Generator polynomial: $g(x) = x^3 + x + 1$
Information: $(1,1,0,0) \rightarrow i(x) = x^3 + x^2$
Encoding: $x^3 i(x) = x^6 + x^5$

$$
\begin{array}{r|l}
 & x^3 + x^2 + x \\
\hline
x^3 + x + 1 & x^6 + x^5 \\
 & x^6 + \quad x^4 + x^3 \\
\hline
 & \quad x^5 + x^4 + x^3 \\
 & \quad x^5 + \quad x^3 + x^2 \\
\hline
 & \quad\quad x^4 + \quad x^2 \\
 & \quad\quad x^4 + \quad x^2 + x \\
\hline
 & \quad\quad\quad\quad x
\end{array}
\qquad
\begin{array}{r|l}
 & 1110 \\
\hline
1011 & 1100000 \\
 & 1011 \\
\hline
 & \ 1110 \\
 & \ 1011 \\
\hline
 & \ \ 1010 \\
 & \ \ 1011 \\
\hline
 & \ \ \ 010
\end{array}
$$

Transmitted codeword:
$b(x) = x^6 + x^5 + x$
$\rightarrow \underline{b} = (1,1,0,0,0,1,0)$

FIGURE 3.68 Example of CRC encoding.

The final step in the encoding procedure obtains the binary codeword $b(x)$ by adding the remainder $r(x)$ from $x^{n-k}i(x)$:

$$b(x) = x^{n-k}i(x) + r(x). \tag{3.53}$$

Because the divisor $g(x)$ had degree $n-k$, the remainder $r(x)$ can have maximum degree $n-k-1$ or lower. Therefore $r(x)$ has at most $n-k$ terms. In terms of the previously introduced register of length n, $r(x)$ will occupy the lower $n-k$ positions. Recall that the upper k positions were occupied by the information bits. We thus see that this encoding process introduces a binary polynomial in which the k higher-order terms are the information bits and in which the $n-k$ lower-order terms are the cyclic redundancy check bits. In the example in Figure 3.68, the division of $x^3 i(x)$ by $g(x)$ gives the remainder polynomial $r(x) = x$. The codeword polynomial is then $x^6 + x^5 + x$, which corresponds to the binary codeword (1,1,0,0,0,1,0). Note how the first four positions contain the original four information bits and how the lower three positions contains the CRC bits.

In Figure 3.66 we showed that in normal division dividing 122 by 35 yields a quotient of 3 and a remainder of 17. This result implies that $122 = 3(35) + 17$. Note that by subtracting the remainder 17 from both sides, we obtain $122 - 17 = 3(35)$ so that $122 - 17$ is evenly divisible by 35. Similarly, the codeword polynomial $b(x)$ is divisible by $g(x)$ because

$$b(x) = x^{n-k}i(x) + r(x) = g(x)q(x) + r(x) + r(x) = g(x)q(x) \tag{3.54}$$

where we have used the fact that in modulo 2 arithmetic $r(x) + r(x) = 0$. Equation (3.54) implies that *all codewords are multiples of the generator polynomial* $g(x)$. This is the pattern that must be checked by the receiver. *The receiver can check to see whether the pattern is satisfied by dividing the received polynomial by* $g(x)$. *If the remainder is nonzero, then an error has been detected.*

The Euclidean Division algorithm can be implemented using a feedback shift-register circuit that implements division. The feedback taps in this circuit are determined by the coefficients of the generator polynomial. Figure 3.69 shows the division

Generator polynomial: $g(x) = x^3 + x + 1$
Information: $(1,1,0,0) \rightarrow i(x) = x^3 + x^2$
Encoding: $x^3 i(x) = x^6 + x^5$

$$
\begin{array}{r|l}
 & x^3 + x^2 + x \\ \hline
x^3 + x + 1 & x^6 + x^5 \\
 & x^6 + x^4 + x^3 \\ \hline
 & x^5 + x^4 + x^3 \\
 & x^5 + x^3 + x^2 \\ \hline
 & x^4 + x^2 \\
 & x^4 + x^2 + x \\ \hline
 & x
\end{array}
\qquad
\begin{array}{r|r}
 & 1110 \\ \hline
1011 & 1100000 \\
 & 1011 \\ \hline
 & 1110 \\
 & 1011 \\ \hline
 & 1010 \\
 & 1011 \\ \hline
 & 010
\end{array}
$$

Transmitted codeword:
$b(x) = x^6 + x^5 + x$
$\rightarrow \underline{b} = (1,1,0,0,0,1,0)$

FIGURE 3.68 Example of CRC encoding.

The final step in the encoding procedure obtains the binary codeword $b(x)$ by adding the remainder $r(x)$ from $x^{n-k}i(x)$:

$$b(x) = x^{n-k}i(x) + r(x). \tag{3.53}$$

Because the divisor $g(x)$ had degree $n-k$, the remainder $r(x)$ can have maximum degree $n-k-1$ or lower. Therefore $r(x)$ has at most $n-k$ terms. In terms of the previously introduced register of length n, $r(x)$ will occupy the lower $n-k$ positions. Recall that the upper k positions were occupied by the information bits. We thus see that this encoding process introduces a binary polynomial in which the k higher-order terms are the information bits and in which the $n-k$ lower-order terms are the cyclic redundancy check bits. In the example in Figure 3.68, the division of $x^3 i(x)$ by $g(x)$ gives the remainder polynomial $r(x) = x$. The codeword polynomial is then $x^6 + x^5 + x$, which corresponds to the binary codeword (1,1,0,0,0,1,0). Note how the first four positions contain the original four information bits and how the lower three positions contains the CRC bits.

In Figure 3.66 we showed that in normal division dividing 122 by 35 yields a quotient of 3 and a remainder of 17. This result implies that $122 = 3(35) + 17$. Note that by subtracting the remainder 17 from both sides, we obtain $122 - 17 = 3(35)$ so that $122 - 17$ is evenly divisible by 35. Similarly, the codeword polynomial $b(x)$ is divisible by $g(x)$ because

$$b(x) = x^{n-k}i(x) + r(x) = g(x)q(x) + r(x) + r(x) = g(x)q(x) \tag{3.54}$$

where we have used the fact that in modulo 2 arithmetic $r(x) + r(x) = 0$. Equation (3.54) implies that *all codewords are multiples of the generator polynomial $g(x)$*. This is the pattern that must be checked by the receiver. *The receiver can check to see whether the pattern is satisfied by dividing the received polynomial by $g(x)$. If the remainder is nonzero, then an error has been detected.*

The Euclidean Division algorithm can be implemented using a feedback shift-register circuit that implements division. The feedback taps in this circuit are determined by the coefficients of the generator polynomial. Figure 3.69 shows the division