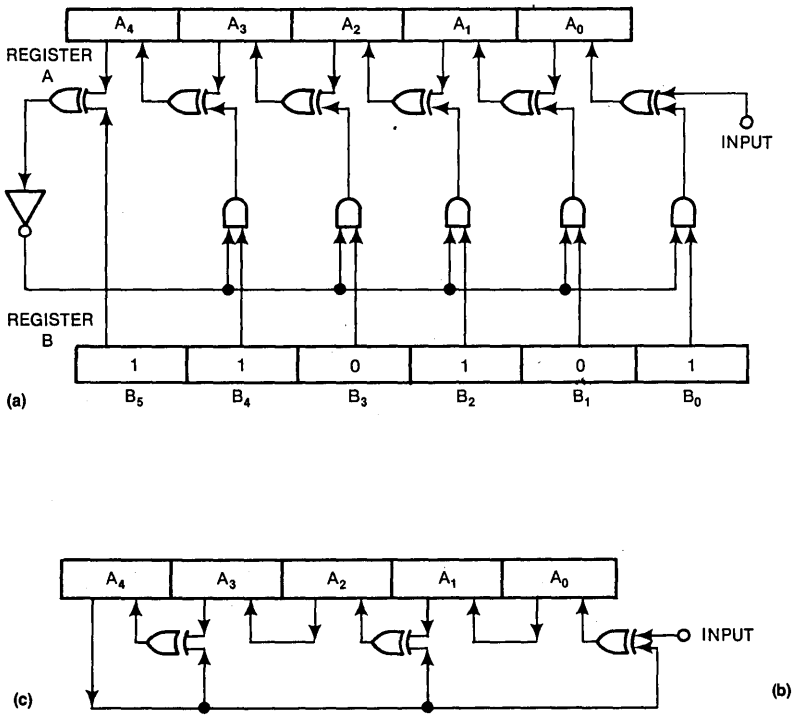
CRC error-detection schemes ensure data accuracy

Used by floppy discs and SDLC protocol, cyclic-redundancy-check error detection relies on simple concepts, math and hardware.

Krishna Rallapalli, Fairchild Camera & Instrument

Error-detection schemes using parity checks are well known and widely used in systems where data is transmitted, received or recorded. A parity check on a character is called vertical parity; a check on corresponding bits of every character in a message is called longitudinal parity. While they provide a satisfactory error-checking scheme when used together, such parity checks have a relatively high level of redundancy. For example, protecting an n-byte message (containing seven data bits and one parity bit for each byte) by vertical and longitudinal parities requires n+8 bits of redundancy. Thus, the check-to-data bit ratio in the message is (n+8)/7n. As n increases, this ratio approaches a limit of 1/7, giving a redundancy of about 14% in the protected message.

Another error-checking scheme, known as polynomial or cyclic checking, can demonstrate greater efficiency than traditional parity methods. The level of protection achieved with a 16-bit cyclic check, for example, is satisfactory for most applications. When such a check is used with a data block consisting of 7n data bits, the ratio of



INPUT	REGISTER				
	A_4	A_3	A_2	A_1	A_0
1	0	0	0	0	1
1	0	0	0	1	1
0	0	0	1	1	0
1	0	1	1	0	1
0	1	1	0	1	0
0	0	0	0	0	1
0	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	1	1
0	1	0	1	1	0
1	1	1	0	0	0
1	0	0	1	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	0	1	0	1
0	1	1	1	1	1
0	0	1	0	1	1

(b)

Fig 1—The polynomial divider in ***(a)*** *produces the results in* ***(b)*** *as input bits are clocked in. A greatly simplified circuit* ***(c)*** *is hardwired to divide only by* $X^5+X^4+X^2+1$.

CRC calculations use modulo-2 arithmetic to detect errors

check bits to data bits is only 16/7n, and this ratio approaches zero as the number of data bits increases.

This greater efficiency is inducing designers to incorporate cyclic check schemes in modern data-communication and peripheral equipment. For example, floppy discs and the SDLC protocol both use cyclic redundancy checking (CRC) for error detection. But how does the method work?

