

Channel Coding

題目：

Implementation of Galois field arithmetic

報告者：

林佳慶 8911604

陳燦琳 8911617

目的：

1. 比起原有的計算方式，因為 Galois Field 的計算並不會產生 carrier，因此計算過程中，可以減少其設計的複雜度。
2. 對於原有的乘法運算，在 Galois Field 中只需要簡單的加法器和 shift registers 即可完成。
3. 組成 Galois Field 的基底具有 cyclic 的特性。
4. Example: Irreducible polynomial (Fig. 1)

Consider a degree of 5 polynomial: 1 45 E

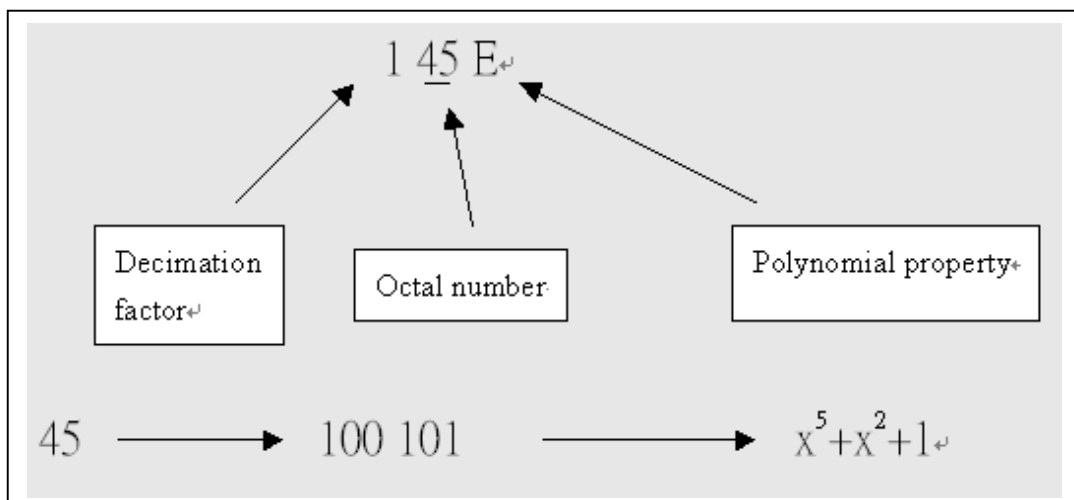


Table C.2. Irreducible Polynomials of Degree ≤ 12 over $GF(2)$.

DEGREE 2	1 7H				
DEGREE 3	1 13F				
DEGREE 4	1 23F	3 37D	5 07		
DEGREE 5	1 45E	3 75G	5 67H		
DEGREE 6	1 103F	3 127B	5 147H	7 111A	9 015
11 155E	21 007				
DEGREE 7	1 211E	3 217E	5 235E	7 367H	9 277E
11 325G	13 203F	19 313H	21 345G		
DEGREE 8	1 435E	3 567B	5 763D	7 551E	9 675C
11 747H	13 453F	15 727D	17 023	19 545E	21 613D
23 543F	25 433B	27 477B	37 537F	43 703H	45 471A
51 037	85 007				
DEGREE 9	1 1021E	3 1131E	5 1461G	7 1231A	9 1423G
11 1055E	13 1167F	15 1541E	17 1333F	19 1605G	21 1027A
23 1751E	25 1743H	27 1617H	29 1553H	35 1401C	37 1157F
39 1715E	41 1563H	43 1713H	45 1175E	51 1725G	53 1225E
55 1275E	73 0013	75 1773G	77 1511C	83 1425G	85 1267E
DEGREE 10	1 2011E	3 2017B	5 2415E	7 3771G	9 2257B
11 2065A	13 2157F	15 2653B	17 3515G	19 2773F	21 3753D
23 2033F	25 2443F	27 3573D	29 2461E	31 3043D	33 0075C
35 3023H	37 3543F	39 2107B	41 2745E	43 2431E	45 3061C
47 3177H	49 3525G	51 2547B	53 2617F	55 3453D	57 3121C
59 3471G	69 2701A	71 3323H	73 3507H	75 2437B	77 2413B
83 3623H	85 2707E	87 2311A	89 2327F	91 3265G	93 3777D
99 0067	101 2055E	103 3575G	105 3607C	107 3171G	109 2047F
147 2355A	149 3025G	155 2251A	165 0051	171 3315C	173 3337H
179 3211G	341 0007				
DEGREE 11	1 4005E	3 4445E	5 4215E	7 4055E	9 6015G
11 7413H	13 4143F	15 4563F	17 4053F	19 5023F	21 5623F
23 4757B	25 4577F	27 6233H	29 6673H	31 7237H	33 7335G
35 4505E	37 5337F	39 5263F	41 5361E	43 5171E	45 6637H
47 7173H	49 5711E	51 5221E	53 6307H	55 6211G	57 5747F
59 4533F	61 4341E	67 6711G	69 6777D	71 7715G	73 6343H
75 6227H	77 6263H	79 5235E	81 7431G	83 6455G	85 5247F
87 5265E	89 5343B	91 4767F	93 5607F	99 4603F	101 6561G
103 7107H	105 7041G	107 4251E	109 5675E	111 4173F	113 4707F
115 7311C	117 5463F	119 5755E	137 6675G	139 7655G	141 5531E
147 7243H	149 7621G	151 7161G	153 4731E	155 4451E	157 6557H
163 7745G	165 7317H	167 5205E	169 4565E	171 6765G	173 7535G
179 4653F	181 5411E	183 5545E	185 7565G	199 6543H	201 5613F
203 6013H	205 7647H	211 6507H	213 6037H	215 7363H	217 7201G
219 7273H	293 7723H	299 4303B	301 5007F	307 7555G	309 4261E
331 6447H	333 5141E	339 7461G	341 5253F		
DEGREE 12	1 10123F	3 12133B	5 10115A	7 12153B	9 11765A
11 15647E	13 12513B	15 13077B	17 16533H	19 16047H	21 10065A
23 11015E	25 13377B	27 14405A	29 14127H	31 17673H	33 13311A
35 10377B	37 13565E	39 13321A	41 15341G	43 15053H	45 15173C
47 15621E	49 17703C	51 10355A	53 15321G	55 10201A	57 12331A
59 11417E	61 13505E	63 10761A	65 00141	67 13275E	69 16663C
71 11471E	73 16237E	75 16267D	77 15115C	79 12515E	81 17545C
83 12255E	85 11673B	87 17361A	89 11271E	91 10011A	93 14755C
95 17705A	97 17121G	99 17323D	101 14227H	103 12117E	105 13617A
107 14135G	109 14711G	111 15415C	113 13131E	115 13223A	117 16475C
119 14315C	121 16521E	123 13475A	133 11433B	135 10571A	137 15437G
139 12067F	141 13571A	143 12111A	145 16535C	147 17657D	149 12147F
151 14717F	153 13517B	155 14241C	157 14675G	163 10663F	165 10621A

Fig. 1 Irreducible polynomial

CMOS addition and multiplication

Addition:

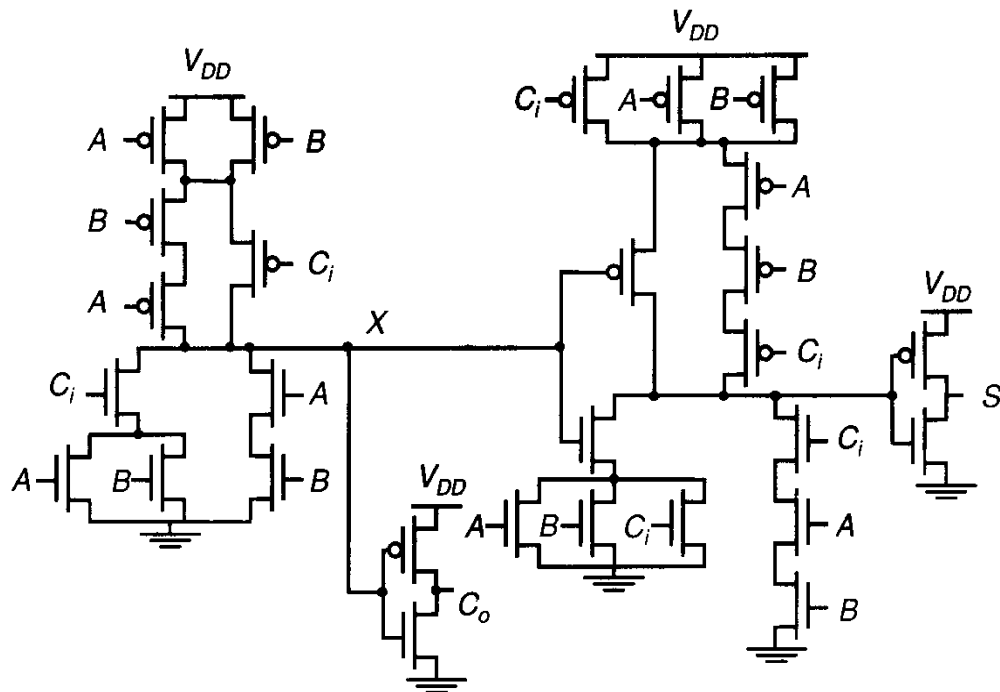


Fig. 2 CMOS adder

CMOS addition includes carrier generator and sum generator. The carrier generator that is not necessary. The full addition reduces to half addition. This structure only needed 28 transistors.

Multiplication

A multiplication includes some full ADDs. and half ADDs. It is needed much times of time than single ADDs. to obtain the output state.

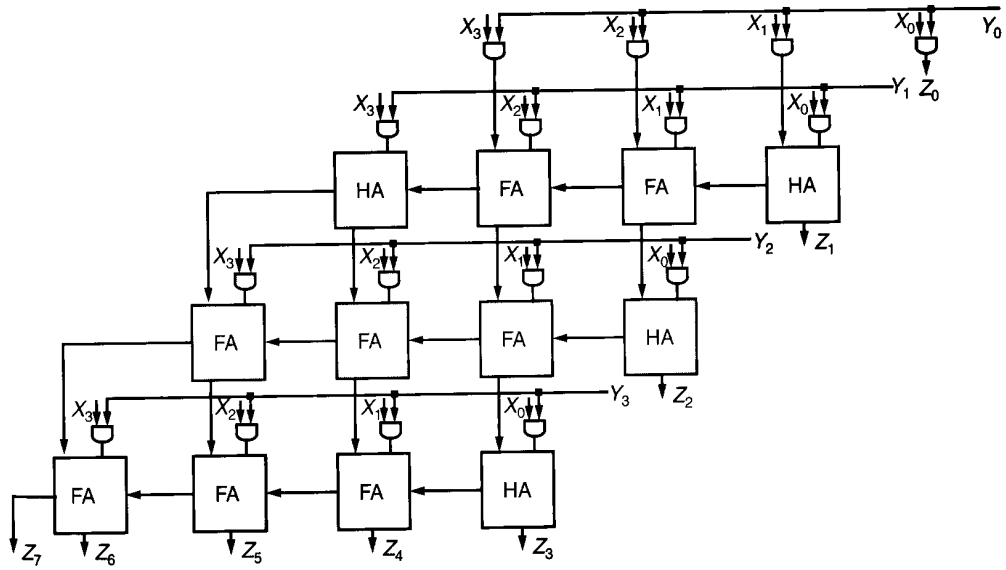


Fig. 3 Multiplier

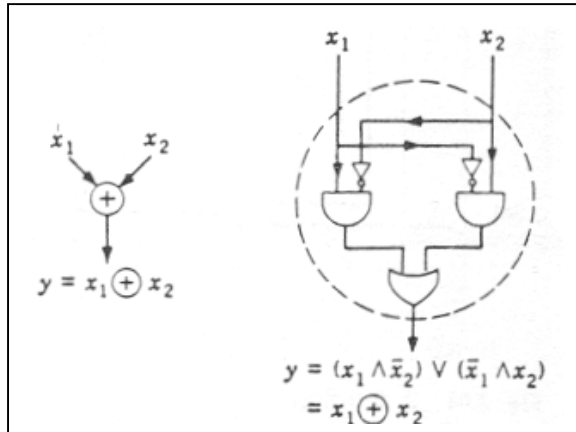
A comparing of propagation time

$$t_{\text{add}} \cong t_{\text{AB to Co}} + t_{\text{Ci to Co}} + t_{\text{Ci to s}} \cong t_{\text{Ci to s}} \cong 1 \text{ n sec}$$

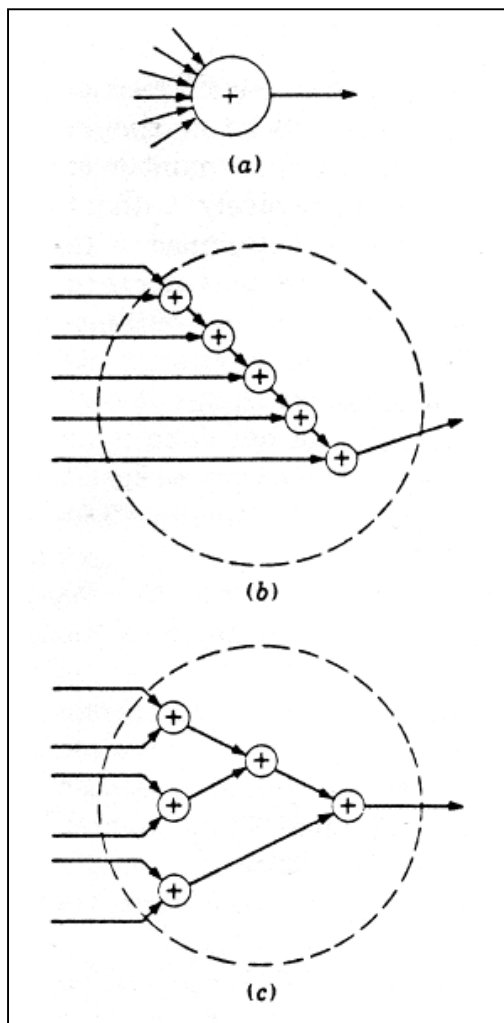
$$t_{\text{mult}} \cong [(M-1) + (N-2)]t_{\text{carry}} + (N-1)t_{\text{sum}} + t_{\text{and}}$$

ADD :

