國 立 交 通 大 學

電機學院　電機與控制學程

碩 士 論 文

快閃記憶體控制器之軟式決策 BCH 解碼晶片設計

Soft-Decision BCH Decoder Design for

NAND Flash Controller

研 究 生：古明豪

指導教授：董蘭榮　博士

中華民國一百零一年七月

快閃記憶體控制器之軟式決策 BCH 解碼晶片設計

Soft-Decision BCH Decoder Design for NAND Flash Controller

研 究 生：古明豪　　Student: Ming-Haur Ku

指導教授：董蘭榮　博士　　Advisor: Dr. Lan-Rong Dung

國 立 交 通 大 學

電機學院　電機與控制學程

碩 士 論 文

A Thesis

Submitted to College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electrical and Control Engineering

July 2012

Hsinchu, Taiwan, Republic of China

中華民國一百零一年七月

快閃記憶控制器之軟式決策 BCH 解碼晶片設計

學生：古明豪　　指導教授：董蘭榮　博士

國立交通大學　電機學院　電機與控制學程 碩士班

摘要

本篇論文研究背景是以目前因為在快閃記憶體(NAND 架構)製程上不斷縮小，優點可以提高儲存容量和減少成本，但是隨之而來的是儲存記憶體中資料排電壓將會互相干擾會提高導致在讀寫上發生隨機錯誤位元機率升高，傳統的 BCH 硬式解碼也將會隨著製程不斷進步將會碰到許多解碼效益的瓶頸，所以本篇論文的目標是發展一個在以快閃記憶體傳輸規格 512 位元組為基礎的軟式決策解碼晶片實現。軟式解碼是以梯度下降和疊代方式尋找出最大概似度碼字，在 2004 年 JN 提出以里德所羅門碼為背景的軟式解碼，在每次疊代更新信任度資料將會重新排列低信任度位置去尋找最大概似度碼字，目的是避免在梯度下降中的資料傳遞發生訊號錯誤機率，減低碼字發散。在本篇論文從分析和比較是在有限碼字長度底下各個軟式解碼找出最好解碼效益的軟式解碼，本篇論文選擇以 BCH 軟式解碼並且結合 JN 排序的概念做為快閃記憶體軟式解碼器。BCH-JN 軟式解碼晶片實作是以 0.18um 製程做實現，硬體規劃是以快閃記憶體傳輸規格大小和接近浮點數的解碼效益找出定點數值所應該規劃的範圍，另外進一步我們在硬體也分析並且比較軟式解碼中的排序、高斯消去法和梯度下降法，是否有減少軟式解碼硬體運算時間和硬體複雜度的空間。

Soft-Decision BCH Decoder Design for NAND Flash Controller

Student: Ming-Haur Ku **Advisor: Dr. Lan-Rong Dung**

Degree Program of Electrical and Computer Engineering

National Chiao Tung University

Abstract

This paper research background is the technology continues to scale down, the advantage of flash memory can reduce power consumption and cost of the hardware but the threshold voltage shift of one floating gate transistor can influence the threshold voltage of its neighboring floating gate transistors through parasitic capacitance coupling effect in the flash memory. The traditional BCH decoder of hard decision can't support to better efficient of decoding in the new manufacture. The purpose is based on the NAND flash specification of the 512 bytes transmission to design a chip of soft decision decoder. The soft decision used the tanner graph and iterations to search the maximum likelihood codeword. In the 2004, JN provide the new sorting concept in the procedure of soft decision of RS code. The purpose of JN could reduce error probability of the reliability exchange in the tanner graph after Gaussian Elimination. The paper would analysis and compares the soft decision structure to find suitable decoder on the decoding efficient and code length less than 512bytes. We decided the BCH-JN decoding of soft decision in the NAND flash. On the chip design, we based on the 0.18um to plan fix point value range to simulate and compare with real value simulation. In the hardware part, we analysis and compared their operation time and complex of the sorting 、 Gaussian elimination and tanner graph circuit.

誌謝

本篇論文得以順利完成，首先要感謝的是我的指導教授──董蘭榮教授，在碩士班的五年間，董教授不厭其煩地指導我，當我陷入瓶頸時，董教授亦適時地指點我正確的方向，論文上的研究和課堂所修習的知識，讓我更清楚了解一件事情的分析方式，特別是使用機率觀念去看待很多生活周遭所發生的一些事情，回想起剛進實驗室，什麼叫機率? 這是董教授第一次問我的問題，那時候真的不懂，只能夠回答該事情發生的成功或失敗的比率，透過此論文研究讓我深刻體會到不同的分析角度，讓我知道機率發生的真正原因並著手用不同方式去分析該狀況底下發生的條件。

在此非常感謝林傳生學長、連昶祥同學討論，特別是林傳生學長的博士論文報告中敘述的第二章快閃記憶體基礎概念，我之前本身在工作或者學習上都沒有踏入該領域，讓我以此去蒐集並整理其他關於快閃記憶體文獻，找出正確的傳輸規格完成軟式解碼硬體實現和通道模擬。

謹將此論文獻給所有關心我的人，在此致上最深的謝意。

目錄

表格目錄

圖目錄

一、概 論

1.1 研究動機

首先本篇論文必須要先介紹浮動閘干擾(Floating gate interference)觀念，特別這幾年快閃記憶體(Flash Memory)製程(manufacture)不斷進步已經來到 40 nm 製程[1]，其中 2011 年東芝(Toshiba)開始宣布投入 NAND 快閃記憶體 24nm 製程，當快閃記憶體製程來到奈米階段，隨之而來在快閃記憶體發生資料排多準位(multi-level per cell，以下均簡稱 MLC)儲存電壓被干擾，被影響浮動閘是透過週圍被動耦合電容電壓的干擾導致儲存電壓資料發生錯誤，此雜訊成為未來快閃記憶體為了增加資料記憶體儲存密度和成本下降在製程不斷下降所面對的挑戰，圖 1 說明由於記憶體儲存資料排彼此相鄰過近，造成彼此耦合電容效應提高，導致相鄰的資料排會發生受到該資料排 MLC 儲存電壓被干擾強度曲線圖，由於儲存快閃記憶體資料是儲存在該元件浮動閘中，所以被定義為浮動閘干擾，而通常也有些文獻也將此干擾定義為資料排元件與資料排元件互相干擾(cell to cell interference)，本篇論文之後所有敘述均稱為浮動閘干擾。

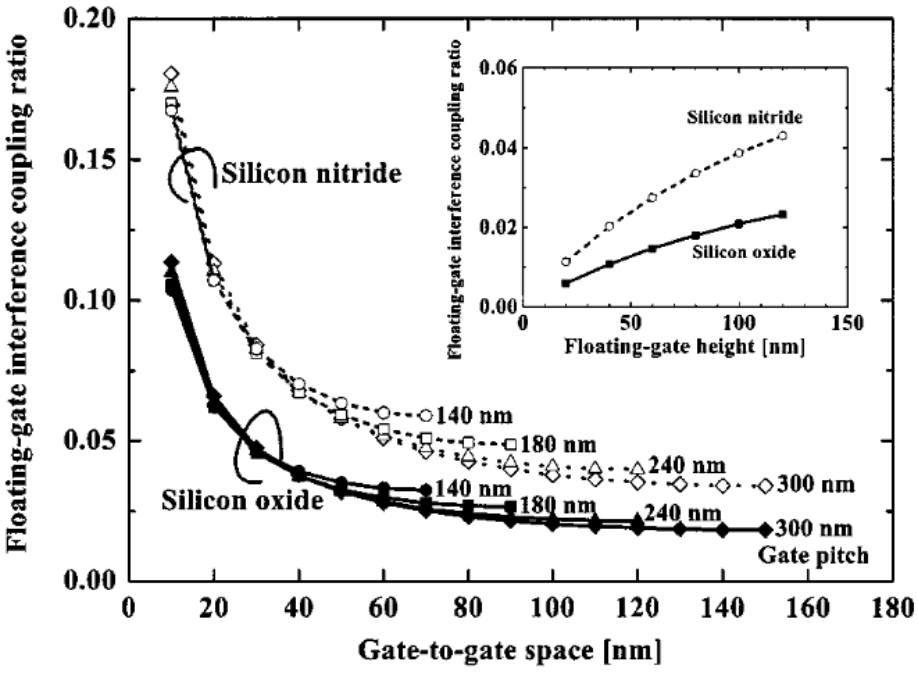


圖 1 在 0.12um 製程，不同閘距離浮動干擾比率

資料來源[2]

這一兩年快閃記憶體硬用不斷蓬勃發展，其中筆記型電腦的儲存設備也已經

改由固態硬碟((Solid State Drive，簡稱為 SSD)成為儲存主流，所以透過圖 2 說明快閃記憶體的浮動閘干擾已經成為我們在高密度製程下面臨儲存資料上最大挑戰，BPD(background pattern dependency，簡稱為 BPD)和 Noise 此兩部分雜訊分別是代表快閃記憶體中儲存裝備耐久度和其他儲存設備干擾分佈圖。

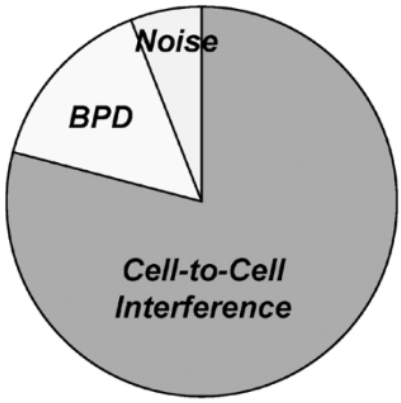


圖 2 在 60nm 製程下資料排雜訊分佈圖
資料來源[1]

傳統快閃記憶體所使用錯誤更正碼(Error control coding，簡稱 ECC)能夠有效提高儲存設備可靠度和耐久度，其中在快閃記憶體使用度最高的錯誤更正碼是 BCH 碼[3] (Bose – Chaudhuri - Hocquenghen，簡稱 BCH code)，硬式決策(hard-decision)解碼器能夠產生更正碼字中的個數為 $t = (N-K) / m$，其中 $N=2^m – 1$、m=位元數以及 K=資料長度，BCH 硬式決策在快閃記憶體被應用最主要的優點是它可以更正隨機錯誤位元的碼字，但是隨著製程在快閃記憶體密度愈高狀況，浮動閘干擾情形會讓 BCH 硬式決策在固定解碼的個數上會逐漸被侷限解碼能力，除非增加校驗碼字(parity check code)的長度得到更多解碼個數，但是會犧牲掉傳輸頻寬和硬體所需要的成本。所以本篇論文針對在快閃記憶體常用傳輸規格 512 位元組(bytes)的浮動閘干擾情形下提出適合軟式決策(soft-decision)解碼方式，常用的軟式決策解碼是低密度奇偶檢查碼(Low-density parity-check code，LDPC code)[3]，軟式決策解碼的核心是使用梯度下降法(Gradient descent algorithm)透過疊代(iteration)找到最大概似度的碼字(maximum likelihood codeword)，優點是能夠找到接近 shannon 最好解碼效益，但是在高碼率狀況下的低碼字長度，低密度奇偶檢查碼的解碼效益反而沒有在相同碼率底下長度高的的碼字所解碼出來的效益這麼好，所以本篇論文透過分析和比較軟式解碼，提出 BCH-JN 軟式決策解碼在快閃記憶體浮動閘干擾情形下使用，該軟式解碼的解碼效益將會比低密度奇

偶檢查碼還要好，此分析報告將會在第三章做詳細說明，其中 JN[4]是一個 2004 年提出低信度更新位置的觀念，改善梯度下降法訊息傳遞發生錯誤機率。

軟式決策解碼在硬體實現上的需要注意最主要就是運算速度，在軟體模擬運算我們考慮以 5 次疊代並且在軟式決策解碼上的規格與實際硬體規格相同，軟體運算時間上我取平均值並且以微控制器(MCU)-50MHz 做為軟體運算時序頻率，結果計算 C 語言大約要執行約 70ms 才能完成一次軟式解碼，但在硬體解碼時間我取可能運算最大時間並且頻率同樣以 50MHz 做模擬，最多需要 0.9ms 即可完成一次軟式解碼。

另外本研究主要提出在快閃記憶體在傳輸規格在 512 位元組的 BCH-JN 軟式解碼器硬體設計並且在各個不同耦合電容干擾強度下探討軟式解碼效益，軟式解碼評估以里德-所羅門碼[1] (Reed-Solomon，RS code)、JN 所提出軟式解碼以及低密度奇偶檢查碼解碼演算法在有限碼字軟式決策解碼效益和硬體複雜度，最後並且透過快閃記憶體規格書[5]訂定出在快閃記憶體控制器軟式解碼器硬體的規格並且尋找出適當的硬體參數，使定點數接近浮點數模擬解碼效益。

1.2 論文架構

第二章是 BCH 碼、軟式解碼演算法和快閃記憶體基礎理論，定義論文所提出 BCH 軟式解碼演算法中所需的基礎定義以及快閃記憶體控制器基本硬體架構。

第三章是快閃記憶體軟式決策解碼，評估 BCH 碼、里德-所羅門碼和低密度奇偶檢查碼解碼演算法在碼字長度 2000 的評估和比較，並且說明軟式解碼演算法在硬體實現之考量，以定點(fixed-point)運算來取代浮點數模擬，分析其因有限字元長度運算及量化所造成誤差。

第四章是快閃記憶體控制器硬體實現，模擬並驗證軟式解碼器在硬體上的表現，並且評估和分析軟式解碼器在硬體適合的方法為何。

第五章為結論和未來方向，會針對前面章節之結果做綜合性的探討與困難提出，並陳述研究心得及個人認為未來值得繼續研究方向。

二、

BCH 碼、軟式解碼演算法和快閃記憶體基礎理論及架構

本章節中將定義相關的符號標示法將介紹軟式解碼演算法基礎理論和快閃記憶體控制器，軟式解碼理論基最後在此章節將以圖形化方式說明其運作動作。

2.1 BCH 碼基礎理論

BCH 碼[3] (Bose – Chaudhuri - Hocquenghen，簡稱 BCH code)，目前已被大量運用在快閃記憶體(Flash Memory)產品運用上，錯誤更正碼(Error control coding, 簡稱 ECC)能夠有效提高儲存設備可靠度和耐久度；對一個(N,K)BCH 碼而言，使用傳統的硬式決策(hard-decision)解碼器能夠產生更正碼字中的個數為 t = (N-K) / m，其中 $N=2^m - 1$。

2.1.1 BCH 碼之符號定義

(*N,K*) BCH 碼運作於 GF(2^m)之下時，則碼字長度為 $N=2^m-1$，資料長度為 K，則可求得其更正錯誤能力 t = (N-K) / m。而奇偶校驗矩陣(1)式是由文獻[3]中所介紹 BCH 碼校驗矩陣的形式來表示:

$$H_s = \begin{bmatrix} 1 & \beta & \beta^2 & \beta^3 & \cdots & \beta^{n-1} \\ 1 & (\beta^3) & (\beta^3)^2 & (\beta^3)^3 & \cdots & (\beta^3)^{n-1} \\ 1 & (\beta^5) & (\beta^5)^2 & (\beta^5)^3 & \cdots & (\beta^5)^{n-1} \\ \vdots & & & & & \vdots \\ 1 & (\beta^{2t-1}) & (\beta^{2t-1})^2 & (\beta^{2t-1})^3 & \cdots & (\beta^{2t-1})^{n-1} \end{bmatrix} \quad (1)$$

H_s 為 2t-1 列 N 行之矩陣，在這裡 β 為 GF(2^m)中的質元素(primitive element)，在有現場 β 是獨立並且唯一元素。

透過(1)式，H_s 可藉由 GF(2^m)中選擇的基底，$1, \beta, \beta^2, \cdots, \beta^{n-1}$，由原有的符號

奇偶校驗矩陣映射成大小為 $(n-k)\times n$ 的二位元奇偶校驗矩陣 H_b。經過二位元映射後，我們可以將 BCH 的解碼問題，看待成大小為$(n-k)\times n$的二位元線性區塊碼 (binary linear block code) 解碼問題。

考慮一 BCH 碼其 $N=2^3-1=7$，K=3，m=3，t=(7-1)/3 = 2，且奇偶校驗矩陣 H_s 為(2)式

$$H_s=\begin{bmatrix}1 & \beta & \beta^2 & \beta^3 & \beta^4 & \beta^5 & \beta^6\\ 1 & \beta^3 & \beta^6 & \beta^2 & \beta^5 & \beta & \beta^4\end{bmatrix} \quad (2)$$

此校驗矩陣(Parity Matrix)長度的位置會將會對應到各個編碼位置，每個位置皆為獨立向量，並以此校驗矩陣做為軟式解碼演算法的初始矩陣。

2.1.2 BCH 碼二位元映射基底和加法運算

本研究中，以 GF(2^m)中的符號 (symbol) $1,\beta,\beta^2,\cdots,\beta^{m-1}$ 做為符號映射至二位元向量之基底。如 A∈GF(2^m)唯一符號，其可投影在 $1,\beta,\beta^2,\cdots,\beta^{m-1}$ 所組成的基底上，而以$A=\sum_{i=0}^{m-1}A_b^{(i)}\beta^i$或$A_{VECTOR}=\left[A_b^{(0)}\ A_b^{(1)}\ \cdots A_b^{(m-1)}\right]$的形式來表示，其中$A_b^{(i)}\in GF(2)$.

於 GF(2^3)時，使用的質多項式 (primitive polynomial) 為 $1+x+x^3$，質多項式一般就稱之為線性循環編碼(cyclic code)，在循環編碼中就可以獲得除法過後所得的餘數，餘數被稱之為校驗碼(parity ceck code)，所以線性循環編碼就是一個除法器，透過循環編碼字則 GF(2^3)中的每個元素均可用 GF(2) [x] mod($1+x+x^3$)的(3)式方式表示如下，其中 β=x,

$0=0+0\cdot x+0\cdot x^2$=[0,0,0], $1=1+0\cdot x+0\cdot x^2$ =[1,0,0],
$\beta=0+1\cdot x+0\cdot x^2$=[0,1,0], $\beta^2=0+0\cdot x+1\cdot x^2$=[0,0,1],
$\beta^3=1+1\cdot x+0\cdot x^2$=[1,1,0], $\beta^4=0+1\cdot x+1\cdot x^2$=[0,1,1],
$\beta^5=1+1\cdot x+1\cdot x^2$=[1,1,1], $\beta^6=1+0\cdot x+1\cdot x^2$=[1,0,1], (3)

每個符號映射為二位係數的向量後，仍然需要滿足有限場的觀念在 GF(2^m)中的加法特性，其中在 GF(2)二位元加法取代原有的符號加法運算就是以 XOR 取代原有加法。先前所選取的基底 $1,\beta,\beta^2,\cdots,\beta^{n-1}$，其在加法上可直接由向量的 GF 相加，使其(4)式加法仍然滿足原有的代數特性。

$$\beta+\beta^5=(0+1\cdot x+0\cdot x^2)+(1+1\cdot x+1\cdot x^2)=[0,1,0]+[1,1,1]$$
$$=[1,0,1]=1+0\cdot x+1\cdot x^2=\beta^6 \quad (4)$$

2.2 軟式解碼演算法基礎理論

2.2.1 通道環境符號定義

在本章節中假設所使用的通道環境均為可加性高斯白雜訊 (additive white Gaussian noise，AWGN) 通道；調變方式為二位元移鍵 (binary phase shift keying，BPSK) 調變，以下均簡稱 BPSK，調變將 0 映射至-1 且 1 映射至+1 後，最後傳送於高斯通道的過程可以由 (5) 式所表示

$$y = (2c-1) + n \quad (5)$$

其中 $y=[y_1,y_2,\cdots,y_n]$ 表示經過 BPSK 調變和高斯通道後的接收信號，而 $n=[n_1,n_2,\cdots,n_n]$ 表示高斯通道上的雜訊，且其雙邊雜訊功率(σ^2)為 $N_0/2$。

獲得接收信號後，透過對數概似比例 (log likelihood ration，以下均簡稱為 LLR)(6)式，即可計算碼字中每個位元的信任度(Reliability)，透過高斯通道轉換所得到位元錯誤機率可以用此式表示 $p(y_i \mid d_i=+1/-1)$，此式被定義為機率密度方程式(probability density function,簡稱 pdf)，取對數(log)物理意義就是在接收信號中，我們要將所信任度的值區分出來，透過(6)式推導，根據此推導式子可以得到雜訊的分佈密度，我們可以定義此高斯通道初始的信任度，最後在計算信任度初始資料中，概似(likelihood)物理意義就是用已知的資料(N_0)，藉以求取碼字上的初始信任度大小。

$$L(c_i)=\log_e\left[\frac{p(y_i\mid d_i=+1)}{p(y_i\mid d_i=-1)}\right]=\log_e\frac{\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{y_i-1}{\sigma}\right)^2\right]}{\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{y_i+1}{\sigma}\right)^2\right]}$$

$$=-\frac{1}{2}\left(\frac{y_i-1}{\sigma}\right)^2+\frac{1}{2}\left(\frac{y_i+1}{\sigma}\right)^2=\frac{2}{\sigma^2}y_i=\frac{4}{N_0}y_i \quad (6)$$