Bit streams and polynomials

A convenient way to represent a bit stream (message) k bits long is to think of it as a polynomial with k terms in a dummy variable X. The data bits of the message are simply the coefficients of the polynomial. Thus, the 12-bit message 100100011011 can be written as an 11th degree polynomial M(X):

$$M(X) = 1 \cdot X^{11} + 0 \cdot X^{10} + 0 \cdot X^9 + 1 \cdot X^8 + 0 \cdot X^7 + 0 \cdot X^6 + 0 \cdot X^5 + 1 \cdot X^4 + 1 \cdot X^3 + 0 \cdot X^2 + 1 \cdot X^1 + 1 \cdot X^0.$$

Computing r bits of cyclic check on this message requires another polynomial P(X), called the generator polynomial. The degree r of P(X) is greater than zero but less than the degree of M(X). Moreover, P(X) has a nonzero coefficient

Modulo-2—it's simpler than binary

In modulo-2 arithmetic, addition and subtraction yield the same results because modulo-2 has no carries or borrows. Therefore, you need consider only three arithmetic operations—addition, multiplication and division.

Adding two polynomials $X^6+X^5+X^2+1$ and $X^5+X^4+X^3+X^2$ yields $X^6+X^4+X^3+1$:

$$\begin{array}{r} X^6 + X^5 + 0 + 0 + X^2 + 0 + 1 = 1100101 \\ X^5 + X^4 + X^3 + X^2 + 0 + 0 = \underline{\quad 111100} \\ X^6 + 0 + X^4 + X^3 + 0 + 0 + 1 = 1011001. \end{array}$$

Multiplication of two polynomials $X^7+X^6+X^5+X^2+1$ and $X+1$ results in $X^8+X^5+X^3+X^2+X+1$:

$$(X^7 + X^6 + X^5 + X^2 + 1)(X + 1) = (11100101)(11)$$

Summing partial products:

$$\begin{array}{r} X^8 + X^7 + X^6 + 0 + 0 + X^3 + 0 + X + 0 = 111001010 \\ X^7 + X^6 + X^5 + 0 + 0 + X^2 + 0 + 1 = \underline{011100101} \\ X^8 + 0 + 0 + X^5 + 0 + X^3 + X^2 + X + 1 = 100101111. \end{array}$$

Note that multiplication of a polynomial by X^m gives a left-shifted bit pattern identical to the original except for zeros in the low-order m positions. For example:

$$X^5(X^{11} + X^{10} + X^8 + X^4 + X^3 + X + 1) = X^{16} + X^{15} + X^{13} + X^9 + X^8 + X^6 + X^5$$

where

$X^{11} + X^{10} + X^8 + X^4 + X^3 + X + 1 = 110100011011$

and $X^{16} + X^{15} + X^{13} + X^9 + X^8 + X^6 + X^5 =$ 1101000110110000.

Dividing $X^{13}+X^{11}+X^{10}+X^7+X^4+X^3+X+1$ by $X^6+X^5+X^4+X^3+1$ results in a quotient of $X^7+X^6+X^5+X^2+X+1$ and a remainder of X^4+X^2 as shown in this long division:

$X^{13} + X^{11} + X^{10} + X^7 + X^4 + X^3 + X + 1 =$ 10110010011011

$X^6 + X^5 + X^4 + X^3 + 1 = 1111001$

```
          11100111
1111001)10110010011011
        1111001
        -------
         1000000
         1111001
         -------
          1110010
          1111001
          -------
           1011110
           1111001
           -------
            1001111
            1111001
            -------
             1101101
             1111001
             -------
              10100.
```

Thus, the quotient Q(X) is 11100111 ($X^7+X^6+X^5+X^2+X+1$), and the remainder R(X) is 10100 (X^4+X^2).

for the X^0 term; it ends in +1. Obviously then, for a given message length, you can specify more than one generator polynomial. Fortunately, several accepted standard generator polynomials exist. For example, the SDLC protocol and floppy-disc equipment use the 16th-degree generator $X^{16}+X^{12}+X^5+1$.

Generating check bits

Cyclic-check error detection involves manipulating the message M(X) and the generator polynomial P(X), using the laws of ordinary algebra and modulo-2 arithmetic (see **box**) rather than binary. The procedure is as follows:

- The message polynomial is multiplied by X^r, where r is the degree of the generator polynomial. This multiplication results in ZEROs in the lower r positions of M(X), preparatory to appending the r check bits to the message. For example, let $M(X)=X^{11}+X^{10}+X^8+X^4+X^3+X+1$ and $P(X)=X^5+X^4+X^2+1$. Here r=5, giving $X^rM(X)=X^{16}+X^{15}+X^{13}+X^9+X^8+X^6+X^5$. $X^rM(X)$ can also be represented by the bit stream 11010001101100000.
- The result from the previous step is divided (still using modulo-2) by P(X), giving a quotient Q(X) and the check-bit pattern as a remainder R(X). In the example, Q(X)=100001100111 and R(X)=1011.
- The quotient is discarded and the remainder added to the result of the first step. The data bits, together with the remainder at the end, constitute the transmitted polynomial T(X). Here T(X)=11010001101101011. Note that transmission occurs from left to right and that data bits are unchanged, with the check bits following at the end.

