國　立　交　通　大　學

電信工程學系碩士班

碩士論文

多輸入多輸出正交分頻多工系統之偵測與 Viterbi 解碼:設計及實現

Joint Detection and Viterbi Decoding for MIMO-OFDM Systems – Design and Implementation

研 究 生：李峰宇

指導教授：吳 文 榕　博士

中華民國九十四年七月

多輸入多輸出正交分頻多工系統之偵測與 Viterbi 解碼:設計及實現

Joint Detection and Viterbi Decoding for MIMO-OFDM Systems – Design and Implementation

研 究 生：李峰宇　　Student：Fong-Yu Lee

指導教授：吳文榕 博士　　Advisor：Dr. Wen-Rong Wu

國 立 交 通 大 學

電信工程學系碩士班

碩 士 論 文

A Thesis

Submitted to Department of Communication Engineering

College of Electrical Engineering and Computer Science

National Chiao-Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

In

Communication Engineering

July 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年七月

多輸入多輸出正交分頻多工系統之偵測與 Viterbi 解碼:設計及實現

研究生：李峰宇　　指導教授：吳文榕 教授

國立交通大學電信工程學系碩士班

中文摘要

傳統上，在 MIMO-OFDM 系統中 V-BLAST 及 Viterbi 解碼器是分開考量的。由於 V-BLAST 演算法會在 Viterbi 解碼之前做符元(symbol)決策(decision)，因此高錯誤率便往往導致嚴重錯誤延展(error propagation)問題。在本論文中，我們發展出一個結合傳送器之位元資料流先前排序(stream pre-ordering)、接收器之 V-BLAST 處理及 Viterbi 解碼器的系統來將錯誤率最小化。在本方法中，首先接收器會將能使 V-BLAST 有最大效能(在每個子載波上)的解碼順序回傳給傳送器。傳送器便根據這個順序將位元資料流做先前排序。接收訊號之後，接收器便將接收訊號做反排序的動作再經過 V-BLAST 處理及 Viterbi 解碼。模擬結果也証實了此演算法的確能達到較佳的效能。在 IEEE 802.11n WLAN 系統下，我們也使用了 FPGA 設計流程實現了 MMSE 估測器、軟性反對映(soft-bit demapper)及 Viterbi 解碼器。

Joint Detection and Viterbi Decoding for MIMO-OFDM Systems – Design and Implementation

Student: Fong-Yu Lee Advisor: Dr. Wen-Rong Wu

Institute of Communication Engineering

National Chiao-Tung University

Abstract

Conventionally, the V-BLAST and Viterbi decoding algorithms are considered separately in MIMO-OFDM systems. Since the V-BLAST algorithm makes its own symbol decisions before Viterbi decoding, the performance of the overall system is not satisfactory. If the V-BLAST algorithm can use the decisions from the Viterbi decoder, the performance can be enhanced. In this thesis, we propose a scheme to realize this idea. The distinct feature of the proposed scheme is a pre-ordering technique allowing an optimal combination of V-BLAST processing and Viterbi decoding. Simulations conform that the proposed algorithm significantly outperforms the conventional approach. Using an FPGA design flow, we also implement an IEEE 802.11n back-end receiver including an MMSE estimator, a soft-bit demapper, and a Viterbi decoder.

誌謝

首先我要感謝指導老師吳文榕教授，在研究所求學期間對於論文研究詳盡的指導，使我在論文研究中研究方向都不會偏離正軌。而老師細心、嚴謹的求學態度更使我受益匪淺。同時感謝口試委員李大嵩教授、紀翔峰教授與鐘嘉德教授，對本篇論文提出寶貴意見與建議，使得論文內容更佳充實、完備。

其次，我要感謝謝雨滔學長、陳仁智學長、楊華龍學長、李彥文學長和許兆元學長他們在研究及課業學習上不吝指導及鼓勵，且同時感謝寬頻傳輸與訊號處理實驗室所有同學與學弟妹們的幫忙。再來感謝我女友多年來不斷地鼓勵與支持。最後致上我最深的感謝給我父母，他們給予我精神和經濟上的支持，使我無後顧之憂順利完成研究所的碩士學位。

內容目錄

表目錄

圖目錄

第1章簡介

MIMO 系統，該技術最早是由 Marconi 於 1908 年提出的，它利用多天線來抑制通道衰落。根據收發兩端天線數量，相對於普通的 SISO (Single-Input Single-Output)系統，MIMO 還可以包括 SIMO (Single-Input Multi-ple-Output)系統和 MISO(Multiple-Input Single-Output)系統。原則上，通道容量(capacity)隨著天線數量的增大而線性增大。也就是說可以利用 MIMO 系統倍數地提高無線通道容量，在不增加頻寬和天線發送功率的情況下，頻譜利用率可以倍數地提高。

MIMO 的核心概念為利用多根發射天線與多根接收天線所提供之空間自由度提升傳輸速率與改善通訊品質；它主要有兩種功能形式：一為空間多工(spatial multiplex)，另一為空間分集(spatial diversity)。前者是在發射端利用多根天線傳送不同資料序列，並在接收端利用多根天線的空間自由度將該組資料序列分別解出。經由此一程序，在發射端與接收端之間彷彿形成一組虛擬的平行空間通道，可在同一時間、同一頻段，以同一功率傳送多個資料序列。如此一來，整體系統的有效資料傳輸率便可以在不增加任何通訊資源的前題下提升數倍。而後者是利用利用發射或接收端的多根天線所提供的多重傳輸途徑來對抗通道衰落(fading)的影響；所謂分集意即多重選擇性，它可由多個獨立的傳輸途徑中選擇或組合出衰落現象較輕微的接收訊號，以維持穩定的鏈路品質。空間多工接收機的演算法主要有貝爾實驗室的 BLAST 演算法、ZF 演算法、MMSE 演算法、ML 演算法等[1]-[6]。而空間分集主要代表便是空時區塊編碼(Space-Time Block Coding, STBC)[7]，它於發射端將待傳送之資料符元(data symbol)在空間與時間上作預前編碼，產生適當的冗餘(redundancy)，並在接收端經由簡易的處理將此冗餘轉化為「分集增益」(diversity gain)。

OFDM 的優點歸納如下：1.相較於單載波調變，頻帶上不需額外的保護區間(guard band)，擁有較佳的頻譜使用效率 2.能有效對抗多路徑通道，降低接收端通道等化器之複雜度。3.硬體上可採用制式化之 FFT 與 IFFT 模組實現調變與解

調變，縮短開發時程與降低硬體成本。

雖然 OFDM 有著諸多優點，但其調變系統的複雜度相較於單載波系統仍顯得複雜許多，且系統對於時間與頻率同步誤差非常敏感，一旦存在同步誤差，則接收機將遭遇嚴重之載波間干擾(Inter-Carrier Interference, ICI)，影響系統效能甚鉅。此外，OFDM 發射訊號之峰均功率比(Peak-to-Average Power Ratio, PAPR)較單載波訊號大，不利於功率放大器之運作，也會使得接收訊號失真。

近年來，BICM(bit-interleaved coded modulation)已成為一個有效率的技術而廣泛地應用在無線區域網路系統上。BICM 原先是由 Zehavi 所提出來的[15]，它透過由位元上的交錯(bit-level interleaver)而將編碼和調變分開。事實上在瑞雷快速型衰落(Rayleigh fast fading)通道中，BICM 比起將編碼和調變一起做的傳統系統有更好的碼分集(code diversity)。當通道是頻率選擇衰落(frequency selective fading)時，BICM 將會和 OFDM(orthogonal frequency division multiplexing)結合而使得訊號可以在頻域上獲得分集(diversity)而將通道轉為平衰落(flat fading)。例如 IEEE802.11a 無線區域網路便是在 5-GHZ 的頻帶上採用 BICM-OFDM 的系統。MIMO(multi-input multi-output) OFDM 則是藉由多傳送接收天線及頻率選擇衰落通道而獲得在空間及頻率上的分集。

在本篇論文中，吾人模擬比較現存 MIMO 偵測(detection)方法並提出一個結合 MIMO 偵測及 Viterbi 解碼的 BICM MIMO OFDM 系統。現存的系統中大多以 MMSE 及 VBLAST 為 MIMO 偵測的方法，再經過解碼器後輸出。然而在 VBLAST 中依序干擾消除(SIC)的效能則取決於前一級決策(Decision)的錯誤率，若能結合解碼器勢必能提高效能表現。本論文以 Pre-ordering 及 iterative 二種系統討論如何在 BICM MIMO OFDM 系統中結合 VBLAST 及 Viterbi 解碼器。最後再根據在西元 2005, 3 月由 TGN SYNC 所提出的 802.11n 的草案[8]來設計並以 VHDL 實做出簡化的接收機。值得注意的是，本篇論文中所發展的 Pre-ordering 系統同時間在[17]也有類似的作法。

本篇論文的結構如下: 第二章將簡介 BICM MIMO OFDM 系統，包括此系

統的數學模型建立、主要的 MIMO 偵測演算法(MMSE 及 VBLAST)、BICM 的軟式(soft) de-mapping 和解碼(decoding)和在 TGN SYNC 所提出的 11n 草案中的編碼過程。第三章則說明吾人所提出的系統並將其和其它系統模擬比較。第四章介紹吾人所實現的簡化之 802.11n 接收機，主要分成二部分，一部分為 MIMO 偵測，另一部分則為 Viterbi 解碼器。最後總結在第五章中。

第2章 BICM MIMO OFDM 系統

2.1 BICM MIMO OFDM 系統

我們考慮一個 BICM MIMO OFDM[22]的系統，包括了 N_t 的傳送天線及 N_r 的接收天線，系統如圖 2-1 所示。傳送端包括了迴旋編碼器(FEC encoder)，位元交錯器(π)，將位元對映到 QAM(2^M)訊號的 Mapping 及 OFDM 系統中 IFFT(size N_c)及循環前序(cyclic prefix)的插入。接收端則主要由 OFDM-demodulation 的 FFT 及 CP 去除、符元反對映(demapping)計算位元計量值(bit metrics)、反位元交錯器(π^{-1})及 Viterbi 解碼器所構成。

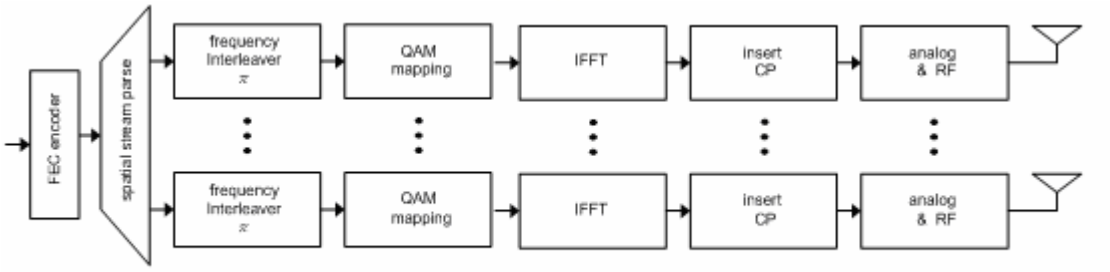

圖 2-1 BICM-MIM-OFDM 傳送端方塊圖

資料位元一開始經過一個編碼率(coding rate)為 R_c 的迴旋編碼器。這些位元再通過一個位元上的(bitwise)交錯器產生新的位元。這些經過編碼且交錯器的位元資料接著會被以大小為 $N_t \times M \times N_c$ 分塊，成為多天線的 OFDM symbol。在這塊的資料中又以大小為 $N_t \times M$ 分群，對應到 $N_t \times 1$ 複數訊號的向量，令此向量為 $S(k,n)$，$k = 1,2,...N_c$。之後對應至相同 OFDM symbol 的訊號便通過 IFFT 及加 CP 方塊。每根多天線的 OFDM symbol 的資料率為 $R_c \times N_t \times M \times N_c$。

我們假設 MIMO 通道為頻率選擇衰落通道(frequency selective channel)，並有

適當長度的 CP 及完美的同步。在頻域上，在第 n 個 OFDM symbol 的第 k 個子載波(subcarrier)中，假設傳送訊號為 $X(k,n)$ ，我們將會收到 $N_r \times 1$ 的複數訊號 $Y(k,n)$ $k = 1,2,...N_c$ 。

$$Y(k,n) = H(k,n)S(k,n) + N(k,n) \tag{2-1}$$

這裡 $N(k,n)$ 是指 AWGN 而 $H(k,n)$ 則是指在第 n 個 OFDM symbol 中的第 k 個子載波中的通道。$H(k,n)$ 的第 (i,j) 項則為從第 i 根傳送天線到第 j 個接收天線的增益(gain)。另外在理想的交錯(interleaving)下，我們可假設 $H(k,n)$ 對不同的 k 和 n 而言是獨立且相等的分佈(iid)的。而為了表示方便，在本論文之後的數學模式中都將 n 省略，也就是(2-1)可表示為(2-2):

$$Y(k) = H(k)S(k) + N(k) \tag{2-2}$$

2.2 MMSE 及 V-BLAST 偵測

在上一節中，我們可以將 BICM MIMO OFDM 系統如(2-2)所示，傳送訊號 $S(k)$ 經過通道 $H(k)$ 後，加上高斯雜訊 $N(k)$ 得到接收訊號 $Y(k)$。在空間多工的應用中， MIMO 偵測(detection)就是要在接收端將其它根天線的訊號消除，從 $Y(k)$ 去推算出 $S(k)$ ，在現行的方法中，大致可以分成三大類的偵測方法，一為最大可能性解碼為基礎的 Sphere decoding[13]-[14]，二為以 Bell lab 所提出來 MMSE 加 SIC 的 VBLAST[4]-[6]，三則是以 QR 分解為主的方法[10]-[12]。底下則介紹在本論文中所採用 MMSE 及 V-BLAST[4]演算法。

