Module 6

Channel Coding

Lesson 34

Block Codes

After reading this lesson, you will learn about

- ***Hamming Code;***
- ***Hamming Decoder;***
- ***Reed-Solomon Codes;***

In this lesson, we will have short and specific discussions on two block codes, viz. the Hamming code and the Reed- Solomon code. The Hamming code is historically important and it is also very suitable for explaining several subtle aspects of encoding and decoding of linear binary block codes. The Reed- Solomon code is an important example of linear non-binary code, which can correct random as well as burst errors. It is one of the most extensively applied block codes in digital transmission an digital storage media.

Hamming Code

Invented in 1948 by R. W. Hamming, this single error correcting code is an elegant one. It should be easy to follow the encoding and decoding methods. Though the original code, developed by Hamming, was a specific (7,4) code with n = 7 and k = 4, subsequently all single error correcting linear block codes have come to be known as Hamming codes too.

In general, a Hamming code is specified as $\left(2^m-1, 2^m-m-1\right)$, i.e., a block of k = $\left(2^m-m-1\right)$ bits are encoded together to produce a codeword of length n = $\left(2^m-1\right)$bits. So, 'm' parity bits are added for every 'k' information bits. Some examples of Hamming codes are (7, 4), (15, 11), (31, 26) and (63, 57). All these codes are capable of detecting 2 errors and correcting single error as the Hamming distance for them is 3.

By adding an overall parity bit, an (n, k) Hamming code can be modified to yield an (n+1, k+1) code with minimum distance 4. Such a code is known as extended Hamming code.

We follow the polynomial approach to describe the (7,4) Hamming code. The generator polynomial for (7,4) Hamming code is:

g(x)= Generator polynomial = x^3+x+1 6.34.1

Here, the coefficients of x^i –s are elements of GF(2), i.e. the coefficients are '0' or '1'.

However, all the polynomials are defined over an 'extended' field of GF(2). The extension field for (7,4) Hamming code is GF (2^3). In general, the extension field, over which the generator polynomial is defined, for $\left(2^m-1, 2^m-m-1\right)$Hamming code is GF($2^m$).

A finite field of polynomials is defined in terms of i) a finite set of polynomials as its elements and ii) a clearly defined set of rules for algebraic operations. For convenience, the algebraic operations are often named as 'addition', 'subtraction', 'multiplication' and 'inverse'. It is also mandatory that there will be one identity element for addition (which is termed as '0') and one identity element for 'multiplication' (which

is termed as '1') in the set of polynomials. Note that '0' and '1' are viewed as polynomials in 'x', which is an indeterminate. The field is said to be finite as the result of any allowed algebraic operations one two or more elements will be a polynomial belonging to the same finite set of polynomials.

Interestingly, if we can identify a polynomial in 'x', say p(x) defined over GF(2), such that p(x) is a prime and irreducible polynomial with degree 'm', then we can define a finite field with 2^m polynomial elements, i.e., GF(2^m) easily. Degree of a polynomial is the highest exponent of 'x' in the polynomial.

For the (7,4) Hamming code, m =3 and a desired prime, irreducible p(x) = g(x) = $x^3 + x + 1$. **Table 6.34.1** shows one possible representation of GF(2^3). Note that all polynomials whose degree are less than m = 3, are elements of GF(2^3) and it takes only a group of 'm' bits to represent one element in binary form.

Field Elements	Polynomials	Possible binary representation
0	0	000
$\alpha^0 = 1 = \alpha^7$	1	001
α	X	010
α^2	X^2	100
α^3	$x^3 = x + 1$	011
α^4	$x^4 = x.x^3 = x^2 + x$	110
α^5	$x^5 = x.x^4 = x^2 + x + 1$	111
α^6	$x^6 = x.x^5 = x^2 + 1$	101

Table 6.34.1: *An illustration on the elements of GF(2^3). Note that, $x^7 = 1$, $x^8 = x$ and so on*

Let us note that, for the (7,4) Hamming code, the degree of a message polynomial m(x) ≤ 3, the degree of the generator polynomial g(x) = 3 = (n – k) and the degree of a codeword polynomial c(x) ≤ 6.

