Description

This application is a continuation in-part of the co-pending application entitled "Apparatus for Computing Multiplicative Inverses in Error Encoding and Decoding Devices", Serial Number 050,725 filed May 15, 1987 on behalf of the same inventor.

FIELD OF INVENTION

This invention relates to error correction code decoding mechanisms in general and more particularly to the decoding of Bose-Chaudhuri-Hocquenghem (BCH) error correction codes, including Reed-Solomon error correction codes.

DESCRIPTION OF PRIOR ART

The importance of error correction coding of data in digital computer systems has increased greatly as the density of the data recorded on mass storage media, more particularly magnetic disks, has increased. With higher recording densities, a tiny imperfection in the recording surface of a disk can corrupt a large amount of data. In order to avoid losing that data, error correction codes ("ECC's") are employed to, as the name implies, correct the erroneous data.

Before a string of data symbols is recorded on a disk, it is mathematically encoded to form ECC symbols. The ECC symbols are then appended to the data string to form code words - data symbols plus ECC symbols - and the code words are then stored on the disk. When the stored data is to be accessed from the disk, the code words containing the data symbols are retrieved from the disk and mathematically decoded. During decoding any errors in the data are detected and, if possible, corrected through manipulation of the ECC symbols [For a detailed description of decoding see Peterson and Weldon, Error Corection Codes, 2d Edition, MIT Press, 1972].

Stored digital data can contain multiple errors. One of the most effective types of ECC used for the correction of multiple errors is a Reed-Solomon code [For a detailed description of Reed-Solomon codes, see Peterson and Weldon, Error Correction Codes]. Error detection and correction techniques for Reed-Solomon ECC's are well known. Id. One such technique begins with again encoding the code word data to generate ECC symbols and then comparing these ECC symbols with the ECC symbols in the code word, i.e. the ECC symbols generated by the pre-storage encoding of the data, to detect any errors in the retrieved data. [For a detailed discussion of this error detection technique, see United States Patent 4,413,339 issued to Riggle and Weng].

The comparison of the two sets of ECC symbols is made by exclusive OR'ing ("XOR'ing") them. If there are no errors in the retrieved data, the XOR'ing will produce a string of symbols containing only zeros. If there are errors in the retrieved data, the result of the XOR'ing will be non-zero and error correction will be performed, if possible. If the number of non-zero symbols which result from the XOR'ing is less than the number of errors the code is designed to correct, all the errors are in the ECC symbols and no error correction is required. If the number of non-zero resulting symbols is equal to or greater than the number of errors the code is designed to correct, the data contains one or more errors and error correction is then performed.

In order to correct the data errors, the locations and the magnitudes or "values" of the errors must be determined. The first step in determining the error locations and the error values is to form error syndrome equations from the results of the XOR'ing of the ECC symbols. [For a detailed discussion of the generation of the error syndromes refer to United States Patent 4,413,339 issued to Riggle and Weng]. The error syndromes are then used to generate polynomials which, when evaluated, will identify the location and the value of each of the errors.

Alternatively the error syndromes may be formed using the entire retrieved code word. This avoids the steps of re-encoding the retrieved data and comparing the ECC symbols with the retrieved ECC symbols. However, error syndromes will be generated for every code word, even those without errors, although in the latter case the syndromes will be identically zero. Where there are errors, i.e. the error syndromes are not zero, the syndromes are then used to locate and correct any errors in exactly the same manner as the syndromes formed from the results of the XOR'ing of the ECC symbols. The choice between these two arrangements may, for example, depend on whether the syndromes are generated in software or hardware.

The calculation of the values of the errors is necessarily dependent on their locations. Typically, the error locations are calculated first by using the error syndromes to generate an error locator polynomial $\delta(x)$. The solutions or roots of the equation, $\delta(x) = 0$, designate the locations of the errors. The roots, x_r, are then substituted into an error evaluator polynomial $\Phi(x)$ and the polynomial is evaluated. The results are then divided by the corresponding values of the first derivative, $\delta'(x_r)$, of the error locator polynomial to provide the values of the errors. Once both the error locations and the corresponding error values are known, the data can then be corrected.

The time it takes to perform error correction significantly affects the rate at which data can be retrieved from a storage device. As the potential for errors in stored data increases with the use of higher recording densities, the effect of slow error correction has a material effect on the average speed with which the data can be retrieved. The increased

data retrieval time in turn limits the speed of all computer applications that require the retrieval of the stored data. This speed limitation occurs at a time when computer system technology advances have otherwise made possible faster data transfer operations.

What is needed is an error correction technique that can be performed significantly more quickly than the methods currently in use. Such a technique will enable computer systems to operate at the faster rates permitted by the advances in the state of the art computer technology.

SUMMARY OF THE INVENTION

The invention calculates the locations and values of the errors simultaneously and thus more quickly corrects the errors. Using this technique, errors can be corrected in "real-time" even with fast data transfer rates. The error locations and error values are found by evaluating various expressions, at values of x, where x is an element of the Galois Field used to encode the data. All values of x corresponding to possible error locations are systematically tried using an iterative evaluation technique which uses the values of each of the terms of the equations calculated for one value of x in evaluating the equations at the next value of x, in, for example, the error locator equation

$$\delta(x) = 0 \tag{1}$$

where $\delta(x)$ is the error locator polynomial, to find all the existing solutions to the equation. Each solution to the equation indicates the location of an error.

The error locator polynomial

$$\delta(x) = 1 + \delta_1 * x + \delta_2 * x^2 + \ldots + \delta_n * x^n \tag{2}$$

can be rewritten

$$\delta(x) = 1 + \delta_{even}(x) + \delta_{odd}(x) \tag{2a}$$

and equation (1) can be rewritten

$$\delta_{even}(x) + \delta_{odd}(x) = 1 \tag{1a}$$

where $\delta_{even}(x)$ and $\delta_{odd}(x)$ are the even- and odd-power terms of polynomial (2) and the δ's are calculated by the error syndrome generator. Thus the iterative evaluation technique is used to find the solutions, x_r, of equation (1), where $\delta(x)$ is in the form of polynomial (2a).

