航空技術學院學報 第九卷 第一期 第 97－104 頁（民國九十九年）
Journal of Air Force Institute of Technology, Vol. 9, No. 1, pp.97-104, 2010

基於 FPGA 之 Golay code 步階解碼法研究與實現

Research and implementation of step-by-step decoding algorithm for Golay code based on FPGA

柳復華[1]，池慶龍[2]，程威瑞[2]，池宗修[2]

[1] 空軍航空技術學院航空通訊電子系

[2] 樹德科技大學電腦通訊系

Fu-Hwa Liu[1], Ching-Lung Chi[2], Wei-Jui Cheng[2], Tsung-Hsiu Chi[2]

[1] Department of Aviation & Communication Electronics, Air Force Institute of Technology.

[2] Department of Computer & Communication, SHU-TE University.

摘要

本論文針對二位元（23, 12, 7）Golay Code 提出一種改良式的步階解碼演算法，經由邏輯分析我們推導出一種可以直接判斷接收碼字中每一位元正確性的簡易規則。這種方法降低了運算複雜度，也明顯的比傳統捕錯解碼法降低了解碼執行時間。

關鍵詞：二位元（23，12，7）Golay Code、步階解碼法、複雜性、error-trapping 解碼法。

Abstract

In this paper, a modified step-by-step algorithm for decoding the (23, 12, 7) Binary Golay code is presented. Logical analysis yielded a simple rule for directly determining whether a bit in the received word is correct. The computational complexity can be reduced using this scheme. The computation time for the decoder is reduced significantly, compared with the error-trapping decoding algorithm.

key words: (23, 12, 7) binary Golay code, step-by-step decoding, complexity, error-trapping decoding

1. Golay code 簡介

可以從[1]當中得知，Golay code 在二位元的 BCH code 裡是一較特殊例子。從其中理論 2 與 3 當中得知無論是理論亦或是實現，都是種完美的多重糾錯碼(multiple-error-correcting)。它的解碼方法早期只有 error-trapping 的系統搜尋解碼法[2]，這種解碼法適用於低碼率與碼長較短、更正錯誤能力較少的碼。而 Massey 在 1965 年所發表的論文[3]，提出了以傳統 BCH Code 的步階(Step By Step)的解碼方法來解 Golay code。步階解碼法為每改變一個位元就判斷他的錯誤位元有無增加或減少，利用此種方式將錯誤位元改正回來[4]。但是，對於解 Golay code 並不能將兩個錯與三個錯有

效的判斷並改正回來。於是在 1987 年 Elia 所發表的文章[5]解決了這個問題。而在[6]中提到判斷 Golay code 錯誤多項式在運算上的除法與平方根上所需的複雜度相對較多，對於硬體上的實現也較困難，所以本文利用[7]中的錯誤判斷多項式，來減少在除法與平方根的運算，再利用[8]之低複雜度步階解碼法與傳統之 error-trapping 的方法將（23, 12, 7）Golay Code 分別利用兩種型式以 Altera Quartus II 發展軟體實現並比較其解碼所需花的時脈。

2.文獻探討

（23，12，7）Golay Code 是二位元 BCH code 的一種[1]，其總碼長為 23 個，可傳送訊息的長度為 12。它的造碼多項式 $g(x)$ 表示為：

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1 \quad (1)$$

其 $g(x)$ 為 $x^{23} - 1$ 分解出的一個最簡多項式。（23, 12, 7）Golay Code 的傳輸訊息為 12 個，於是將收到訊息的多項式表示為：

$$i(x) = c_0 + c_1 x + \ldots + c_{11} x^{11} \quad (2)$$

由於還須加上核位元，於是我們將收到訊息與造碼多項式作相除後所得到餘數當作我們核位元，其核位元多項式如下：

$$p(x) = c_{12} x^{12} + \ldots + c_{22} x^{22} \quad (3)$$

於是整個 Code word 表示為：

$$c(x) = i(x) + p(x) = \sum_{i=0}^{22} c_i x^i \quad (4)$$

由於傳送時會受到雜訊干擾於是錯誤多項式與收到的 code word 多項式表示為：

$$e(x) = \sum_{i=0}^{22} e_i x^i \quad (5)$$

$$r(x) = c(x) + e(x) = \sum_{i=0}^{22} r_i x^i \quad (6)$$

由於收到干擾後的 codeword 必須將徵狀值求出，於是令 α 為這 23 個 code word 所產生之徵狀值的根，其中 $\alpha \in GF(2^{11})$。於是定義徵狀值為 $s_i = e(\alpha^i)$，其中 $i \neq 0$，$j \in \{1,2,3,4,6,8,9,12,13,16,18\}$ 時 $g(\alpha^j) = 0$。從[5]中我們可得知，只要利用 $\alpha, \alpha^3, \alpha^9$ 這三個值就可包含 j 裡
的所有值，並可算出算出三個徵狀值

