The conventional solution to the problem is shown in Fig. 10. A few moments of study will indicate that it does in fact perform the required operation. The Flow Table Logic solution is shown in Fig. 11. Again the problem is easily checked for correct performance.

The simplicity of the wiring and the regularity of the circuit are quite apparent. Such an approach should certainly prove valuable when batch-fabricated devices become a reaility. As was stated earlier, when applied to one such technology (EL-PC) the design and fabrication of circuits is greatly simplified. An early model of an EL-PC combination lock is shown in Fig. 12.

Conclusion

The Flow Table Logic technique for circuit design presented here was intended for use with batch-fabricated (or perhaps micro-miniature) devices, hence the emphasis on simplicity and regularity. These are obtained in some cases at the expense of actual component count. The exchange was felt to be acceptable, however, since minimizing the number of active elements is not guarantee of minimum cost. An interesting point that should be considered is the logical delay associated with circuits designed by the Flow Table Logic technique. The circuit can go from one state to any other state in approximately two logical delays. Thus one has not sacrificed speed in the quest for regularity. The ease of circuit design should also be an advantage of this technique.

As is true in most developments, there are some problem areas that still require investigation. The coding of the input lines and, in fact, the coding of the states of the flow table are far from optimum. Only further work can reveal whether this can be improved without sacrificing the simplicity of the circuit. In addition, the necessary delay required is not found in all technologies and thus must be carefully considered in the solution of a problem.

Acknowledgment

The material presented represents the efforts of the authors plus that of R. J. Domenico. In addition, the authors are indebted to E. J. Skiko, J. Earle, and R. Robelen for many helpful discussions. The EL-PC combination lock was constructed by Dr. J. A. O'Connell and B. Narken.

Cyclic Codes for Error Detection*

W. W. PETERSON†, MEMBER, IRE, AND D. T. BROWN‡, MEMBER, IRE

***Summary*—Cyclic codes are defined and described from a new viewpoint involving polynomials. The basic properties of Hamming and Fire codes are derived. The potentialities of these codes for error detection and the equipment required for implementing error detection systems using cyclic codes are described in detail.**

Introduction

OF THE many developments in the area of error-detection and error-correcting codes during the past three years, probably the most important have pertained to cyclic codes. Since their introduction by Prange,[1] very attractive burst-error correcting cyclic codes have been found by Abramson,[2,3] Fire,[4] Melas,[5] and Reiger.[6] Cyclic codes for correcting random errors have been found by Prange,[1] Green and San Soucie,[7,8] Bose and Ray-Chaudhuri,[9] and Melas.[10] Encoding and

* Received by the IRE, August 1, 1960; revised manuscript received October 28, 1960.

† University of Florida, Gainesville, Fla.

‡ IBM Corp., Poughkeepsie, N. Y.

[1] E. Prange, "Cyclic Error-Correcting Codes in Two Symbols," Air Force Cambridge Research Center, Bedford, Mass., Tech. Note AFCRC-TN-57-103, September, 1957; "Some Cyclic Error-Correcting Codes with Simple Decoding Algorithms," Tech. Note AFCRC-TN-58-156, April, 1958; "The Role of Coset Equivalence in the Analysis and Decoding of Group Codes," Tech. Note AFCRC-TR-59-164; June, 1959.

[2] N. M. Abramson, "A Class of Systematic Codes for Non-Independent Errors," Electronics Res. Lab., Stanford University, Stanford, Calif., Tech. Rept. No. 51; December, 1958.

[3] N. M. Abramson, "Error Correcting Codes from Linear Sequential Networks," presented at the Fourth London Symp. on Information Theory, London, Eng.; August, 1960.

[4] P. Fire, "A Class of Multiple-Error-Correcting Binary Codes for Non-Independent Errors," Sylvania Electric Products, Inc., Mountain View, Calif., Rept. No. RSL-E-2; March, 1959.

[5] C. M. Melas, "A new group of codes for correction for dependent errors in data transmission," *IBM J. Res. Dev.*, vol. 4, pp. 58–65; January, 1960.

[6] S. H. Reiger, "Codes for the correction of clustered errors," IRE Trans. on Information Theory, vol. IT-6, pp. 16–21; March, 1960.

[7] J. H. Green, Jr., and R. L. San Soucie, "An error-correcting encoder and decoder of high efficiency," Proc. IRE, vol. 46, pp. 1744–1755; October, 1958.

[8] N. Zeiler, "On a Variation of the First Order Reed-Muller Codes," M.I.T. Lincoln Lab., Lexington, Mass., pp. 34–80; October, 1958.

[9] R. C. Bose and D. K. Ray-Chaudhuri, "A class of error-correcting binary group codes," *Information and Control*, vol. 3, pp. 68–79, March, 1960; "Further results on error correcting binary group codes," *Information and Control;* to be published.