As an example, let us consider a block of 4 message bits (0111) where '0' is the last bit in the sequence. So, the corresponding message polynomial is, $m(x) = x^2 + x + 1$.
If we follow non-systematic coding, the codeword polynomial is:
$c(x) = m(x).g(x) = (x^2 + x + 1)(x^3 + x + 1) = x^5 + x^3 + x^2 + x^4 + x^2 + x + x^3 + x + 1$
$= x^5 + x^4 + (x^3 \oplus x^3) + (x^2 \oplus x^2) + (x \oplus x) + 1 = x^5 + x^4 + 1$
So in binary form, we get the code word (0110001).
Similarly, if the message polynomial is $m(x) = x^3 + x^2 + x + 1$, verify that the codeword is (1101001).

If we consider a general form of the message polynomial, $m(x) = m_3 x^3 + m_2 x^2 + m_1 x + m_0$ where $m_i \in$ GF(2), a general form of the code word is

$$c(x) = m_3 x^6 + m_2 x^5 + (m_3 \oplus m_1) x^4 + (m_3 \oplus m_2 \oplus m_0) x^3 + (m_2 \oplus m_1) x^2 + (m_1 \oplus m_0) x + m_0$$

6.34.2

The above expression can used for direct implementation of the encoder with five two-input Exclusive-OR gates.

If we wish to go for systematic encoding where the parity bits will be appended to the message bits, we use the following expression for the codeword polynomial as introduced in Lesson #33:

$$c'(x) = x^{n-k}m(x) - R_{g(x)}\left[x^{n-k}m(x)\right] \qquad 6.34.3$$

Now, m(x) = $x^2 + x + 1$.

$\therefore\ x^{n-k}.m(x) = x^3\,(x^2 + x + 1) = x^5 + x^4 + x^3$

Now, divide this by the generator polynomial g(x) as bellow and remember that addition and subtraction operations are the same over GF(2).

x^3+x+1) $x^5+x^4+x^3$ (x^2+x

$x^5+\ \ +x^3+x^2$

\-

$0+x^4+0.x^3+x^2$

$x^{4+}\qquad +x^2+x$

---

x

So, the remainder is x.

Therefore, the codeword in systematic form is:

$c'(x) = (x^5+x^4+x^3) + x$

=(0 1 1 1 0 1 0)

Message bits (0 1 1 1) — Parity bits (0 1 0)

Hamming Decoder

Upon receiving a polynomial r(x) = c(x) + e(x), the decoder computes the syndrome polynomial s(x) $= R_{g(x)}[r(x)]$. If the remainder is '0', the decoder declares the received word as a correct one. If s(x) is a non-zero polynomial, the decoder assumes that there is a single error in one of the 7 bits. In fact, the precise location of the erroneous bit is uniquely related to the syndrome polynomial. Note that the degree of the syndrome polynomial is 2 or less. So, the number of non-zero syndrome polynomials is $(2^3-1) = 7$. Each polynomial form indicates a single error at a bit position. Once the location of the erroneous bit is found from the syndrome polynomial, the bit is simply inverted to obtain the correct transmitted codeword. For a systematic code, the information bits can be retrieved easily from the corrected

codeword. For a nonsystematic code, another division operation is necessary. However, for practical implementation, one can use a Look Up Table (LUT) for the purpose.

Reed-Solomon Codes

Reed Solomon (R-S) codes form an important sub-class of the family of Bose-Chaudhuri-Hocquenghem (BCH) codes and are very powerful linear non-binary block codes capable of correcting multiple random as well as burst errors. They have an important feature that the generator polynomial and the code symbols are derived from the same finite field. This enables to reduce the complexity and also the number of computations involved in their implementation. A large number of R-S codes are available with different code rates. A few R-S codes with their code parameters are shown in **Table 6.34.2.**

