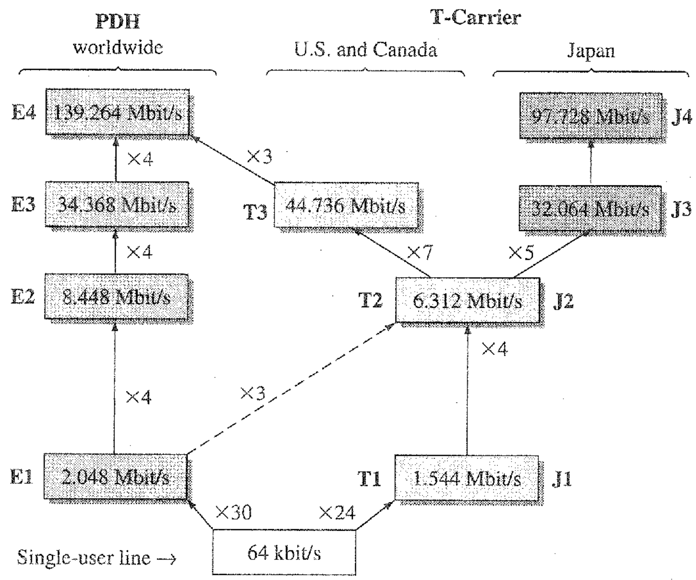

Figure 6.26 Plesiochronous digital hierarchy (PDH) according to ITU-T Recommendation G.704.

6.5 DIFFERENTIAL PULSE CODE MODULATION (DPCM)

PCM is not a very efficient system because it generates so many bits and requires so much bandwidth to transmit. Many different ideas have been proposed to improve the encoding efficiency of A/D conversion. In general, these ideas exploit the characteristics of the source signals. DPCM is one such scheme.

In analog messages we can make a good guess about a sample value from knowledge of past sample values. In other words, the sample values are not independent, and generally there is a great deal of redundancy in the Nyquist samples. Proper exploitation of this redundancy leads to encoding a signal with fewer bits. Consider a simple scheme; instead of transmitting the sample values, we transmit the difference between the successive sample values. Thus, if $m[k]$ is the kth sample, instead of transmitting $m[k]$, we transmit the difference $d[k] = m[k]-m[k-1]$. At the receiver, knowing $d[k]$ and several previous sample value $m[k-1]$, we can reconstruct $m[k]$. Thus, from knowledge of the difference $d[k]$, we can reconstruct $m[k]$ iteratively at the receiver. Now, the difference between successive samples is generally much smaller than the sample values. Thus, the peak amplitude m_p of the transmitted values is reduced considerably. Because the quantization interval $\Delta v = m_p/L$, for a given L (or n), this reduces the quantization interval Δv, thus reducing the quantization noise, which is given by $\Delta v^2/12$. This means that for a given n (or transmission bandwidth), we can increase the SNR, or for a given SNR, we can reduce n (or transmission bandwidth).

We can improve upon this scheme by estimating (predicting) the value of the kth sample $m[k]$ from a knowledge of several previous sample values. If this estimate is $\hat{m}[k]$, then we transmit the difference (prediction error) $d[k] = m[k]-\hat{m}[k]$. At the receiver also, we determine the estimate $\hat{m}[k]$ from the previous sample values, and then generate $m[k]$ by adding the received $d[k]$ to the estimate $\hat{m}[k]$. Thus, we reconstruct the samples at the receiver iteratively. If our prediction is worth its salt, the predicted (estimated) value $\hat{m}[k]$ will be close to $m[k]$, and their difference (prediction error) $d[k]$ will be even smaller than the difference between the successive samples. Consequently, this scheme, known as the **differential PCM (DPCM)**, is superior to the naive prediction described in the preceding paragraph, which is a special case of DPCM, where the estimate of a sample value is taken as the previous sample value, that is, $\hat{m}[k] = m[k-1]$.

