Chapter 6
Reed-Solomon Codes

1. Introduction

- **The Reed-Solomon codes (RS codes) are nonbinary cyclic codes with code symbols from a Galois field. They were discovered in 1960 by I. Reed and G. Solomon. The work was done when they were at MIT Laboratory.**

- **In the decades since their discovery, RS codes have enjoyed countless applications from compact discs and digital TV in living room to spacecraft and satellite in outer space.**

- **The most important RS codes are codes with symbols from GF(2^m).**

- **One of the most important features of RS codes is that the minimum distance of an (*n, k*) RS code is *n–k+1*.**

  Codes of this kind are called "maximum-distance-separable codes".

2. RS Codes with Symbols from GF(2^m)

- **Let α be a primitive element in GF(2^m).**

- **For any positive integer $t \le 2^m - 1$, there exists a *t*-symbol-error-correcting RS code with symbols from GF(2^m) and the following parameters:**

$$\begin{aligned} n &= 2^m - 1 \\ n - k &= 2t \\ k &= 2^m - 1 - 2t \\ d_{\min} &= 2t + 1 = n - k + 1 \end{aligned} \qquad (6\text{- }1)$$

- **The generator polynomial is**

$$g(x) = (x+\alpha)(x+\alpha^2)\cdots(x+\alpha^{2t})$$
$$= g_0 + g_1 x + g_2 x^2 + \cdots + g_{2t-1} x^{2t-1} + x^{2t} \quad \text{(6-2)}$$

Where $g_i \in \mathrm{GF}(2^m)$

Note that $g(x)$ **has** $\alpha, \alpha^2, \cdots, \alpha^{2t}$ **as roots.**

Example:

$$m = 8, t = 16$$
$$n = 255$$
$$k = n - 2t = 223$$
$$d_{\min} = 33$$

It is a (255, 223) RS code. This code is NASA standard code for satellite and space communications.

3. Encoding of RS Codes

- **Let $m(x) = m_0 + m_1 x + \cdots + m_{k-1} x^{k-1}$ be the message polynomial to be encoded**

  Where $m_i \in \mathrm{GF}(2^m)$ and $k = n - 2t$.

- **Dividing $x^{2t} m(x)$ by $g(x)$, we have**

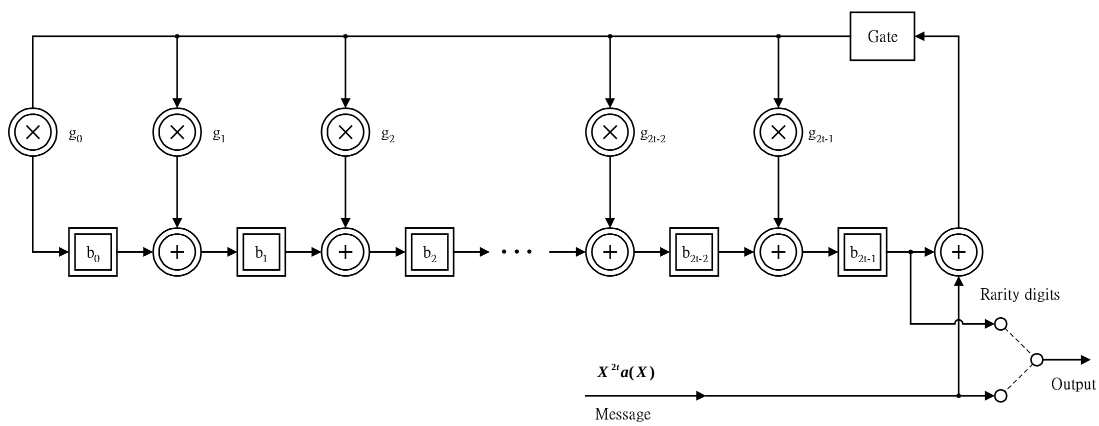
  $$x^{2t} m(x) = a(x) g(x) + b(x) \qquad (6\text{- }3)$$

  where $$b(x) = b_0 + b_1 x + \cdots + b_{2t-1} x^{2t-1} \qquad (6\text{- }4)$$

  is the remainder.

  Then $b(x) + x^{2t} m(x)$ is the codeword polynomial for the message $m(x)$.