$$S_1 = r(\alpha) \quad S_9 = r(\alpha^9) \quad S_3 = r(\alpha^3)$$

因此徵狀值多項式可表示為；

$$s_j = r(\alpha^j) = c(\alpha^j) + e(\alpha^j) = e(\alpha^j) \quad (7)$$

在基於[5]的理論下[6]提出了一個判定錯誤個數的多項式 F 表示為：

$$F\begin{cases} S_3 + S_1[(T_3)^2 + T_9/T_3]^{1/3} & T_3 \neq 0 \\ 0 & T_3 = 0 \end{cases} \quad (8)$$

在這個多項式當中 T_3 代表著 $S_1^3 + S_3$，T_9 表示為 $S_1^9 + S_9$。當中在 $T_3 \neq 0$ 會影響到錯誤數量在 2 與 3 時的判定，而 $S_1 \neq 0$ 則影響錯誤數量在 1 與 2 時的判定。由這多項式可知，載運算除法與平方根的時候所需耗費的時間與複雜度相對較高，硬體實現亦是如此；因此我們基於 F 多項式將其修改為：

$$M = S_3^4 + S_1^9 S_3 + S_1^6 S_3^2 + S_1^3 S_9 \quad (9)$$

由此多項式可看出只需利用加法與乘法運算就可得出相同的結果。基於以上描述我們可

以利用 S_1、T_3 與 M 的值，條列出 4 種錯誤情況：

1)在沒有錯誤的情況下：

$S_1 = T_3(T_9) = M$

2)在 1 個錯誤的情況下：

$S_1 \neq 0$，$T_3(T_9) = 0$ $M = 0$

3)在 2 個錯誤的情況下：

$S_1 \neq 0$，$T_3(T_9) \neq 0$，$M = 0$

4)在 3 個錯誤的情況下：

$S_1 \neq 0$，$T_3(T_9) \neq 0$，$M \neq 0$

我們將定義一組決定錯誤向量為 $H = (h_1, h_2, h_3)$ 來了解錯誤數量的變化；

若 $S_1 \neq 0$ 則 $h_1 = 1$。
若 $T_3(T_9) \neq 0$ 則 $h_2 = 1$。
若 $M \neq 0$ 則 $h_3 = 1$。

所以我們可以得知在 $h_1 = h_2 = h_3 = 0$ 時完全沒有錯誤產生，向量的表示為(0,0,0)，一個錯時 $h_1 \neq 0$，$h_2 = h_3 = 0$，向量表示為(1,0,0)以此類推我們可以得出決定錯誤向量 H 在對於錯誤數量改變時的所有可能：

若 $h_1 = h_2 = h_3 = 0$ 則 $H = (0,0,0)$。
若 $h_1 = 1$ 則 $H = (1,0,0)$。
若 $h_2 = 1$ 則 $H = (1,1,0)$。
若 $h_3 = 1$ 則 $H = (1,1,1)$。

得出決定錯誤向量就可以來進行步階解碼法。步階解碼法是在收到的訊息 $r(x) = \sum_{i=0}^{22} r_i x^i$ 中，改變 $r(x)$ 一個位置的值定義為 x^p，則收到的 Code word 可表示成：

$$r_{\bar{p}}(x) = r(x) + x^p = c(x) + e(x) + x^p = c(x) + e_{\bar{p}}(x) \quad (10)$$

其中 $e_{\bar{p}}(x) = e(x) + x^p$。由於改變了其中一個值需要重新計算徵狀值，經過修改過後的徵狀值表示為：

$$S_{i,\bar{p}} = e_{\bar{p}}(\alpha^i) = e(\alpha^i) + (\alpha^i)^p = S_i + \alpha^{ip}, i = 1,3,9 \quad (11)$$