Code No.	**Code Name**	**Block Length (n)**	**No.of information symbols in a code word (k)**	**Error correcting power (t)**	**Code Rate (R)**	**Field of Definition**
1	(31, 27) R – S	31	27	2	0.871	
2	(31, 21) R – S	31	21	5	0.677	
3	(31, 15) R – S	31	15	8	0.484	$GF(2^5)$
4	(31, 11) R – S	31	11	10	0.355	
5	(63, 55) R – S	63	55	4	0.873	
6	(63, 47) R – S	63	47	8	0.746	
7	(63, 39) R – S	63	39	12	0.619	$GF(2^6)$
8	(63, 31) R – S	63	31	16	0.492	
9	(63, 23) R – S	63	23	20	0.365	
10	(63, 15) R – S	63	15	24	0.238	
11	(255, 233) R – S	255	233	11	0.913	
12	(255, 225) R – S	255	225	15	0.882	
13	(255, 205) R – S	255	205	25	0.804	
14	(255, 191) R – S	255	191	32	0.749	$GF(2^8)$
15	(255, 183) R – S	255	183	36	0.718	
16	(255, 175) R – S	255	175	40	0.686	
17	(255, 165) R – S	255	165	45	0.647	
18	(255, 135) R – S	255	135	60	0.529	

Table 6.34.2 *Parameters of a few selected Reed Solomon codes*

Like other forward error correcting (FEC) decoders the decoding of the R-S codes is more complex than their encoding operation. An iterative algorithm due to E.R Berlekamp with J. L. Massey's extension, a search algorithm due to R. T. Chien and G. D. Forney's method for calculation for error values, together constitute a basic approach for decoding the R-S codes.

An R-S code is described by a generator polynomial g(x) and other usual important code parameters such as the number of message symbols per block (k), number of code symbols per block (n), maximum number of erroneous symbols (t) that can surely be corrected per block of received symbols and the designed minimum symbol Hamming distance (d). A parity-check polynomial h (X) of order k also plays a role in designing the code. The symbol x, used in polynomials is an indeterminate which usually implies unit amount of delay. Even though the R-S codes are non-binary in construction and defined over GF(2^m), the symbols can be expressed and processed in groups of 'm' bits.

For positive integers m and t, a primitive (n, k, t) R-S code is defined as below:
Number of encoded symbols per block: $n = 2^m - 1$
Number of message symbols per block: k
Code rate: R= k/n
Number of parity symbols per block: $n - k = 2t$
Minimum symbol Hamming distance per block: $d = 2t + 1$.

It can be noted that the block length n of an (n, k, t) R-S code is bounded by the corresponding finite field GF(2^m). Moreover, as $n - k = 2t$, an (n, k, t) R-S code has optimum error correcting capability. As we have seen earlier in this lesson, each field elements can be represented by m bits. So, a codeword of (n,k,t) R-S code over GF (2^m), consisting of n symbols, can be represented by $(n \times m)$ bits and the code has sufficient inherent algebraic structure to correct at least t erroneous symbols per block. If the number of erroneous symbols per block exceeds t, there is no guarantee that the errors will be corrected. Practical decoders in such cases behave differently depending on the error pattern. Possibility of correcting more than t erroneous symbols, for some error pattern, also exists. The t erroneous symbols, when represented by m bits each, can be affected in two extreme cases in the following manner:

(i) one bit of each of the t-symbols is corrupted.
(ii) all m bits of each of the t-symbols are corrupted.

So, the random error correcting capability of an (n,k,t) R-S code is up to at least t bits per block of $(n \times m)$ bits. When errors due to a communication channel are not totally uncorrelated, i.e. the errors tend to be bursty in nature, the use of R-S code becomes more relevant and important. Note that a particular symbol, consisting of m bits, may be erroneous due to error in one or more (up to m) bits and the code ensures correct decoding of up to t such erroneous symbols. Thus if there are $(m \times t)$ or less erroneous bits, distributed in t symbols only, the code corrects the erroneous symbols without failure.

The generator polynomial g(X) for the (n, k, t) R-S code is written as,

$$g(x) = \prod_{i=1}^{2t} \left(x - \alpha^i\right) \qquad 6.34.4$$

Here 'α' is the primitive root of an irreducible polynomial of degree 'm' over GF(2) and 'α' is defined over GF(2^m),the extension field. We have briefly discussed about the definition of GF(2^m) earlier in this lesson.

Let us consider an example of an R-S code:

Example #6.34.1: Let the allowable bandwidth expansion for error correcting code alone be about 50% and let the hardware complexity permit us to consider block length of 31.

For such a situation one can think of an R-S code with following parameters:

$N = 31 = 2^5 - 1$; $m = 5$;

Based on the allowed bandwidth expansion, the approximate code rate is 2/3.

So, we can choose $k = 21$; $R = 21/31 = 0.678$.

