one of M frequencies is to be transmitted. The position of the M-ary signal set is shifted pseudorandomly by the frequency synthesizer over a hopping bandwidth W_{ss}. A typical FH/MFSK system block diagram is shown in Figure 12.11. In a conventional MFSK system, the data symbol modulates a *fixed frequency* carrier; in an FH/MFSK system, the data symbol modulates a carrier whose frequency is *pseudorandomly* determined. In either case, a single tone is transmitted. The FH system in Figure 12.11 can be thought of as a two-step modulation process—data modulation and frequency-hopping modulation—even though it can be implemented as a single step whereby the frequency synthesizer produces a transmission tone based on the simultaneous dictates of the PN code and the data. At each frequency hop time, a PN generator feeds the frequency synthesizer a frequency word (a sequence of ℓ chips), which dictates one of 2^{ℓ} symbol-set positions. The frequency-hopping bandwidth W_{ss}, and the minimum frequency spacing between consecutive hop positions Δf, dictate the minimum number of chips necessary in the frequency word.

For a given hop, the occupied transmission bandwidth is identical to the bandwidth of conventional MFSK, which is typically much smaller than W_{ss}. However, averaged over many hops, the FH/MFSK spectrum occupies the entire spread-spectrum bandwidth. Spread-spectrum technology permits FH bandwidths of the order of several gigahertz, which is an order of magnitude larger than implementable DS bandwidths [8], thus allowing for larger processing gains in FH compared to DS systems. Since frequency hopping techniques operate over such wide bandwidths, it is difficult to maintain phase coherence from hop to hop. Therefore, such schemes are usually configured using noncoherent demodulation. Nevertheless, consideration has been given to coherent FH in Reference [9].

In Figure 12.11 we see that the receiver reverses the signal processing steps of the transmitter. The received signal is first FH demodulated (dehopped) by mixing it with the same sequence of pseudorandomly selected frequency tones that was used for hopping. Then the dehopped signal is applied to a conventional bank of M noncoherent energy detectors to select the most likely symbol.

Example 12.1 Frequency Word Size

A hopping bandwidth W_{ss} of 400 MHz and a frequency step size Δf of 100 Hz are specified. What is the minimum number of PN chips that are required for each frequency word?

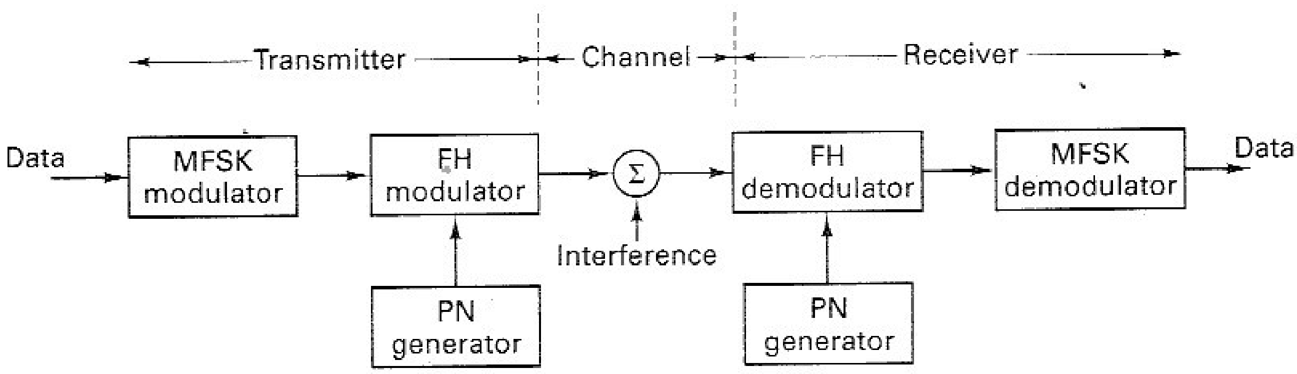


Figure 12.11 FH/MFSK system.

Solution

$$\text{Number of tones contained in } W_{ss} = \frac{W_{ss}}{\Delta f} = \frac{400 \text{ MHz}}{100 \text{ Hz}}$$

$$= 4 \times 10^6$$

$$\text{Minimum number of chips} = \lceil \log_2 (4 \times 10^6) \rceil$$

$$= 22 \text{ chips}$$

where $\lceil x \rceil$ indicates the smallest integer value not less than x.

12.4.1 Frequency Hopping Example

Consider the frequency hopping example illustrated in Figure 12.12. The input data consist of a binary sequence with a data rate of $R = 150$ bits/s. The modulation is 8-ary FSK. Therefore, the symbol rate is $R_s = R/(\log_2 8) = 50$ symbols/s (the symbol duration $T = 1/50 = 20$ ms). The frequency is hopped once per symbol, and the hopping is time synchronous with the symbol boundaries. Thus, the hopping rate is 50 hops/s. Figure 12.12 depicts the time–bandwidth plane of the communication resource; the abscissa represents time, and the ordinate represents the hopping bandwidth, W_{ss}. The legend on the right side of the figure illustrates a set of 8-ary FSK symbol-to-tone assignments. Notice that the tone separation specified is $1/T = 50$ Hz, which corresponds to the minimum required tone spacing for the orthogonal signaling of this noncoherent FSK example (see Section 4.5.4).

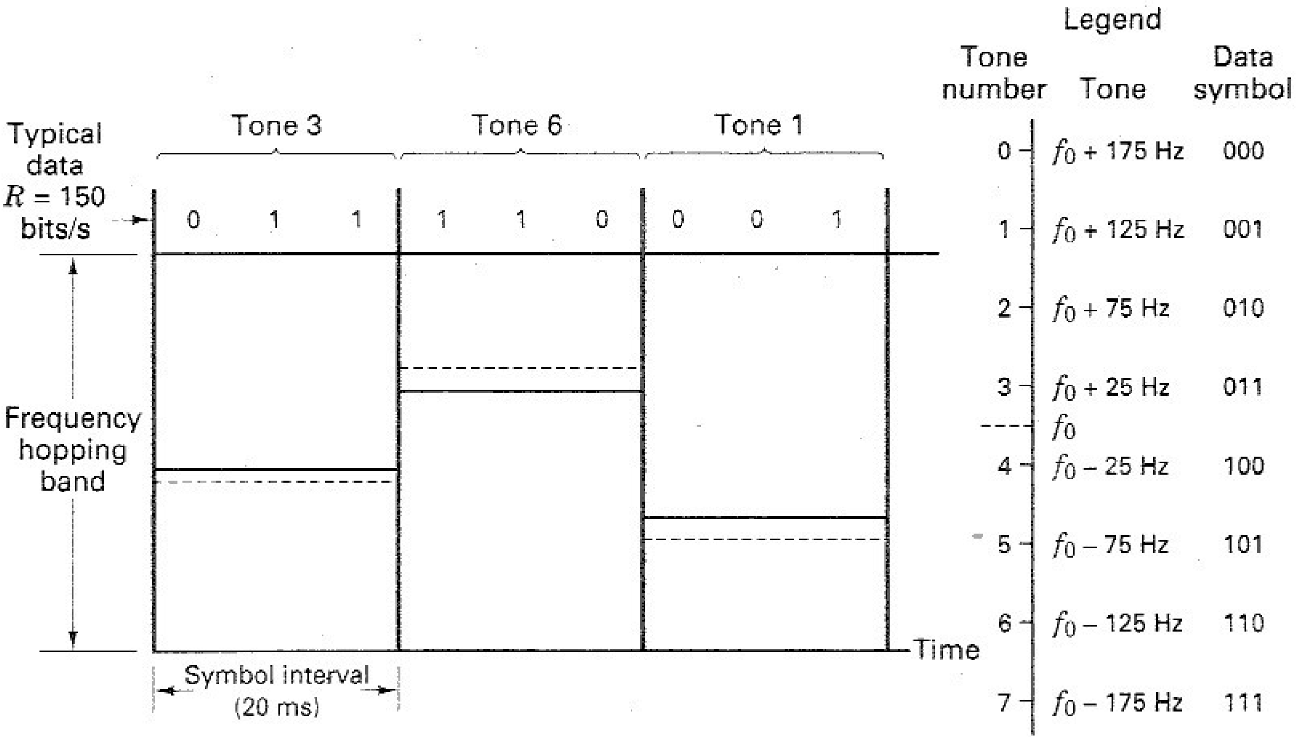


Figure 12.12 Frequency-hopping example using 8-ary FSK modulation.