- **The encoding circuit is shown below: (Lin / Costello p.172)**



Encoding circuit for a nonbinary cyclic code.

4. RS Codes for Binary Data

- **Every element in GF(2^m) can be represented uniquely by a binary *m*-tuple, called a *m*-bit byte.**

- **Suppose an (*n, k*) RS code with symbols from GF(2^m) is used for encoding binary data. A message of *km* bits is first divided into *k* *m*-bit bytes. Each *m*-bit byte is regarded as a symbol in GF(2^m). The *k*-byte message is then encoded into *n*-byte codeword based on the RS encoding rule.**

- **By doing this, we actually expand a RS code with symbols from GF(2^m) into a binary (*nm*, *km*) linear code, called a binary RS code.**

- **Binary RS codes are very effective in correcting bursts of bit errors as long as no more than *t* bytes are affected.**

5. Decoding of RS Codes

- **RS codes are actually a special class of nonbinary BCH codes.**

- **Decoding of a RS code is similar to the decoding of a BCH code except an additional step is needed.**

- **Let** $c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}, c_i \in \mathrm{GF}(2^m)$

  and $r(x) = r_0 + r_1 x + \cdots + r_{n-1} x^{n-1}, r_i \in \mathrm{GF}(2^m)$

  Then the error polynomial is

  $$\begin{aligned} e(x) &= r(x) - c(x) \\ &= e_0 + e_1 x + e_2 x^2 \cdots + e_{n-1} x^{n-1} \end{aligned} \tag{6- 5}$$

  Where $e_i = r_i - c_i$ **is a symbol in GF(** 2^m **).**

- **Suppose** $e(x)$ **has** ν **errors at the locations** $x^{j_1}, x^{j_2}, \cdots, x^{j_\nu}$**, then**

  $$e(x) = e_{j_1} x^{j_1} + e_{j_2} x^{j_2} + \cdots + e_{j_\nu} x^{j_\nu} \tag{6- 6}$$

  The error-location numbers are

  $$Z_{j_1} = \alpha^{j_1}, Z_{j_2} = \alpha^{j_2}, \cdots, Z_{j_\nu} = \alpha^{j_\nu}$$

  The error values are

  $$e_{j_1}, e_{j_2}, \cdots, e_{j_\nu}$$

- **Thus, in decoding a RS code, we not only have to determine the error-locations but also have to evaluate the error values.**

- **If there are *s* erasure symbols and *v* errors in the received polynomial *r*(*x*), then the (*n*, *k*) RS decoder and correct these erasure symbols and errors if** $2v + s \le d - 1 = n - k$

  The received polynomial is represented by

  $$r(x) = c(x) + e(x) + e^*(x) = c(x) + u(x)$$

  Where *e*(*x*) and are *e(*x*) are the error and erasure polynomials, respectively.**

6. Errors-only Decoding of RS Codes

- **For errors-only decoding the received polynomial $r(x)$ has the simply form, $r(x)=c(x)+e(x)$**

  where $e(x)=e_0+e_1x+e_2x^2\cdots+e_{n-1}x^{n-1},\quad e_i\in \mathrm{GF}(2^m)$

  For $e_i\neq 0$**, let** e_i **denote the error value at the *i*-th position.**

- **Suppose there are** ν **errors, where** $2\nu\leq n-k$ **at positions** i_w **for** $w=1,2,\cdots,\nu$.

  The objective of the RS decoder is to find the number of errors, their positions and then their values.

7. Syndrome Computation

- **Let** $r(x) = r_0 + r_1 x + \cdots + r_{n-1} x^{n-1}$ **(6-7)**

  The generator polynomial has $\alpha, \alpha^2, \cdots, \alpha^{2t}$ **as roots.**

  Since $c(\alpha^i) = m(\alpha^i) g(\alpha^i), \quad i = 1,2,\cdots,2t$

  We have the relations

$$r(\alpha^i) = c(\alpha^i) + e(\alpha^i) = e(\alpha^i) = \sum_{j=0}^{n-1} e_j \alpha^{ij} \qquad \textbf{(6- 8)}$$