[10] C. M. Melas, "A cyclic code for double error correction," *IBM J. Res. Dev.*, vol. 4, pp. 364–366; July, 1960.

error correcting procedures for these codes are relatively easily implemented using shift-registers with feedback connections.[11,12]

The first function of this paper is to introduce cyclic codes from a new viewpoint requiring only elementary mathematics and to derive the basic properties of Hamming and Fire codes. Second, the potentialities of cyclic codes for error detection and the equipment required for implementing error detection systems using cyclic codes are described in detail.

Polynomial Representation of Binary Information

We will be concerned with coding a message of k binary digits by appending $n-k$ binary digits as a check and transmitting the k information digits and then the $n-k$ check digits. It is convenient to think of the binary digits as coefficients of a polynomial in the dummy variable X. For example, a message 110101 is represented by the polynomial $1+X+X^3+X^5$. The polynomial is written low-order-to-high-order because these polynomials will be transmitted serially, high-order first, and it is conventional to indicate signal flow as occurring from left to right.

These polynomials will be treated according to the laws of ordinary algebra with one exception. Addition is to be done modulo two:

$$1\,X^a + 1\,X^a = 0\,X^a \quad 1\,X^a + 0\,X^a = 1\,X^a = 0\,X^a + 1X^a$$
$$0\,X^a + 0\,X^a = 0\,X^a \qquad -1\,X^a = 1\,X^a.$$

For example:

addition

$$\begin{array}{r} 1+X \qquad +X^3+X^4 \\ X+X^2 \qquad +X^4 \\ \hline 1+X+X^2 \qquad +X^4 \\ \hline X \qquad +X^3+X^4 \end{array}$$

multiplication

$$\begin{array}{r} 1+X \qquad +X^3+X^4 \\ 1+X \\ \hline 1+X \qquad +X^3+X^4 \\ X+X^2 \qquad +X^4+X^5 \\ \hline 1 \qquad +X^2+X^3 \qquad +X^5 \end{array}$$

In addition to the associative, distributive, and commutative properties of polynomials under this kind of algebra, we have, as in ordinary algebra, unique factorization; that is, every polynomial can be factored into prime or irreducible factors in only one way.[13]

Algebraic Description of Cyclic Codes

A cyclic code is defined in terms of a generator polynomial $P(X)$ of degree $n-k$. A polynomial of degree less than n is a code polynomial, *i.e.*, acceptable for transmission, if and only if it is divisble by the generator polynomial $P(X)$.[14] With this definition, the sum of two code polynomials is also a code polynomial, for if $F_1(X)$ and $F_2(X)$ are polynomials of degree less than n, which are divisible by $P(X)$, then $F_1(X)+F_2(X)$ is also of degree less than n and divisible by $P(X)$. Therefore, these codes are a special case of group codes, as studied by Slepian.[15]

If $P(X)$ has X as a factor, then every code polynomial has X as a factor and, therefore, has its zero-order coefficient equal to zero. Since such a symbol would be useless, we will consider only codes for which $P(X)$ is not divisible by X.

Code polynomials can be formed by simply multiplying any polynomial of degree less than k by $P(X)$. The following method has the advantage, however, that it results in a code polynomial in which the high-order coefficients are message symbols and the low-order coefficients are check symbols. To encode a message polynomial $G(X)$, we divide $X^{n-k}G(X)$ by $P(X)$ and then add the remainder $R(X)$ resulting from this division to $X^{n-k}G(X)$ to form the code polynomial:

$$X^{n-k}G(X) = Q(X)P(X) + R(X),$$

where $Q(X)$ is the quotient and $R(X)$ the remainder resulting from dividing $X^{n-k}G(X)$ by $P(X)$. Since in modulo two arithmetic, addition and subtraction are the same,

$$F(X) = X^{n-k}G(X) + R(X) = Q(X)P(X),$$

which is a multiple of $P(X)$ and, therefore, a code polynomial. Furthermore, $R(X)$ has degree less than $n-k$, and $X^{n-k}G(X)$ has zero coefficients in the $n-k$ low-order terms. Thus the k highest-order coefficients of $F(X)$ are the same as the coefficients of $G(X)$, which are the message symbols. The low order $n-k$ coefficients of $F(X)$ are the coefficients of $R(X)$, and these are the check symbols.

Example: Consider a code for which $n=15$, $k=10$, and $n-k=5$ which uses the generator polynomial $P(X)=1+X^2+X^4+X^5$. To encode the message 1010010001 corresponding to the polynomial $G(X)=1+X^2+X^5+X^9$, we divide $X^5G(X)$ by $P(X)$ and find the remainder. By long division it can be found that

$$X^5+X^7+X^{10}+X^{14} = (1+X^2+X^4+X^5)$$
$$\cdot(1+X+X^2+X^3+X^7+X^8+X^9)+(1+X).$$

The code polynomial is formed by adding the remainder $(1+X)$ to $X^5G(X)$:

[11] J. E. Meggitt, "Error correcting codes for correcting bursts of errors," *IBM J. Res. Dev.*, vol. 4, pp. 329-334; July, 1960.

[12] W. W. Peterson, "Error Correcting and Error Detecting Codes," Technology Press, Cambridge, Mass., to be published.