2.2.1 最小均方差估測(MMSE)

在收到接收訊號的條件下: $Y(k) = H(k)S(k) + N(k)$ ，我們定義一個錯誤向量 $e(k)$ ，此向量代表傳送的訊號與接收訊號乘上 MMSE 壓抑矩陣後的誤差，也就

是 $e(k)=Y(k)-G^{H}Y(k)$，其中 G 是 MMSE 壓抑矩陣。另外，定義一個成本函數 (Cost Function) J，如下式：

$$J=E\{e^{H}(k)e(k)\}=tr[E\{e(k)e(k)^{H}\}] \tag{2-3}$$

為了使成本最小，我們對 J 微分，使其等於零，並找出此時的 MMSE 壓抑矩陣的值，於是可以推導出 Wiener-Hopf equation：

$$G^{H}R_{YY}=R_{SY} \tag{2-4}$$

其中

$$R_{YY}=E\{Y(k)Y^{H}(k)\}\quad R_{XY}=E\{S(k)Y^{H}(k)\} \tag{2-5}$$

前者為接收訊號向量的協方差矩陣(Covariance Matrix)，後者為傳輸和接收向量的交互相關矩陣(Cross-Correlation Matrix)。然後我們假設:

$$E[SS^{H}]=\frac{1}{\alpha}I_{Nt},\ E[NN^{H}]=I_{Nr},\ E[SN^{H}]=0 \tag{2-6}$$

這裡 $\alpha=\frac{N_t}{\sigma^2}$。換句話說，訊號 S 及外加雜訊 N 在空間上是白色(white)及沒有相關的隨機向量，而變異數分別為 $\frac{1}{\alpha}I_{Nt}$ 及 I_{Nr} (我們可以將 $\frac{1}{\alpha}$ 想像成在每根天線上 SNR)，便可以推導出 MMSE 壓抑矩陣 G 如下：

$$G=[HH^{H}+\alpha I_{N_r\times N_r}]^{-1}H \tag{2-7}$$

2.2.2 V-BLAST

在 V-BLAST 的過程中，偵測的順序對於系統整體的效能影響很大，若第一個解出來的訊號是錯的，在接下來的遞迴過程中，錯誤會延續到第二個訊號上，也就是所謂的錯誤延展(Error Propagation)。因此，有效的排序將可以對系統效能有很

大的提升。因此，我們底下討論如何對訊號排序。首先，我們知道 MMSE 壓抑矩陣可表示成 $G = QH$，且 $Q = [H^H H + \alpha I_{N_t \times N_t}]^{-1}$。當我們計算誤差向量 $e(k)$ 的協方差矩陣時，可得下式：

$$\begin{aligned} R_{ee} &= E\{e(k)e^H(k)\} \\ &= \sigma_n^2 Q \end{aligned} \tag{2-8}$$

觀察(2-8)，擁有 SNR 最高的訊號就是有最小的誤差變異數，所以要解的第一個訊號可以從(2-9)獲得，其中 q_{mm} 代表矩陣 Q 之第 m 個對角線的值。

$$p_1 = \arg\min_m q_{mm} \tag{2-9}$$

而 V-BLAST 的流程主要分為三大步驟，依序如下：

I. 使用(2-7)MMSE 壓抑矩陣，我們可以使用(2-10)計算 $Z(k)$:

$$Z(k) = G^H Y(k) \tag{2-10}$$

II. 利用(2-9)找出 $Z(k)$ 裡面 SNR 最高的子資料符元，並將此符元偵測出來。假設找出在 Z(k) 裡面的第 p_1 個子資料符元擁有最高的 SNR，於是可對第 p_1 個符元作判斷(slice)：$\hat{S}_{p_1}(k) = Q[Z_{p_1}(k)]$ 其中，Q[.] 是根據訊號的星狀圖(Constellation)對符元作判斷。

III. 假設 $\hat{S}_{p_1}(k) = S_{p_1}(k)$，我們就可以把 $S_{p_1}(k)$ 從接收訊號向量 $Y(k)$ 中移除，並獲得一組新的接收向量 $Y_2(k)$，如下所示：

$$\begin{aligned} Y_2(k) &= Y(k) - \hat{S}_{p_1}(k) h_{:p_1} \\ &= \sum_{m \neq p_1} h_{:m} S_m + N(k) \\ &= H_{N_t-1}(k) S_{N_t-1}(k) + N(k) \end{aligned} \tag{2-11}$$

其中 $h_{:p_1}$ 是 $H(k)$ 中的第 p_1 列，而 $H_{N_t-1}(k)$ 是指把 $H(k)$ 的第 p_1 個行移除而獲

得的一個 $N_r \times (N_t - 1)$ 之矩陣。而 $S_{N_t-1}(k)$ 是把 $S(k)$ 的第 p_1 個資料移除，獲得一個長度為 $N_t - 1$ 的向量。

由步驟 I 到 III，每得到一個新的 H 就要重新計算(2-9)以獲得新的排序。經由算出來的 $p_2,...,p_{N_t}$ 順序，並搭配更新後的接收向量 $Y_2(k),...,Y_{N_t}(k)$，如此遞迴的運算直到所有 $S(k)$ 的資料都判斷出來後才停止。

2.3 Soft de-mapping and decoding

2.3.1 軟性反對映(Soft de-mapping)

符元對映(symbol mapping)在 BICM 的系統中扮演著很重要的角色，在非遞迴解碼裡[15]，証明了葛雷碼(Gray-code)對映的最佳性。然而，在遞迴解碼裡葛雷 mapping 便並沒有辦法提供明顯的效果，這是因為在相鄰的 M-QAM 裡，對映位元最多只相差 2 位元而已，如此一來遞迴迴授並沒有辦法明顯地改善 soft-bit metrics。雖然在本論文中有提到使用遞迴的方式來做，但在本論文中還是只考慮使用葛雷對映。在使用葛雷編碼下，如果使用最大可能性解碼時，則反對映(de-mapping)則有明顯簡化的方法[16]，我們將在下面說明。

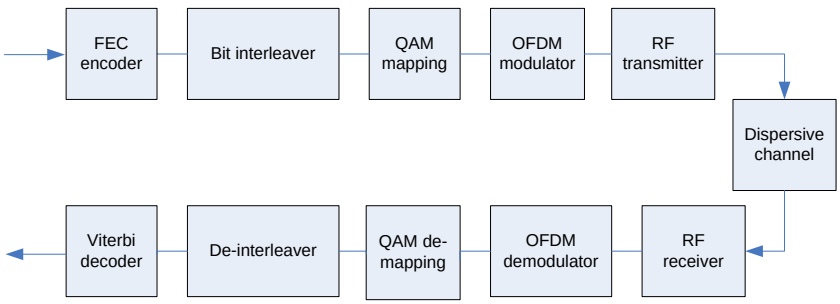

圖 2-2 單天線傳送接收器

所謂的軟性反對映(soft de-mapping)即在決定在每個接收到的 symbol 裡的位

元的機率是 0 或是 1，metric 的值愈大便代表著位元是 1 的機率愈大，反之值愈小的話便代表著位元是 0 的機率大。在說明應用在 MIMO 的情況之前，我們先說明在單天線的情況。

單天線傳送接收器如圖 2-2 所示，在頻域上訊號可以表示為:

$$Z(k) = H(k)S(k) + N(k)$$

$$Y(k) = S(k) + \frac{N(k)}{H(k)} = S(k) + N'(k)$$

(2-12)

$H(k)$ 是通道頻率響應(Channel Frequency Response, CFR)在第 k 個子載波的複數係數，S 是傳送訊號，N 則是指外加雜訊，$\sigma_{N'}^2 = \frac{\sigma_N^2}{\|H[k]\|^2}$。傳送訊號 S 是一個從有限星狀圖(constellation) $\Upsilon = \{S_1, S_2, ...S_{|\Upsilon|}\}$ 取出的 QAM symbol， 由資料位元 $b = [b_0, b_1, ...b_M]$ 所對映而來。位元數 M 則是根據使用的 QAM 大小所決定。在接收到每個 $Z(n)$ 都有 $4M$ 個 metric 需要計算，in-phase 和 quadrature 位元 $b_{I,k}$、$b_{Q,k}$ 各需計算 0 和 1 的可能性，對 $b_{I,k}$ ($b_{Q,k}$ 是同樣的)而言，我們將 QAM 集合 Υ 分成二部分，$S_{I,k}^{(0)}$ 代表著在位置 (I,k) 的位元是 0 的點，也就是在 in-phase(I)位元群中第 k 個位元。相對的，是 1 的點便用 $S_{I,k}^{(1)}$ 表示。如此這二個 metrics 即可由下式表示:

$$m_c(b_{I,k}) = \max_{\alpha \in S_{I,k}^{(c)}} \log p(Z(k) \mid S(k) = \alpha), c = 0,1$$

(2-13)

$Z(k)$ 的機率是個高斯函數

$$p(Z(k) \mid S(k) = \alpha) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left\{ -\frac{1}{2} \frac{|Z(k) - H(k)\alpha|^2}{\sigma^2} \right\}$$

(2-14)

且 $Z(n)=H(n)Y(n)$ ，如此(2-13)即可改寫成

$$m_c(b_{I,k})=\left|H(k)\right|^2 \min_{\alpha\in S_{I,k}^{(c)}}\left|Y(k)-\alpha\right|^2 , c=0,1$$

(2-15)

LLR(log likelihood ratio):

$$L(b_{I,k})\equiv \log\frac{p(b_{I,k}=1\,|\,Z(k))}{p(b_{I,k}=0\,|\,Z(k))} = \log\frac{\sum_{\alpha\in S_{I,k}^{(1)}} p(S(k)=\alpha\,|\,Z(k))}{\sum_{\alpha\in S_{I,k}^{(0)}} p(S(k)=\alpha\,|\,Z(k))}$$

(2-16)

提供了第 (I,k) 的位元到底是 0 或是 1 的可靠度計算方法，由 $L(b_{I,k})$ 的正負號便可決定第 m 個位元是 0 或是 1，而值的大小則表示著可靠度。

當雜訊並非很大時，我們可以將 log-sum 簡化成: $\log\sum_j Z_j \approx \max_j \log Z_j$ ，如此(2-16)可寫成:

$$L(b_{I,k})= \log\frac{\max_{\alpha\in S_{I,k}^{(1)}} p(S(k)=\alpha\,|\,Z(k))}{\max_{\alpha\in S_{I,k}^{(0)}} p(S(k)=\alpha\,|\,Z(k))}$$

(2-17)

將(2-14)代入(2-17)可得:

$$L(b_{I,k})= \frac{\left|H(k)\right|^2}{2\sigma^2}\{\min_{\alpha\in S_{I,k}^{(0)}}\left|Y(k)-\alpha\right|^2-\min_{\alpha\in S_{I,k}^{(1)}}\left|Y(k)-\alpha\right|^2\}$$

(2-18)

在圖 2-3 我們以 16QAM 為例子，為了計算 $L(b_{I,k})$ 而在 $(S_{I,k}^{(1)},S_{I,k}^{(0)})$ 這二個集合裡找離接收訊號最近的點時，我們可以發現在這二個集合裡最近的點都是會在同一條垂直的線上，同樣地為了計算 $L(b_{Q,k})$ 在 $(S_{Q,k}^{(1)},S_{Q,k}^{(0)})$ 裡找最近的點時，這二點便會在同一條水平線上。因此(2-18)可以簡化成:

$$
\begin{aligned}
L(b_{I,k}) &= \frac{|H(k)|^2}{2\sigma^2}\{\min_{\alpha\in S'^{(0)}_{I,k}}|Y(k)-\alpha|^2 - \min_{\alpha\in S'^{(1)}_{I,k}}|Y(k)-\alpha|^2\} \\
&= \frac{|H(k)|^2}{2\sigma^2}\times 4\times\frac{1}{4}\times\{\min_{\alpha\in S'^{(0)}_{I,k}}|Y(k)-\alpha|^2 - \min_{\alpha\in S'^{(1)}_{I,k}}|Y(k)-\alpha|^2\} \\
&= \frac{2|H(k)|^2}{\sigma^2}\times\frac{1}{4}\{\min_{\alpha\in S'^{(0)}_{I,k}}|Y(k)-\alpha|^2 - \min_{\alpha\in S'^{(1)}_{I,k}}|Y(k)-\alpha|^2\} \\
&\equiv CSI\times D_{I,k}
\end{aligned}
\tag{2-19}
$$

式中子集合$S'^{(c)}_{I,k}$定義為:$S'^{(c)}_{I,k}\equiv\Re\{S^{(c)}_{I,k}\}$ ， $c=0,1$。因此在計算$L(b_{Q,k})$便是先由通道和雜訊算出在(2-19)中所需的 CSI(channel state information)，而$D_{I,k}$可利用圖 2-3 而計算得(2-20):

$$
D_{I,1}=\begin{cases}
\frac{1}{4}\left[(Y_I(k)+1)^2-(Y_I(k)-1)^2\right]=Y_I(k), & |Y_I(k)|\le 2\\
\frac{1}{4}\left[(Y_I(k)+1)^2-(Y_I(k)-3)^2\right]=2(Y_I(k)-1), & Y_I(k)<2\\
\frac{1}{4}\left[(Y_I(k)+3)^2-(Y_I(k)-1)^2\right]=2(Y_I(k)+1), & Y_I(k)>-2
\end{cases}
$$

$$
D_{I,2}=\begin{cases}
\frac{1}{4}\left[(Y_I(k)-1)^2-(Y_I(k)-3)^2\right]=-Y_I(k)+2, & Y_I(k)>0\\
\frac{1}{4}\left[(Y_I(k)+3)^2-(Y_I(k)+1)^2\right]=Y_I(k)+2, & Y_I(k)<0
\end{cases}
\tag{2-20}
$$