- **The syndrome of the received polynomial is**

$$\overline{S} = (S_1, S_2, \cdots, S_{2t})$$

  where $S_i = r(\alpha^i)$

- **The syndrome can be obtained by using the relationship**

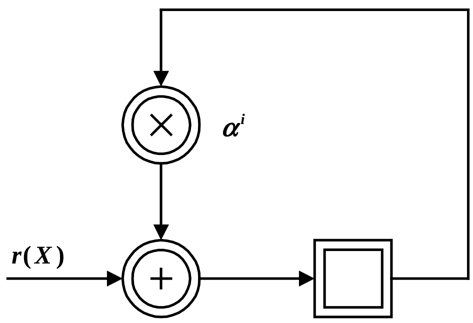
$$r(x) = a(x)(x + \alpha^i) + b_i$$

  where $b_i = \mathrm{GF}(2^m)$

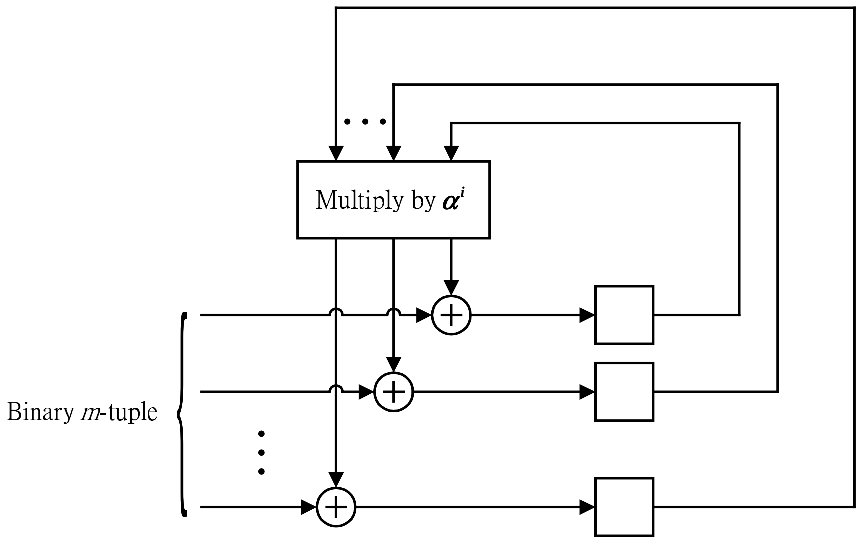
  Thus, $S_i = r(\alpha^i) = b_i$

- **The syndrome computation circuit is shown below.**

  (Lin /Costello p. 174)



(a)



(b)

Figure 6.14 Syndrome computation circuits for Reed-Solomon codes:

(a) over GF(2^m); (b) in binary form.

8. Determination of Error-Location Polynomial

- **Peterson Algorithm can be used to determine the error-locator polynomial for small number of errors in *r*(*x*).**

  However, the complexity of the Peterson-type algorithm is $O(n^3)$.

- **In 1967, E. Berlekamp demonstrated an extremely efficient algorithm for both BCH and RS codes.**

  Berlekamp's algorithm allowed for the first time the possibility of a quick and efficient decoding of dozens of symbol errors in some powerful RS codes.

  Ref:

(1) **E. R. Berlekamp, "On Decoding Binary Bose-Chaudhuri-Hocquenghem Codes", *IEEE Trans. Information Theory*, pp. 577-580, Oct. 1965.**

(2) **E. R. Berlekamp, "Nonbinary BCH Decoding", *Int. Symposium on Information Theory*, Italy, 1967.**

(3) **E. R. Berlekamp, Algebraic Coding Theory, *McGraw Hill*, New York, 1968.**

9. Berlekamp's Iterative Method for Finding $L(z)$ [Chen / Reed pp. 263-269]

- **The Berlekamp-Massey algorithm is an efficient algorithm for determining the error-locator polynomial.**

- **The algorithm solves the following modified problem:**

  Find the smallest ν for $\nu \le 2t$ such that the system of eq. (5-19) has a solution and find a vector $(\Lambda_1, \Lambda_2, \cdots, \Lambda_\nu)$ that is a solution.