2.2.2 梯度下降法理論基礎(Gradient descent algorithm)

梯度下降法的基礎理論就是來自於低密度奇偶檢查碼中的 Tanner，與硬式解碼器比較此梯度下降法就等於求碼字的徵狀(syndrome)和發生錯誤位置。

在進行梯度下降法之前，摘錄於 JN 文獻[10]中的第(9)式，將 LLR 對應到 tanh 域函數$[-\infty,+\infty]\rightarrow[-1,+1]$，所以透過(7)式代表在外部額外訊息位元傳遞中的信任度大小。

$$v(L)=\tanh\left(\frac{L}{2}\right) \tag{7}$$

將每個碼字位置的 LLR 轉換至 tanh 域，所以透過(8)式可以表達從接收信號求得每個位置的 LLR(c_i)。

$$T=[T_1,T_2,\cdots T_n]=[v(L(c_1)),\cdots,v(L(c_n))] \tag{8}$$

其中 T 為經過轉換後的 tanh 域向量。接下來，利用(8)式定義 H_b 中第 j 校驗方程式 (即 H_b 中的第 j 列) 的決策信任度 γ_j (decision reliability) 於式(9)式 中，連乘在式子中是代表訊息傳遞，利用校驗矩陣提供各點訊息找出所求的校驗節點信任度大小，

$$\gamma_j=\prod_{p\in\Gamma j}^{n}v(L(c_p)) \tag{9}$$

其中 $\Gamma_j\in\lfloor a\mid\forall 1\le a\le n,H_{bja}=1\rfloor$。

在軟式解碼中，γ_j 同時也代表外部額外消息資料量(extrinsic information)。γ_j 在軟式解碼所代表的意義有兩項

1. 若將初始 LLR 信任度經過硬式決策後，則可計算 Tanh 相對於 H_b 的徵狀(syndrome)。當 $\gamma_j > 0$ 時，表示徵狀中第 j 個位置為 1。若 $\gamma_j < 0$，則表示徵狀中第 j 個位置為 0。
2. 一般訊號傳遞信任度，若 H_b 中第 j 個校驗方程式所校驗到的位元位置 (即 H_b 的第 j 列於該位置之值為 1)，其 LLR 的絕對值較大時，會認為該位置的硬式決策較不易發生錯誤，若校驗到位元位置其 LLR 之絕對值較小時，會認為該位元位置的硬式決策較易發生錯誤。同樣的概念應用於此處的決策信任度上，當$|\gamma_j|$愈大時，會認為徵狀中第 j 個位置的決策較不易發生錯誤，當$|\gamma_j|$愈小時，會認為該決策較易發生錯誤。由 (8)、(9)兩式可知，γ_j 必定介於[-1,+1]間。

因此若 LLR 經過 H_b 中第 j 個校驗方程式校驗為一合法碼字，則 $\gamma_j > 0$，且$|\gamma_j|$愈大表示我們能夠愈信任這個校驗決策，反之$|\gamma_j|$愈小，則表示這個校驗的決策愈容易發生錯誤。

接下來以(10)式定義一位能函數 (potential function) J，也就是透過校驗矩陣和 γ_j 信任度大小的額外消息資料量總合，很多文獻將此定義為和積演算法(Sum Product Algorithm)，接下來下面所訂義位能的含義就是將上述所獲得的額外消息量與初始信任度互相比較，以此方式找出最大概似度碼字。

定義 2.1: 定義位能函數 J 為

$$J(H_b,T) = -\sum_{j=1}^{(n-k)} \gamma_j = -\sum_{j=1}^{(n-k)} \prod_{p\in\Gamma j} T_p \tag{10}$$

其中 $\Gamma_j \in \lfloor a \mid \forall 1 \le a \le n, H_{bja} = 1 \rfloor$，且 J 是一個同時受 H_b 和 T 所影響的函數。

由以上的討論可知，若一個校驗方程式對於 LLR 的校驗結果一定為正確時，我們會認為其$|\gamma_j|$為 1。而當 $\gamma_j > 0$ 且$|\gamma_j| = 1$ 時，則表示我們認為該校驗方程式對 LLR 所做的校驗結果必定徵狀為 0，也就是該校驗方程式認為其一定為合法的碼字。

以一個合法的碼字而言，其必定滿足所有的校驗方程式，而徵狀必定均為 0。因此在校驗合法碼字的情況下，每個校驗方程式的 γ_j 必定為 1，且此時所有校驗方程式的 γ_j 總和為(n-k)。由(10)式可知此時位能函數值 J 會達到最小值-(n-k)。故若能由通道所得到的信任度 T 為初始起點，使接收信號的 LLR 往位能函數最低點的方向前進，並收斂於位能函數的最低點時，則被認為此過程成功找到了傳送的碼字而成功解碼。

最後以(11)式來描述梯度下降法搜尋位能函數最低點的過程

$$T^{(l)} - \alpha \nabla J(H_b, T^{(l)}) \rightarrow T^{(l+1)} \tag{11}$$

這裡的 α 為一抑制係數，如同梯度下降法中控制每次疊代的步距 (step size) 係數，上標(l)表示目前為第 l 次疊代。由於信任度 T_i 以 tanh 域表示時，其被定義在[-1,+1]的區間內，因此需要修正原來梯度下降法的疊代函數，以保證信任度 T_i 使終於[-1,+1]的區間內。經過最後修正後，可得到(12)式的形式。

$$v^{-1}(T_i^{(l)}) - \alpha\left[-\sum_{j\in\Lambda i} v^{-1}\left(\prod_{p\in\Omega j} T_p^{(l)}\right)\right] \rightarrow v^{-1}(T_i^{(l+1)}) \tag{12}$$

此時 $\Lambda_j \in \lfloor q \mid \forall 1 \le q \le n-k, H_{bqi} = 1 \rfloor$ 且 $\Omega_j \in \lfloor s \mid \forall 1 \le s \le n, H_{bjs} = 1, s \ne i \rfloor$。$T_p^{(l)}$和 $T_i^{(l)}$之上標(l)表示目前為第 l 次疊代。

實際運作上，由(7)和(12)式，每個位置的 LLR ($L(c_i)$) 在第 l 次疊代過程中所接收到的額外消息量(extrinsic information)總值$L_{ext}^{(l)}(c_i)$可以表示如(13)式。

$$L_{ext}^{(l)}(c_i) = \sum_{j\in\Lambda_i} v^{-1}\left(\prod_{p\in\Omega_j} T_p^{(l)}\right) = \sum_{j\in\Lambda_i} 2\tanh^{-1}\left(\prod_{p\in\Omega_j} \tanh\left(\frac{L^{(l)}(c_p)}{2}\right)\right) \tag{13}$$

其中 $\Lambda_j \in \lfloor q \mid \forall 1 \le q \le n-k, H_{bqi} = 1 \rfloor$ 且 $\Omega_j \in \lfloor s \mid \forall 1 \le s \le n, H_{bjs} = 1, s \ne i \rfloor$，$T_p^{(l)}$和 $T_i^{(l)}$之上標(l)表示目前為第 l 次疊代。而由(13)式可知，額外消息量總值中，由第 j 個校驗方程式所提供的消息量可用(14)式來表示，這裡以上標(l)(j)來表示額外消息量 $L_{ext}^{(l)(j)}(c_i)$ 由第 j 個校驗節點所提供。

$$L_{ext}^{(l)(j)}(c_i) = 2\tanh^{-1}\left(\prod_{p\in\Omega_j} \tanh\left(\frac{L^{(l)}(c_p)}{2}\right)\right) \tag{14}$$

其中 $\Omega_j \in \lfloor s \mid \forall 1 \le s \le n, H_{bjs} = 1, s \ne i \rfloor$。

接下來我們將梯度下降法的解碼執行步驟陳述如下:

步驟1 設定抑制係數 α，最大疊代次數 $l_{\max}$，並計算接收信號中每個元素的初始 LLR 為 $L^{(0)}(c_i) = (4y_i / N_0)$，並以初始 LLR $\left(L^{(0)}(c_i)\right)$開始進行作解碼動作。

步驟2 利用梯度下降法於每次疊代中，計算每個位元位置所得到的額外消息量 $L_{ext}^{(l)}(c_i)$，計算方式如前小節之(14)式。

步驟3 經由方程式 $L^{(l+1)}(c_j) = L^{(l)}(c_j) + \alpha L_{ext}^{(l)}(c_i)$，將每個位元位置的 LLR $\left(L^{(l)}(c_i)\right)$ 進行更新，此過程及前一小節中(13)式的運作。解碼過程中，若已達預設的最大疊代次數 $l_{\max}$，或者疊代過程中經硬式決策後能得到一個合法碼字時，則停止解碼並輸出目前的解碼結果。若未達預設最大疊代次數且未得到合法

碼字時，則以更新後的$\left(L^{(l+1)}(c_i)\right)$，回到步驟 2 進行下一次疊代。

1.1梯度下降法之圖形化運作方式

經過二位元映射後的奇偶校驗矩陣均可用以下的 Tanner 圖來表示，如

$$H_b = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

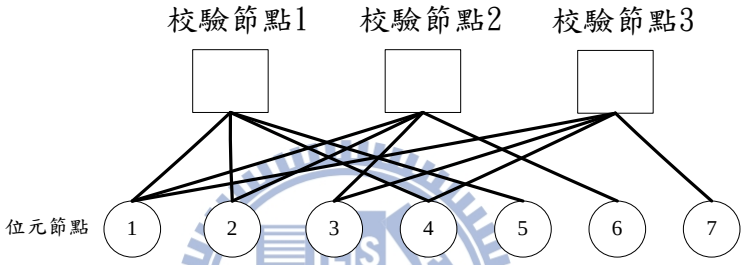


圖 3 使用 Tanner 圖表示二位元奇偶校驗矩陣

H_b 上的每一列，在 Tanner 圖上是使用校驗節點 (check node) 表示；而碼字上的每個元素 c_i，在 Tanner 圖上則是用相對應的位元節點 (bit node) 來表示。當 H_b 中的第 i 列第 j 行的位置為 1 時，則將第 i 列所代表的校驗節點和第 j 行的位元節點在圖形上以相連來表示。則原有的梯度下降法對應至 Tanner 圖上時，可利用簡單的圖形化動作來表示其運作過程。

圖 4 表示梯度下降法的初始過程，即前一小節中的步驟 1。每個位元節點均可經由接收信號得到一初始的 LLR 值: $L^{(0)}(c_i) = \left(4y_i / N_0\right)$，並且傳送給相連的校驗節點。

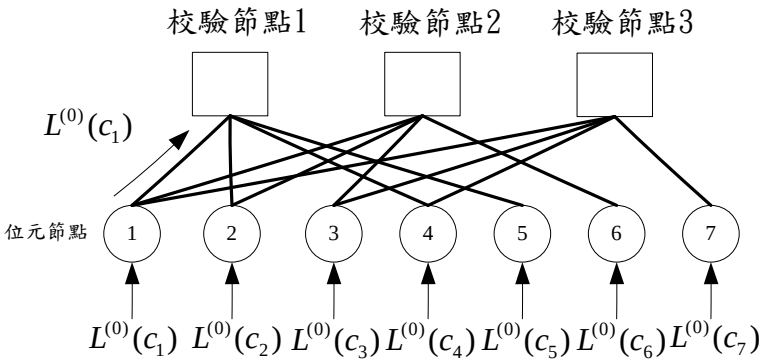


圖 4 給予每個位元節點初始 LLR 示意圖

圖 5 為梯度下降法計算第 l 次疊代時，校驗節點提供位元節點額外消息量的表示圖，即(14)式的過程。以校驗節點 1 為範例，計算其給予位元節點 5 之額外消息量時，其消息來源分別來自位元節點 1、位元節點 2、位元節點 4。

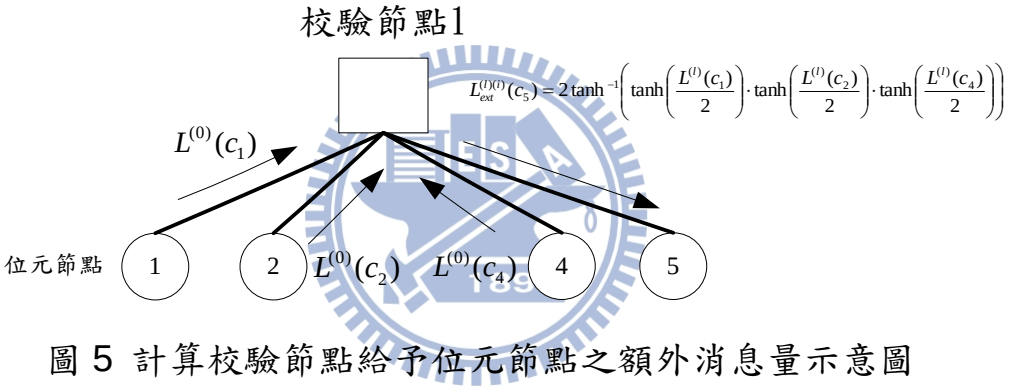
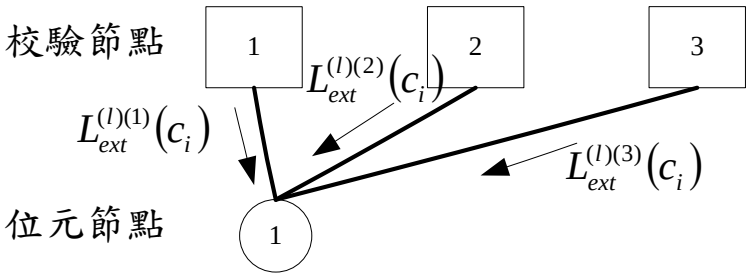


圖 5 計算校驗節點給予位元節點之額外消息量示意圖

圖 6 則為梯度下降法更新位元節點 LLR 的動作，即式 (2.14) 的過程。以位元節點 1 為範例，其於該次疊代中得到校驗節點 1、校驗節點 2 和校驗節點 3 所提供的額外消息量 $L_{ext}^{(l)(1)}(c_i)$、$L_{ext}^{(l)(2)}(c_i)$和 $L_{ext}^{(l)(3)}(c_i)$。



$$L^{(l+1)}(c_1) = L^{(l)}(c_1) + \alpha \cdot \left[L_{ext}^{(l)(1)}(c_1) + L_{ext}^{(l)(2)}(c_1) + L_{ext}^{(l)(3)}(c_1)\right]$$

圖 6 更新每個位元之 LLR 示意圖

2.2.3 JN 解碼演算法基礎理論

在軟式決策解碼方面，Forney[6] 和 Chase[7] 提出基於信任度的解碼方式，利用翻轉碼字的最小信任位置 (least reliability positions，簡稱 LRPs) ，試圖尋找和接收信號相關信最大的合法碼字。2004 年 Jiang 和 Narayann 就提出隨機平移解碼演算法 (stochastic shifting based iterative decoding scheme，簡稱 SSID) [4]。其利用里德所羅門碼經二位元映射後的二位元奇偶校驗矩陣 (binary parity check matrix) 和其為循環碼的特性，在每次疊代進行信任度傳遞演算法中的梯度下降演算法 (gradient descent algorithm) ，亦進行二位元奇偶校驗矩陣的隨機平移動作，以減少錯誤傳遞所造成的影響。

首先我們二位元映射後得到二位元奇偶校驗矩陣 H_b，再利用此 H_b 進行接下來的解碼動作。解碼過程中，每次的疊代動作主要可分為兩個階段如下:

1. 更新奇偶校驗矩陣階段

於此階段，首先將所有位元節點 LLR 依照其信任度排序，其後按照信任度由小至大的對應位置進行高斯消去法，使 H_b 成為一簡化梯式 (row reduced echelon form) 矩陣。此動作由最小信任度的位置開始，利用列運算將該位置正規化後，再對信任度次高的位置進行同樣的動作，若發現該位置無法正規化的位置，則略過該位置進行信任度次高的下一個位置，直到得到 H_b 的簡化梯式矩陣 H_b'和 (n-k)個成功正規化的位置，稱為最小信任獨立位置 (least reliable independent positions，簡稱 LRIPs)，這些位置可以經由重新排序而成為依正規化矩陣 (identity matrix)。而非最小信任獨立位置的部分於本論文中則稱為最大信任度位置 (most reliable positions，簡稱 MRIPs)，由於 H_b'中對應到這些 LRIPs 的行向量，均只有一個 1，因此這些部分亦稱為 H_b'的低密度位置，反之這些位置以外則稱為 H_b'的高密度位置，如圖 7 所示。

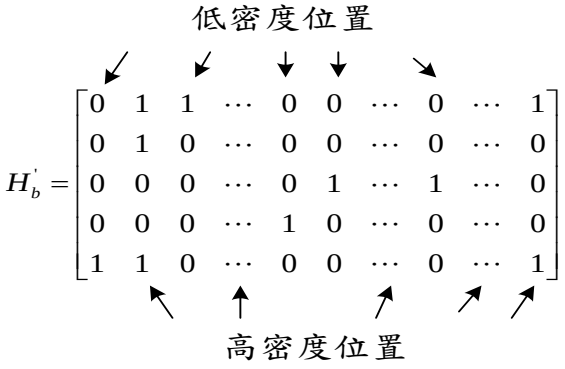


圖 7 經過高斯消去法後，二位元奇偶校驗矩陣內的低密度及高密度位置

2. 更新位元節點之 LLR 階段

Jing Jiang 和 Narayanna 於[4]中提出，一般的梯度下降法運作時，若一個校驗節點相連至多個低信任度的位元節點，則受這些低信任度的位元節點的影響，校驗節點傳送給位元節點的訊息量大部分會近似為 0，因此造成梯度下降法無法提供位元節點足夠的額外消息量以更正錯誤，此現象稱為校驗節點飽和化 (check node saturation)，如圖 8 所示。因此使用 JN 解碼演算法，則可使一個校驗節點僅連接至一個低信任度的位元節點，減少校驗節點飽和化所帶來的影響，以增進其效能。

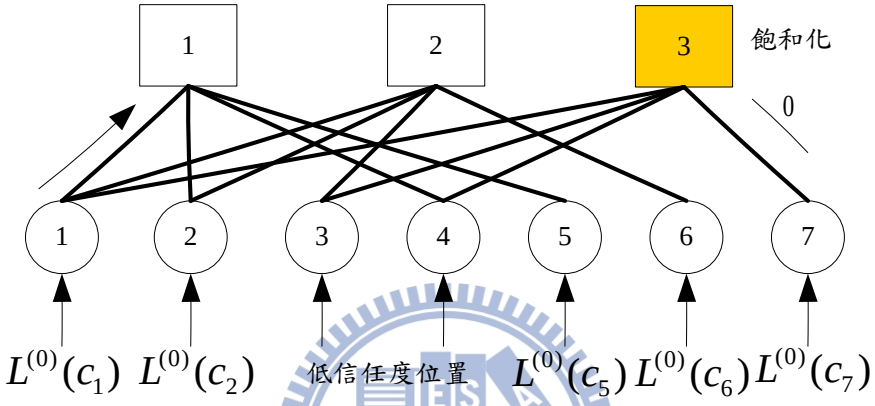


圖 8 過多低信任度位置造成梯度下降演算法無法提供位元節點

JN 演算法執行的詳細步驟說明如下:

步驟1. 設定抑制係數 α，最大疊代次數 $l_{\max}$，並計算每個位元節點初始的 LLR 為 $L^{(0)}(c_i)=(4y_i/N_0)$，並以初始 $LLR(L^{(0)}(c_i))$ 開始進行解碼。

步驟2. 更新奇偶校驗矩陣: $H_b^{(l)}=\phi\left(H_b,\left|L^{(l)}\right|\right)$，$\phi$ 表示更新 H_b 的動作。

A. 依據 LLR 絕對值 $\left|L^{(l)}\right|$ 的大小將每個位元位置進行排序，並且記錄排序後的順序。

B. 利用高斯消去法，按照信任度由小至大的位置將 $H_b^{(l)}$ 消為簡化梯式矩陣，並得到(n-k)個 LRIPs。

步驟3. 利用高斯消去法正規後的 $H_b^{(l)}$，進行梯度下降法得到每個位元節點的額外消息量: $L_{ext}^{(l)}=\phi\left(H_b^{(l)},L^{(l)}\right)$，即式(2.14)的過程。$\phi$ 表示計算額外消息量之動作。

步驟4. 更新每個位元節點的 LLR: $L^{(l+1)} = L^{(l)} + \alpha L_{ext}^{(l)}$ ，即式(12)的過程。在這裡 0< α <1。

步驟5. 硬式決策: $\hat{c}_i = \begin{cases} 1, & L^{(l+1)}(c_i) > 0 \\ 0, & L^{(l+1)}(c_i) < 0 \end{cases}$ 。

步驟6. 終止條件:若所有的校驗節點均被滿足，或當到達最大的預設疊代次數時，則停止疊代並且輸出目前經硬式決策後每個位元的決策值。若未達最大疊代次數，則令 $l = l + 1$ ，回到步驟 2 進行下一次的疊代。

