

# MODULATION AND DEMODULATION

*D*ata modulators, especially those intended to produce constant-envelope output signals, are “high-leverage” components in that even very small deviations from ideal in their behavior can lead to large degradations in overall system performance. Therefore, successful simulation of wireless communication systems depends upon the use of modulator models that capture all of the significant deviations from ideal behavior.

In the “usual” development of data modulation techniques as presented in most communications texts, the various techniques are presented in order of complexity, starting with the simplest. Thus BPSK would be presented first, then QPSK followed by *m*-PSK, and so on. Because of its relationship to complex-envelope representations of signals, quadrature modulation plays a central role in simulation of wireless communication systems and models for quadrature modulators, and demodulators serve as building blocks for most other types of data modulators and demodulators. Therefore, this chapter begins with a discussion of *quadrature phase shift keying* (QPSK) and uses this discussion as a vehicle for development of generic models for quadrature modulation and demodulation. The discussion then moves to *binary phase shift keying* (BPSK) and shows how this simpler format is modeled using the generic quadrature modulation models. A similar approach is then taken for developing models for *multiple phase shift keying* (*m*-PSK), *minimum shift keying* (MSK), and *frequency shift keying* (FSK).

## 9.1 Simulation Issues

The tasks of carrier recovery and symbol-clock regeneration, which are usually considered part of the demodulation process, are an essential part of any data communication system. There are a number of different techniques for accomplishing these

tasks, and these techniques can be used across a wide range of modulation formats and demodulation schemes. If we were to implement every possible combination of demodulation algorithm, carrier-recovery technique, and clock regeneration as a distinct model, the combinatorial explosion of different models would become unmanageable. Therefore, it is a common practice to implement the demodulation, carrier recovery, and clock regeneration as separate models that can be put together in any desired combination in a simulation. In some particular situations (such as a cross-strapped Costas loop demodulator) it still makes sense to combine all three tasks into a single demodulator model.

### 9.1.1 Using the Recovered Carrier

When a signal passes through a nonideal channel, it is subjected to a certain amount of phase shift. For bandpass simulations, the recovered carrier can be used directly as a phase reference mimicking the way most real-world demodulators operate. For complex baseband simulations, the recovered carrier consists of a phase angle which, in general, may be slowly time-varying. We have several choices with regard to how such a carrier is used in simulations:

1. Use a model external to the demodulator to rotate the phase of the received signal so that the reference phase effectively becomes zero. The rotated signal can then be input to a demodulator model that assumes a reference phase of zero. The phase rotation requires one complex multiplication for each sample of the received signal. The external model must fetch each sample of the signal and each sample of the reference from memory, perform the rotation, and then store the result back into memory. The demodulator must then fetch each sample of the rotated signal from memory.
2. Pass the recovered carrier phase into the demodulator model where it can be used to rotate the phase of the received signal so that the reference phase effectively becomes zero. This phase rotation requires one complex multiplication for each sample of the received signal, regardless of whether the recovered carrier phase is constant or time-varying. The demodulator can operate on each sample of the rotated result as it is generated—there is no need to store the result and then fetch it again as in the first approach.
3. Pass the recovered carrier phase into the demodulator model where it can be used directly as the reference phase. The computational expense of this approach depends upon the particular modulation format and demodulation algorithm being implemented. In a correlation receiver for 8-PSK, the recovered carrier phase can be used directly as the zero-phase reference and each

of the other seven ideal phase references must be shifted by an amount equal to the recovered carrier phase. If the received carrier phase is assumed to be constant, each of the seven reference values will have to be shifted only once at the beginning of the simulation. If the received carrier phase is time-varying, each reference will have to be shifted once for each new sample of the recovered carrier phase.

4. Follow approach 3 with one change. Instead of automatically shifting the set of references for each new sample of the recovered carrier phase, first compare the sample to the previous sample to see if there has been a change. If there has been no change in the recovered carrier phase, there is no need to generate a new set of reference values.

The relative computational burdens of these four approaches are summarized in Table 9.1. Typically, the changes in carrier phase are very slow compared to changes in the modulated signal. Therefore, it may frequently be the case that the recovered carrier will be simulated using a sample rate that is lower than the sample rate used for the modulated signal. Therefore the table is constructed for the case of a fixed-duration block that contains  $N$  samples of the received signal and  $N_{\text{ref}}$  samples of recovered carrier phase. Clearly, approach 1 is the most expensive in terms of computation. The relative cost of the other approaches depends upon the values of  $N$  and  $N_{\text{ref}}$  as well as the relative difficulties of performing a sample rotation and of calculating the demodulation references for a given carrier phase.

## 9.2 Quadrature Phase Shift Keying

In QPSK, the data bits to be modulated are grouped into symbols, each containing two bits, and each symbol can take on one of four possible values: 00, 01, 10, or 11. During each symbol interval, the modulator shifts the carrier to one of four possible phases corresponding to the four possible values of the input symbol. In the ideal case, the phases are each 90 degrees apart, and these phases are usually selected such that the signal constellation matches the configuration shown in Figure 9.1.

Practical QPSK modulators are often implemented using structures similar to the one shown in Figure 9.2. This structure uses the trigonometric identity

$$I \cos \omega_c t + Q \sin \omega_c t = R \cos (\omega_c t + \theta) \quad (9.2.1)$$

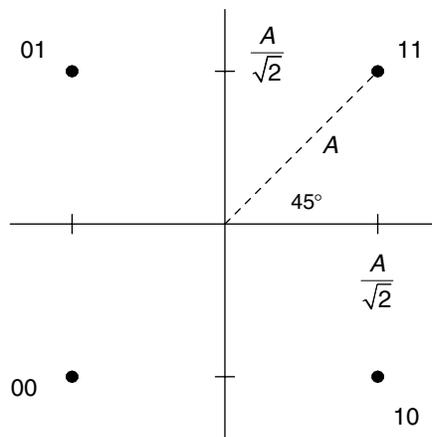
where

$$R = \sqrt{I^2 + Q^2}$$

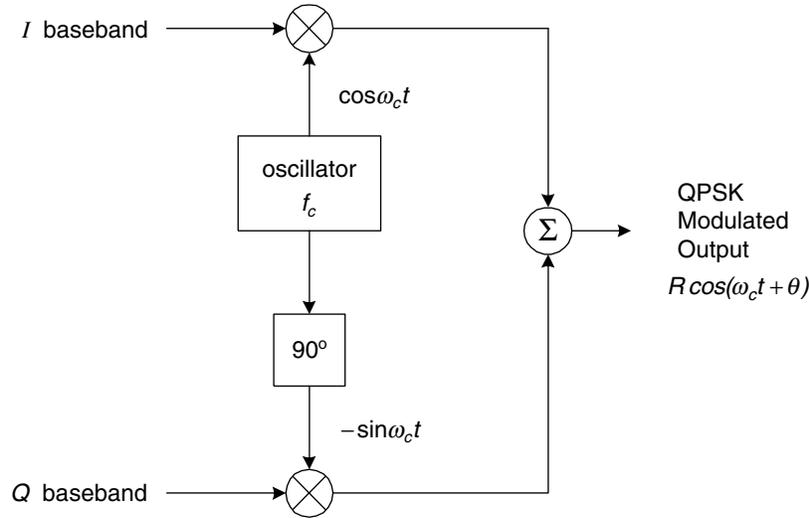
$$\theta = \tan^{-1} \left( \frac{Q}{I} \right)$$

**Table 9.1** Computational burdens of different approaches for handling recovered carrier phase.

	1	2	3	4
	External rotation of signal	Internal rotation of signal	Internal rotation of references	Internal rotation of references as needed
Operation				
Fetch sample of carrier	$N_{\text{ref}}$	$N_{\text{ref}}$	$N_{\text{ref}}$	$N_{\text{ref}}$
Fetch sample of signal	$N$	$N$	$N$	$N$
Rotate sample	$N$	$N$	0	0
Store rotated sample	$N$	0	0	0
Fetch rotated sample	$N$	0	0	0
Set demod references	$M$	$M$	$MN_{\text{ref}}$	$M \leq x \leq MN_{\text{ref}}$
Total memory access	$3N + N_{\text{ref}}$	$N + N_{\text{ref}}$	$N + N_{\text{ref}}$	$N + N_{\text{ref}}$

**Figure 9.1** Ideal QPSK constellation.

When  $I$  and  $Q$  take on values of  $\pm A/\sqrt{2}$  in all possible combinations, the phase of the resulting output signal takes on values of 45, 135, 225, and 315 degrees. If the output of this modulator is to be represented in complex-envelope form referenced



**Figure 9.2** Block diagram of QPSK modulator.

to the carrier frequency, the modulated signal is given simply as

$$\tilde{x}(t) = I(t) + jQ(t) \quad (9.2.2)$$

Simulation of this idealized signal requires only a trivial model of the modulator. The complex signal  $\tilde{x}(t)$  is formed by simply using the inphase baseband signal  $I(t)$  as the real part and the quadrature baseband signal  $Q(t)$  as the imaginary part.

### 9.2.1 Nonideal Behaviors

A practical QPSK modulator implementation similar to the one shown in Figure 9.2 is subject to several sources of nonideal behavior. Some of these sources can be included in the modulator model, and some are best captured in other models external to the modulator.

#### 9.2.1.1 Quadrature Component Unbalances

The 90-degree phase shifter used to generate the quadrature component of the carrier may exhibit both amplitude and phase errors producing  $-\beta \sin(\omega_c t + \phi)$  instead of the desired  $-\sin \omega_c t$ . These errors can be viewed as a distortion of the  $I$ - $Q$  plane as shown in Figure 9.3. The  $Q$  axis is effectively mapped into the  $Q'$  axis, which is rotated from the  $Q$  axis by the angle  $\phi$ . This mapping also stretches the scale

such that the interval between tics on the  $Q'$  axis is  $\beta$  times the interval between corresponding tics on the  $Q$  axis. The amplitude error  $\beta$  is sometimes called the *amplitude unbalance* because it represents the unbalance between the nominally equal scales along the  $I$  and  $Q$  axes. Similarly, the phase error  $\phi$  is sometimes called the *phase unbalance*. When  $-\beta \sin(\omega_c t + \phi)$  is used as the quadrature component of the carrier, the modulator output becomes

$$I_{in} \cos \omega_c t - Q_{in} \beta \sin(\omega_c t + \phi) = R \cos(\omega_c t + \theta) \quad (9.2.3)$$

where

$$R = \sqrt{I_{in}^2 + Q_{in}^2 \beta^2 - 2I_{in} Q_{in} \beta \sin \phi}$$

$$\theta = \tan^{-1} \left( \frac{\beta Q_{in} \cos \phi}{I_{in} - \beta Q_{in} \sin \phi} \right)$$

The points in the resulting signal constellation are displaced from the ideal points as shown in Figure 9.3. The amplitude unbalance used in this figure is  $\beta = 1.2$ , and the phase unbalance is  $\phi = 15$  deg. These are atypically large values that were used to exaggerate the distortions in the constellation so that the relationships between the unbalances and the signal points can be more clearly shown. Typical values for modulators operating at microwave frequencies are  $\beta = 1.1$  and  $\phi = 5$  deg.

A model designed to produce a real-valued bandpass output signal could implement Eq. (9.2.3) directly. For models that produce a complex-envelope representation of the output signal, it can be shown that the modulated signal defined by Eq. (9.2.3) has inphase and quadrature components that are given by

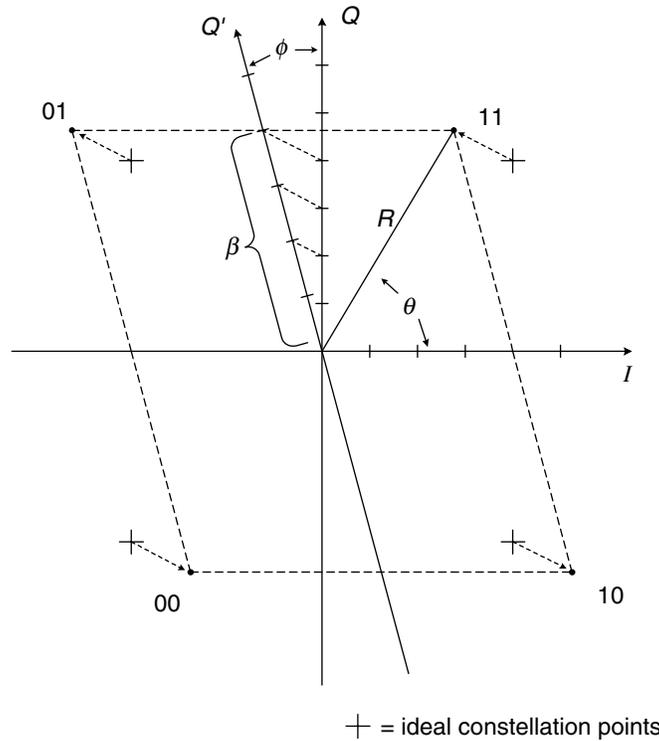
$$I_{out} = I_{in} - \beta Q_{in} \sin \phi \quad (9.2.4)$$

$$Q_{out} = \beta Q_{in} \cos \phi$$

These equations can be exploited to generate the complex envelope of the output signal directly from the  $I$  and  $Q$  baseband inputs. Furthermore, if the amplitude unbalance and phase unbalance are assumed to be constant over time, each pair of output samples at time index  $k$  can be computed via the following code fragment:

```
template< class T >
i_out[k] = i_in[k] - i_inbal*q_in[k];
q_out[k] = q_unbal * q_in[k];
```

where `i_unbal` and `q_unbal` are each computed once at the beginning of the simulation as



**Figure 9.3** QPSK constellation showing the effects of amplitude unbalance  $\beta$  and phase unbalance  $\phi$ .

```
template< class T >
i_unbal = amp_unbal * sin(phase_unbal);
q_unbal = amp_unbal * cos(phase_unbal);
```

### 9.2.1.2 Baseband Data Imperfections

The  $I$  and  $Q$  data signals are typically logic signals not suitable for direct multiplication with the quadrature carrier components. As discussed in Chapter 3, Section 3.6, driver circuits are used to convert logic signals into a bipolar baseband signal having nominal levels of  $\pm A/\sqrt{2}$ . Individually, the  $I$  and  $Q$  baseband signals are subject to degradations such as amplitudes asymmetries in the bipolar drivers and bandlimiting of the signals prior to their reaching the multipliers in the modulator. When two or more baseband signals are used together, as they are in a QPSK modulator, there is also the issue of synchronization between the signals to consider.