Spirits of Taylor, Maclaurin, and Wiener

Before describing DPCM, we shall briefly discuss the approach to signal prediction (estimation). To the uninitiated, future prediction seems like mysterious stuff, fit only for psychics, wizards, mediums, and the like, who can summon help from the spirit world. Electrical engineers appear to be hopelessly outclassed in this pursuit. Not quite so! We can also summon the spirits of Taylor, Maclaurin, Wiener, and the like to help us. What is more, unlike Shakespeare's spirits, our spirits come when called.* Consider, for example, a signal $m(t)$, which has derivatives of all orders at t. Using the Taylor series for this signal, we can express $m(t+T_s)$ as

$$m(t+T_s) = m(t) + T_s\dot{m}(t) + \frac{{T_s}^2}{2!}\ddot{m}(t) + \frac{{T_s}^3}{3!}\dddot{m}(t) + \cdots \tag{6.42a}$$

$$\approx m(t) + T_s\dot{m}(t) \qquad \text{for small } T_s \tag{6.42b}$$

Equation (6.42a) shows that from a knowledge of the signal and its derivatives at instant t, we can predict a future signal value at $t + T_s$. In fact, even if we know just the first derivative, we can still predict this value approximately, as shown in Eq. (6.42b). Let us denote the kth sample of $m(t)$ by $m[k]$, that is, $m(kT_s) = m[k]$, and $m(kT_s \pm T_s) = m[k \pm 1]$, and so on. Setting $t = kT_s$ in Eq. (6.42b), and recognizing that $\dot{m}(kT_s) \approx [m(kT_s) - m(kT_s - T_s)]/T_s$, we obtain

$$\begin{aligned} m[k+1] &\approx m[k] + T_s\left[\frac{m[k]-m[k-1]}{T_s}\right] \\ &= 2m[k] - m[k-1] \end{aligned}$$

This shows that we can find a crude prediction of the $(k+1)$th sample from the two previous samples. The approximation in Eq. (6.42b) improves as we add more terms in the series on the right-hand side. To determine the higher order derivatives in the series, we require more samples in the past. The larger the number of past samples we use, the better will be the prediction. Thus, in general, we can express the prediction formula as

$$m[k] \approx a_1 m[k-1] + a_2 m[k-2] + \cdots + a_N m[k-N] \tag{6.43}$$

The right-hand side is $\hat{m}[k]$, the predicted value of $m[k]$. Thus,

$$\hat{m}[k] = a_1 m[k-1] + a_2 m[k-2] + \cdots + a_N m[k-N] \tag{6.44}$$

This is the equation of an Nth-order predictor. Larger N would result in better prediction in general. The output of this filter (predictor) is $\hat{m}[k]$, the predicted value of $m[k]$. The input consists of the previous samples $m[k-1]$, $m[k-2]$, ..., $m[k-N]$, although it is customary to say that the input is $m[k]$ and the output is $\hat{m}[k]$. Observe that this equation reduces to

* From Shakespeare, Henry IV, Part 1, Act III, Scene 1:
Glendower: *I can call the spirits from vasty deep.*
Hotspur: *Why, so can I, or so can any man;*
But will they come when you do call for them?

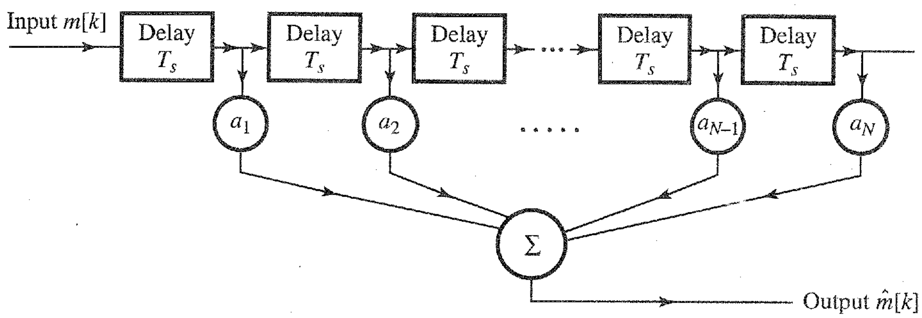

Figure 6.27 Transversal filter (tapped delay line) used as a linear predictor.

$\hat{m}[k] = m[k-1]$ in the case of the first-order prediction. It follows from Eq. (6.42b), where we retain only the first term on the right-hand side. This means that $a_1 = 1$, and the first-order predictor is a simple time delay.