Solutions to polynomial (2a) are found by simultaneously inserting a first value of x, x_{a1}, into the expressions $\delta_{even}(x)$ and $\delta_{odd}(x)$ and also into an error evaluator polynomial $\Phi(x)$. Next, while the error locator equation (1a) is evaluated at the calculated values of $\delta_{even}(x_{a1})$ and $\delta_{odd}(x_{a1})$ to determine if x_{a1} is a solution, the now known values of the error evaluator polynomial $\Phi(x_{a1})$ and the expression $\delta_{odd}(x_{a1})$ are substituted into an error value formula

$$v = \frac{x * \Phi(x)}{\delta_{odd}(x)} \tag{3}$$

where the expression

$$\frac{\delta_{odd}(x)}{x} \tag{4}$$

represents a simplified version of the first derivative, $\delta'(x)$ of $\delta(x)$. Equation (3) is then evaluated using the method for Galois Field division disclosed in the co-pending application entitled "Apparatus for Computing Multiplicative Inverses in Error Encoding and Decoding Devices", Serial Number 067,712 filed June 26, 1987. Thus as soon as an error location is found, i.e., x_{a1} satisfies equation (1a), the error value, v_{a1}, associated with that location is also known. The error can then be quickly corrected.

Next, the values of expressions $\delta_{odd}(x)$, $\delta_{even}(x)$ and the polynomial $\Phi(x)$ are calculated for another value of x, x_{a2}. The respective terms of $\delta_{odd}(x_{a2})$, $\delta_{even}(x_{a2})$ and $\Phi(x_{a2})$ are calculated using the previously calculated terms of $\delta_{odd}(x_{a1})$, $\delta_{even}(x_{a1})$ and $\Phi(x_{a1})$, and the calculations are thus simplified.

The newly evaluated expressions $\delta_{even}(x_{a2})$, $\delta_{odd}(x_{a2})$ and polynomial $\Phi(x_{a2})$ are then substituted into the error locator equation (1a) and the error value equation (3) to determine if x_{a2} is a solution, and if so, where the corresponding error is located. If x_{a2} is a solution, the error value v_{a2} which was simultaneously calculated for x_{a2} is used to correct the error. If x_{a2} is not a solution, the calculated error value is ignored. Then, the expressions $\delta_{odd}(x)$, $\delta_{even}(x)$ and the polynomial $\Phi(x)$ are similarly evaluated at the other values of x. Once all the solutions are found, that is, the last error in the data is located, the data is corrected, if possible, and thus the data can then be immediately transferred. The system is then ready to process the next block of stored data.

DESCRIPTION OF THE DRAWINGS

This invention is pointed out with particularity in the appended claims. The above and further advantages of this invention may be better understood by referring to the following description taken in conjunction with the accompanying drawings, in which:

Figure 1 is a functional block diagram of an error correcting system in accordance with the preferred embodiment of the invention. The system simultaneously determines data error locations and values and corrects the errors.
Figure 2 is a functional block diagram of an error correction system constructed in accordance with the preferred embodiment of the invention;
Figure 3 is a functional block diagram of the calculators shown in Figure 2.

DETAILED DESCRIPTION

It should be understood that all addition, multiplication and division operations performed during the error correction process are Galois Field operations, the variable x is a Galois Field element of the form α^i, where α is the primitive polynomial associated with a Galois Field, and the Galois Field is of the form $GF(2^P)$.

with reference to Figures 1 and 2, a retrieved code word is first checked for the presence of any data errors (Steps 10-16), as described above. Specifically, the data portion of the retrieved code word is encoded in an encoder 42 (Step 12), and the ECC symbols generated by this encoding are compared in a comparator 44, or XOR'ed, with the code word ECC symbols. (Step 14).

If the results of the comparison indicate that there are one or more errors in the retrieved data, error correction is performed. The results of the comparison are used to generate the error syndromes (Step 18b) in an error syndrome generator 48 which is enabled by a gate 46. The error syndromes are then utilized to generate an error locator polynomial, $\delta(x)$, and an error evaluator polynomial, $\Phi(x)$, (Step 20) in generators 50 and 54, respectively. The error locator polynomial and the error evaluator polynomial are evaluated to identify the locations and the values of the data errors.

The error locator polynomial $\delta(x)$ is defined as:

$$\delta(x) = 1 + \delta_1 * x + \delta_2 * x^2 + \ldots + \delta_n * x^n \qquad (2)$$

The locations of the errors are typically identified by the solutions of the error locator equation (1), $\delta(x) = 0$. Each root, x_r, of this equation identifies the location of one error.

In conventional error code correction methods, the roots of the error locator equation are substituted into an error evaluator polynomial and into the first derivative, $\delta'(x)$, of the error locator polynomial to calculate the values of the respective errors. The error evaluator polynomial $\Phi(x)$ is defined as:

$$\Phi(x) = \Phi_0 * x^0 + \Phi_1 * x^1 + \Phi_2 * x^2 + \ldots + \Phi_u * x^u \qquad (5)$$

where $u = n - 1$. The value v_r of an error corresponding to a root x_r is given by the error value formula

$$v = \frac{x^{c} * \Phi(x)}{\delta'(x)}$$

where $\delta'(x)$ is the first derivative of $\delta(x)$, and C is a constant, the value of which depends on the ECC generator polynomial selected and the roots of the generator polynomial. The generator polynomial has been selected such that x^1 is the lowest root, and therefore, C is equal to one. If a generator polynomial is selected which does not have x^1 as the lowest root, the expression $(x^{c-1} * \Phi(x))$ should be substituted for $\Phi(x)$.