In ideal QPSK, the transitions in the  $I$  and  $Q$  waveforms coincide. In a practical modulator, there is some unavoidable offset between the  $I$  and  $Q$  transitions. Values for maximum allowable offset are often listed in modulator specifications as maximum  $I$ - $Q$  data skew. The baseband waveforms are also each subject to some bandlimiting, which may not be exactly the same for  $I$  and  $Q$ . Some of this bandlimiting may occur in the modulator itself, and some may occur in the circuitry that drives the modulator inputs. Regardless of where they occur in the actual system, in simulations, the time misalignment, bandlimiting, and bipolar level errors are most easily applied to the baseband  $I$  and  $Q$  signals before the signals are input to the modulator model. This approach allows maximum flexibility for generating different combinations of baseband impairments and modulation formats from a reasonably small set of constituent models.

## 9.2.2 Quadrature Modulator Models

The structure shown in Figure 9.2 produces a QPSK signal when the baseband signals  $I$  and  $Q$  take on values of  $\pm A/\sqrt{2}$ . For general values of  $I$  and  $Q$ , the structure shown in the figure is usually referred to as a *quadrature modulator*. As we will see in subsequent sections, when the proper  $I$  and  $Q$  signals are supplied, a quadrature modulator can generate BPSK,  $m$ -PSK, and MSK. It makes sense to implement quadrature modulation as a separate model that can accept arbitrary  $I$  and  $Q$  inputs and that can therefore be used as a building block to construct modulators for QPSK, OQPSK, BPSK,  $m$ -PSK, and MSK.

### 9.2.2.1 Bandpass Model

The model `QuadBandpassModulator`, summarized in Table 9.2, implements Eq. (9.2.3) to produce a real-valued bandpass output signal. Use of this model is only practical in situations where the ratio of carrier frequency to bit rate is not terribly large or where the simulation needs to be run for only a relatively small number of data bits. One common application for this model is for modeling the modulation of a subcarrier, which is subsequently combined with other subcarriers to model the operation of a frequency-division multiplexing scheme. The model design assumes that degradations to the baseband input signals will be applied by models external to the modulator.

### 9.2.2.2 Baseband Model

The model `QuadratureModulator`, summarized in Table 9.3, implements Eq. (9.2.4) to produce a complex-valued lowpass output signal. The model design

**Table 9.2** Summary of model QuadBandpassModulator.**Constructor:**

```
QuadBandpassModulator( char* instance_name,
    PracSimModel *outer_model,
    Signal<float> *i_in_sig,
    Signal<float> *q_in_sig,
    Signal<float> *out_sig,
    Signal<float> *filt_error_sig,
    Signal<float> *vco_freq_sig,
    Signal<float> *vco_phase_sig,
    Signal<float> *vco_out_sig);
```

**Parameters:**

```
double Phase_Unbal_Deg;
double Amp_Unbal;
double Carrier_Freq;
double Phase_Offset_Deg;
```

**Notes:**

1. Source code is contained in file quadmod\_bp.cpp.

assumes that degradations to the baseband input signals will be applied by models external to the modulator.

### 9.2.3 Correlation Demodulator Models for QPSK

The correlation receiver shown in Figure 9.4 is an optimum demodulator for a QPSK signal in an AWGN channel. The integrate-and-dump (I&D) filters shown in the figure each integrate over one symbol interval, and the decision device samples the integrated values at the end of this interval. The I&D filters all reset to zero before beginning the integration for the next symbol interval. The correlation receiver is optimum in the maximum a posteriori, minimum error sense, or in other words, the receiver selects as its output decision the symbol most likely to have been sent based on the value of the a posteriori PDFs  $p(s_i|y)$ .

As discussed in Section 9.1.1, there are a number of ways to actually implement a model of a correlation receiver. In order to select the most efficient approach, we must refer back to Table 9.1 and determine the QPSK-specific cost of setting the

**Table 9.3** Summary of model QuadratureModulator.**Constructor:**

```

QuadratureModulator( char* instance_name,
                    PracSimModel *outer_model,
                    Signal<float> *i_in_sig,
                    Signal<float> *q_in_sig,
                    Signal< std::complex<float> > *
                    cmpx_out_sig);

```

**Parameters:**

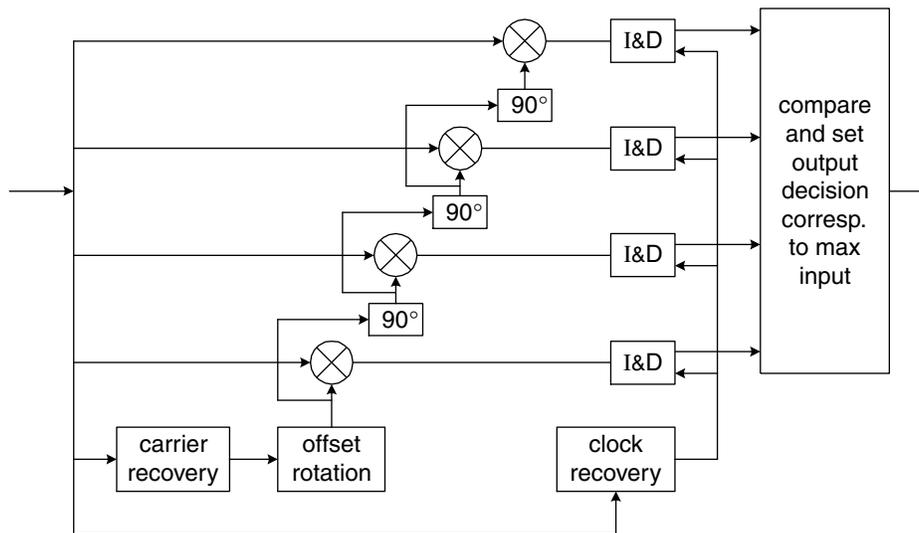
```

double Phase_Unbal;
double Amp_Unbal;

```

**Notes:**

1. Source code is contained in file quadmod.cpp.

**Figure 9.4** Correlation receiver for QPSK.

demodulator references. The brute-force approach for generating the correlation references for QPSK is given by

$$s_k = \cos\left(\pi \frac{2k+d}{4} + \theta\right) - j \sin\left(\pi \frac{2k+d}{4} + \theta\right) \quad k = 0, 1, 2, 3 \quad (9.2.5)$$

where  $\theta$  is the phase of the recovered carrier. For the case where the transmitted constellation points lie on the  $I$  and  $Q$  axes,  $d = 0$ , and for the case where the transmitted constellation points are offset from the axes by 45 degrees,  $d = 1$ . With respect to computational burden listed in Table 9.1, the worst case occurs when  $\theta$  is time-varying and sampled at the same rate as the signal. In this case, approach 3 from Section 9.1.1 would require  $4N$  cosine evaluations and  $4N$  sine evaluations for each simulation block. This burden does not compare favorably with the  $N$  sample rotations, 4 cosine evaluations, and 4 sine evaluations needed for approach 2. However, for QPSK, there is a more efficient way to determine the demodulator references. If the carrier phase is actually provided to the demodulator model as a real-valued phase angle, the set of correlation references  $\{s_0, s_1, s_2, s_3\}$  can be obtained as

$$a = \cos\left(\frac{d\pi}{4} + \theta\right) \quad (9.2.6a)$$

$$b = \sin\left(\frac{d\pi}{4} + \theta\right) \quad (9.2.6b)$$

$$s_0 = a - jb \quad (9.2.6c)$$

$$s_1 = b + ja = js_0 \quad (9.2.6d)$$

$$s_2 = -a + jb = js_1 = -s_0 \quad (9.2.6e)$$

$$s_3 = -b - ja = js_2 = -s_1 = -js_0 \quad (9.2.6f)$$

If the recovered carrier is provided to the demodulator model as a complex value  $C$ , the set of references is obtained as

$$s_0 = \begin{cases} C & d = 0 \\ \left(\frac{\sqrt{2}}{2} + j\frac{\sqrt{2}}{2}\right)C & d = 1 \end{cases} \quad (9.2.7a)$$

$$s_1 = js_0 \quad (9.2.7b)$$

$$s_2 = js_1 = -s_0 \quad (9.2.7c)$$

$$s_3 = js_2 = -s_1 = -js_0 \quad (9.2.7d)$$

If approach 3 from Section 9.1.1 is being implemented, the carrier should be provided as a complex value because otherwise Eqs. (9.2.6a) and (9.2.6b) must be evaluated for each sample of the demodulator input signal—evaluation of Eq. (9.2.7a) for each sample is by far the smaller burden. The model `QpskOptimalBitDemod`, summarized in Table 9.4, accepts the recovered carrier as a complex-valued input signal.

**Table 9.4** Summary of model `QpskOptimalBitDemod`.

**Constructor:**

```
QpskOptimalBitDemod( char* instance_nam,
                    PracSimModel *outer_model,
                    Signal< std::complex< float > >
                    *in_sig,
                    Signal< std::complex< float > >
                    *carrier_ref_sig,
                    Signal< bit_t > *symb_clock_in,
                    Signal< bit_t > *i_decis_out,
                    Signal< bit_t > *q_decis_out);
```

**Parameters:**

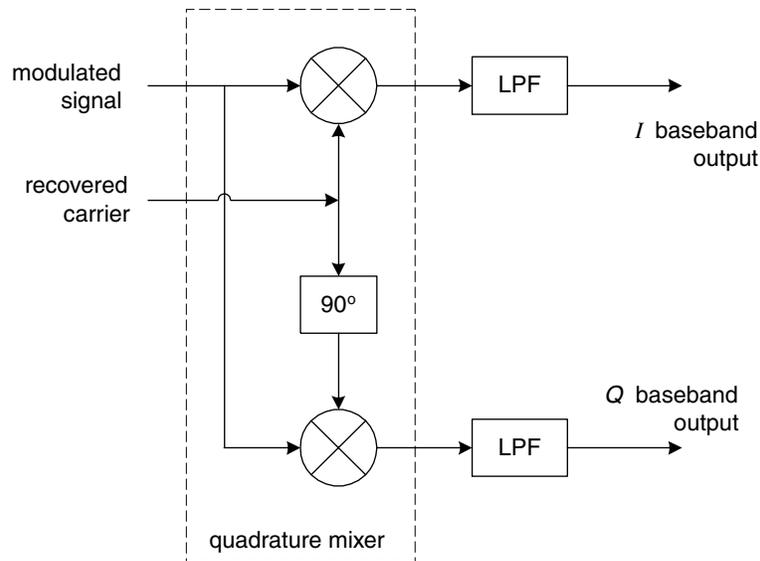
```
int Samps_Per_Symb;
bool Constel_Offset_Enabled;
```

**Notes:**

1. Source code is contained in file `qpskoptbitdem.cpp`.
2. This is a multirate model.

## 9.2.4 Quadrature Demodulator Models

The block diagram for a quadrature demodulator is shown in Figure 9.5. Prior to lowpass filtering, the output of each multiplier will contain both baseband components and “double-frequency” components. A segment of the *I*-channel multiplier output for the case of QPSK is shown in Figure 9.6. The lowpass filters remove the double-frequency components, leaving just the baseband components. In the ideal case, the filtered *I* and *Q* outputs duplicate the *I* and *Q* waveforms that were originally input to the quadrature modulator at the transmitter. For the case of QPSK, the *I* and *Q* outputs from the quadrature demodulator can be processed separately



**Figure 9.5** Quadrature demodulator.

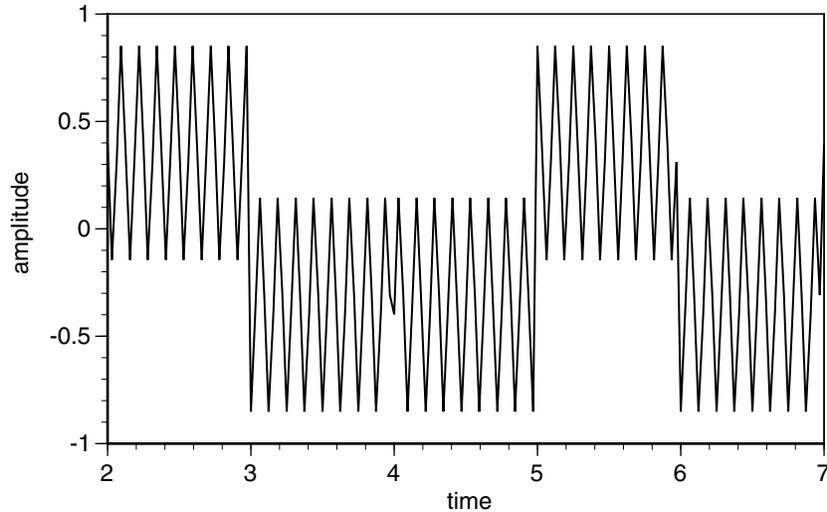
as independent binary waveforms. Figure 9.7 shows how a quadrature demodulator can be augmented for demodulation of QPSK. The processing needed to recover bit clocks from the  $I$  and  $Q$  baseband waveforms is discussed at in Chapter 12.

#### 9.2.4.1 Bandpass Model

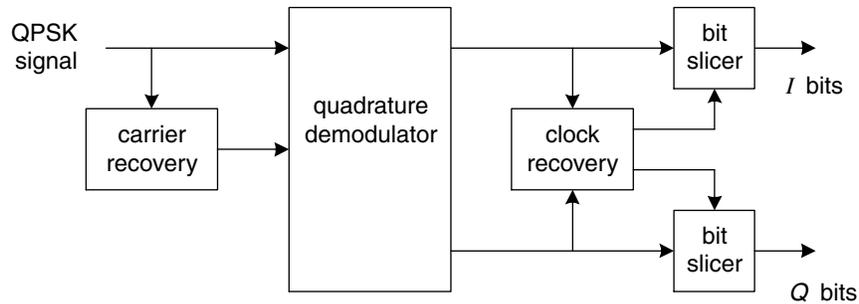
To allow for maximum flexibility with regard to modeling the lowpass filters, `PracSim` does not implement a quadrature bandpass demodulator as a single model. Instead, a bandpass model of a quadrature mixer is used with separate  $I$  and  $Q$  lowpass filter models. Furthermore, the 90-degree phase shifter is moved outside of the quadrature mixer to yield the configuration shown in Figure 9.8. The model `QuadBandpassMixer` is summarized in Table 9.5.