We have outlined here a very simple procedure for predictor design. In a more sophisticated approach, discussed in Sec. 8.5, where we use the minimum mean squared error criterion for best prediction, the **prediction coefficients** a_j in Eq. (6.44) are determined from the statistical correlation between various samples. The predictor described in Eq. (6.44) is called a *linear predictor*. It is basically a transversal filter (a tapped delay line), where the tap gains are set equal to the prediction coefficients, as shown in Fig. 6.27.

Analysis of DPCM

As mentioned earlier, in DPCM we transmit not the present sample $m[k]$, but $d[k]$ (the difference between $m[k]$ and its predicted value $\hat{m}[k]$). At the receiver, we generate $\hat{m}[k]$ from the past sample values to which the received $d[k]$ is added to generate $m[k]$. There is, however, one difficulty associated with this scheme. At the receiver, instead of the past samples $m[k-1]$, $m[k-2]$, ..., as well as $d[k]$, we have their quantized versions $m_q[k-1]$, $m_q[k-2]$, ... Hence, we cannot determine $\hat{m}[k]$. We can determine only $\hat{m}_q[k]$, the estimate of the quantized sample $m_q[k]$, in terms of the quantized samples $m_q[k-1]$, $m_q[k-2]$, ... This will increase the error in reconstruction. In such a case, a better strategy is to determine $\hat{m}_q[k]$, the estimate of $m_q[k]$ (instead of $m[k]$), at the transmitter also from the quantized samples $m_q[k-1]$, $m_q[k-2]$, ... The difference $d[k] = m[k] - \hat{m}_q[k]$ is now transmitted via PCM. At the receiver, we can generate $\hat{m}_q[k]$, and from the received $d[k]$, we can reconstruct $m_q[k]$.

Figure 6.28a shows a DPCM transmitter. We shall soon show that the predictor input is $m_q[k]$. Naturally, its output is $\hat{m}_q[k]$, the predicted value of $m_q[k]$. The difference

$$d[k] = m[k] - \hat{m}_q[k] \tag{6.45}$$

is quantized to yield

$$d_q[k] = d[k] + q[k] \tag{6.46}$$

where $q[k]$ is the quantization error. The predictor output $\hat{m}_q[k]$ is fed back to its input so that the predictor input $m_q[k]$ is

$$\begin{aligned} m_q[k] &= \hat{m}_q[k] + d_q[k] \\ &= m[k] - d[k] + d_q[k] \\ &= m[k] + q[k] \end{aligned} \tag{6.47}$$

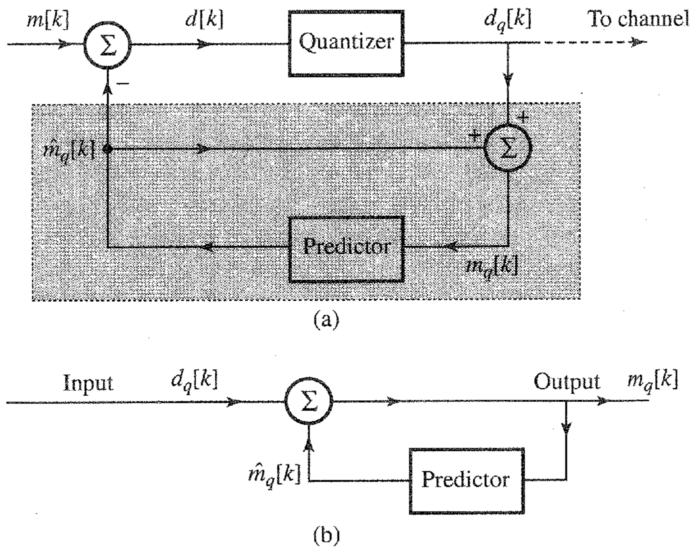

Figure 6.28 DPCM system: (a) transmitter; (b) receiver.

This shows that $m_q[k]$ is a quantized version of $m[k]$. The predictor input is indeed $m_q[k]$, as assumed. The quantized signal $d_q[k]$ is now transmitted over the channel. The receiver shown in Fig. 6.28b is identical to the shaded portion of the transmitter. The inputs in both cases are also the same, namely, $d_q[k]$. Therefore, the predictor output must be $\hat{m}_q[k]$ (the same as the predictor output at the transmitter). Hence, the receiver output (which is the predictor input) is also the same, viz., $m_q[k] = m[k] + q[k]$, as found in Eq. (6.47). This shows that we are able to receive the desired signal $m[k]$ plus the quantization noise $q[k]$. This is the quantization noise associated with the difference signal $d[k]$, which is generally much smaller than $m[k]$. The received samples $m_q[k]$ are decoded and passed through a low-pass filter for D/A conversion.