$n-k = 2t = 10$; $t = 5$ and $d = 2t + 1 = 11$.

Thus the chosen code is a (31, 21, 5) R-S code.

The generator polynomial g(x) of this code is given below:

$$g(X) = \prod_{i=1}^{10}\left(X + a^i\right)$$

$$= X^{10} + a^{18}X^9 + X^8 + a^{25}X^7 + a^{10}X^6 + a^9X^5 + a^{12}X^4 + a^{16}X^3 + a^2X^2 + X + a^{24} \qquad 6.34.5$$

Here, 'α' is the primitive root of the irreducible polynomial $p(x) = x^5 + x^2 + 1$ of degree 5 and defined over GF (2)

■

A general block diagram of a digital communication system employing Reed-Solomon forward error correcting codec is shown in **Fig. 6.34.1**. Serial message bits at the output of the source encoder are de-multiplexed suitably and groups of such m bits are fed to the R-S encoder at a time. Let the polynomial m (x) of degree (k – 1) or less denote the message polynomial and the polynomial $rem(x)$ of degree (n – k – 1) or less denote the remainder parity polynomial.

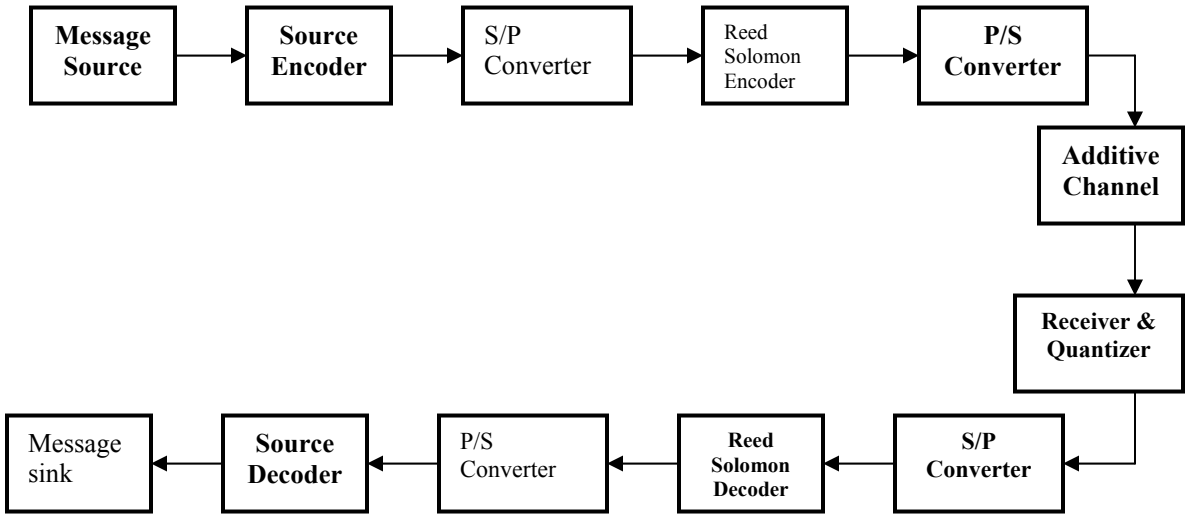

Fig 6.34.1 *Block diagram of a digital communication system using Reed-Solomon codes for error control*

We can write the encoded codeword polynomial v (x) of degree (n – 1) or less in the following form:

$$v(x) = \sum_{i=0}^{n-k} v_i x^i = x^{n-k}.m(x) + rem(x) = q(x).g(x) \qquad 6.34.6$$

Here q (X) is a polynomial of degree (k – 1) or less. The coefficients of all the polynomials are elements of GF(2^m). The parity-check polynomial $rem(x)$ can be found as the remainder polynomial when the message polynomial, shifted by (n – k) times, is divided by the generator polynomial g(x). **Fig.6.34.2** shows a block diagram of a systematic Reed-Solomon encoder employing (n – k) stage shift registers. Each stage consists of 'm' set of delay units. The encoder outputs m bits at a time and this set of m bits are multiplexed suitably depending on the front-end modulation technique employed.