#### 9.2.4.2 Baseband Model

Unlike a bandpass mixer, a complex-envelope lowpass model of a mixer does not produce double-frequency components. Therefore, the lowpass filters shown in Figure 9.8 are not mandatory. However, if the system being modeled does not include bandlimiting filters prior to the mixer, lowpass filters after the mixer may need to be included for reduction of broadband noise. The model `QuadratureDemod`



**Figure 9.6** Quadrature demodulator's  $I$ -channel output prior to lowpass filtering.



**Figure 9.7** Quadrature demodulator used to demodulate QPSK.

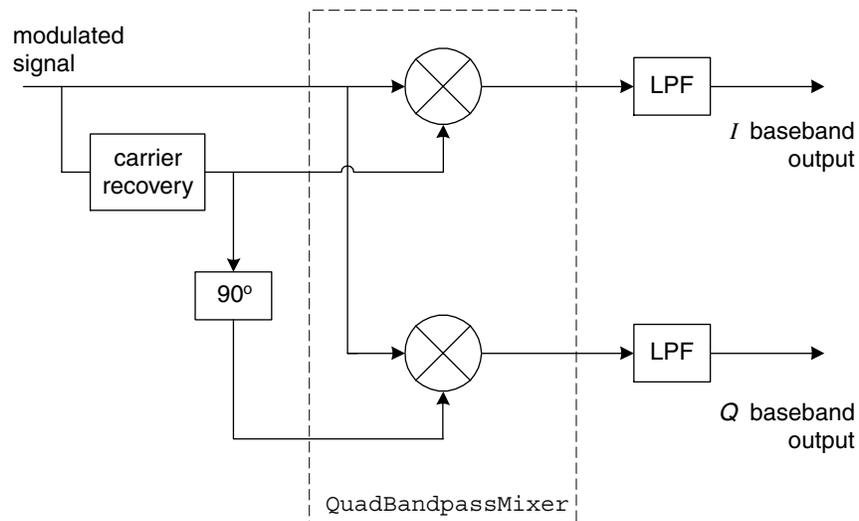
summarized in Table 9.6, implements a baseband quadrature mixer.

### 9.2.5 QPSK Simulations

Three different simulations of QPSK are provided on the companion Web site: a bandpass simulation using a quadrature demodulator, a complex baseband simulation using a correlation demodulator, and a complex baseband simulation using a quadrature demodulator.

**Table 9.5** Summary of model QuadBandpassMixer.

<p><b>Constructor:</b></p> <pre>QuadBandpassMixer( char* instance_nam,     PracSimModel *outer_model,     Signal&lt;float&gt; *in_sig,     Signal&lt;float&gt; *i_ref_sig,     Signal&lt;float&gt; *q_ref_sig,     Signal&lt;float&gt; *i_out_sig,     Signal&lt;float&gt; *q_out_sig );</pre>
<p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file quad_mixer_bp.cpp.</li> </ol>

**Figure 9.8** Block diagram of model QuadBandpassMixer.

### 9.2.5.1 Bandpass Simulation

Figure 9.9 shows how the models `QuadBandpassModulator` and `QuadBandpassMixer` can be used in a bandpass simulation of QPSK. The carrier reference for the mixer is provided by the model `QuadCarrierRecovGenie`, which is summarized

**Table 9.6** Summary of model QuadratureDemod.**Constructor:**

```

QuadratureDemod( char* instance_nam,
                 PracSimModel *outer_model,
                 Signal< complex<float> > *in_sig,
                 Signal< complex<float> >
*carrier_ref_sig,
                 Signal<float> *i_wave_out,
                 Signal<float> *q_wave_out);

```

**Notes:**

1. Source code is contained in file quaddem.cpp.

in Table 9.7. This model is tagged as a “genie” because rather than actually recovering the carrier from the received signal, the model “magically” obtains the carrier by reading from the parameter file the same values for carrier frequency and phase offset that are used by the transmitter models. The inphase and quadrature subcarriers are provided as two separate real-valued signals so that the mixer does not need to implement a 90-degree phase shift.

**Table 9.7** Summary of model QuadCarrierRecovGenie.**Constructor:**

```

QuadCarrierRecovGenie( char* instance_nam,
                       PracSimModel *outer_model,
                       Signal<float> *i_recov_carrier,
                       Signal<float> *q_recov_carrier);

```

**Parameters:**

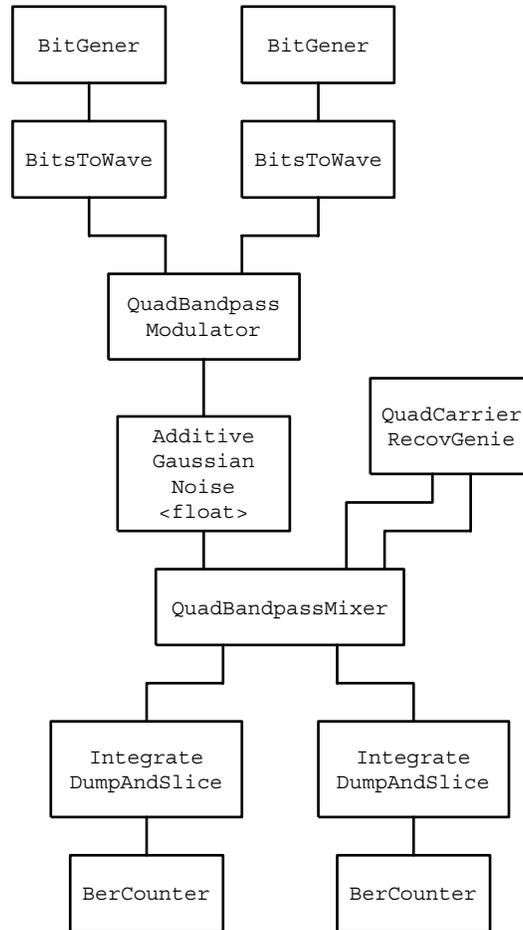
```

double Carrier_Freq;
double Phase_Offset_Deg;

```

**Notes:**

1. Source code is contained in file quad\_carr\_genie.cpp.



**Figure 9.9** Block diagram for bandpass simulation of QPSK using a quadrature demodulator.

The `IntegrateAndDump` filter models shown in the figure take the place of both the lowpass filters and the bit-slicing units shown in Figure 9.7. The clock input for these filters is “hardwired” from the clock outputs of the two baseband waveform generators used in the transmitter. A simulation that follows the configuration of Figure 9.9 is provided as the project `QpskSim_Bp`, which uses the main program contained in `qpsk_sim_bp.cpp`.

### 9.2.5.2 Baseband Simulations

Figure 9.10 shows how the models `QuadratureModulator` and `QuadratureDemod` can be used in a complex baseband simulation of QPSK. The carrier reference for the mixer is provided by the model `PhaseRecoveryGenie` found in the file `phase_genie.cpp`. The carrier is provided as a single complex-valued signal. This simulation uses the same `IntegrateAndDump` filters that are used in the bandpass simulation. A simulation that follows the configuration of Figure 9.10 is provided as the project `QpskSim`, which uses the main program contained in `qpsk_sim.cpp`.

The configuration of Figure 9.10 can be modified by substituting `QpskOptimalBitDemod` for the `QuadratureMixer` model and both instances of the `IntegrateAndDump` filters. A simulation that follows this modified configuration is provided as the project `QpskSim_Corr`, which uses the main program contained in `qpsk_sim_corr.cpp`.

### 9.2.6 Properties of QPSK Signals

The PSD of a QPSK signal using rectangular data pulses is given by

$$P_{\text{QPSK}} = \frac{E_S}{2} \left[ \left( \frac{\sin[\pi(f - f_c)T_S]}{\pi(f - f_c)T_S} \right)^2 + \left( \frac{\sin[\pi(f + f_c)T_S]}{\pi(f + f_c)T_S} \right)^2 \right] \quad (9.2.8a)$$

$$= \frac{E_S}{2} \left[ \text{sinc}^2((f - f_c)T_S) + \text{sinc}^2((f + f_c)T_S) \right] \quad (9.2.8b)$$

$$= \frac{E_S}{2} \left[ \text{sinc}^2\left(\frac{f - f_c}{R_S}\right) + \text{sinc}^2\left(\frac{f + f_c}{R_S}\right) \right] \quad (9.2.8c)$$

where

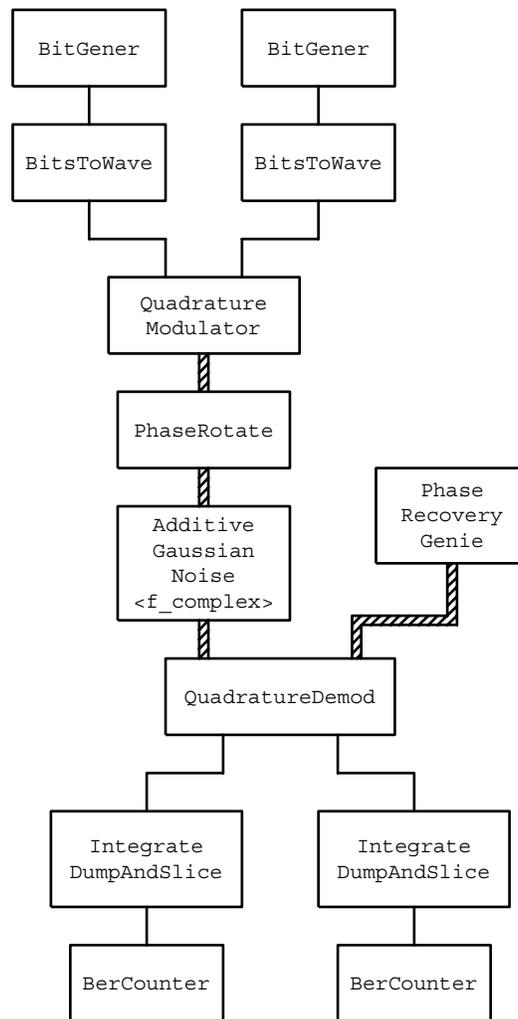
$E_S$  = energy per symbol

$T_S$  = symbol interval

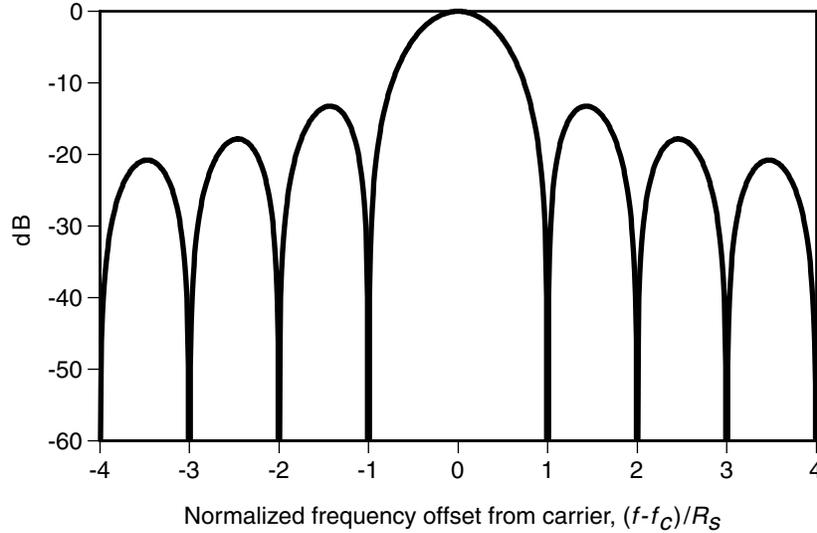
$R_S = T_S^{-1}$  = symbol rate

$f_c$  = carrier frequency

A normalized plot of Eq. (9.2.8) is shown in Figure 9.11. The data for this plot was generated using `QpskPsd`, which can be found in file `qpsk_theory.cpp` in the `utils` directory on the companion Web site. An estimated PSD for a complex baseband simulation of a QPSK signal is shown in Figure 9.12. Notice that the sidelobe structure of this one-sided spectrum is nearly identical to the sidelobe structure shown for frequencies above  $f_c$  in Figure 9.11.



**Figure 9.10** Block diagram for complex baseband simulation of QPSK using a quadrature demodulator.



**Figure 9.11** PSD for a QPSK signal using rectangular data pulses.

### 9.2.6.1 Error Performance

It can be shown (see [15]) that the probability of bit error for QPSK is obtained as

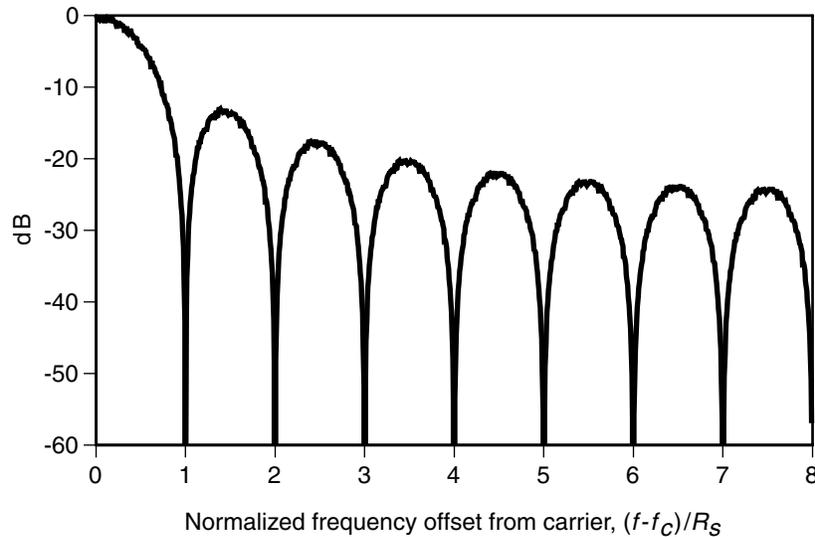
$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = \frac{1}{2} \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right) \quad (9.2.9)$$

where  $Q(x)$  is the  $Q$ -function defined in Appendix B, Section B.2. The probability of making a correct bit decision is  $(1 - P_b)$  and the probability of making a correct symbol decision equals the probability of making correct decisions for both bits in the symbol or

$$\begin{aligned} P_c &= (1 - P_b)^2 \\ &= 1 - 2P_b + P_b^2 \end{aligned}$$

The probability of symbol error is obtained as

$$\begin{aligned} P_s &= 1 - P_c \\ &= 2P_b - P_b^2 \\ &= \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right) \left[1 - \frac{1}{4} \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right)\right] \end{aligned}$$



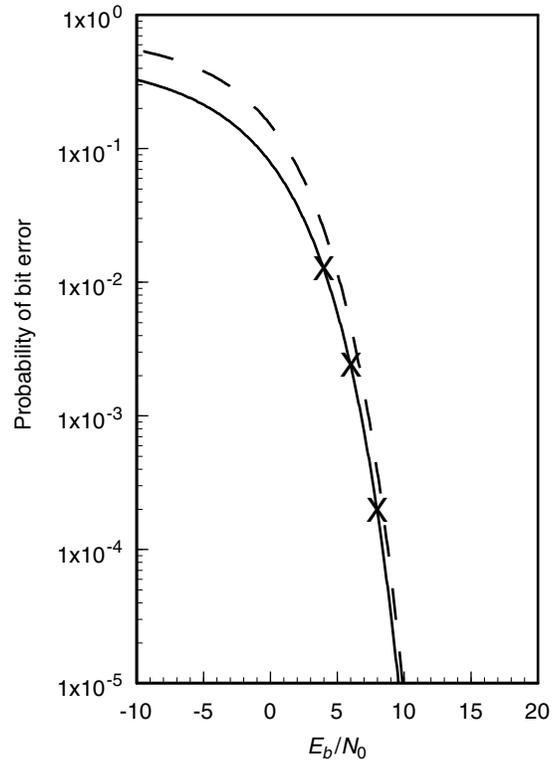
**Figure 9.12** PSD estimated from a complex baseband simulation of a QPSK signal using rectangular data pulses.