改變的值 x^p 有可能是錯的或是對的，相對的也會影響到決定錯誤向量 H，所以定義改變 x^p 值後的決定錯誤向量為：

$H_{\bar{p}} = (h_{1,\bar{p}}, h_{2,\bar{p}}, h_{3,\bar{p}})$

影響 $h_{1,\bar{p}}, h_{2,\bar{p}}, h_{3,\bar{p}}$ 這三個值的因素分別為；

若 $S_{1,\bar{p}} \neq 1$ 則 $h_{1,\bar{p}} = 1$。
若 $T_{3,\bar{p}}(T_{9,\bar{p}}) \neq 0$ 則 $h_{2,\bar{p}} = 1$。
若 $M_{\bar{p}} \neq 0$ 則 $h_{3,\bar{p}} = 1$。

其中有加上 $\bar{p}$ 指的是改變 x^p 後所重新計算與定義出的值。

接下來用一個簡單的例子來說明向量 H 與 $H_{\bar{p}}$ 之間對於改變 x^p 後錯誤數量關係；錯誤位元數量為一個時：$H = (1,0,0)$ 改變 x^p 是錯誤的位置時候，$H_{\bar{p}} = (0,0,0)$。反之，若 x^p 是正確的位置，$H_{\bar{p}} = (1,1,0)$。透過定義向量 H 與 $H_{\bar{p}}$ 可以看出改變 x^p 是否有將錯誤的值解回正確的值。

本篇是利用文獻[8]裡所提到的低複雜度步階解碼法來對（23,12,7）Golay Code 進行解碼的硬

體實現。其理論與證明下所示：

定理 1：

對於（23，12，7）Golay Code 錯誤模式 e 會因為改變了 x^p 而改變，其中 p 的範圍落在 $0 \le p \le 22$，所以改變 x^p 後的錯誤模式令為 $\hat{e}_p$ 並可利用決定錯誤向量 H 與 $H_{\bar{p}}$ 裡的 $h_{1,\bar{p}}$，h_2 與 $h_{3,\bar{p}}$ 的值進行解碼，其式子表示為：

$$\hat{e}_p = \overline{h}_{1,\bar{p}} \overline{h}_2 \vee h_2 \overline{h}_{3,\bar{p}} \quad (12)$$

其 p 的範圍為：$0 \le p \le 22$

證明：

由文獻[1]提到 Golay Code 的錯誤務模式不會超過 3 個以上，因此稱 Golay Code 為一個完美的 code。由文獻[8]可得知解碼能力在 3 的情況需要用到多項式(12)進行解碼。其證明如表(1)表示：

表(1) $\hat{e}_p$ 所需用的變異數

v	0	1		2		3	
e_p	0	1	0	1	0	1	0
h_2	0	0	0	1	1	1	1
$v_{\bar{p}}$	1	0	2	1	3	2	3
$h_{1,\bar{p}}$	1	0	1	1	×	1	×
$h_{3,\bar{p}}$	0	0	0	0	1	0	1
$\hat{e}_p = \overline{h}_{1,\bar{p}}\overline{h}_2 \vee h_2\overline{h}_{3,\bar{p}}$	0	1	0	1	0	1	0

在此說明表(1)的變數所代表的意義如下；v 代表的是錯誤的數量，由於（23, 12, 7）Golay Code 解碼能力為 3，表列出 0~3 個錯誤。由於改變訊息其中一個值時我們無法得知你所改變的位置對或錯，因此 e_p 所代表的是每改變訊息其中一個位置時在錯誤量 0~3 時所產生的每種可能。$v_{\bar{p}}$ 則是隨著 e_p 做改變，假設訊息的第 x^p 的值是錯的，經過你的轉換後變成對的，那麼錯誤量就會減少，反之，$v_{\bar{p}}$ 的值就增加。而 h_2、$h_{1,\bar{p}}$ 與 $h_{3,\bar{p}}$ 為決定錯誤向量 H 與 $H_{\bar{p}}$ 所顯示的值。表(2)為傳統步階與低複雜度步階解碼法所需用到的變異數與所需用到的向量值之比較。

表(2) 傳統與低複雜度步階運算函數比較

	傳統步階解碼法	**低複雜度步階解碼法**
$\hat{e}_p$	$[(h_1\overline{h}_{1,\bar{p}})\overline{h}_2\overline{h}_{2,\bar{p}} \vee h_1\overline{h}_{1,\bar{p}}h_2\overline{h}_{2,\bar{p}}]\overline{h}_3\overline{h}_{3,\bar{p}} \vee h_{1,\bar{p}}h_2h_{2,\bar{p}}h_3\overline{h}_{3,\bar{p}}$	$\overline{h}_{1,\bar{p}}\overline{h}_2 \vee h_2\overline{h}_{3,\bar{p}}$
乘法數量	12	2
計算變異數	S_1, T_3, F $S_{1,\bar{p}}, T_{3,\bar{p}}, F_p$	T_3 $S_{1,\bar{p}}, M_{\bar{p}}$