SNR Improvement

To determine the improvement in DPCM over PCM, let m_p and d_p be the peak amplitudes of $m(t)$ and $d(t)$, respectively. If we use the same value of L in both cases, the quantization step Δv in DPCM is reduced by the factor d_p/m_p. Because the quantization noise power is $(\Delta v)^2/12$, the quantization noise in DPCM is reduced by the factor $(m_p/d_p)^2$, and the SNR is increased by the same factor. Moreover, the signal power is proportional to its peak value squared (assuming other statistical properties invariant). Therefore, G_p (SNR improvement due to prediction) is at least

$$G_p = \frac{P_m}{P_d}$$

where P_m and P_d are the powers of $m(t)$ and $d(t)$, respectively. In terms of decibel units, this means that the SNR increases by $10\log_{10}(P_m/P_d)$ dB. Therefore, Eq. (6.41) applies to DPCM also with a value of α that is higher by $10\log_{10}(P_m/P_d)$ dB. In Example 8.24, a second-order predictor processor for speech signals is analyzed. For this case, the SNR improvement is found to be 5.6 dB. In practice, the SNR improvement may be as high as 25 dB in such cases as short-term voiced speech spectra and in the spectra of low-activity images.[12] Alternately, for the same SNR, the bit rate for DPCM could be lower than that for PCM by 3 to 4 bits per sample. Thus, telephone systems using DPCM can often operate at 32 or even 24 kbit/s.

6.6 ADAPTIVE DIFFERENTIAL PCM (ADPCM)

Adaptive DPCM (ADPCM) can further improve the efficiency of DPCM encoding by incorporating an adaptive quantizer at the encoder. Figure 6.29 illustrates the basic configuration of ADPCM. For practical reasons, the number of quantization level L is fixed. When a fixed quantization step Δv is applied, either the quantization error is too large because Δv is too big or the quantizer cannot cover the necessary signal range when Δv is too small. Therefore, it would be better for the quantization step Δv to be adaptive so that Δv is large or small depending on whether the prediction error for quantizing is large or small.

It is important to note that the quantized prediction error $d_q[k]$ can be a good indicator of the prediction error size. For example, when the quantized prediction error samples vary close to the largest positive value (or the largest negative value), it indicates that the prediction error is large and Δv needs to grow. Conversely, if the quantized samples oscillate near zero, then the prediction error is small and Δv needs to decrease. It is important that both the modulator and the receiver have access to the same quantized samples. Hence, the adaptive quantizer and the receiver reconstruction can apply the same algorithm to adjust the Δv identically.

Compared with DPCM, ADPCM can further compress the number of bits needed for a signal waveform. For example, it is very common in practice for an 8-bit PCM sequence to be encoded into a 4-bit ADPCM sequence at the same sampling rate. This easily represents a 2:1 bandwidth or storage reduction with virtually no loss.

ADPCM encoder has many practical applications. The ITU-T standard G.726 specifies an ADPCM speech coder and decoder (called **codec**) for speech signal samples at 8 kHz.[7] The G.726 ADPCM predictor uses an eighth-order predictor. For different quality levels, G.726 specifies four different ADPCM rates at 16, 24, 32, and 40 kbit/s. They correspond to four different bit sizes for each speech sample at 2 bits, 3 bits, 4 bits, and 5 bits, respectively, or equivalently, quantization levels of 4, 8, 16, and 32, respectively.