Plots of  $P_b$  and  $P_s$  are provided in Figure 9.13. The data for these plots was obtained using the function `QpskBer`, which can be found in file `qpsk_theory.cpp` in the `utils` directory. The three points marked “ $\times$ ” in the figure were obtained via simulation using rectangular pulses and a perfectly synchronized I&D demodulator.

### 9.2.7 Offset QPSK

Consider the constellation shown in Figure 9.14. When a single input bit changes value, the output signal “moves” from one vertex of the dotted parallelogram to an adjacent vertex. However, when both input bits change value at the same time, the output signal moves from one vertex to the opposite vertex along a diagonal passing through the origin of the  $I$ - $Q$  plane. In the ideal case, the difference between single-bit and double-bit changes is not important because the transition from one vertex to another occurs instantaneously. However, in the case of  $I$  and  $Q$  baseband waveforms having nonzero rise and fall times, the transitions are not instantaneous and the difference becomes very noticeable.

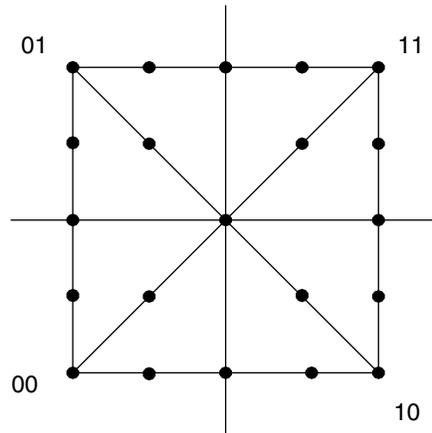
Figure 9.14 shows the constellation with transition trajectories included for a QPSK modulator driven by baseband  $I$  and  $Q$  signals that have been smoothed



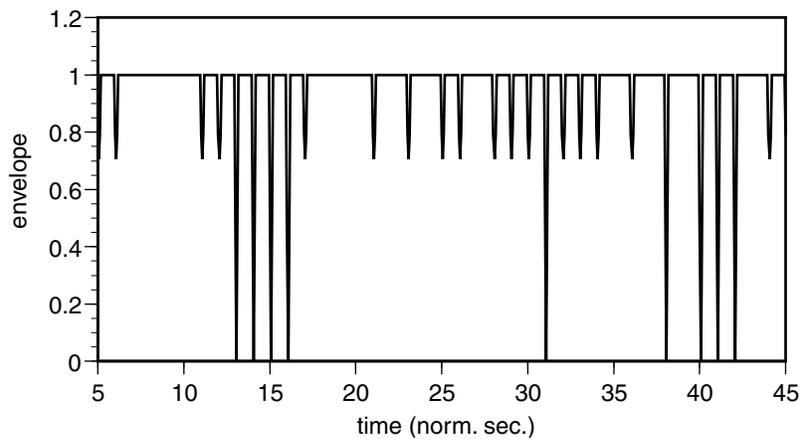
**Figure 9.13** Probability of error for QPSK signals.

using a four-sample rectangular averaging window. (There are 16 samples per bit.) Samples along the transition paths are shown as dots. As shown in Figure 9.15, the envelope of the modulated signal contains deep nulls corresponding to the transitions that pass through the origin of the  $I$ - $Q$  plane. These nulls are very undesirable if the signal is to be subsequently passed through a nonlinear amplifier.

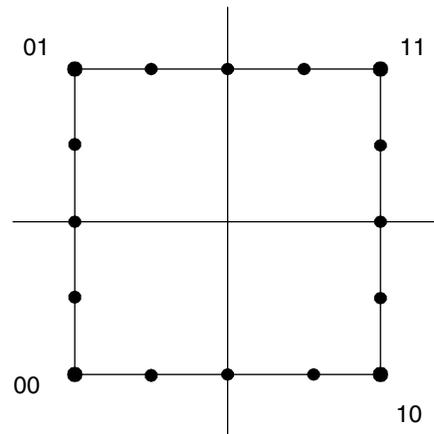
If the  $I$  and  $Q$  baseband signals are offset by half a bit interval, one input is guaranteed to remain relatively constant while the other input changes in value. As shown in Figure 9.16, this eliminates transitions that go through the origin. This form of QPSK is called *offset quadrature phase shift keying* (OQPSK) or occasionally *staggered quadrature phase shift keying* (SQPSK). As shown in Figure 9.17, the envelope of OQPSK is still not constant, but the deep nulls have been eliminated. Standard QPSK simulation models can be used to generate OQPSK by simply delaying the  $Q$  baseband signal by half a bit period prior to the modulator input. A simulation of OQPSK is provided as the project `OqpskSim`, which uses the main program contained in `oqpsk_sim.cpp`.



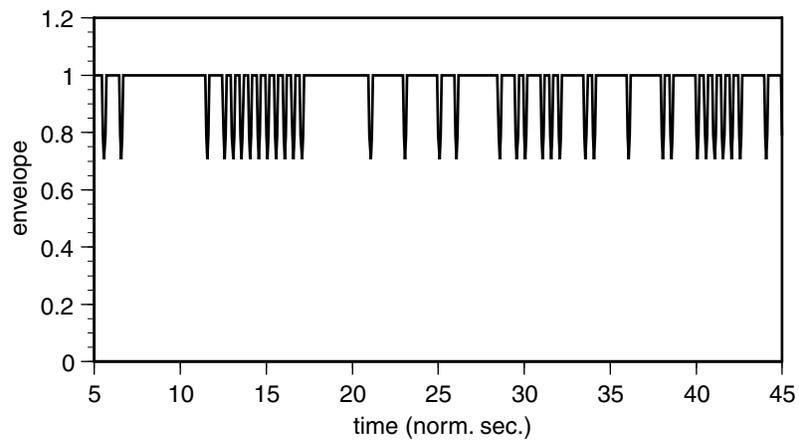
**Figure 9.14** QPSK constellation and interstate transitions for the case when the baseband waveforms are smoothed using a four-sample rectangular averaging window.



**Figure 9.15** Envelope of QPSK signal showing deep nulls due to double-bit transitions.



**Figure 9.16** Constellation with transition trajectories for OQPSK.



**Figure 9.17** Envelope of OQPSK signal.

### 9.3 Binary Phase Shift Keying

In BPSK, individual data bits are used to control the phase of the carrier. During each bit interval, the modulator shifts the carrier to one of two possible phases, which are 180 degrees or  $\pi$  radians apart. This can be accomplished very simply by using a bipolar baseband signal to modulate the carrier's amplitude, as shown in Figure 9.18. The output of such a modulator can be represented mathematically as

$$x(t) = R(t) \cos(\omega_c t + \theta) \quad (9.3.1)$$

where  $R(t)$  is the bipolar baseband signal,  $\omega_c$  is the carrier frequency, and  $\theta$  is the phase of the unmodulated carrier. If the output of the modulator is to be represented in complex-envelope form referenced to the carrier frequency, the modulated signal is given as

$$\tilde{x}(t) = I(t) + jQ(t) \quad (9.3.2)$$

where

$$\begin{aligned} I(t) &= R(t) \cos \theta \\ Q(t) &= R(t) \sin \theta \end{aligned}$$

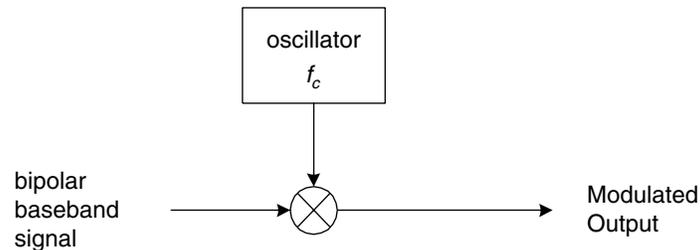
In the special case of  $\theta = 0$ , Eq. (9.3.2) reduces to

$$\tilde{x}(t) = R(t)$$

and the real-valued baseband signal can be used directly as the complex-envelope representation of the modulator output. However, to allow for subsequent phase shifting, the signal's complex-envelope representation should always be implemented as a complex-valued signal. For the special case of  $\theta = 0$ , the imaginary part of the complex signal is simply set to zero.

#### 9.3.1 BPSK Modulator Models

For all but the highest data rates, it is usually sufficient to model the multiplier in Figure 9.18 as ideal, with all of the modulator's nonideal behavior being attributed to degradations of the baseband data waveform. Two different BPSK models are provided on the Web site. The model `BpskBandpassModulator`, summarized in Table 9.8, implements Eq. (9.3.1) to produce a real-valued bandpass output signal. The file `bpsk_mod.cpp` contains the model `BpskModulator`, summarized in Table 9.9, which produces a complex-valued lowpass output signal.



**Figure 9.18** BPSK modulator.

**Table 9.8** Summary of model `BpskBandpassModulator`.

**Constructor:**

```

BpskBandpassModulator( char* instance_nam,
                       PracSimModel *outer_model,
                       Signal<float> *in_sig,
                       Signal<float> *out_sig);
  
```

**Parameter:**

```
double Carrier_Freq;
```

**Notes:**

1. Source code is contained in file `bpskmod_bp.cpp`.

### 9.3.2 BPSK Demodulation

A correlation receiver for BPSK is shown in Figure 9.19. The modulated signal is multiplied by the recovered carrier, and this product is integrated over a bit interval. If the integration result is positive, the received bit is deemed to be 1; if the integration result is negative, the received bit is deemed to be 0. The BPSK demodulator models provided on the Web site include only those functions shown inside the dotted box of the figure. Carrier recovery and clock recovery are provided by separate models that are developed in Chapter 12.

Table 9.10 summarizes the model `BpskBandpassDemod`, which accepts as input a real-valued bandpass input signal. The recovered carrier input to the model is in the form of a real-valued sinusoid, and the recovered clock input to the model is in the form of an integer-valued sequence that has zero values everywhere except

**Table 9.9** Summary of model BpskModulator.

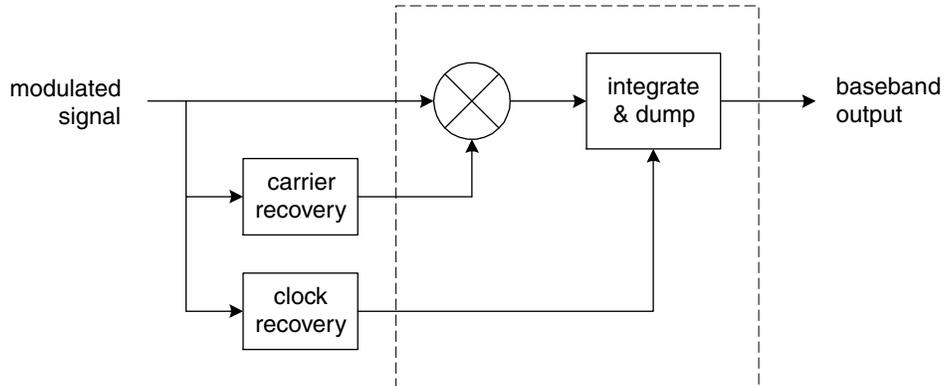
<p><b>Constructor:</b></p> <pre>BpskModulator( char* instance_nam,                 PracSimModel *outer_model,                 Signal&lt;float&gt; *in_sig,                 Signal&lt; complex&lt;float&gt; &gt;                 *cmpx_out_sig);</pre> <hr/> <p><b>Parameter:</b></p> <pre>double Carrier_Phase_Deg;</pre> <hr/> <p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file bpsk_mod.cpp.</li> </ol>
--

at the sampling instants corresponding to the end of each bit interval.

**Table 9.10** Summary of model BpskBandpassDemod.

<p><b>Constructor:</b></p> <pre>BpskBandpassDemod( char* instance_nam,                     PracSimModel *outer_model,                     Signal&lt;float&gt; *in_sig,                     Signal&lt;float&gt; *ref_sig,                     Signal&lt;bit_t&gt; *symb_clock_in,                     Signal&lt;bit_t&gt; *decis_out);</pre> <hr/> <p><b>Parameters:</b></p> <pre>int Samps_Per_Symb; double Dly_To_Start;</pre> <hr/> <p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. This is a multirate model.</li> <li>2. Source code is contained in file bpsk_demod_bp.cpp.</li> </ol>
--

Table 9.11 summarizes the model BpskCorrelationDemod, which accepts as



**Figure 9.19** Correlation receiver for BPSK.

input a complex envelope representation of the modulated signal. The recovered carrier input to the model is in the form of a real-valued signal that represents the instantaneous phase of the recovered carrier. As it is for the bandpass model, the recovered clock input is in the form of an integer-valued sequence that has zero values everywhere except at the sampling instants corresponding to the end of each bit interval.

### 9.3.3 BPSK Simulations

#### 9.3.3.1 Bandpass Simulation

Figure 9.20 shows how the `BpskBandpassModulator` and `BpskBandpassDemodulator` models can be used to simulate a simple BPSK system. A simulation that follows the configuration of Figure 9.20 is provided as the project `BpskBpSim` which uses the main program contained in `bpsk_bp_sim.cpp`.

#### 9.3.3.2 Complex Baseband Simulation

Figure 9.21 shows how the `BpskModulator` and `BpskDemodulator` models can be used to simulate a simple BPSK system in complex-envelope form. The clock-signal input to the demodulator is generated by the `BitsToWave` model. This is analogous to a laboratory situation in which a symbol clock from the transmitter is hardwired to the receiver, bypassing the receiver's normal clock-recovery mechanism. The carrier signal input to the demodulator is generated by the `PhaseRecoveryDaemon` model which generates the appropriate signal after

**Table 9.11** Summary of model BpskCorrelationDemod.

<p><b>Constructor:</b></p> <pre>BpskCorrelationDemod( char* instance_nam,                       PracSimModel *outer_model,                       Signal&lt; complex&lt;float&gt; &gt; *in_sig,                       Signal&lt; complex&lt;float&gt; &gt; *phase_ref_sig,                       Signal&lt;bit_t&gt; *symb_clock_in,                       Signal&lt;bit_t&gt; *decis_out);</pre> <hr/> <p><b>Parameters:</b></p> <pre>int Samps_Per_Symb; double Dly_To_Start;</pre> <hr/> <p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file bpsk_corr_demod.cpp.</li> </ol>
--

reading the modulator's carrier phase from the simulation's parameter input file. More realistic carrier-recovery models are developed in Chapter 12. A simulation that follows the configuration of Figure 9.21 is provided as the project BpskSim, which uses the main program contained in bpsk\_sim.cpp.