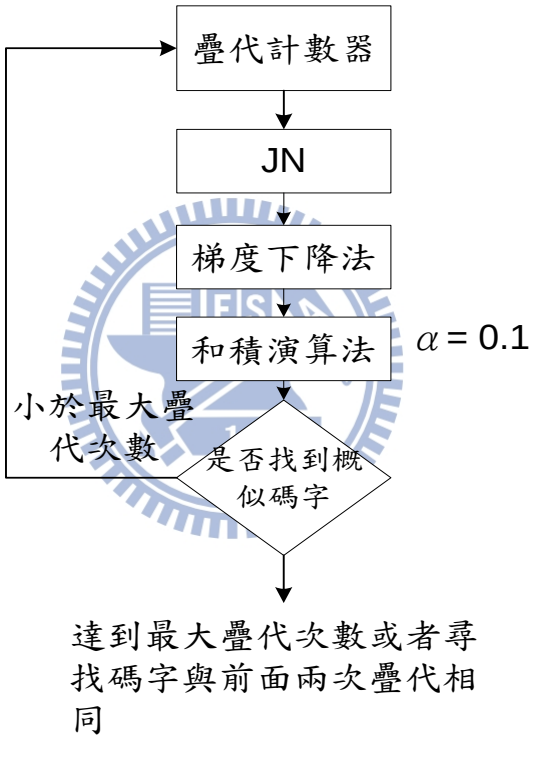


圖 9 JN 解碼演算法示意圖

接下來利用圖 9，本篇論文使用(7,4)漢明碼作為範例，圖 10 將原始的二位元奇偶校驗矩陣 H_b 和相對應每個位元位置的初始信任度 LLR。

$$H_b = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

位元 1	位元 2	位元 3	位元 4	位元 5	位元 6	位元 7
1.2	-0.01	10.0	4.3	-5.2	-6.6	-7

圖 10 JN 演算法之初使化範例

首先經過排序，由小至大的順序分別為圖 11 所示。

位元 1	位元 2	位元 3	位元 4	位元 5	位元 6	位元 7
1.2	-0.01	10.0	4.3	-5.2	-6.6	-7.0
順序 2	順序 1	順序 7	順序 3	順序 4	順序 5	順序 6

圖 11 JN 解碼演算法排序範例

接下來將二位元奇偶校驗矩陣，依信任度排序由小到大進行高斯消去動作後，得到對應的簡化梯式矩陣 $H_b^{'}$、LRIP 對應的 Tanner 圖，如圖 12 所示。

$$H_b^{'} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

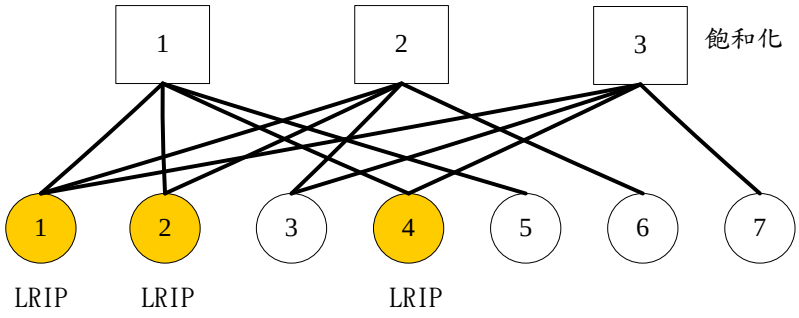
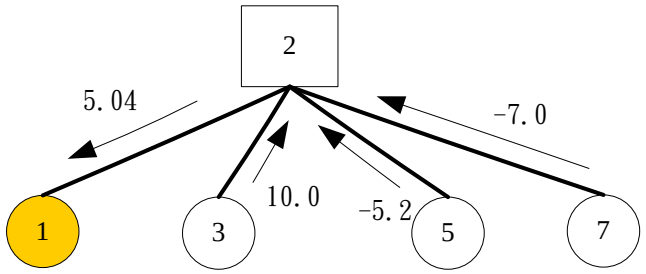


圖 12 JN 演算法之高斯消去動作範例

接下來利用經高斯消去動作後的簡化梯式矩陣 $H_b^{'}$進行梯度下降演算法，並得到每個位元節點的額外消息量，更新原有之 LLR，如圖 13 所示。

位元節點1之額外消息量計算和LLR更新



$$L_{ext}^{(l)(2)} = 2\tanh^{-1}\left(\tanh\left(\frac{L^{(l)}(c_3)}{2}\right)\cdot\tanh\left(\frac{L^{(l)}(c_5)}{2}\right)\cdot\tanh\left(\frac{L^{(l)}(c_7)}{2}\right)\right)$$

$$= 2\tanh^{-1}\left(\tanh\left(\frac{10.0}{2}\right)\cdot\tanh\left(\frac{-5.2}{2}\right)\cdot\tanh\left(\frac{-7.0}{2}\right)\right)$$

$$= 5.04$$

$$L^{(l+1)}(c_1) = L^{(l)}(c_1) + \alpha\left[L_{ext}^{(l)(2)}(c_1)\right] = 1.2 + 1\cdot[5.04]$$

此時α預設為 1，如以上之步驟，可計算所有位元節點之額外消息量和並進行 LLR 更新

	位元 1	位元 2	位元 3	位元 4	位元 5	位元 6	位元 7
原始的 LLR	1.2	-0.01	10.0	4.3	-5.2	-6.6	-7.0
	位元 1	位元 2	位元 3	位元 4	位元 5	位元 6	位元 7
額外消息量	5.04	6.07	1.17	-4.86	2.95	3.92	2.72
	位元 1	位元 2	位元 3	位元 4	位元 5	位元 6	位元 7
更新的 LLR	6.24	6.06	11.17	-0.56	-2.15	-2.68	-4.28

圖 13 JN 演算法之 LLR 更新範例

接下來以更新後的 LLR 進行下一次的疊代過程。

2.3 快閃記憶體 (Flash Memory)

快閃記憶體常用傳輸規格為512位元組(bytes)，快閃記憶體的標準物理結構，稱之為基本位元(cell)；一般 MOS 閘極(Gate)和通道的間隔為氧化層之絕緣(gate oxide)，而快閃記憶體的特色是在控制閘(Control gate)與通道間多了一層稱為“浮閘”(floating gate)的物質，透過這層浮閘儲存外部進來電壓，使得快閃記憶體可快速完成讀、寫、抹除等三種基本操作模式；就算在不提供電源給記憶體的環境下，也能透過此浮閘，來保存資料的完整性。目前大部分快閃記憶體是以 2 位元(bits/cell)為主，此儲存方式能夠有效降低成本以及增加儲存密度，所以廣泛被應用在各種快閃記憶體儲存產品中。目前在不同的製程底下，軟式決策解碼(Soft-decision decoder)已經開始在文獻[8]，[9]討論在快閃記憶體在浮動閘干擾的解碼效益，分別為低密度奇偶檢查碼[1] (Low-density parity-check code，簡稱 LDPC code)和 BCH-Chase 軟式決策解碼演算法的應用。

2.3.1 快閃記憶體介紹

快閃記憶體(Flash Memory)是非掉電易失性記憶體的一種，具有關掉電源仍可保存資料的優點，同時又具備掉電易失性記憶體可重複讀寫且讀寫速度快、單位元體積內可儲存最多資料量，以及低功耗特性等優點。

快閃記憶體的構造及工作原理與 EEP-ROM 相似，不同的是以大區塊(通常是千位元組)為清除單位來抹除資料，抹除與寫入的速度很快，故稱為「快閃(Flash)」。依照不同的 IC 設計方式，又可以分為「NOR 閘型快閃記憶體」與「NAND 閘型快閃記憶體」兩大類，具有不同的特性，分別應用在不同的產品上。

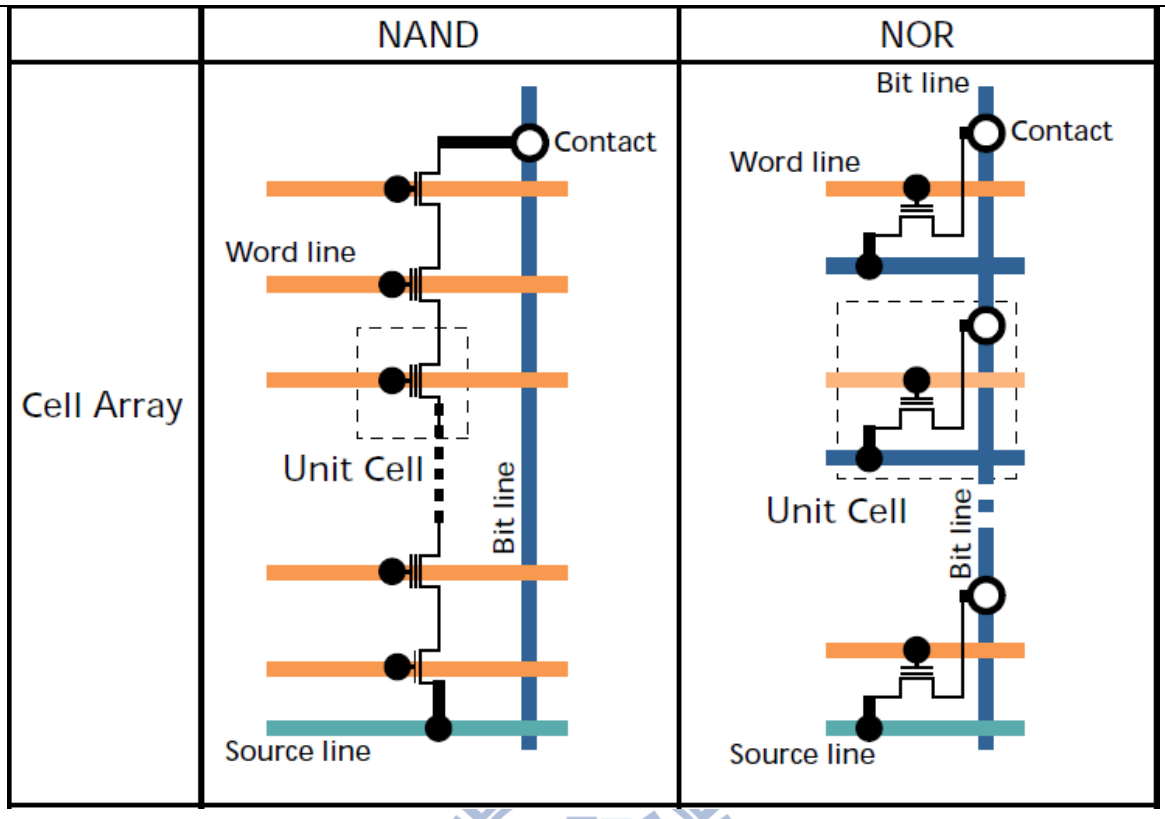


圖 14 快閃記憶體硬體架構圖

資料來源[10]

1. NOR 閘型快閃記憶體(NOR gate flash memory)

使用「NOR 閘」為基本邏輯元件所設計的快閃記憶體，特性是容量較低，價格較高，又稱為「Code storage flash」，一般用來儲存資料量較少的軟體程式，例如：行動電話內的開機程式(作業系統)、個人數位助理(PDA)或智慧型手機的作業系統(例如：Linux、WinCE、Windows mobile)等，讀寫速度較快，可以快速隨機存取，適合快速開機載入作業系統時使用。

2. NAND 閘型快閃記憶體(NAND gate flash memory)

使用「NAND 閘」為基本邏輯元件所設計的快閃記憶體，容量較高，價格較低，又稱為「Data storage flash」，一般用來儲存資料量較大的使用者資料庫，例如：行動電話的電話簿、錄音筆的錄音內容、數位相機與數位錄影機的記憶卡等，讀寫速度較慢，不可快速隨機存取，必須依照寫入順序讀取資料，因此比較不適合快速開機載入作業系統時使用，而是應用在大量資料儲存。 NAND 和 NOR 兩種陣列的比較分析:

表 1 NOR/NAND 特性比較

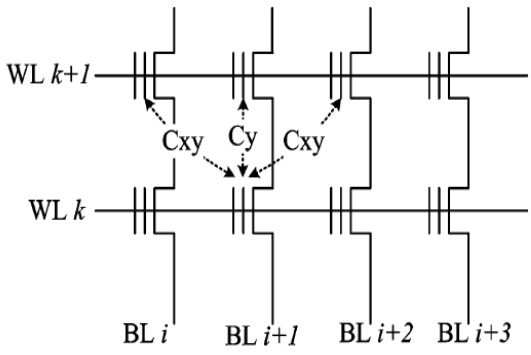
NOR 架構	NAND 架構
優點: 資料可直接讀取/隨機讀取速度快 缺點: 面積大 製作成本高 較慢的寫入和抹除速度	優點: 密度高 較快的寫入和抹除速度 缺點: 隨機讀取速度慢 有浮動閘干擾問題

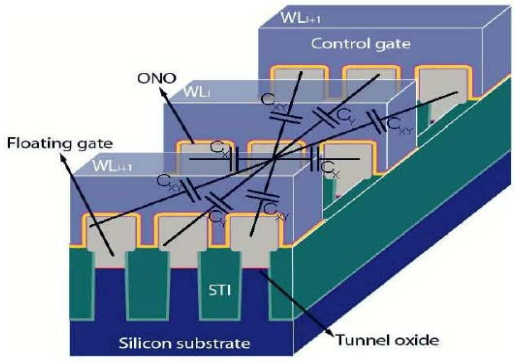
由於在儲存上 NAND 架構因為儲存資料排均共用同一條源極作串接動作，所以當資料以電壓方式寫入時鄰近資料排就會有浮動閘電壓干擾問題，本篇論文架構將以軟式編碼探討在 NAND 架構下的解碼效益。

2.3.2 快閃記憶體浮動閘(floating gate)干擾通道

快閃記憶體儲存架構分為兩種，奇偶位元架構(even/odd bit-line structure)[1]、[25]和所有位元架構(all bit-line structure)[26]、[27]，快閃記憶體讀取方式是一次讀取一頁(page)也就是在儲存裝置上的一條線(word line)，奇偶位元架構在讀寫的差別就是他會分成兩次讀寫一次在偶位元下一次在奇位元，而另外一種所有位元架構就是可以配合高速傳輸架構，所以在通道架構上我們以所有位元為本篇論文架構，接下來在快閃記憶體所有模擬及參數皆以所有位元架構為背景做探討。

臨限電壓(Threshold Voltage)在快閃記憶體中透過耦合電容(couple capacitor)會影響鄰近浮動閘(floating gate)上的電壓造成電壓平移(shift)，導致有概率的會將鄰近所儲存電壓數值會有變動；在快閃記憶體布局，當第 k+1 條線注入資料時, 會耦合第 K 條線所儲存的電荷.。導致資料發生錯誤機率， 此現象被定義為浮動閘干擾或者 cell to cell 干擾。特別是製程這幾年到了奈米階段，以前所忽略的電容效應再也無法被忽略。





閘長度 = 閘寬度 = 閘高度 = 浮動閘高度 = 通道寬度 = 120nm

Tunnel oxide thickness : 7.5nm；ONO thickness : 15.5nm

C_{TUN} : 5.3 × 10^{-17}F，C_{ONO} : 1.2 × 10^{-16}F，C_{FGX} : 1.7 × 10^{-18}F，

C_{FGY} : 7.4 × 10^{-18}F，C_{FGXY} : 3.3 × 10^{-19}F，C_{FGCG} : 5.2 × 10^{-18}F。

圖 15 快閃記憶體 0.12um 製程參數(All-bit Structure)

資料來源[2]

本篇論文將參考文獻[2]、[8]所提供臨界電壓在浮動閘所干擾的參數整理至圖15，此圖以 0.12um 製程參數做為本篇在臨界電壓平移浮動閘干擾通道。浮動閘干擾耦合電壓在此被定義為透過鄰近讀寫所產生的多準位(multi-level per cell，以下均簡稱 MLC)電壓透過相鄰近電容比所產生的干擾，我們在此以(15)、(16)式來表示其干擾通道。

$$\Delta V_{fg} = \frac{C_{FGX}}{C_{TUN} + C_{ONO} + 2C_{FGX} + 2C_{FGY} + 2C_{FGOG}} \times \Delta V_1 \quad (15)$$

ΔV_1:鄰近浮動閘平移電壓。

$$\gamma_{fg}(\square\ \square\ \square\ 干擾電容耦合比例) = \frac{C_{FG}}{C_{TUN} + C_{ONO} + 2C_{FGX} + 2C_{FGY} + 2C_{FGOG}} \quad (16)$$

為了增加儲存密度，在產品規劃上面一般在快閃記憶體中每一筆儲存在 Cell 中的電壓，會有多 MLC 代表各個數位資料，本篇論文通道將以快閃記憶體 2bits/cell

架構做為軟式解碼的研究基礎。

接下來介紹儲存記憶體在快閃記憶體寫入類比電壓進入儲存快閃記憶體位元中基礎架構;增量階躍脈衝程式[11]、[12] (Incremental Step Pulse Program, ISPP)是一個很有效率控制 V_{th} 的一種方式使寫入的儲存電壓值確保大於儲存電壓規格，根據參考資料，在此我們定義本篇論文多層式儲存電壓(MLC)為 1.2V(“11”)、2.55V(“10”)、3.15V(“01”)、3.75V(“00”)。

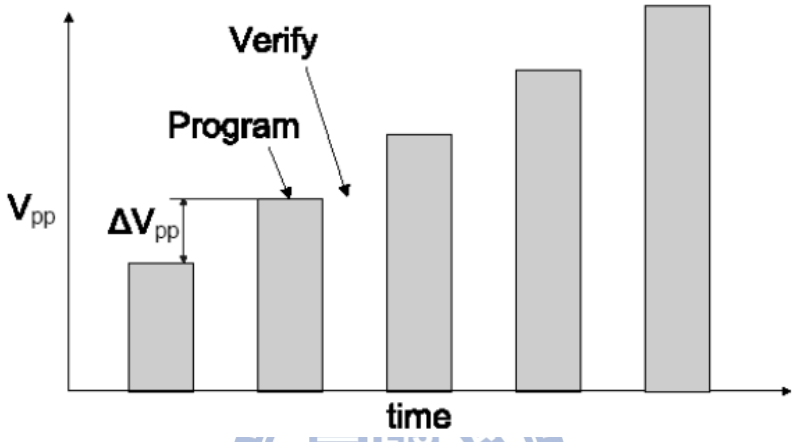


圖 16 控制閘躍脈衝程式

資料來源[13]

透過增量階躍脈衝程式方式，快閃記憶體發生錯誤機率只會在單方面，結合浮動閘電壓干擾或者快閃記憶體位元跟位元電壓干擾，我們透過圖 16 和 15 式可以獲得圖 17 理想和實際在浮動閘干擾底下的分佈圖。

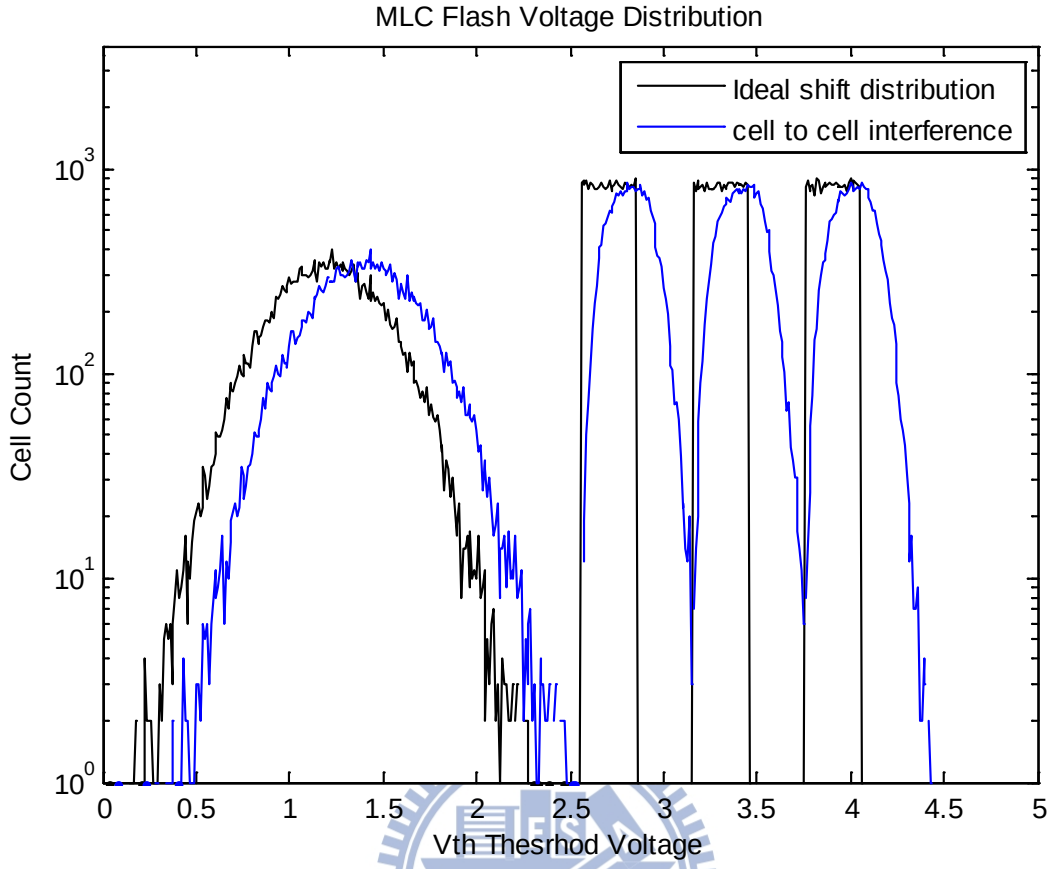


圖 17 控制閘躍脈衝程式與浮動閘干擾臨界電壓分佈

透過上圖 17，在這邊我們分別定義 x_n 和 y_n，x_n 為浮動閘干擾之前訊號，y_n 為浮動閘干擾之後訊號，浮動閘干擾(F)可以被表示成(17)式。

$$F = \sum_{n=0}^{N-1} \gamma_n (y_n - x_n) = \sum_{n=0}^{N-1} \gamma_n y_n - \sum_{n=0}^{N-1} \gamma_n x_n \tag{17}$$