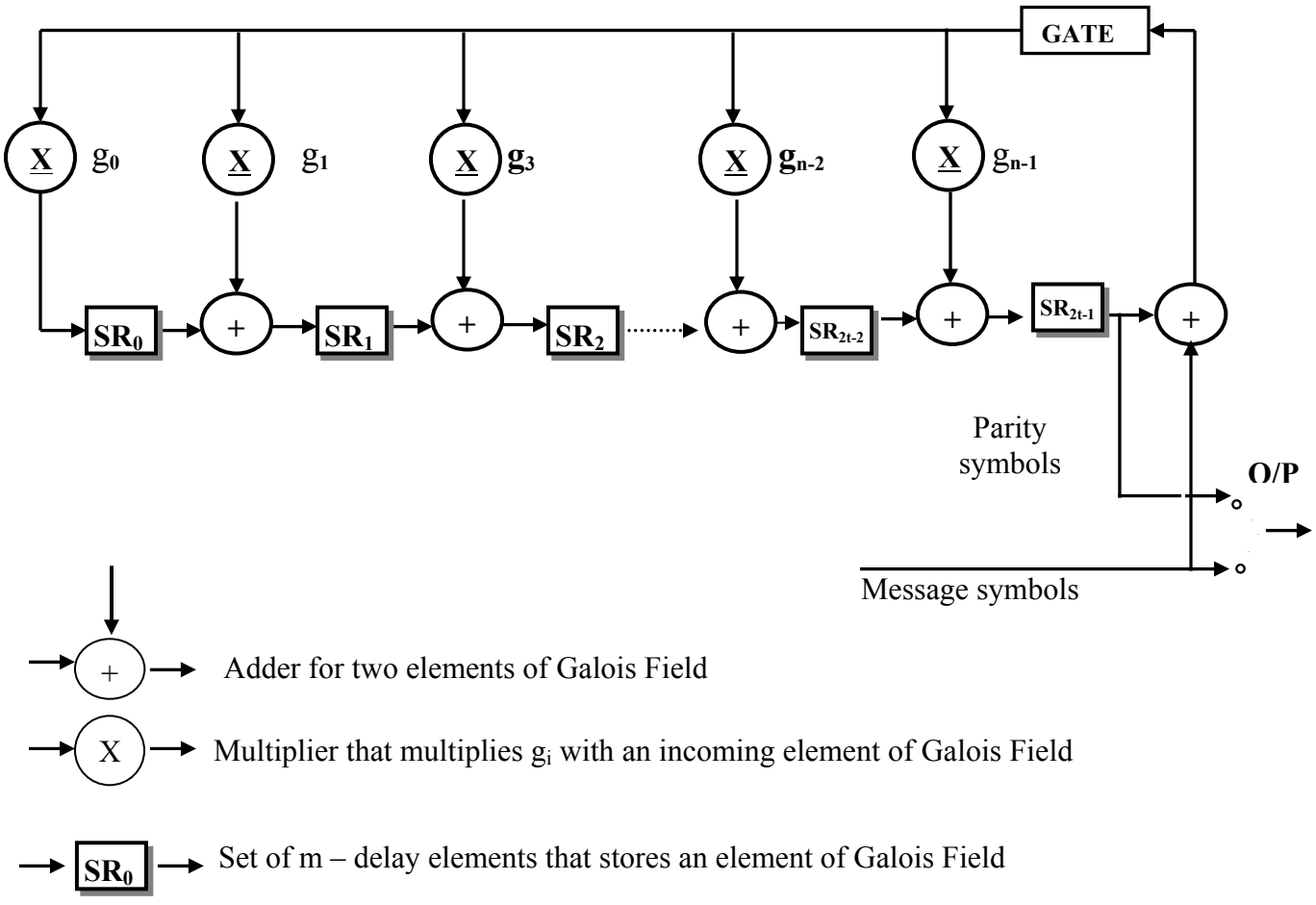

Fig 6.34.2 *Reed-Solomon encoder employing (n – k) shift register stages*

At the input-end of the R-S decoder in the receiver path in **Fig. 6.34.1**, hard-quantization of received signal has been assumed. Now, due to noisy transmission channel or receiver imperfection or constraints in other system parameters (e.g. transmitter power limitation, intentional degradation of difficult-to-maintain system components etc.) the code symbols received by the R-S decoder may be erroneous. The task of the decoder is to recover correct set of message symbols efficiently from these received symbols. A logical approach towards this is outlined in the form of a flow chart shown in **Fig. 6.34.3**. The received code symbols are represented sequentially by the coefficients of the received code polynomial f (x). This polynomial can be considered as the sum of the transmitted code polynomial v(x) and the error polynomial e(x). Since v(x) is a valid code polynomial, it is divisible by the generator polynomial g(x).