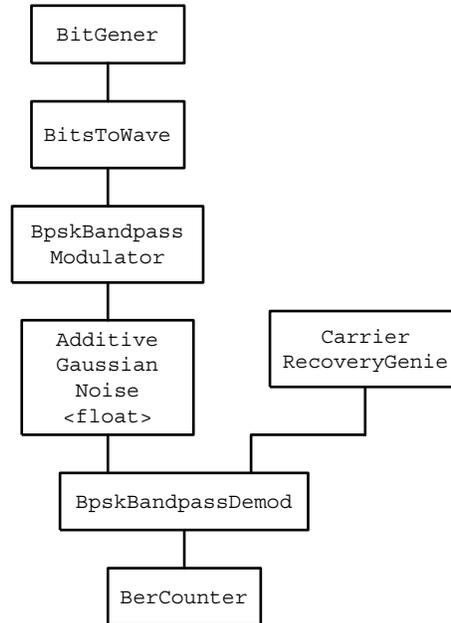
### 9.3.4 Properties of BPSK Signals

The power spectral density of a BPSK signal using rectangular data pulses is given by

$$P_{\text{BPSK}} = \frac{E_b}{2} \left[ \left( \frac{\sin[\pi(f-f_c)T_b]}{\pi(f-f_c)T_b} \right)^2 + \left( \frac{\sin[\pi(f+f_c)T_b]}{\pi(f+f_c)T_b} \right)^2 \right] \quad (9.3.3a)$$

$$= \frac{E_b}{2} [\text{sinc}^2((f-f_c)T_b) + \text{sinc}^2((f+f_c)T_b)] \quad (9.3.3b)$$

$$= \frac{E_b}{2} \left[ \text{sinc}^2\left(\frac{f-f_c}{R_b}\right) + \text{sinc}^2\left(\frac{f+f_c}{R_b}\right) \right] \quad (9.3.3c)$$



**Figure 9.20** Block diagram for bandpass simulation of BPSK.

where

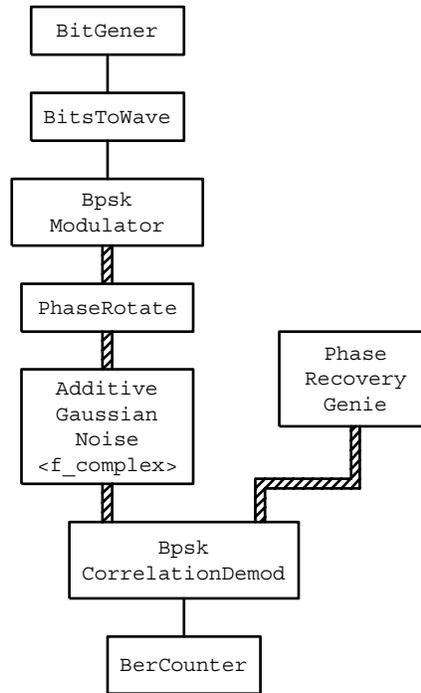
$E_b$  = energy per bit

$T_S$  = bit interval

$R_b = T_b^{-1}$  = bit rate

$f_c$  = carrier frequency

A normalized plot of Eq. (9.3.3) is shown in Figure 9.22. The data for this plot was generated using `BpskPsd`, which can be found in file `bpsk_theory.cpp` in the `utils` directory. An estimated PSD for a complex baseband simulation of a BPSK signal is shown in Figure 9.23. Notice that the sidelobe structure of this one-sided spectrum is nearly identical to the sidelobe structure shown for frequencies above  $f_c$  in Figure 9.22.



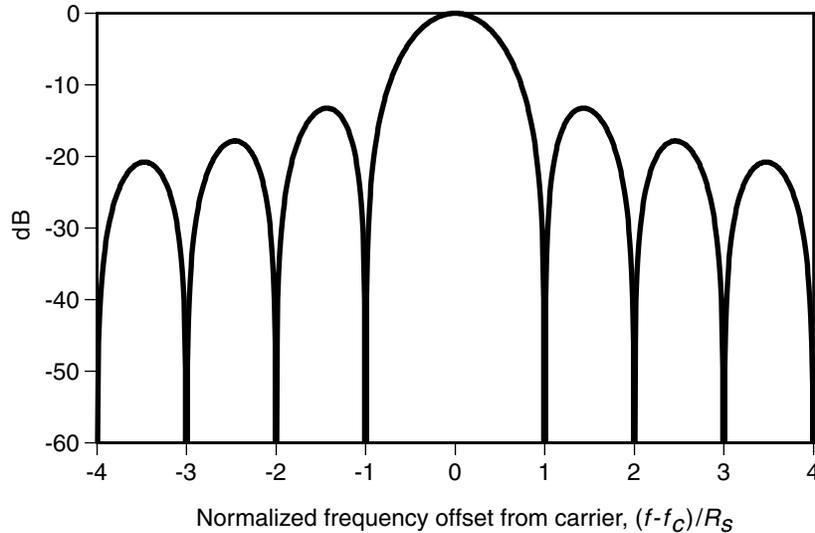
**Figure 9.21** Block diagram for complex-envelope simulation of BPSK.

### 9.3.5 Error Performance

It can be shown (see [25]) that the probability of bit error for BPSK is obtained as

$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = \frac{1}{2} \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right) \quad (9.3.4)$$

where  $Q(x)$  is the  $Q$ -function defined in Appendix G.1. A plot of  $P_b$  is provided in Figure 9.24. The data for this plot was obtained using the function `BpskBer`, which can be found in the file `bpsk_theory.cpp` in the `utils` directory. The three points marked “×” in the figure were obtained via simulation using project `BpskSim`.



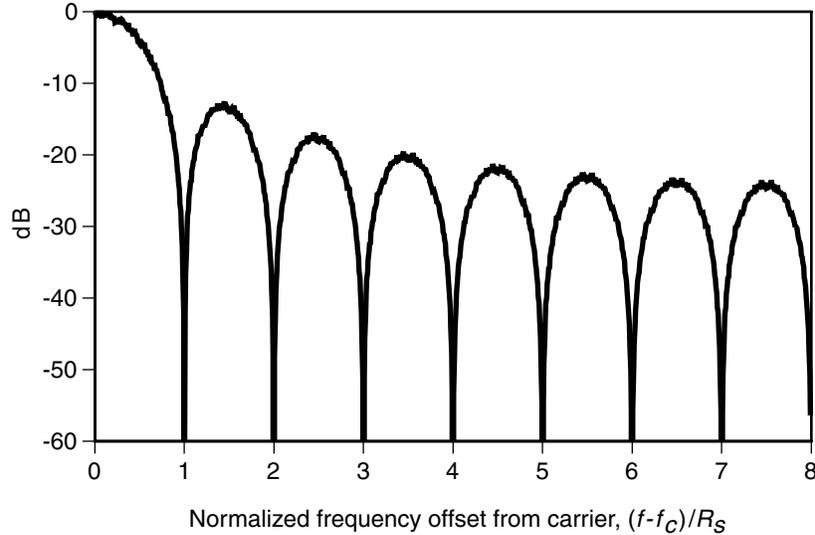
**Figure 9.22** PSD for a BPSK signal using rectangular data pulses.

## 9.4 Multiple Phase Shift Keying

In  $m$ -PSK (or MPSK), the data bits to be modulated are grouped into symbols, each containing  $\log_2 m$  bits, and each symbol can take on one of the  $m$  possible values  $0, 1, \dots, m-1$ . During each symbol interval the modulator shifts the carrier to one of  $m$  possible phases corresponding to the  $m$  possible values of the input symbol. In the ideal case, the phases are each  $360/m$  degrees apart. Typical constellations for 8-PSK and 16-PSK are shown in Figures 9.25 and 9.26. QPSK modulation, discussed in Section 9.2, is really just  $m$ -PSK for  $m = 4$ . QPSK is often considered separately from  $m$ -PSK, because in QPSK modulators and demodulators, the inphase and quadrature components can be processed as independent BPSK signals. Different types of modulators and demodulators are needed for 8-PSK and 16-PSK because I and Q cannot be processed as independent data signals.

### 9.4.1 Ideal $m$ -PSK Modulation and Demodulation

For an 8-PSK signal in an AWGN channel, the optimum receiver is a correlation receiver such as the one shown in Figure 9.27. The I&D filters shown in the figure each integrate over one symbol interval, and the decision device samples the integrated values at the end of this interval. The I&D filters all reset to zero before



**Figure 9.23** PSD estimated from a complex baseband simulation of a BPSK signal using rectangular data pulses.

beginning the integration for the next symbol interval.

A correlation receiver for 8-PSK or 16-PSK is rarely implemented in practice. However, simulation models of correlation receivers are often useful for establishing a baseline against which the performance of other demodulators can be compared. For a bandpass simulation, the model follows directly from the block diagram in Figure 9.27. The model `MpskOptimalBandpassDemod`, summarized in Table 9.12, implements the correlation and decision tasks depicted in the figure. The tasks of carrier recovery and clock recovery are performed by separate models external to the demodulator.

For a complex baseband simulation, the ideal carrier is represented as  $s_0 = 1$ , and the  $m - 1$  phase-shifted reference signals are obtained as

$$s_k = \cos \frac{2k\pi}{m} + j \sin \frac{2k\pi}{m} \quad k = 1, 2, \dots, m - 1 \quad (9.4.1)$$

Note that as defined by Eq. (9.4.1), the reference signals for  $m$ -PSK when  $m = 4$  would have phase angles of 0, 90, 180, and 270 degrees rather than 45, 135, 225, and 315 degrees as defined by Eq. (9.2.5).

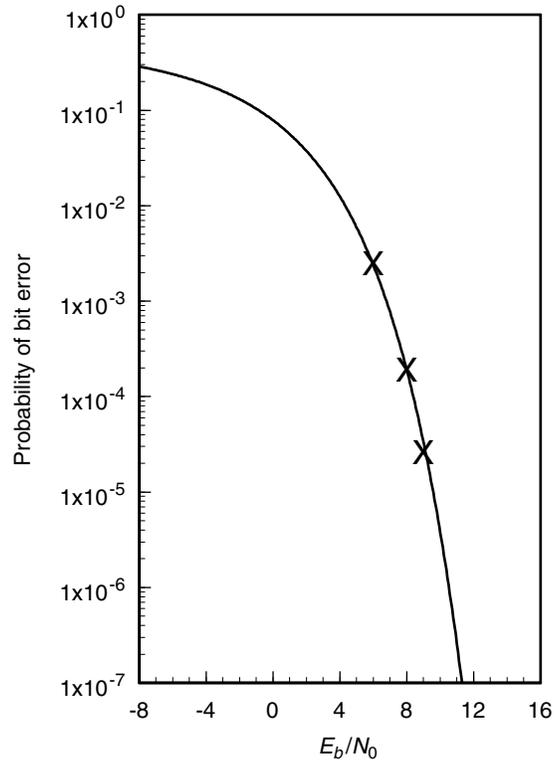


Figure 9.24 Probability of error for BPSK signals.

### 9.4.2 Power Spectral Densities of $m$ -PSK Signals

The PSD of an  $m$ -PSK signal using rectangular data pulses is given by

$$P_{\text{MPSK}} = \frac{E_S}{2} \left[ \text{sinc}^2 \left( \frac{f - f_c}{R_S} \right) + \text{sinc}^2 \left( \frac{f + f_c}{R_S} \right) \right] \quad (9.4.2)$$

Notice that this result does not depend on the value of  $m$ . Also notice that Eq. (9.4.2) is identical to Eq. (9.2.8). For a given symbol rate, the PSDs for QPSK, 8-PSK, and 16-PSK are the same. However, for a given symbol rate, the bit rate for 8-PSK is  $\frac{3}{2}$  times the bit rate for QPSK, and the bit rate for 16-PSK is  $\frac{4}{3}$  times the bit rate for 8-PSK and  $\left(\frac{4}{3}\right) \left(\frac{3}{2}\right) = 2$  times the bit rate for QPSK. The PSDs for  $m$ -PSK signals are typically compared on the basis of a constant bit rate, as shown in Figure 9.28.

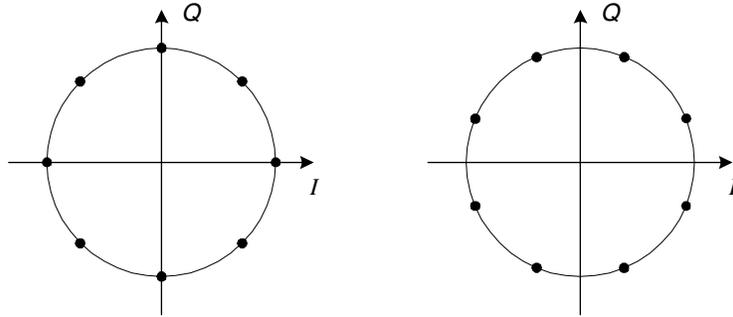


Figure 9.25 Signal constellation for 8-PSK.