可以看得出來低複雜度步階不管是變異數的計算與所需用到的向量值，都比傳統步階少了許多。將整個低複雜度步階解碼法對於（23,12,7）Golay Code 的過程條列出來：

1) 算出徵狀值 S_i ($i=1,3,9$) 與 $T_3(T_9)$ 、 M 值，定義決定錯誤向量 H。

2) p 的範圍為 $0 \le p \le 22$。

3) 利用方程式(2)算出修改 x^p 後的徵狀值 $S_{i,\bar{p}}$ ($i=1,3,9$)，接著計算 $T_{3,\bar{p}}(T_{9,\bar{p}})$ 、 $M_{\bar{p}}$ 值並定義修改後的決定錯誤向量 $H_{\bar{p}}$，利用方程式(12)將改變 x^p 位置後的 $\hat{e}_p$ 算出。

4) 最後輸出的值為 $\hat{r}_p = r_p + \hat{e}_p$。

5) 真正需進行解碼的為 code word 的前 12 個位元，因此解碼動作在 $p=11$ 之前都一直重複步驟 4，重複到 $p=11$ 就解碼完成。

error-trapping 解碼方式是通過徵狀值與 $r(x)$ 的循環移位運算，把錯誤移至核位元再進行更正錯誤。設碼字 $c(x)$ 是某一更正 t 個錯誤的 $[n, k]$ 循環碼的碼字，當它通過有干擾通道到達接收端解碼器時成為 $r(x)=c(x)+e(x)$。相應的徵候為：

$$s(x) \equiv r(x) \equiv e(x) \equiv e_I(x) + e_P(x) \qquad (13)$$

其中

$$e_I(x) = e_{n-1}x^{n-1} + \ldots + e_{n-k}x^{n-k} \qquad (14)$$

$$e_P(x) = e_{n-k-1}x^{n-k-1} + \ldots + e_0 \qquad (15)$$

分別是在碼字訊息組（或前 k 位）和校驗元（或後 n-k 位）上的錯誤圖樣。$\partial^\circ e(x) = n-k-1$，即所有≦t 個錯誤集中在校驗元的 n-k 位上，則 $e_I(x)=0$，$e(x)=e_p$。

e_p 的最高次數是 n-k-1，$\partial^\circ g(x) = n-k$，所

以當 $e(x)=e_p$ 被 $g(x)$ 進行除法動作後的餘式仍為 $e_p(x)$，即 $s(x)\equiv e(x)=e_P(x) \pmod{g(x)}$ 說明錯誤圖樣就是 $s(x)$。這樣，只要把 $r(x)$ 直接減去 $s(x)$ 就進行了更正錯誤，得到了 $\hat{c}(x)$。

只要錯誤集中地出現在任意的連續 n-k 位碼段以內，根據循環碼的特點，把 $R(x)$ 和相應的徵候 $S_0(x)$，一起同時在解碼器的各自移位暫存器中移位 i 次，使這些錯誤全部集中到 $x^iR(x)$ 的後 n-k 位上。

一但檢測電路檢測到徵候 $S_i(x)$ 的 $\omega(S_i(x))\le t$ 時，就認為此時的 $S_i(x)\equiv x^iS_0(x) \pmod{g(x)}$，就是 $x^iR(x) \pmod{x^n-1}$ 的徵候，並且錯誤已全部集中到 $x^iR(x)=R_i(x) \pmod{x^n-1}$ 的後 n-k 位以內。此時，只要在 $R_i(x)$ 中減去 $S_i(x)$，就對 $R_i(x)$ 進行了更正錯誤。然後，再把已更正錯誤過的 $\hat{R}_i(x)$，再繼續循環移位 x^{n-i} 次，即為 $x^{n-i}x^i\hat{R}(x)=x^n\hat{R}(x)=\hat{R}(x) \pmod{x^n-1}$，便恢復到原來接收的 $R(x)$ 中各碼元的順序關係，從而更正了 $R(x)$ 中的錯誤。由於這種解碼方式是通過 $S_0(x)$ 與 $R(x)$ 的循環移位運算，把錯誤捕獲到 n-k 位低次位元碼段（從 x^{n-k-1} 至 x^0）內，然後再進行更正錯誤。