The most common ADPCM speech encoders use 32 kbit/s. In practice, there are multiple variations of ADPCM speech codec. In addition to the ITU-T G.726 specification,[7] these include the OKI ADPCM codec, the Microsoft ADPCM codec supported by WAVE players, and the Interactive Multimedia Association (IMA) ADPCM, also known as the DVI ADPCM. The 32 kbit/s ITU-T G.726 ADPCM speech codec is widely used in the DECT (digital enhanced cordless telecommunications) system, which itself is widely used for residential and business cordless phone communications. Designed for short-range use as an access mechanism to the main networks, DECT offers cordless voice, fax, data, and multimedia communications. DECT is now in use in over 100 countries worldwide. Another major user of the 32 kbit/s ADPCM codec is the Personal Handy-phone System (or PHS), also marketed as the Personal Access System (PAS) and known as Xiaolingtong in China.

PHS is a mobile network system similar to a cellular network, operating in the 1880 to 1930 MHz frequency band, used mainly in Japan, China, Taiwan, and elsewhere in Asia. Originally developed by the NTT Laboratory in Japan in 1989, PHS is much simpler to implement and

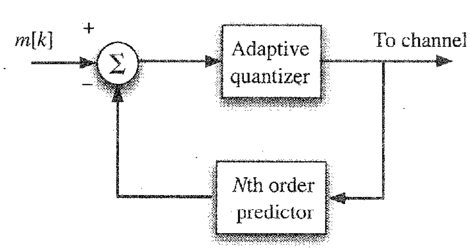

Figure 6.29 ADPCM encoder uses an adaptive quantizer controlled only by the encoder output bits.

deploy. Unlike cellular networks, PHS phones and base stations are low-power, short-range facilities. The service is often pejoratively called the "poor man's cellular" because of its limited range and poor roaming ability. PHS first saw limited deployment (NTT-Personal, DDI-Pocket, and ASTEL) in Japan in 1995 but has since nearly disappeared. Surprisingly, PHS has seen a resurgence in markets like China, Taiwan, Vietnam, Bangladesh, Nigeria, Mali, Tanzania, and Honduras, where its low cost of deployment and hardware costs offset the system's disadvantages. In China alone, there was an explosive expansion of subscribers, reaching nearly 80 million in 2006.

6.7 DELTA MODULATION

Sample correlation used in DPCM is further exploited in **delta modulation (DM)** by oversampling (typically four times the Nyquist rate) the baseband signal. This increases the correlation between adjacent samples, which results in a small prediction error that can be encoded using only one bit ($L = 2$). Thus, DM is basically a 1-bit DPCM, that is, a DPCM that uses only two levels ($L = 2$) for quantization of $m[k] - \hat{m}_q[k]$. In comparison to PCM (and DPCM), it is a very simple and inexpensive method of A/D conversion. A 1-bit codeword in DM makes word framing unnecessary at the transmitter and the receiver. This strategy allows us to use fewer bits per sample for encoding a baseband signal.

In DM, we use a first-order predictor, which, as seen earlier, is just a time delay of T_s (the sampling interval). Thus, the DM transmitter (modulator) and receiver (demodulator) are identical to those of the DPCM in Fig. 6.28, with a time delay for the predictor, as shown in Fig. 6.30, from which we can write

$$m_q[k] = m_q[k-1] + d_q[k] \tag{6.48}$$

Hence,

$$m_q[k-1] = m_q[k-2] + d_q[k-1]$$

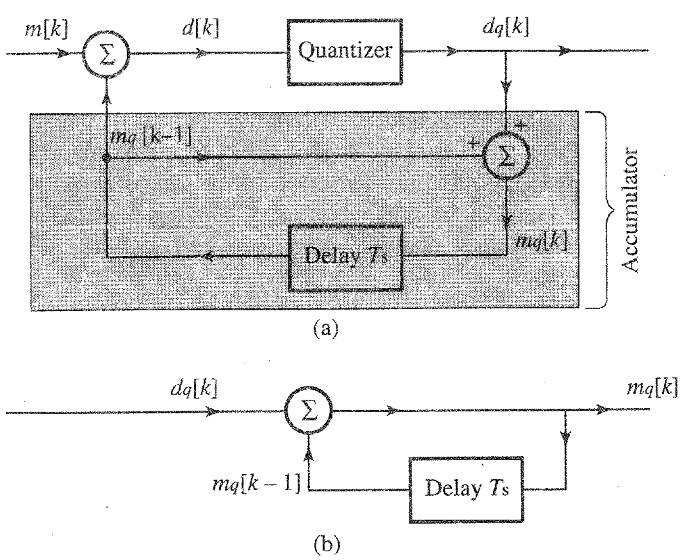

Figure 6.30 Delta modulation is a special case of DPCM.