On the receiving end

Depending on whether the transmission has acquired errors, the transmitted polynomial arrives at the receiver either intact or modified. Clearly, one of the ways to check data validity is to have the receiver recompute the check bits using the same P(X) as the transmitter. If the

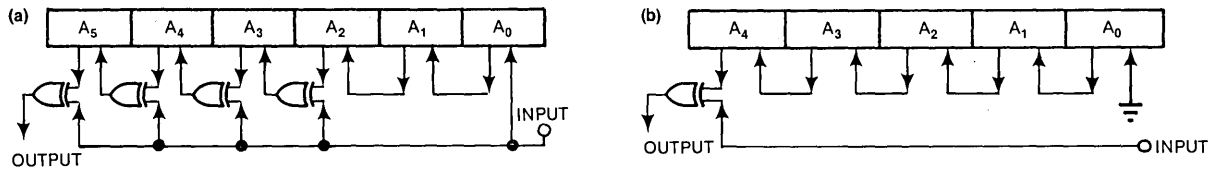
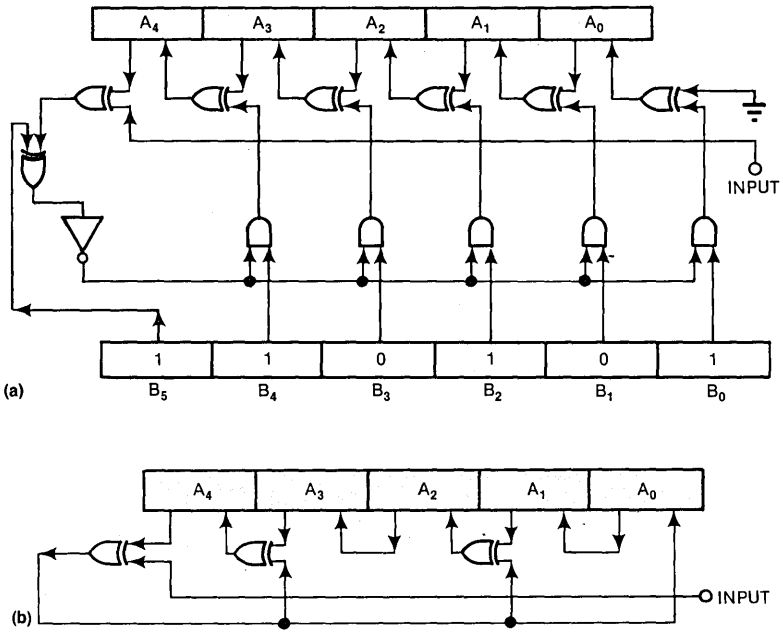


Fig 2—You can multiply *a bit stream by $X^6+X^5+X^4+X^3+1$ using the circuit in* **(a)**. *For many CRC applications, an easy multiplication by X^5 suffices* **(b)**.



INPUT	REGISTER				
	A_4	A_3	A_2	A_1	A_0
1	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	0
1	0	1	0	0	0
0	1	0	0	0	0
0	1	0	1	0	1
0	1	1	1	1	1
1	1	1	1	1	0
1	1	1	1	0	0
0	0	1	1	0	1
1	0	1	1	1	1
1	0	1	0	1	1

(c)

Fig 3—A complete cyclic-check generator *appears in* **(a)**, *with a simplified version for $P(X)=X^5+X^4+X^2+1$ in* **(b)**. *Both circuits produce the results in* **(c)**.

Cyclic redundancy checks can be blinded by certain errors

computed check bits agree with the received check bits, the received data is good.

Another, simpler method has the receiver divide the received polynomial by P(X). If there are no errors, this division results in a remainder of zero. You can check this yourself by dividing the transmitted polynomial T(X)= 11010001101101011 by P(X)=110101. If a nonzero remainder results, you can assume that T(X) has been corrupted by errors (or that you've made a mistake).