Calculation of the first derivative of $\delta(x)$ can be simplified by expioiting a property of Galois Fields of characteristic 2, i.e. $GF(2^P)$. Consider polynomial (2) with n being an even number. This expression can be rewritten:

$$\delta(x) = 1 + \underbrace{[\delta_1 * x + \delta_3 * x^3 + \ldots + \delta_{n-1} * x^{n-1}]}_{\text{odd}} + \underbrace{[\delta_2 * x^2 + \delta_4 * x^4 + \ldots + \delta_n * x^n]}_{\text{even}}$$

which is

$$\delta(x) = 1 + \delta_{odd}(x) + \delta_{even}(x) \qquad (2a)$$

The first derivative of polynomial (2a) is

$$\delta'(x) = 0 + \delta_{odd}'(x) + \delta_{even}'(x) \qquad (2b)$$

In a Galois Field of characteristic 2, the derivative of a variable "Y" raised to an even power, for example Y^{2n}, whose

derivative is $2n * Y^{2n-1}$, is equal to zero. The derivative of Y^{2n-1}, however, which is $(2n-1)*Y^{2n-2}$, is non-zero; it is equal to Y^{2n-2}. Thus the first derivative of $\delta(x)$ can be rewritten

$$\delta'(x) = 0 + \delta_1 * x^0 + 3*\delta_3 * x^2 + 5*\delta_5 * x^4 + \ldots + (n-1)*\delta_{n-1} * x^{n-2} . \quad (6)$$

. By taking advantage of the properties of Galois Fields of characteristic 2, this can be further simplified to:

$$\delta'(x) = 0 + \delta_1 + \delta_3 * x^2 + \delta_5 * x^4 + \ldots + \delta_{n-1} * x^{n-2} \quad (6a)$$

. Formula (6a) can be further rewritten as

$$\delta'(x) = \frac{\delta_{odd}(x)}{x} \quad (6b)$$

As stated above, error locations are indicated by the roots of equation (1). This equation can be rewritten, using polynomial (2a) as:

$$\delta_{even}(x) + \delta_{odd}(x) = 1 \quad (1a)$$

The corresponding error value formula will then be

$$v = \frac{x * \Phi(x)}{\delta_{odd}(x)} \quad (3)$$

The solutions of equation (1a) are found by using an iterative technique to evaluate the expressions $\delta_{odd}(x)$ and $\delta_{even}(x)$ for successive values of x in calculators 56 and 58, respectively. For each such evaluation, the values of $\delta_{odd}(x)$ and $\delta_{even}(x)$ are then added to determine if the sum is equal to 1.

Basically, the iterative technique tries the values of x, where x represents possible locations of errors in the encoded data, in a specified order, as solutions to equation (1a), until all the values of x have been tried. At the same time that the expressions $\delta_{odd}(x)$ and $\delta_{even}(x)$ are being evaluated at each value of x, the iterative technique is used to evaluate the error evaluator polynomial $\Phi(x)$ at the same x value in calculator 60. The iterative technique is discussed in detail in connection with Figure 3.

Continuing with reference to Figures 1 and 2, an adder 64 sums the evaluations of $\delta_{odd}(x_{a1})$ and $\delta_{even}(x_{a1})$. At the same time, results of the evaluations of the error evaluator polynomial $\Phi(x)$ and the expression $\delta_{odd}(x)$ are substituted into the error value formula (3) (Step 28) and the error value is calculated in an error value generator 62. The generator 62 may, for example, use the method for Galois Field division disclosed in the co-pending U.S. patent application entitled "Apparatus for Computing Multiplicative Inverses in Error Encoding and Decoding Devices", Serial Number 067,712 filed on June 26, 1987.

The output of adder 64 is applied to a one-detector 65, which asserts its output whenever $\delta_{odd}(x) + \delta_{even}(x) = 1$, that is, whenever a particular x is a root of equation (1a) (Step 32). The asserted output of the one-detector 65 enables an error corrector 66 which quickly corrects the error with the then known error value v obtained from the generator 62. If a particular x is not a root of equation (1a), the output of the one-detector 65 is not asserted and the calculated error value v is disregarded, that is, the error corrector 66 is not enabled. The process is continued until all the values of x corresponding to possible error locations have been tried as solutions to equation (1a) (Steps 36, 38).

If, after all values of x corresponding to possible error locations have been tried (Step 36), the number of solutions found for equation (1a), which are totaled in a counter 70, is equal to the degree of the error locator polynomial $\delta(x)$, the corrected data is passed to the circuitry requesting the data. If the number of solutions found for equation (la) is less than the degree of the error locator polynomial $\delta(x)$, the data is labeled uncorrectable.

Figure 3 illustrates the iterative technique used in evaluating the expressions $\delta_{odd}(x)$, $\delta_{even}(x)$ and the polynomial $\Phi(x)$ in calculators 56, 58, and 60, respectively. Calculator 56 contains a plurality of Galois Field multipliers 100, 102 and 103. Each Galois Field multiplier computes the values of a particular term of $\delta_{odd}(x)$ at various values of x. Thus multiplier 100 calculates the value of $x*\delta_1$, multiplier 102 calculates $x^3*\delta_3$, etc. The calculated values of each of the terms at a particular x are stored in the multipliers and then added modulo 2 in a summer 101 to evaluate $\delta_{odd}(x)$.