由(2-20)可整理成(2-21):

$$
\begin{aligned}
D_{I,1} &= \begin{cases} Y_I(k), & |Y_I(k)|\le 2\\ 2(Y_I(k)-1), & Y_I(k)>2\\ 2(Y_I(k)+1), & Y_I(k)<-2\end{cases}\\
D_{I,2} &= -|Y_I(k)|+2
\end{aligned}
\tag{2-21}
$$

我們可以很容易地推出$D_{Q,k}$的式子，只要將(2-21)中的$Y_I(k)$改成$Y_Q(k)$即可。

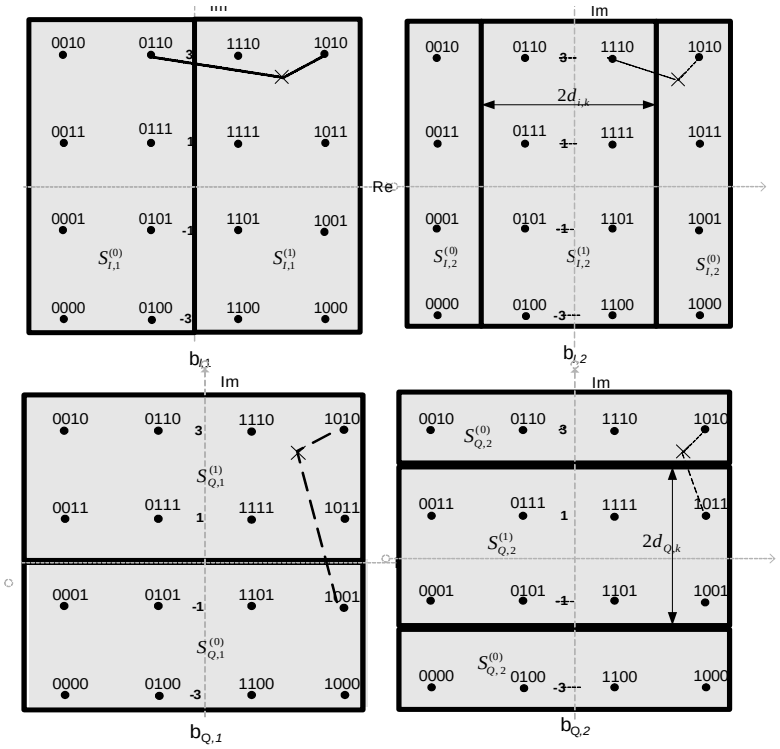

圖 2-3 16QAM 星狀圖的分割示意圖

(2-21)在計算上還是令人困擾，但如果犧牲一點效能的話，則可以再進一步化簡為:

$$D_{I,1} \cong Y_I(k)$$
$$D_{I,2} = -|Y_I(k)| + 2 \qquad (2\text{-}22)$$

(2-21)及(2-22)所算出之 $D_{I,1}$ 及 $D_{I,2}$ 的比較如圖 2-4 所示:

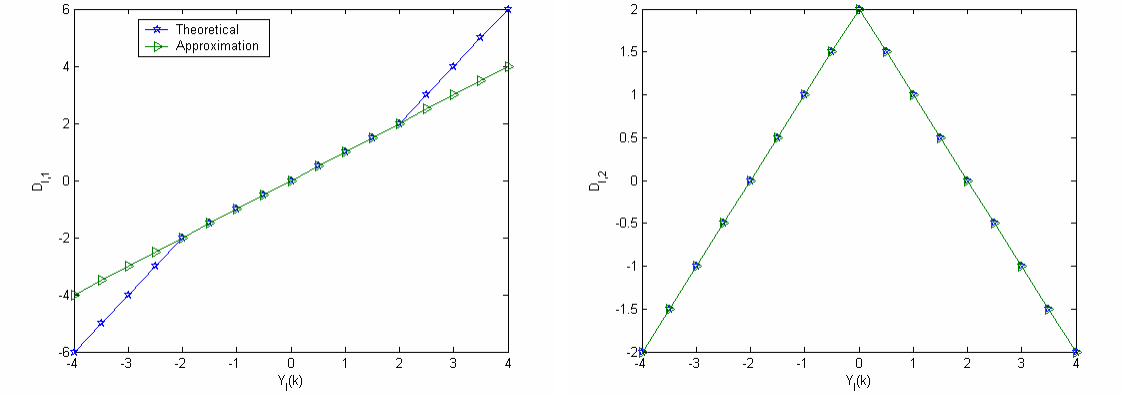

圖 2-4 In-phase 位元中，16QAM 之簡化對精確 LLR 計算方法比較圖

相同地，在 64QAM 的情況下可以導出(2-23)及其簡化式(2-24)

$$D_{I,1}=\begin{cases} Y_I(k), & |Y_I(k)|\le 2 \\ 2(Y_I(k)-1), & 2<Y_I(k)\le 4 \\ 3(Y_I(i)-2), & 4<Y_I(k)\le 6 \\ 4(Y_I(i)-3) & Y_I(k)>6 \\ 2(Y_I(i)+1) & -4\le Y_I(k)<-2 \\ 3(Y_I(i)+2) & -6\le Y_I(k)<-4 \\ 4(Y_I(i)+3) & Y_I(k)<-6 \end{cases}$$

$$D_{I,2}=\begin{cases} 2(-|Y_I(k)|+3), & |Y_I(k)|\le 2 \\ 4-|Y_I(k)| & 2<|Y_I(k)|\le 6 \\ 2(-|Y_I(k)|+5) & |Y_I(k)|>6 \end{cases}$$

$$D_{I,3}=\begin{cases} |Y_I(k)|-2, & |Y_I(k)|\le 4 \\ -|Y_I(k)|+6, & |Y_I(k)|>4 \end{cases}$$

(2-23)

$$D_{I,1}\cong Y_I(k)$$
$$D_{I,2}\cong -|Y_I(k)|+4$$
$$D_{I,3}\cong -|-|Y_I(k)|+4|+2$$

(2-24)

(2-23)及(2-24)所算出之 $D_{I,1}$、$D_{I,2}$ 及 $D_{I,3}$ 的比較如圖 2-5 所示:

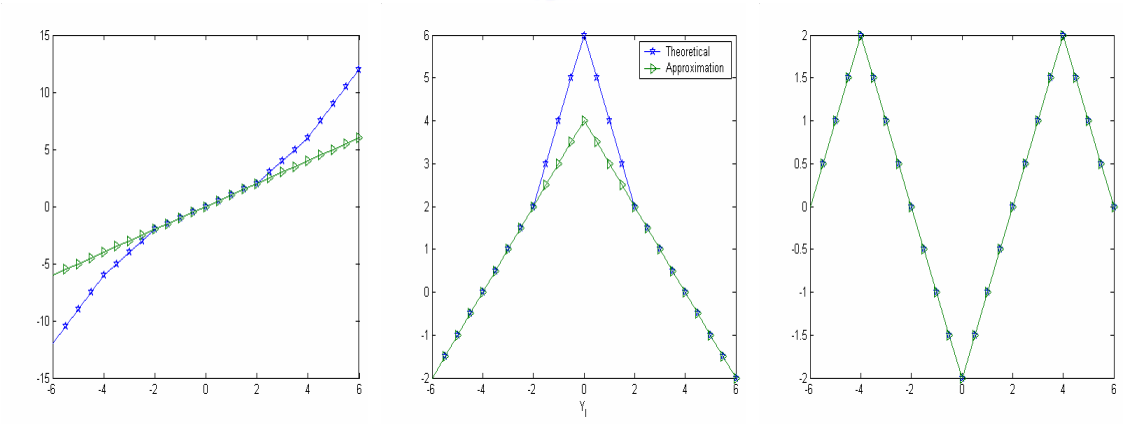

圖 2-5 In-phase 位元中，64QAM 之簡化對精確 LLR 計算方法比較圖

(2-24)可以推展到使用類似葛雷編碼的任意平方 QAM 的星狀圖上，假設 $d_{I,k}$ 及 $d_{Q,k}$ 為當 $k>1$ 時，二個集合之間距離的一半當如圖 2-3 所示，則從(2-21)到(2-24)

便可一般化為:

$$D_{I,k}=\begin{cases}Y_I(k), & k=1\\ -\left|D_{I,k-1}\right|+d_{i,k}, & k>1\end{cases}$$

$$L(b_{I,k})=CSI\times D_{I,k}$$

and

$$D_{Q,k}=\begin{cases}Y_Q(k), & k=1\\ -\left|D_{Q,k-1}\right|+d_{Q,k}, & k>1\end{cases}$$

$$L(b_{Q,k})=CSI\times D_{Q,k}$$

(2-25)

2.3.2 軟性輸入軟性輸出之MMSE接收機

這個接收器如圖 2-6 所示，為了方便說明，這裡用 2×2 的模型，也很容易推廣到多根傳送接收天線模型。相對於將接收訊號視為訊號向量，我們直接使用線性的接收器將此向量分成二條獨立的資料流，並分開地計算其軟性輸出(Soft output)。

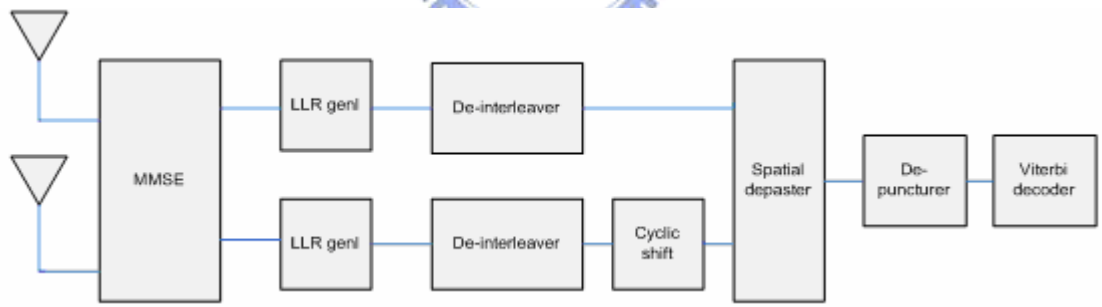

圖 2-6 2×2 之軟性輸入軟性輸出 MMSE 接收器

在 2×2 的情況下(2-2)中的接收訊號可以表示為:

$$Y_k=\begin{bmatrix}h_{1,k} & h_{2,k}\end{bmatrix}\begin{bmatrix}s_{1,k}\\ s_{2,k}\end{bmatrix}+N_k$$

(2-26)

$h_{i,k}$ 是 $s_{i,k}$ 的通道影響，也就是 H 的第 i 列。而 MMSE 的等化器則可以表示為

$$W = (H_k^H H_k + \frac{2}{\rho} I)^{-1} H_k^H = \begin{bmatrix} w_1^T \\ w_2^T \end{bmatrix}$$

(2-27)

ρ 是 SNR $\frac{E_S}{N_0}$,則此等化過後的訊號即變成 $Z_k = WY_k$ ，假設 w_i^T 是 W 的第 i 行，則對第 i 根天線而言，等化後的訊號即可表示為:

$$Z_{i,k} = \underbrace{w_i^T h_{i,k}}_{H_{eff}} s_{i,k} + \underbrace{w_i^T h_{j\neq i,k} s_{j,k} + w_i^T N_k}_{N_{eff}}$$

(2-28)

使用(2-28)中 Z_k 、 H_{eff} 及 N_{eff} 則可得等效之雜訊變異數為:

$$\sigma_{N_{eff}}^2 = w_i^H h_{j\neq i,k} h_{j\neq i,k}^H w_i + \left\| w_i \right\|^2 N_0$$

(2-29)

如此(2-28)便和在 2.3.1 中所介紹軟性反對映有相同的格式。注意本方法的計算複雜度跟 QAM 的大小和天線數成線性增加，且在這裡我們將其它根天線的殘餘天擾訊號當成雜訊處理而避免向量的解碼。

2.4 802.11n 編碼器及交錯器（Interleaver）

2.4.1 總覽

在 2005 3 月由 TGN SYNC 所提出來的 11n 提案[8]中，介紹了幾種傳輸技術及模式，如表 2-1 所示。而傳送架構便如圖 2-7 所示。

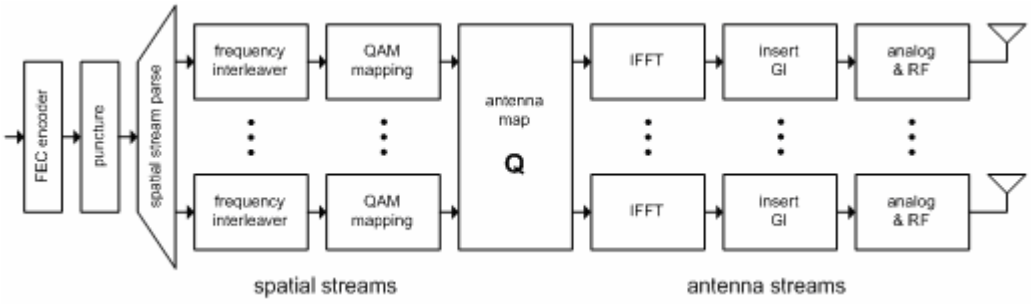

圖 2-7 TGN Sync 之傳送方塊圖

在 FEC 架構中有一個編碼器，對迴旋碼(convolutional code ,CC)來說，其主要的編碼率是 1/2。而這些編碼過後的位元資料可以經由壓縮(puncture)而產生 2/3、3/4 或 5/6 共四種編碼率。而對 LDPC 碼來說，則不應用其它的編碼率。