其中 γ_n 為浮動閘干擾電容耦合比例。

接下來快閃記憶體每一個位元儲存電壓單位皆以 V_{th} 表式，透過(6)式把接收信號更改成 V_{th} 的分佈(18)式，其中此分布依然以高斯分佈為主。

$$L(y_i) = \log\left[\frac{p(V_{th} \mid y_i = 1)}{p(V_{th} \mid y_i = 0)}\right] \tag{18}$$

$p^{(k)}(x)$ ，(k)為快閃記憶體儲存電壓(1.2/2.55/3.15/3.75)狀態下的 V_{th} 分佈機率。

我們可以定義透過(18)式，估算所有接受進來的初始信任度 LLR。

$$L(y_i) = \log\left[\frac{\sum_{k\in S_i} p^{(k)}(V_{th})}{\sum_{k} p^{(k)}(V_{th}) - \sum_{k\in S_i} p^{(k)}(V_{th})}\right] \quad (19)$$

皆下來以後所有在快閃記憶體的模擬皆會以(19)式，做為估算 LLR 初始值。

2.3.3 快閃記憶體控制器(Flash Memory Controller)

透過文獻[11]、[14]、[15]，1989 年，東芝公司發表了快閃記憶體結構，強調降低每位元(bit)的成本，更高的性能，並且可以像磁片一樣可以通過介面輕鬆升級。經過 20 年的發展與技術推進。在蘋果電腦和各家大廠大量使用快閃記憶體作為 MP3、iPhone 手機的資料儲存裝置之後，快閃記憶體已經是消費性電子產品在高密度資料儲存裝置裡的理想解決方案。

快閃記憶體結構能提供極高的單位密度，可以達到高存儲密度，並且寫入和擦除的速度也很快。快閃記憶體的應用與 DRAM 不同在於快閃記憶體的管理，需要特殊的系統介面，所以需要一個快閃記憶體控制晶片(NAND Flash controller)。

快閃記憶體的基本存儲單元是頁(Page)，快閃記憶體的頁就類似硬碟的磁區，硬碟的 1 個磁區也為 512 位元組。每頁的有效容量是 512 位元組的倍數。所謂的有效容量是指用於資料存儲的部分，實際上還要加上 13 位元組的校驗資訊，因此我們可以在快閃記憶體廠商的技術資料當中看到「(512+13)Byte」的表示方式。

NAND 是以塊為單位進行擦除操作。快閃記憶體的寫入操作必須在空白區域進行，如果目標區域已經有資料，必須先擦除後寫入，因此擦除操作是快閃記憶體的基本操作。一般每個塊包含 32 個 512 位元組的頁，容量 16KB；而大容量快閃記憶體採用 2KB 頁時，則每個塊包含 64 個頁，容量 128KB。

SLC、MLC 和 TLC 的差別

SLC(Single Level Cell；SLC)即單層式儲存，主要由三星電子(Samsung Electronics)、海力士(Hynix)、美光(Micron)、東芝(toshiba)等公司生產。

SLC 技術特點是在浮置閘極與源極之中的氧化薄膜更薄，在寫入數據時通過對浮置閘極的電荷加電壓，然後透過源極，即可將所儲存的電荷消除，透過這樣的方式，便可儲存 1 個信息單元，這種技術能提供快速的程序編程與讀取，不過此技術受限於矽效率(Silicon efficiency)的問題，必須要由較先進的流程強化技術(Process enhancements)，才能向上提升 SLC 製程技術。

MLC(Multi Level Cell；MLC)即多層式儲存。主要由東芝、瑞薩(Renesas)、三星生產銷售。MLC 是英特爾(Intel)在 1997 年 9 月最先開發成功，其作用是將兩個單位的信息存入 1 個 Floating Gate(Flash Memory 存儲單元中存放電荷的部分)，然後利用不同電位(Level)的電荷，透過儲存的電壓控制精準讀寫。MLC 通過使用大量的電壓等級，每個單元儲存兩位數據，數據密度比較大。SLC 架構是 0 和 1 兩個值，而 MLC 架構可以 1 次儲存 4 個以上的值，因此，MLC 架構可以有比較好的儲存密度。

TLC(Trinary-Level Cell；TLC)，即 3bit/cell，也就是 1 個記憶體儲存單元可存放 3 位元。TLC 速度慢，壽命短，但成本低，價格也較便宜，有 500~1,000 次擦寫壽命。

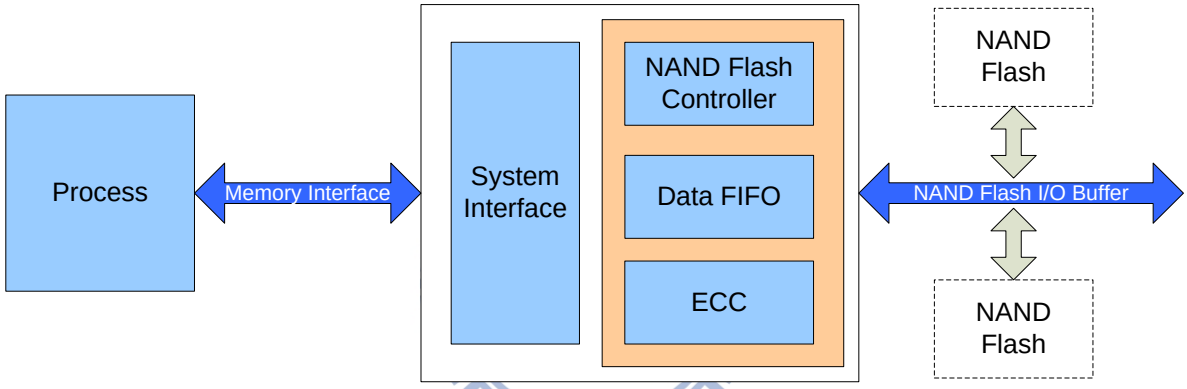


圖 18 快閃記憶體基礎架構

資料來源[16]

接下來將介紹快閃記憶體基本架構和操作，快閃記憶體控制器的操作是透過 8 位元的 I/O 腳位完成，控制命令包含 3 部份: 1. 命令(command)模式 2. 位置(Address)模式 3. 資料(Data)模式。透過 ONF[I1](Open NAND Flash Interface Specification)規格書，其控制訊號被定義如表 2。

表 2 快閃記憶體控制訊號

Symbol		Type	Description
Asynchronous	Source synchronous		
ALE	ALE	Input	Address latch enable
CE_n	CE_n	Input	Chip enable
CLE	CLE	Input	Command latch enable
I/O[7:0]	DQ[7:0]	I/O	Data inputs/outputs
—	DQS	I/O	Data strobe
RE_n	W/R_n	Input	Read enable / (Write / Read_n direction)
WE_n	CLK	Input	Write enable / Clock
WP_n	WP_n	Input	Write protect
R/B_n	R/B_n	Output	Ready / Busy_n

資料來源 Open Nand Flash Interface Specification[5]

控制模式被分成非同步和同步兩種操作模式，傳統快閃記憶體大部分是使用非同步模式，這兩種模式最大不同在於讀寫控制訊號。

1. 非同步(asynchronous)：使用 RE_n 讀取資料，和 WE_n 寫入資料。
2. 同步(synchronous)：使用時序(Clock)訊號做為讀寫資料。

其中此兩種模式控制訊號被定義如表 3 和表 4。

表 3 快閃記憶體非同步模式控制訊號

CE_n	ALE	CLE	WE_n	RE_n	Asynchronous Bus State
1	X	X	X	X	Standby
0	0	0	1	1	Idle
0	0	1	0	1	Command cycle
0	1	0	0	1	Address cycle
0	0	0	0	1	Data input cycle
0	0	0	1	0	Data output cycle
0	1	1	X	X	Undefined

資料來源 Open Nand Flash Interface Specification[5]

表 4 快閃記憶體同步模式控制訊號

CE_n	ALE	CLE	W/R_n	CLK	Source Synchronous Bus State
1	X	X	X	X	Standby
0	0	0	1	Rising edge to rising edge	Idle[1]
0	0	0	0	Rising edge to rising edge	Bus Driving[1]
0	0	1	1	Rising edge to rising edge	Command cycle
0	1	0	1	Rising edge to rising edge	Address cycle
0	1	1	1	Rising edge to rising edge	Data input cycle[2]
0	1	1	0	Rising edge to rising edge	Data output cycle[2]
0	0	1	0	Rising edge to rising edge	Reserved
0	1	0	0	Rising edge to rising edge	Reserved

NOTE:
1. When W/R_n is cleared to '0', the device is driving the DQ bus and DQS signal. When W/R_n is set to '1' then the DQ and DQS signals are not driven by the device.
2. There are two data input/output cycles from the rising edge of CLK to the next rising edge of CLK.

資料來源 Open Nand Flash Interface Specification[5]

三、快閃記憶體 BCH 軟式決策解碼(Soft Decision Decoding)

在快閃記憶體解碼中，目前已經有不少文獻開始探討在浮動閘電壓干擾下使用低密度奇偶檢查碼(Low-density parity-check code，LDPC code)[8]和 Chase[9]軟式解碼在快閃記憶體的表現。由於快閃記憶體在傳輸規格中，有 512bytes / 1024bytes …等等，本篇論文將會選擇快閃記憶體傳輸規格以常用 512 位元組作為評估和分析最適合軟式解碼規格。本章並且透過定點數值(Fix-Point Value)模擬出定訂數位硬體的初始架構。

3.1 解碼演算法有限碼字長度比較

此節分析並選擇常用在快閃記憶體的解碼方式，為了能夠與相關探討低密度奇偶檢查碼在有限長度碼字文獻[17]，本節使用 BPSK 高斯通道做為分析軟式解碼評估。

BCH 和 里德-所羅門(Reed-Solomon, RS Code)一般是目前用的最廣的線性循環碼，里德-所羅門碼廣泛運用在硬碟(Hard-Disk Drive)和光碟(Optical Disk Driver)產品上。BCH 碼是由 Bose，Chaudhur 與 Hocqenghen 三個人獨立發現的錯誤訂正碼，BCH 碼可以更正隨機發生位元錯誤(bit error)的解碼系統，因為快閃記憶體在寫入，擦失，浮動閘干擾發生錯誤皆是位元錯誤，除此之外，在編碼解碼架構上， BCH 碼比里德-所羅門碼在錯誤更正碼中更為簡單，表 5 為 BCH 和里德-所羅門碼編解碼架構比較表。

表 5 BCH 碼和里德-所羅門碼編解碼架構比較表

項目	BCH 碼	里德-所羅門碼
錯誤更正能力	隨機位元錯誤更正	叢錯誤更正
運作單位	二位元	非二位元
硬式編碼架構	多項式線性循環碼(bit)	多項式線性循環碼 GF(2^m)(並行)[20]
硬式解碼架構	徵狀(syndrome)產生器 徵狀位置錯誤多項式	徵狀(syndrome)產生器 徵狀位置錯誤多項式

	秦式(Chien)搜尋法 錯誤位置修正	秦式(Chien)搜尋法 錯誤值修正 錯誤位置修正

所以在快閃記憶體中，目前皆使用 BCH 碼做為錯誤更正碼基礎架構，最主要原因就是快閃記憶體在讀寫時只會隨機發生位元錯誤。

接下來我們就直接探討 BCH 碼和低密度奇偶檢查碼在 AWGN 通道上表現效益。其中在論文題目上，快閃記憶體選擇 512 位元組做為傳輸規格，傳輸碼字長度為(512 + 13 = 525bytes = 1Page)4200 位元，本篇論文快閃記憶體選定傳送方式是以 2bits/cell 為架構，所以選擇碼字近似長度為 2100 單位的低密度奇偶檢查碼的文獻做為選取軟式解碼的依據。

在軟式解碼的分析，我們選擇密度奇偶檢查碼文獻[17]碼長度為 2000 單位，碼率 8/9 做為此解軟式解碼效益比較並且透過圖 19 可以知道在高編碼率的低密度奇偶檢查碼解碼效益式會比在低碼率解碼效率還要差。

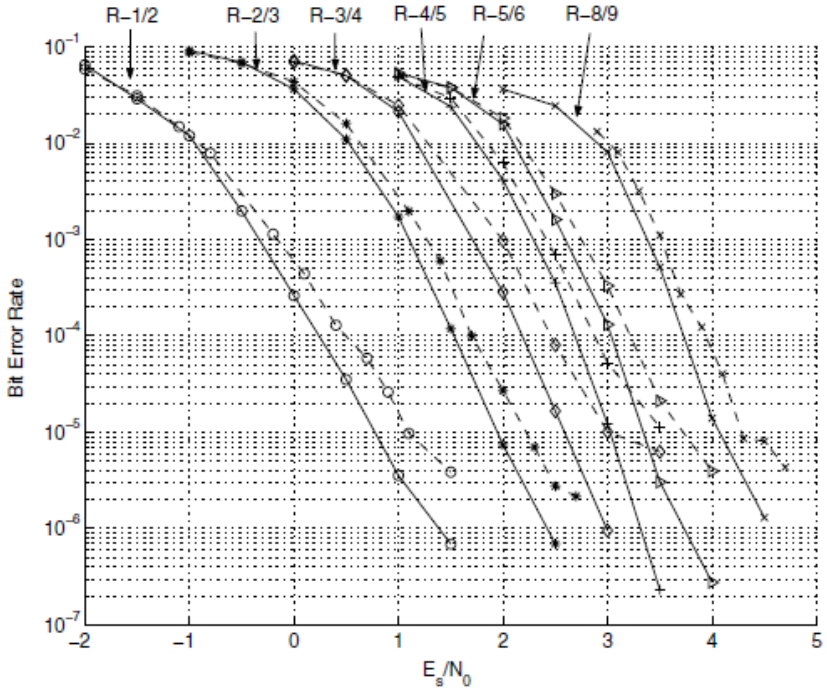


圖 19 低密度奇偶檢查碼不同碼率解碼效益比較圖
資料來源[17]

透過圖 20 模擬結果，BCH 軟式解碼與此篇文獻低密度奇偶檢查碼比較後，

在位元誤差 10^{-5}(Bit-Error Rate)改善 0.3dB。

此分析報告和透過其他幾篇文獻中，低密度奇偶檢查碼字長度要愈大，軟式解碼在整體的解碼效益才會提高，但在快閃記憶體有限長度的條件下，本篇論文決定所使用演算法方向以 BCH 軟式解碼為主軸。

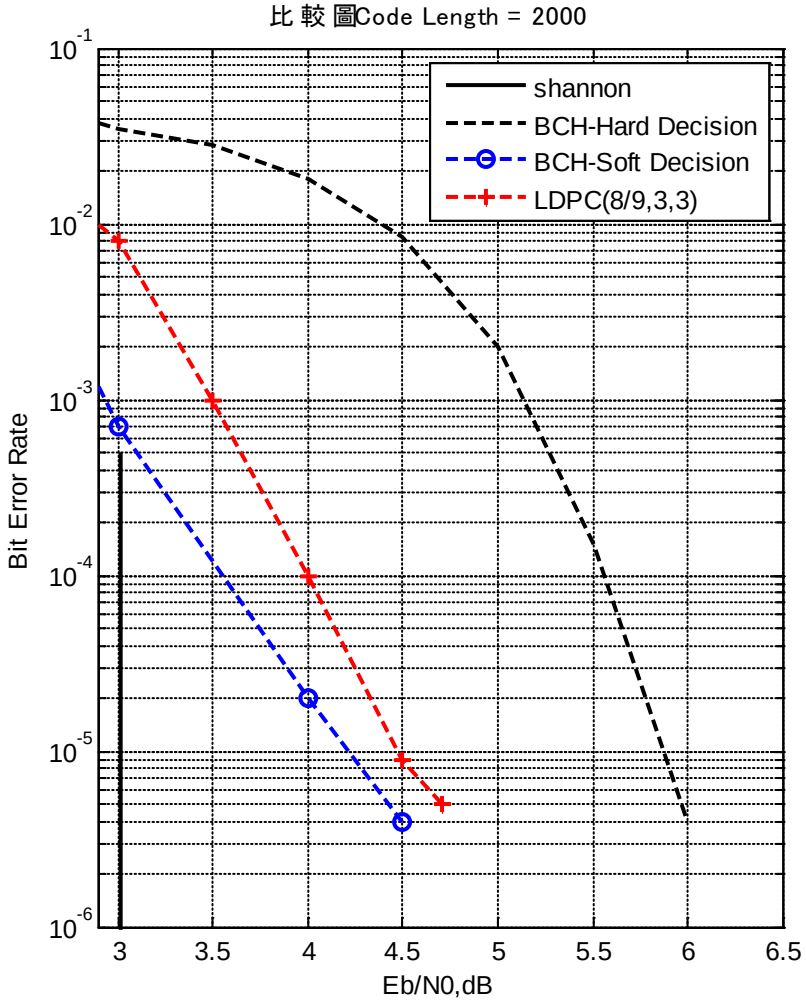


圖 20 BCH 碼和低密度奇偶檢查碼解碼效益比較圖

3.2 快閃記憶體軟式解碼演算法概念和比較

軟式解碼演算法在錯誤更正碼的發展，重點就是透過疊代方式尋找出最大概似度的碼字，在說明 BCH 軟式解碼演算法之前，我想透過之前研究 JN 所提出軟

式編碼的數據說明為何我要使用軟式解碼架構是在梯度下降法在快閃記憶體當中，而不是其他軟式解碼演算法。

表 6 各 JN 應用軟式解碼演算法

軟式解碼演算法	內容
Chase 解碼演算法	產生多組錯誤樣本(Error patterns)與接收信號做加洛法(GF)相加，將產生的多組碼字送進後面硬式解碼找出最有可能碼字的解碼演算法。
OSD 解碼演算法 [18]	利用二位元生成矩陣由信任度大至小的位置進行高斯消去法，並結合 Chase 方式透過生成矩陣產生多組候選碼做比較直接輸出
Outer-round 演算法 [19]	利用多次疊代迴圈不斷改變步階函數(alpha)找出可能碼字比較並做輸出
增加硬式解碼	在軟式解碼後，再做一次硬式解碼做輸出。

為了設計並規劃能夠在硬體能夠實現的軟式解碼演算法，在表 6 所評估各相關 JN 所探討的演算法，在硬體上都會佔據不少成本以及增加解碼時間，所以對於在硬體上的應用將會以最原始軟式解碼演算法-梯度下降法，找出最有大概似度碼字。

快閃記憶體軟式和積演算法解碼架構與傳統軟式解碼架構碼字發生出錯比較。透過 Log 計算我們可以得知該碼字各個位元節點信任度，經過高斯演算法所得到的校驗矩陣搭配梯度下降法，傳統軟式解碼的信任度值不見得會透過梯度下降法透過訊號傳遞(9)式，因為梯度下降法是參考校驗矩陣每一行(row)為 1 的位置做訊號傳遞，所以正負數不見得在每次疊代都會一樣，導致最後無法解決此點錯誤位置，這也是就是梯度下降法所發散的一個問題，如下圖 21 所示。

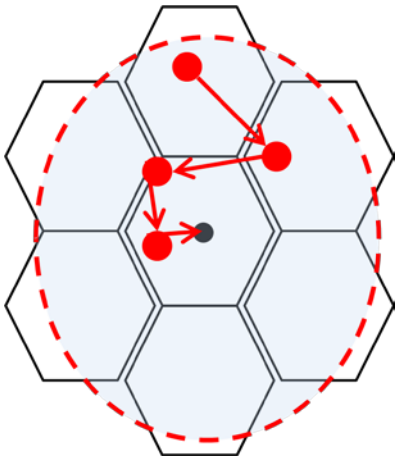


圖 21 傳統梯度下降法示意圖

快閃記憶體因為在寫入的時候會有增量階躍脈衝程式，所以快閃記憶體碼字出錯機率會發生在是電壓平移(shift)過大導致該 MLC 電壓會跑到隔壁較高準位電壓，如下圖 22。在和積演算法中，快閃記憶體因為發生錯誤事件只會在單方面，所以碼字發生在信任度的編緣地區碼字不會像有傳統解碼上的問題，所以在快閃記憶體和積演算法修正如下式(20)。

$$L^{(l+1)} = L^{(l)} + \alpha L_{ext}^{(l)} \quad \text{修正為} \quad L^{(l+1)} = L^{(l)} - \alpha L_{ext}^{(l)} \tag{20}$$