[13] See, for example, R. D. Carmichael, "Introduction to the Theory of Groups of Finite Order," Dover Publications, Inc., New York, N. Y., p. 256; 1956.

[14] According to the usual definition, a cyclic code is a group code with the added property that the cyclic shift of a code vector is also a code vector. Codes obtained by making a number of the leading information symbols identically zero and dropping them are called shortened cyclic codes. The codes described in this paper are cyclic codes if X^m-1 is evenly divisible by $P(X)$, and otherwise are shortened cyclic codes. See Prange, footnote 1, and Peterson, footnote 12.

[15] D. Slepian, "A class of binary signaling alphabets," *Bell Sys. Tech. J.*, vol. 35, pp. 203-234; January, 1956.

$$F(X) = (1 + X) + (X^5 + X^7 + X^{10} + X^{14})$$
$$\underbrace{1\ 1\ 0\ 0\ 0}_{\text{check symbols}}\ \underbrace{1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1}_{\text{information symbols}}.$$

Principles of Error Detection and Error Correction

An encoded message containing errors can be represented by

$$H(X) = F(X) + E(X)$$

where $F(X)$ is the correct encoded message and $E(X)$ is a polynomial which has a nonzero term in each erroneous position. Because the addition is modulo two, $F(X)+E(X)$ is the true encoded message with the erroneous positions changed.

If the received message $H(X)$ is not divisible by $P(X)$, then clearly an error has occurred. If, on the other hand, $H(X)$ is divisible by $P(X)$, then $H(X)$ is a code polynomial and we must accept it as the one which was transmitted, even though errors may have occurred. Since $F(X)$ was constructed so that it is divisible by $P(X)$, $H(X)$ is divisible by $P(X)$ if and only if $E(X)$ is also. Therefore, an error pattern $E(X)$ is detectable if and only if it is not evenly divisible by $P(X)$. To insure an effective check, the generator polynomial $P(X)$ must be chosen so that no error pattern $E(X)$ which we wish to detect is divisible by $P(X)$.

To detect errors, we divide the received, possibly erroneous, message $H(X)$ by $P(X)$ and test the remainder. If the remainder is nonzero, an error has been detected. If the remainder is zero, either no error or an undetectable error has occurred.

Example:

$$\begin{aligned}
F(X) &= 1 + X + X^5 + X^7 + X^{10} + X^{14} \\
&= 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1, \\
E(X) &= X^3 + X^6 + X^7 \\
&= 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0, \\
H(X) &= F(X) + E(X) \\
&= 1 + X + X^3 + X^5 + X^6 + X^{10} + X^{14} \\
&= 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1.
\end{aligned}$$

This $F(X)$ was taken from the previous example. The remainder after $H(X)$ is divided by $P(X)=1+X^2+X^4+X^5$ is $X^2+X^3+X^4$, and the fact that this is not zero shows that an error must have occurred. The same remainder occurs if $E(X)$ is divided by $P(X)$, since $F(X)$ is divisible by $P(X)$.

The ability of a code to correct errors is related to its ability to detect errors. For example, any code which detects all double errors is capable of correcting any single error. This can be seen by noting that if only a single error occurs, we can try to correct it by trying to change each symbol. A polynomial with one error and one symbol changed can be a code polynomial only if the erroneous symbol is the one which was changed, since all other combinations are equivalent to double errors and, therefore, are detectable. Similarly, a code which detects all combinations of $2t$ errors can correct any combination of t errors, since if t or fewer errors occur, changing all combinations of t or fewer positions results in a code polynomial only if all the erroneous positions are changed. The same argument shows that any code capable of detecting any two error bursts of length b or less can correct any single burst of length b or less. Finally, the converse of these statements is also true; any t-error correcting code can detect any combination of $2t$ errors and any code capable of correcting any single burst of length b can be used instead to detect any combination of two bursts of length b.

Detection of Single Errors

Theorem 1: A cyclic code generated by any polynomial $P(X)$ with more than one term detects all single errors.

Proof: A single error in the ith position of an encoded message (counting from the left and numbering the leftmost position zero) corresponds to an error polynomial X^i. To assure detection of single errors, it is necessary only to require that $P(X)$ does not divide X^i evenly. Certainly no polynomial with more than one term divides X^i evenly. Q.E.D.

The simplest polynomial with more than one term is $1+X$:

Theorem 2: Every polynomial divisible by $1+X$ has an even number of terms.

Proof: Let $F(X)=X^a+X^b+X^c+\cdots=(1+X)Q(X)$. Substituting $X=1$ gives

$$F(1) = 1 + 1 + 1 + \cdots = (1 + 1)Q(1) = 0.$$

There is one "1" in $F(1)$ for each term, and since the sum is zero, there must be an even number of terms. Q.E.D.

It follows that the code generated by $P(X)=1+X$ detects not only any single error, but also any odd number of errors. In fact, the check symbol must simply be an over-all parity check, chosen to make the number of ones in the code polynomial even.