A typical binary data sequence is shown at the top of Figure 12.12. Since the modulation is 8-ary FSK, the bits are grouped three at a time to form symbols. In a *conventional* 8-ary FSK scheme, a single-sideband tone (offset from f_0, the *fixed* center frequency of the data band), would be transmitted according to an assignment like the one shown in the legend. The only difference in this FH/MFSK example is that the center frequency of the data band f_0 is *not fixed.* For each new symbol, f_0 hops to a new position in the hop bandwidth, and the entire data-band structure moves with it. In the example of Figure 12.12, the first symbol in the data sequence, 0 1 1, yields a tone 25 Hz above f_0. The diagram depicts f_0 with a dashed line and the symbol tone with a solid line. During the second symbol interval, f_0 has hopped to a new spectral location, as indicated by the dashed line. The second symbol, 1 1 0, dictates that a tone indicated by the solid line, 125 Hz below f_0, shall be transmitted. Similarly, the final symbol in this example, 0 0 1, calls for a tone 125 Hz above f_0. Again, the center frequency has moved, but the relative positions of the symbol tones remain fixed.

12.4.2 Robustness

A common dictionary definition describes the term *robustness* as the state of being strong and healthy; full of vigor; hardy. In the context of communications, the usage is not too different. Robustness characterizes a signal's ability to withstand impairments from the channel, such as noise, jamming, fading, and so on. A signal configured with multiple replicate copies, each transmitted on a different frequency, has a greater likelihood of survival than does a single such signal with equal total power. The greater the diversity (multiple transmissions, at different frequencies, spread in time), the more robust the signal against random interference.

The following example should clarify the concept. Consider a message consisting of four symbols: s_1, s_2, s_3, s_4. The introduction of diversity starts by repeating the message N times. Let us choose $N = 8$. Then, the repeated symbols, called *chips*, can be written.

$$s_1\,s_1\,s_1\,s_1\,s_1\,s_1\,s_1\,s_1\,s_2\,s_2\,s_2\,s_2\,s_2\,s_2\,s_2\,s_2\,s_3\,s_3\,s_3\,s_3\,s_3\,s_3\,s_3\,s_3\,s_4\,s_4\,s_4\,s_4\,s_4\,s_4\,s_4\,s_4$$

Each chip is transmitted at a different hopping frequency (the center of the data bandwidth is changed for each chip). The resulting transmissions at frequencies f_i, f_j, f_k, . . . yield a more robust signal than without such diversity. A target-shooting analogy is that a pellet from a barrage of shotgun pellets has a better chance of hitting a target, compared with the action of a single bullet.

12.4.3 Frequency Hopping with Diversity

In Figure 12.13 we extend the example illustrated in Figure 12.12, with the additional feature of a chip repeat factor of $N = 4$. During each 20-ms symbol interval, there are now four columns, corresponding to the four separate chips to be transmitted for each symbol. At the top of the figure we see the same data sequence, with $R = 150$ bps, as in the earlier example; and we see the same 3-bit partitioning to form the 8-ary symbols. Each symbol is transmitted four times, and for each transmission the center frequency of the data band is hopped to a new region of the hopping band, under the control of a PN code generator. Therefore, for this example, each chip interval, T_c, is equal to $T/N = 20$ ms/4 = 5 ms in duration, and the hopping rate is now

$$\frac{NR}{\log_2 8} = 200 \text{ hops/s}$$

Notice that the spacing between frequency tones must change to meet the changed requirement for orthogonality. Since the duration of each FSK tone is now equal to the chip duration, that is, $T_c = T/N$, the minimum separation between tones is $1/T_c = N/T = 200$ Hz. As in the earlier example, Figure 12.13 illustrates that the center of the data band (plus the modulation structure) is shifted at each new chip time. The position of the solid line (transmission frequency) has the same relationship to the dashed line (center of the data band) for each of the chips associated with a given symbol.

12.4.4 Fast Hopping versus Slow Hopping

In the case of direct-sequence spread-spectrum systems, the term "chip" refers to the PN code symbol (the symbol of shortest duration in a DS system). In a similar sense for frequency hopping systems, the term "chip" is used to characterize the

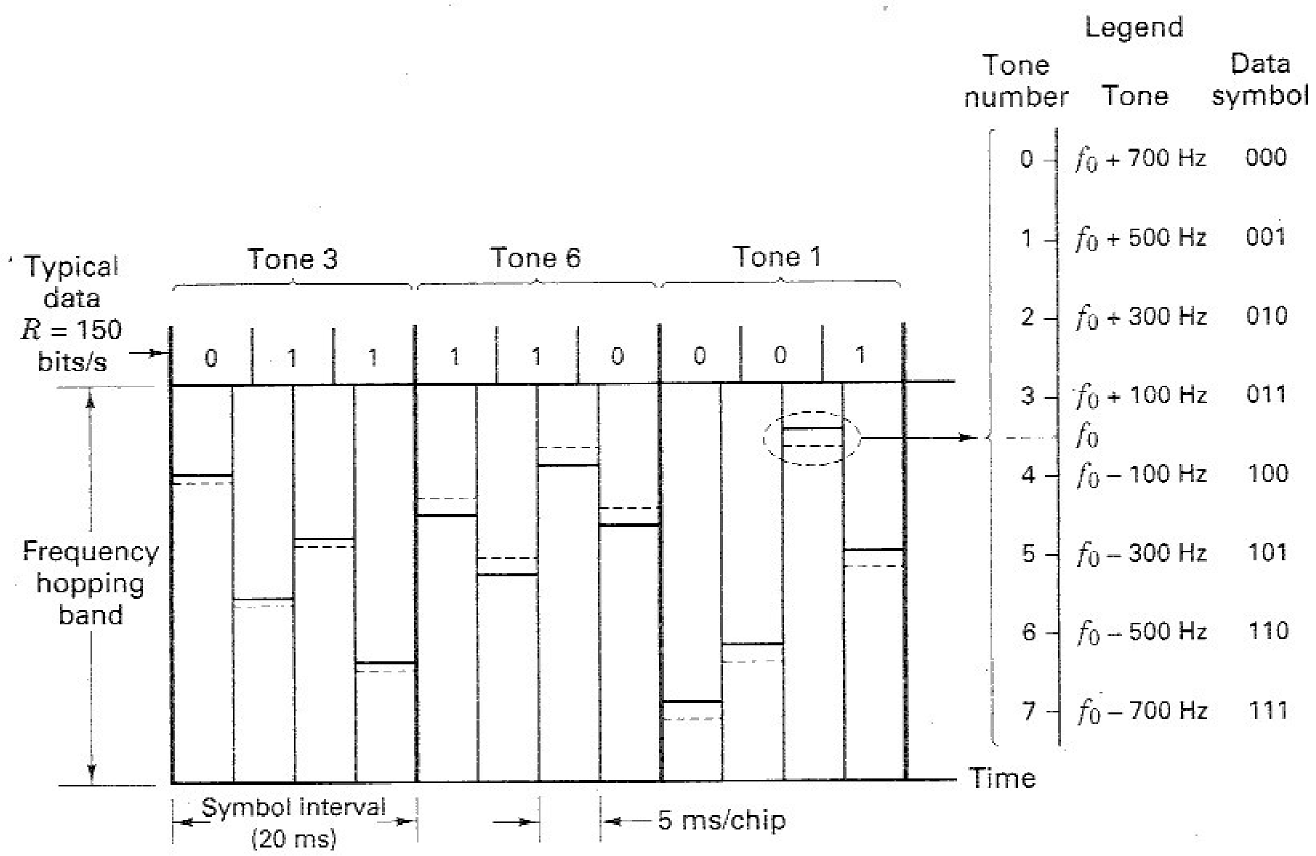


Figure 12.13 Frequency hopping example with diversity ($N = 4$).