顯然，只有當所有 $\le t$ 個錯誤全部集中在連續 n-k 位以內時，才是這種解碼方式可以更正的錯誤圖樣。

對於更正 t 個錯誤的循環碼來說，必須使 t 個錯誤能連續的出現在 n-k 位以內，這等價於要求有連續 k 位碼元無錯，或錯誤圖樣中連續 k 位的值為 0。由於 t 個錯誤均勻分布在 n 位上最難滿足連續 k 位無錯這一要求，因此可以用補錯方法解碼的[n,k]循環碼，n、k、t 之間必須滿足下列條件；$k<n/t$ 或 $t<n/k$ 或 $R<1/k$　n 為 code word 總長，k 為訊息的長度，t 為解碼能力。

表(3)為步階與 error-trapping 解碼時所需用到的時間比較表所需用到的時間

表(3)步解與 error-trapping 解碼時間比較

解碼方法	耗費時間
error-trapping	7.55ns
步階	2.3ns

3.解碼電路設計

利用低複雜度步階解碼法的原理，設計解碼電路的流程將條列如下：

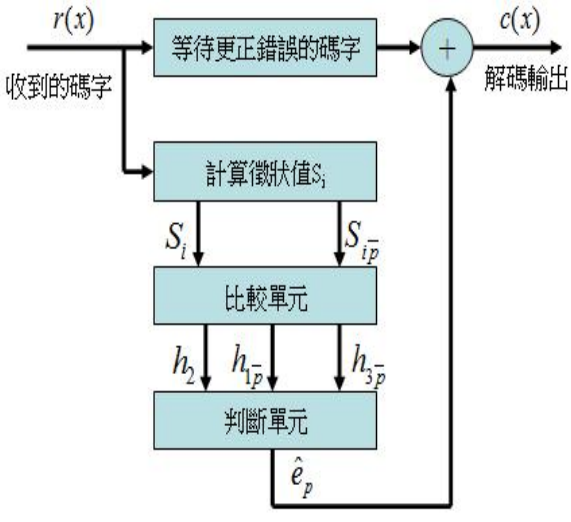
1)將受過雜訊干擾的訊息 $r(x)$ 傳送到一暫存電路裡。

2)接著也將訊息送到計算徵狀值電路，將 S_i 與轉換過 x^p 位置之後的 $S_{i,\bar{p}}$ 算出，其中 $i=1,3,9$ 由於 code word 前 12 個位元為真正要傳得訊息，所以 p 的範圍在 $11\le p\le 22$。

3)接著送到比較電路將決定錯誤向量 H、$H_{\bar{p}}$ 裡的 $h_{1,\bar{p}}$、h_2 與 $h_{3,\bar{p}}$ 值求出。

4)接著利用比較電路求出的 $h_{1,\bar{p}}$、h_2 與 $h_{3,\bar{p}}$ 送到解碼判斷方程式(2)電路將 $\hat{e}_p$ 算出，並與暫存電路裡的 $r(x)$ 做 XOR 後解回正解 code word。