You can view the dropping or acquiring of bits during transmission as an addition of an error polynomial, E(X), to T(X). The bit pattern of E(X) determines the nature of the error. For example, since T'(X)=T(X)+E(X), if T'(X)= 10010001101101011 is received instead of T(X), then E(X)=01010001101101011. Note that if T'(X) happens to be exactly divisible by P(X), the receiver is blind to the errors. Multiple-bit errors can thus cancel each other's detectability, as can also occur with normal parity-checking schemes.

Consisting of divisor/dividend alignments and

CRC and LSI

Widespread use of cyclic checks and the existence of several standardized polynomials have spurred the availability of LSI CRC circuits. Manufacturers have even embedded some of these circuits in more complex chips such as floppy-disc controllers. The 9401, a CRC-only chip, illustrates standard features.

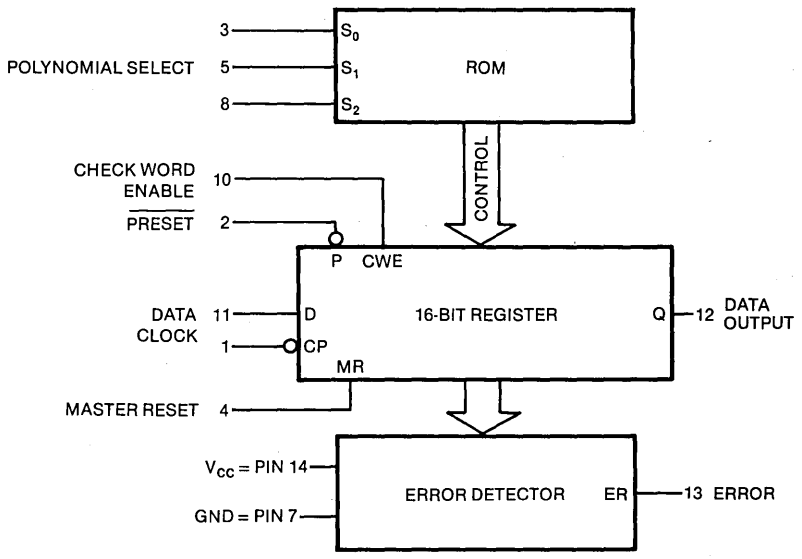
It consists of a 16-bit register with gating networks like those in **Fig 1.** Accommodating up to eight 16th-degree generator polynomials, the circuit selects which one to use when presented with appropriate signals on $S_{0\text{-}2}$. An on-chip ROM decodes these inputs to make the correct XOR and AND connections on the register.

The **table** shows the polynomials normally programmed into the 9401. Use of a ROM instead of the usual hardwired decoding allows easy mask changes for other generator polynomials. For example, the chip can be easily programmed to generate maximal-length pseudorandom numbers through use of an irreducible generator polynomial.

The Preset and Master Reset inputs control register initialization, and an error-detector circuit monitors the register for all-ZERO conditions. Computation of check bits starts with data entered using the HIGH-to-LOW transition of the clock. The Check Word Enable (CWE) input is held HIGH while a block of data is entered. After entry of the last data bit, CWE goes LOW and the check bits are clocked out.

Checking an incoming data block for errors involves entering data and check bits while holding CWE HIGH. With no detectable errors, the Error output is LOW after entry of the last check bit.

SELECT CODE			POLYNOMIAL	REMARKS
S_2	S_1	S_0		
L	L	L	$X^{16}+X^{15}+X^2+1$	CRC-16
L	L	H	$X^{16}+X^{14}+X+1$	CRC-16 REVERSE
L	H	L	$X^{16}+X^{15}+X^{13}+X^7+X^4+X^2+X+1$	
L	H	H	$X^{12}+X^{11}+X^3+X^2+X+1$	CRC-12
H	L	L	$X^8+X^7+X^5+X^4+X+1$	
H	L	H	X^8+1	LRC-8
H	H	L	$X^{16}+X^{12}+X^5+1$	CRC-CCITT
H	H	H	$X^{16}+X^{11}+X^4+1$	CRC-CCITT REVERSE



All required circuitry *for CRC error detection and check-bit generation resides on the 9401.*

subtractions, the modulo-2 division shown in the **box** can be easily embodied in a circuit: Shift registers implement the alignments; XOR gates, the subtractions.

Consider the two registers connected as shown in **Fig 1a**, with A cleared and B containing the divisor bit pattern 110101. The dividend is clocked into register A (MSB first). So long as A_1 is clear and B_5 is set, the AND gates are inhibited, and A behaves like a shift-left register. Eventually, the most significant data bit arrives in A_4.

