = 7$.

Note that the choice of 100 as the initial state is arbitrary. Any of the other six permissible states could serve equally well as an initial state. The resulting output sequence would then simply experience a cyclic shift.

■ PROPERTIES OF MAXIMAL-LENGTH SEQUENCES[2]

M-Sequences. M-sequences, or maximal length sequences, are pseudonoise sequences generated by LFSR that have maximum period. That is, any sequence that is generated by an n-bit LFSR and has period $2^n - 1$ is an m-sequence. An m sequence has several desirable properties. It has an almost uniform distribution of 0s and 1s. Specifically, in any window of $2^n - 1$ bits, it has 2^{n-1} ones and $2^{n-1} - 1$ zeros. Moreover, in any window of $2^n - 1$ bits, it has one run of ones of length n, one run of zeros of length $n - 1$, and in general, 2^{n-i-2} runs of ones and 2^{n-i-2} runs of zeros of length i, $1 \leq i \leq n - 2$.

balanced properties

run properties

0 0 1 1 1 0 1 0 $1 \to 4 - 2^{3-1}$ $n = 3$

$0 \to 3 \to 2^{3} - 1 = 3$

For spread spectrum applications, one important measure of the "randomness" of a pseudonoise sequence is its autocorrelation function, which gives the correlation of a period of the sequence with its cyclic shifts. The smaller the correlation, the easier is it for the sender-receiver pair to synchronize; furthermore, interference effects are also somewhat alleviated. For a sequence $B_1, \ldots, B_N$ of ± 1, the periodic autocorrelation function R is given by

# run of 1s of length 3 = 1

# run of 0s of length $\boxed{3}$ - \# 1

✓

$$R(\tau) = \frac{1}{N} \sum_{k=1}^{N} B_k B_{k-\tau}.$$

It can be shown that the periodic autocorrelation of an m-sequence is

$$R(\tau) = \begin{cases} 1 & \tau = 0, N, 2N, \ldots \\ -\frac{1}{N} & \text{otherwise} \end{cases}$$

In spread spectrum applications, when the codes for different users are generated from different sources, then a similar measure of interest is the cross-correlation of the two codes. The cross-correlation function of two sequences $A_1, \ldots, A_n$ and $B_1, \ldots, B_N$ of ± 1s is defined as

$$R_{A,B}(\tau) = \frac{1}{N} \sum_{k=1}^{N} A_k B_{k-\tau}.$$

It is intuitive that we would like the cross-correlation function of two user spread spectrum codes to be low since that would allow a receiver to discriminate among spread spectrum signals generated by different m-sequences.

7.3 A Notion of Spread Spectrum

An important attribute of spread-spectrum modulation is that it can provide protection against externally generated interfering (jamming) signals with finite power. The jamming signal may consist of a fairly powerful broadband noise or multitone waveform that is directed at the receiver for the purpose of disrupting communications. Protection against jamming waveforms is provided by purposely making the information-bearing signal occupy a bandwidth far in excess of the minimum bandwidth necessary to transmit it. This has the effect of making the transmitted signal assume a noiselike appearance so as to blend into the background. The transmitted signal is thus enabled to propagate through the channel undetected by anyone who may be listening. We may therefore think of spread spectrum as a method of "camouflaging" the information-bearing signal.

One method of widening the bandwidth of an information-bearing (data) sequence involves the use of *modulation*. Let $\{b_k\}$ denote a binary data sequence, and $\{c_k\}$ denote a pseudo-noise (PN) sequence. Let the waveforms $b(t)$ and $c(t)$ denote their respective polar nonreturn-to-zero representations in terms of two levels equal in amplitude and opposite in polarity, namely, ± 1. We will refer to $b(t)$ as the information-bearing (data) signal, and to $c(t)$ as the PN signal. The desired modulation is achieved by applying the data signal $b(t)$ and the PN signal $c(t)$ to a product modulator or multiplier, as in Figure 7.5*a*. We know from Fourier transform theory that multiplication of two signals produces a signal whose spectrum equals the convolution of the spectra of the two component signals. Thus, if the message signal $b(t)$ is narrowband and the PN signal $c(t)$ is wideband, *the product (modulated) signal $m(t)$ will have a spectrum that is nearly the same as the wideband PN signal*. In other words, in the context of our present application, the PN sequence performs the role of a *spreading code*.

By multiplying the information-bearing signal $b(t)$ by the PN signal $c(t)$, each information bit is "chopped" up into a number of small time increments, as illustrated in the waveforms of Figure 7.6. These small time increments are commonly referred to as *chips*.

For *baseband* transmission, the product signal $m(t)$ represents the *transmitted signal*. We may thus express the transmitted signal as

$$m(t) = c(t)b(t) \tag{7.7}$$

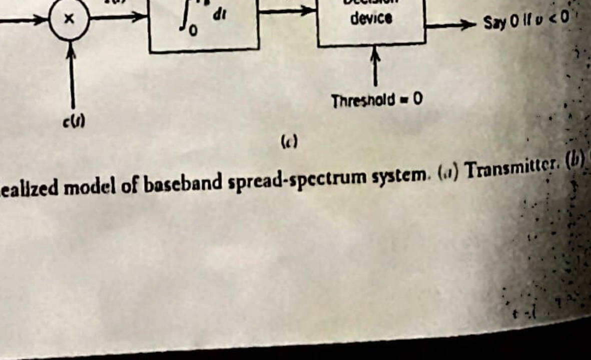


FIGURE 7.5 Idealized model of baseband spread-spectrum system. (*a*) Transmitter. (*b*) Channel. (*c*) Receiver.