被壓縮後的位元透過空間資料流分離器(spatial stream parser)將資料分成 N_{SS} 個空間資料流。對每個資料流都會成塊地做交錯(interleaving)而會對映到一個 OFDM symbol 的大小。這樣的交錯稱為「頻率交錯」，這是因為資料是在不同子載波中間交錯的。這頻率交錯器和在 802.11a[9]中是十分相似的。結合空間資料流分離和頻率交錯便成了空時(space-time)交錯。交錯後的位元將會對映到 QAM 的星狀圖上，這對映可能是 BPSK、QPSK、16QAM、64QAM 或是 256QAM。

Feature	Mandatory	Optional
Number of Spatial Streams	1 and 2	3 and 4
Number of Transmit Antennas	2	Greater than 2
Channelization bandwidth	20MHz	40MHz
Number of Occupied Subcarriers	56 in 20MHz	114 in 40MHz
Number of Data Subcarriers	52	108
Number of Pilot Subcarriers	4	6
Modulation Order	BPSK, QPSK, 16-QAM, 64-QAM	256-QAM in Beamforming mode
Code Rate	1/2, 2/3, 3/4, 5/6	
Guard Interval	400ns and 800ns	
Convolutional Coding	R=1/2, K=7, ($g_1=133_8$, $g_2=171_8$)	
LDPC		TX and RX Optional
TX Beamforming	RX mandatory	TX Optional

表 2-1

下一步是天線對應轉換(antenna map transformation)。這轉換是將 N_{SS} 個空間資料流對應到 N_{Tx} 個傳送天線資料流。注意，在 MIMO 的傳送下必須要 $N_{SS} \le \min\{N_{Tx}, N_{Rx}\}$。特別地，我們總是讓 $N_{SS} \le N_{Tx}$。將 N_{SS} 資料對應到 N_{Tx} 根天線的動作可以由一個矩陣 Q 表示。Q 轉換模式總共有三種，在下一節中將只介紹本論文所模擬實現的直接對映(direct map)轉換架構，且主要介紹在本論文中所用到的元件，這包括了編碼、壓縮、空間分離及交錯。

2.4.2 直接對映(direct map) MIMO

直接對映轉換是在這提案中的基本模式，如圖 2-8 直接對映轉換之傳送方塊圖所示。在這基本模式中，矩陣Q是一個簡單的單位矩陣，也就是說每個空間資料流都對應到一根天線。因此對每個空間資料流都有個一對一的對映。

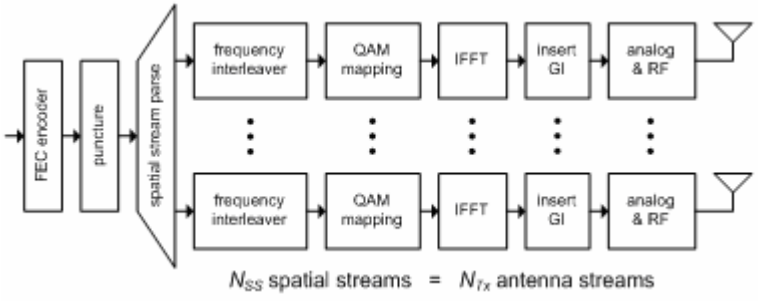

圖 2-8 直接對映轉換之傳送方塊圖

2.4.2.1 編碼(Encoding)

在 11n 所用的是使用限定長度(constrain length)為 7 的迴旋碼，編碼多項式分別為 $g_0 = 133_8$　$g_1 = 171_8$ 如所示。

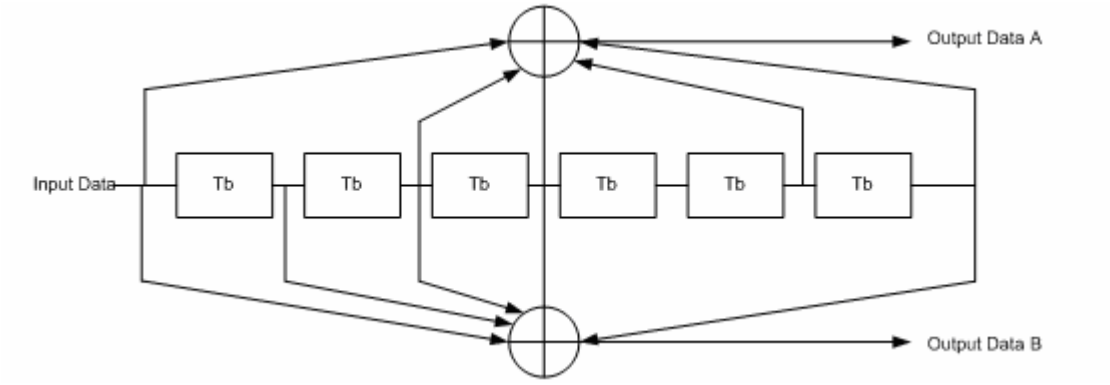

圖 2-9 迴旋編碼器

2.4.2.2 壓縮(Puncture)

資料位元在經過編碼後的編碼率是 1/2，如果想要有更高的編碼率則藉由在將固定順序的編碼後的資料位元打掉不送，而在解碼器時便在相同位置補上沒有意義的位元而使得編碼率變高。在這提案中，共有 1/2、2/3、3/4、5/6 四種速率可以選擇，其中除了 5/6 是 11n 所新增的，其它的速率都和 11a 相同。圖 2-10 壓縮流程圖為編碼率為 2/3、3/4 及 5/6 的例子。

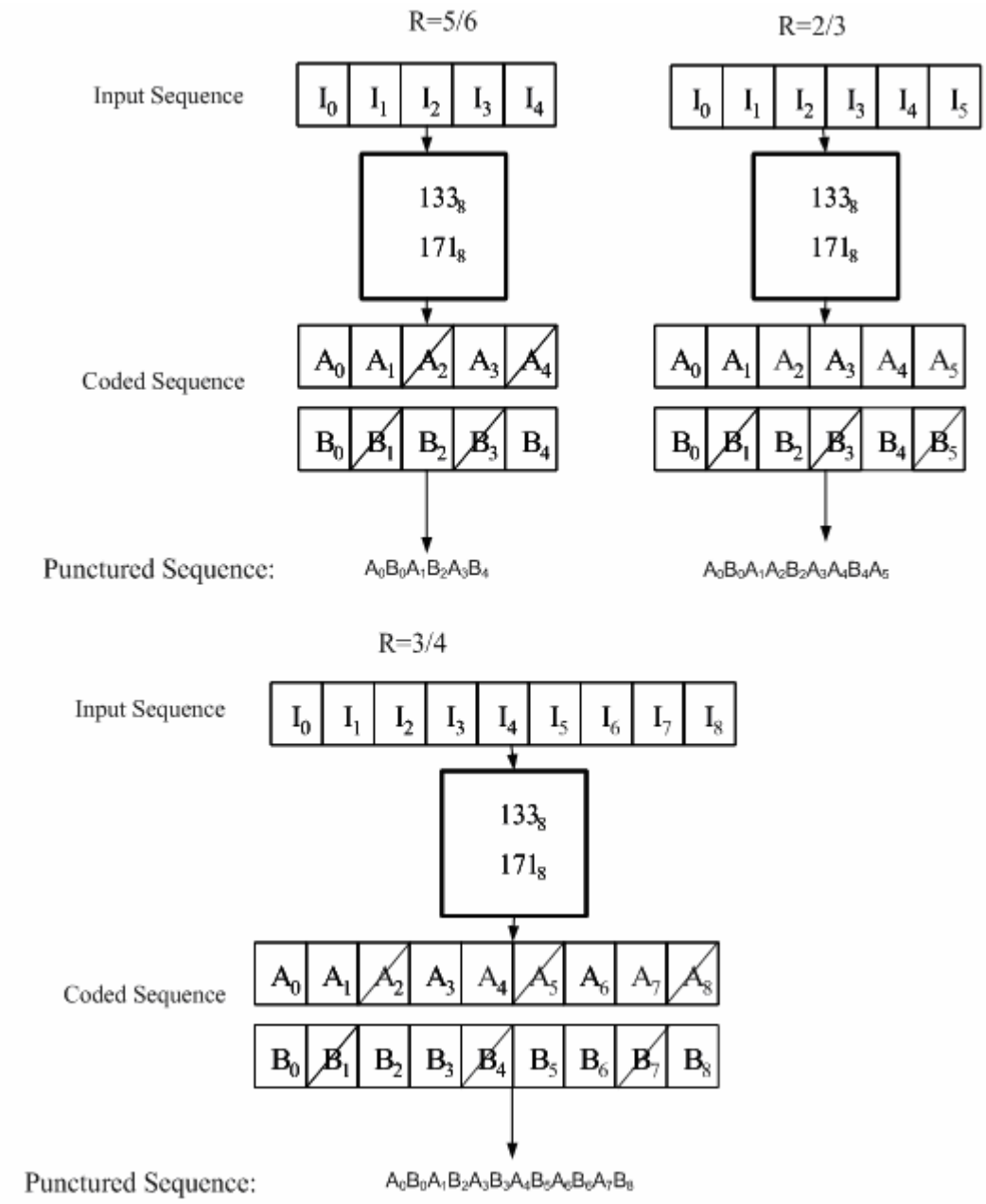

圖 2-10 壓縮流程圖

2.4.2.3 交錯器(Interleaver)

資料在經過編碼及壓縮後，會交錯地送到各個天線及子載波上。而此交錯器共有二個步驟:

1. 空間分離(space parsing):

$$s = \max\{N_{BPSC} / 2, 1\} \tag{2-30}$$

N_{BPSC} 是指在每個子載波上的位元數(number of bits per subcarrier)也就是使用的 QAM 的等級(order)，例如當 BPSK 時，N_{BPSC}=1，而 QPSK 時是 4。在壓縮後的資料會以大小為 s 的方塊，以 round-robin 的方式從第一根天線開始地放到各天線上去。

2. 頻率交錯器(frequency interleaver)

所有被分到各個天線上的資料位元都會被獨立的交錯到不同的子載波上去，而每個天線都有各自己的交錯器，大小為一個 OFDM symbol 的長度，N_{CBPS}。交錯器是根據 802.11a 中所定義的交錯器而加以修改成適用於多天線及 40MHZ 的傳輸。如圖 2-11 所示。

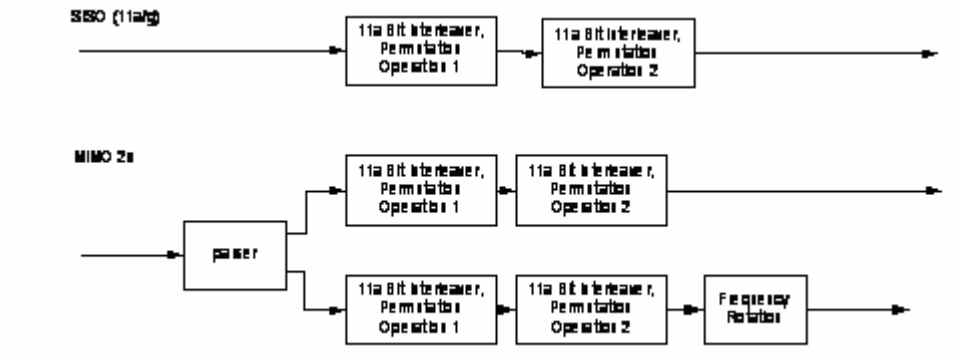

圖 2-11 頻率交錯器

此交錯器分為三步驟。第一次對調(permutation)為保證相隣的位元能對映到不相隣的子載波上。第二次對調則是能讓資料位元輪流地對映到星狀點上的 MSB 及 LSB 上。第三次對調保證讓資料位元透過頻率旋轉(frequency rotation)在所有的天線之間達到較好的頻率-空間分集。

基本的交錯器陣列有 N_{row} 行及 N_{column} 列，且頻率旋轉的基底則表示為 N_{rot} 。這些參數如表 2-2 及表 2-2 所示。

	N_{row}	N_{column}	N_{rot}
20MHZ	13	$4N_{BPSC}$	11
40MHZ	18	$6N_{BPSC}$	29

表 2-2 頻率交錯參數表

不同的天線則有頻率旋轉數，列表如下:

Frequency Rotation	Total Number of Streams			
	1	2	3	4
1st stream	0	0	0	0
2nd stream		$2N_{rot}$	$2N_{rot}$	$2N_{rot}$
3rd stream			N_{rot}	N_{rot}
4th stream				$3N_{rot}$

表 2-3 頻率旋轉參數表

假設在第一次對調之前資料的索引(index)為 k ，而在第一次對調之後第二次對調之前的索引為 i ， j 則為第二次對調及第三次對調之間的索引， r 是第三次對調之後 QAM 對映之前的索引。則我們可以將這三次對調用數學表示:

第一次對調是根據以下規則所定義:

$$i = N_{row} \times \left(k \bmod N_{column}\right) + \text{floor}\left(k / N_{column}\right) \quad k = 0,1,\ldots,N_{CBPS} - 1 \tag{2-31}$$

第二次對調是根據以下規則所定義:

$$j = s \times \text{floor}\left(i / s\right) + \left(i + N_{CBPS} - \text{floor}\left(N_{column} \times i / N_{CBPS}\right)\right) \bmod s \quad i = 0,1,\ldots,N_{CBPS} - 1 \tag{2-32}$$

其中 $s = \max\{N_{BPSC} / 2, 1\}$。

第三次對調是根據以下規則所定義:

$$r = \left(j - \left(\left(2 \times i_{ss}\right) \bmod 3 + 3 \times \text{floor}(i_{ss} / 3)\right) \times N_{rot} \times N_{BPSC}\right) \bmod N_{CBPS} \quad j = 0,1,\ldots,N_{CBPS} - 1 \tag{2-33}$$

$i_{SS} = 0,1,\ldots,N_{SS} - 1$，代表天線的索引，第一根天線 i_{SS} =1。

反交錯器(deinterleaver) 則相對應地分成三次對調來定義:

假設在第一次對調之前資料的索引(index)為 r，而在第一次對調之後第二次對調之前的索引為 j，i 則為第二次對調及第三次對調之間的索引，k 是第三次對調之後 QAM 的反對映之前的索引。則我們可以將這三次對調用數學表示:

第一次對調是根據以下規則所定義:

$$j = \left(r + \left(\left(2 \times i_{ss}\right) \bmod 3 + 3 \times \text{floor}(i_{ss} / 3)\right) \times N_{rot} \times N_{BPSC}\right) \bmod N_{CBPS}, \qquad r = 0,1,\ldots,N_{CBPS} - 1 \tag{2-34}$$

第二次對調是根據以下規則所定義:

$$i = s \times \text{floor}\left(j / s\right) + \left(j + \text{floor}\left(N_{column} \times j / N_{CBPS}\right)\right) \bmod s, \qquad j = 0,1,\ldots,N_{CBPS} - 1 \tag{2-35}$$

第三次對調是根據以下規則所定義:

$$k = N_{column} \times i - \left(N_{CBPS} - 1\right) \text{floor}\left(i / N_{row}\right), \qquad i = 0,1,\ldots,N_{CBPS} - 1 \tag{2-36}$$

(2-36)至(2-34)即是(2-31)至(2-33)的反對調。

第3章BICM MIMO-OFDM 系統演算法

現存的系統中大多以 MMSE 及 VBLAST 為 MIMO 偵測的方法，再經過解碼器後輸出。然而在 VBLAST 中依序干擾消除演算法(SIC)的效能則取決於前一級決策(Decision)的錯誤率，若能結合解碼器勢必能提高效能表現。本論文以多(平行)解碼器架構及單一解碼器(iterative)二種系統討論如何在 BICM MIMO OFDM 系統中結合 VBLAST 及 Viterbi 解碼器。在 3.1 中的多(平行)解碼器架構中，訊號在一定的延遲後便可有和在第二章中介紹的系統有相同的吞吐量(through put)，然而硬體的增加則是它的代價。相對的在 3.2 中的具遞迴之單一解碼器架構，其硬體增加不大，但吞吐量則隨著遞迴次數增加而降低。

3.1 Pre-ordering for V-BLAST and coding

3.1.1 V-BLAST 及解碼

在本節中我們先介紹在[18]所提出的之多解碼器之架構，如圖 3-1 所示。在第二章所介紹的 VBLAST，由於訊號在頻域上處理，所以訊號在經過 FFT 之後，共會分成 N_c (FFT size)個 VBLAST 處理單元，和 2.2 中所介紹的不同的是，在這架構中，並無法使用最佳排序，所以訊號總是從第一根天線依序往下解。在[18]所提出的架構並沒有說明是否使用軟性偵測，而在本節將使用如 2.3 中所介紹軟性偵測輸出。定義在經過 Soft Demapping 之後的軟性訊號為 $v_{k,i}^{n}$ ，n 是第幾個位元， k 是第幾個幾個子載波，i 則是在第幾根天線上，則(2-19)可以表示為

$$v_{k,i}^{n}=CSI_{k,i}\times D(\tilde{s}_{k,i},n) \tag{3-1}$$

定義 $Z_{k,i}$ 為經過 MMSE 等化後的訊號，而 $CSI_{k,i}$ 可以解釋為偵測訊號對干擾雜訊比(SINR)。

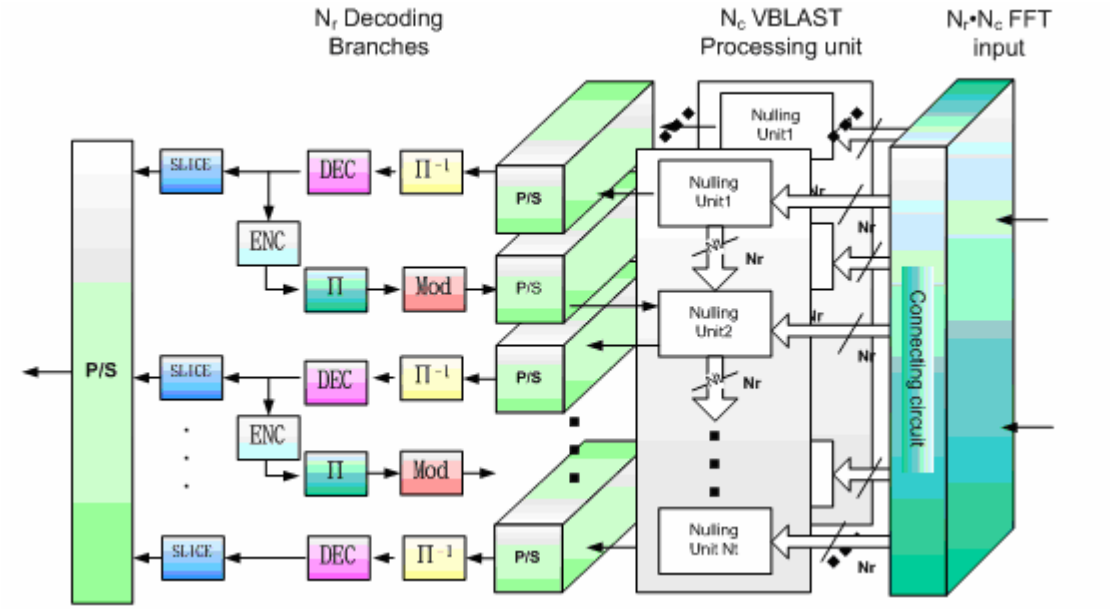

圖 3-1 V-BLAST 偵測及解碼方塊圖

在(3-1)中 $Z_{k,i}$ 及 $CSI_{k,i}$ 是根據不同偵測方法而有不同的算法。例如在這裡所要介紹的 MMSE-SIC 偵測器，$Z_{k,i}$ 便可以由以下的式子所表示:

$$Z_{k,i} = [(H_k^H(j)H(j) + \alpha I)^{-1}]_i H_k^H(j) Y_k(j) \tag{3-2}$$

$$H_k(j) = [h_{k,i} \ldots h_{k,N_t}]$$

$$Y_k(j) = Y_k(i-1) - h_{k,i-1}\hat{s}_{k,i-1}, Y_k(1) = Y_i$$

$[A]_i$ 代表 A 的第 i 行。$H_k(i)$ 及 $Y_k(i)$ 代表『等效』的通道矩陣及接收訊號向量。在(3-2)中 $\hat{s}_{k,i-1}$ 原本在 MMSE-SIC 的演算中是指 $Q[Z_{k,i-1}]$，也就是將 $Z_{k,i-1}$ 判斷(slice)到最近的 QAM 上所得到的訊號。然而這裡所代表的是指 $Z_{k,i-1}$ 經過反交錯器、解碼器之後，再重建回來訊號。在使用 MMSE-SIC 偵測器的情況下，$CSI_{k,i}$ 的計算便如同 2.3.2 中所介紹的步驟一樣。首先由 (3-2)算出在(2-23)中的 N_{eff} 及 H_{eff}，再套入 CSI 的定義 $CSI_{k,i} = \frac{\left|(H_{eff})_{k,i}\right|^2}{\sigma_{eff}^2}$。值得注意的是，這樣的 CSI 算法是建立在

假設 $\hat{s}_{k,i-1} = s_{k,i-1}$，然而這樣的假設卻不一定成立，這使得 CSI 的算法要有所改變。但是考慮 $\hat{s}_{k,i-1}$ 的錯誤率並不容易計算。而有一更簡單的做法是直接使用 MMSE 的 CSI 如同 2.3.2 中所描述，不同的是，CSI 產生用 MMSE 的方法，訊號偵測使用本節所介紹的 MMSE 加上 SIC。

3.1.2 Pre-ordering V-BLAST 及解碼

在 3.1.1 在介紹的系統都是從第一根天線開始解，如果第一根天線的通道狀況差的話，便有可能導致錯誤傳遞(error propagation)，因此吾人提出一個系統，修改由 3.1.1 所提出的架構，在傳送端及接收端都有一個資料流排序(stream ordering)的動作，而使得第一根天線開始解的訊號是 SNR 最高的通道所傳送的訊號，系統修改如圖 3-2 及圖 3-3 所示。在吾人所提出的系統中，在傳送端的排序是最重要的部分，而在各頻率排序所需要的資訊便得從接收端傳送過來。

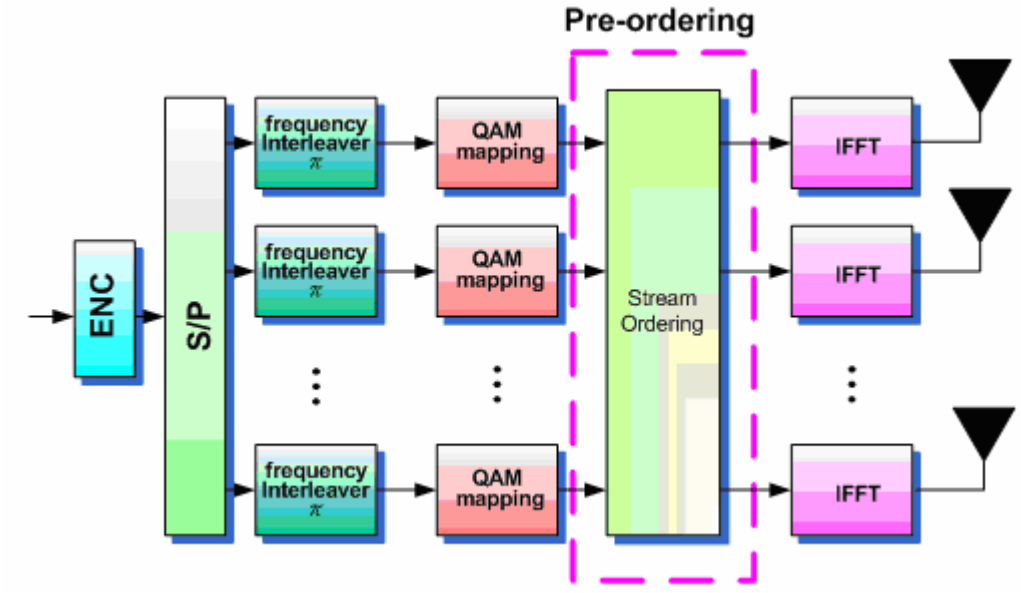

圖 3-2 Pre-ordering 系統傳送方塊圖

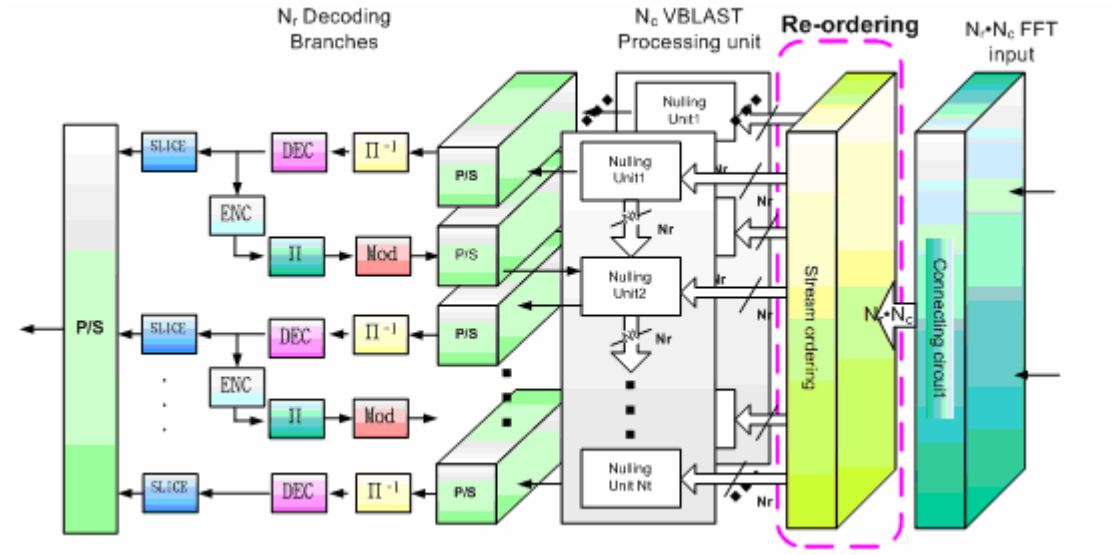

圖 3-3 Pre-ordering 系統接收偵測及解碼方塊圖

我們舉個簡單的例子來說明傳送端是如何做排序的。為了說明方便，我們將 FFT 的大小設定為 6，而傳送天線數為 3，如此一來整個要傳送的 MIMO-OFDM symbol 大小即是 18，我們將整個在 QAM 對映後的 MIMO-OFDM symbol S 編號為:

$$S = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ s_7 & s_8 & s_9 & s_{10} & s_{11} & s_{12} \\ s_{13} & s_{14} & s_{15} & s_{16} & s_{17} & s_{18} \end{bmatrix}$$

而假設在上個時間由接收端傳送回來的 SNR 順序可用一個矩陣 O 表示:

$$O = \begin{bmatrix} 1 & 1 & 2 & 1 & 1 & 2 \\ 2 & 2 & 1 & 3 & 2 & 3 \\ 3 & 3 & 3 & 2 & 3 & 1 \end{bmatrix}$$

O 矩陣中，同一行代表同一天線而不同頻率，而同一列則代表同一頻率而不同天線，而 V-BLAST 則是同一頻率下做不同天線之間的干擾消除，在同一頻率下(同一列)的數字(1、2 或是 3)代表著 SNR 的高低順序，〝1〞代表在這頻率下，這根天線有著最高的 SNR，也就是希望在接收端時這根天線的訊號能最先解。則 pre-ordering 的動作便是將 S 根據 O 重新排序為 S':

$$S' = \begin{bmatrix} s_1 & s_2 & s_9 & s_4 & s_5 & s_{12} \\ s_7 & s_8 & s_3 & s_{16} & s_{11} & s_{18} \\ s_{13} & s_{14} & s_{15} & s_{10} & s_{17} & s_6 \end{bmatrix}$$