  Eq. (5-19) can be expressed as

$$S_j = -\sum_{i=1}^{\nu} \Lambda_i S_{j-i} \quad \text{for} \quad j = \nu+1, \nu+2, \cdots, 2t \qquad \text{(6- 9)}$$

- **For a fixed $\Lambda(x)$, eq. (6-9) is the equation for the classical auto-regressive filter.**

  It is known that such a filter can be implemented as a linear-feedback shift register (LFSR) with taps, given by the coefficients of $\Lambda(x)$.

- **The design procedure is iterative.**

For each *i*, starting with *i* = 1, a minimum length LFSR $\Lambda^{(i)}(x)$ **is designed for producing the syndrome components** $S_1, S_2, \cdots, S_i$**. This shift register for stage *i* need not be unique and several choices may exist.**

$$\Lambda(x) = 1 + \Lambda_1 x + \Lambda_2 x^2 + \cdots + \Lambda_\nu x^\nu$$

The length L_i **of this shift register can be greater than the degree of** $\Lambda^{(i)}(x)$**.**

A shift register is designated by the pair $(\Lambda^{(i)}(x), L_i)$**. At the start of the *i*-th iteration, it is assumed that a list of LFSR had been constructed,**

i.e. the list $(\Lambda^{(1)}(x), L_1), (\Lambda^{(2)}(x), L_2), \cdots, (\Lambda^{(i-1)}(x), L_{i-1})$ **for** $L_1 \le L_2 \le \cdots \le L_{i-1}$ **already is found.**

The operation principle of the Berlekamp-Massey algorithm is iterative and inductive:

At the iteration step *i*, compute the next output of the (*i*-1)-th LFSR to obtain the next estimate of the *i*-th syndrome as follows:

$$\hat{S}_i = -\sum_{j=1}^{L_{i-1}} \Lambda_j^{(i-1)} S_{i-j} \qquad \text{(6-10)}$$

Next, subtract this “estimated” $\hat{S}_i$ from the desired syndrome output S_i to get an “error” quantity Δ_i, that is called the *i*-th discrepancy, i.e.

$$\Delta_i = S_i - \hat{S}_i = S_i + \sum_{j=1}^{L_{i-1}} \Lambda_j^{(i-1)} S_{i-j} \qquad \text{(6-11)}$$

or, equivalently,

$$\Delta_i = \sum_{j=0}^{L_{i-1}} \Lambda_j^{(i-1)} S_{i-j} \qquad \text{(6- 12)}$$

If $\Delta_i = 0$, then set $(\Lambda^{(i)}(x), L_i) = (\Lambda^{(i-1)}(x), L_{i-1})$ and the *i*-th iteration is complete.