Any polynomial of the form $1+X^c$ contains a factor $1+X$ since $1+X^c=(1+X)(X^{c-1}+X^{c-2}+\cdots+1)$. Therefore, if $P(X)$ contains a factor $1+X^c$, any odd number of errors will be detected.

Double and Triple Error Detecting Codes (Hamming Codes)

A polynomial $P(X)$ is said to belong to an exponent e if e is the least positive integer such that $P(X)$ evenly divides $X^e-1(=X^e+1 \bmod 2)$.

Theorem 3: A code generated by the polynomial $P(X)$ detects all single and double errors if the length n of the code is no greater than the exponent e to which $P(X)$ belongs.

Proof: Detection of all double errors requires that

$P(X)$ does not evenly divide X^i+X^j for any $i, j<n$. We can factor X^i+X^j (assuming $i<j$) to $X^i(1+X^{j-i})$. It is sufficient to require that $P(X)$ should not divide $1+X^{j-i}$, since $P(X)$ is assumed not to be divisible by X. But $j-i<n\leqq e$, and therefore, since $P(X)$ belongs to the exponent e, $P(X)$ cannot divide $1+X^{j-i}$. Thus the code will detect double errors. Since $P(X)$ is not divisible by X and certainly could not be just the constant 1, it must have more than one term, and will, by Theorem 1, detect single errors also. Q.E.D.

It can be shown that for any m there exists at least one polynomial $P(X)$ of degree m that belongs to $e=2^m-1$. This is the maximum possible value of e. Polynomials with this property (usually called primitive polynomials) are always irreducible. A few such polynomials are listed in Appendix II, and more extensive tables are available.[12,16] Thus for any m there is a double-error detecting code of length $n=2^m-1$ generated by a polynomial $P(X)$ of degree m, which therefore, has m check symbols and 2^m-1-m information symbols. These codes can be shown to be completely equivalent to Hamming single-error correcting codes.[2,3,12,17]

Theorem 4: A code generated by $P(X)=(1+X)P_1(X)$ detects all single, double, and triple errors if the length n of the code is no greater than the exponent e to which $P_1(X)$ belongs.

Proof: The single and triple errors are detected by the presence of the factor $1+X$, as is shown by Theorem 2, and double errors are detected because $P_1(X)$ belongs to the exponent $e\geqq n$, exactly as in Theorem 3. Q.E.D.

Codes of maximum length result if $P_1(X)$ is a primitive polynomial, and these codes are equivalent to Hamming single-error correcting, double-error detecting codes.[1,2,15]

Detection of a Burst-Error

A burst-error of length b will be defined as any pattern of errors for which the number of symbols between the first and last errors, including these errors, is b.

Example:

$$\text{The } E(X) = X^3 + X^6 + X^7$$
$$= 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0$$

of the previous example is a burst of length 5.

Theorem 5: Any cyclic code generated by a polynomial of degree $n-k$ detects any burst-error of length $n-k$ or less.

Proof: Clearly, any burst-error polynomial can be factored into the form $E(X)=X^iE_1(X)$ where $E_1(X)$ is of degree $b-1$. This burst can be detected if $P(X)$ does not evenly divide $E(X)$. Since $P(X)$ is assumed not to have X as a factor, it could divide $E(X)$ only if it could divide $E_1(X)$. But if $b\leqq n-k$, $P(X)$ is of higher degree than $E_1(X)$ and, therefore, certainly could not divide $E_1(X)$. Q.E.D.

A high percentage of longer bursts are detected as well.

Theorem 6: The fraction of bursts of length $b>n-k$ that are undetected is

$$2^{-(n-k)} \text{ if } b > n-k+1, \quad 2^{-(n-k-1)} \text{ if } b = n-k+1.$$

Proof: The error pattern is $E(X)=X^iE_1(X)$ where $E_1(X)$ has degree $b-1$. Since $E_1(X)$ has terms X^0 and X^{b-1}, there are $b-2$ terms X^j, where $0<j<b-1$, that can have either zero or one coefficients, and so there are 2^{b-2} distinct polynomials $E_1(X)$.

The error is undetected if and only if $E_1(X)$ has $P(X)$ as a factor.

$$E_1(X) = P(X)Q(X).$$

Since $P(X)$ has degree $n-k$, $Q(X)$ must have degree $b-1-(n-k)$. If $b-1=n-k$, then $Q(X)=1$, and there is only one $E_1(X)$ which results in one undetected error, namely $E_1(X)=P(X)$. The ratio of the number of undetected bursts to the total number of bursts is, therefore, $1/2^{b-2}=2^{-(n-k-1)}$ for this case. If $b-1>n-k$, $Q(X)$ has terms X^0 and $X^{b-1-(n-k)}$ and has $b-2-(n-k)$ arbitrary coefficients. There are, therefore, $2^{b-2-(n-k)}$ choices of $Q(X)$ which give undetectable error patterns. The ratio for this case is $2^{b-2-(n-k)}/2^{b-2}=2^{-(n-k)}$. Q.E.D.