可以看出原本在S中打算給第一根天線傳送的$s_1, s_2 \cdots s_6$在重新排序後的被放到在O矩陣中數字為 1 的位置上去了，這代表著$s_1, s_2 \cdots s_6$將會在不同頻率中，SNR 最高的天線上傳送。相對的在接收端時，便是將S'再重新排回S，當然通道矩陣也要有相對應的排列，而之後的動作便和 3.1.1 中描述的相同。如此一來，在圖 3-3 中最上面的那根天線所解的必是在不同頻率中，通道最好的訊號。而且這些訊號是連續的，這樣一來訊號可以經過 Viterbi 解碼器後，再回授給偵測器而幫助 MIMO 偵測。值得注意的是，在使用這樣的方法時，我們需要通道狀況的資訊來排序，所以如果通道狀況變化太快便使得上個時間點由接收端回傳的資訊不太準確，這會使得效能降低。因此 Pre-ordering 演算法是較適用在通道變化較為緩和的環境中。

3.2 遞迴解碼(Iterative decoding)

一般而言，「遞迴(iterative)」這個字代表著「渦輪(turbo)」，也就是藉由著傳遞各個接收器的所有單元的外在資訊(extrinsic information)，這接收器的所有處理單元都是軟性輸入軟性輸出(soft input soft output)，包括 Viterbi 解碼器。而方面的處理技術[19][20]雖然效能很好，但運算複雜度卻非常高，這並不是本論文所要討論的重點。相對的，在本論文中所描述的遞迴是指資料經過解碼器之後錯誤率降低，進而將資料回授再去幫助 MIMO 偵測所產生之更好的 MIMO 偵測的軟性輸出值，而有更好的效能表現。這裡的使用的解碼器是軟性輸入硬性輸出(soft input hard output)，如圖 3-4 所示。

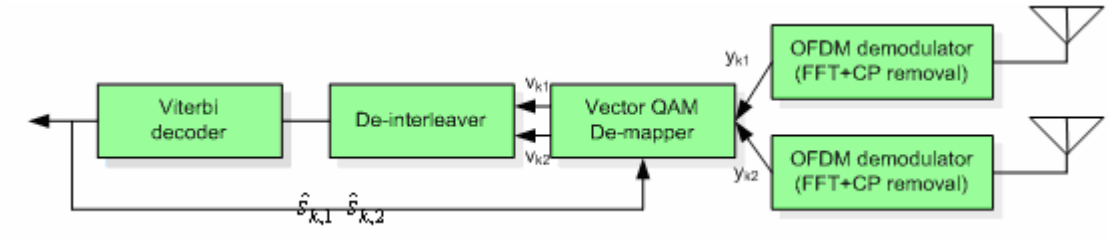

圖 3-4 遞迴偵測及解碼方塊圖

在圖 3-4 中，Vector QAM De-mapper 這處理單元中，做法可以有很多，舉例來說，因為 V-BLAST 在解碼時，需要準確的估計值來消除它對其它根天線的干擾，然而如果沒有經過編碼器而直接判斷(slice)的話，錯誤率會有太高會有錯誤延展的可能，也使得 CSI 的計算變的複雜而困難，結果反而使用 V-BLAST 在結合 Viterbi 時效能表現反而不如 MMSE 好，因此，可以在第一次用 MMSE 先解出值來，經過 Viterbi 之後再回來使用 V-BLAST 做 MIMO 偵測。另外有一種更簡單而有效的方式，是第一次使用 MMSE 而回授時再使用最大比例結合(MRC)的方式做偵測[21]。底下將介紹這演算法:

套用之前假設: $v_{k,i}^{n}=\mathrm{CSI}_{\mathrm{k,i}}\times D(\tilde{s}_{k,i},n)$，$v_{k,i}^{n}$ 是指在第 i 個天線的第 k 個子載波上第 n 個位元的機率值，而 $\tilde{s}_{k,i}$ 則是第 i 根天線第 k 個子載波上的將接收訊號 Y 等化過後的 QAM 訊號。我們將這演算法分成二部分，分別為初始階段及遞迴階段，在二階段中，目的都是求出 $Z_{k,i}$，再代入 (3-1)求出 $v_{k,i}^{n}$，分述二階段如下:

I. Initial Stage(MMSE Receiver):

$$Z_{k,i}^{(1)}=(H_k^H H_k+\frac{1}{\rho}I)^{-1}H_k^H Y_k \tag{3-3}$$

II. Iterative Stage(Maximal Ratio Combining with Interference Cancellation)

$$Z_{k,1}^{(2)}=\frac{h_{k,1}^H}{\left\|h_{k,1}\right\|^2}(Y_k-h_{k,2}\hat{s}_{k,2}^{(1)})$$

$$Z_{k,2}^{(2)}=\frac{h_{k,2}^H}{\left\|h_{k,2}\right\|^2}(Y_k-h_{k,1}\hat{s}_{k,1}^{(1)})$$

(3-4)

在遞迴階段的演算法如圖 3-5 所示。在(3-3)及(3-4)中，便是列出如何求得在(3-1)所需要的 $Z_{k,1}$，$\hat{s}_{k,i}^{(m)}$ 是代表由 $Z_{k,1}^{(m)}$ 算出的 $v_{k,i}^{(m)}$ 再經過解碼器後重組回來的訊號，上標 m 則是代表遞迴次數。這裡要強調的是不管在初始階段還是遞迴階段，我們都使用由 MMSE 所算出來的 CSI，這和在[20]中所描述的傳統做法不同，它描述 CSI 應該要在根據前次的估計值是正確的假設下而更新。然而在我們模擬中卻是在這樣的 CSI 有較好的效能改進。在假設前次的輸出值是正確的情況下，(3-4)則等效於 1 根傳送天線 2 根接收天線所做的 MRC。嚴格上來說，我們應該使用 MRC 的 CSI， 即 $CSI_{k,i}=\left\|h_{k,i}\right\|^2$，然而這個值總是比 MMSE 的 CSI 大，所以在決策值有誤差的情況下，這可能使得雜訊放大，而讓效能表現不如預期。

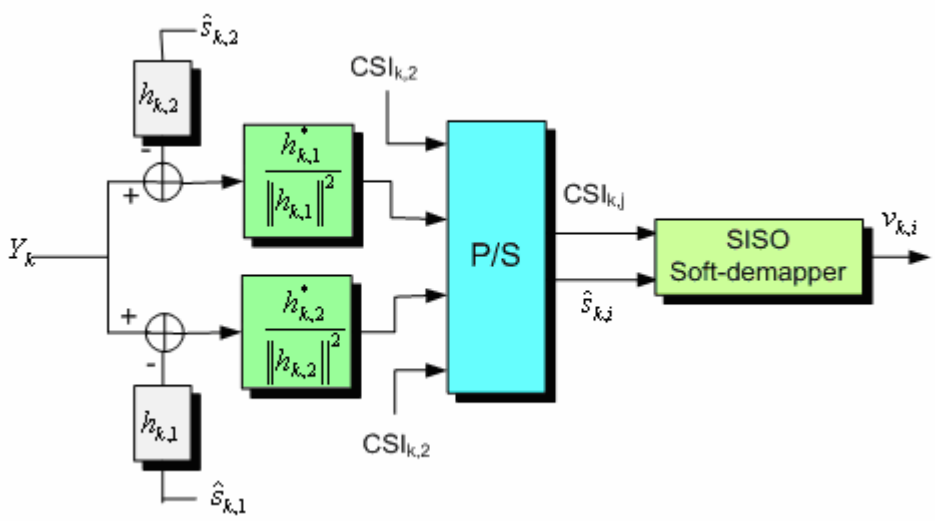

圖 3-5 簡化之遞迴軟性反對映方塊圖

3.3 模擬結果

在本節中，我們主要模擬 11n 提案中的 2×2 的基本模式，詳細的環境已在第二章中說明，在這樣的環境下我們模擬第二章及第三章所提及的各種演算法。首先我們先定義模擬用的通道，在每根傳送天線 i 跟每根接收天線 j 之間的通道可以模擬為一個 FIR 濾波器，並且可以用序列 $[...H_{ij,l}...], l=0,\cdots,L$ 表示，L 代表最大 tap 數，且這些 taps 都是由獨立複數高斯隨機變數所產生。而每個 tap 可用表示如下:

$$H_{ij,l} = N(0,\sigma_l^2) + j \cdot N(0,\sigma_l^2)$$
$$\sigma_l^2 = \sigma_0^2 \exp(-lT_s / T_{RMS})$$
$$\sigma_0^2 = 1 - \exp(-T_s / T_{RMS}) \quad (3\text{-}5)$$

我們令 $T_s = 50ns$ 且 $T_{RMS} = 150ns$，且在模擬用的 SNR 是定義在接收到每個 OFDM 符碼後，計算出經過通道後的訊號能量加入在這 SNR 下該有的雜訊能量，因此我們並沒有將通道正規化。

模擬的演算法主要分成 3 部分，第一部分主要在模擬比較 Soft MMSE 及 V-BLAST 演算法在上述的環境中的效能表現，而 CSI(channel state information) 便是影響 MMSE 及 V-BLAST 的效能最主要的因素，這部分的模擬結果包括了圖 3-6 至圖 3-10。圖 3-6 比較了 2 種 MMSE 作法，包括了二種 Soft CSI 計算方式，其中 SC 是代表單一編碼器的架構，這是相對於之後多編碼器架構的 MC。MMSE 的 CSI 計算方式是使用 2.3.2 所介紹，而在圖中 NO CSI MMSE 則是令 CSI 為 1 的情況。圖 3-7 是比較不同 CSI 計算方式對 V-BLAST 的效能表現，其中 CSI1 是在 V-BLAST 演算法中，假設前一級的決策是正確的情況下，使用一級一級降低的通道矩陣所計算出來的 CSI，而 CSI2 則是直接使用第一級 MMSE 的計算出來的 CSI，也就是使用演算法 MMSE 的 CSI。而 V-BLAST bound 則是指在演算法中遞迴階段需要扣除前一級的決策時，我們使用完全正確的值來扣除，這提供

了 Soft V-BLAST 演算法的界線。而圖 3-8 至圖 3-10 則綜合比較 MMSE 及 Soft V-BLAST 在其它環境下的表現，共包括了不同的星狀圖及不同天線。從模擬結果發現，MMSE 的效能明顯較 V-BLAST 好，這結果與一般的認知相反，這可能是 Soft V-BLAST 在計算 CSI 上並不容易，而假設前一級的決策是正確的情況下，所計算出來的 CSI1 效能又不如預期。而使用 CSI2 的 Soft V-BLAST 所表現出來的效果又和單純使用 MMSE 差不多。

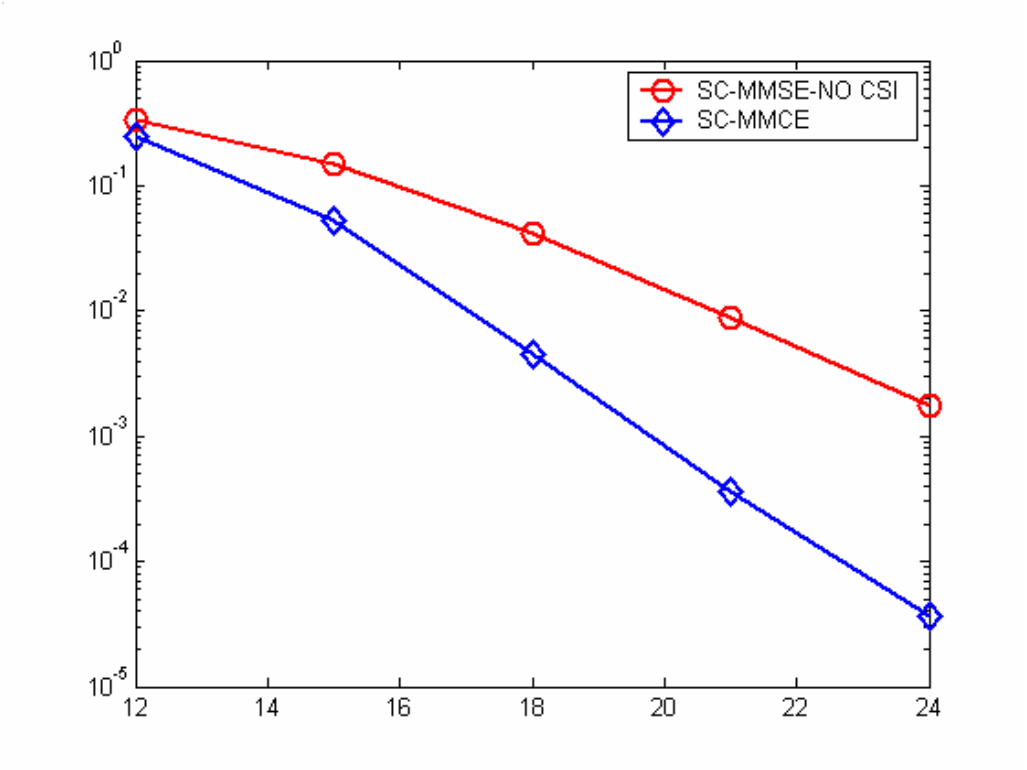

圖 3-6 MMSE(64QAM)