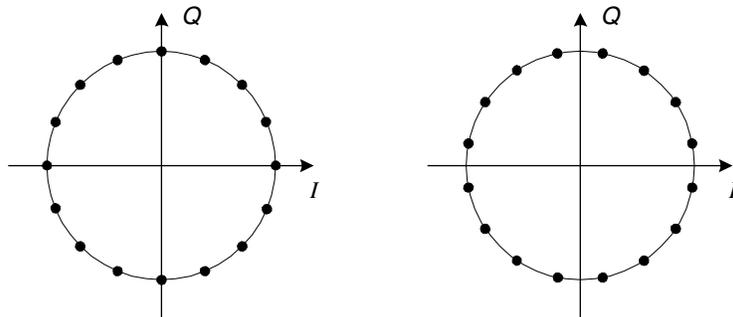
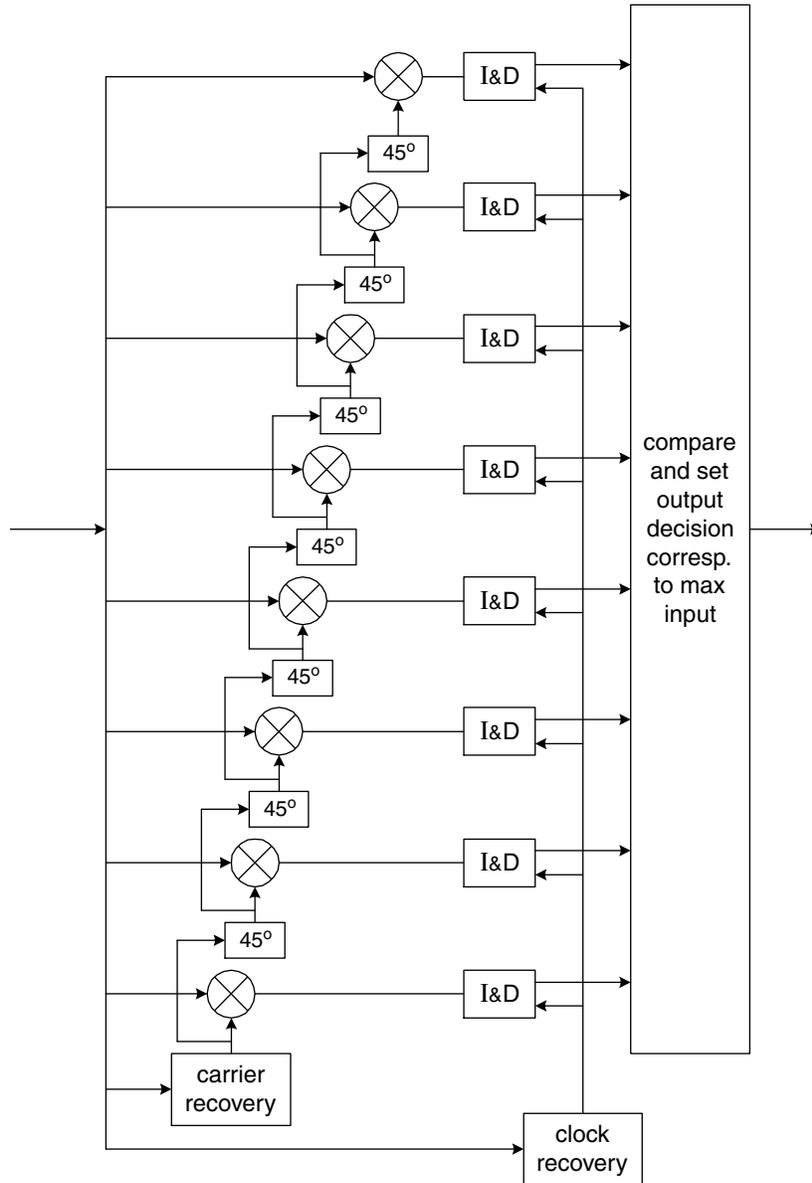


Figure 9.26 Signal constellation for 16-PSK.

Each  $m$ -PSK symbol comprises  $\log_2 m$  bits, so it follows that

$$\begin{aligned}
 E_S &= E_b \log_2 m \\
 T_S &= T_b \log_2 m \\
 P_{\text{MPSK}} &= \frac{E_b \log_2 m}{2} \left[ \text{sinc}^2 \left( \frac{(f - f_c) \log_2 m}{R_b} \right) + \text{sinc}^2 \left( \frac{(f + f_c) \log_2 m}{R_b} \right) \right]
 \end{aligned}$$

The data for Figure 9.28 was generated using the function `M_PskPsd`, which can be found in the file `m_psk_theory.cpp` in the `utils` directory.



**Figure 9.27** Correlation receiver for 8-PSK.

**Table 9.12** Summary of model `MpskOptimalBandpassDemod`.

<p><b>Constructor:</b></p> <pre>MpskOptimalBandpassDemod( char* instance_nam,                           PracSimModel *outer_model,                           Signal&lt;float&gt; *in_sig,                           Signal&lt;bit_t&gt; *symb_clock_in,                           Signal&lt;byte_t&gt; *out_sig);</pre>
<p><b>Parameters:</b></p> <pre>double Phase_Unbal; double Carrier_Freq; int Bits_Per_Symb; int Samps_Per_Symb; double Dly_To_Start;</pre>
<p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file <code>mpskoptdem_bp.cpp</code>.</li> </ol>

### 9.4.3 Error Performance

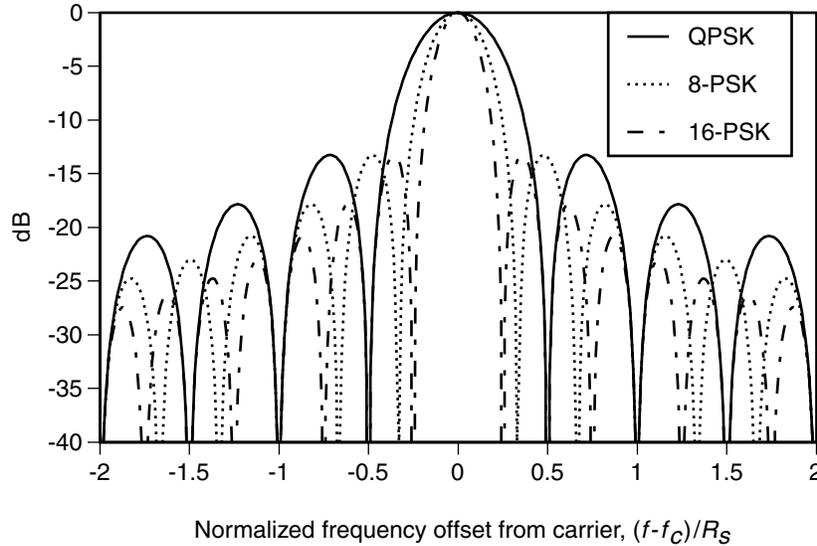
It can be shown that the average symbol error probability for  $m$ -PSK is approximated as

$$P_s \simeq 2Q \left[ \sqrt{\frac{2E_b \log_2 m}{N_0}} \sin \left( \frac{\pi}{m} \right) \right]$$

where  $Q(x)$  is the  $Q$ -function defined in Appendix B, Section B.2. If we assume that Gray coding is employed and that the preponderance of symbol errors will be adjacent-symbol errors, we can approximate the bit-error probability as

$$P_b = \frac{P_s}{\log_2 m}$$

Plots of  $P_s$  for several values of  $m$  are provided in Figure 9.29. The data for these plots was obtained using the function `M_PskBer`, which can be found in file `m_psk_theory.cpp` in the `utils` directory on the Web site. The points marked “×” in the figure were obtained via simulation using rectangular pulses and an ideal demodulator.



**Figure 9.28** PSDs for  $m$ -PSK signals using rectangular data pulses.

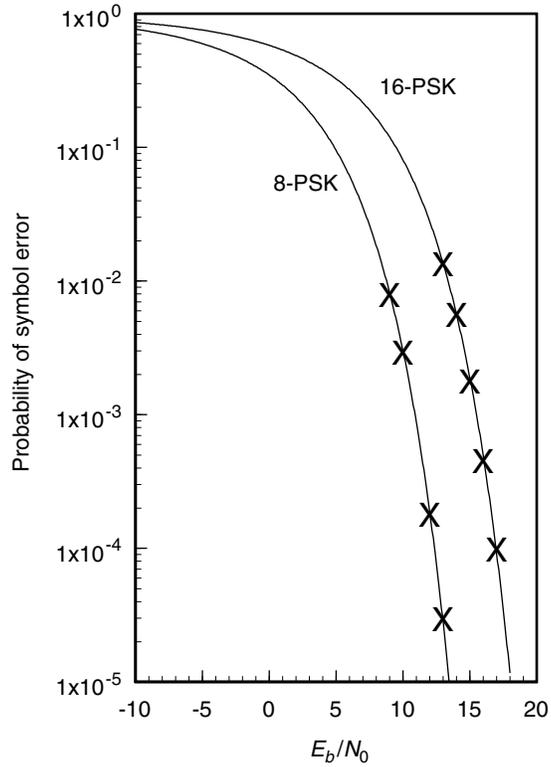
## 9.5 Frequency Shift Keying

In binary FSK, individual data bits are used to control the frequency of the carrier. During each bit interval, the modulator shifts the carrier to one of two possible frequencies depending upon the value of the bit being modulated. There are several different ways that FSK signals can be generated and demodulated. Performance of the system depends upon the particular modulation and demodulation approaches used as well as upon the pair of frequencies employed.

As with other data modulation formats, a correlation receiver provides a convenient baseline for comparison of other FSK demodulation approaches. In general, the FSK signal set can be represented as

$$s_1 = \left(\frac{2E}{T}\right)^{1/2} \sin(\omega_1 t + \phi_1) \quad 0 \leq t \leq T$$

$$s_0 = \left(\frac{2E}{T}\right)^{1/2} \sin(\omega_0 t + \phi_0) \quad 0 \leq t \leq T$$



**Figure 9.29** Probability of symbol error for ideal  $m$ -PSK signals.

The performance of a correlation receiver obviously depends upon the correlation properties of this signal set. The correlation coefficient over a bit interval is given by

$$\begin{aligned} \rho &= \frac{1}{E} \int_0^T \frac{2E}{T} \sin(\omega_1 t + \phi_1) \sin(\omega_0 t + \phi_0) dt \\ &= \frac{1}{T} \int_0^T [\cos[(\omega_1 - \omega_0)t + (\phi_1 - \phi_0)] \\ &\quad - \cos[(\omega_1 + \omega_0)t + (\phi_1 + \phi_0)]] dt \end{aligned}$$

$$\begin{aligned}
&= \cos(\phi_1 - \phi_0) \frac{\sin(\omega_1 - \omega_0)T}{(\omega_1 - \omega_0)T} \\
&\quad + \frac{\sin(\phi_1 - \phi_0)}{(\omega_1 - \omega_0)T} [\cos(\omega_1 - \omega_0)T - 1] \\
&\quad - \cos(\phi_1 + \phi_0) \frac{\sin(\omega_1 + \omega_0)T}{(\omega_1 + \omega_0)T} \\
&\quad - \frac{\sin(\phi_1 + \phi_0)}{(\omega_1 + \omega_0)T} [\cos(\omega_1 + \omega_0)T - 1] \tag{9.5.2}
\end{aligned}$$

The first term in Eq. (9.5.2) vanishes when

$$(\omega_1 - \omega_0)T = n\pi \tag{9.5.3}$$

or when

$$\phi_1 - \phi_0 = \frac{2n + 1}{2}\pi \tag{9.5.4}$$

In these two equations, as well as in the six that follow,  $n$  denotes a positive integer. The second term vanishes when

$$(\omega_1 - \omega_0)T = 2n\pi \tag{9.5.5}$$

or when

$$\phi_1 - \phi_0 = n\pi \tag{9.5.6}$$

The third term vanishes when

$$(\omega_1 + \omega_0)T = n\pi \tag{9.5.7}$$

or when

$$\phi_1 + \phi_0 = \frac{2n + 1}{2}\pi \tag{9.5.8}$$

The fourth term vanishes when

$$(\omega_1 + \omega_0)T = 2n\pi \tag{9.5.9}$$

or when

$$\phi_1 + \phi_0 = n\pi \tag{9.5.10}$$

The FSK signal set is orthogonal only when all four terms in Eq. (9.5.2) vanish. Clearly, when Eq. (9.5.5) is satisfied, Eq. (9.5.3) is also satisfied. Likewise, when Eq. (9.5.9) is satisfied, so is Eq. (9.5.7). Therefore, Eq. (9.5.5) and Eq. (9.5.9) together constitute a sufficient condition for orthogonality. Equations (9.5.5) and (9.5.9) are both satisfied when the two FSK frequencies are chosen such that

$$\frac{\omega_1 - \omega_0}{n} = \frac{2\pi}{T} = \frac{\omega_1 + \omega_0}{m}$$

or

$$\frac{f_1 - f_0}{n} = \frac{1}{T} = \frac{f_1 + f_0}{m}$$

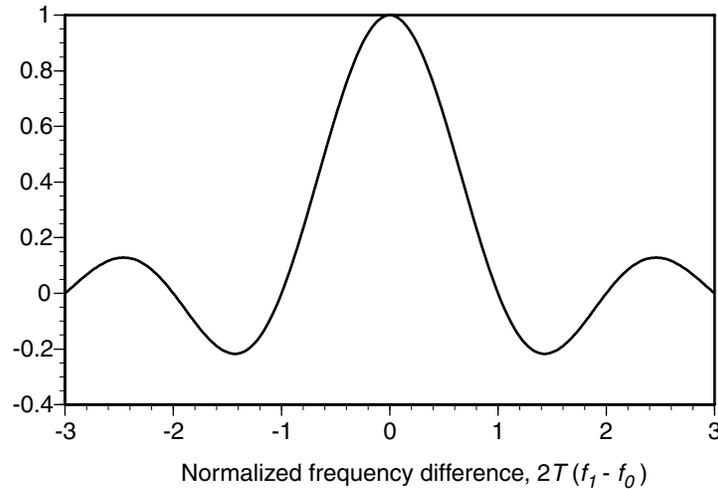
where  $m$  and  $n$  are integers. The phases  $\phi_1$  and  $\phi_0$  can be set to any arbitrary values.

There are other combinations of frequencies and phases that lead to an orthogonal signal set. The first two terms in Eq. (9.5.2) can be made to vanish by selecting values for  $\phi_1$  and  $\phi_0$  that satisfy Eq. (9.5.6) along with values for  $\omega_1$  and  $\omega_0$  that satisfy Eq. (9.5.3) but not Eq. (9.5.5). Likewise, the third and fourth terms can be made to vanish by selecting values for  $\phi_1$  and  $\phi_0$  that satisfy Eq. (9.5.10) along with values for  $\omega_1$  and  $\omega_0$  that satisfy Eq. (9.5.7) but not Eq. (9.5.9). The viable combinations are summarized in Table 9.13.

**Table 9.13** Phase and frequency constraints for orthogonal FSK.

Equations satisfied	Frequency relationship	Phase relationship
(9.5.5) (9.5.9)	$\frac{f_1 - f_0}{n} = \frac{1}{T} = \frac{f_1 + f_0}{m}$	$\phi_0, \phi_1$ arbitrary
(9.5.3) (9.5.6) (9.5.9)	$\frac{f_1 + f_0}{m} = \frac{1}{T} = \frac{2(f_1 - f_0)}{2n + 1}$	$\phi_1 - \phi_0 = k\pi$
(9.5.5) (9.5.7) (9.5.10)	$\frac{f_1 - f_0}{m} = \frac{1}{T} = \frac{2(f_1 + f_0)}{2n + 1}$	$\phi_1 + \phi_0 = k\pi$
(9.5.3) (9.5.6) (9.5.7) (9.5.10)	$\frac{2(f_1 - f_0)}{2n + 1} = \frac{1}{T} = \frac{2(f_1 + f_0)}{2m + 1}$	$\frac{\phi_1 - \phi_0}{j} = \pi = \frac{\phi_1 + \phi_0}{k}$

If  $(\omega_1 + \omega_0) \gg 2\pi/T$  (a condition that is often satisfied in practice), the contributions of terms 3 and 4 in Eq. (9.5.2) become insignificant and the correlation



**Figure 9.30** Correlation coefficient of binary FSK signal set as a function of frequency difference.

coefficient can be approximated by only the first two terms. If it is further assumed that  $\phi_1 = \phi_0$ , the second term vanishes and the expression for the correlation coefficient simplifies to

$$\rho = \frac{\sin(\omega_1 - \omega_0)T}{(\omega_1 - \omega_0)T} = \text{sinc}[2T(f_1 - f_0)] \quad (9.5.11)$$

A plot of Eq. (9.5.11) is provided in Figure 9.30. As shown in the figure, the FSK signals are orthogonal when  $f_1 - f_0$  is an integer multiple of  $(2T)^{-1}$ .