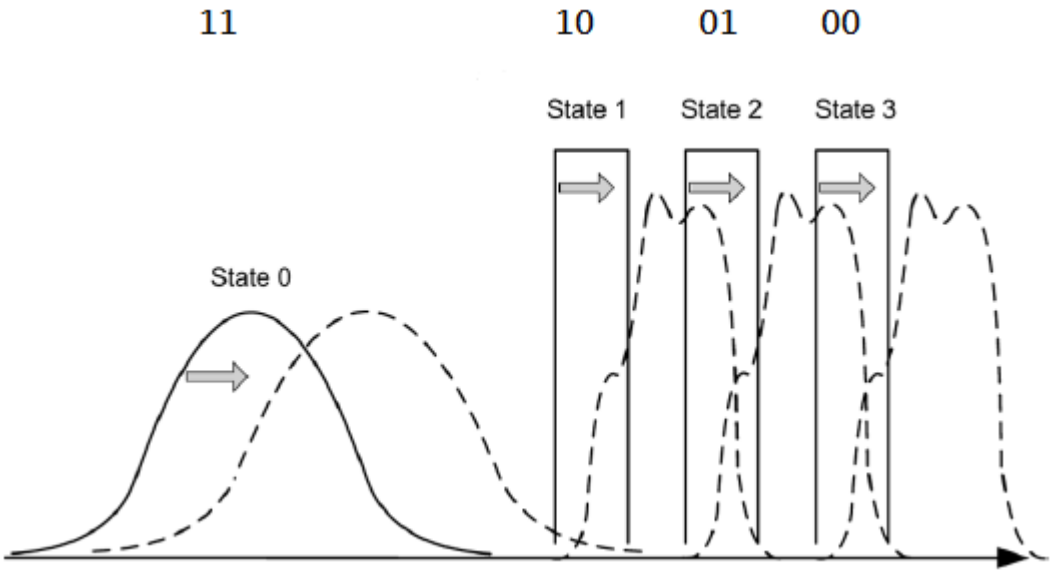


圖 22 快閃記憶體電壓耦合電容平移電壓示意圖
資料來源[8]

為了與作者文獻在通道環境是一樣，圖 23 我們使用相同 BCH 碼字長度與文獻[8]通道做相同碼字率做硬式解碼的驗證，使用(15)式浮動干擾閘和圖 15 0.12um 製程中的電容耦合參數完成浮動閘電壓干擾通道上的 BCH 碼字(32767,31133)上碼字驗證。

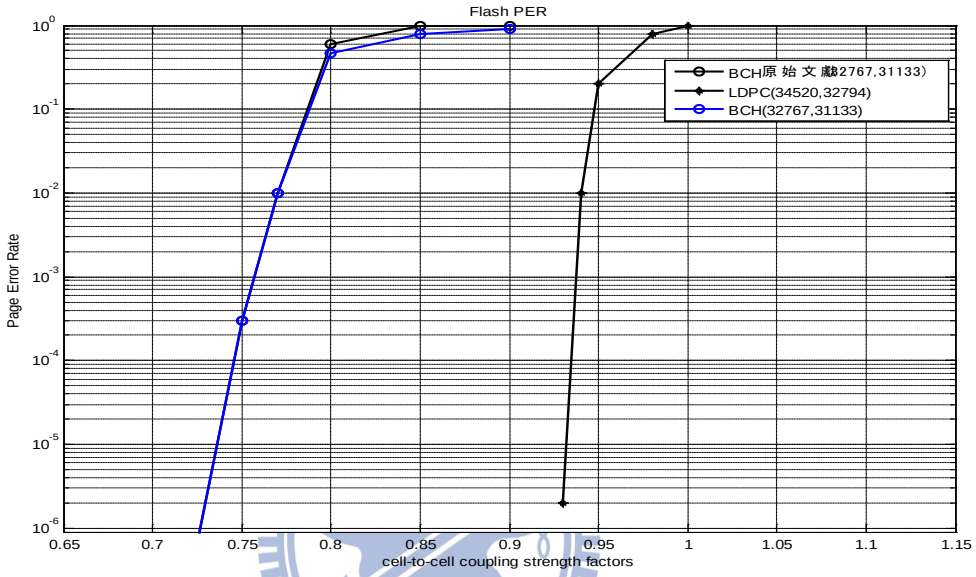


圖 23 快閃記憶體浮動閘電壓干擾通道驗證

3.3 快閃記憶體 BCH-JN 軟式解碼步驟

於本節中，將 BCH-JN 演算法執行的詳細步驟說明如下

步驟1. 設定抑制係數 α，最大疊代次數 $l_{\max}$，並計算每個位元節點初始的 LLR 為 $L^{(0)}(c_i)=(4y_i/N_0)$，並以初始 $LLR(L^{(0)}(c_i))$ 開始進行解碼。

步驟2. 更新奇偶校驗矩陣: $H_b^{(l)}=\phi\left(H_b,\left|L^{(l)}\right|\right)$，$\phi$ 表示更新 H_b 的動作。

A. 依據 LLR 絕對值 $\left|L^{(l)}\right|$ 的大小將每個位元位置進行排序，並且記錄排序後的順序。

B. 利用高斯消去法，按照信任度由小至大的位置將 $H_b^{(l)}$ 消為簡化梯式矩陣，並得到(n-k) × (n-k) 單位矩陣即為低信任度矩陣。

步驟3. 利用高斯消去法正規後的 $H_b^{(l)}$，進行梯度下降法得到每個位元節點的額外消息量: $L_{ext}^{(l)} = \phi\left(H_b^{(l)}, L^{(l)}\right)$，即(13)式的過程。$\phi$ 表示計算額外消息量之動作。

步驟4. 更新每個位元節點的 LLR: $L^{(l+1)} = L^{(l)} - \alpha L_{ext}^{(l)}$ ，即(20)式的過程。在這裡 0< α <1。

步驟5. 硬式決策: 直接找出小於零的 LLR 並記錄其位置及更改碼字。

步驟6. 終止條件:若所有的校驗節點均被滿足，或當到達最大的預設疊代次數時，則停止疊代並且輸出目前經硬式決策後每個位元的決策值。若未達最大疊代次數，則令 $l = l + 1$，回到步驟 2 進行下一次的疊代。

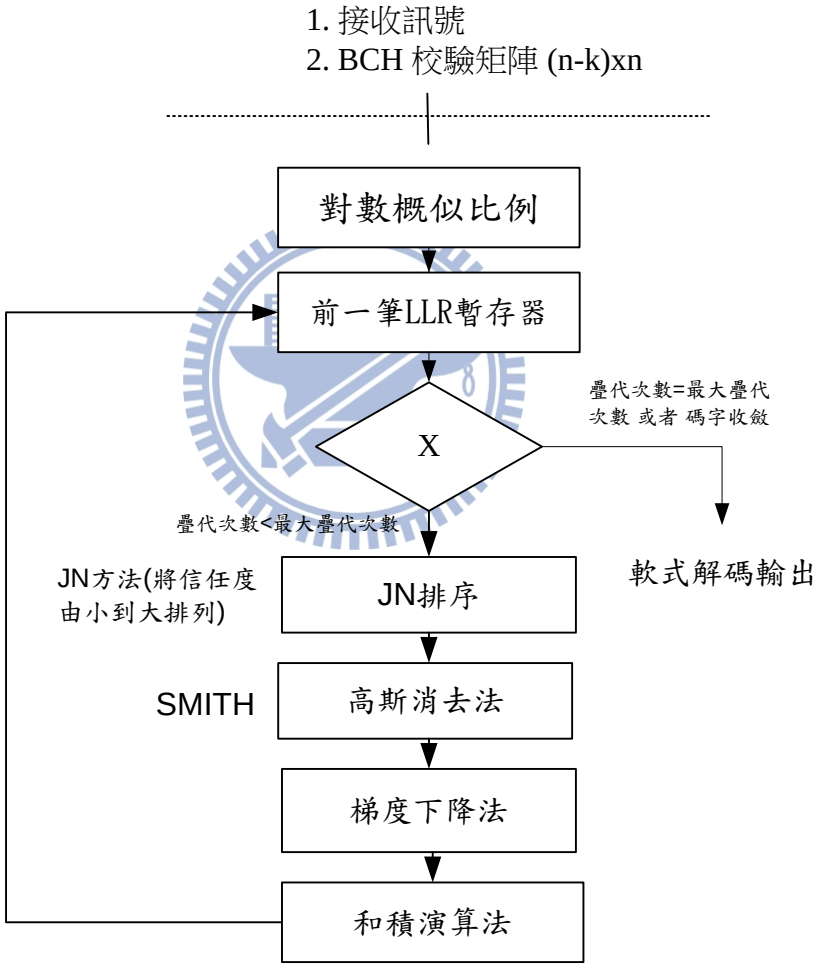


圖 24 BCH-JN 快閃記憶體軟式解碼流程圖

常見快閃記憶體會產生錯誤的資料產生發生在 4 種狀況，如圖 25[21], [22]，說明快閃記憶體常發生錯誤的曲線被統計在不同使用時期的位元錯誤率(Bit Error Rate，簡稱 BER)表現上；

快閃記憶體發生錯誤資料統計分為以下 4 個狀況

1. 記憶體資料保存年限(Retention Errors)。
2. 擦失錯誤(Erase Errors)。
3. 讀取錯誤(Read Errors)。
4. 鄰近資料排干擾(Program Interference Errors)。

本篇論文的模擬環境選擇以 1 年記憶體資料保存年限和 3 千次讀寫次數的 10^{-5} 位元錯誤率，做為模擬環境條件。

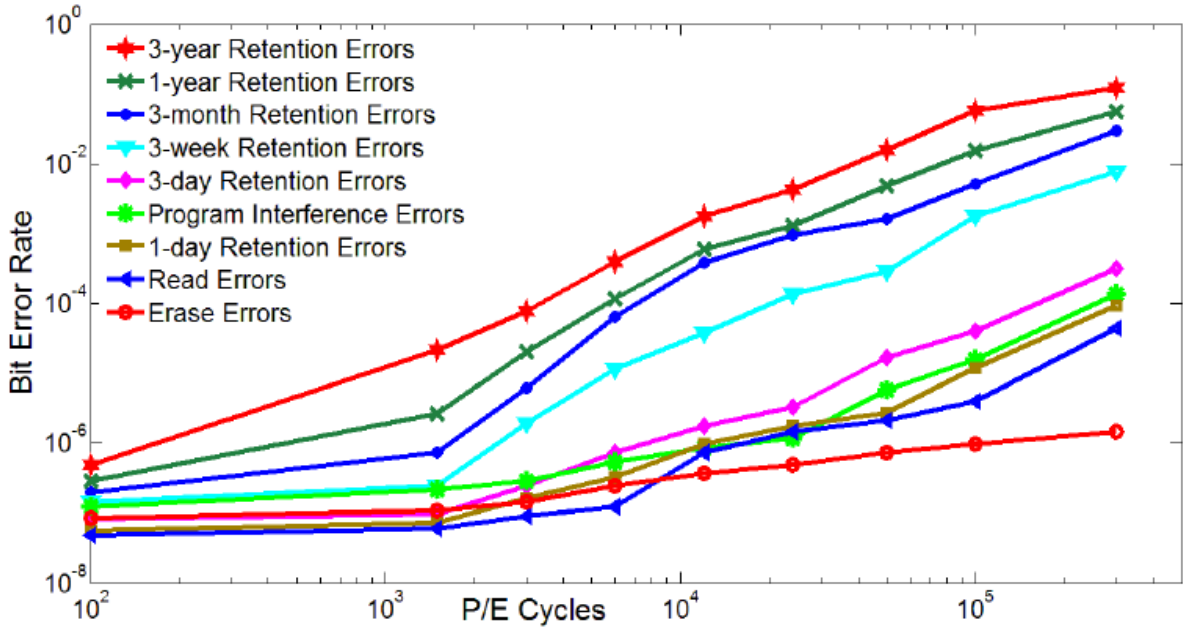


圖 25 快閃記憶體各種錯誤漸進線

資料來源[21]

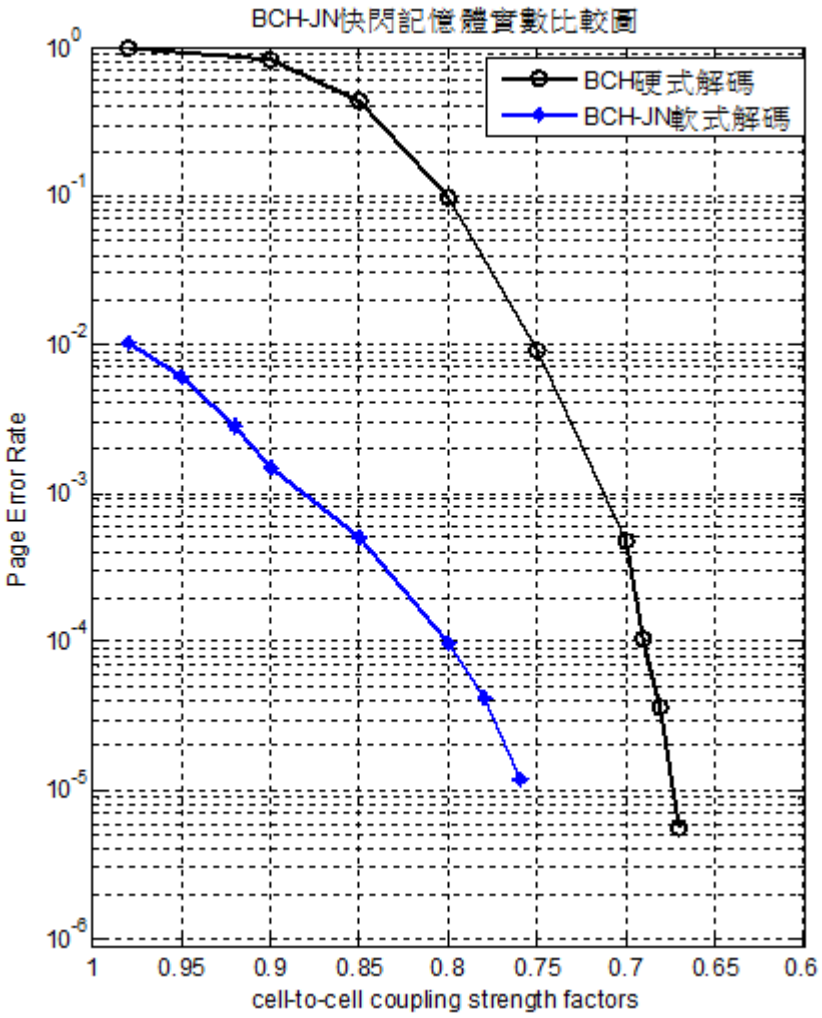


圖 26 BCH-JN 快閃記憶體軟式解碼效益圖

3.4 定點數值(Fix-Point Value)模擬

定點數值表示法是以有限位元的精準度，表現一個固定範圍的浮點數數值，定點數優點就是降低硬體複雜度和尋求以最少成本達到與浮點數接近的解碼效益。定點數通常是以 2 補數(2's complement)格式表式，由符號位元(sign bit)、整數部分(integer part)位元及小數部分(fraction part)位元組成。定點模擬時其字元通常較短，因此必須考慮所造成的有限字元長度效應(Finite World-length Effect)，當有限字元長度效應是浮點數轉到定點格式時，下面兩點是硬體實現關鍵的項目:

1. 溢位(Overflow)誤差 ：超出有限位元表示範圍。
2. 量化(Quantization)誤差 ：有限位元小數部分與浮點數的差異。

使用較多的位元能夠改善溢位及量化誤差，但會造成硬體成本的增加，為了找出適當的位元數但又對於解碼算效益不會影響太大，在這節將直接規劃有限位

元寬度大小並與浮點數比較在解碼演算法的差異。

結合上一節快閃記憶體錯誤機率只發生電壓偏移單方向與傳統高斯通道會有正負符號規格比較，我們只需要知道運算是不是小於零就能知道更改錯誤碼字的位置，所以在目前軟式演算法不增加符號位元在快閃記憶體有限位元模擬。不論疊代或者有限位元規劃寬度需要多少，目的就是要在軟式解碼演算法硬體實現中，減低對浮點數的量化誤差。

針對軟式解碼在有限位元規劃，整理以下影響因素有哪些:

1. 模擬疊代次數。
2. 校驗矩陣寬度與低信任度矩陣和查表法(look-up Table，LUT)關係。
3. 軟式解碼效益。

3.4.1 模擬疊代次數

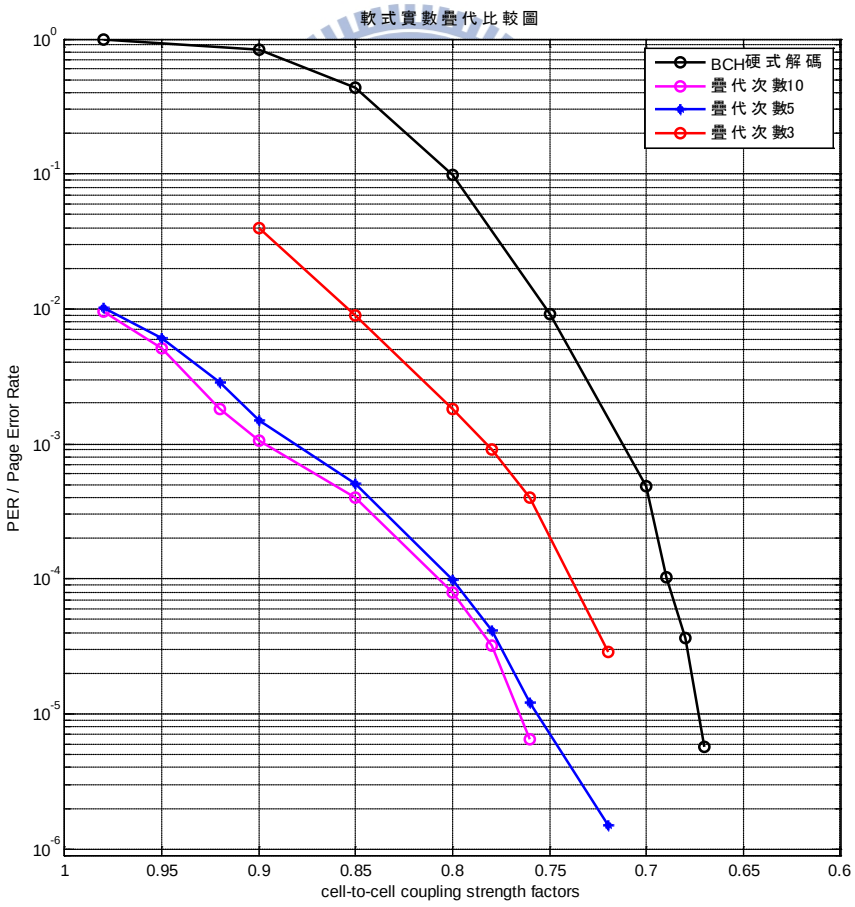


圖 27 BCH-JN 快閃記憶體浮點數疊代次數實驗

初始模擬我們先假定需要 10 次浮點數疊代，在實驗過程中本人在程式中，每次做完程式就會回傳這一次疊代次數需要多少最後將疊代平均次數算出來為 5.2 次，透過此實驗我們選擇 5 次軟式疊代次數，最接近 10 次疊代解碼效益，在圖 26，10^{-5} PER 小於 0.01 係數(factor)，係數指的是浮動閘上的電容耦合電壓強度。

3.4.2 校驗矩陣寬度與低信任度矩陣關係:

圖 28 說明在傳輸快閃記憶體為 512 位元組時，我們實際上收到的信任度大小為 2100 單位長度(2bits/cell)，對應到 BCH 校驗矩陣將會是(N-K)×N，[104 × 2100]。假設定點模擬選定為 10 位元時，所以只會有 1024 種排列矩陣可能性，原始浮點數排序數量是 2100 筆資料，在定點模擬排序數量將會是 1024 筆資料，根據梯度下降法傳遞訊號概念，視為發生同一事件(訊息傳遞的概念，相同信任度可視為發生同一狀況)，所以透過校驗矩陣縮減是不會影響梯度下降法傳遞信任度。

