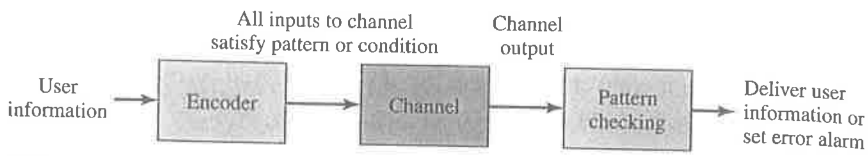

FIGURE 3.60 General error-detection system.

This code is an example of the so-called linear codes because the parity bit is calculated as the modulo 2 sum of the information bits:

$$b_{k+1} = b_1 + b_2 + \cdots + b_k \quad \text{modulo } 2 \tag{3.42}$$

where $b_1, b_2, \ldots, b_k$ are the information bits.

Recall that in modulo 2 arithmetic $0+0=0$, $0+1=1$, $1+0=1$, and $1+1=0$. Thus, if the information bits contain an even number of 1s, then the parity bit will be 0; and if they contain an odd number, then the parity bit will be 1. Consequently, the above rule will assign the parity bit a value that will produce a *codeword* that *always contains an even number of 1s*. This pattern defines the single parity check code.

If a codeword undergoes a single error during transmission, then the corresponding binary block at the output of the channel will contain an odd number of 1s and the error will be detected. More generally, if the codeword undergoes an odd number of errors, the corresponding output block will also contain an odd number of 1s. Therefore, the single parity bit allows us to detect all error patterns that introduce an odd number of errors. On the other hand, the single parity bit will fail to detect any error patterns that introduce an even number of errors, since the resulting binary vector will have even parity. Nonetheless, the single parity bit provides a remarkable amount of error-detection capability, since the addition of a single check bit results in making half of all possible error patterns detectable, regardless of the value of k.

Figure 3.61 shows an alternative way of looking at the operation of this example. At the transmitter a checksum is calculated from the information bits and transmitted along with the information. At the receiver, the checksum is recalculated, based on the received information. The received and recalculated checksums are compared, and the error alarm is set if they disagree.

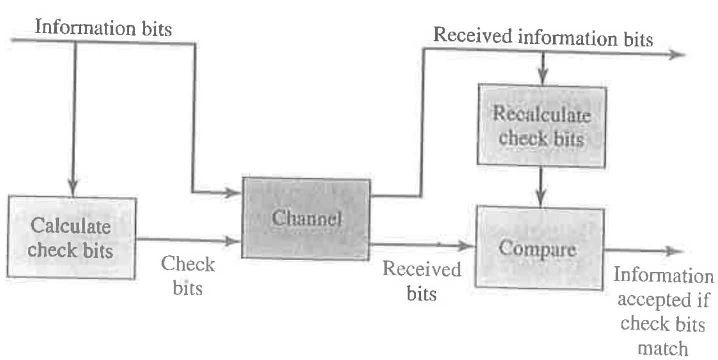

FIGURE 3.61 Error-detection system using check bits.