Detection of Two Bursts of Errors (Abramson and Fire Codes)

Theorem 7: The cyclic code generated by $P(X)=(1+X)P_1(X)$ detects any combination of two burst-errors of length two or less if the length of the code, n, is no greater than e, the exponent to which $P_1(X)$ belongs.

Proof: There are four types of error patterns.

1) $E(X) = X^i + X^j$
2) $E(X) = (X^i + X^{i+1}) + X^j$
3) $E(X) = X^i + (X^j + X^{j+1})$
4) $E(X) = (X^i + X^{i+1}) + (X^j + X^{j+1})$

2) and 3) have odd numbers of errors and so they are detected by the $1+X$ factor in $P(X)$. For 4), $E(X)=(1+X)(X^i+X^j)$. The $1+X$ factor is cancelled by the $1+X$ factor in $P(X)$ so we will require for both 1) and 4) that X^i+X^j is not evenly divisible by $P_1(X)$. X^i+X^j is not evenly divisible by $P_1(X)$ as is shown in the proof of Theorem 3. Q.E.D.

These codes are equivalent to the Abramson codes, which correct single and double adjacent errors.[2,3] They are also the same as the Hamming single-error correcting, double-error detecting codes of Theorem 6.

Theorem 8: The cyclic code generated by

$$P(X) = (X^c + 1)P_1(X)$$

will detect any combination of two bursts

[16] A. A. Albert, "Fundamental Concepts of Higher Algebra," University of Chicago Press, Chicago, Ill.; 1956. This book contains a table of irreducible polynomials giving the exponent e to which they belong (see p. 161).

[17] N. M. Abramson, "A note on single error correcting binary codes," IRE Trans. on Information Theory, vol. IT-6, pp. 502–503; September, 1960.

$$E(X) = X^{i}E_1(X) + X^{j}E_2(X),$$

provided $c+1$ is equal to or greater than the sum of the lengths of the bursts, $P_1(X)$ is irreducible and of degree at least as great as the length of the shorter burst, and provided the length of the code is no greater than the least common multiple of c and the exponent e to which $P_1(X)$ belongs.

The proof, which is elementary but rather long, is given in Appendix IV. These are Fire codes.[4,12]

Other Cyclic Codes

There are several important cyclic codes which have not been discussed. Burst-error correcting codes have been treated also by Melas,[5] Meggitt,[11] and Reiger.[6] Codes for correcting independent random errors have been discovered by Melas.[10] Prange,[1] and Bose and Chaudhuri.[9,12,18] Any of these codes can also be used for error detection. The Bose-Chaudhuri codes are particularly important. For any choice of m and t there exists a Bose-Chaudhuri code of length 2^m-1 which is capable of correcting any combination of t errors (or alternatively, detecting any combination of $2t$ errors) and which requires a generator polynomial of degree no greater than mt. The description of the structure of these codes and the methods for choosing the polynomials is beyond the scope of this paper.

Implementation

Thus far, an algebraic method has been given for encoding and decoding to detect various types of errors. Briefly, to encode a message, $G(X)$, $n-k$ zeros are annexed (*i.e.*, the multiplication $X^{n-k}G(X)$ is performed) and then $X^{n-k}G(X)$ is divided by a polynomial $P(X)$ of degree $n-k$. The remainder is then subtracted from $X^{n-k}G(X)$. (It replaces the $n-k$ zeroes.) This encoded message is divisible by $P(X)$, but we have shown that if $P(X)$ is properly chosen, the message will not be evenly divisible if it contains detectable errors. The only nontrivial manipulation to be performed for both encoding and error detection is division by a fixed polynomial, $P(X)$.

The following is an example of division under addition modulo two:

$$\begin{array}{r} 1X^3+1X^2+0X+1 \\ 1X^2+0X+1\,\overline{)\,1X^5+1X^4+1X^3+0X^2+1X+0} \\ 1X^5+0X^4+1X^3 \\ \hline 1X^4+0X^3+0X^2+1X+0 \\ 1X^4+0X^3+1X^2 \\ \hline 0X^3+1X^2+1X+0 \\ 1X^2+0X+1 \\ \hline 1X+1 \end{array}.$$

We now repeat this division employing only the coefficients of the polynomials:

$$\begin{array}{r} 1\ 1\ 0\ 1 \\ 1\ 0\ 1\,\overline{)\,1\ 1\ 1\ 0\ 1\ 0} \\ 1\ 0\ 1 \\ \hline 1\ 0\ 0\ 1\ 0 \\ 1\ 0\ 1 \\ \hline 0\ 1\ 1\ 0 \\ 1\ 0\ 1 \\ \hline 1\ 1 \end{array}.$$

It can be seen that modulo two arithmetic has simplified the division considerably. Furthermore, we do not require the quotient, so the division to find the remainder can be described as follows:

1) Align the coefficient of the highest degree term of the divisor and the coefficient of the highest degree term of the dividend and subtract (the same as addition).
2) Align the coefficient of the highest degree term of the divisor and the coefficient of the highest degree term of the difference and subtract again.
3) Repeat the process until the difference has lower degree than the divisor. The difference is the remainder.