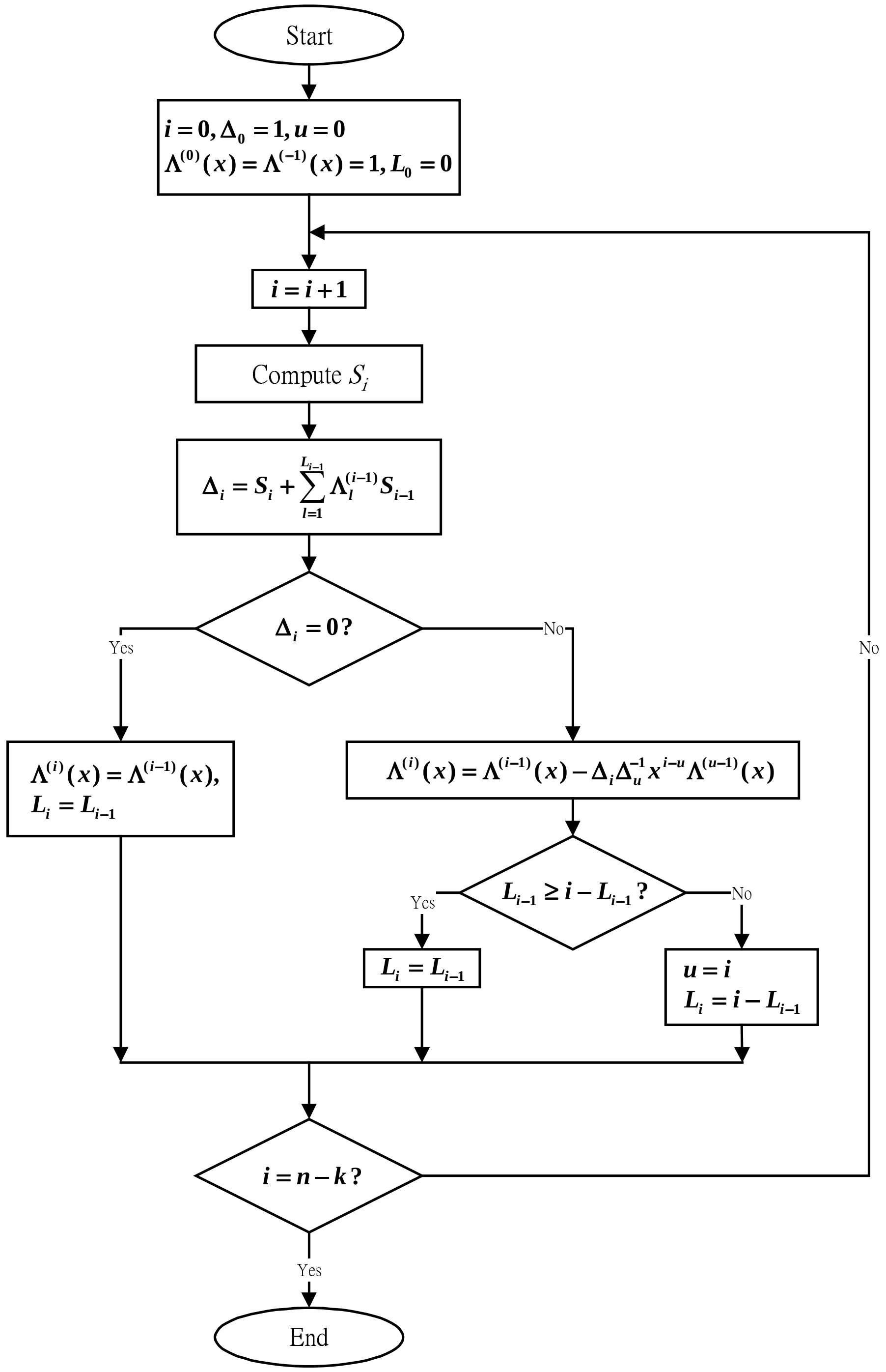
Otherwise, the LFSR are modified as follows:

$$\Lambda^{(i)}(x) = \Lambda^{(i)}(x) + \Delta_i A(x) = \sum_{l=0}^{L_1} (\Lambda_l^{(i-1)} + \Delta_i a_l) x^l \qquad \text{(6- 13)}$$

for some polynomial $A(x) = \sum_{l=0}^{L_i} a_l x^l$ yet to be found, where L_i is specified by a mathematical lemma [demonstrated in reference [10] by J. L. Massey, 1969].

The length of $A(x)$ is $L_u + i - u$. This length is minimum if $i - u$ has the smallest value. This happens only if u is the most recent step for $u < i$ such that $\Delta_u \neq 0$ and $L_u > L_{u-1}$.

- **A flowchart of the Berlehamp-Massey algorithm for the errors-only decoding of (nonbinary & binary) BCH codes as RS codes is shown in Fig. 6.7.**



<u>Example</u>: (Example 6.9 p.269)

For the (15, 9) RS code over GF(2^4)

$$m = 4, \quad \alpha^{15} = 1$$
$$r(x) = x^8 + \alpha^{11}x^7 + \alpha^8 x^5 + \alpha^{10}x^4 + \alpha^4 x^3 + \alpha^3 x^2 + \alpha^8 x + \alpha^{12}$$
$$S_1 = 1$$
$$S_2 = 1$$
$$S_3 = \alpha^5$$
$$S_4 = 1$$
$$S_5 = 0$$
$$S_6 = \alpha^{10}$$

Use the Berlekamp-Massey algorithm to find the error-locator polynomial.