Substituting this equation into Eq. (6.48) yields

$$m_q[k] = m_q[k-2] + d_q[k] + d_q[k-1]$$

Proceeding iteratively in this manner, and assuming zero initial condition, that is, $m_q[0] = 0$, we write

$$m_q[k] = \sum_{m=0}^{k} d_q[m] \tag{6.49}$$

This shows that the receiver (demodulator) is just an accumulator (adder). If the output $d_q[k]$ is represented by impulses, then the accumulator (receiver) may be realized by an integrator because its output is the sum of the strengths of the input impulses (sum of the areas under the impulses). We may also replace with an integrator the feedback portion of the modulator (which is identical to the demodulator). The demodulator output is $m_q[k]$, which when passed through a low-pass filter yields the desired signal reconstructed from the quantized samples.

Figure 6.31 shows a practical implementation of the delta modulator and demodulator. As discussed earlier, the first-order predictor is replaced by a low-cost integrator circuit (such as an *RC* integrator). The modulator (Fig. 6.31a) consists of a comparator and a sampler in the direct path and an integrator-amplifier in the feedback path. Let us see how this delta modulator works.

The analog signal $m(t)$ is compared with the feedback signal (which serves as a predicted signal) $\hat{m}_q(t)$. The error signal $d(t) = m(t) - \hat{m}_q(t)$ is applied to a comparator. If $d(t)$ is positive, the comparator output is a constant signal of amplitude E, and if $d(t)$ is negative, the comparator output is $-E$. Thus, the difference is a binary signal ($L = 2$) that is needed to generate a 1-bit DPCM. The comparator output is sampled by a sampler at a rate of f_s samples per second, where f_s is typically much higher than the Nyquist rate. The sampler thus produces a train of narrow pulses $d_q[k]$ (to simulate impulses) with a positive pulse when $m(t) > \hat{m}_q(t)$ and a negative pulse when $m(t) < \hat{m}_q(t)$. Note that each sample is coded by a single binary pulse (1-bit DPCM), as required. The pulse train $d_q[k]$ is the delta-modulated pulse train (Fig. 6.31d). The modulated signal $d_q[k]$ is amplified and integrated in the feedback path to generate $\hat{m}_q(t)$ (Fig. 6.31c), which tries to follow $m(t)$.

To understand how this works, we note that each pulse in $d_q[k]$ at the input of the integrator gives rise to a step function (positive or negative, depending on the pulse polarity) in $\hat{m}_q(t)$. If, for example, $m(t) > \hat{m}_q(t)$, a positive pulse is generated in $d_q[k]$, which gives rise to a positive step in $\hat{m}_q(t)$, trying to equalize $\hat{m}_q(t)$ to $m(t)$ in small steps at every sampling instant, as shown in Fig. 6.31c. It can be seen that $\hat{m}_q(t)$ is a kind of staircase approximation of $m(t)$. When $\hat{m}_q(t)$ is passed through a low-pass filter, the coarseness of the staircase in $\hat{m}_q(t)$ is eliminated, and we get a smoother and better approximation to $m(t)$. The demodulator at the receiver consists of an amplifier-integrator (identical to that in the feedback path of the modulator) followed by a low-pass filter (Fig. 6.31b).

DM Transmits the Derivative of $m(t)$

In PCM, the analog signal samples are quantized in L levels, and this information is transmitted by n pulses per sample ($n = \log_2 L$). A little reflection shows that in DM, the modulated signal carries information not about the signal samples but about the difference between successive samples. If the difference is positive or negative, a positive or a negative pulse (respectively) is generated in the modulated signal $d_q[k]$. Basically, therefore, DM carries the information about the derivative of $m(t)$, hence, the name "delta modulation." This can also be seen from