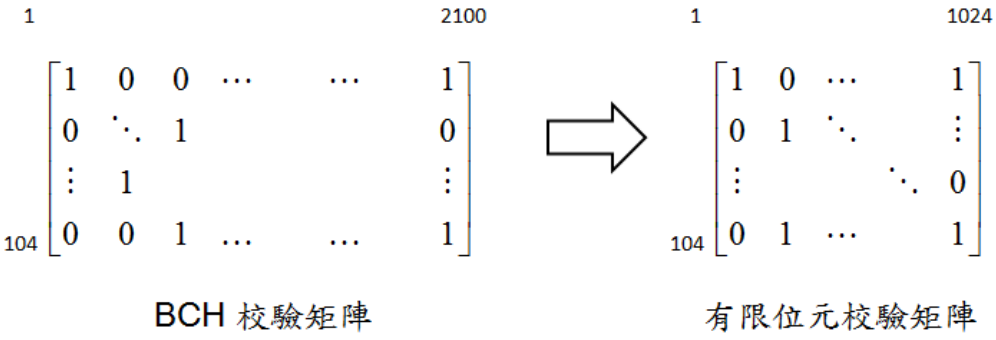


圖 28 BCH-JN 定點模擬校驗矩陣規劃

查表法(Look-Up Table)是一個常在硬體運算中一個轉換數學運算的方式，能夠減少硬體成本和運算時間方式。在 Matlab 模擬中，透過定點模擬範圍在有限位元被假設為 10 位元做規劃，如圖 29。

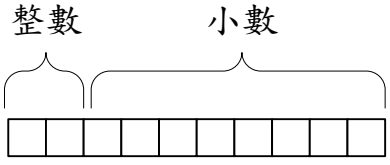


圖 29 定點模擬 10 位元規劃範圍

本篇論文利用快閃記憶體估算信任度大小(19)式，將定點數輸入和輸出做 Log 初始運算時，若輸出有重覆就會忽略該輸入值，最後我們將會獲得多少組輸出完成查表法。其中在定點模擬減少小數位元解析度(resolution)所造成的誤差即被定義為量化誤差，多少量化誤差我們可以接受，在軟式解碼中，必須考慮到傳輸碼字規格和解碼最後效益。在傳輸碼字規格中，512+13 位元組將會有 13×8=104 單位長度的單位矩陣(Identity matrix)，此單位長度在 JN 軟式解碼被定義為最小信任獨立位置 (least reliable independent positions，簡稱 LRIPs)，在此提出單位矩陣寬度與查表法彼此是會互相影響規格大小，在查表法我們假定 10 位元，從接受資料轉換成軟式解碼初始信任度，透過模擬將會有 158 組信任度大小規格，下面表格為信任度轉換表格大小。

表 7 軟式解碼定點數值查表法大小

位元數量	查表法表格大小	可否適用 512 位元組傳輸規格
11 位元	225	是
10 位元	158	是
9 位元	110	否
8 位元	75	否
7 位元	49	否

透過上面表格評估，本篇論文將以較少位元並在和浮點數軟式解碼在解碼效益不會差太多，做為定點數值模擬軟式解碼基礎。圖 30 為我們目前結合定點數值規劃校驗矩陣大小。

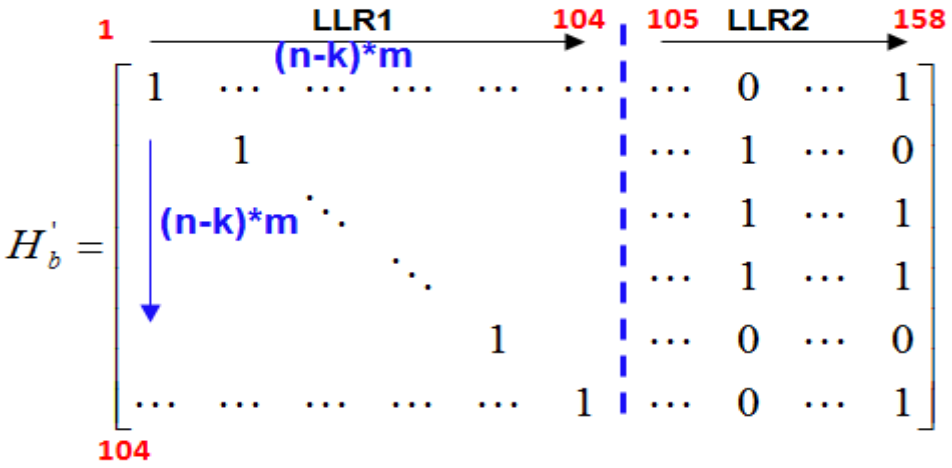


圖 30 定點模擬軟式解碼校驗矩陣規格

圖 30 校驗矩陣在梯度下降法中，必須提前規劃在單位矩陣中的信任度大小是在 0.5 以下(因為透過 tanh 轉換信任度大小將會在 0 到 1 之間)，這部分對於提供每次疊代額外消息量作為下次疊代參考信任度大小很重要，傳遞信任度位置不對容易造成發散。

3.5 定點數值通道模擬和評估浮點數與定點複雜度

在此節為了找出適當的位元數，圖 31 以 10 位元定點數與浮點數軟式解碼比較解碼效益，在 10^{-5} PER 相差為 0.1 factor，所以在定點數上本篇論文選擇 10 位元做為硬體規劃。透過更高位元數模擬根據以前做的相關實驗會越靠近浮點數值，但是相對在硬體的成本會支付相當高，特別在我們即將在第 4 章將會討論高斯電路和梯度下降法在軟式硬體解碼器設計的規格。

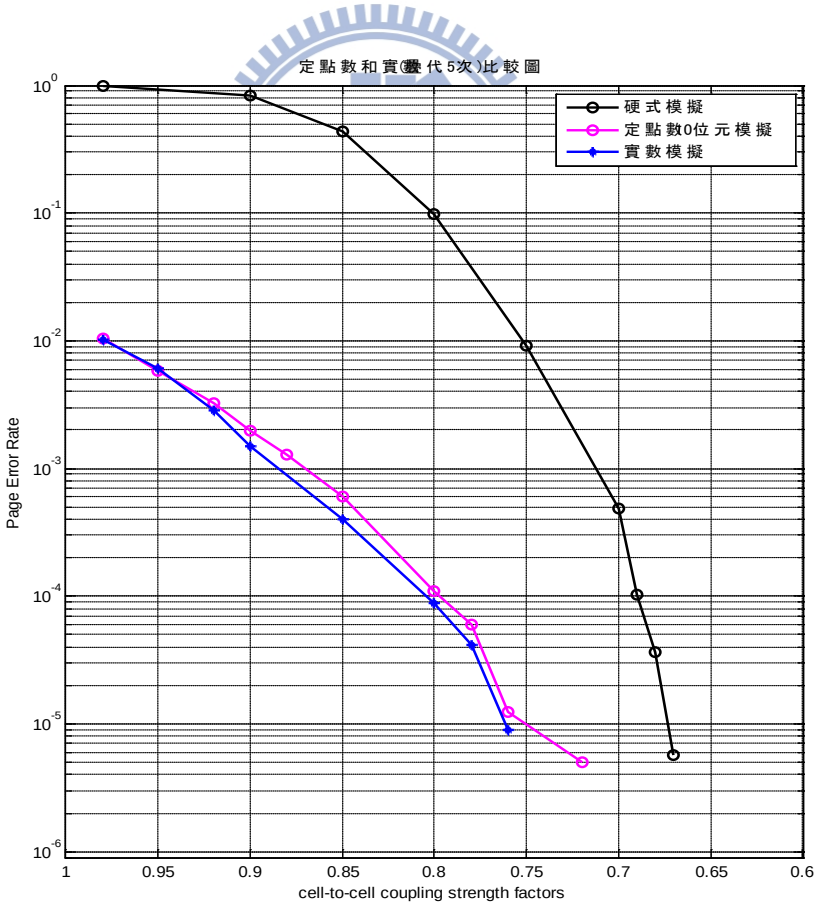


圖 31 定點數值通道模擬

透過 Matlab 執行疊代 5 次的軟式編碼器，我們分別對浮點數軟式解碼器和所規劃的定數數值解碼器分別量測在軟式解碼器執行時，各個區塊(block)方程式所執行的時間，其中以梯度下降法差異度最大，主要在浮點數所規劃的校驗矩陣為 104x2100 單位，與定點數值矩陣 104x158，相差約 12 倍的矩陣大小以及校驗節點個別通知校驗節點的運算時間，分別在梯度下降法量測執行時間為 12.79 秒和 0.031 秒。

另外在高斯校驗矩陣所量測的浮點數和定點數執行時間為 2.45 秒和 0.659 秒。

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
abp_soft_decision_real	1	15.397 s	-0.000 s	
external_llr_caculation_v2	5	12.791 s	11.466 s	
HB_matrix_gaussian	5	2.449 s	0.811 s	
gfadd (MEX-function)	26380	1.591 s	1.591 s	
prod (Builtin-function)	518725	1.295 s	1.295 s	
Sortingllr	5	0.156 s	0.078 s	
sort (Builtin-function)	439	0.047 s	0.047 s	
find (Builtin-function)	13544	0.047 s	0.047 s	
sum (Builtin-function)	25	0.031 s	0.031 s	
randint	259	0.016 s	0.000 s	
length (Builtin-function)	12864	0.016 s	0.016 s	
isempty (Builtin-function)	518	0.016 s	0.016 s	

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
abp_soft_decision_fix	1	0.781 s	0.000 s	
GauGorden	5	0.659 s	0.015 s	
HB_matrix_gaussian	5	0.597 s	0.429 s	
randint	872	0.107 s	0.107 s	
gfadd (MEX-function)	26339	0.107 s	0.107 s	
Sortingllr_real	5	0.092 s	0.061 s	
find (Builtin-function)	14557	0.046 s	0.046 s	
Tanner_real	5	0.031 s	0.015 s	

圖 32 浮點數與定點複雜度比較

四、快閃記憶體控制器硬體實現

4.1 初步硬體規劃

硬體設計時，通常會將系統分為兩個部分：控制單元(control unit)與資料運算單元(datapath)，如圖33 所示。 控制單元產生控制訊號(control signals)用以控制資料運算單元；而資料運算單元產生的狀態訊號(status signals)則會送至控制單元當成部分的輸入訊號。

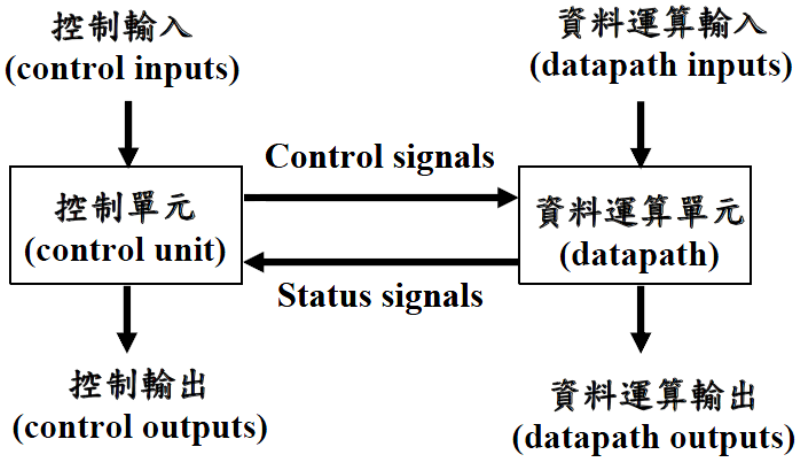


圖 33 數位電路的基本架構
資料來源數位系統講義

控制單元的設計也稱之為有限狀態機(finite state machine，簡寫為FSM)的設計。其中在控制單元分成三個模組：

(a) 次狀態邏輯：組合電路，根據現在輸入(control inputs)與目前狀態(current state)，產生下一個電路狀態(next state)。

(b) 狀態記憶暫存器：記憶單元，儲存目前的電路狀態(由正反器/暫存器組成，需使用脈波clock來同步)。

(c) 輸出邏輯：組合電路，根據現在輸入與目前狀態，產生輸出控制訊號來控制datapath單元進行所需運算。

資料運算單元(datapath)通常包含:算術邏輯單元(arithmetic logic unit，簡寫為ALU)、加減法器、乘法器、比較器、選擇器、暫存器(register)、記憶體等。

4.2 快閃記憶體硬體架構規劃

圖 34，說明 BCH-JN 軟式解碼器在快閃記憶體規劃如下。

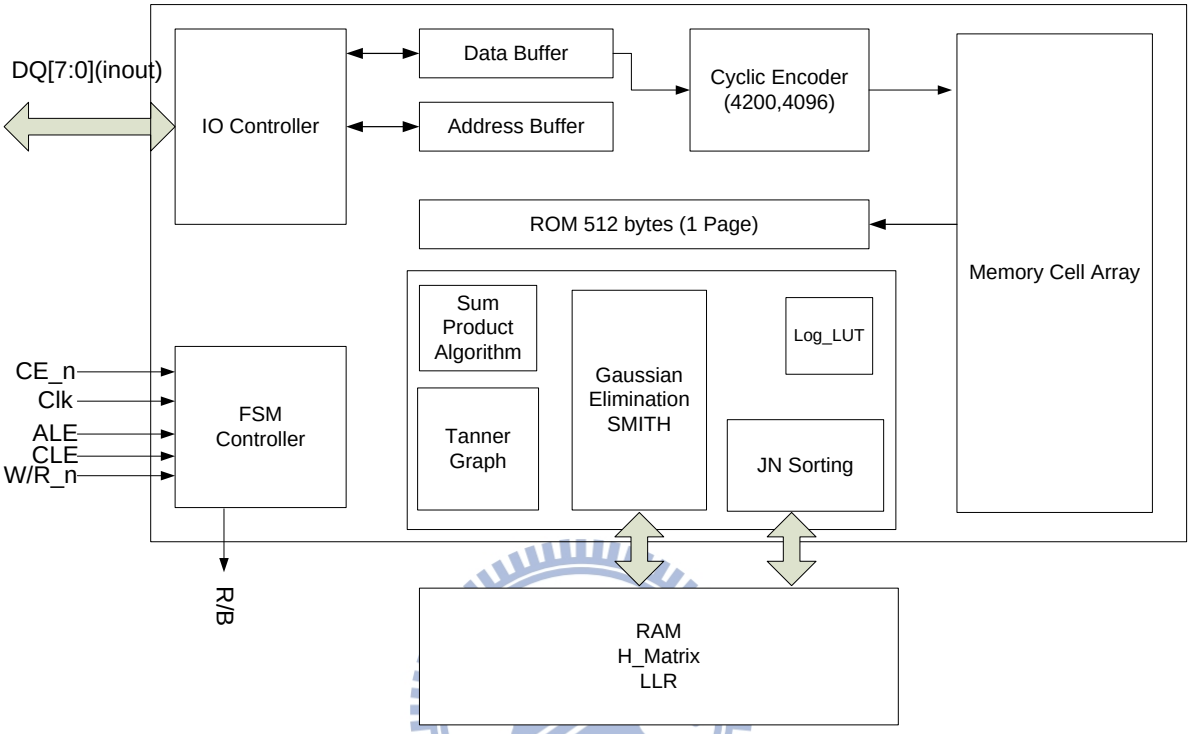


圖 34 快閃記憶體軟式解碼器硬體架構

在快閃記憶體控制器中，傳輸端(Host Interface)會有 3 個控制資料傳送進來分別為命令(command)、位置(Address)和資料(Data)，在命令模式中請參考表 4，傳輸端會透過 CE_n、ALE、CLK、W/R_n 進入 FSM 控制器告訴輸入埠(IO)控制器，輸入埠控制器會根據有限狀態機控制訊號傳送到後資料緩衝器(Data Buffer)和位置緩衝器(Address Buffer)。圖 35 資料緩衝器會直接將資料傳送到後端循環編碼器，循環編碼器採用位元串接平移方式得到校驗碼，使用(21)式生成多項式作為串接編碼器乘法控制開關。

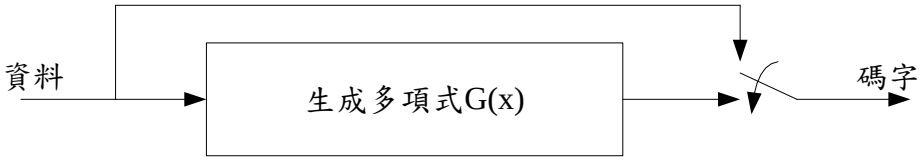


圖 35 編碼器生成多項式

資料來源[1]

$$G(x)=1+x+x^5+x^8+x^9+x^{11}+x^{12}+x^{13}+x^{14}+x^{15}+x^{18}+x^{22}+x^{23}+x^{24}+x^{26}+x^{30}+x^{31}+x^{32}+x^{38}+x^{40}+x^{41}+x^{42}+x^{47}+x^{48}+x^{49}+x^{52}+x^{58}+x^{59}+x^{64}+x^{65}+x^{67}+x^{68}+x^{69}+x^{70}+x^{77}+x^{78}+x^{79}+x^{82}+x^{84}+x^{88}+x^{91}+x^{92}+x^{93}+x^{94}+x^{95}+x^{96}+x^{98}+x^{100}+x^{104} \quad (21)$$

資料原始方式是與生成多項式相除，其中資料是被除多項式，校驗碼就是相除過後所得到的餘數，循環編碼器就是除法器最後產生餘數即為校驗碼(Parity check code)，(21)式為生成多項式，在圖 36 控制每個位元平移過後是否需要做加洛法乘法做開關動作。

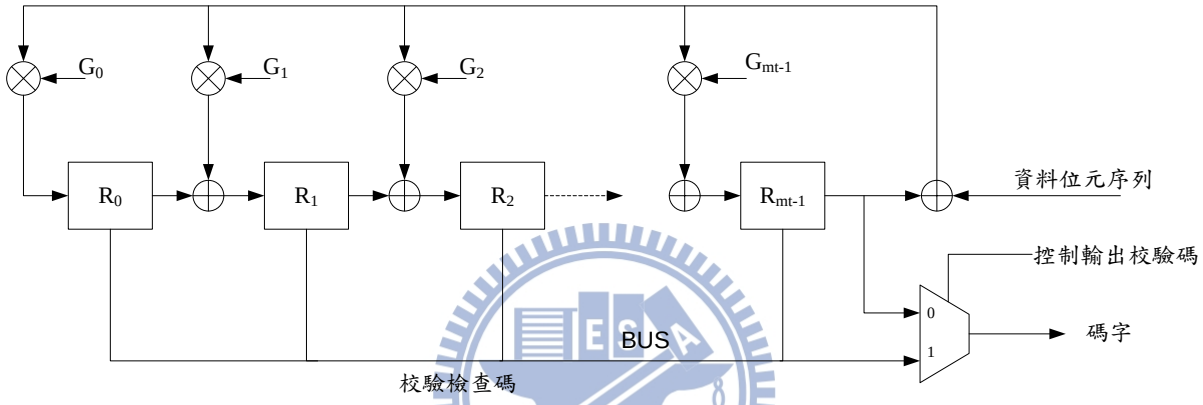


圖 36 循環編碼平移串接架構

4.3 軟式解碼硬體電路

模擬程式裡面所有 Log、Tanh、Atanh 動作將在軟式解碼電路中全部使用查表法取代，在硬體語言中，本篇論文查表法撰寫皆使用 case 做為信任度運算輸出架構。

4.3.1 排序硬體電路

圖 37 為傳統最簡單排序電路架構利用比較器方式找出輸入的最大值或最小值，只能適用於較小的資料比較長度。

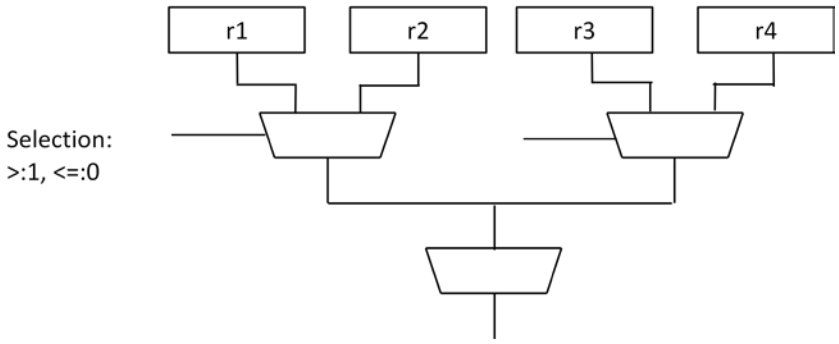


圖 37 排序比較器架構

在文獻[23]介紹利用串接概念，圖 38 將輸入數值用串接概念一筆一筆送進來直接按照大小做排序當最後一筆資料進來時，下一個時序(clock)開始就可以將 SU 排序後的資料輸入至緩衝器(F.I.F.O)輸出至 SU 判斷是否該資料是為最小值然後做輸出，缺點就是暫存器將會多出一倍。