Two input binary adder

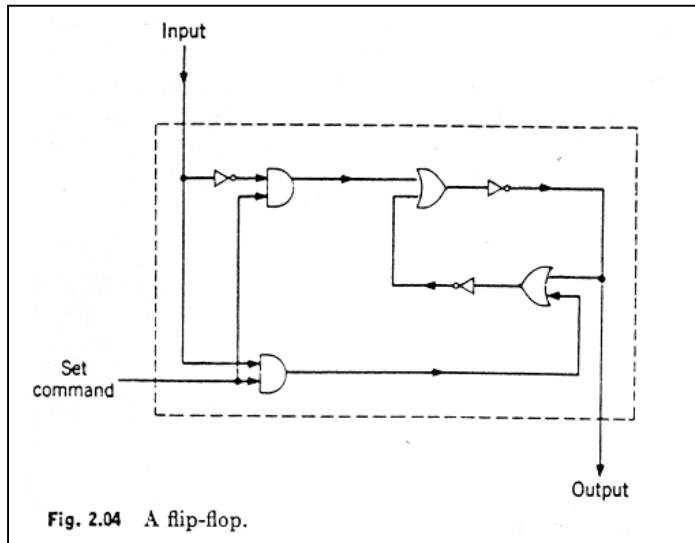


Multi-input binary adder



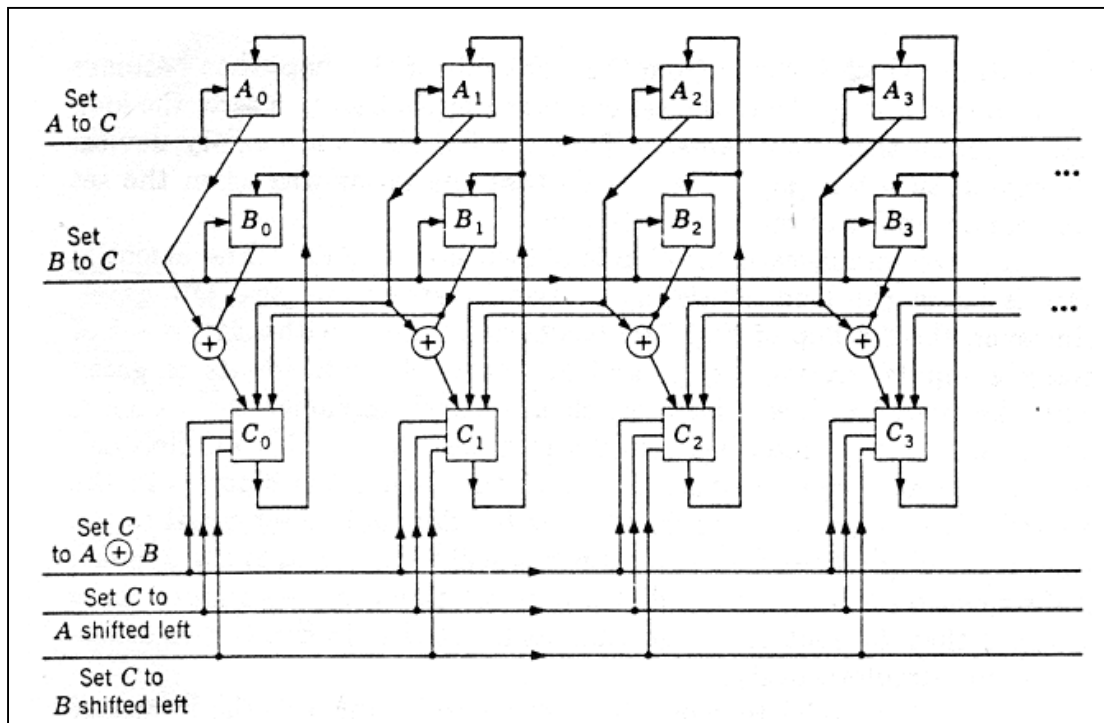
(c)的效能會比(b)好，因為(c)的 critical path 較短

A flip-flop



當 Set command 為 1 時，input 資料才可進入；當 Set command 為 0 時，輸出資料維持之前所抓取的。

Shifting a register or adding



Example: Two element over Galois field

Assume

$$\alpha = (1 \ 1 \ 0 \ 1), \beta = (1 \ 0 \ 1 \ 1), \text{ over } GF(2^4),$$

satisfied $\alpha^4 + \alpha + 1 = 0$

$$\text{than } \alpha + \beta = (0 \ 1 \ 1 \ 0)$$

where without carries

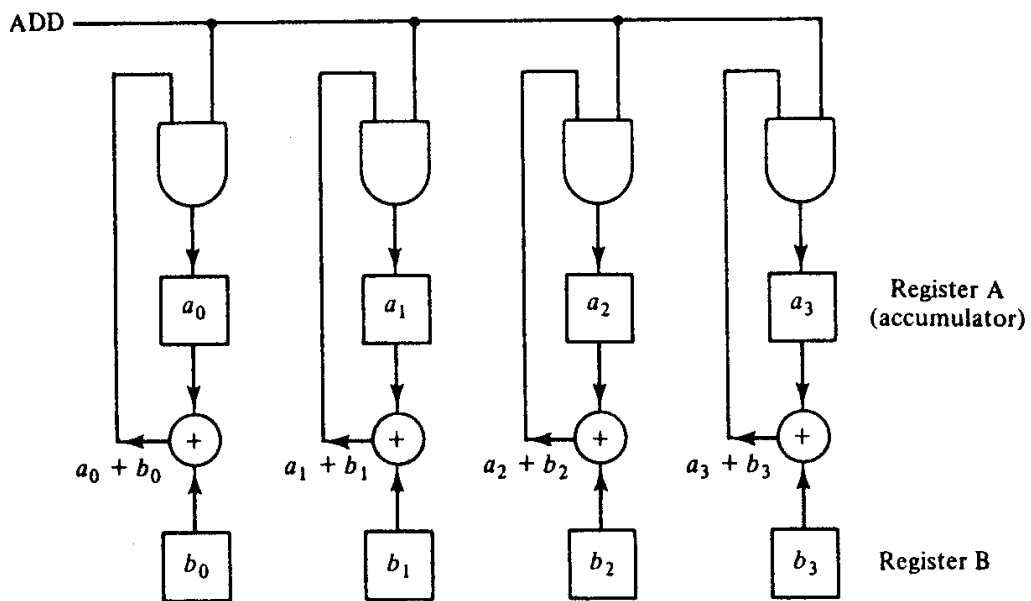


Fig. 4 Galois field adder

Multiplication:

Example: scale multiplication

Assume $\beta(\alpha) = b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3$ over $GF(2^4)$,

satisfied $\alpha^4 + \alpha + 1 = 0$

then $\alpha\beta = b_0\alpha + b_1\alpha^2 + b_2\alpha^3 + b_3\alpha^4$

$$= b_3 + (b_0 + b_3)\alpha + b_1\alpha^2 + b_2\alpha^3$$

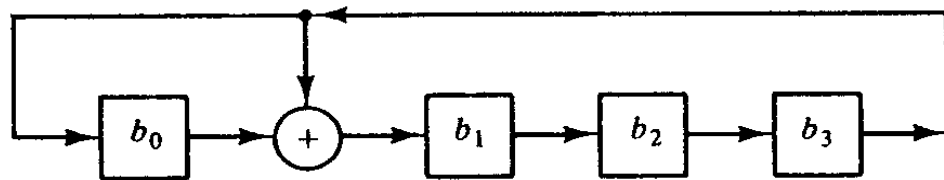


Fig. 5 Galois field multiplier with scale and element

where register plus once can obtain $\alpha\beta$, if we want to obtain $\alpha^3\beta$, such that need three times of plus, it is wasting times. They are another structure of multiplication which propagation times less than previous.