shortest uninterrupted waveform in the system. Frequency hopping systems are classified as *slow-frequency hopping* (SFH), which means there are several modulation symbols per hop, or as *fast-frequency hopping* (FFH), which means that there are several frequency hops per modulation symbol. For SFH, the shortest uninterrupted waveform in the system is that of the data symbol; however, for FFH, the shortest uninterrupted waveform is that of the hop. Figure 12.14a illustrates an example of FFH; the data symbol rate is 30 symbols/s and the frequency hopping rate is 60 hops/s. The figure illustrates the waveform $s(t)$ over one symbol duration ($\frac{1}{30}$ s). The waveform change in (the middle of) $s(t)$ is due to a new frequency hop. In this example, a chip corresponds to a hop since the hop duration is shorter than the symbol duration. Each chip corresponds to half a symbol. Figure 14.14b illustrates an example of SFH; the data symbol rate is still 30 symbols/s, but the frequency hopping rate has been reduced to 10 hops/s. The waveform $s(t)$ is shown over a duration of three symbols ($\frac{1}{10}$ s). In this example, the hopping boundaries appear only at the beginning and end of the three-symbol duration. Here, the changes in the waveform are due to the modulation state changes; therefore, in this

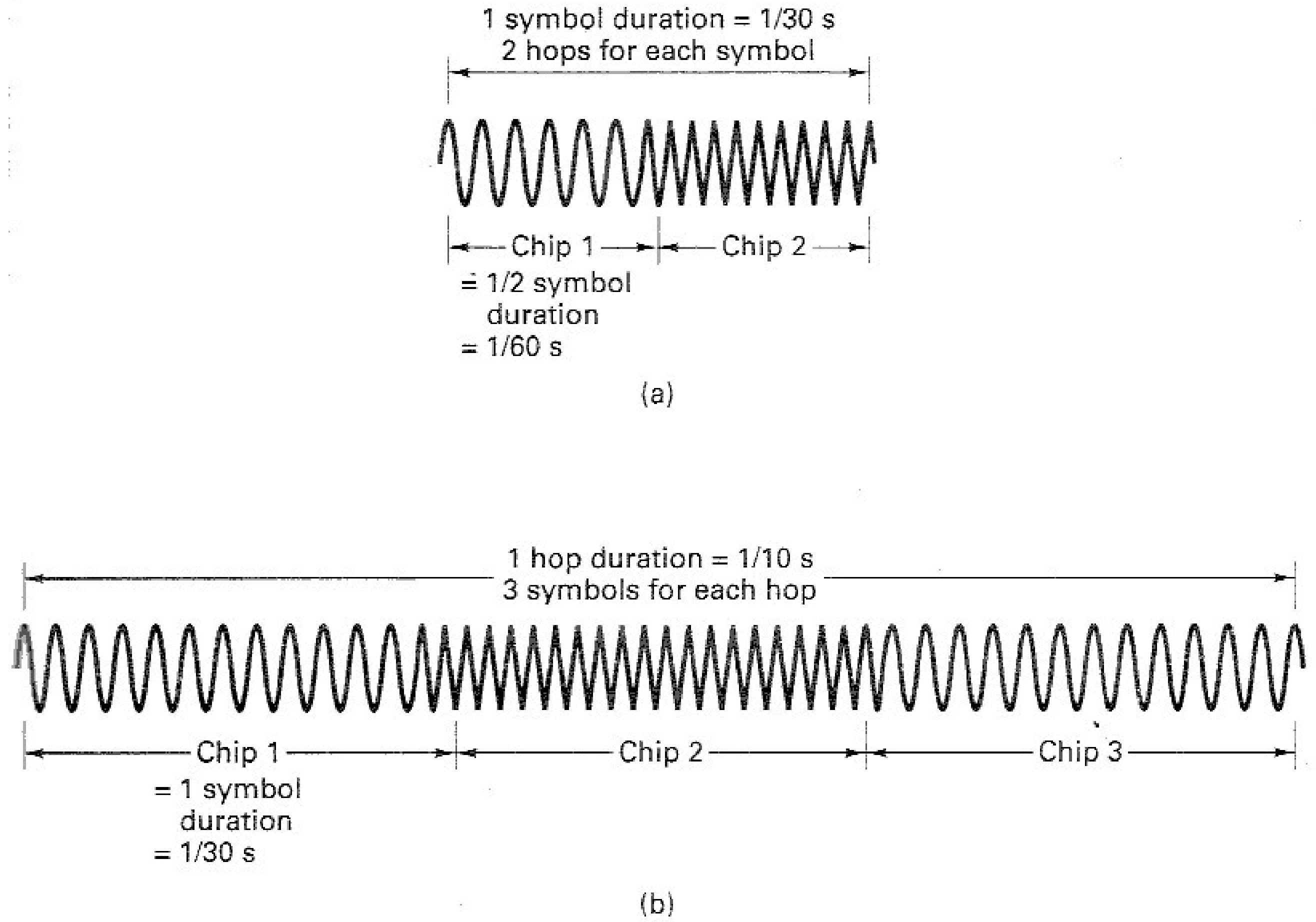


Figure 12.14 Chip—in the context of an FH/MFSK system. (a) Example 1: Frequency hopping MFSK system with symbol rate = 30 symbols/s and hopping rate = 60 hops/s. 1 chip = 1 hop. (b) Example 2: Same as part (a) except hopping rate = 10 hops/s. 1 chip = 1 symbol.

example a chip corresponds to a data symbol, since the data symbol is shorter than the hop duration.

Figure 12.15a illustrates an FFH example of a binary FSK system. The diversity is $N = 4$. There are 4 chips transmitted per bit. As in Figure 12.13, the dashed line in each column corresponds to the center of the data band and the solid line corresponds to the symbol frequency. Here, for FFH, the chip duration is the hop duration. Figure 12.15b illustrates an example of an SFH binary FSK system. In this case, there are 3 bits transmitted during the time duration of a single hop. Here, for SFH, the chip duration is the bit duration. If this SFH example were changed from a binary system to an 8-ary system, what would the chip duration then correspond to? If the system were implemented as an 8-ary scheme, each 3 bits would be transmitted as a single data symbol. The symbol boundaries and the hop boundaries would then be the same, and the chip duration, the hop duration, and the symbol duration would all be the same.

12.4.5 FFH/MFSK Demodulator

Figure 12.16 illustrates the schematic for a typical fast frequency hopping MFSK (FFH/MFSK) demodulator. First, the signal is dehopped using a PN generator identical to the one used for hopping. Then, after filtering with a low-pass filter that

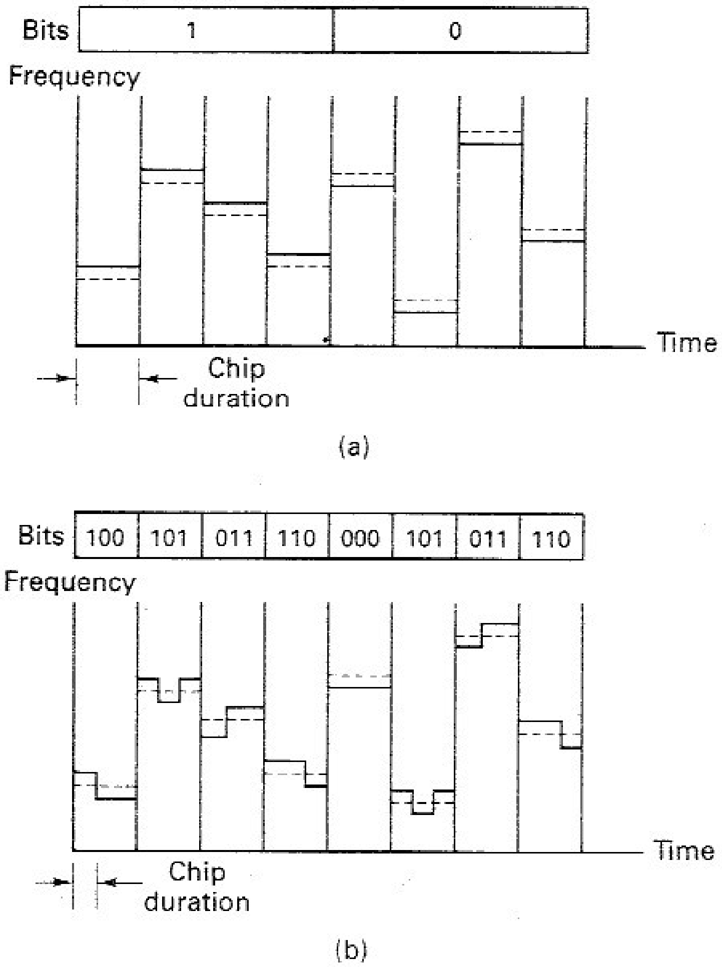


Figure 12.15 Fast hopping versus slow hopping in a binary system. (a) Fast-hopping example: 4 hops/bit. (b) Slow-hopping example: 3 bits/hop.

has a bandwidth equal to the data bandwidth, the signal is demodulated using a bank of M envelope or energy detectors. Each envelope detector is followed by a clipping circuit and an accumulator. The clipping circuit serves an important function in the presence of an intentional jammer or other strong unpredictable interference; it is treated in a later section. The demodulator does *not* make symbol decisions on a chip-by-chip basis. Instead, the energy from the N chips are accumulated, and after the energy from the Nth chip is added to the $N-1$ earlier ones, the demodulator makes a symbol decision by choosing the symbol that corresponds to the accumulator, z_i ($i = 1, 2, \ldots, M$), with maximum energy.