Solution: $n - k = 6$

For $i = 0$

$$\Delta_0 = 1, \quad u = 0 \quad \Lambda^{(-1)}(x) = 1 \quad \Lambda^{(0)}(x) = 1$$
$$L_0 = 0$$

For $i = 1$

$$\Delta_1 = S_1 + \sum_{l=0}^{L_{i-1}} \Lambda_l^{(i-1)} S_{i-l} = S_1 = 1$$

$$\begin{aligned}\Lambda^{(1)}(x) &= \Lambda^{(0)}(x) - \Delta_1 \Delta_0^{-1} x^{(1-0)} \Lambda^{(0-1)}(x) \\ &= 1 - x\Lambda^{(-1)}(x) \\ &= 1 - x \\ &= 1 + x\end{aligned}$$

$$L_0 < 1 - L_0 \quad \text{then} \quad u = 1$$

$$\begin{aligned}L_1 &= 1 - L_0 \\ &= 1 - 0 \\ &= 1\end{aligned}$$

For $i = 2$

$$S_2 = 1 \qquad i - 1 = 1$$

$$\begin{aligned}\Delta_2 &= S_2 + \Lambda_1^{(1)} S_1 \\ &= S_2 + 1 \\ &= 1 + 1 \\ &= 0\end{aligned}$$

Note that $\Lambda^{(1)}(x) = 1 + x$, $\Lambda_1^{(1)} = 1$

$$\Lambda^{(2)}(x) = \Lambda^{(1)}(x) = 1 + x$$

$$L_2 = L_1 = 1$$

$u = 1$ **(unchanged)**

For $i = 3$

$$S_3 = \alpha^5 \qquad i-1=2 \qquad L_{i-1} = L_2 = 1$$

$$\Delta_3 = S_3 + \Lambda_1^{(2)} S_2$$
$$= \alpha^5 + S_2$$
$$= 1 + \alpha^5$$
$$= \alpha^{10} \qquad \text{(From Table 2.5 p.65)}$$

$$\Lambda^{(3)}(x) = \Lambda^{(2)}(x) - \Delta_3 \Delta_1^{-1} x^{3-1} \Lambda^{(0)}(x)$$
$$= 1 + x - \alpha^{10} x^2$$
$$= 1 + x + \alpha^{10} x^2$$

$$\because L_2 < 3 - L_2$$
$$L_3 = 3 - L_2 = 2$$
$$u = 3$$

For $i = 4$

$$S_4 = 1 \qquad i-1=3 \qquad L_{i-1} = L_3 = 2$$

$$\Delta_4 = S_4 + \Lambda_1^{(3)} S_3 + \Lambda_1^{(2)} S_2$$
$$= S_4 + S_3 + \alpha^{10} S_2$$
$$= 1 + \alpha^5 + \alpha^{10}$$
$$= 0$$

$$\Lambda^{(4)}(x) = \Lambda^{(3)}(x) = 1 + x + \alpha^{10} x^2$$

$$L_4 = L_3 = 2$$

For $i = 5$

$$S_5 = 0 \qquad i-1=4 \qquad L_4 = 1$$

$$\Lambda^{(4)}(x) = 1 + x + \alpha^{10}x^2$$

$$\begin{aligned}\Delta_5 &= S_5 + \Lambda_1^{(4)}S_4 + \Lambda_2^{(4)}S_3 \\ &= 0 + S_4 + \alpha^{10}\alpha^5 \\ &= 1 + \alpha^{15} \\ &= 1 + 1 \\ &= 0\end{aligned}$$

$$\Lambda^{(5)}(x) = \Lambda^{(4)}(x) = 1 + x + \alpha^{10}x^2$$

$$L_5 = L_4 = 2$$

For $i = 6$

$$S_6 = \alpha^{10} \qquad i-1=5 \qquad L_{i-1} = L_5 = 2$$

$$\begin{aligned}\Delta_6 &= S_6 + \Lambda_1^{(5)}S_5 + \Lambda_2^{(5)}S_4 \\ &= \alpha^{10} + 0 + \alpha^{10}\cdot 1 \\ &= 0\end{aligned}$$

$$\Lambda^{(6)}(x) = \Lambda^{(5)}(x) = 1 + x + \alpha^{10}x^2$$

Thus we have $\Lambda(x) = 1 + x + \alpha^{10}x^2$