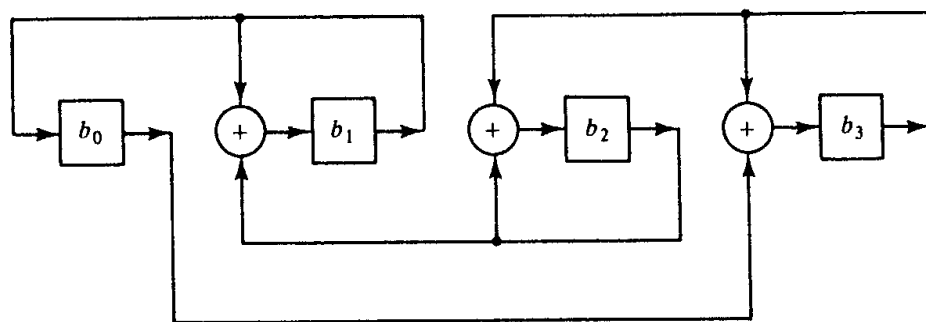


Fig.6 Another Galois field multiplier

where $\alpha^3 \beta = b_0 \alpha^3 + b_1 \alpha^4 + b_2 \alpha^5 + b_3 \alpha^6$

$$= b_1 + (b_1 + b_2) \alpha + (b_2 + b_3) \alpha^2 + (b_0 + b_3) \alpha^3$$

Example: multiplication two elements over Galois field

Assume β, γ over $GF(2^4)$, satisfied $\alpha^4 + \alpha + 1 = 0$

where $\beta(\alpha) = b_0 + b_1 \alpha + b_2 \alpha^2 + b_3 \alpha^3$

$$\gamma(\alpha) = c_0 + c_1 \alpha + c_2 \alpha^2 + c_3 \alpha^3$$

than $\beta \gamma = (((c_3 \beta) \alpha + c_2 \beta) \alpha + c_1 \beta) \alpha + c_0 \beta$

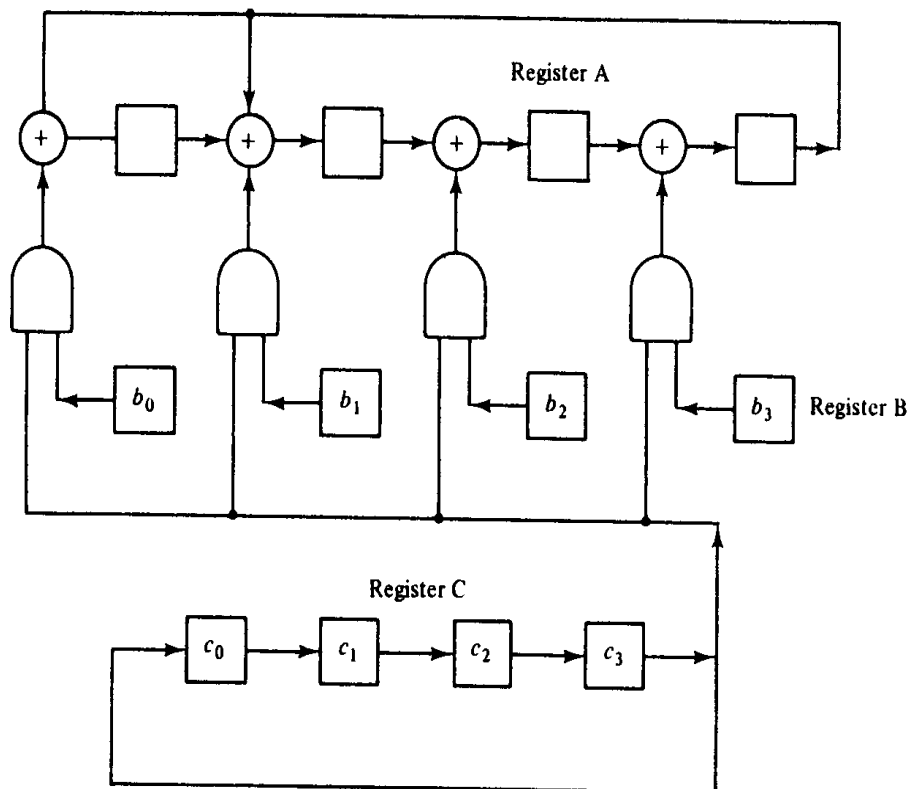


Fig. 7 Galois field multiplier with two elements

Example: computing $r(\alpha)$, over $GF(2^4)$

$$r(\alpha) = r_0 + r_1 \alpha + r_2 \alpha^2 + \dots + r_{14} \alpha^{14}$$

$$= (\dots(((r_{14} \alpha + r_{13}) \alpha + r_{12}) \alpha + \dots) \alpha + r_0$$

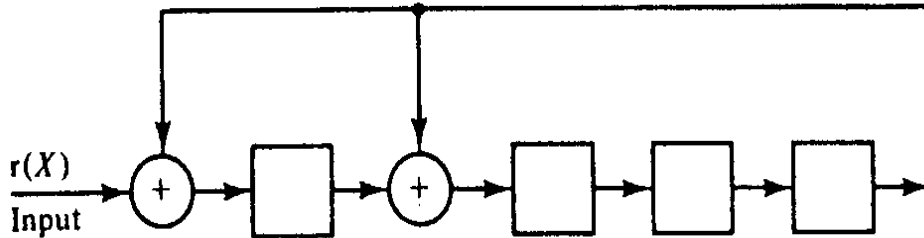


Fig.8 Computing $r(\alpha)$ circuit

Another circuit

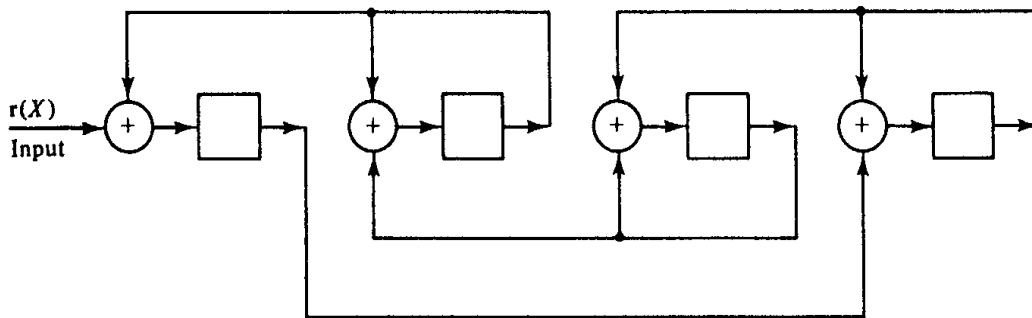


Fig.9 Another circuit ($r(\alpha^3)$)