12.4.6 Processing Gain

Equation (12.27) shows the general expression for processing gain as $G_p = W_{ss}/R$. In the case of direct-sequence spread-spectrum, W_{ss} was set equal to the chip rate R_{ch}. In the case of frequency hopping, Equation (12.27) still expresses the processing gain, but here we set W_{ss} equal to the frequency band over which the system may hop. We designate this band as the *hopping band* $W_{hopping}$, and thus the processing gain for frequency hopping systems is written as

$$G_p = \frac{W_{hopping}}{R} \tag{12.29}$$

12.5 SYNCHRONIZATION

For both DS and FH spread-spectrum systems, a receiver must employ a *synchronized* replica of the spreading or code signal to demodulate the received signal successfully. The process of synchronizing the locally generated spreading signal with

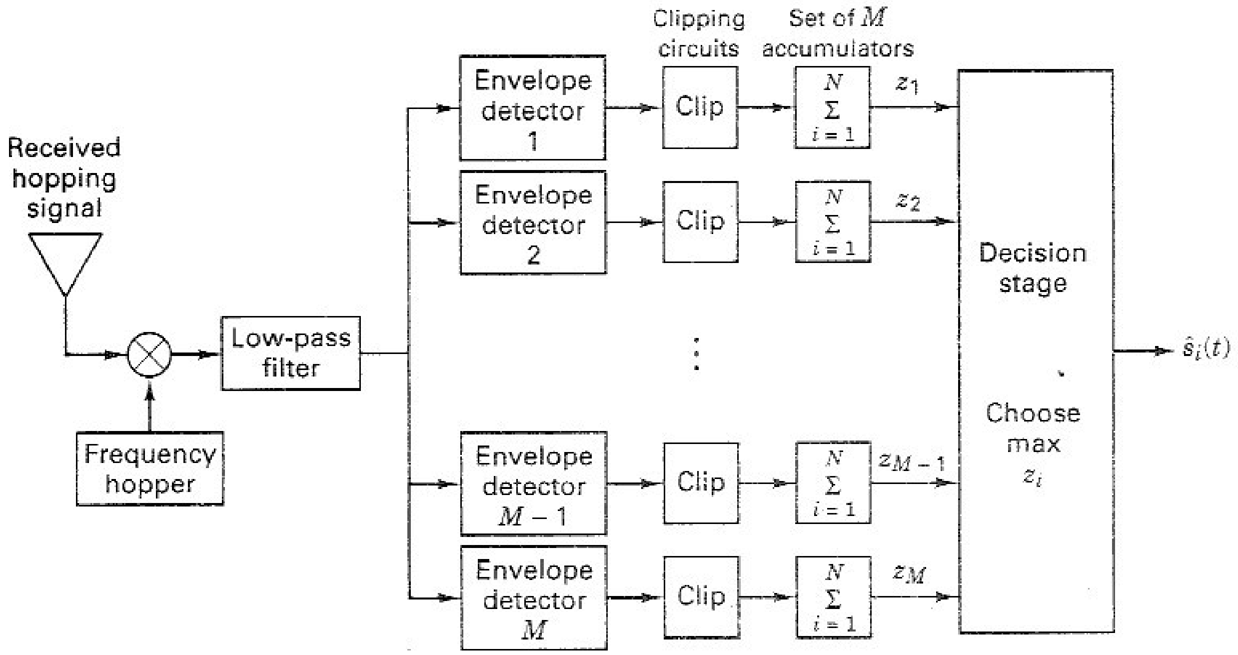


Figure 12.16 FFH/MFSK demodulator.

the received spread-spectrum signal is usually accomplished in two steps. The first step, called *acquisition,* consists of bringing the two spreading signals into *coarse* alignment with one another. Once the received spread-spectrum signal has been acquired, the second step, called *tracking,* takes over and continuously maintains the best possible waveform *fine* alignment by means of a feedback loop.

12.5.1 Acquisition

The acquisition problem is one of searching throughout a region of time and frequency uncertainty in order to synchronize the received spread-spectrum signal with the locally generated spreading signal. Acquisition schemes can be classified as coherent or noncoherent. Since the despreading process typically takes place before carrier synchronization, and therefore the carrier phase is unknown at this point, most acquisition schemes utilize noncoherent detection. When determining the limits of the uncertainty in time and frequency, the following items must be considered:

1. Uncertainty in the distance between the transmitter and the receiver translates into uncertainty in the amount of propagation delay.
2. Relative clock instabilities between the transmitter and the receiver result in phase differences between the transmitter and receiver spreading signals that will tend to grow as a function of elapsed time between synchronization.
3. Uncertainty of the receiver's relative velocity with respect to the transmitter translates into uncertainty in the value of Doppler frequency offset of the incoming signal.
4. Relative oscillator instabilities between the transmitter and the receiver result in frequency offsets between the two signals.

12.5.1.1 Correlator Structures

A common feature of all acquisition methods is that the received signal and the locally generated signal are first correlated to produce a measure of similarity between the two. This measure is then compared with a threshold to decide if the two signals are in synchronism. If they are, the tracking loop takes over.* If they are not, the acquisition procedure provides for a phase or frequency change in the locally generated code as a part of a systematic search through the receiver's phase and frequency uncertainty region, and another correlation is attempted.

Consider the direct-sequence *parallel-search* acquisition system shown in Figure 12.17. The locally generated code $g(t)$ is available with delays that are spaced one-half chip ($T_c/2$) apart. If the time uncertainty between the local code and the received code is N_c chips, and a complete parallel search of the entire time uncertainty region is to be accomplished in a single search time, $2N_c$ correlators are used. Each correlator simultaneously examines a sequence of λ chips, after which the $2N_c$

*Quite often to maintain a small false alarm probability, the threshold crossing must be further verified by a suitable verification algorithm before the tracking loop takes over [4].