### 9.5.1 FSK Modulators

A binary FSK signal can be generated either by selecting between the outputs of two different oscillators or by switching the frequency of a single oscillator. When the two-oscillator approach is used, the relative phasing of the two outputs can be uncontrolled, or the oscillators can be phase-locked to a common reference so that the relative phasing of their outputs is fixed and known. Sometimes, the baseband data clock is also phase-locked to the same reference so that data transitions can be aligned with zero crossings of the carrier waveforms. In the ideal case using rectangular data waveforms, it is a straightforward task to construct a model of either a single-oscillator or two-oscillator FSK modulator.

Often, an FSK signal set is selected such that for both  $f_1$  and  $f_0$  an integer number of subcarrier cycles will exactly fit in each bit interval  $T$ . In this case, the sinusoidal burst for each bit will begin and end at the same relative phase regardless of the sequence of prior data bits. Coherent reception is facilitated by the fact that once recovered subcarriers are phase-locked to the FSK signal at the receiver, phase-lock can be easily maintained because the relative phasing of the received bursts does not change from bit to bit. In fact, it is not really necessary that an integer number of cycles of  $f_1$  and  $f_0$  fit within a bit interval. Phase continuity can be maintained from bit to bit so long as an integer number of cycles of the difference frequency  $f_1 - f_0$  fit exactly within a bit interval. (This is also a sufficient condition for orthogonality.) Under these conditions, assuming zero initial phase, the single-oscillator modulator and the two-oscillator modulator can both be modeled as a sinusoid that abruptly changes its frequency (but not its phase) at the beginning of each bit interval. The difference between the two modulator designs does not become apparent until the effects of nonideal data waveforms are considered.

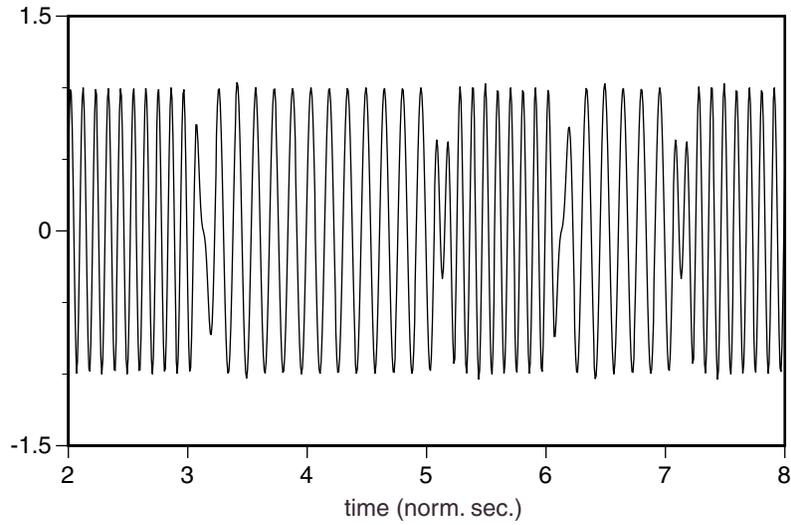
#### 9.5.1.1 Two-Oscillator Modulator

A two-oscillator FSK modulator can be implemented by using the baseband data waveform to gate the outputs from a pair of free-running oscillators. When the data waveform is in a high state, the output from the oscillator at  $f_1$  is enabled, and the output from the oscillator at  $f_0$  is disabled; when the data waveform is in a low state, the output from the oscillator at  $f_0$  is enabled, and the output from the oscillator at  $f_1$  is disabled. As the data waveform slews from a low state to a high state, the amplitude of the  $f_1$  tone increases from zero to full scale as the amplitude of the  $f_0$  tone decreases from full scale to zero. As a result of finite switching times, there will be some glitching in the signal envelope at the bit boundaries as shown in Figure 9.31. The model `FskTwoToneModulator_Bp`, summarized in Table 9.14, implements a two-oscillator FSK modulator.

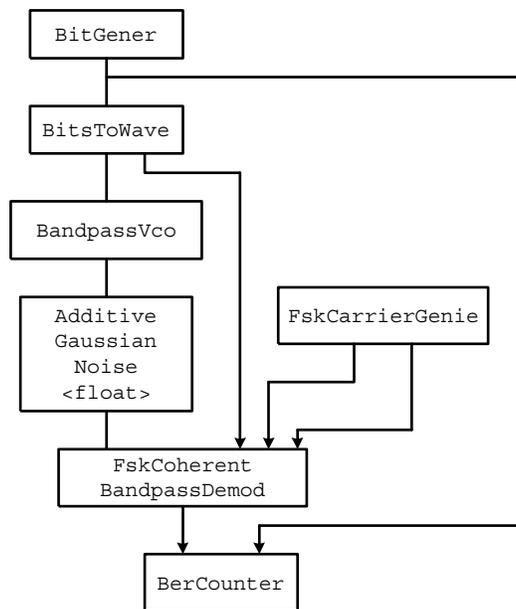
#### 9.5.1.2 Single-Oscillator Modulator

A single-oscillator FSK modulator can be implemented by using the baseband data waveform to drive the control input of a *voltage-controlled oscillator* (VCO). When the data waveform is in a high state, the VCO operates at  $f_1$ , and when the data waveform is in a low state, the VCO operates at  $f_0$ . As the data waveform slews from a low state to a high state, the frequency of the VCO slews from  $f_0$  to  $f_1$ , but the amplitude of VCO output remains approximately constant.

The model `BandpassVco` provided in file `vco_bp.cpp` is an idealized model of a simple VCO. Figure 9.32 shows how the `BandpassVco` model can be used to generate a simulated FSK signal.



**Figure 9.31** FSK signal generated by a two-oscillator modulator.



**Figure 9.32** Generation of an FSK signal using the BandpassVco model.

**Table 9.14** Summary of model `FskTwoToneModulator_Bp`.

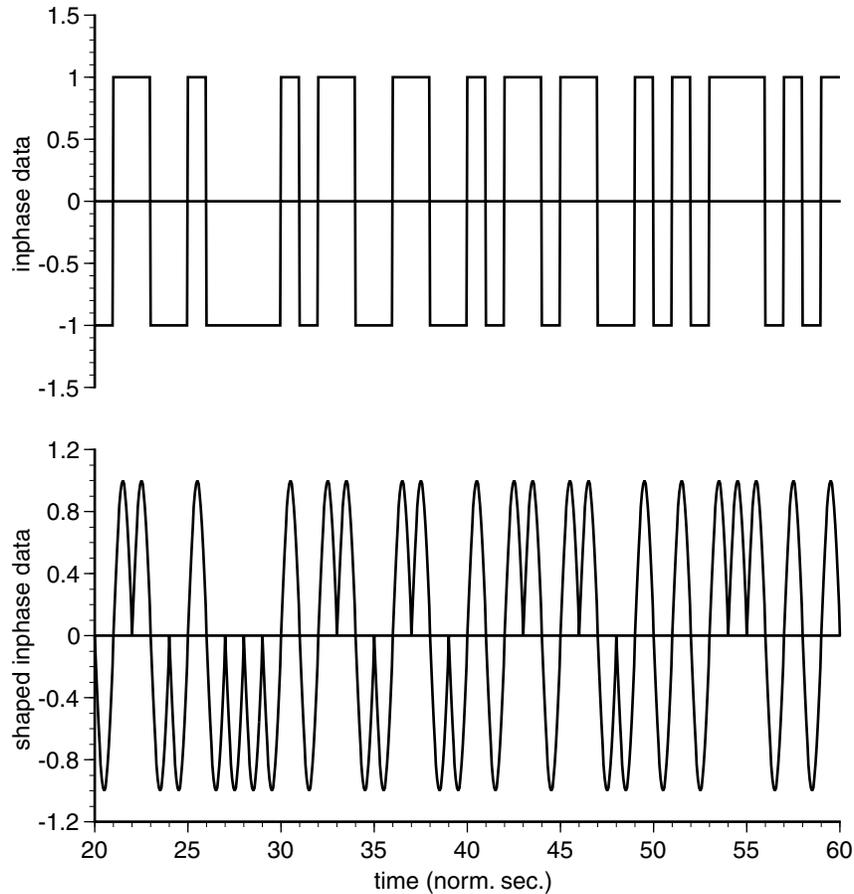
<p><b>Constructor:</b></p> <pre>FskTwoToneModulator_Bp( char* instance_nam,                         PracSimModel *outer_model,                         Signal&lt;float&gt; *in_sig,                         Signal&lt;float&gt; *out_sig);</pre> <hr/> <p><b>Parameters:</b></p> <pre>double Freq_Lo_Hz; double Freq_Hi_Hz; float Lo_Control_Val; float Hi_Control_Val; double Full_Scale_Output_Level;</pre> <hr/> <p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file <code>fsk_2tone_bp.cpp</code>.</li> </ol>
---

## 9.6 Minimum Shift Keying

In Section 9.2.7, it was shown that offsetting the  $I$  and  $Q$  baseband signal inputs to a QPSK modulator eliminates the phase transitions that pass through the origin of the  $I - Q$  plane and therefore eliminates the deep nulls from the envelope of the modulated signal. MSK is another form of modulation that employs a half-bit offset between the  $I$  and  $Q$  baseband signals. In addition, MSK employs sinusoidal shaping of the baseband data pulses. The period of the shaping waveform is set to twice the bit interval so that the pulses are shaped as shown in Figure 9.33. This shaping causes the interstate transition trajectories to follow arcs along a circle in the  $I$ - $Q$  plane as shown in Figure 9.34. The result is an output signal having a constant envelope. This is a very desirable property for signals that will be passed through a nonlinear amplifier.

### 9.6.1 Nonideal Behaviors

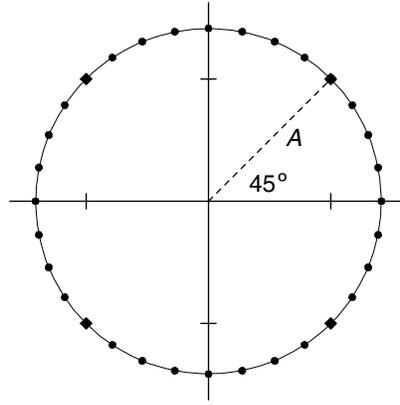
A practical MSK modulator is subject to several sources of nonideal behavior. In addition to all of the degradations described in Section 9.2.1 for QPSK modulators, there are several degradations unique to MSK.



**Figure 9.33** Sinusoidal shaping of data pulses in MSK.

One conceptual approach for implementing an MSK modulator is shown in Figure 9.35. This approach can be viewed as an OQPSK modulator with waveshaping applied to the baseband input signals. If the absolute-value operation shown in the figure is removed, the polarity of the alternate bits is inverted. The correct polarity can be easily restored at the receiver, so practical modulators do not usually implement the absolute value.

When the absolute-value operation is removed, the approach shown in the figure can be viewed as the  $I$  and  $Q$  baseband data signals, each modulating subcarriers at  $(2T_b)^{-1}$  Hz, and these subcarriers subsequently being modulated onto the quadrature



**Figure 9.34** Ideal MSK constellation showing samples along the transition trajectories.

carriers at  $f_0$  Hz. Regardless of how the input shaping operation is viewed, this modulator is subject to the quadrature component unbalances that are discussed in Section 9.2.1.1. Further degradations occur due to imperfect synchronization between the baseband data pulses and the sinusoidal shaping pulses. In any practical MSK modulator design, measures must be incorporated for synchronizing the data waveforms and the shaping waveform. However, in simulation, it is not necessary to model the details of this synchronization mechanism as long as the imperfect nature of the resulting alignment can be adequately characterized and incorporated into the model.

Any useful synchronization strategy must ensure that the average frequency of the subcarrier is exactly half the average bit rates for each of the baseband inputs. If any long-term relative frequency errors between these signals were allowed to exist, there would eventually be disastrous cycle slips between the waveforms. In fact, the desirable properties of the MSK output waveform would be obliterated long before the first cycle slip. Fortunately, it is easy to avoid relative frequency errors by deriving the subcarrier and the two baseband data clocks from the same master oscillator. However, it is not as easy to match the delays in the paths from the master oscillator for the three derived signals. Consequently, there is almost always some relative phase error between the subcarrier(s) and the two baseband data signals. In general, these phase errors each consist of a static component and a dynamic component. In many practical modulator designs, the static component dominates,

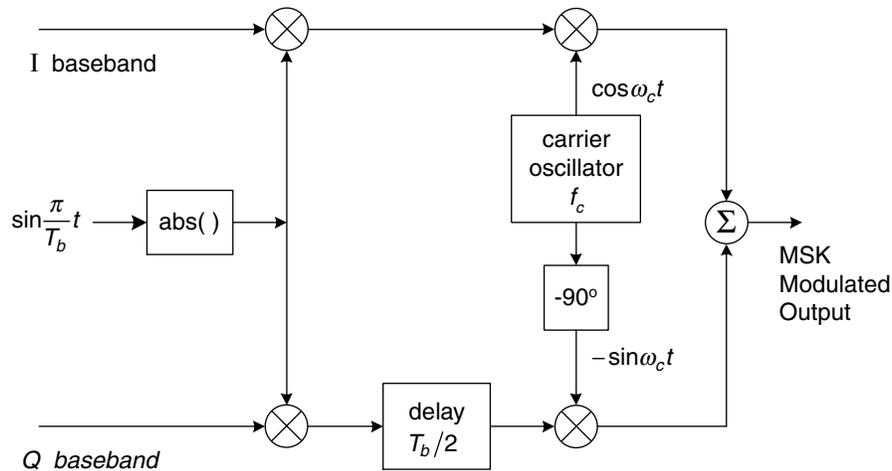


Figure 9.35 Block diagram of MSK modulator.