Example: computing by remainder polynomial

assume dividing $r(x)$ by $\alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1 = 0$

then remainder $b(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3$

so that $r(\alpha^3) = b(\alpha^3)$

$$= b_0 + b_3 \alpha + b_2 \alpha^2 + (b_1 + b_2 + b_3) \alpha^3$$

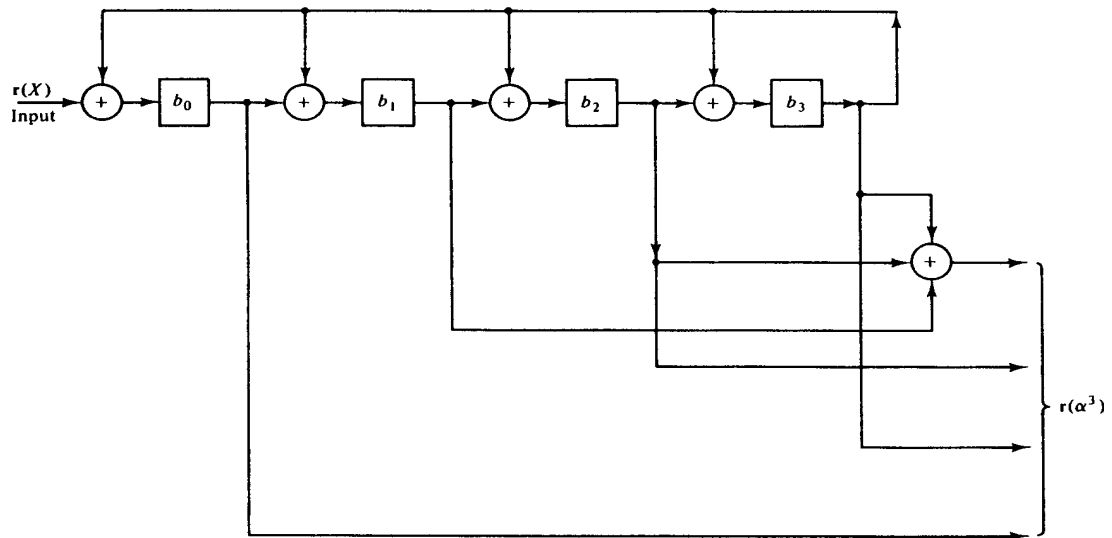


Fig. 10 Computing $r(\alpha^3)$ using remainder polynomial

Divider:

$$\frac{r(x)}{\phi_i(x)} = \frac{r_0 + r_1 \cdot x + r_2 \cdot x^2 + \dots + r_{10} \cdot x^{10} + r_{11} \cdot x^{11} + r_{12} \cdot x^{12} + r_{13} \cdot x^{13} + r_{14} \cdot x^{14}}{\phi_i(x)}$$

令

$$\phi_i(x) = 1 + x + x^2 + x^3 + x^4$$

$$\frac{r(x)}{\phi_i(x)} = \frac{r_0 + r_1 \cdot x + r_2 \cdot x^2 + \dots + r_8 \cdot x^8}{1 + x + x^2 + x^3 + x^4} + x^9 \cdot \frac{(r_9 + r_{10}) + (r_{11} + r_{14}) \cdot x + \dots + r_{14} \cdot x^4}{1 + x + x^2 + x^3 + x^4} + r_{14} \cdot x$$

$$(r_{10} + r_{11}x + \dots + r_{14}x^4) / (1 + x + x^2 + x^3 + x^4)$$

$$\text{mod} = ((r_{10} + 1) + (r_{11} + 1)x + (r_{12} + 1)x^2 + (r_{13} + 1)x^3$$

$$\text{when } r_{14} = 1$$

$$x^4 = x^3 + x^2 + x + 1$$

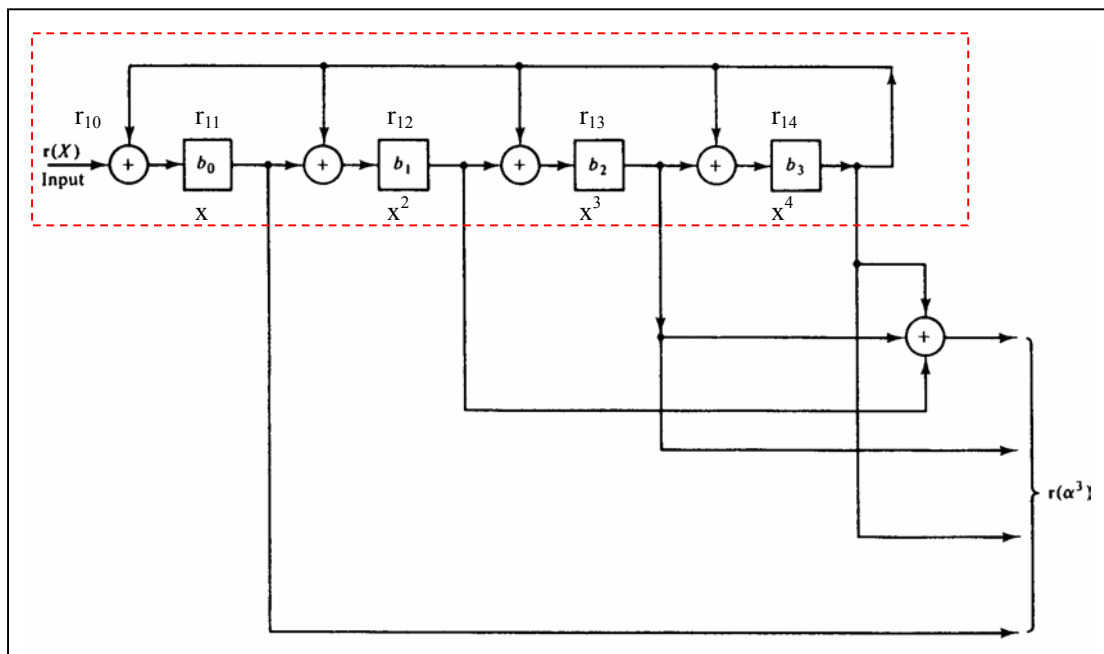
$$= \frac{r_0 + r_1 \cdot x + r_2 \cdot x^2 + \dots + r_8 \cdot x^8}{1 + x + x^2 + x^3 + x^4} + x^9 \cdot \left(\frac{(r_9 + r_{14}) + (r_{11} + r_{14}) \cdot x + (r_{12} + r_{14}) \cdot x^2 + (r_{13} + r_{14}) \cdot x^3}{1 + x + x^2 + x^3 + x^4} + r_{14} \right)$$

$$= \frac{r_0 + r_1 \cdot x + r_2 \cdot x^2 + \dots + r_8 \cdot x^8}{1 + x + x^2 + x^3 + x^4} + x^9 \cdot \left(\frac{r_9 + (r_{10} + r_{14}) \cdot x + (r_{11} + r_{14}) \cdot x^2 + (r_{12} + r_{14}) \cdot x^3 + (r_{13} + r_{14}) \cdot x^4}{1 + x + x^2 + x^3 + x^4} + r_{14} \cdot x \right)$$

$$= \frac{r_0 + r_1 \cdot x + r_2 \cdot x^2 + \dots + r_8 \cdot x^8}{1 + x + x^2 + x^3 + x^4}$$

$$+ x^9 \cdot \left(\frac{(r_9 + (r_{13} + r_{14})) + (r_{10} + r_{14} + (r_{13} + r_{14})) \cdot x + (r_{11} + r_{14} + (r_{13} + r_{14})) \cdot x^2 + (r_{12} + r_{14} + (r_{13} + r_{14})) \cdot x^3}{1 + x + x^2 + x^3 + x^4} + (r_{13} + r_{14}) + r_{14} \cdot x \right)$$

=



BCH :

Encode:

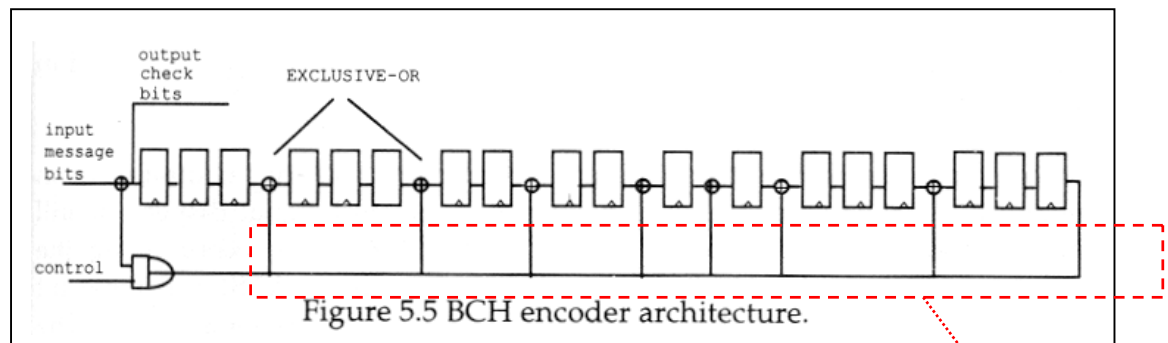
$$n=2^m-1$$

$$n-k \leq mt$$

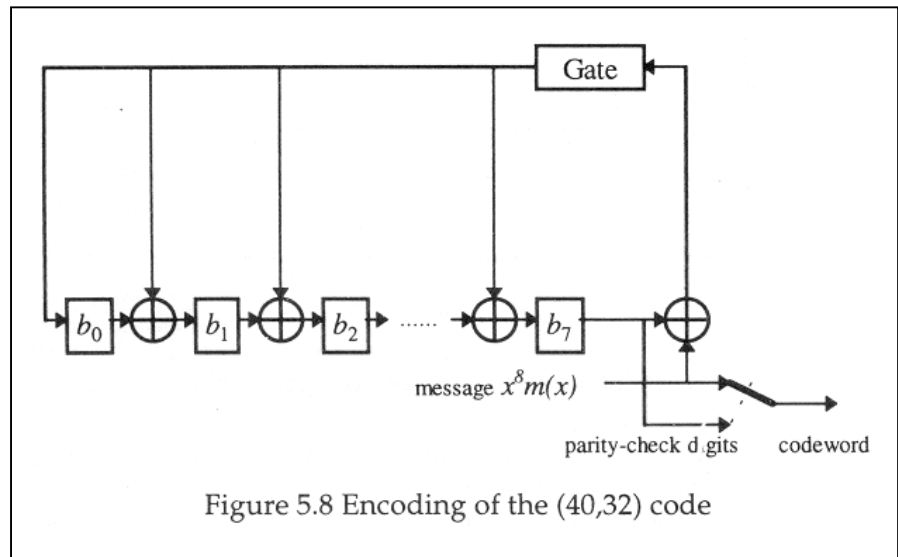
$$d_{\min} \geq 2t+1$$

$$g(x) = \text{LCM}\{\Phi_1(x), \dots, \Phi_{2^t-1}(x)\}$$

encoder



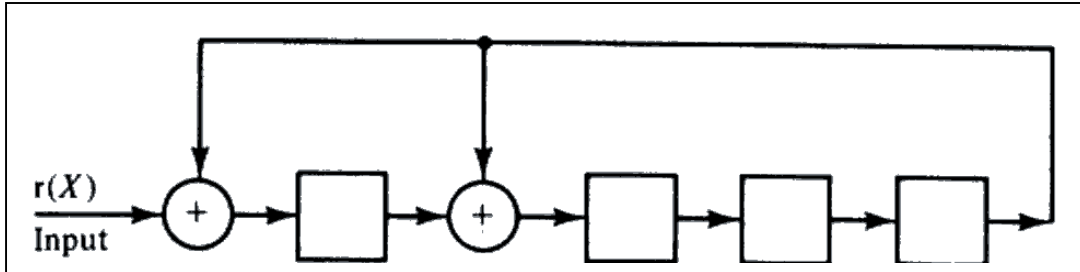
由 $g(x)$ 決定



Decode:

在做 BCH 解碼時，我們必須先算出徵狀值 (syndrome)， $S_i = r(\alpha^i)$

$$\begin{aligned}
 r(\alpha) &= r_0 + r_1 \cdot \alpha + r_2 \cdot \alpha^2 + \cdots + r_{14} \cdot \alpha^{14} \\
 &= (\cdots(((r_{14} \cdot \alpha + r_{13}) \cdot \alpha + r_{12}) \cdot \alpha + \cdots) \cdot \alpha + r_0
 \end{aligned}$$



假設我們的碼是建構在 $GF(2^4)$ 中，當計算 $2t$ 個徵狀值時，就必須有 $2t$ 個個別電路去計算，所佔面積過大，因此發展出另一套方法：

由於 BCH 的 generator polynomial 為

$$g(x) = LCM(\phi_1(x) \cdots \phi_{2t-1}(x))$$

code sequence 的產生：

$$c(x) = q(x) \cdot g(x)$$

接收到的 sequence:

$$r(x) = q(x) \cdot g(x) + b(x)$$

因此徵狀值可以化簡成

$$r(\alpha^i) = b(\alpha^i)$$

這部分由 $S_i=r(\alpha^i)$ 決定其電路

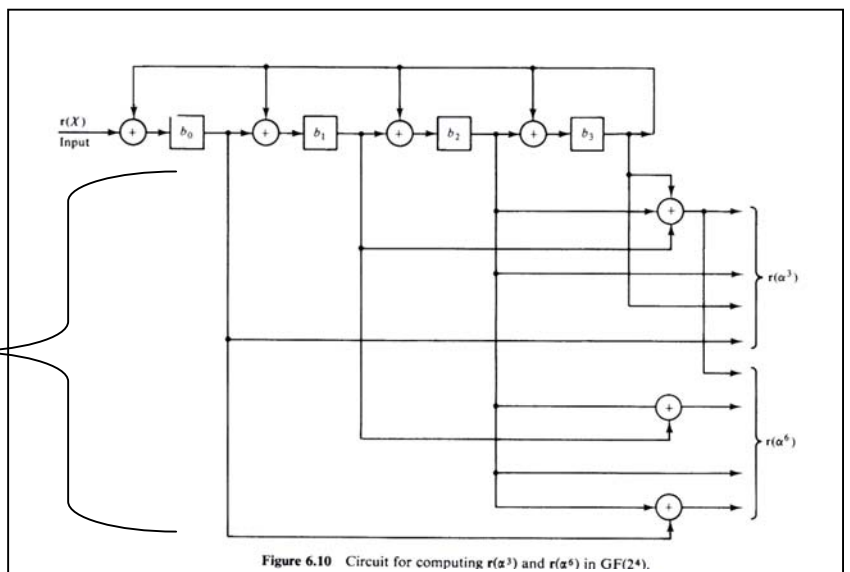


Figure 6.10 Circuit for computing $r(\alpha^3)$ and $r(\alpha^6)$ in $GF(2^4)$.