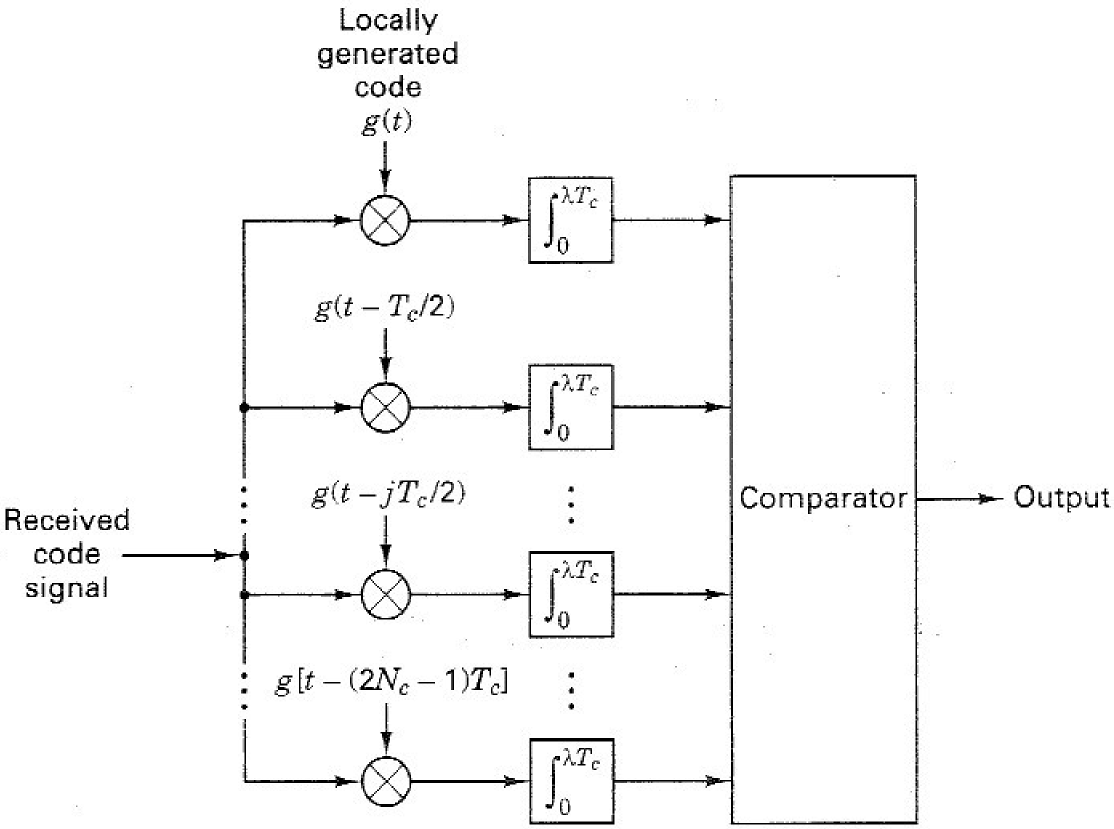


Figure 12.17 Direct-sequence parallel search acquisition.

correlator outputs are compared. The locally generated code, corresponding to the correlator with the largest output is chosen. Conceptually, this is the simplest of the search techniques; it considers all possible code positions (or fractional code positions) in parallel and uses a maximum likelihood algorithm for acquiring the code. Each detector output pertains to the identical observation of received signal plus noise. As λ increases, the synchronization error probability (i.e., the probability of choosing the incorrect code alignment) decreases. Thus, λ is chosen as a compromise between minimizing the probability of a synchronization error and minimizing the time to acquire.

Figure 12.18 illustrates a simple acquisition scheme for a frequency hopping system. Assume that a sequence of N consecutive frequencies from the hop sequence is chosen as a synchronization pattern (without data modulation). The N noncoherent matched filters each consists of a mixer followed by a bandpass filter (BPF) and a square-law envelope detector (an envelope detector followed by a square-law device). If the frequency hopping sequence is $f_1, f_2, \ldots, f_N$, delays are inserted into the matched filters so that when the correct frequency hopping sequence appears, the system produces a large output, indicating detection of the synchronization sequence. Acquisition can be accomplished rapidly because all possible code offsets are examined simultaneously. Note that the presence of bandpass filters (BPF) in Figure 12.18 indicates that the local oscillator frequencies $f_1, f_2, \ldots, f_N$ are chosen to have offsets by some intermediate frequency (IF) from

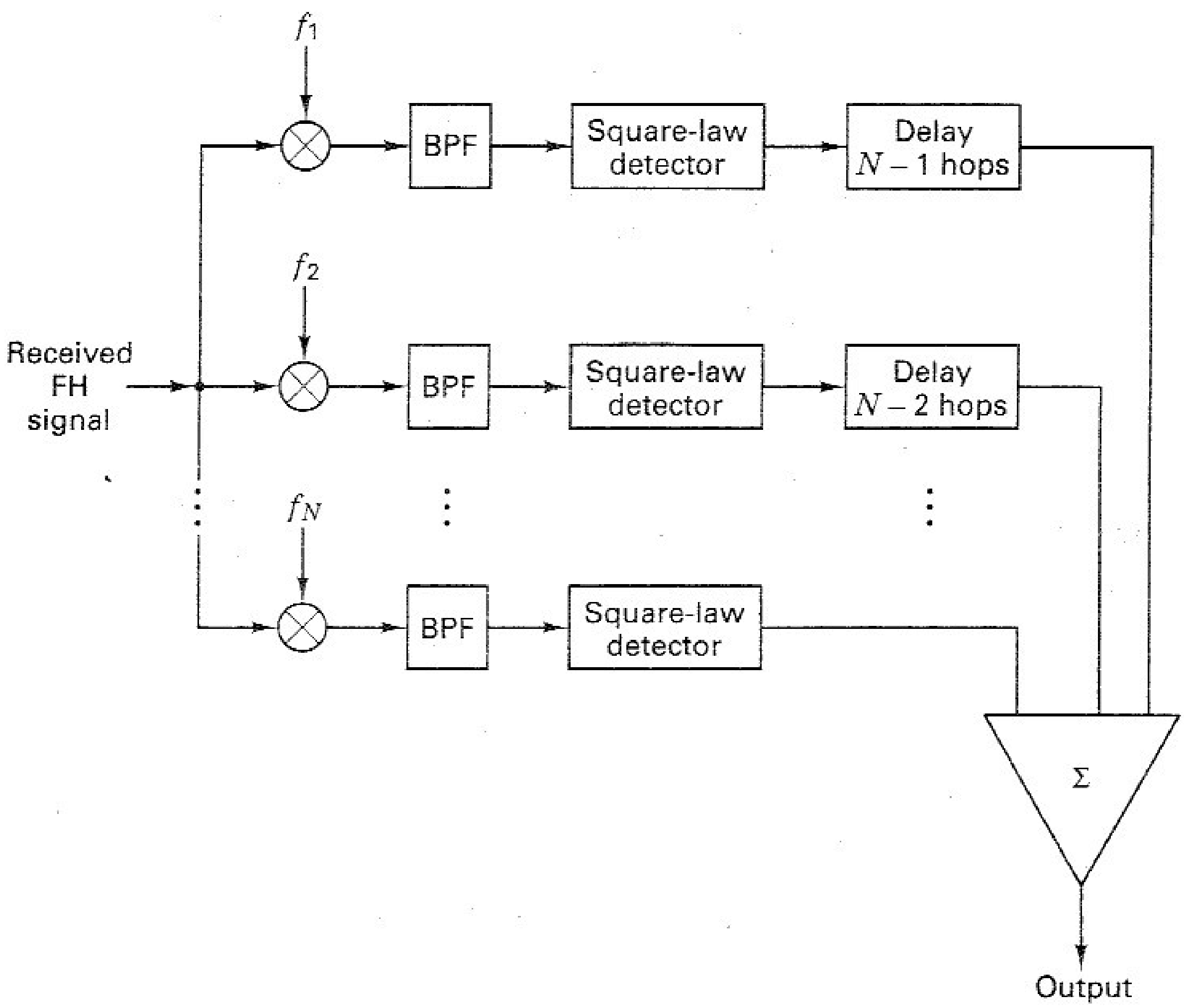


Figure 12.18 Frequency hopping acquisition scheme.

the expected received hop sequence. The same system can be implemented with local oscillator frequencies chosen (without offsets) so that the mixers yield baseband signals, and thus the filters would need to be low-pass filters (LPF). The mixers are typically complex, yielding in-phase and quadrature terms.

If, during each correlation, λ chips (each chip having a duration of T_c) are examined, the maximum time required for a fully parallel search is

$$(T_{\text{acq}})_{\max} = \lambda T_c \tag{12.30}$$

The mean acquisition time of a parallel search system can be approximated by noting that after integrating over λ chips, a correct decision will be made with probability P_D, called the *probability of detection.* If an incorrect output is chosen, an additional λ chips are again examined to make a determination of the correct output. Therefore, on the average, the acquisition time is [4]

$$\begin{aligned}\overline{T}_{\text{acq}} &= \lambda T_c P_D + 2\lambda T_c P_D(1 - P_D) + 3\lambda T_c P_D(1 - P_D)^2 + \cdots \\ &= \frac{\lambda T_c}{P_D}\end{aligned} \tag{12.31}$$

Since the required number of correlators or matched filters can be prohibitively large, fully parallel acquisition techniques are not usually used. In place of Figures 12.17 and 12.18, a single correlator or matched filter can be implemented that will *serially search* until synchronization is achieved. Naturally, trade-offs between fully parallel, fully serial, and combinations of the two involve hardware complexity versus time to acquire for the same uncertainty and chip rate.