The hardware to implement this algorithm is a shift register and a collection of modulo two adders. (A modulo two adder is equivalent to the logical operation EXCLUSIVE OR). The number of shift register positions is equal to the degree of the divisor, $P(X)$, and the dividend is shifted through high order first and left to right. As the first one (the coefficient of the high-order term of the dividend) shifts off the end we subtract the divisor by the following procedure:

1) In the subtraction the high-order terms of the divisor and the dividend always cancel. As the high-order term of the dividend is shifted off the end of the register, this part of the subtraction is done automatically.
2) Modulo two adders are placed so that when a *one* shifts off the end of the register, the divisor (except the high-order term which has been taken care of) is subtracted from the contents of the register. The register then contains a difference that is shifted until another *one* comes off the end and then the process is repeated. This continues until the entire dividend is shifted into the register.

Fig. 1 gives a register that performs a division by $1+X^2+X^4+X^5$. Note that if alignment of divisor and dividend is considered to be accomplished when the high-order term of the dividend shifts off the end, then the divisor is automatically subtracted.

[18] W. W. Peterson, "Encoding and error-correction procedures for the Bose-Chaudhuri codes," IRE Trans. on Information Theory, vol. IT-6, pp. 459–470; September, 1960.

The shift register shown in Fig. 1 has, if used for encoding, one drawback that can be overcome by a slight modification. Recall that when encoding a message polynomial, $G(X)$, we calculate the remainder of the division of $X^{n-k}G(X)$ by $P(X)$. The straightforward procedure is to shift the message followed by $n-k$ zeroes into the register. When the last *zero* is in the register we obtain the remainder. Because this remainder replaces the $n-k$ zeros to form the encoded message, it is necessary to delay the message $n-k$ shift times so that the remainder can be gated in from the encoder register at the proper time.

An example of this method of encoding is given in Fig. 2. Initially, the gate G_1 is open and the gate G_2 is shorted, allowing the remainder on dividing $X^{n-k}G(X)$ to be calculated. After the message plus $n-k$ *zeros* is shifted in, G_1 is shorted and G_2 is opened. This allows the remainder which is now in the register to replace the $n-k$ zeros in the output. Error detection with this circuit requires that gate G_1 be open and gate G_2 be shorted. After $H(X)$ has been shifted in, the register contains the remainder. If this is nonzero, an error has occurred.

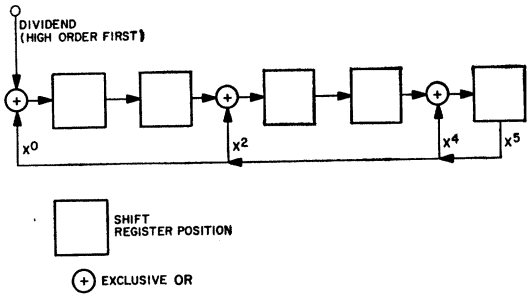

Fig. 1—A shift register for dividing by $1+X^2+X^4+X^5$.

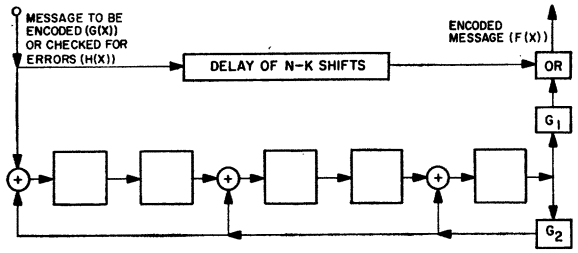

Fig. 2—One method of encoding on detecting errors. (In this example, $P(X)=1+X^2+X^4+X^5$.)

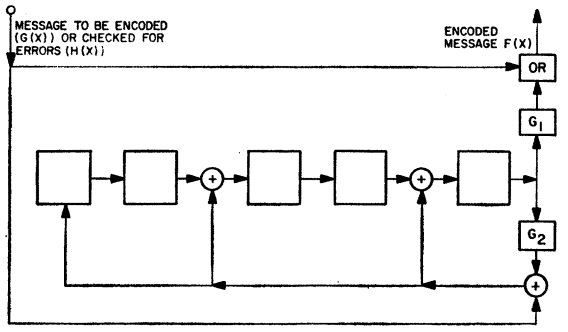

Fig. 3—A more efficient circuit for encoding and error detection. (In this example, $P(X)=1+X^2+X^4+X^5$.)

The delay of $n-k$ shifts can be avoided if Fig. 2 is modified to give the circuit of Fig. 3. In Fig. 3, instead of shifting the polynomial into the low-order end of the register, it is treated as if it were shifting out of the high-order end. This is equivalent to advancing every term in the polynomial by $n-k$ positions, or multiplying by X^{n-k}. Now in encoding, as soon as $G(X)$ has been completely shifted into the register, the register contains the remainder on dividing $X^{n-k}G(X)$ by $P(X)$. Then gate G_1 is shorted, gate G_2 is opened, and the remainder follows the undelayed $G(X)$ out of the encoder to form $F(X)$.