At this point, both A_4 and B_5 are set, which means that the MSB's of the divisor and dividend are aligned. This alignment enables the AND gates but does not affect the XOR gates with inputs derived from ZERO bits in register B. The shift-left nature of register A at these locations is preserved. Thus, in **Fig 1** A_1 IN comes from A_0 OUT and A_3 IN comes from A_2 OUT. The remaining bit positions in register A receive the result of a modulo-2 subtraction. In summary, when register A is clocked after MSB alignment, the partial remainder gets loaded into register A. If clocking continues until all dividend bits have been processed, the remainder P(X) resides in register A (**Fig 1b**).

Close examination of **Fig 1a** reveals the possibility of considerably simplifying the circuit. **Fig 1c** presents such a simpler but arithmetically identical scheme. Suitable interconnections of shift registers and XOR gates result in the division algorithm. The total number of register bits equals the degree of the divisor, and the number of XOR gates is one less than the number of nonzero terms in the divisor.

A complete CRC circuit

Fig 1c's polynomial divider could serve as a CRC generator. But it does not provide the check bits (the remainder of the division) until the trailing-zero dividend bits have been processed. If the remainder must be appended to the data stream, there is a delay before the check bits are available, and such a gap is rarely acceptable. A circuit that can multiply two polynomials while dividing by a third eliminates this drawback. **Fig 2a** shows a circuit that multiplies an incoming polynomial by $X^6+X^5+X^4+X^3+1$. For cyclic-check applications, though, multiplication by one term of the form X^r suffices; **Fig 2b** presents a multiply-by-X^5 circuit.

Combining the multiplier in **Fig 2b** with the divider in **Fig 1c** gives a simultaneous multiply-by-X^5 and divide-by-$X^5+X^4+X^2+1$ circuit (**Fig 3a**). Again simplifying, **Fig 3b** shows a cyclic-check generator for $P(X)=X^5+X^4+X^2+1$. From **Fig 3c**, you can see that the remainder is available as soon as the last data bit is processed. Disabling the feedback through the XOR gates lets you shift out the check bits.

Reading backwards

Cyclic checks often find use in tape cassette and cartridge systems, most of which can read data in both forward and reverse directions. When the write circuitry uses a data-followed-by-check-bit format, the read circuitry encounters the check bits first when reading data in the opposite direction. Moreover, the bit order of the whole record is reversed. Thus, if the same check circuitry is used to validate data in both directions, erroneous error indications result.

The use of reverse polynomials to validate data in the reverse-read mode solves this problem. A reverse polynomial (or reciprocal) P'(X) of P(X) is defined as $P'(X)=X^rP(1/X)$. If $P(X)=X^{16}+X^{15}+X^2+1$, then $P'(X)=X^{16}+X^{14}+X+1$.

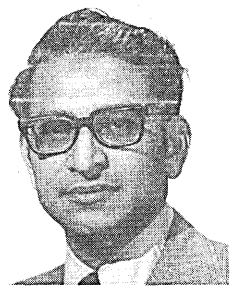
The foregoing discussion has assumed that the CRC register is clear before the start of any computations. If a block of data with all ZERO bits is processed by such a circuit, the check bits are also all ZEROs. In some applications, this property is unacceptable. However, the register can hold any initial bit pattern so long as the transmitter and receiver employ the same starting pattern. For example, as noted earlier, floppy-disc systems use the generator $P(X)=X^{16}+X^{12}+X^5+1$, with the register preset to all ONEs.

If a data polynomial M(X) with order k is protected by r bits of check, and all ONEs form the initialization pattern,

$$\frac{X^rM(X) + X^k(X^{r-1} + X^{r-2} + X^{r-3} + \cdots + X^2 + X + 1)}{P(X)} = Q(X) + \frac{R(X)}{P(X)}.$$

Addition of $X^k(X^{r-1}+X^{r-2}+\ldots+X+1)$ to $X^rM(X)$ corresponds to all-ONEs initialization. With no detectable read errors, an all-ZERO remainder still results from the check procedure. EDN

Author's biography



Krishna Rallapalli, who was application- engineering manager at Fairchild when he wrote this article, is now manager for MOS μP applications at Advanced Micro Devices, Sunnyvale, CA. After leaving Fairchild but before joining AMD, he worked at National Semiconductor. Holder of a BSEE from Jadaypur University, India, and an MSEE from the University of Saskatchewan, Canada, he lists traveling and writing among his hobbies.

Article Interest Quotient (Circle One)
High **476** ***Medium*** **477** ***Low*** **478**