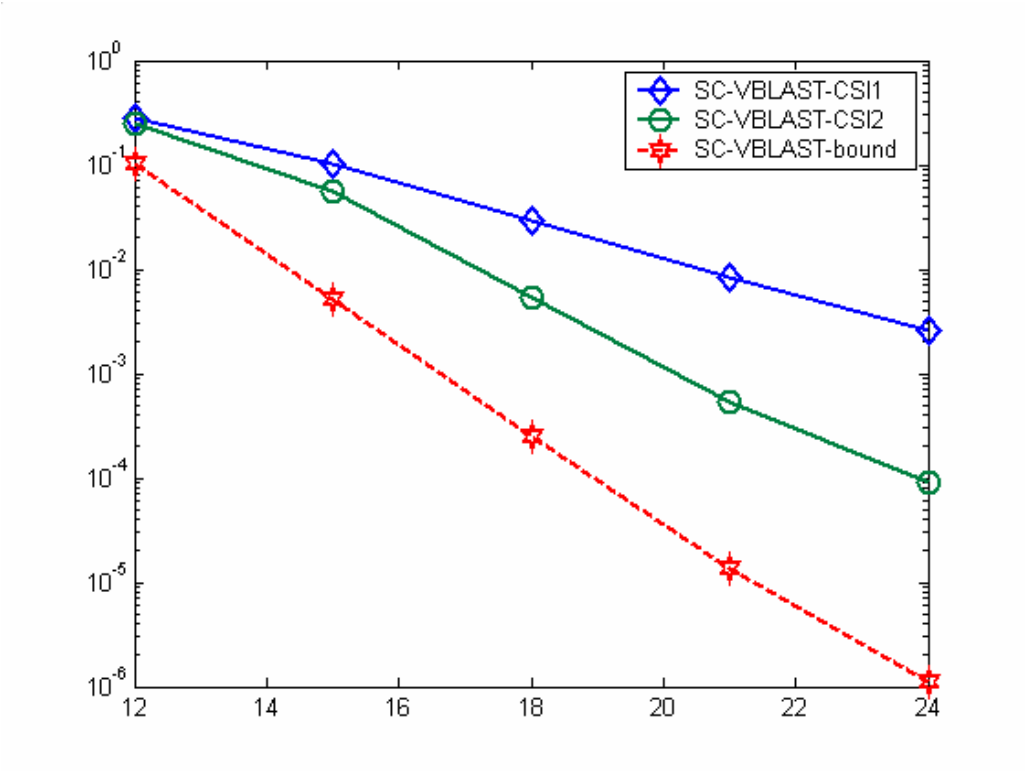

圖 3-7 V-BLAST (64QAM)

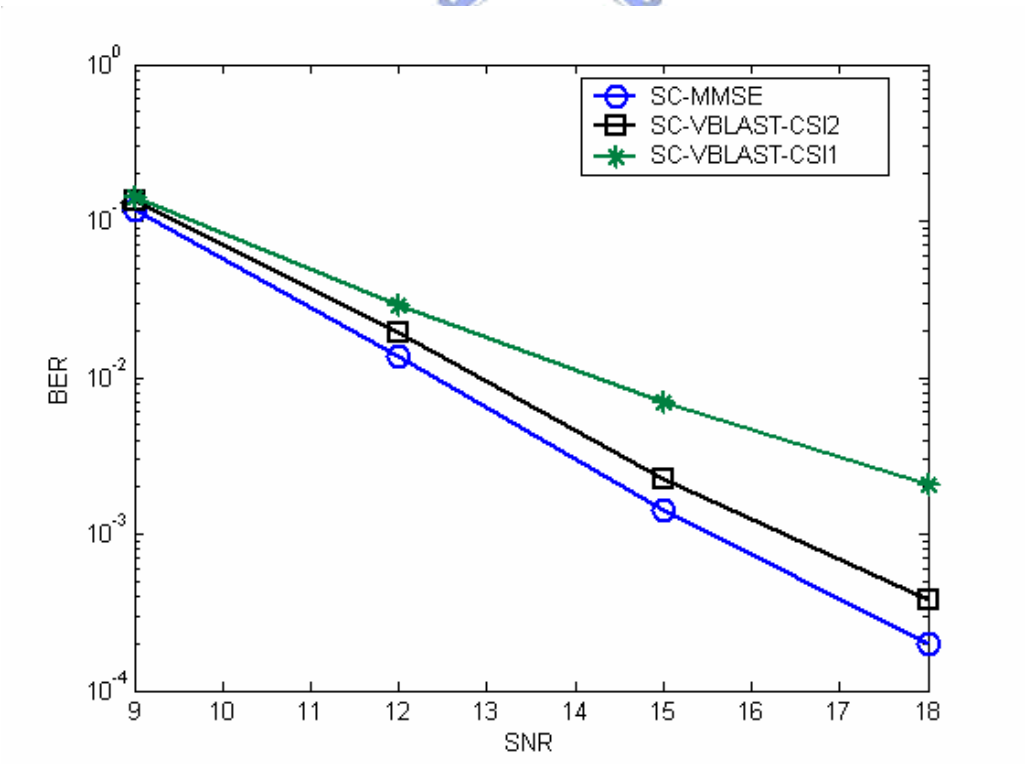

圖 3-8 comparison soft V-BLAST with MMSE (16QAM)

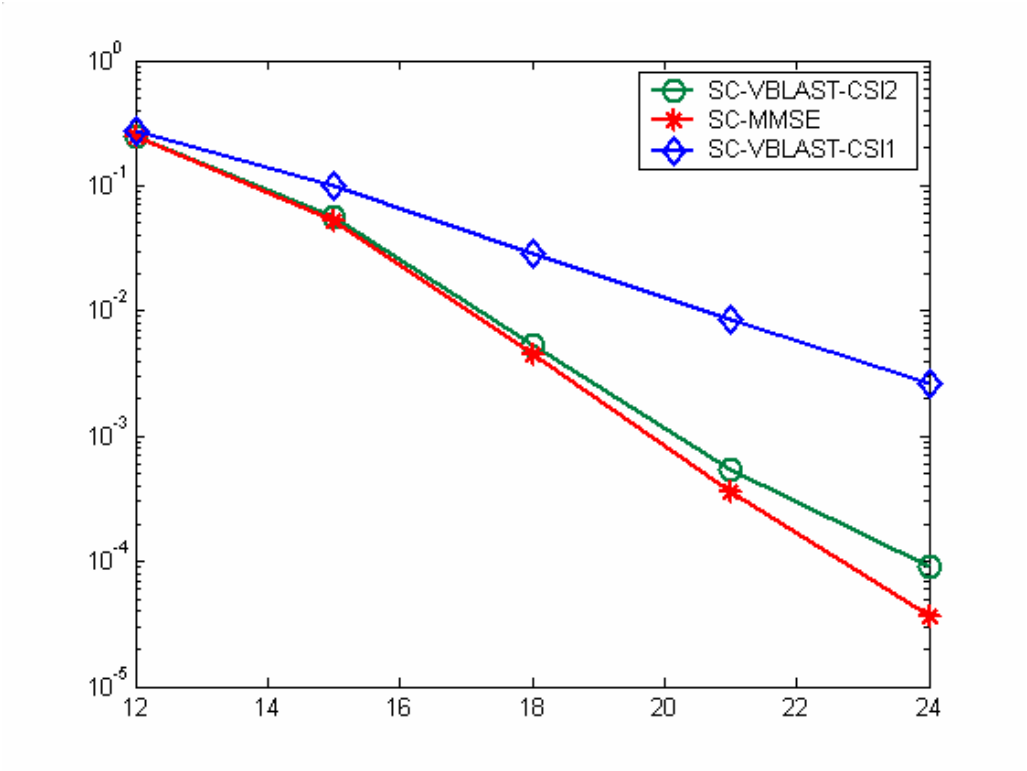

圖 3-9 comparison soft V-BLAST with MMSE (64QAM)

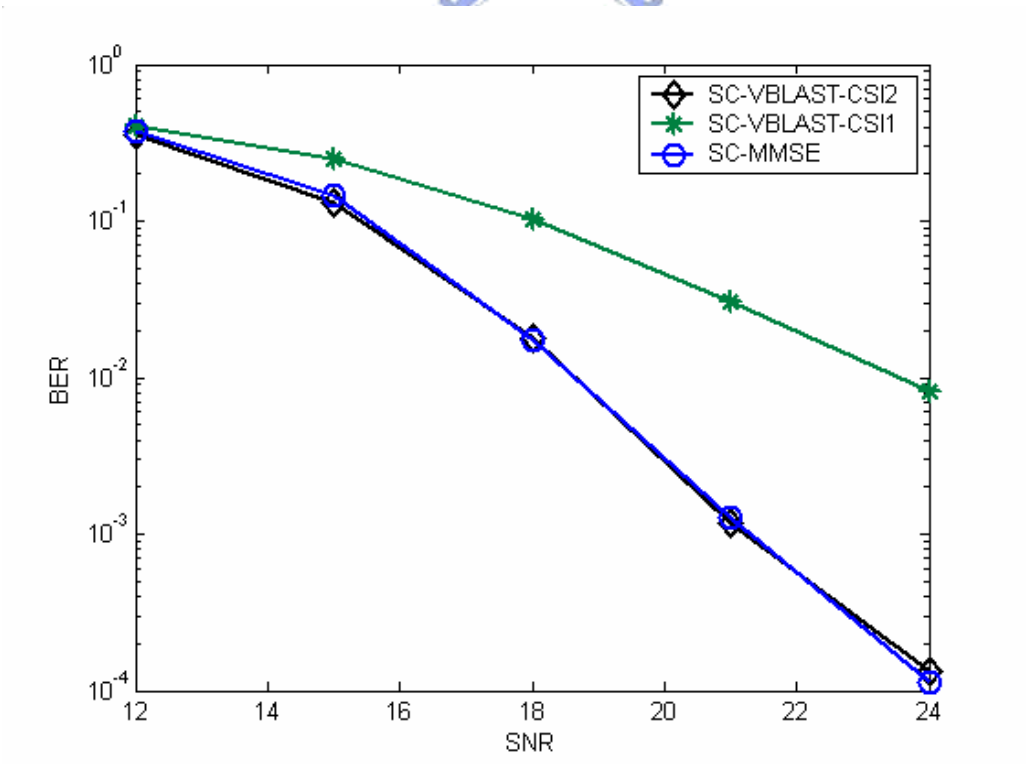

圖 3-10 3x3 comparison soft V-BLAST with MMSE (64QAM)

第二部分的模擬則是模擬在 3.2 中所介紹的使遞迴的方來增加效能，遞迴的演算法主要包括二個階段，第一階段是使用 MMSE，而第二階段便可使用 MRC combining 及 soft V-BLAST 二種方法。這部分的模擬便是比較 MRC combining 及 V-BLAST 二種演算及各別使用 CSI1 及 CSI2 的情況。同樣的，CSI1 是指在第二階段中，使用假設決策值是正確的情況下所算出來的 CSI，而 CSI2 則是使用 MMSE 的 CSI 也就是和第一階段 soft MMSE 使用同樣的 CSI。此部分的模擬結果包括了圖 3-11 至圖 3-14。在圖 3-11 中 iterative bound 的曲線是指我們使用正確的值來供演算法中第二階段使用。在這部分的模擬中，我們可以發現其實不管是 MRC combining 或是 V-BLAST 對效能的改進都並不是非常明顯，這可能是因為在吾人使用的通道環境下，使用 MMSE 演算法已經有相當好的效能，而使得其它演算法並無法提供很大的幫助。

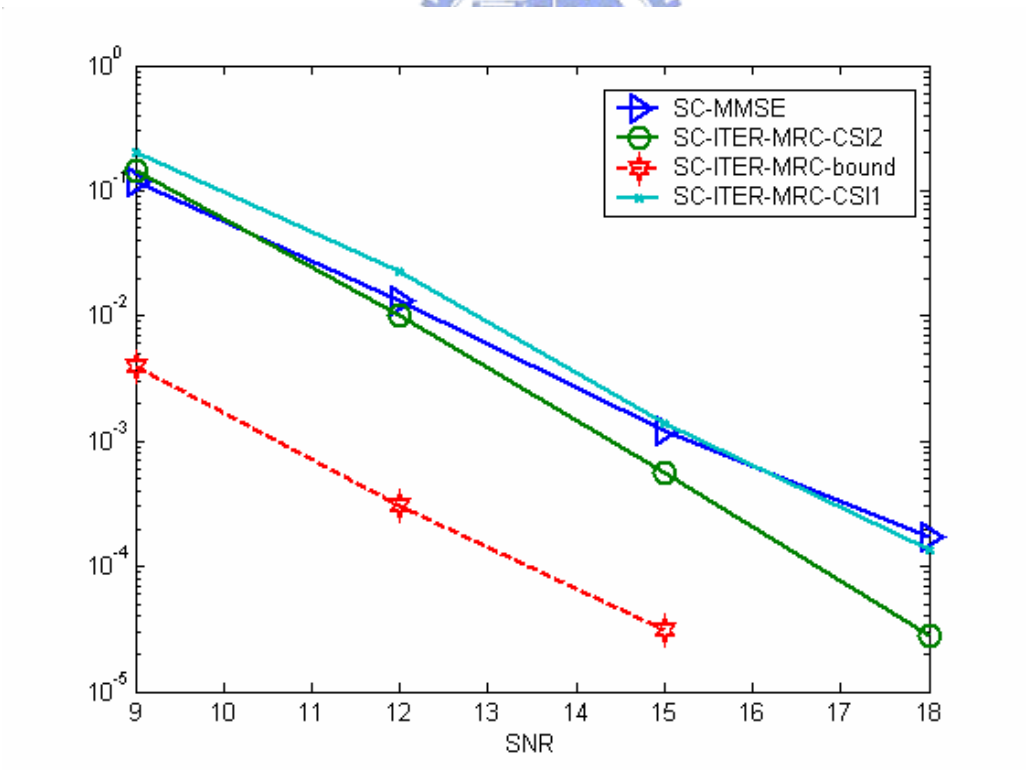

圖 3-11　iterative MRC (16QAM)