12.5.1.2 Serial Search

A popular strategy for the acquisition of spread-spectrum signals is to use a single correlator or matched filter to serially search for the correct phase of the DS code signal or the correct hopping pattern of the FH signal. A considerable reduction in complexity, size, and cost can be achieved by a serial implementation that repeats the correlation procedure for each possible sequence shift. Figures 12.19 and 12.20 illustrate the basic configuration for DS and FH spread-spectrum schemes, respectively. In a stepped serial acquisition scheme for a DS system, the timing epoch of the local PN code is set, and the locally generated PN signal is correlated with the incoming PN signal. At fixed examination intervals of λT_c (search dwell time), where $\lambda >> 1$, the output signal is compared to a preset threshold. If the output is below the threshold, the phase of the locally generated code signal is incremented by a fraction (usually one-half) of a chip and the correlation is reexamined. When the threshold is exceeded, the PN code is assumed to have been acquired, the phase-incrementing process of the local code is inhibited, and the code tracking procedure will be initiated. In a similar scheme for FH systems, shown in Figure 12.20, the PN code generator controls the frequency hopper. Acquisition is accomplished when the local hopping is aligned with that of the received signal.

The maximum time required for a fully serial DS search, assuming that the search proceeds in half-chip increments, is

$$(T_{\text{acq}})_{\max} = 2N_c \lambda T_c \tag{12.32}$$

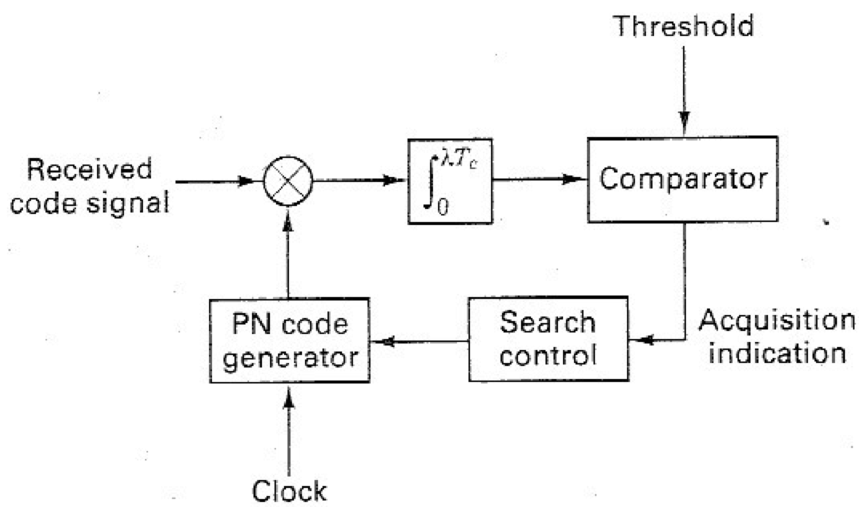


Figure 12.19 Direct-sequence serial search acquisition.

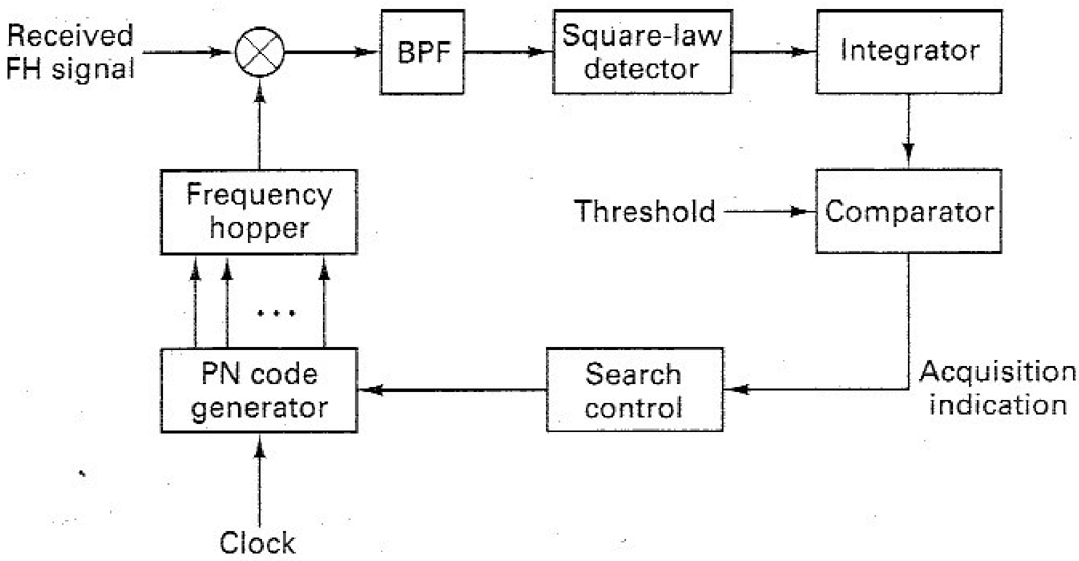


Figure 12.20 Frequency hopping serial search acquisition.

where the uncertainty region to be searched is N_c chips long. The mean acquisition time of a serial DS search system can be shown, for $N_c >> \frac{1}{2}$ chip, to be [10]

$$\overline{T}_{\text{acq}} = \frac{(2 - P_D)(1 + KP_{\text{FA}})}{P_D}(N_c \lambda T_c) \tag{12.33}$$

where λT_c is the search dwell time, P_D is the probability of correct detection, and P_{FA} is the probability of false alarm. We can regard the time interval $K\lambda T_c$, where $K >> 1$, as the time needed to verify a detection. Therefore, in the event of a false alarm, $K\lambda T_c$ seconds is the time penalty incurred. For $N_c >> \frac{1}{2}$ chip and $K << 2N_c$, the variance of the acquisition time is

$$(\text{var})_{\text{acq}} = (2N_c\lambda T_c)^2(1 + KP_{\text{FA}})\left(\frac{1}{12} + \frac{1}{P_D^2} - \frac{1}{P_D}\right) \tag{12.34}$$

12.5.1.3 Sequential Estimation

Another search technique, called *rapid acquisition by sequential estimation* (RASE), proposed by Ward [11], is illustrated in Figure 12.21. The switch is initially in position 1. The RASE system enters its best estimate of the first n received code chips into the n stages of its local PN generator. The fully loaded register defines a starting state from which the generator begins its operation. A PN sequence has the property that the next combination of register states depends only on the present combination of states. Therefore, if the first n received chips are correctly estimated, all the following chips from the local PN generator will be correctly generated. The switch is next thrown to position 2. If the starting state had been correctly estimated, the local generator generates the same sequences as the incoming waveform, in the absence of noise. If the correlator output after λT_c exceeds a preset threshold level, we assume that synchronization has occurred. If the output is less than the threshold, the switch is returned to position 1, the register is reloaded

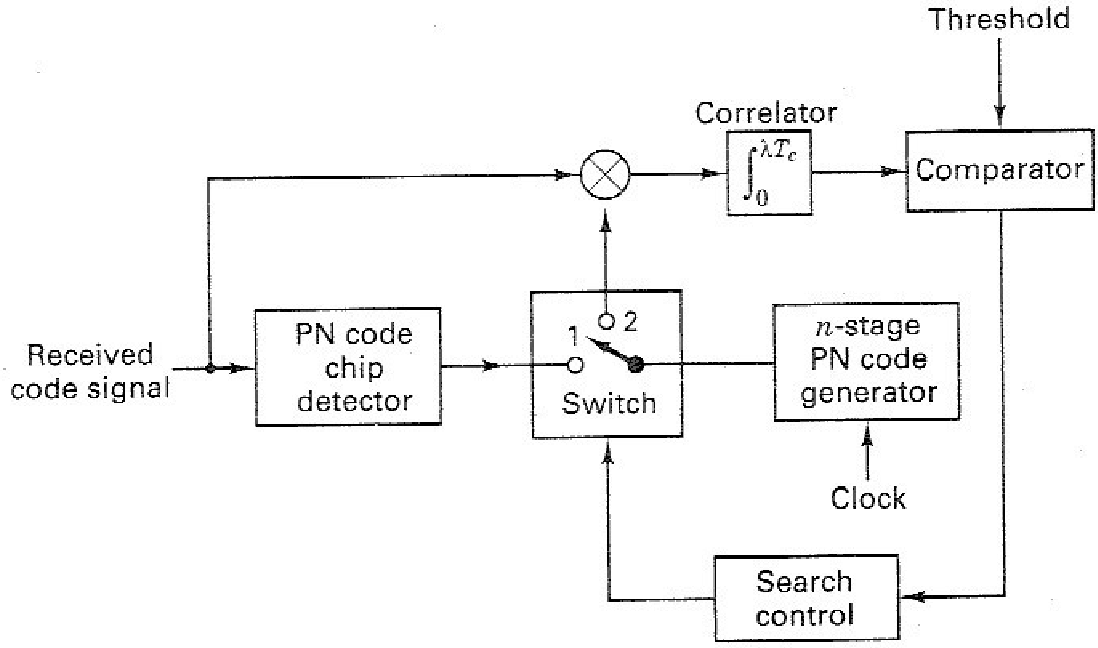


Figure 12.21 Rapid acquisition by sequential estimation.