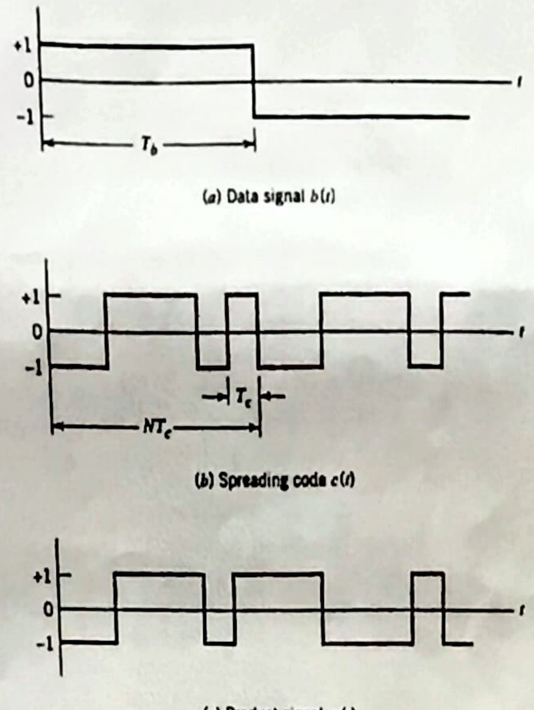


FIGURE 7.6 Illustrating the waveforms in the transmitter of Figure 7.5*a*.

The received signal $r(t)$ consists of the transmitted signal $m(t)$ plus an additive *interference* denoted by $i(t)$, as shown in the channel model of Figure 7.5*b*. Hence,

$$\begin{aligned} r(t) &= m(t) + i(t) \\ &= c(t)b(t) + i(t) \end{aligned} \tag{7.8}$$

To recover the original message signal $b(t)$, the received signal $r(t)$ is applied to a *demodulator* that consists of a multiplier followed by an integrator, and a decision device, as in Figure 7.5*c*. The multiplier is supplied with a locally generated PN sequence that is an exact *replica* of that used in the transmitter. Moreover, we assume that the receiver operates in perfect *synchronism* with the transmitter, which means that the PN sequence in the receiver is lined up exactly with that in the transmitter. The multiplier output in the receiver is therefore given by

$$\begin{aligned} z(t) &= c(t)r(t) \\ &= c^2(t)b(t) + c(t)i(t) \end{aligned} \tag{7.9}$$

Equation (7.9) shows that the data signal $b(t)$ is multiplied *twice* by the PN signal $c(t)$, whereas the unwanted signal $i(t)$ is multiplied only *once*. The PN signal $c(t)$ alternates between the levels -1 and $+1$, and the alternation is destroyed when it is squared; hence,

$$c^2(t) = 1 \qquad \text{for all } t \tag{7.10}$$

Accordingly, we may simplify Equation (7.9) as

$$z(t) = b(t) + c(t)i(t) \tag{7.11}$$

We thus see from Equation (7.11) that the data signal $b(t)$ is reproduced at the multiplier output in the receiver, except for the effect of the interference represented by the additive term $c(t)i(t)$. Multiplication of the interference $i(t)$ by the locally generated PN signal $c(t)$ means that the spreading code will affect the interference just as it did the original signal at the transmitter. We now observe that the data component $b(t)$ is narrowband, whereas the spurious component $c(t)i(t)$ is wideband. Hence, by applying the multiplier output to a baseband (low-pass) filter with a bandwidth just large enough to accommodate the recovery of the data signal $b(t)$, most of the power in the spurious component $c(t)i(t)$ is filtered out. The effect of the interference $i(t)$ is thus significantly reduced at the receiver output.

In the receiver shown in Figure 7.5*c*, the low-pass filtering action is actually performed by the integrator that evaluates the area under the signal produced at the multiplier output. The integration is carried out for the bit interval $0 \leq t \leq T_b$, providing the sample value v. Finally, a decision is made by the receiver: If v is greater than the threshold of zero, the receiver says that binary symbol 1 of the original data sequence was sent in the interval $0 \leq t \leq T_b$, and if v is less than zero, the receiver says that symbol 0 was sent; if v is exactly zero the receiver makes a random guess in favor of 1 or 0.

In summary, the use of a spreading code (with pseudo-random properties) in the transmitter produces a wideband transmitted signal that appears *noiselike* to a receiver that has *no* knowledge of the spreading code. From the discussion presented in Section 7.2, we recall that (for a prescribed data rate) the longer we make the period of the spreading code, the closer will the transmitted signal be to a truly random binary wave, and the harder it is to detect. Naturally, the price we have to pay for the improved protection against interference is increased transmission bandwidth, system complexity, and processing delay. However, when our primary concern is the security of transmission, these are not unreasonable costs to pay.