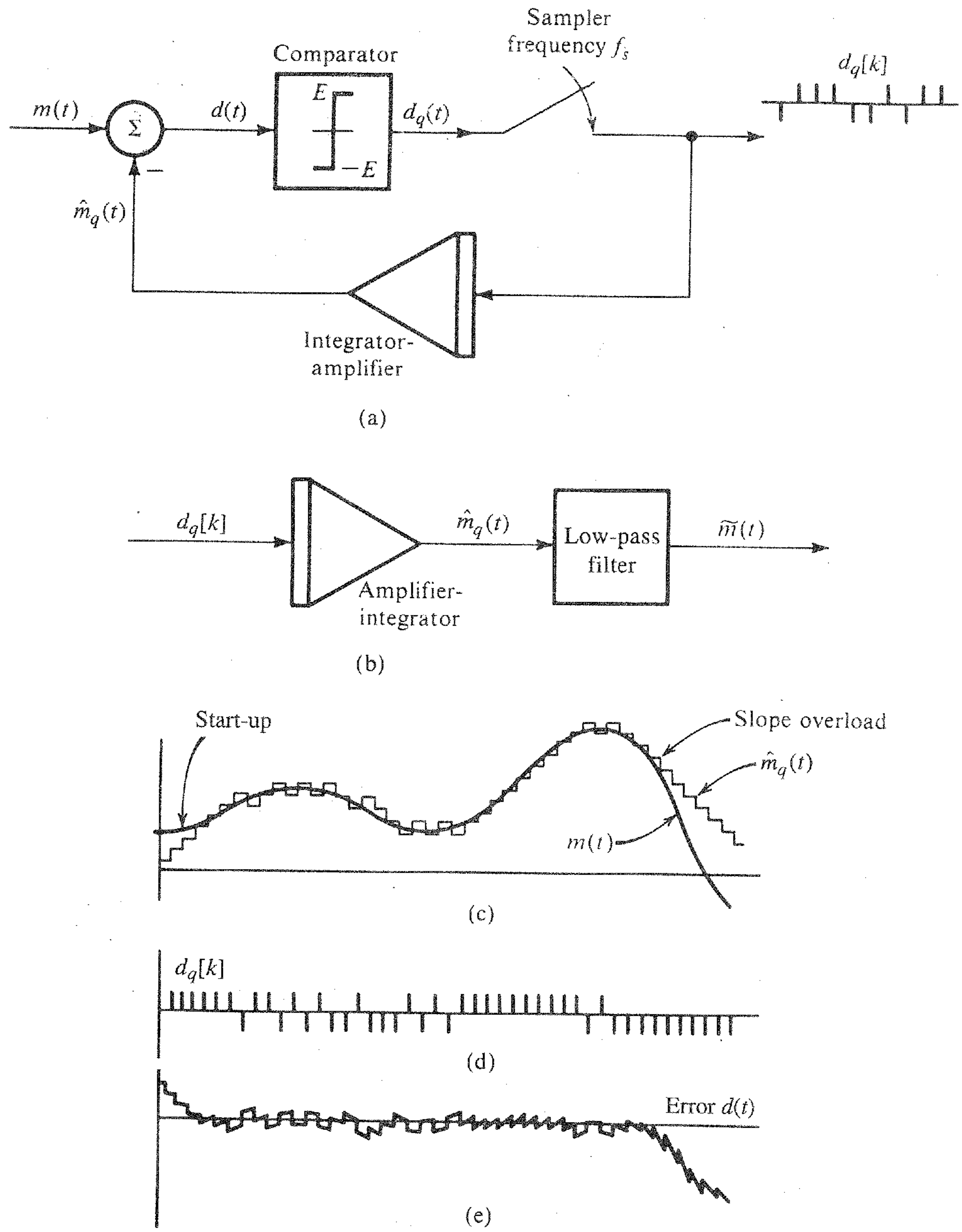

Figure 6.31 Delta modulation: (a) and (b) delta demodulators; (c) message signal versus integrator output signal (d) delta-modulated pulse trains; (e) modulation errors.

the fact that integration of the delta-modulated signal yields $\hat{m}_q(t)$, which is an approximation of $m(t)$.

In PCM, the information of each quantized sample is transmitted by an n-bit code word, whereas in DM the information of the difference between successive samples is transmitted by a 1-bit code word.

Threshold of Coding and Overloading

Threshold and overloading effects can be clearly seen in Fig. 6.31c. Variations in $m(t)$ smaller than the step value (threshold of coding) are lost in DM. Moreover, if $m(t)$ changes too fast,