Specifically, to evaluate $\delta_{odd}(x)$ at various x's the multipliers 100, 102 and 103 are set to an inital condition using the $\delta_{odd}(x)$ terms. Multiplier 100, for example, is set to δ_1, and multiplier 102 is set to δ_3. Next, to calculate the values of the $\delta_{odd}(x)$ terms at a first value of x, x_{a1}, where x_{a1} is typically α, the contents of the multipliers are multiplied by the constants α, α^3..., α^{n-1}, respectively. Thus, for example, the δ_{n-1} term is multiplied by the constant a^{n-1} to calculate the value of $(x_{a1})^{n-1}*\delta_{n-1}$. The products are then stored in the multipliers, and added by the summer 101 to calculate $\delta_{odd}(x_{a1})$.

Thereafter, to calculate the value of $\delta_{odd}(x)$ at the next value of x, x_{a2}, i.e. α^2 if $x_{a1} = \alpha$, the contents of the multipliers are again multiplied by the constants α, α^3, α^5... a^{n-1}, respectively, the products are stored in the multipliers, and the stored terms are again added to calculate $\delta_{odd}(x_{a2})$. This calculation process is repeated until $\delta_{odd}(x)$ is calculated for all values of x corresponding to possible error locations.

Calculators 58 and 60 (Figure 2), consisting of Galois Field multipliers 104-109, operate similarly to calculate the values of $\delta_{even}(x)$ and $\Phi(x)$ at the various values of x.

If the first value of x tried as a solution is x_n, rather than x_1, the constants used in the respective multipliers are α^{-1}, α^{-2}, α^{-3}... α^{-n}.

Evaluating the expressions $\delta_{odd}(x)$, $\delta_{even}(x)$ and $\Phi(x)$ for the various values of x is simplified using the iterative evaluation technique. The iterative evaluation technique is adapted from the well-known Chien search technique.

Ascertaining the locations and the values of the data errors simultaneously provides that the errors can be quickly and efficiently corrected. Using conventional techniques for the individual components of the process, the error locations are first calculated and then the error values are found, and only after these two calculations have been performed can the errors be corrected. Moreover, the technique described above takes advantage of the properties of Galois Fields $GF(2^P)$ to simplify the evaluation of the first derivative of the error locator polynomial, and thus the calculation of the error values is simplified and quickened. The ability to simultaneously ascertain the locations and values of the data errors and to then immediately correct them permits a computer system to take advantage of faster transfer rates which are technologically feasible.

The foregoing description has been limited to a specific embodiment of this invention. It will be apparent, however, that variations and modifications may be made to the invention, with the attainment of some or all of the advantages of the invention. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the scope of the invention.

Claims

1. A method of correcting data which is encoded into code words using a BCH error correction code based on Galois Field $GF(2^P)$ and a selected generator polynomial, said method comprising the steps of:

   A. generating error syndromes corresponding to a given data code word containing errors;
   B. using said error syndromes, generating an error evaluator polynomial $\Phi(x)$ and an error locator polynomial $\delta(x)$ of the form $\delta(x) = 1 + \delta_{even}(x) + \delta_{odd}(x)$, where $\delta_{even}(x)$ and $\delta_{odd}(x)$ are the even- and odd- terms of $\delta(x)$, respectively;
   C. for each successive value of x, which are each elements of $GF(2^P)$ corresponding to possible locations of errors in the code word, simultaneously substitutins x into expressions for the error evaluator polynomial, $\Phi(x)$, and $\delta_{even}(x)$ and $\delta_{odd}(x)$;
   D. evaluating said expressions;
   E. substituting the results of step D into the error locator polynomial and an error value formula;
   F. evaluating said error locator polynomial and said error value formula to obtain the location of a possible error in the code word and a corresponding error value, respectively;
   G. (a) if said error locator polynomial is equal to zero for a given x, indicating that there is an error at the code word location corresponding to x, correcting the error with the corresponding calculated error value, or
   (b) if said error locator polynomial is not equal to zero for a given x, indicating that there is no error at the code word location corresponding to x, disregarding the corresponding calculated error value; and
   H. repeating steps C-G for all remaining values of x corresponding to possible error locations in the code word.

2. The method for correcting errors in Claim 1 wherein:

   A. if the generator polynomial has x^1 as a root, said error value formula for determining the value of an error, v, at the code word location corresponding to x is:

$$v = \frac{x * \Phi(x)}{\delta_{odd}(x)}$$

   and
   B. if the generator polynomial does not have x^1 as a root, said error value formula for determining the value of an error, v, at the code word location corresponding to x is:

$$v = \frac{x^c * \Phi(x)}{\delta_{odd}(x)}$$

   where C is a constant which depends on the generator polynomial.

3. The method for correcting errors in Claim 2 wherein the error correction code used to encode the data is a Reed-Solo-

Calculators 58 and 60 (Figure 2), consisting of Galois Field multipliers 104-109, operate similarly to calculate the values of $\delta_{even}(x)$ and $\Phi(x)$ at the various values of x.

If the first value of x tried as a solution is x_n, rather than x_1, the constants used in the respective multipliers are α^{-1}, α^{-2}, α^{-3}... α^{-n}.

Evaluating the expressions $\delta_{odd}(x)$, $\delta_{even}(x)$ and $\Phi(x)$ for the various values of x is simplified using the iterative evaluation technique. The iterative evaluation technique is adapted from the well-known Chien search technique.

Ascertaining the locations and the values of the data errors simultaneously provides that the errors can be quickly and efficiently corrected. Using conventional techniques for the individual components of the process, the error locations are first calculated and then the error values are found, and only after these two calculations have been performed can the errors be corrected. Moreover, the technique described above takes advantage of the properties of Galois Fields $GF(2^P)$ to simplify the evaluation of the first derivative of the error locator polynomial, and thus the calculation of the error values is simplified and quickened. The ability to simultaneously ascertain the locations and values of the data errors and to then immediately correct them permits a computer system to take advantage of faster transfer rates which are technologically feasible.