Moreover, as $g(x) = \prod_{i=1}^{2t}\left(x - \alpha^i\right)$ it can be easily seen that, $v(a^j) = 0$, for $1 \le j \le 2t$

so, we can write, for$1 \le j \le 2t$,

$$f(a^j) = \sum_{i=0}^{n-1} f_t (a^j)^i \; = v(a^j) + e(a^j) = 0 + e(a^j)$$

This shows that the quantities $f\left(\alpha^j\right)$ contain information about the error polynomial e(x) only. These quantities, named syndromes, are only 2t in number and they play pivotal role in the decoding process.

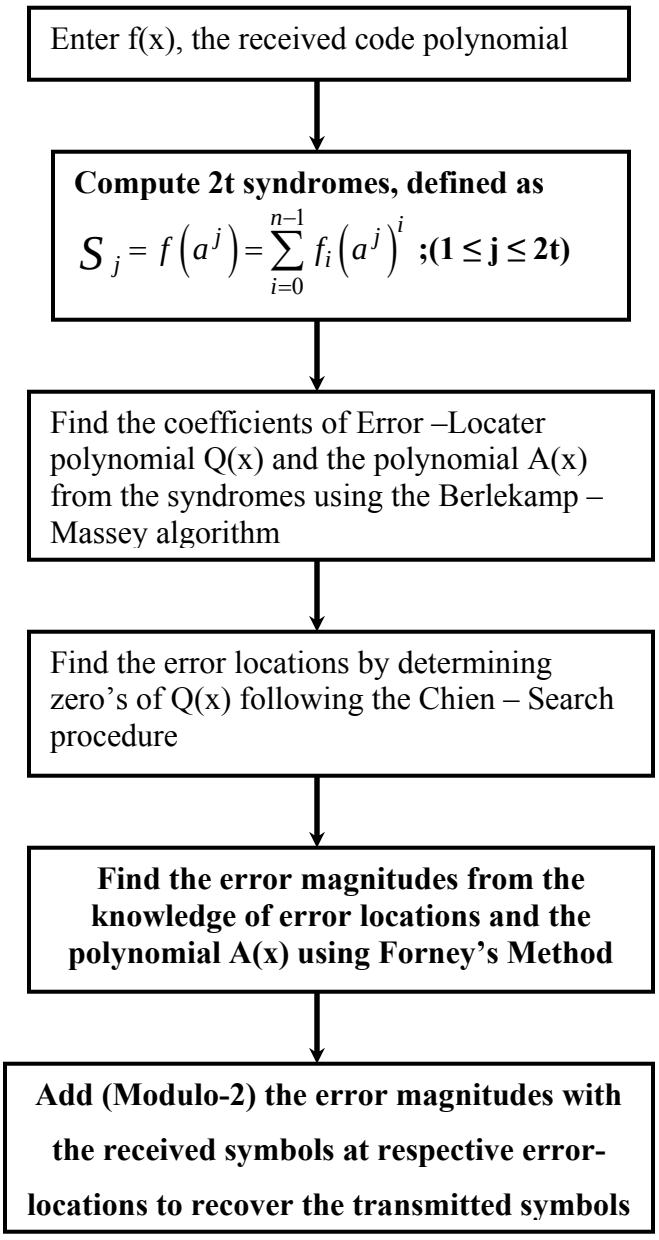

Fig 6.34.3 *Major steps in the decoding of the Reed-Solomon codes*

After determining the syndromes, the next step is to find the coefficients of a suitably defined error-locator polynomial Q(x). This is carried out following the Berlekamp-Massey algorithm. The coefficients of the error-locator polynomial are necessary for finding the error-locations which is done following the cyclic Chien search

procedure in the third step. The next task of finding the error magnitudes is performed efficiently using Forney's method. The remaining task is to subtract the error polynomial from the received polynomial and then extract the message polynomial from the corrected codeword.