To minimize hardware, it is desirable to use the same register for both encoding and error detection, but if the circuit of Fig. 3 is used for error detection we will get the remainder on dividing $X^{n-k}H(X)$ by $P(X)$ instead of the remainder on dividing $H(X)$ by $P(X)$. It turns out that this makes no difference, for if $H(X)$ is evenly divisible by $P(X)$ then obviously $H(X)X^{n-k}$ is evenly divisible, and if $H(X)$ is not evenly divisible by $P(X)$ then $H(X)X^{n-k}$ will not be evenly divisible either, provided the divisor $P(X)$ does not have a factor X. Any useful $P(X)$ will satisfy this restriction. The circuit of Fig. 3 can, then, be used for both encoding and error detection.

Error correction is by its nature a much more difficult task than error detection. It can be shown that each different correctable error pattern must give a different remainder after division by $P(X)$. Therefore, error correction can be done as follows:

1) Divide the received message $H(X)=F(X)+E(X)$ by $P(X)$ to obtain the remainder.
2) Obtain the $E(X)$ corresponding to the remainder from a table or by some calculation.
3) Subtract $E(X)$ from $H(X)$ to obtain the correct transmitted message $F(X)$.

Both the encoding and step 1 of the decoding are the same for error correction as for error detection. The error-correction equipment is more complex in that it requires equipment for the table look-up or computation of step 2, and it requires that the entire received message $H(X)$ be stored temporarily while the remainder is being calculated and $E(X)$ is being determined. The calculation required in step 2 can be done simply with a shift register for burst-error or single-error correcting codes, but is quite complex for codes that correct multiple random errors. Details of error-correction procedures are beyond the scope of this paper, but can be found in references.[11,12,17]

Conclusion

A simple presentation of cyclic codes has been given in terms of polynomials. The attractive features of these codes for error detection, both their high efficiency and the ease of implementation, have been emphasized.

Appendix I

Notation

k = number of binary digits in the message before encoding,
n = number of binary digits in the encoded message,
$n-k$ = number of check digits,
b = length of a burst of errors,
$G(X)$ = message polynomial (of degree $k-1$),
$P(X)$ = generator polynomial (of degree $n-k$),
$R(X)$ = remainder on dividing $X^{n-k}G(X)$ by $P(X)$, $R(X)$ is of degree less than $n-k$,
$F(X)$ = encoded message polynomial, $F(X)=X^{n-k}G(X)-R(X)$,
$E(X)$ = error polynomial,
$H(X)$ = received encoded message polynomial, $H(X)=F(X)+E(X)$.

Appendix II

A Short Table of Primitive Polynomials

Primitive Polynomial	e
$1+X$	1
$1+X+X^2$	3
$1+X+X^3$	7
$1+X+X^4$	15
$1+X^2+X^5$	31
$1+X+X^6$	63
$1+X^3+X^7$	127
$1+X^2+X^3+X^4+X^8$	255
$1+X^4+X^9$	511
$1+X^3+X^{10}$	1023
$1+X^2+X^{11}$	2047
$1+X+X^4+X^6+X^{12}$	4095
$1+X+X^3+X^4+X^{13}$	8191
$1+X+X^6+X^{10}+X^{14}$	16383
$1+X^{14}+X^{15}$	32767

Appendix III

Data for Some Representative Codes

Detection Capabilities	k_{max}	$n-k$	$P(X)$	Reference
Any odd number of errors	any value	1	$1+X$	Theorem 2
Two errors, a burst of length 4 or less, 88 per cent of the bursts of length 5, 94 per cent of longer bursts*	11	4	$1+X+X^4$	Theorems 3, 5, 6
Two errors, a burst of 9 or less, 99.6 per cent of the bursts of length 10, 99.8 per cent of longer bursts	502	9	$1+X^4+X^9$	Theorems 3, 5, 6
Two bursts of length 2 or less, any odd number of errors, a burst of 5 or less, 93.8 per cent of the bursts of length 6, 96.9 per cent of longer bursts†	10	5	$(1+X+X^4)(1+X)=1+X^2+X^4+X^5$	Theorems 2, 5, 6, 7
Two bursts of combined length 12 or less, any odd number of errors, a burst of 22 or less, 99.99996 per cent of the bursts of length 23, 99.99998 per cent of longer bursts	22495	22	$(1+X^2+X^{11})(1+X^{11})=1+X^2+X^{13}+X^{22}$	Theorems 2, 5, 6, 8
Any combination of 6 or fewer errors, a burst of length 11 or less, 99.9 per cent of bursts of length 12, 99.95 per cent of longer bursts	12	11	$1+X^2+X^4+X^5+X^6+X^{10}+X^{11}$	Theorems 5, 6, and footnote 1
Any combination of 7 or fewer errors, any odd number of errors, a burst of length 31 or less, all but about 1 in 10^9 of longer bursts	992	31	$(1+X)(1+X^3+X^{10})(1+X+X^2+X^3+X^{10})(1+X^2+X^3+X^8+X^{10})$	Theorems 2, 5, 6, and footnotes 9, 12, 18

* *Note:* $1+X+X^4$ belongs to $e=15$ and $11+4=15$.
† *Note:* This is the code used in all examples.