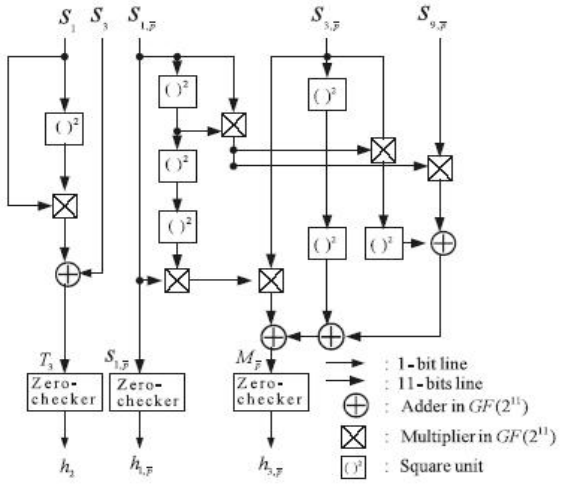
設計流程如圖(1)所表示：



圖(1) 設計步階解碼電路流程圖

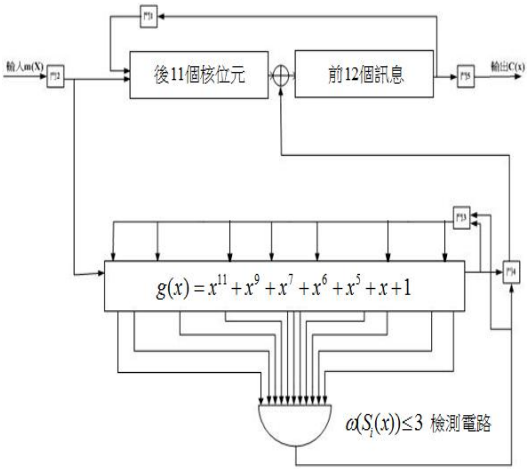
將算出的徵狀值利用圖(2)電路將決定錯誤向量的 h_2、$h_{1,\bar{p}}$ 與 $h_{3,\bar{p}}$ 值求出。圖(2)需用到的加法器與乘法器設計都必須符合 $GF(2^{11})$ 這個條件。因此，我們不需用到文獻[6]當中

的多項式 $F = S_3 + S_1[(T_3)^2 + T_9/T_3]^{1/3}$ 也可利用 T_3 、$S_{1,\bar{p}}$ 與 $M_{\bar{p}}$ 算出向量 h_2 、$h_{1,\bar{p}}$ 與 $h_{3,\bar{p}}$ 利用式子(12)將 $\hat{e}_p$ 值求出，達到解碼的目的。實際電路圖如圖(3)所示。



圖(2) 比較單元電路設計圖

圖(4)為 error-trapping 解碼器設計電路：



圖(4) error-trapping 解碼器電路圖

基本功過原理如下條列出：

1) 開始工作時，所有暫存器和緩存器清除 0，n=23 次位移後，$R(x)$ 的 23 個碼元全部移入（11+12）＝23 級緩存器，訊息元在前 12 級，同時徵候計算電路也完成了徵候計算得到了 $S_0(x)$ 。若 $S_0(x) = 0$，說明無錯，輸出緩存器中的 12 個訊息元。若 $S_0(x) \neq 0$，則進行以下各步驟。

2) 若檢測電路 $\omega(S_0(x)) \le 3$，此時存在徵候暫存器中的內容就是 $R(x)$ 後 11 位內的錯誤圖樣，移位 23 次，在移位過程中，徵候中的錯誤圖樣 $S_0(x) = E_p(x)$ 與 $R(x)$ 中的錯誤圖樣逐位模 3 加，完成了更正錯誤。再移位 23 次，輸出已更正的 $\hat{C}(x)$ 。

3) 若 $\omega(S_0(x)) > 3$，則 23 級緩存器和 $g(x)$ 除法電路都循環移位一次，並檢查 $\omega(S_1(x))$ ，若仍大於 3，則繼續循環移位。如果第 i 次循環移位後，檢測電路檢測到 $\omega(S_i(x)) \le 3$，則說明錯誤已捕獲到此時 $R_i(x) = x^i R(x) \pmod{x^n - 1}$ 的最後 11 位中，這時繼續移位 n-i=23-i 次，徵候暫存器中輸出的錯誤圖樣與緩存器中 $R_i(x)$ 的後 11 位中的碼元逐位模 3 加，以更正其中的錯誤。23-i 次移位完畢後更正錯誤結束。解碼器再移 15 次，就把緩存器中的結果 $\hat{C}(x)$ 全部送出。若始終沒有 $\omega(S_i(x)) \le 3$ ，則說明 $R(x)$ 中有不可更正的錯誤。實際電路如圖(5)所示。

4.範例

在這裡利用 ASCII 裡的"A"裡當作我們收到的訊息其"A"在 ASCII 裡的 10 進位表示為 65 將它化為 2 位元型式為：$c(x) = (000001000001)$ 我們將錯誤位置設 $e(x) = (000000101001)$，最後收到訊息為：$r(x) = c(x) + e(x) = (000001101000)$ 對照 ASCII 裡可得知原訊息受到干擾後變為"h"，藉由步階解碼法理論，求出徵狀值如下：

$S_1 = \alpha^{1893}$ ， $S_3 = \alpha^{263}$ ， $S_9 = \alpha^{1539}$ ，因此 $T_3 = \alpha^{1917}$ ，而 $M = 1$ ，$\therefore H = (1,1,1)$