Computation of error-location numbers and error correction

假設 error location polynomial 為：

$$\sigma(x) = 1 + \sigma_1 \cdot x + \sigma_2 \cdot x^2 + \cdots + \sigma_v \cdot x^v$$

其中 σ_i 可由 S_i 決定

Register B

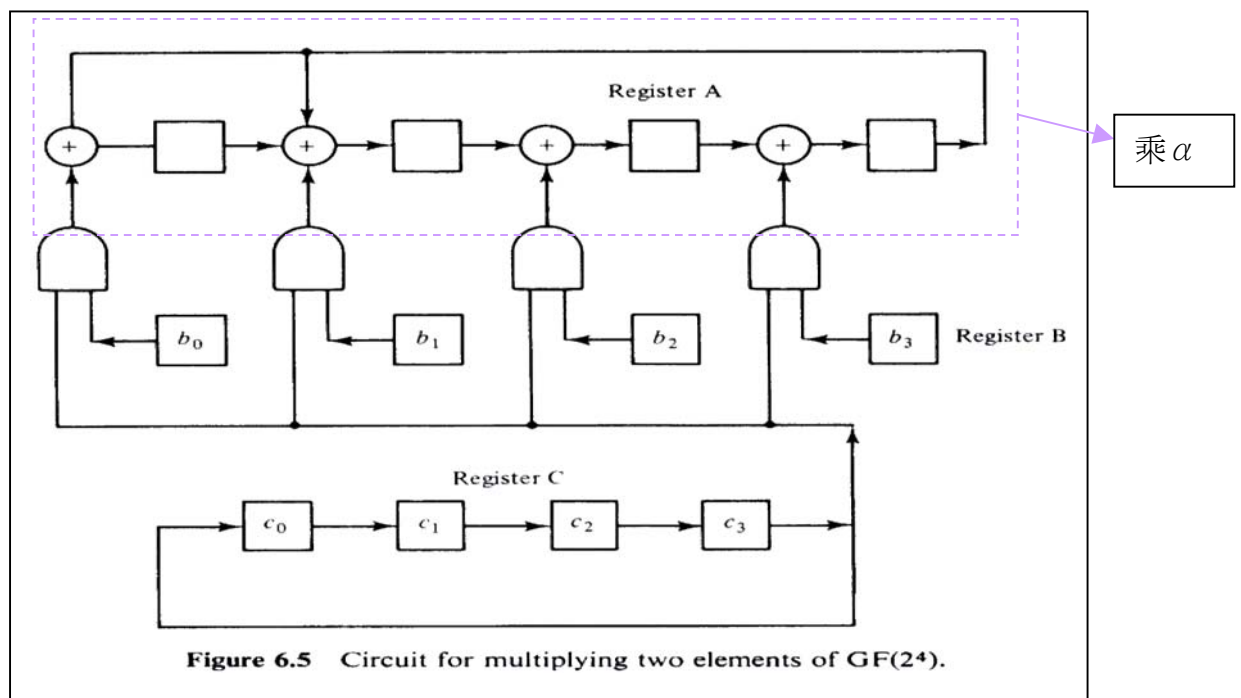
將 $\sigma(x)$ 的係數載入暫存器

Register C

$$\alpha^i = c_3 \cdot \alpha^3 + c_2 \cdot \alpha^2 + c_1 \cdot \alpha + c_0$$

$$\begin{aligned} \sigma_i &= b_0 + b_1 \alpha + b_2 \alpha^2 + b_3 \alpha^3 \\ \alpha^i &= c_0 + c_1 \alpha + c_2 \alpha^2 + c_3 \alpha^3 \\ \sigma_i * \alpha^i &= (((c_3 \beta) \alpha + c_2 \beta) \alpha + c_1 \beta) \alpha + c_0 \beta \end{aligned}$$

經過 m 個 clock， $\sigma_i * \alpha^i$ 存在於暫存器 A 中，如要計算 $\sigma_i * \alpha^{2i}$ ，將 $\sigma_i * \alpha^{2i}$ 存到暫存器 B 中，在經過 m 個 clock 後， $\sigma_i * \alpha^{2i}$ 存在於暫存器 A 中。依此類推，最後將各個階段暫存器 A 中的加起來，可以得到 $\sigma(\alpha^i)$ 。



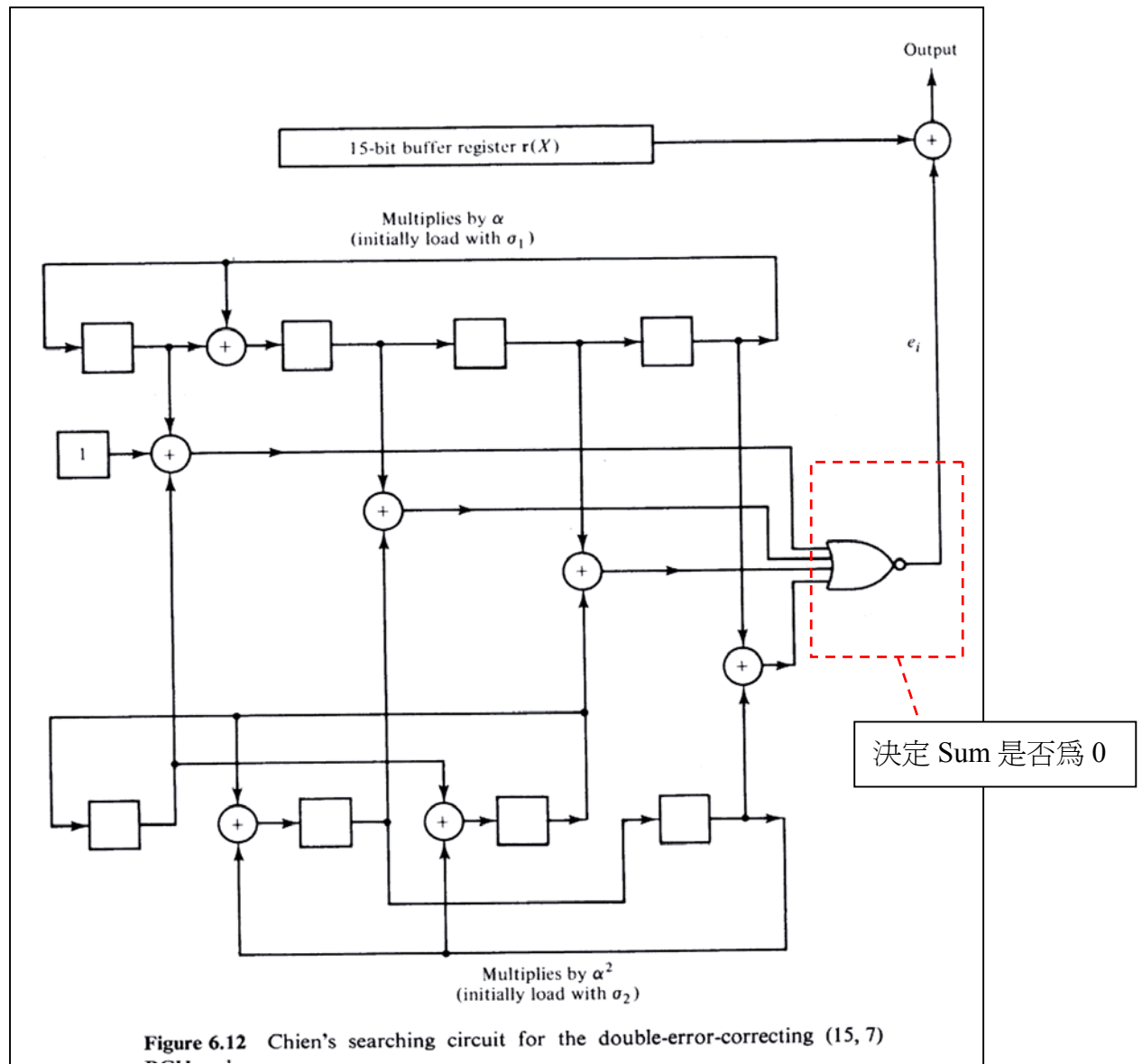
Chien's searching circuit

先將 $\sigma_1 \cdots \sigma_v$ 載入各個乘法器，然後做位移的動作，
則將各個暫存器的結果相加起來，如果為 0，則 α^i 為
error-location polynomial 的一根

$$\sigma(\alpha^i) = 1 + \sigma_1 \cdot \alpha^i + \sigma_2 \cdot (\alpha^i)^2 + \cdots + \sigma_v \cdot (\alpha^i)^v$$