with estimates of the next n received chips, and the procedure is repeated. Once synchronization has occurred, the system no longer needs estimates of the input code chips. We can calculate the *minimum* acquisition time for the case when no noise is present. The first n chips will be correctly loaded into the register, and therefore, the acquisition time is

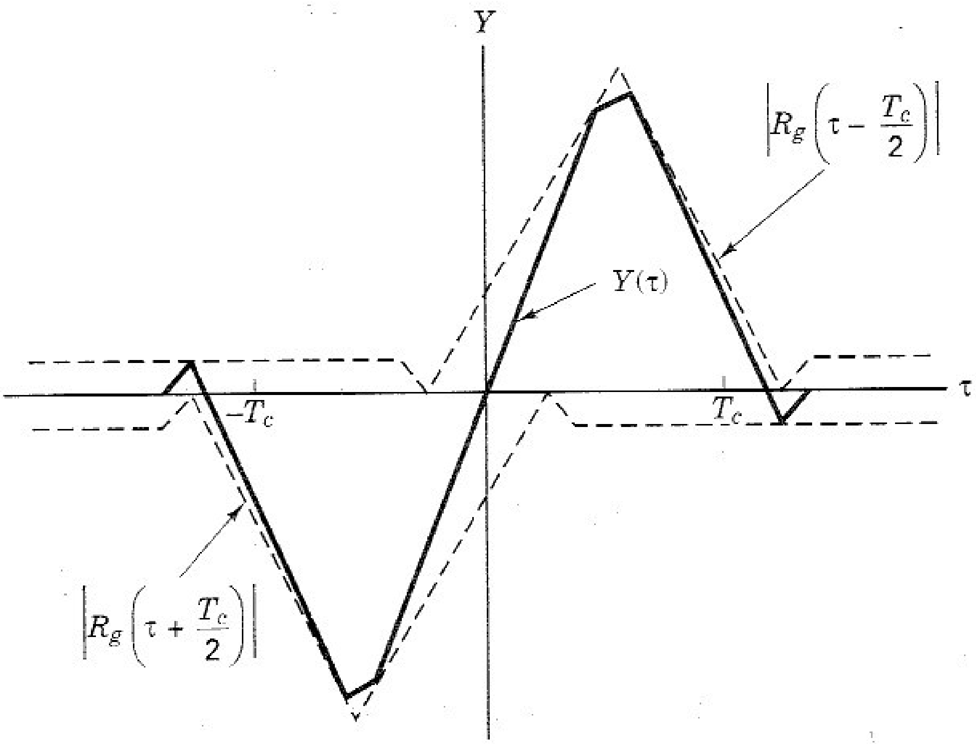
$$T_{\text{acq}} = nT_c \tag{12.35}$$

While the RASE system has a rapid acquisition capability it has the drawback of being highly vulnerable to noise and interference signals. The reason for this is that the estimation process consists of a simple chip-by-chip hard-decision demodulation, without using the interference rejection benefits of the PN code.

For an extensive treatment of sequential estimation, see Reference [4].

12.5.2 Tracking

Once acquisition or coarse synchronization is completed, tracking or fine synchronization takes place. Tracking code loops can be classified as coherent or noncoherent. A coherent loop is one in which the carrier frequency and phase are known exactly so that the loop can operate on a baseband signal. A noncoherent loop is one in which the carrier frequency is not known exactly (due to Doppler effects, for example), nor is the phase. In most instances, since the carrier frequency and phase are not known exactly, a priori, a noncoherent code loop is used to track the received PN code. Tracking loops are further classified as a *full-time* early-late tracking loop, often referred to as a *delay-locked loop* (DLL), or as a *time-shared* early-late tracking loop, frequently referred to as a *tau-dither loop* (TDL). A basic



12.23 DLL feedback signal $Y(\tau)$.

despread signal $Z(t)$, which is then applied to the input of a conventional data demodulator. Detailed analysis of the DLL can be found in References [4, 12–14].

A problem with the DLL is that the early and late arms must be precisely gain balanced or else the feedback signal $Y(\tau)$ will be offset and will not produce a zero signal when the error is zero. This problem is solved by using a time-shared tracking loop in place of the full-time delay-locked loop. The time-shared loop time shares the use of the early-late correlators. The main advantages are that only one correlator need be used in the design of the loop, and further, that dc offset problems are reduced.

A problem with some control loops is that if things are going well and the loop is tracking accurately, the control signal is essentially zero. When the control signal is zero, the loop can get "confused" and do erratic things. This is especially the case in more sophisticated tracking loops that modify their own loop gain in response to the perceived environment. An offshoot of the time-shared tracking loop, called the *tau-dither loop* (TDL), shown in Figure 12.24, tends to deal with this potential problem by intentionally injecting a small error in the tracking correction, so that the loop kind of vibrates around the correct answer. This vibration is typically small, so that the loss in performance is minimal. This design has the advantage that only one correlator is needed to provide the code *tracking* function *and* the *despreading* function. Just as in the case of a DLL, the received signal is correlated with an early and a late version of the locally generated PN code. As shown in Figure 12.24, the PN code generator is driven by a clock signal whose phase is *dithered* back and forth with a square-wave switching function; this eliminates the necessity of ensuring identical transfer functions of the early and late

Amp less
freq. loop

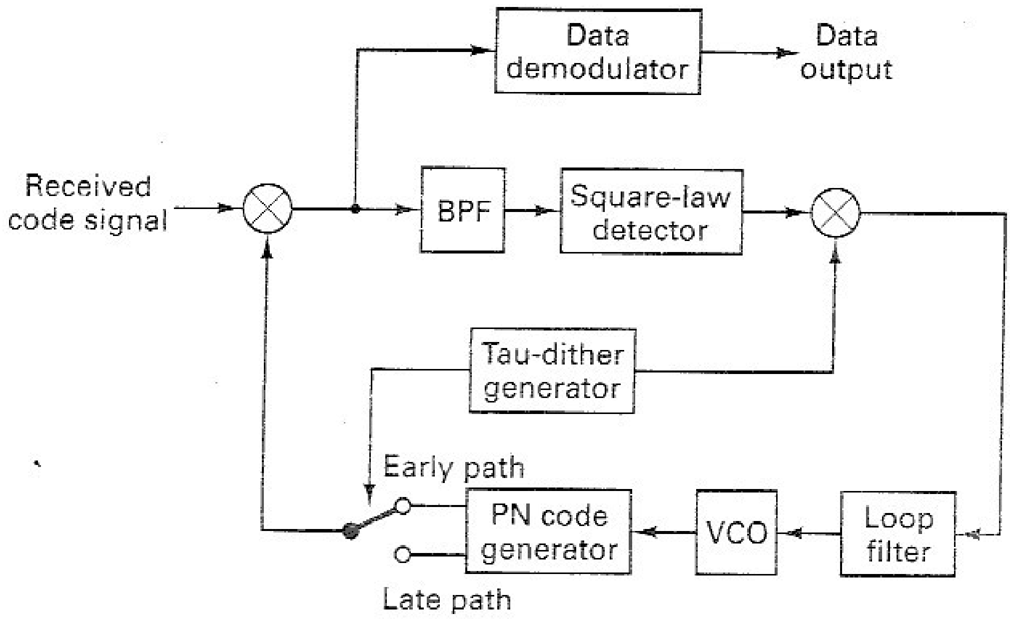


Figure 12.24 Tau-dither tracking loop.

paths. The signal-to-noise performance of the TDL is only about 1.1 dB worse than that of the DLL if the arm filters are designed properly [4]. (For a comprehensive treatment of synchronization of PN codes, see References [4, 15, 16].)