The foregoing description has been limited to a specific embodiment of this invention. It will be apparent, however, that variations and modifications may be made to the invention, with the attainment of some or all of the advantages of the invention. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the scope of the invention.

Claims

1. A method of correcting data which is encoded into code words using a BCH error correction code based on Galois Field $GF(2^P)$ and a selected generator polynomial, said method comprising the steps of:

   A. generating error syndromes corresponding to a given data code word containing errors;
   B. using said error syndromes, generating an error evaluator polynomial $\Phi(x)$ and an error locator polynomial $\delta(x)$ of the form $\delta(x) = 1 + \delta_{even}(x) + \delta_{odd}(x)$, where $\delta_{even}(x)$ and $\delta_{odd}(x)$ are the even- and odd- terms of $\delta(x)$, respectively;
   C. for each successive value of x, which are each elements of $GF(2^P)$ corresponding to possible locations of errors in the code word, simultaneously substitutins x into expressions for the error evaluator polynomial, $\Phi(x)$, and $\delta_{even}(x)$ and $\delta_{odd}(x)$;
   D. evaluating said expressions;
   E. substituting the results of step D into the error locator polynomial and an error value formula;
   F. evaluating said error locator polynomial and said error value formula to obtain the location of a possible error in the code word and a corresponding error value, respectively;
   G. (a) if said error locator polynomial is equal to zero for a given x, indicating that there is an error at the code word location corresponding to x, correcting the error with the corresponding calculated error value, or
   (b) if said error locator polynomial is not equal to zero for a given x, indicating that there is no error at the code word location corresponding to x, disregarding the corresponding calculated error value; and
   H. repeating steps C-G for all remaining values of x corresponding to possible error locations in the code word.

2. The method for correcting errors in Claim 1 wherein:

   A. if the generator polynomial has x^1 as a root, said error value formula for determining the value of an error, v, at the code word location corresponding to x is:

$$v = \frac{x * \Phi(x)}{\delta_{odd}(x)}$$

   and
   B. if the generator polynomial does not have x^1 as a root, said error value formula for determining the value of an error, v, at the code word location corresponding to x is:

$$v = \frac{x^C * \Phi(x)}{\delta_{odd}(x)}$$

   where C is a constant which depends on the generator polynomial.

3. The method for correcting errors in Claim 2 wherein the error correction code used to encode the data is a Reed-Solo-

* C L M S *

mon code.

4. The method for correcting errors in Claim 2 wherein said error evaluation polynomial and error locator expressions for determinig the location and value of errors in a data code word are evaluated using an iterative evaluation method comprising the steps of:

A. calculating the values of the terms of the two error locator expressions $\delta_{even}(x)$ and $\delta_{odd}(x)$ and the values of the terms of the error evaluator polynomial $\Phi(x)$ for a first value of x corresponding to a given code word location;
B. storing the calculated value of each term of the error locator expressions and the error evaluator polynomials;
C. substituting the stored values into the error locator polynomial and the error value formula to determine the locations and values of errors in the code word corresponding to x;
D. to try a next value of x corresponding to another possible error location, multiplying each stored value by a constant B^n, where:

i. x * B is equal to the next value of x to be tried, and
ii. n is the exponent of the variable in the term having that value; and

E. repeating steps B-D for each x corresponding to a possible error location.

5. An apparatus for correcting data which is encoded into code words using a BCH error correction code based on Galois Field $GF(2^P)$, comprising:

A. an error syndrome generator for generating error syndromes;
B. a polynomial generator connected to receive the error syndromes for generating, using the error syndromes, an error evaluator polynomial $\Phi(x)$ and an error locator polynomial $\delta(x)$ of the form $\delta(x) = 1 + \delta_{even}(x) + \delta_{odd}(x)$, where $\delta_{even}(x)$ and $\delta_{odd}(x)$ are the even- and odd- terms of $\delta(x)$, respectively;
C. code word error correction means for locating the errors in the code word and correcting them, said code word error correction means including:

1. first substitution means for, for each successive value of x, which are elements of $GF(2^P)$ corresponding to possible error locations in the code word, simultaneously substituting x into expressions for the error evaluator polynomial $\Phi(X)$, $\delta_{even}(x)$ and $\delta_{odd}(x)$;
2. first evaluating means for evaluating said expressions at the substituted values of x;
3. second substitution means for substituting the values of the evaluated expressions into the error locator polynomial and an error value formula;
4. second evaluating means for evaluating said error locator polynomial and said error value formula to obtain the locations of errors in the code word and corresponding error values; and
5. correcting means for--

(1) if said erroe locator polynomial is equal to zero, correcting the error at the code word location corresponding to x using the calculated error value, or
(2) if said error locator polynomial is not equal to zero, disregarding the calculated error value;

6. repeating steps 1-5 for all remaining value of x corresponding to possible error locations in the code word.

6. The apparatus for correcting errors in Claim 5 wherein said error value formula into which the first substitution means substitutes the calculated expression values is --

$$\frac{x * \Phi(x)}{\delta_{odd}(x)}$$

if the generator polynomial has x^i as the lowest root the error value formula, or

$$\frac{x^c * \Phi(x)}{\delta_{odd}(x)}$$

if the generator polynomial does not have x^1 as the lowest root the error value formula, where C is a constant which depends on the generator polynomial.

* C L M S *

7. The apparatus for correcting errors in Claim 6, wherein said first evaluating means includes:

1. calculating means for calculating the values of the terms of the two error locator expressions $\delta_{even}(x)$ and $\delta_{odd}(x)$ and the values of the terms of the error evaluator polynomial for a first value of x corresponding to a given code word location;
2. storage means for storing the calculated values of each of the terms of the error locator expressions and the error evaluator polynomial and applying said values to said second substitution means; and
3. multiplication means for calculating a next value of x corresponding to another possible error location by multiplying each of the stored values by a constant B^n, where:

i. x * B is equal to the next value of x to be tried, and
ii. n is the exponent of the variable in the term having that value.