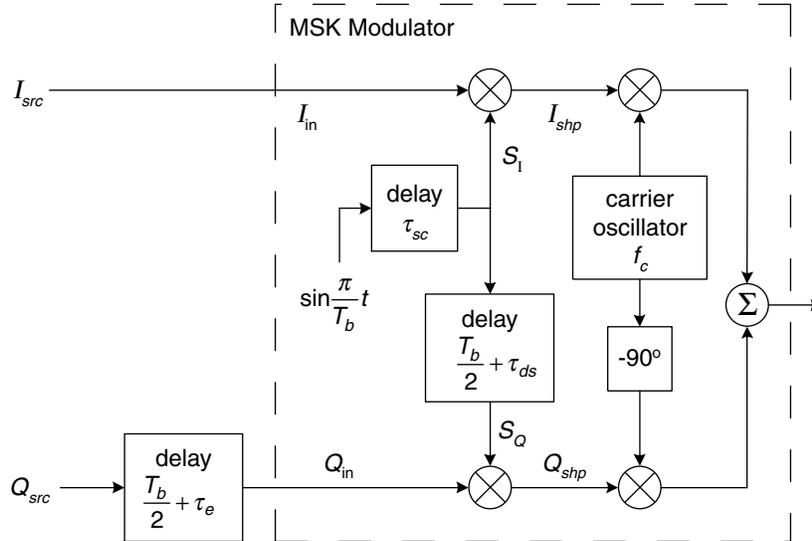
and the dynamic component need not be modeled. The constant-envelope property of MSK has its roots in the trigonometric identity

$$(\pm A \sin x)^2 + (\pm A \cos x)^2 = A^2 \quad (9.6.1)$$

For an ideal MSK signal, each pulse in the  $I$  channel can be viewed as a half-cycle of a sine wave, and each pulse in the  $Q$  channel can be viewed as a half-cycle of a cosine wave. The shaped  $I$  and  $Q$  waveforms are aligned in time such that extrema in the  $I$  waveform coincide with the zero crossings in the  $Q$  waveform, and vice versa. If any unmatched delays disturb this relationship, Eq. (9.6.1) no longer applies, and significant variations are introduced into the envelope of the modulated signal. Therefore, when simulating systems that employ MSK modulators, it is very important that the models incorporate these nonideal delays to avoid erroneously optimistic estimates of system performance.

### 9.6.2 MSK Modulator Models

One approach for modeling the delays in an MSK modulator is shown in Figure 9.36. Assume that the signals  $I_{\text{src}}$ ,  $Q_{\text{src}}$ , and  $\sin(\pi t/T_b)$  are synchronized such that the first bit interval for both the  $I$  waveform and the  $Q$  waveform is  $0 \leq t \leq T_b$ . As shown in the diagram, the  $Q_{\text{src}}$  signal is delayed external to the modulator by an amount equal to the nominal half-bit delay required for MSK plus an error term  $\tau_e$ . The inphase subcarrier  $S_I$  is delayed by  $\tau_{\text{sc}}$  with respect to  $I_{\text{src}}$ . The quadrature



**Figure 9.36** Diagram of MSK modulator showing relative delays between signals.

subcarrier  $S_Q$  is delayed by  $\tau_{sc} + \tau_{ds} - \tau_e$  with respect to either  $Q_{src}$  or  $I_{src}$ . The signal  $Q_{in}$  is delayed by  $\tau_e + T_b/2$  with respect to the signal  $I_{src}$ .

For the modeling approach depicted in the diagram, it can be shown that the complex-envelope representation of the MSK modulated signal is given in terms of  $I_{in}$  and  $Q_{in}$  as

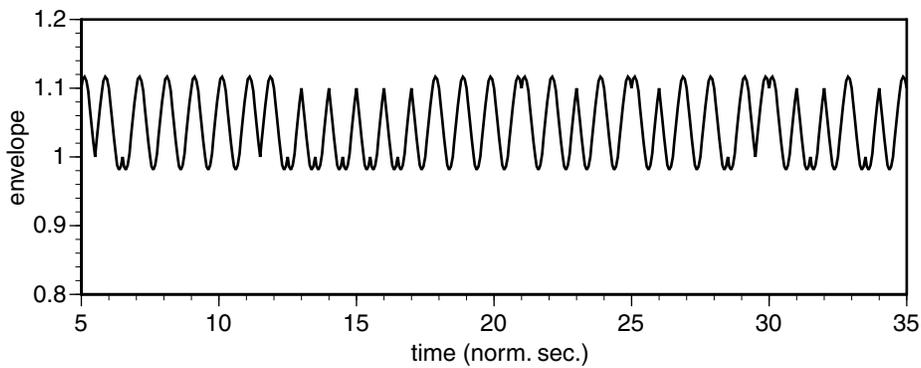
$$I_{out} = I_{in} \sin \left[ \frac{\pi}{T_b} (t - \tau_{sc}) \right] - \beta Q_{in} \sin \phi \cos \left[ \frac{\pi}{T_b} (t - (\tau_{sc} + \tau_{ds})) \right] \quad (9.6.2)$$

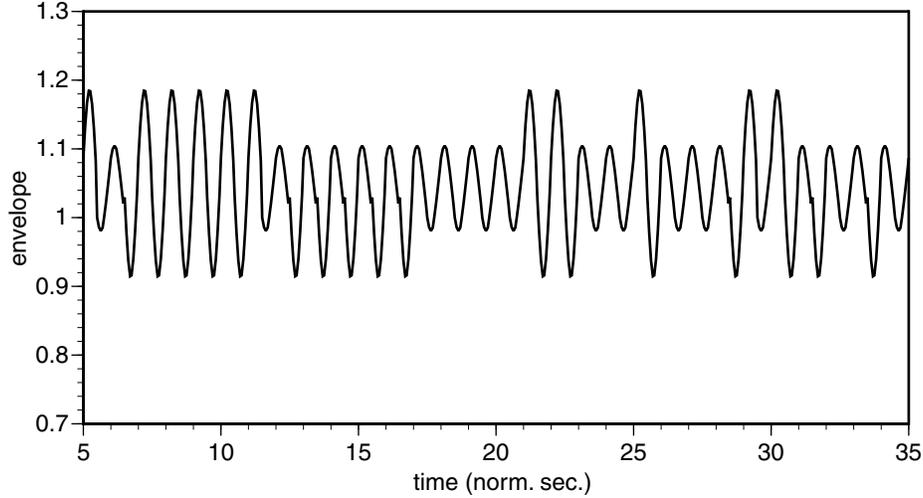
$$Q_{out} = \beta Q_{in} \cos \phi \cos \left[ \frac{\pi}{T_b} (t - (\tau_{sc} + \tau_{ds})) \right] \quad (9.6.3)$$

where  $\beta$  and  $\phi$  are the amplitude and phase unbalances, as defined for QPSK modulators in Section 9.2.1.1. Equations (9.6.2) and (9.6.3) are implemented in the model `MskModulator`, which is summarized in Table 9.15. This model can be used to demonstrate the impacts on the MSK envelope of the amplitude and phase unbalances plus the various misalignment delays. Figure 9.37 shows the output envelope for the case of ideal time alignments with  $\beta = 1.1$  and  $\phi = 5$  deg. The envelope ripples become even more severe, as shown in Figure 9.38, for the case where  $\beta = 1.1$ ,  $\phi = 5$  deg., and  $\tau_{ds} = \tau_e = 0.05$  normalized seconds.

**Table 9.15** Summary of model MskModulator.

<p><b>Constructor:</b></p> <pre>MskModulator( char* instance_nam,               PracSimModel *outer_model,               Signal&lt;float&gt; *i_in_sig,               Signal&lt;float&gt; *q_in_sig,               Signal&lt; complex&lt;float&gt; &gt; *cmpx_out_sig,               Signal&lt;float&gt; *mag_out_sig,               Signal&lt;float&gt; *phase_out_sig);</pre>
<p><b>Parameters:</b></p> <pre>double Bit_Durat; double Data_Skew; double Subcar_Misalign; double Phase_Unbal; double Amp_Unbal; bool Shaping_Is_Bipolar;</pre>
<p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Source code is contained in file mskmod.cpp.</li> </ol>

**Figure 9.37** Envelope variations for an MSK modulator with an amplitude unbalance of 1.1 and a phase unbalance of 5 degrees.



**Figure 9.38** Envelope variations for an MSK modulator with an amplitude unbalance of 1.1, a phase unbalance of 5 degrees, and a data skew of 0.05.

### 9.6.3 Properties of MSK Signals

The power spectral density of an MSK signal is given by

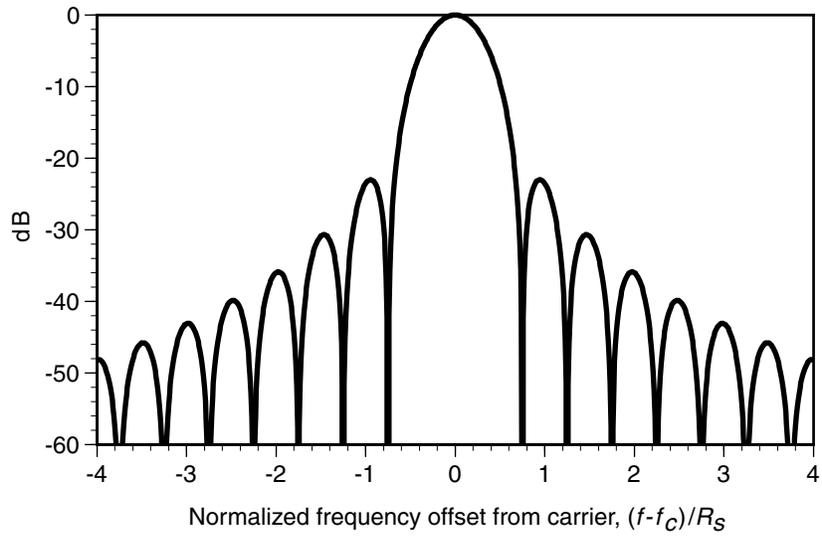
$$P_{MSK} = \frac{8E_b}{\pi^2} \left[ \left( \frac{\cos [2\pi (f - f_c) T_b]}{1 - [4T_b (f - f_c)]^2} \right)^2 + \left( \frac{\cos [2\pi (f + f_c) T_b]}{1 - [4T_b (f + f_c)]^2} \right)^2 \right] \quad (9.6.4)$$

Evaluation of this equation is straightforward except at the values of  $f$  for which the denominators in the two fraction terms become zero. When  $f = f_c \pm (4T_b)^{-1}$ , the denominator of the first term equals zero. For these values of  $f$ , the numerator will also be zero, so we have an indeterminate form of type  $\frac{0}{0}$  that can be evaluated using L'Hospital's rule. Similarly, the second term will be an indeterminate form of type  $\frac{0}{0}$  when  $f = -f_c \pm (4T_b)^{-1}$ . Applying L'Hospital's rule, we obtain

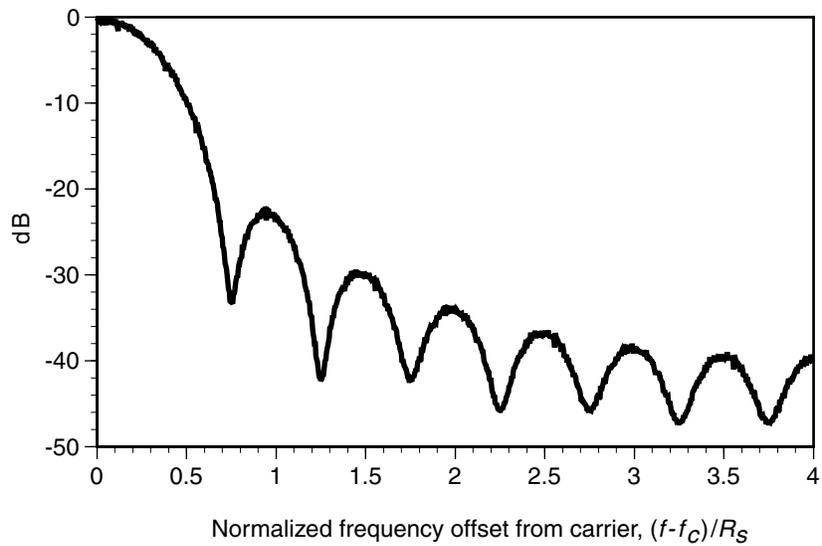
$$\lim_{f \rightarrow f_c \pm (4T_b)^{-1}} \left( \frac{\cos [2\pi (f - f_c) T_b]}{1 - [4T_b (f - f_c)]^2} \right) = \frac{-\pi}{4}$$

$$\lim_{f \rightarrow -f_c \pm (4T_b)^{-1}} \left( \frac{\cos [2\pi (f + f_c) T_b]}{1 - [4T_b (f + f_c)]^2} \right) = \frac{\pi}{4}$$

These values of  $f$  are treated as special cases in the function `MskPsd` that was used to generate the normalized plot of Eq. (9.6.4) shown in Figure 9.39. An estimated PSD for a complex baseband simulation of an MSK signal is shown in Figure 9.40.



**Figure 9.39** PSD for an MSK signal.



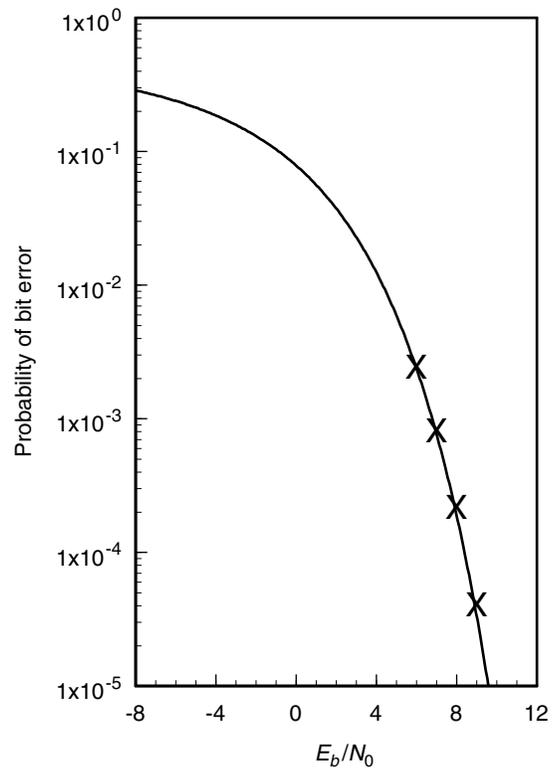
**Figure 9.40** PSD estimated from a complex baseband simulation of an MSK signal.

### 9.6.3.1 Error Performance

The probability of bit error for MSK is the same as for QPSK; that is,

$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = \frac{1}{2} \operatorname{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right)$$

Figure 9.41 contains a plot of  $P_b$  and several estimated bit-error-rate values that were obtained from simulation of an ideal MSK modulator and perfectly synchronized I&D demodulator.



**Figure 9.41** Probability of bit error for MSK signals.