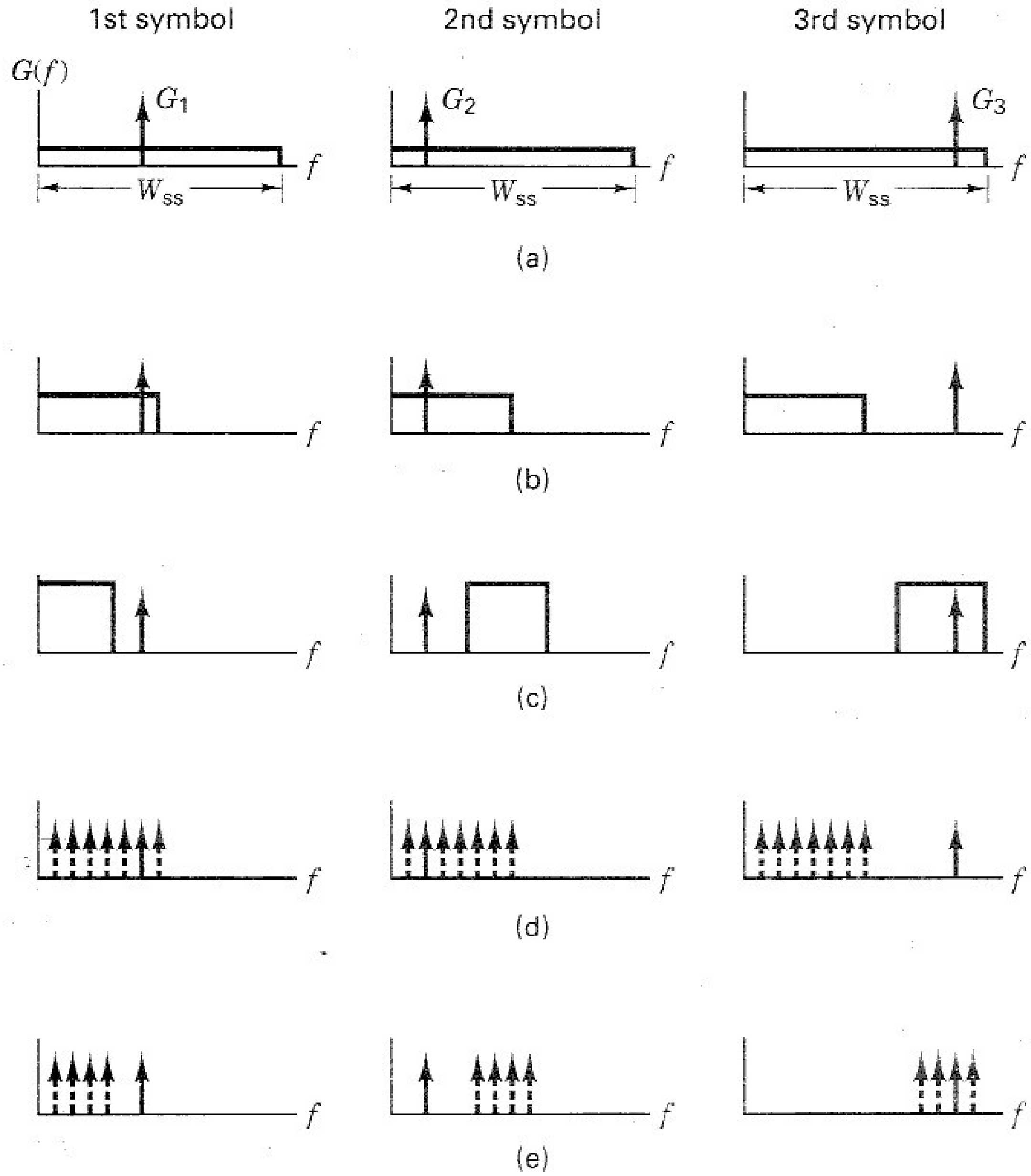
12.6 JAMMING CONSIDERATIONS

12.6.1 The Jamming Game

The goals of a jammer are to deny reliable communications to his adversary and to accomplish this at minimum cost. The goals of the communicator are to develop a jam-resistant communication system under the following assumptions: (1) complete invulnerability is not possible; (2) the jammer has a priori knowledge of most system parameters, such as frequency bands, timing, traffic, and so on; (3) the jammer has *no* a priori knowledge of the PN spreading or hopping codes. The signaling waveform should be designed so that the jammer cannot gain any appreciable jamming advantage by choosing a jammer waveform and strategy other than wideband Gaussian noise (i.e., being clever should gain nothing for the jammer). The fundamental design rule in specifying a jam-resistant system is to make it as costly as possible for the jammer to succeed in jamming the system.

12.6.1.1 Jammer Waveforms

There are many different waveforms that can be used for jamming communication systems. The most appropriate choice depends on the targeted system. Figure 12.25 shows power spectral density plots of examples of jammer waveforms versus a communicator's frequency hopped M-ary FSK (FH/MFSK) tone. The



12.25 Jammer waveforms. (a) Full-band noise. (b) Partial-band noise. (c) Stepped noise. (d) Partial-band tones. (e) Stepped tones.

range of the abscissa represents the spread-spectrum bandwidth W_{ss}. The three columns in the figure represent three instances in time (three hop times) when symbols having spectra G_1, G_2, and G_3, respectively, are being transmitted. Figure 12.25a illustrates a relatively low-level noise jammer occupying the full spread-spectrum bandwidth. In Figure 12.25b the jammer strategy is to trade bandwidth occupancy for greater power spectral density (the total power, or area under the curve, remains the same). The figure indicates that in this case, the jammer noise does not always share the same bandwidth region as the signal, but when it does, the effect can be destructive. In Figure 12.25c the noise jammer strategy is again to jam only part of the band, so that the jammer power spectral density can be increased, but in this case the jammer steps through different regions of the band at random times, thus preventing the communicator from using adaptive techniques to avoid the jamming. In Figure 12.25d and e the jammer uses a group of tones, instead of a continuous frequency band, in partial-band (Figure 12.25d) and stepped fashion (Figure 12.25e). This is a technique most often used against FH systems. Another jamming technique, not shown in Figure 12.25, is a pulse jammer, consisting of pulse-modulated bandlimited noise. Unless otherwise stated, we shall assume that the jammer waveform is wideband noise and that the jammer strategy is to jam the entire bandwidth W_{ss} continuously. The effects of partial band jamming and pulse jamming are considered later.

12.6.1.2 Tools of the Communicator

The usual design goal for an anti-jam (AJ) communication system is to force a jammer to expend its resources over (1) a wide-frequency band, (2) for a maximum time, and (3) from a diversity of sites. The most prevalent design options are (1) frequency diversity, by the use of direct-sequence and frequency-hopping spread-spectrum techniques; (2) time diversity, by the use of time hopping; (3) spatial discrimination, by the use of a narrow-beam antenna, which forces a jammer to enter the receiver via an antenna sidelobe and hence suffer, typically, a 20- to 25-dB disadvantage, and (4) combinations of the previous three options.

12.6.1.3 *J/S* Ratio

In Chapter 5 we were concerned primarily with link error performance as a function of thermal noise interference. Emphasis was placed on the signal-to-noise ratio parameters—required E_b/N_0 and available E_b/N_0 for meeting a specified error performance. In this section we are similarly concerned with link error performance as a function of interference. However, here the source of interference is assumed to be wideband Gaussian noise power from a jammer in addition to thermal noise. Therefore, the SNR of interest is $E_b/(N_0 + J_0)$, where J_0 is the noise power spectral density due to the jammer. Unless otherwise specified, J_0 is assumed equal to J/W_{ss}, where J is the average received jammer power (jammer power referred to the receiver front end) and W_{ss} is the spread-spectrum bandwidth. Since the jammer power is generally much greater than the thermal noise power, the SNR of interest in a jammed environment is usually taken to be E_b/J_0. Therefore, similar to the thermal noise case, we define $(E_b/J_0)_{reqd}$ as the bit energy per jammer noise power spectral density *required* for maintaining the link at a specified error probability. The parameter E_b can be written as

$$E_b = ST_b = \frac{S}{R}$$

where S is the received signal power, T_b the bit duration, and R the data rate in bits/s. Then we can express $(E_b/J_0)_{reqd}$ as

$$\left(\frac{E_b}{J_0}\right)_{reqd} = \left(\frac{S/R}{J/W_{ss}}\right)_{reqd} = \frac{W_{ss}/R}{(J/S)_{reqd}} = \frac{G_p}{(J/S)_{reqd}} \tag{12.38}$$