Patentansprüche

1. Ein Verfahren zum Korrigieren von Daten, die in Codeworte codiert sind, unter Verwendung eines BCH-Fehlerkorrekturcodes beruhend auf einem Galois-Feld $GF(2^p)$ und einem ausgewählten Generator-Polynom, wobei das Verfahren die Schritte umfaßt:

A. Erzeugen von Fehler-Syndromen, die einem gegebenen Daten-Codewort entsprechen, das Fehler enthält;
B. unter Verwendung der Fehler-Syndrome, Erzeugen eines Fehlerauswerte-Polynoms $\Phi(x)$ und eines Fehlerlokalisier-Polynoms $\delta(x)$ der Form $\delta(x) = 1 + \delta_{gerade}(x) + \delta_{ungerade}(x)$, wo $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$ die geraden beziehungsweise ungeraden Terme von $\delta(x)$ sind;
C. für jeden folgenden Wert von x, die alle Elemente von $GF(2^p)$ sind, welche möglichen Stellen von Fehlern in dem Codewort entsprechen, gleichzeitiges Substituieren von x in Ausdrücke für das Fehlerauswerte-Polynom $\Phi(x)$ und $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$;
D. Auswerten der Ausdrücke;
E. Substituieren der Ergebnisse von Schritt D in das Fehlerlokalisier-Polynom und eine Fehlerwertformel;
F. Auswerten des Fehlerlokalisier-Polynoms und der Fehlerwertformel, um die Stelle eines möglichen Fehlers in dem Codewort beziehungsweise einen entsprechenden Fehlerwert zu erhalten;
G. (a) falls das Fehlerlokalisier-Polynom für ein gegebenes x gleich Null ist, was angibt, daß ein Fehler bei der x entsprechenden Codewort-Stelle vorliegt, Korrigieren des Fehlers mit dem entsprechenden berechneten Fehlerwert, oder
(b) falls das Fehlerlokalisier-Polynom für ein gegebenes x nicht gleich Null ist, was angibt, daß kein Fehler bei der x entsprechenden Codewort-Stelle vorliegt, Ignorieren des entsprechenden berechneten Fehlerwertes; und
H. Wiederholen der Schritte C - G für alle verbleibenden Werte von x, die möglichen Fehlerstellen in dem Codewort entsprechen.

2. Das Verfahren zum Korrigieren von Fehlern nach Anspruch 1, worin:

A. falls das Generator-Polynom x^1 als eine Wurzel aufweist, die Fehlerwertformel zum Bestimmen des Wertes eines Fehlers, v, bei der x entsprechenden Codewort-Stelle lautet:

$$v = \frac{x * \Phi(x)}{\delta_{ungerade}(x)}$$

und
B. falls das Generator-Polynom x^1 nicht als eine Wurzel aufweist, die Fehlerwertformel zum Bestimmen des Wertes eines Fehlers, v, bei der x entsprechenden Codewort-Stelle lautet:

$$v = \frac{x^C * \Phi(x)}{\delta_{ungerade}(x)}$$

wo C eine Konstante ist, die von dem Generator-Polynom abhängt.

3. Das Verfahren zum Korrigieren von Fehlern nach Anspruch 2, worin der Fehlerkorrekturcode, der verwendet wird, um die Daten zu codieren, ein Reed-Solomon-Code ist.

4. Das Verfahren zum Korrigieren von Fehlern nach Anspruch 2, worin das Fehlerauswerte-Polynom und die Fehler-

* C L M S *

lokalisièr-Ausdrücke zum Bestimmen der Stelle und des Wertes von Fehlern in einem Daten-Codewort ausgewertet werden unter Verwendung eines iterativen Auswertungsverfahrens, umfassend die Schritte:

A. Berechnen der Werte der Terme der beiden Fehlerlokalisier-Ausdrücke $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$ und der Werte der Terme des Fehlerauswerte-Polynoms $\Phi(x)$ für einen ersten Wert von x, der einer gegebenen Codewort-Stelle entspricht;
B. Speichern des berechneten Wertes von jedem Term von den Fehlerlokalisier-Ausdrücken und dem Fehlerauswerte-Polynom;
C. Substituieren der gespeicherten Werte in das Fehlerlokalisier-Polynom und die Fehlerwertformel, um die Stellen und Werte von Fehlern in dem Codewort entsprechend x zu bestimmen;
D. um einen nächsten Wert von x zu versuchen, der einer anderen möglichen Fehlerstelle entspricht, Multiplizieren jedes gespeicherten Wertes mit einer Konstante B^n, wo:

i. $x * B$ gleich dem nächsten Wert von x ist, der versucht werden soll, und
ii. n der Exponent der Variable in dem Term mit jenem Wert ist; und

E. Wiederholen der Schritte B - D für jedes x, das einer möglichen Fehlerstelle entspricht.

5. Ein Gerät zum Korrigieren von Daten, die in Codeworte codiert sind, unter Verwendung eines BCH-Fehlerkorrekturcodes beruhend auf einem Galois-Feld $GP(2^p)$, umfassend:

A. einen Fehler-Syndrom-Generator zum Erzeugen von Fehler-Syndromen;
B. einen Polynom-Generator, der verbunden ist, um die Fehler-Syndrome zu empfangen, zum Erzeugen, unter Verwendung der Fehler-Syndrome, eines Fehlerauswerte-Polynoms $\Phi(x)$ und eines Fehlerlokalisier-Polynoms $\delta(x)$ der Form $\delta(x) = 1 + \delta_{gerade}(x) + \delta_{ungerade}(x)$, WO $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$ die geraden beziehungsweise ungeraden Terme von $\delta(x)$ sind;
C. Codewort-Fehlerkorrekturmittel zum Lokalisieren der Fehler in dem Codewort und Korrigieren von diesen, wobei das Codewort-Fehlerkorrekturmittel einschließt:

1. erstes Substitutionsmittel zum, für jeden folgenden Wert von x, die Elemente von $GF(2^p)$ sind, welche möglichen Fehlerstellen in dem Codewort entsprechen, gleichzeitigen Substituieren von x in Ausdrücke für das Fehlerauswerte-Polynom $\Phi(x)$, $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$;
2. erstes Auswertemittel zum Auswerten der Ausdrücke bei den substituierten Werten von x;
3. zweites Substitutionsmittel zum Substituieren der Werte der ausgewerteten Ausdrücke in das Fehlerlokalisier-Polynom und eine Fehlerwertformel;
4. zweites Auswertemittel zum Auswerten des Fehlerlokalisier-Polynoms und der Fehlerwertformel, um die Stellen von Fehlern in dem Codewort und entsprechende Fehlerwerte zu erhalten; und
5. Korrekturmittel zum

(1) falls das Fehlerlokalisier-Polynom gleich Null ist, Korrigieren des Fehlers bei der x entsprechenden Codewort-Stelle unter Verwendung des berechneten Fehlerwertes oder
(2) falls das Fehlerlokalisier-Polynom nicht gleich Null ist, Ignorieren des berechneten Fehlerwertes;

6. Mittel zum Wiederholen der Schritte 1 - 5 für alle verbleibenden Werte von x, die möglichen Fehlerstellen in dem Codewort entsprechen.

6. Das Gerät zum Korrigieren von Fehlern nach Anspruch 5, worin die Fehlerwertformel, in die das zweite Substitutionsmittel die berechneten Ausdruckswerte substituiert,

$$\frac{x * \Phi(x)}{\delta_{ungerade}(x)}$$

lautet, falls das Generator-Polynom x^1 als die niedrigste Wurzel aufweist, oder

$$\frac{x^C * \Phi(x)}{\delta_{ungerade}(x)}$$

falls das Generator-Polynom x^1 nicht als die niedrigste Wurzel aufweist, wo C eine Konstante ist, die von dem Generator-Polynom abhängt.

* C L M S *

7. Das Gerät zum Korrigieren von Fehlern nach Anspruch 6, worin das erste Auswertemittel einschließt:

1. Berechnungsmittel zum Berechnen der Werte der Terme der beiden Fehlerlokalisier-Ausdrücke $\delta_{gerade}(x)$ und $\delta_{ungerade}(x)$ und der Werte der Terme des Fehlerauswerte-Polynoms für einen ersten Wert von x, der einer gegebenen Codewort-Stelle entspricht;
2. Speichermittel zum Speichern der berechneten Werte von jedem der Terme der Fehlerlokalisier-Ausdrücke und des Fehlerauswerte-Polynoms und Anwenden der Werte auf das zweite Substitutionsmittel; und
3. Multiplikationsmittel zum Berechnen einen nächsten Wertes von x, der einer anderen möglichen Fehlerstelle entspricht, durch Multiplizieren jedes der gespeicherten Werte mit einer Konstante B^n, wo:

i. $x * B$ gleich dem nächsten Wert von x ist, der versucht werden soll, und
ii. n der Exponent der Variable in dem Term mit jenem Wert ist.

Revendications

1. Procédé de correction de données qui sont codées en mots de code par utilisation d'un code de correction d'erreurs BCH selon un Corps de Galois $GF(2^P)$ et un polynôme générateur sélectionné, ledit procédé comprenant les étapes de :

A. génération de syndromes d'erreur correspondant à un mot donné de code de données contenant des erreurs ;
B. utilisation desdits syndromes d'erreur, génération d'un polynôme évaluateur d'erreurs $\Phi(x)$ et d'un polynôme localisateur d'erreurs $\delta(x)$ de la forme $\delta(x) = 1 + \delta_{pair}(x) + \delta_{impair}(x)$, où $\delta_{pair}(x)$ et $\delta_{impair}(X)$ sont respectivement les termes pair et impair de $\delta(x)$;
C. pour chaque valeur successive de x, qui sont chacune des éléments de $GF(2^P)$ correspondant à des emplacements possibles d'erreurs dans le mot de code, substitution simultanée de x dans les expressions pour le polynôme évaluateur d'erreurs $\Phi(x)$, et $\delta_{pair}(x)$ et $\delta_{impair}(X)$;
D. évaluation desdites expressions ;
E. substitution des résultats de l'étape D dans le polynôme localisateur d'erreurs et une formule de valeur d'erreur ;
F. évaluation dudit polynôme localisateur d'erreurs et de ladite formule de valeur d'erreur pour obtenir l'emplacement d'une erreur possible dans le mode de code et une valeur d'erreur correspondante, respectivement ;
G. (a) si ledit polynôme localisateur d'erreurs est égal à zéro pour une valeur de x donnée, indiquant qu'il y a une erreur à l'emplacement du mot de code correspondant à x, correction de l'erreur avec la valeur d'erreur calculée correspondante, ou
(b) si ledit polynôme localisateur d'erreurs n'est pas égal à zéro pour une valeur de x donnée, indiquant qu'il n'y a pas d'erreur à l'emplacement du mot de code correspondant à x, omission de la valeur d'erreur calculée correspondante ; et,
H. répétition des étapes C-G pour toutes les autres valeurs de x correspondant aux emplacements d'erreurs possibles dans le mot de code.