下圖的例子，為 $t=2$ 的 BCH，所以需要兩個乘法器。

如果 t 很大時，將會佔很大的面積。



Reed-Solomon codes :

Block length:	$n=q-1$
Number of parity-check bits:	$n-k=2t$
Minimum distance:	$d_{\min}=2t+1$

Encoder:

generator polynomial of t-error-correcting

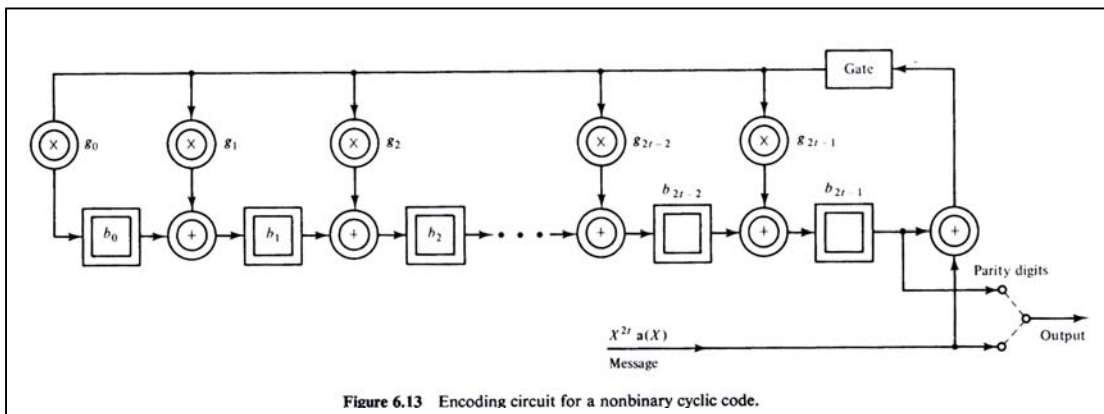
$$g(x) = (x + \alpha)(x + \alpha^2) \cdots (x + \alpha^{2t})$$

產生 systematic RS code 的方法 :

$$m(x) = m_0 + m_1x + m_2x^2 + \cdots + m_{k-1}x^{k-1}$$

$$x^{2t}m(x) = q(x)g(x) + b(x)$$

所以資料由接近輸出的地方輸入，代表乘上 x^{2t} 。當資料輸入完畢，Gate 自動關閉，留在暫存器的值，就是 parity-check codes。



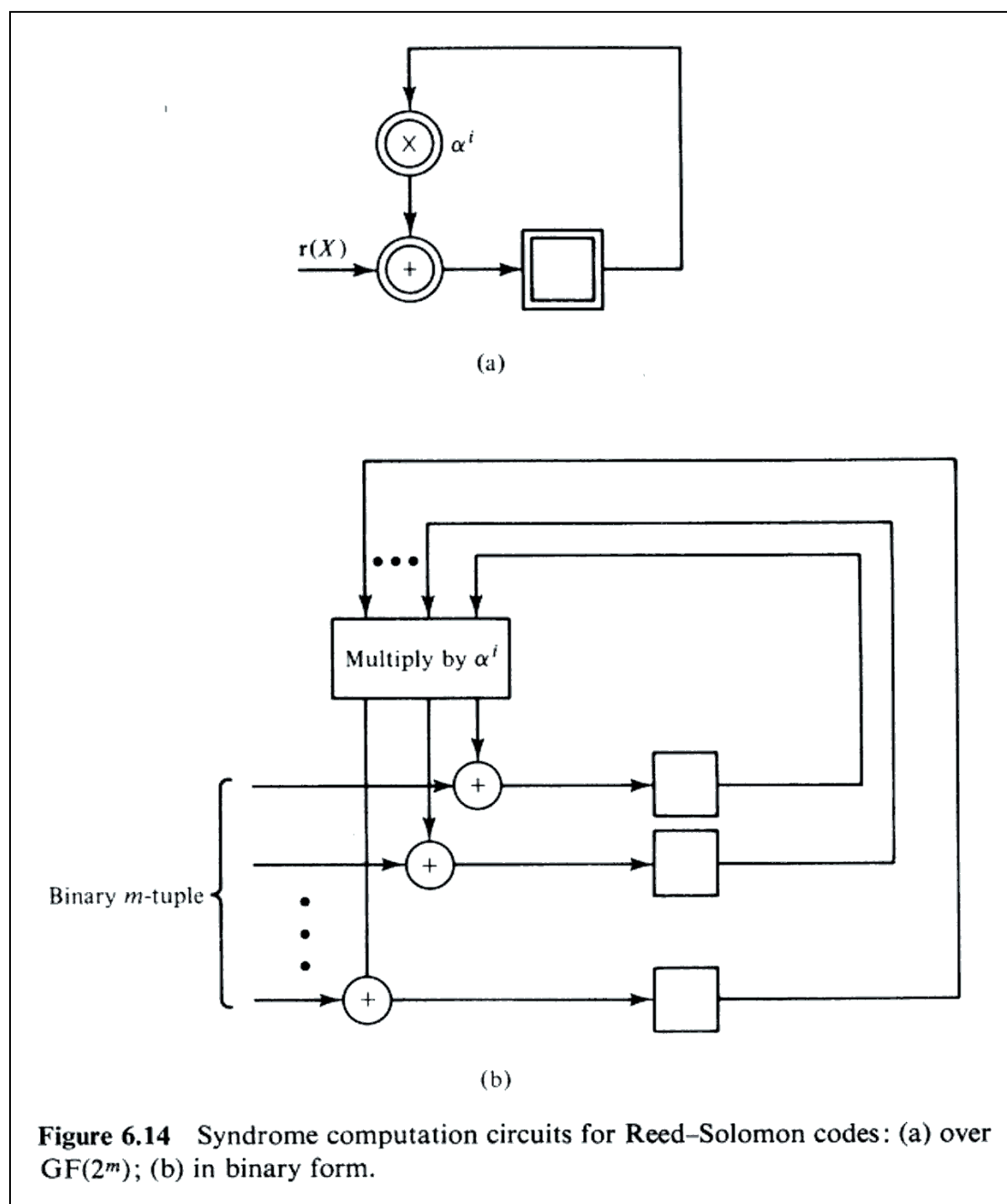
計算 Syndrome:

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

由於 $r(x)$ 可表示成下列式子，

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

所以先將 $r(x)$ 除以 $(x + \alpha^i)$ ，當 $r(x)$ 完全輸入時，最後留在暫存器的值，就是餘數，也就是 α^i 的 syndrome



結論：

在 Galois Field 中，不論是加法或者是乘法，甚至是除法，我們都可以使用簡單的加法和移位暫存器 (shift register) 來做運算。因此我們可大大地減低運算複雜度，且運算速度也有所提昇。對於這樣的結果，剛好符合我們在通訊上所要的東西，Low Power、Low Cost、High Performance 的要求。

Reference:

1. Error Control Coding

Shu Lin / Daniel J. Costello, Jr.

2. Digital integrated circuits

Rabaey, Jan M. 1995

3. Coherent spread spectrum systems

Holmes, Jack K. 1982

4. Algebraic coding theory

Elwyn R. Berlekamp 1984

5. Error-control coding for data networks

Irving S. Reed/Xuemin Chen 1999