由於以判定出 h_2 值，利用方程式(2)可得出 $\hat{e}_p$ 值如下：

$$\begin{aligned}\hat{e}_p &= \overline{h}_{1,\bar{p}}\overline{h}_2 \vee h_2\overline{h}_{3,\bar{p}} \\ &= \overline{h}_{1,\bar{p}} \cdot 0 \vee 1 \cdot \overline{h}_{3,\bar{p}} \\ &= \overline{h}_{3,\bar{p}}\end{aligned}$$

由於需改正的位元為前 12 個，因此 p 的範圍為$11 \le p \le 22$。改變過後的徵狀值如下所示：

$$S_{i,\bar{p}} = S_i + \alpha^{ip}, i = 1,3,9 \qquad (16)$$

則求出的$M_{\bar{p}}$值如下：

$h_{3,1\bar{1}} = 0,\ h_{3,1\bar{2}} = 1,\ h_{3,1\bar{3}} = 1,\ h_{3,1\bar{4}} = 0,\ h_{3,1\bar{5}} = 1,$

$h_{3,1\bar{6}} = 0,\ h_{3,1\bar{7}} = 1,\ h_{3,1\bar{8}} = 1,\ h_{3,1\bar{9}} = 1,\ h_{3,2\bar{0}} = 1,$

$h_{3,2\bar{1}} = 1, h_{3,2\bar{2}} = 1$。

由$M_{\bar{p}}$值可看出只有在$h_{3,1\bar{1}}$、$h_{3,1\bar{4}}$與$h_{3,1\bar{6}}$值是 0，藉由前面求出的$\hat{e}_p = \overline{h}_{3,\bar{p}}$，因此，只有在$\hat{e}_{11}$、$\hat{e}_{14}$與$\hat{e}_{16}$的值是 1。因此所有的錯誤數值
可定義如下：

$$(\hat{e}_{11},\hat{e}_{12},\hat{e}_{13},\hat{e}_{14},\hat{e}_{15},\hat{e}_{16},\hat{e}_{17},\hat{e}_{18},\hat{e}_{19},\hat{e}_{20},\hat{e}_{21},\hat{e}_{22}) = (1,0,0,1,0,1,0,0,0,0,0,0)$$

5.結論

經過本篇論文提出的兩種解碼方法來對於(23,12,7)Golay code進行解碼後，可得知無論在理論亦是電路實現在解碼時間的比較上，步階解碼法明顯的優於error-trapping。而在判斷多項式運算M也比F運算上少去了平方根與除法的運算，因此，較適合利用在硬體實現

6.文獻

[1] M.J.E. Golay, “Notes on digital coding,” Proc. IEEE, vol.37, p.657,1949.

[2] S. Lin and D.J. Costello, Jr., Error Control Coding, 2nd ed., Prentice-Hall, 2004.

[3] J.L. Massey, “Step-by-step decoding of the Bose-Chaudhuri-Hocquenghem codes,” IEEE Trans. Inf. Theory, vol.IT-11, no.4,pp.580–585, 1965.

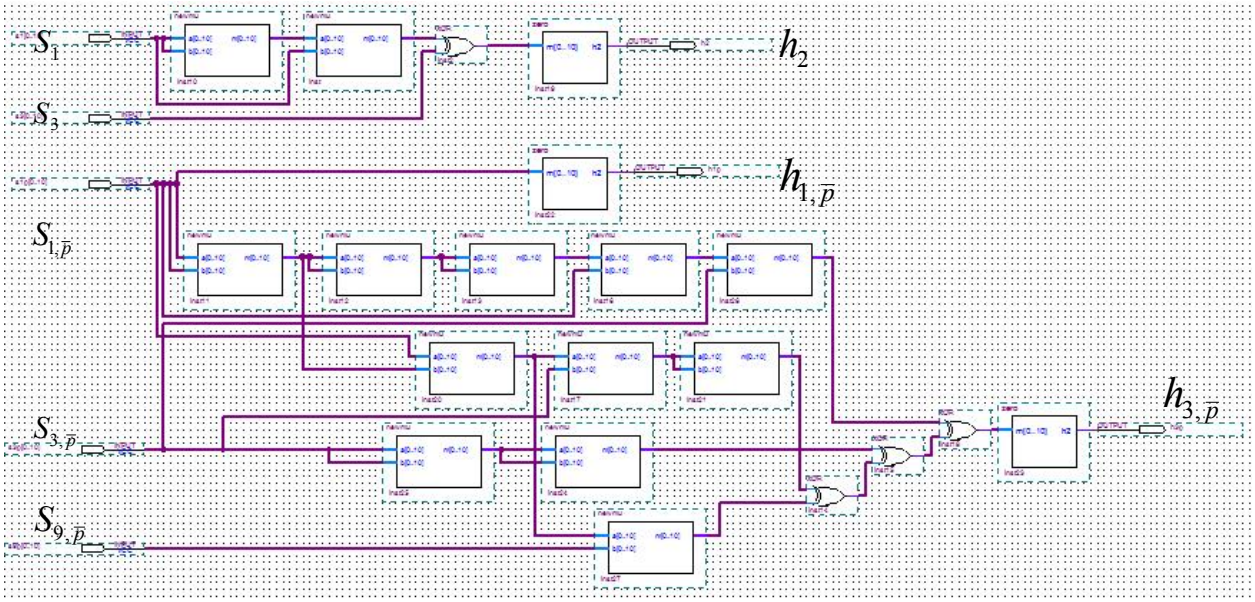
[4] E.-H. Lu and T. Chang, “New decoder for double-error-correcting binary BCH codes,” IEE Proc., Commun., vol.143, no.3, pp.129–132, 1996.