2. Procédé pour corriger des erreurs selon la revendication 1, dans lequel :

A. si le polynôme générateur a x^1 comme racine, ladite formule de valeur d'erreur pour déterminer la valeur d'une erreur, v, à l'emplacement du mode de code correspondant à x est :

$$v = \frac{x * \Phi(x)}{\delta_{impair}(x)}$$

et
B. si le polynôme générateur n'a pas x^1 comme racine, ladite formule de valeur d'erreur pour déterminer la valeur d'une erreur, v, à l'emplacement du mot de code correspondant à x est :

$$v = \frac{x^C * \Phi(x)}{\delta_{impair}(x)}$$

où C est une constante qui dépend du polynôme générateur.

3. Procédé pour corriger des erreurs selon la revendication 2, dans lequel le code de correction d'erreurs utilisé pour coder les données est un code de Read-Solomon.

4. Procédé pour corriger des erreurs selon la revendication 2, dans lequel ledit polynôme d'évaluation d'erreurs et les expressions localisatrices d'erreurs pour déterminer l'emplacement et la valeur d'erreurs dans un mot de code de données sont évalués par utilisation d'une méthode d'évaluation itérative comprenant les étapes de :

A. calcul des valeurs des termes des deux expressions localisatrices d'erreurs δ_{pair} (x) et δ_{impair} (x) et des valeurs des termes du polynôme évaluateur d'erreurs Φ (x) pour une première valeur de x correspondant à un emplacement de mot de code donné ;
B. enregistrement de la valeur calculée de chaque terme des expressions localisatrices d'erreurs et du polynôme évaluateur d'erreurs ;
C. substitution des valeurs enregistrées dans le polynôme localisateur d'erreurs et la formule de valeur d'erreur pour déterminer les emplacements et les valeurs d'erreurs dans le mot de code correspondant à x ;
D. pour essayer une valeur suivante de x correspondant à un autre emplacement d'erreur possible, multiplication de chaque valeur enregistrée par une constante B^n, où :

i. x * B est égal à la valeur suivante de x à essayer, et
ii.n est l'exposant de la variable dans le terme ayant cette valeur ; et,

E. répétition des étapes B-D pour chaque valeur de x correspondant à un emplacement d'erreur possible.

5. Dispositif pour corriger des données qui sont codées en mots de code par utilisation d'un code de correction d'erreurs BCH selon un Corps de Galois $GP(2^P)$, comprenant :

A. un générateur de syndromes d'erreur pour générer des syndromes d'erreur ;
B. un générateur de polynôme connecté pour recevoir les syndromes d'erreur pour générer, par utilisation des syndromes d'erreur, un polynôme évaluateur d'erreurs Φ (x) et un polynôme localisateur d'erreurs à (x) de la forme $\delta(x) = 1 + \delta_{pair}(x) + \delta_{impair}(x)$, où δ_{pair} (x) et δ_{impair} (x) sont les termes pair et impair de δ (x), respectivement ;
C. un moyen de correction d'erreurs de mot de code pour localiser les erreurs dans le mot de code et les corriger, ledit moyen de correction d'erreurs de mot de code incluant :

1. un premier moyen de substitution pour substituer simultanément x dans les expressions pour le polynôme évaluateur d'erreurs Φ (x), δ_{pair} (x) et δ_{impair} (x), pour chaque valeur successive de x, qui sont chacune des éléments de $GF(2^P)$ correspondant à des emplacements d'erreurs possibles dans le mot de code ;
2. un premier moyen d'évaluation pour évaluer lesdites expressions aux valeurs de x substitution ;
3. un second moyen de substitution pour substituer les valeurs des expressions évaluées dans le polynôme localisateur d'erreurs et une formule de valeur d'erreur ;
4. un second moyen d'évaluation pour évaluer ledit polynôme localisateur d'erreurs et ladite formule de valeur d'erreur afin d'obtenir les emplacements d'erreurs dans le mot de code et les valeurs d'erreur correspondantes ; et,
5. un moyen de correction pour :

(1) si ledit polynôme localisateur d'erreurs est égal à zéro, corriger l'erreur à l'emplacement du mot de code correspondant à x par utilisation de la valeur d'erreur calculée, ou
(2) si ledit polynôme localisateur d'erreurs n'est pas égal à zéro, omettre la valeur d'erreur calculée ;

6. un moyen de répétition des étapes 1-5 pour toutes les autres valeurs de x correspondant à des emplacements d'erreurs possibles dans le mot de code.

6. Dispositif pour corriger des erreurs selon la revendication 5, dans lequel ladite formule de valeur d'erreur dans laquelle le premier moyen de substitution substitue les valeurs d'expression calculées est :

$$\frac{x * \Phi(x)}{\delta_{impair}(x)}$$

si le polynôme générateur a x^1 comme plus petite racine, comme formule de valeur d'erreur, ou

$$\frac{x^C * \Phi(x)}{\delta_{impair}(x)}$$

si le polynôme générateur n'a pas x^1 comme plus petite racine, comme formule de valeur d'erreur, où C est une

* C L M S *

constante qui dépend du polynôme générateur.

7. Dispositif pour corriger des erreurs selon la revendication 6, dans lequel ledit premier moyen d'évaluation inclut :

1. moyen de calcul pour calculer les valeurs des termes des deux expressions localisatrices d'erreurs δ_{pair} (x) et δ_{impair} (x) et les valeurs des termes du polynôme évaluateur d'erreurs pour une première valeur de x correspondant à un emplacement de mot de code donné ;
2. un moyen d'enregistrement pour enregistrer les valeurs calculées de chacun des termes des expressions localisatrices d'erreurs et du polynôme évaluateur d'erreurs et pour appliquer lesdites valeurs audit second moyen de substitution ; et,
3. un moyen de multiplication pour calculer une valeur suivante de x correspondant à un autre emplacement d'erreur possible par multiplication de chacune des valeurs enregistrées par une constante B^n, où :

i. x * B est égal à la valeur suivante de x à essayer, et
ii. n est l'exposant de la variable dans le terme ayant cette valeur.