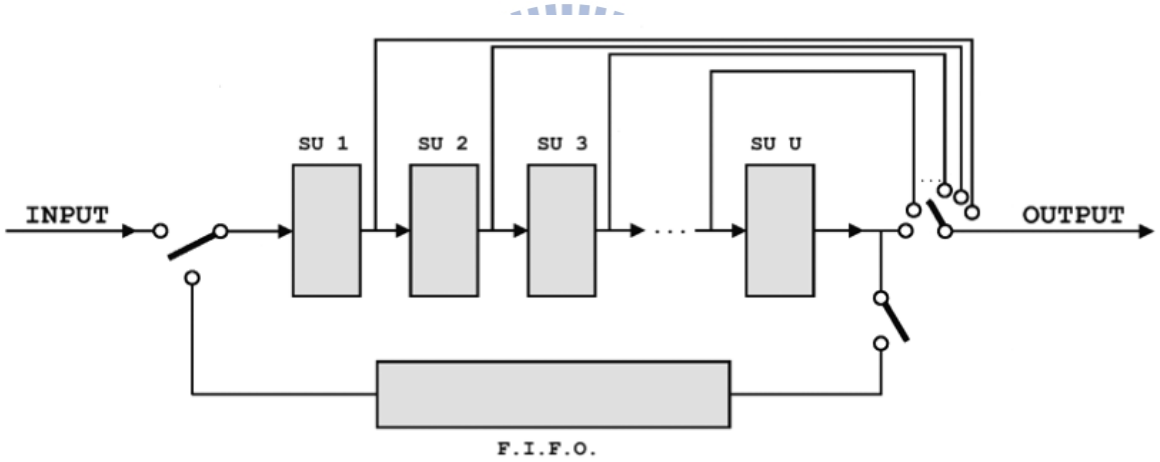


圖 38 排序串接架構
資料來源[23]

上述串接架構在硬體架構選擇較高精準度或者輸入資料的範圍硬體無法掌握排序資料就可以使用串接架構。本論文選擇圖 37 比較器架構是因為只針對 158 筆資料做有限範圍排序，在查表法電路方式直接使用旗標方式對 158 種序列做排序組合，在連續輸入 2100 筆資料中直接輸出到後面的記憶體做位置定位動作，但是在傳輸規格 512 位元組底下因為快閃記憶體讀寫所造成的隨機錯誤透過實驗的統計不會到 104 筆，所以導致有些低信任度的每一個位置不見得都會被選到，因為每一筆排序到位置的碼字會對應到校驗矩陣，在記錄這些位置會將該位置上面的

列向量存入到新的校驗矩陣，此時若沒有位置將會以預設方式的位置提供給下面一節的高斯電路做為初始的校驗矩陣。

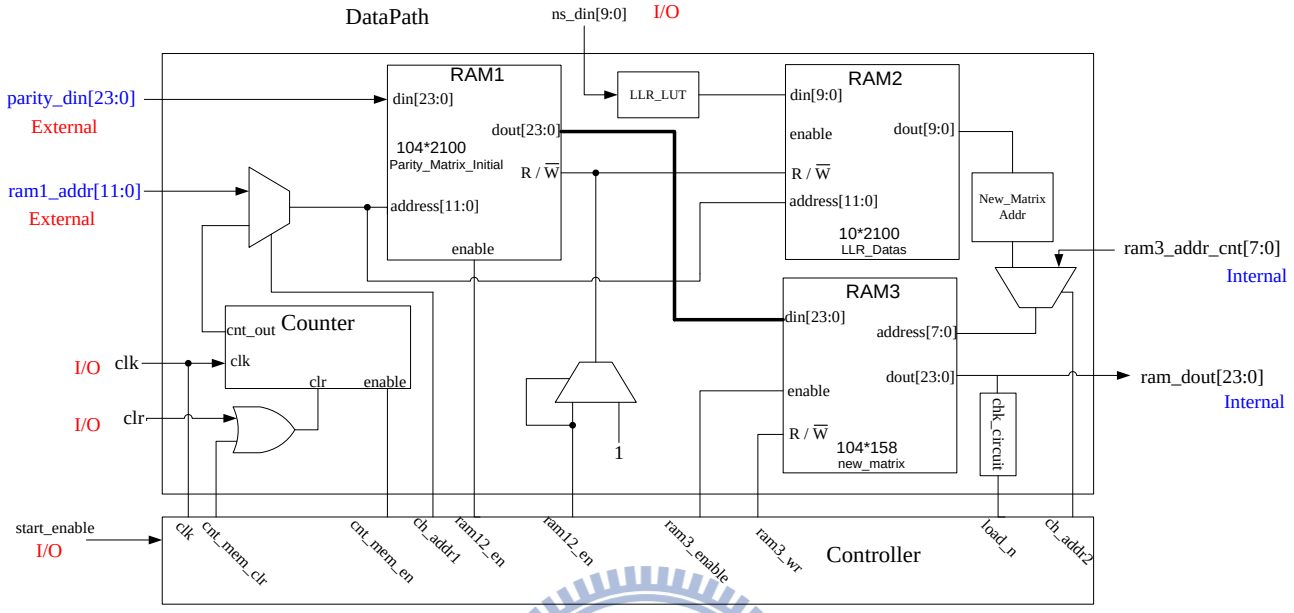


圖 39 硬體排序架構

接下來會開始將會不斷說明硬體架構，其中外部記憶體(SRAM/DRAM)也會被包含到資料運算單元，圖 39 在硬體部分分為兩部分，控制單元和資料運算單元，控制器分別控制 RAM1(初始校驗矩陣架構)、RAM2(儲存初始信任度資料)、RAM3(儲存排序過後新的校驗矩陣)，計數器(Counter)個別控制 RAM1 和 RAM2 讀寫位置，透過 New_Matrix Addr 將 RAM2 信任度資料排序經過 7 位元多工器，RAM1 根據 RAM2 讀取位置將出使校驗子矩陣做輸出，若新的校驗矩陣透過 chk_circuit(圖 40)若為零則沒有重複，就將該子矩陣的位置儲存新的校驗矩陣向量到 RAM3，若不為零就跳過將，重複發生同樣信任度位置的資料視為同一個信任度事件 。

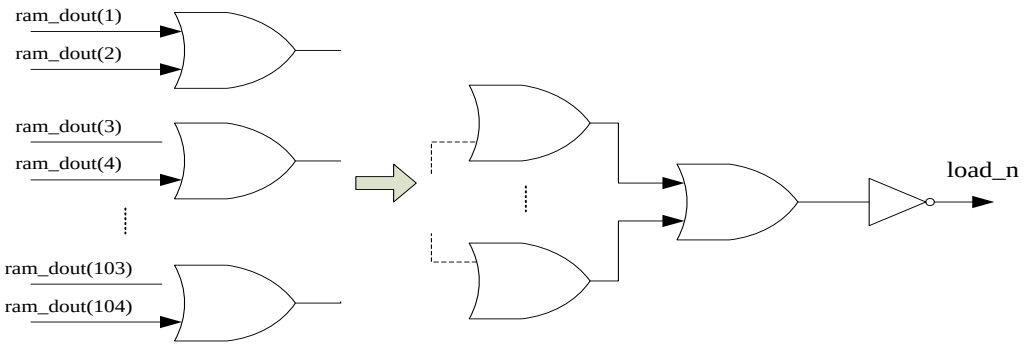


圖 40 RAM3 檢查新校驗矩陣電路

4.3.2 高斯硬體電路

傳統高斯電路使用喬登(Jordan)做法，圖 41 此 jordan 消去法的程序本篇已經稍為作一些修改與原本高斯消去法有些不一樣，原始高斯消去法會先作下三角矩陣消去法再來做上三角消去法，稱為高斯 LU 消去法，我只是將他消去法步驟稍為改一下，將要執行消去法 LU 的動作改為先判斷逐行逐列是否合乎消去法規則，若不是就透過交換方式(swapping)找到合乎高斯消去法規則的行(row)或列(column)，透過行交換，每一列皆還是獨立向量，透過行交換，則要透過暫存器或記憶體記錄該位置，最後才能做消去法動作。

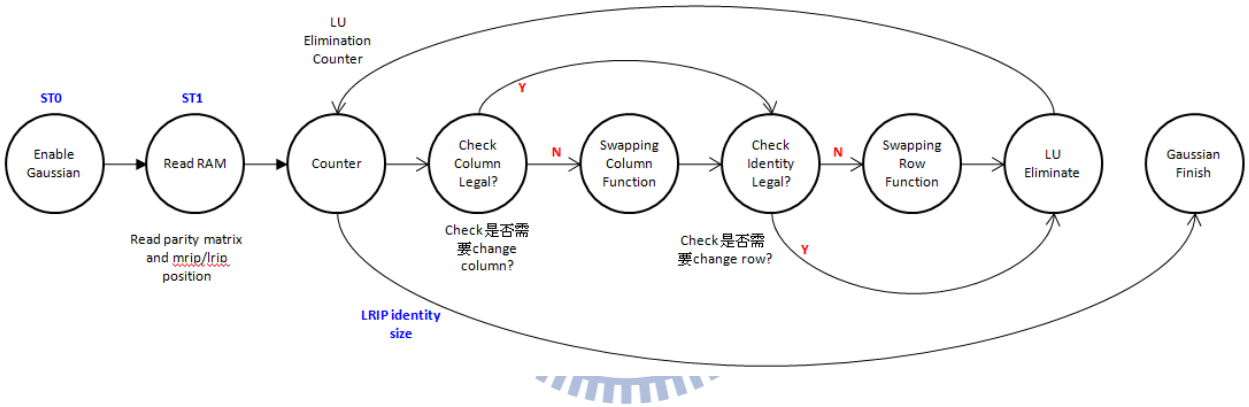


圖 41 Jordan 高斯消去法程序

但是無論怎麼將硬體改變都無法改變 jordan 高斯消去法在硬體的運算時間，這是非常令人詬病的一個問題，碰到消去法運算的時間將會耗時很久，有時候有些矩陣會讓 Jordan 不斷做交換行列動作甚至會在搜尋行或列中，沒有一個符合消去法矩陣的時候，導致硬體運算會卡在這邊，所以在此本篇論文的高斯消去法將會使用史密斯(SMITH)高斯消去法[24]電路，達到對硬體運算時間符合的需求。

史密斯高斯消去法的運算動作，它是由 A. Bogdanov, M.C. Mertens, C. Paar, J. Pelzl, A. Rupp 命名。下面程式敘述是以一個 A 矩陣 nxn 做說明，史密斯是透過該矩陣行和列第一個值 a_{11} 做判斷是否需要做消去法或列平移，圖 42 史密斯消去法與傳統 jordan 高斯消去法節省了至少一個迴圈。

For i = 1 : 單位矩陣寬度

while $a_{11} = 0$

搜尋第一列為 1 的行，將此列透過平移(shift)方式，放進第一行。

end while;

Eliminate(A);

End for 迴圈。

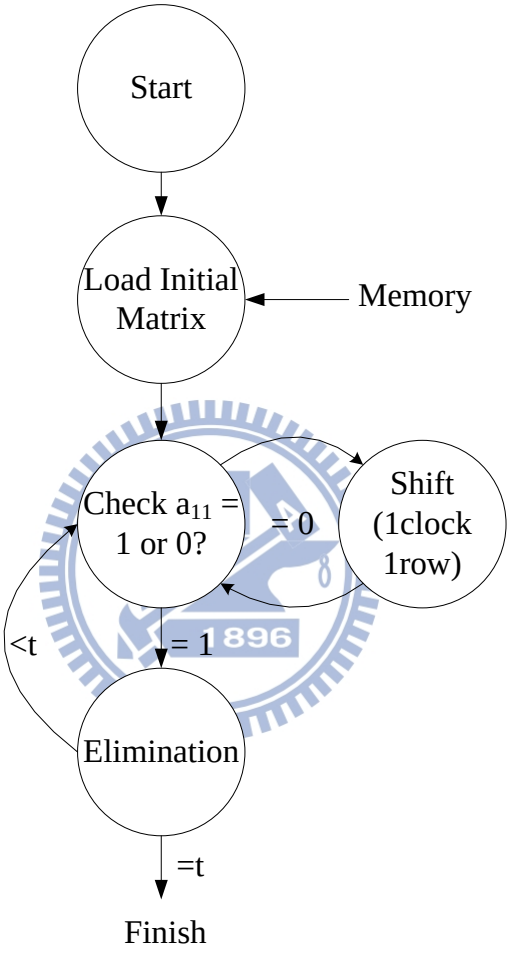


圖 42 史密斯高斯簡易示意圖

$$\begin{pmatrix} a_{22} \oplus (a_{12} \wedge a_{21}) & \cdots & a_{2n} \oplus (a_{1n} \wedge a_{21}) & 0 \\ \vdots & & \vdots & 0 \\ a_{n2} \oplus (a_{12} \wedge a_{n1}) & \cdots & a_{nn} \oplus (a_{1n} \wedge a_{n1}) & \vdots \\ a_{12} & \cdots & a_{1n} & 1 \end{pmatrix}$$

圖 43 史密斯高斯消去動作

一般我們看到圖 43 在解高斯硬體運算法程序上的步驟變得更簡單，有效改善硬體在高斯法所需要時間，但是與傳統 Jordan 高斯消去法還有一點不同就是消去法在史密斯是以全部矩陣直接做做消去動作，這樣會佔硬體相當大空間，在本篇論文會選擇史密斯架構做為軟式解碼演算法的高斯運算基礎架構，原因在於透過定點模擬以及查表法有效並且利用訊息傳遞概念，使得原本高斯運算是在[104x2100]單位長度做運算，現在修正成為[104x158]，另外最主要原因是 jordan 消去法碰到特殊矩陣將無法解決消去動作。

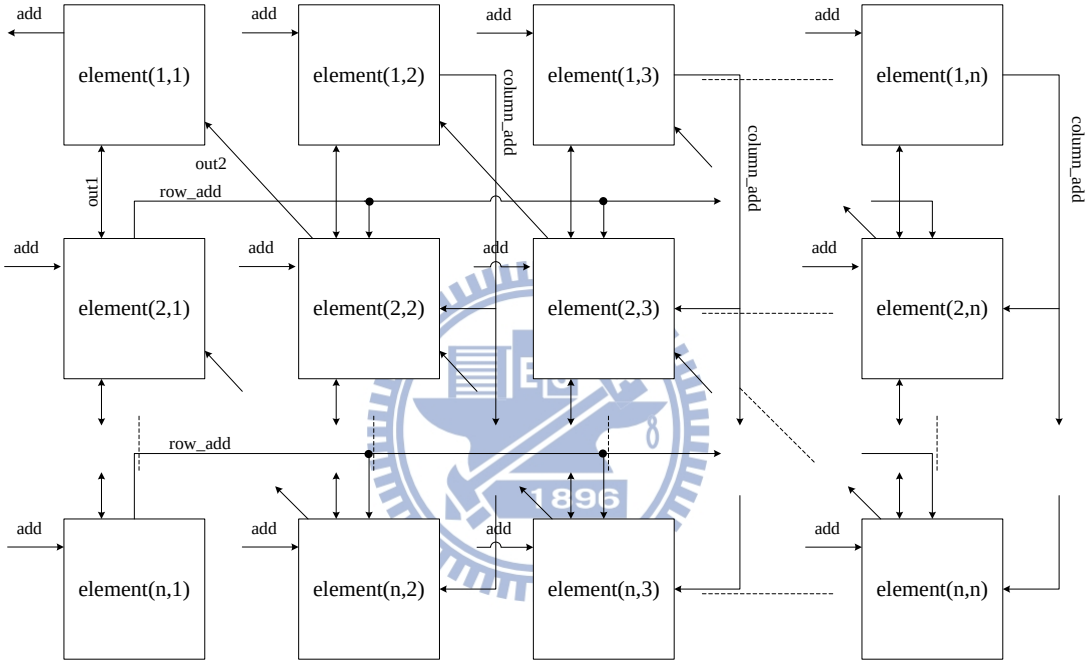


圖 44 史密斯高斯消去法架構

圖 44 史密斯高斯消去法是由[104x158]各個子方矩陣程式所組合而成，每一筆子元素 $e^{(n,n)}$(element)消去資料是由 $e^{(n+1,n+1)}$子元素所得到，消去資料的輸出是由每個元素定義的 out2 做輸出，平移資料是透 $e^{(n+1,n)}$子元素 out1 得到。此架構在數位硬體中被定義為流水線(pipeline)，在硬體中的優點就是處理速度很快，缺點就是因為所有指令都會在同一個時序進行所以相對佔據硬體空間也相當大。

在史密斯高斯消去法是以每一次消去的動作當作計算史密斯所需要的工作時間，當時間等於單位矩陣的寬度就停止硬體動作，在每一個元素裡面都有一個暫存器保存上一次消去法所做的資料，圖 45 就是史密斯消去法子元素的架構。

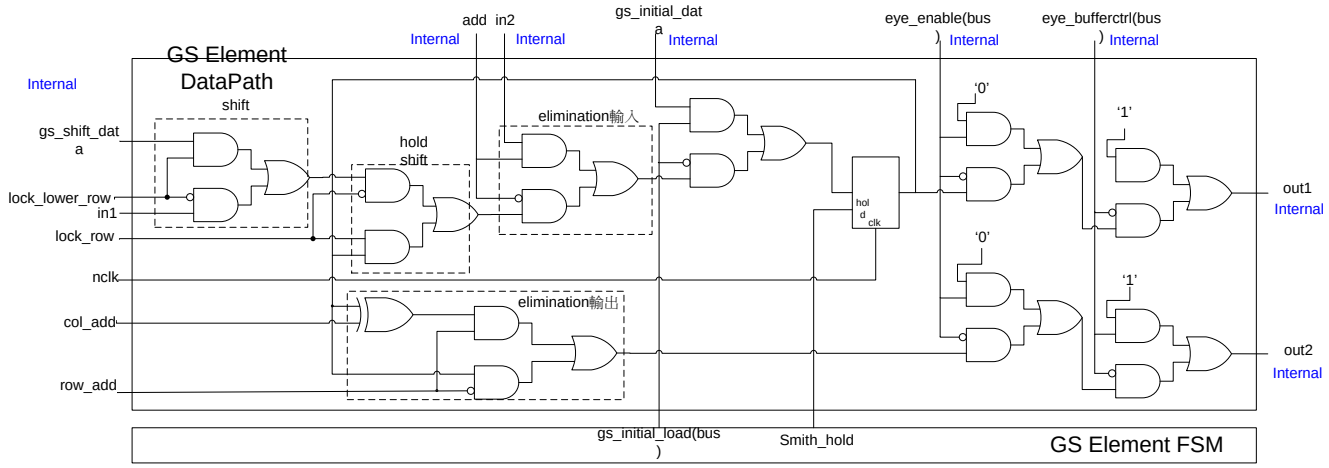


圖 45 史密斯高斯消去法子元素架構

在子元素架構中，有兩條控制線是來自於 FSM 控制器為 gs_initial_load 和 Smith_hold，分別控制初始輸入排序後矩陣值和將史密斯動作完後的值保持住，為了考慮每筆時序硬體皆要有動作避免有閒置發生，在此將資料輸出到後端記憶體做儲存，表 8 說明子元素架構中的資料運算單元的訊號。

表 8 高斯消去法子元素控制訊號

消去法(Elimination)動作	平移(Shift)動作
add = '1' lock_lower_row = add; lock_row = '1';	add = '0'; lock_lower_row = add; lock_row = '0'

圖 46 是史密斯高斯消去電路所規劃狀況，計數器 1 負責提供初始資料下載位置，計數器 2 透過 add 做為平移行和控制每一行的 lock_lower_row、lock_lower 和單位矩陣控制訊號，本論文透過史密斯高斯架構運算時間會從 0.5us 到 13.65us 完成，平移運算時間是最佔時間運算工作，它是透過每一行不斷平移找到合乎條件的規格，才會做消去法動作。但是在有些特定矩陣下，當你做完消去法後，想要找到平移合乎規格的矩陣，會發現找不到，在此我設立一個計數器去計算若超過平移時間，本控制會直接將第一個值 a_{11} 設定為'1'，前提下必須要在該矩陣找不到合乎史密斯所規範的消去規則，我在這邊將會針對該平移做強制設定第一行 a_{11} 設定為'1'。

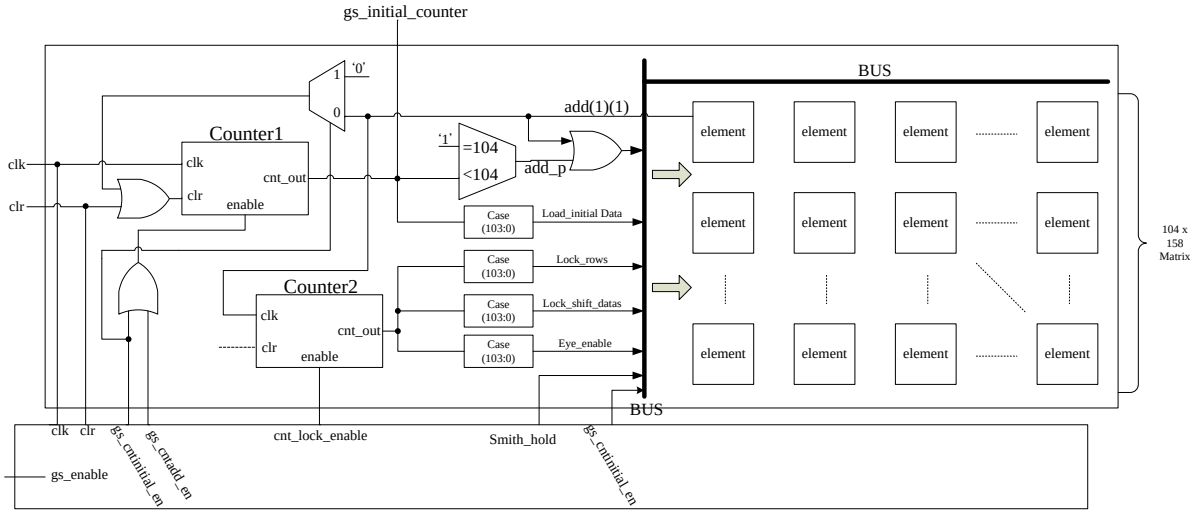


圖 46 史密斯高斯平移架構

4.3.3 梯度下降法硬體電路

在梯度下降演算法中，它是透過該行為 1 的校驗節點，在訊息傳遞上告訴該節點發生錯誤機率是多少，在本篇論文我透過比較器將最後校驗節點所需要知道的信任度傳遞出去，在硬體規劃上，必須由於校驗矩陣是在 158 單位長度，所以我使用並行(parallel)方式將訊號一起做動作，而該行為'1'的節點做為每一筆節點的開關。