[5] M. Elia, “Algebraic decoding of the (23, 12, 7) Golay code,” IEEE Trans. Inf. Theory, vol.IT-33, no.1, pp.150–151, 1987.

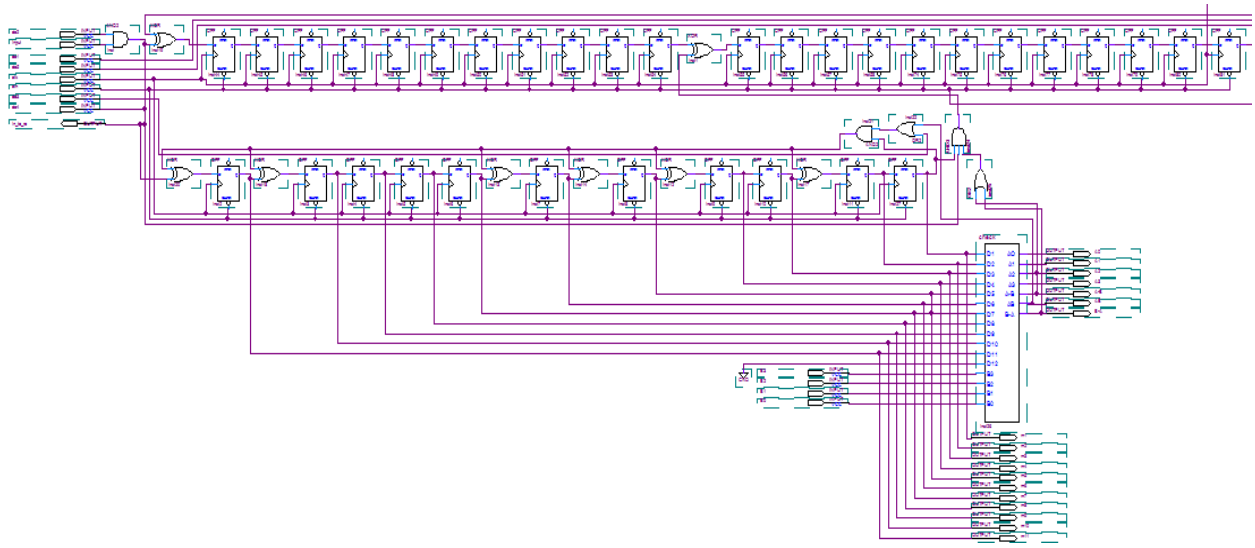
[6] S.W. Wei and C.H. Wei, “On high-speed decoding of the (23, 12,7) Golay code,” IEEE Trans. Inf. Theory, vol.36, no.3, pp.692–695,1990.

[7]C.-L. Chr, S.-L. Su, and S.-W. Wu, “Decoding the (23, 12, 7) Golay Code Using a Low-Complexity Scheme” IEICE Trans. Fundamentals, vol.E89–A,no.8 august 2006.

[8] C.-L. Chr, S.-L. Su, and S.-W. Wu, “A low-complexiy step-by-step decoding algorithm for binary BCH codes,” IEICE Trans Fundamentals, vol.E89–A,no.8 august 2006



圖(3) *M* 函數之電路設計圖



圖(5) error-trapping 解碼器之電路設計圖