Appendix IV

Proof of Theorem 8

The error polynomial has the form:

$$E(X) = X^i[E_1(X) + X^{j-i}E_2(X)].$$

$E_1(X)$ has degree b_1-1 and $E_2(X)$ has degree b_2-1.

The generator polynomial $P(X)$ cannot have a factor X so we need only consider the factor of $E(X)$ in brackets. Let $j-i=d$, assume $E'(X)=E_1(X)+X^dE_2(X)$ is divisible by X^c+1, and let $d=cq+r$ with $r<c$.

Then,

$$\begin{aligned} E'(X) &= E_1(X) + X^{cq+r}E_2(X) \\ &= E_1(X) + X^rE_2(X) + [X^rE_2(X)]\cdot[X^{cq}+1]. \quad (1) \end{aligned}$$

Now $X^{cq}+1$ contains a factor X^c+1 for

$$\begin{aligned} X^{cq}+1 = (X^c+1)(&X^{c(q-1)} + X^{c(q-2)} \\ &+ X^{c(q-3)} + \cdots + X^0). \end{aligned}$$

Hence the rightmost term in (1) is divisible by X^c+1. $E'(X)$ was assumed divisible by X^c+1 and so from (1), $E_1(X)+X^rE_2(X)$ must be divisible by X^c+1. Using this result, we can let

$$\begin{aligned} E_1(X) + X^rE_2(X) &= [X^c+1][Q(X)] \\ E_1(X) + X^rE_2(X) &= Q(X) + X^cQ(X). \quad (2) \end{aligned}$$

We will assume that $Q(X)\neq 0$. Let the degree of $Q(X)$ be h. The degree of the right-hand side of (2) is $c+h$ and the degree of the left-hand side is either b_1-1 or $r+b_2-1$. Then, for (2) to be true we must have either $c+h=b_1-1$ or $c+h=r+b_2-1$. Since it was assumed that $c\geqq b_1+b_2-1$ we must have the second relation.

$$c + h = r + b_2 - 1.$$

Again using $c\geqq b_1+b_2-1$ we have

$$b_1 + b_2 - 1 + h \leqq r + b_2 - 1 \quad \text{or} \quad b_1 + h \leqq r.$$

From this, $b_1\leqq r$ or $b_1-1<r$ and as $b_1\neq 0$, $h<r$.

Applying these results to (2), we see that both $E_1(X)$ and $Q(X)$ are of lower degree than any of the terms in $X^rE_2(X)$. It follows then, given the assumption that $Q(X)\neq 0$, that

$$X^rE_2(X) = X^cQ(X). \quad (3)$$

As $E_2(X)$ always contains an X^0 term, the lowest order term in $X^rE_2(X)$ is of degree r. The lowest order term in $X^cQ(X)$ is of degree at least c but $r<c$ so (3) can never be satisfied. Therefore, the only solution of (2) is with $Q(X)=0$ giving $E_1(X)+X^rE_2(X)=0$.

As $E_1(X)$ always contains an X^0 term, $r=0$ and $E_1(X)=E_2(X)$. Substituting in (1) gives

$$E'(X)=E_2(X)\,[X^{cq}+1].$$

This is the form of the error polynomial if it is evenly divisible by X^c+1. It is sufficient to show that this polynomial is not evenly divisible by $P_1(X)$ to guarantee that $E(X)$ is never evenly divisible by $P(X)=P_1(X)$, $[X^c+1]$. $P_1(X)$ is irreducible, so to divide $E'(X)=E_2(X)$ $[X^{cq}+1]$ it must divide one of the factors. For this special case, $E_1(X)=E_2(X)$ so $b_1=b_2$, and since both bursts have the same length, this is the length of the shorter burst. It was specified that $P_1(X)$ is of degree no less than the length of the shorter burst so it is of higher degree than $E_2(X)$ and cannot divide $E_2(X)$.

It remains to show that $P_1(X)$ does not evenly divide $X^{cq}+1$. Make the substitution $cq=ue+v$ where e is the exponent to which $P_1(X)$ belongs and $v<e$. Now $v\neq 0$ because cq is less than or equal to the length of the message and the length of the message is less than or equal to the least common multiple of c and e. Since cq is a multiple of c, it cannot be a multiple of e.

$$\begin{aligned} X^{cq} + 1 &= X^{ue+v} + 1 \\ X^{cq} + 1 &= X^v + 1 + X^v(X^{ue} + 1). \end{aligned}$$

As was shown previously, $X^{ue}+1$ is divisible by X^e+1. Furthermore, $P_1(X)$, by definition, divides X^e+1; therefore, $P_1(X)$ divides $X^{ue}+1$. However, X^e+1 is the lowest degree polynomial of this form that $P_1(X)$ divides, so $P_1(X)$ does not divide X^v+1. As $v\neq 0$, we have shown that $P_1(X)$ does not divide $X^{cq}+1$, completing the proof.