透過上述方式，圖 47 在硬體成本上需要 158 個 Tanh 查表法和 314 個比較器、AND 和多工器做組合。

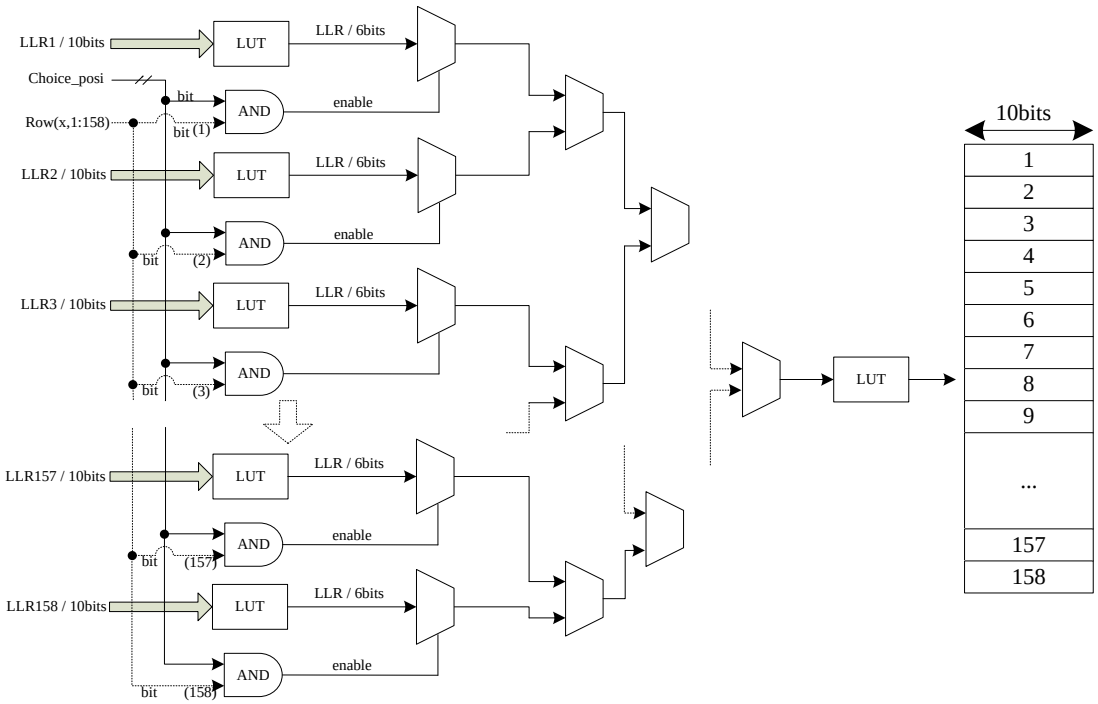


圖 47 梯度下降法硬體架構圖

透過下圖很清楚知道在軟式解碼演算法所需要硬體成本是以 158 單位長度做為梯度下降法解碼單位，透過平移動作尋找節點為 1 的位置，做梯度下降法，但是在高斯解碼演算法後，在單位矩陣只會有 1 個為'1'其他為'0'。

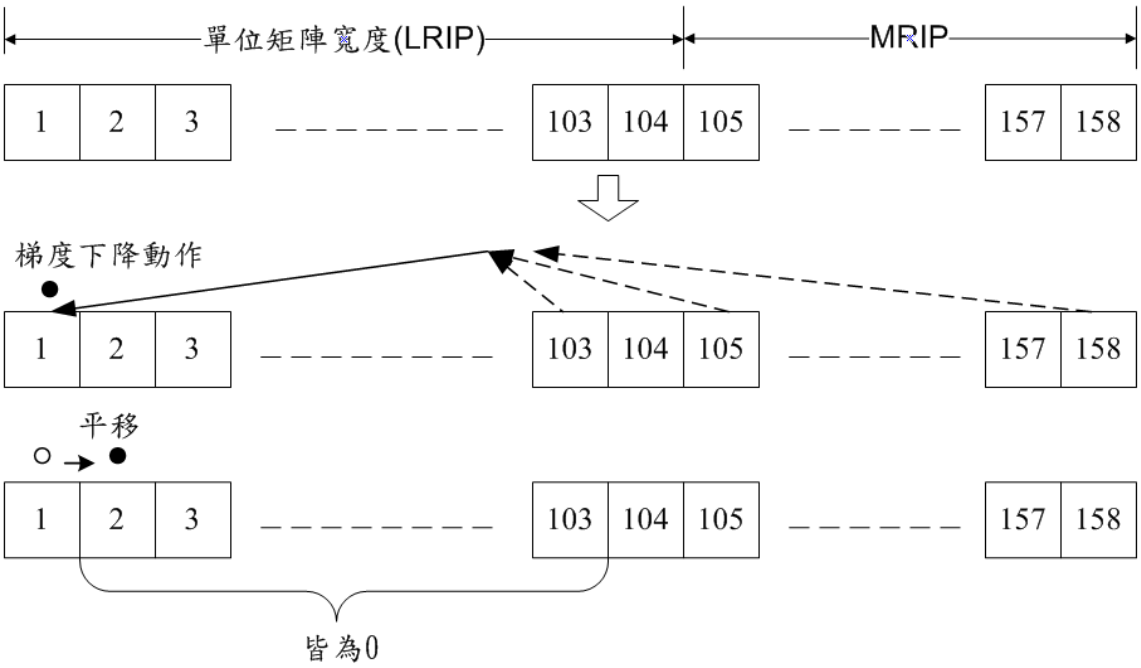


圖 48 梯度下降法硬體原始參考行長度

在單位矩陣中，所提出的校驗信任度只會發生一次，在硬體上本篇論文將會把硬體梯度下降法中的單位矩陣寬度只使用 1 組表示，如圖 49，由於在搜尋下一筆校驗節點本篇論文是以平移架構方式去找符合運作梯度下降法的節點，所以與原始梯度下降法硬體比較將會節省 103 筆查表法和 203 組比較器如表 9。

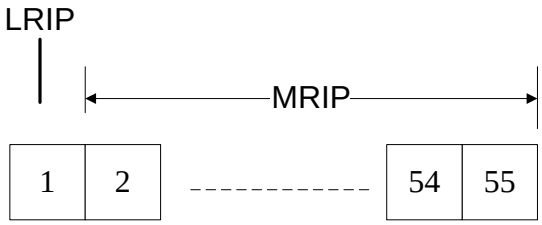


圖 49 梯度下法硬體修改後參考行長度

表 9 梯度下降法硬體改善比較表

傳統梯度下降法硬體表	修改梯度下降法硬體表
LUT:158 Tanh，1 Atanh 比較器:314 AND:158	LUT:55 Tanh，1 Atanh 比較器:111 AND:55

表 10 快閃記憶體梯度下降法硬體改善表

硬體電路區塊(block)	邏輯閘數量	速度	功率消耗
梯度下降電路	89.3K	50MHz	0.78 mW
梯度下降電路(改善)	37K	50MHz	0.46 mW

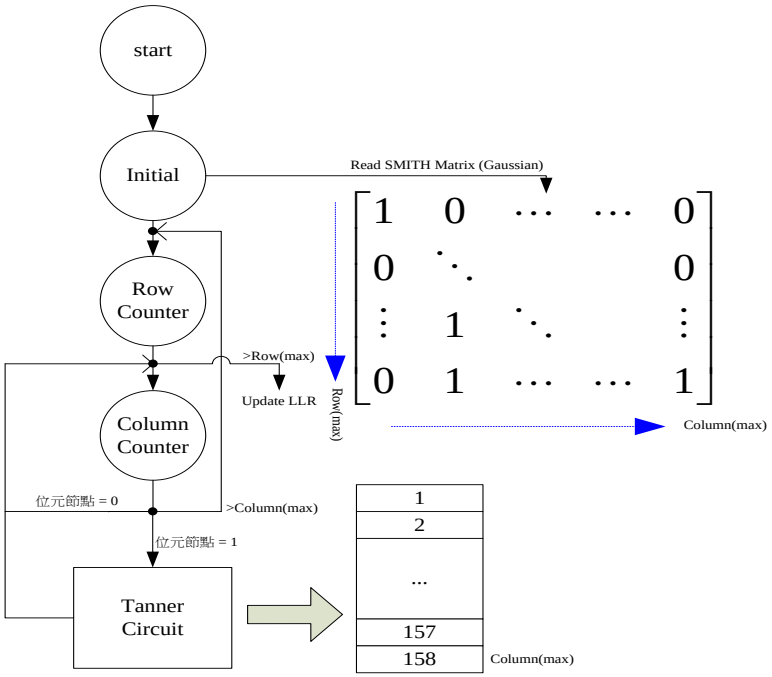


圖 50 梯度下降法硬體動作示意圖

最後得到的訊息將會儲存到記憶體，和積演算法在論文中，也直接在梯度下降法中被包含進去，前面規劃中，我們以 $\alpha=0.125=2^{-3}$ 做為梯度下降法中的階度(step)，階度的廣義就是在每一次尋找最大概似度碼字中的疊代或者影響疊代數值得參數皆被統稱為階度。將梯度下降法所得到的資訊直接左移 3 位元，若在這範圍內有等於或大於"001"都直接進位，這就直接求的額外信任度資料，並且在下一個時序直接將記憶體資料做反向，將此資料與前一筆資料相加即為下一筆疊代的信任度值。

4.4 軟式解碼硬體實驗結果與驗證

軟式編碼和硬式編碼在硬體設計比較起來的優點就是不需要設計乘法器或加法器尋找另一個在有限場(finite field)元素以及提供更好的解碼效益在快閃記憶體在不同強度浮動閘干擾狀況下，但在硬體的成本及運算時間將會付出比硬式解碼器多出很多，表 11 透過兩篇文獻[28]、[29]分別介紹在 LDPC 解碼器與本篇所做硬體比較表，透過此表格，在解碼硬體電路上我們同時以高編碼率做比較，因為在電路上本篇論文電路運算皆是以查表法(look-up table)做計算，所以在功率與頻率比為 4.26 (mW/MHz)比另外兩篇文獻所消耗功率頻率比優於 2.06(mW/MHz)。

表 11 編解碼器功率比較表

錯誤更正碼	LDPC-TDMP [28]	LDPC-TDMP [29]	BCH-JN
碼率(rate)	1/2 ~ 23/24	8/16 ~ 14/16	23/24
碼字長度(bits)	2304	2048	2048
速度(MHz)	100	125	50
高斯 Gate(K)	N/A	N/A	515
解碼器 Gates(K)	463	220	557
消耗功率(mW)	856	787	213
功率頻率比 (mW / MHz)	8.56	6.3	4.26
製程	180nm,1.8V	180nm,1.8V	180nm,1.8V

表 12 為在快閃記憶體(4200,4096)在各個軟式解碼硬體所耗費的功率以及使用邏輯閘數量，以及最後是快閃記憶體控制器所消耗功率。

表 12 快閃記憶體各個軟式解碼硬體功耗和邏輯閘數量

硬體電路區塊(block)	邏輯閘數量	速度	功率消耗
循環編碼器	3.5K	50MHz	0.99mW
排序電路	1.5K	50MHz	0.17mW
ROM 電路	0.88K	50MHz	0.3mW
史密斯高斯電路	730K	50MHz	12.5mW
梯度下降電路	37K	50MHz	0.46 mW
查表法電路	0.3K	50MHz	0.6 mW
快閃記憶體軟式解碼器電路	828K	50MHz	185.28 mW

五、 結論和未來方向

在本論文中，一開始介紹了快閃記憶體在未來製程不斷變小，導致在快閃記憶體在寫入會影響鄰近的資料，讓儲存資料上發生錯誤機率提高成為目前成為在各個快閃記憶體做為軟式錯誤解碼研究背景。接下來說明了梯度下降法的理論基礎、運作方式、JN 演算法並且將快閃記憶體控制的規格之運作背景與詳細步驟。

本論文在快閃記憶體提出的 BCH-JN 軟式解碼適用於傳輸規格在 512 位元組編碼率底下並且以記憶體為 1 年的錯誤率作模擬，JN 軟式解碼透過回授機制會不斷更新低信任度位置，促使在高斯消去法後的低信任度位置能夠在單位矩陣範圍內，能夠提高梯度下降法後額外所獲得較高信任度資料，減少錯誤資料傳遞的情況。在模擬中，在此針對 JN 所提出的其他演算法並且比較，在軟式解碼所額外增加的解碼會對於解碼有改善，但是將會讓軟式解碼的硬體複雜度與成本將會變很龐大。

在硬體中，將史密斯高斯消去的方法應用在硬體實現上，減少傳統 Jordan 高斯消去法運算所需要長時間才能夠完成，另外透過定數模擬我們規範出梯度下降法的校驗矩陣的長度以及在單位矩陣中所模擬的校驗節點排序成為 1 個，節省 Tanh 查表法在硬體的需求。在功率和硬體複雜度目前與 BCH 硬式解碼比較還要高，BCH-JN 軟式解碼在 0.18um 製程下所需要功耗為 14.05mW 與另外一篇文獻所提出 BCH 硬式解碼在 0.25um 所需要功耗為 8.17mW，已經差了快一倍，在硬體複雜度軟式解碼史密斯高斯消去法已經佔去 9 成的面積，但運算時間卻也因為疊代和梯度下降法需求運算時間會很多。

BCH-JN 軟式解碼與 LDPC 在未來上在快閃記憶體的應用，其實都是以梯度下降法背景動作這兩個軟式解碼，同樣是越需要簡單的方式減少高斯消去法的硬體複雜度和因為疊代和梯度下降法所需要的運算時間，另外在提高梯度下降法在統計資料的正確性同時並且減少低信任度位置錯誤的影響，是未來在需要軟式解碼上可以思考的課題。

參考文獻

[1]. Ki-Tae Park, Myounggon Kang, Doogon Kim, Soon-Wook Hwang, Byung Yong Choi, Yeong-Taek Lee, Changhyun Kim, Senior Member, IEEE, and Kinam Kim., A Zeroing Cell-to-Cell Interference Page Architecture With Temporary LSB Storing and Parallel MSB Program Scheme for MLC NAND Flash Memories, IEEE Journal of solid-state circuits, vol. 43, no. 4, April 2008.

[2]. Jae-Duk Lee, Member, Sung-Hoi Hur, and Jung-Dal Choi., "Effects of Floating-Gate Interference on NAND Flash Memory Cell Operation, IEEE Electron Device Letters, vol. 23, no. 5, May 2002.

[3]. Shu Lin, D. J. Costell., Error Control Coding : Fundamentals and Applications, 3nd ed. Prentice Hall, 1983.

[4]. J. Jiang and K. R. Narayanan., Iterative Soft-Input-Soft-Output Decoding of Reed-Solomon Codes by Adapting the Parity Check Matrix, in Proc. Int. Symp. Inform. Theory, Chicago, Illinois, , pp.261, Aug.2006.

[5]. Open NAND Flash Interface Specification, Revision 2.3 25-August-2010.

[6]. G. D. Forney, Jr., Generalized minimum distance decoding, IEEE Trans. Inform. Theory, vol.IT-4, pp.38-39, Sep. 1954.

[7]. Chase., A class of algorithm for decoding block codes with channel measurement information, IEEE Trans. Inform. Theory, vol. IT-18, pp.170-182, Jan. 1972.

[8]. Guiqiang Dong, Student Member, IEEE, Ningde Xie, and Tong Zhang, Senior Member, On the Use of Soft-Decision Error Correction Codes in NAND Flash Memory, IEEE Transactions on circuits and system—I: regular papers, vol. 58, no. 2, February 2011.

[9]. Xinmiao Zhang, Jiangli Zhu Yingquan Wu., Efficient One-Pass Chase Soft Decision BCH Decoder for Multi-Level Cell NAND Flash Memory, Circuits and Systems (MWSCAS), 7-10 Aug. 2011.

[10]. NAND Flash 101 Introduction, Micron.

[11]. R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti., Introduction to flash memory, Proceedings of the IEEE, vol. 91, pp. 489–502, Apr. 2003.

[12]. K.-D. Suh et al., A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme, IEEE J. Solid-State Circuits, vol. 30, pp. 1149–1156, Nov. 1995.

[13]. Guiqiang Dong, Student Member, IEEE, Shu Li, and Tong Zhang., Using Data Postcompensation and Predistortion to Tolerate Cell-to-Cell Interference in MLC

NAND Flash Memory, IEEE Transactions on circuits and system-i: regular papers, vol. 57, no. 10, October 2010.

[14].W. T. Huang C. T. Chen Y. S. Chen C. C. Liu, The Nand Type Flash Memory Management Model based on the Block Base, 93 年 8 月 2 日.

[15].Toshiba Semiconductor Co., NAND Flash Memory Data Book 2007, http://www.semicon.toshiba.co.jp/eng/product/memory/selection/.

[16].NAND Flash Controller Data Sheet, QuickLogic, 2008.

[17].Wen-Yen Weng, Aditya Ramamoorthy and Richard D. Wesel., Lowering the Error Floors of Irregular High-Rate LDPC Codes by Graph Conditioning, Vehicular Technology Conference, 2004. VTC2004-Fall. 26-29 Sept. 2004.

[18].M. P. C. Fossorier and S. Lin., Soft-decision decoding of linear block codes based on ordered statistics, IEEE Trans. Information Theory, vol. 41, pp. 1379–1396, Sep. 1995.

[19].J. Jiang and K. R. Narayanan., Iterative Soft Decoding of Reed Solomon Codes, in Proc. Int. Symp. Inform. Theory, Chicago, Illinois, pp.244-246, Jul. 2004.

[20].Aqib. Al Azad, Minhazul. Huq, Iqbalur. Rahman Rokon, Efficient Hardware Implementation of Reed Solomon Encoder and Decoder in FPGA using Verilog, International Conference on Advancements in Electronics and Power Engineering (ICAEPE'2011) Bangkok Dec., 2011.

[21].Yu Cai , Erich F. Haratsch, Onur Mutlu and Ken Mai, Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis, Design, Automation & Test in Europe Conference & Exhibition (DATE), March, 2012.

[22].Hyojin Choi, Wei Liu, and Wonyong Sung, VLSI Implementation of BCH Error Correction for Multilevel Cell NAND Flash Memory, IEEE Transactions on very large scale integration(VLSI) systems, vol. 18, no. 5, May 2010.

[23].Enzo Mumolo, Gabriele Capello and Massimiliano Nolich., VHDL Design of a Scalable VLSI Sorting Device Based on Pipelined Computation, Journal of Computing and Information Technology - 1, 1–14 CIT 12, 2004.

[24].A.Bogdanov, M.C. Mertens, C. Paar, J. Pelzl, A. Rupp., SMITH - A Parallel Hardware Architecture for fast Gaussian Elimination over GF(2), 2006.

[25].K. Takeuchi et al., A 56-nm CMOS 99mm^2 8-Gb multi-level NAND flash memory with 10-MB/s program throughput, IEEE Journal of Solid-State Circuits, vol. 42, pp. 219–232, Jan. 2007.

[26].Y. Li et al., A 16 Gb 3 b/cell NAND flash memory in 56 nm with 8 MB/s write rate, in Proc. of IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2008, pp. 506–632.

[27].R.-A. Cernea et al., A 34 MB/s MLC write throughput 16 Gb NAND with all bit line architecture on 56 nm technology, IEEE Journal of Solid-State Circuits, vol. 44, pp. 186–194, Jan. 2009.

[28].Dan Bao, Bo Xiang, Rui Shen, An Pan, Yun Chen,. Programmable Architecture for Flexi-Mode QC-LDPC Decoder Supporting Wireless LAN/MAN Applications and Beyond, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, VOL. 57, NO. 1, JANUARY 2010.

[29].M. M. Mansour and N. R. Shanbhag,. A 640-Mb/s 2048-bit programmable LDPC decoder chip, IEEE J. Solid-State Circuits, vol. 41, no. 3, pp. 684–698, Mar. 2006.