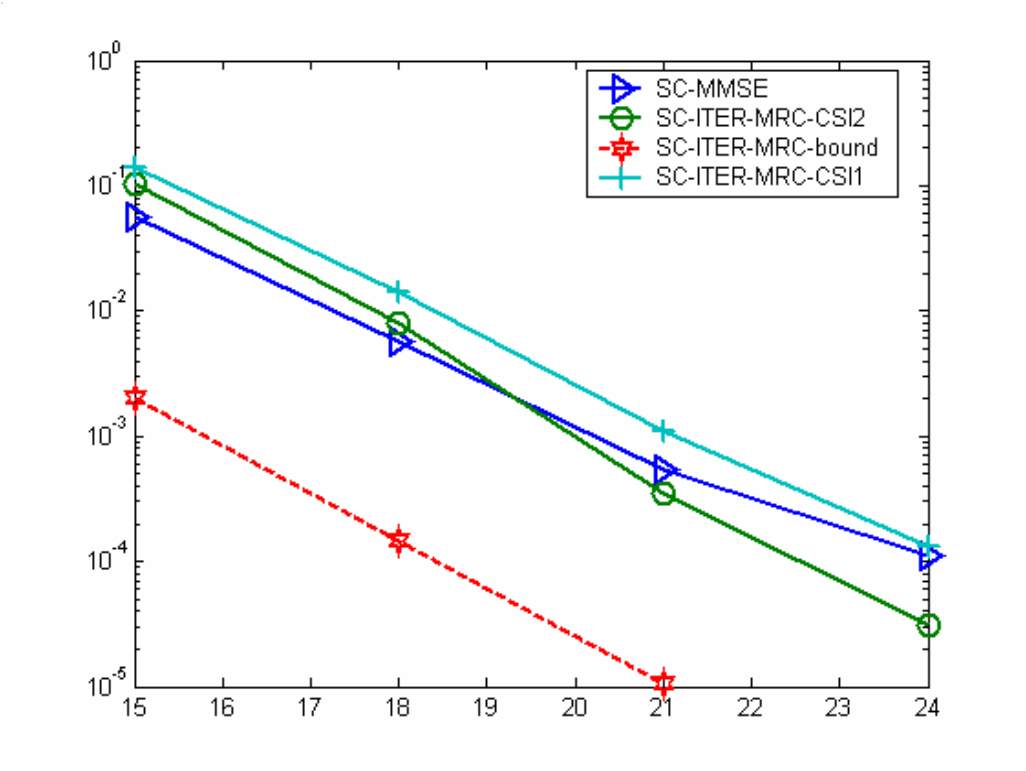

圖 3-12 iterative MRC (64QAM)

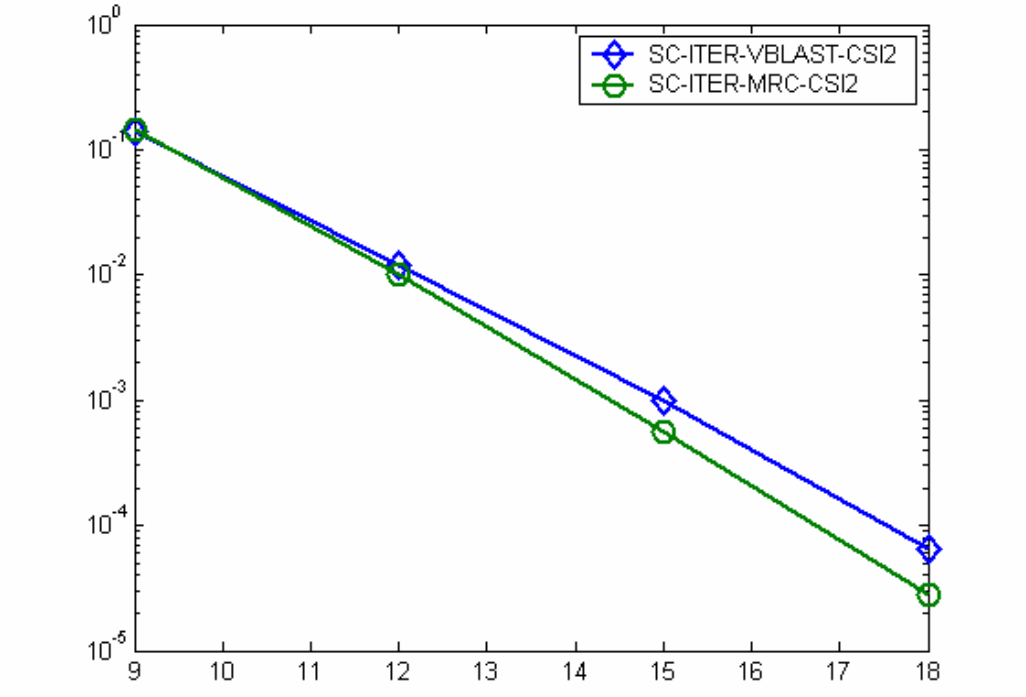

圖 3-13 iterative MRC V.S iterative V-BLAST (16QAM)

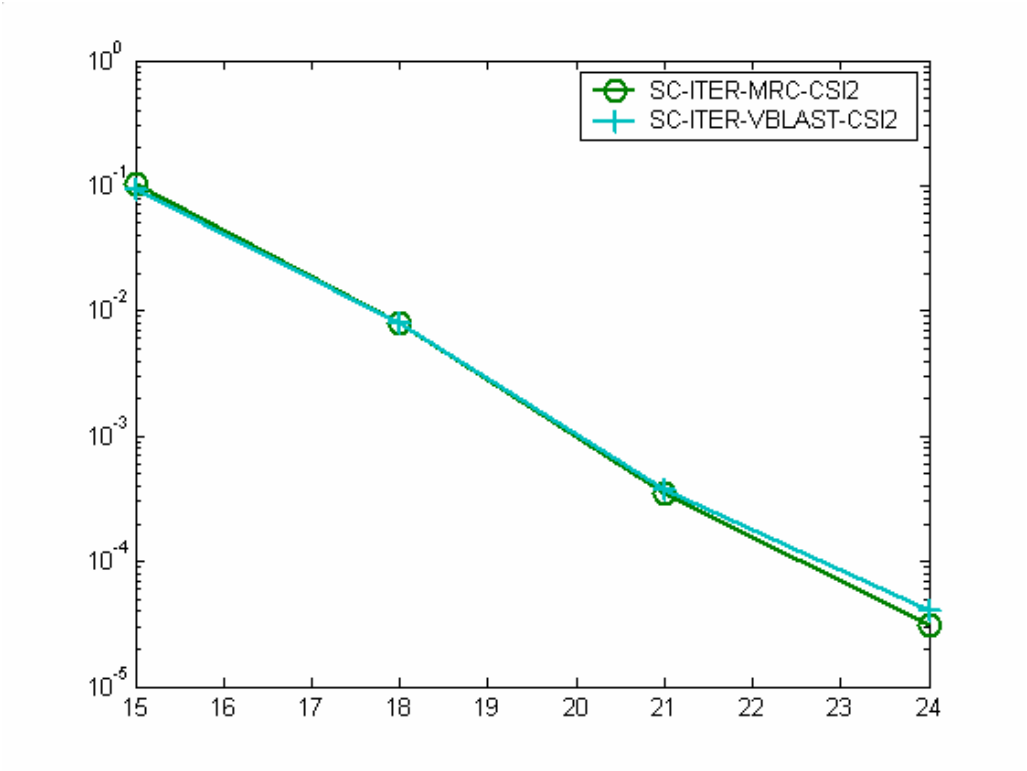

圖 3-14　iterative MRC V.S iterative V-BLAST (64QAM)

最後第三部分則是模擬在 3.1 節中所介紹的二種結合 V-BLAST 及 Viterbi 的偵測演算法。這部分的模擬和前二部分不同的地方是在傳送端使用多個迴旋編碼器，也就是在每根傳送天線上都有各自己的編碼器，這使得可以接收便端同時使用多個 Viterbi 解碼器幫助偵測，此部分的模擬結果包括了由圖 3-15 至圖 3-18。圖 3-15 中 MC-NOR 的情況是指在第 3.1.1 節中所介紹在接收端偵測時是由第一根天線依序往下偵測的演算法，CSI1 同樣是指假設前一級的決策是正確的情況所算出來的 CSI，CSI2 則是指使用 MMSE 的 CSI。我們可以發現在 Pre-ordering(POR)的演算法中，效能表現明顯較其它的 Soft 演算法來的好，且使用 CSI1 及 CSI2 幾乎沒有差別。

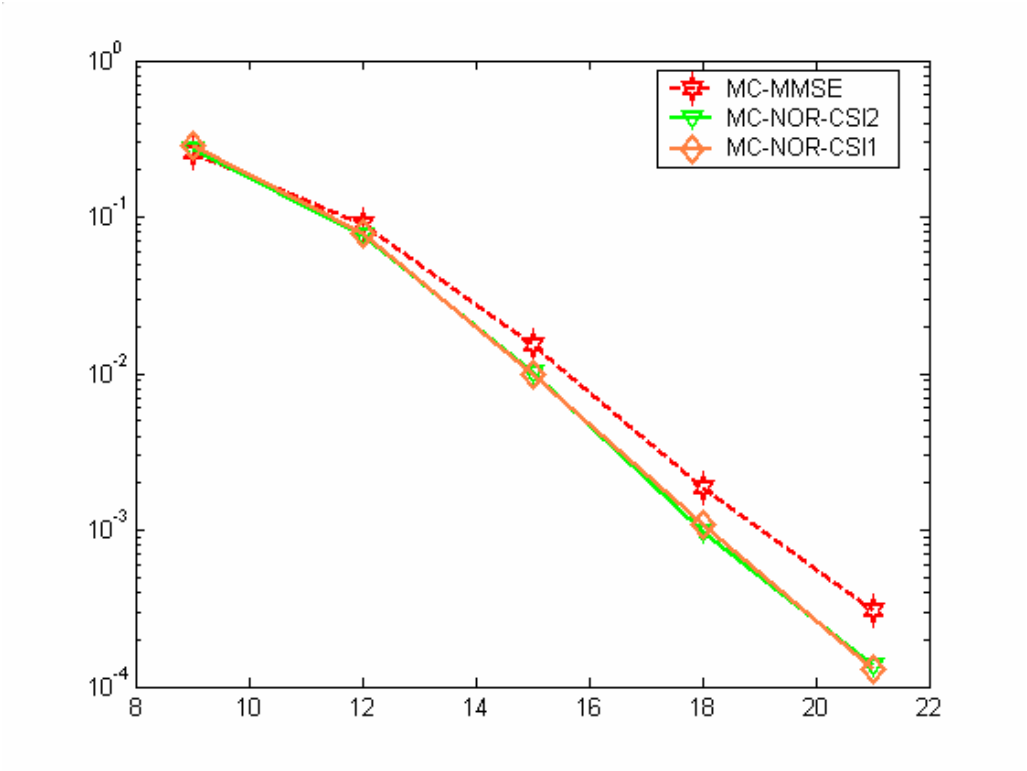

圖 3-15 NO ordering (64QAM)

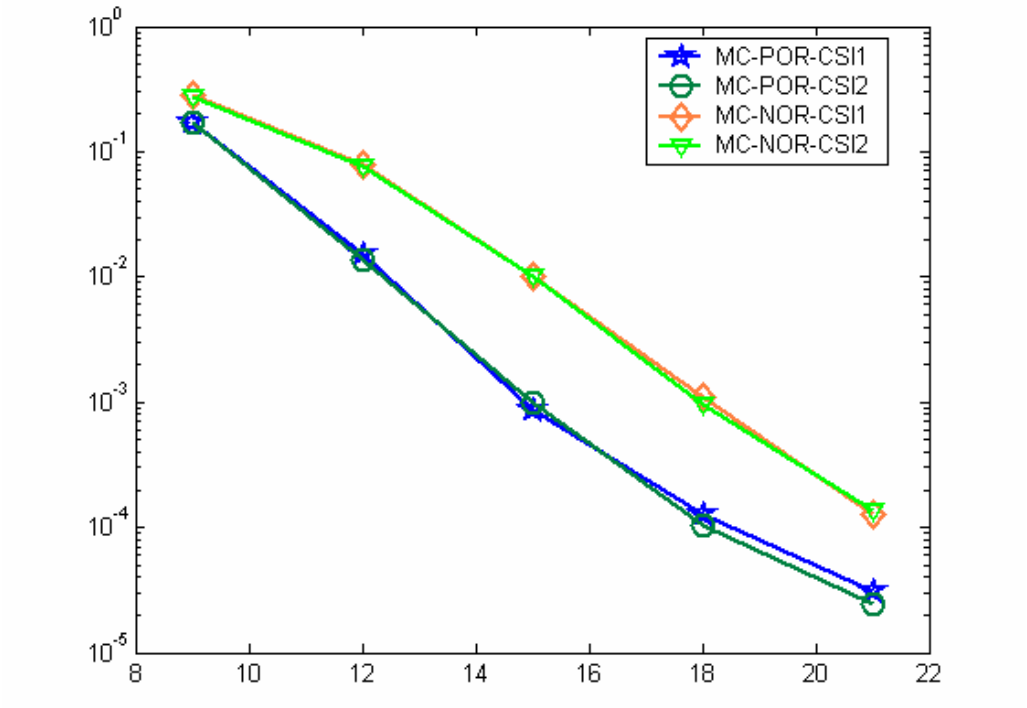

圖 3-16 Pre-ordering v.s No ordering (16QAM)

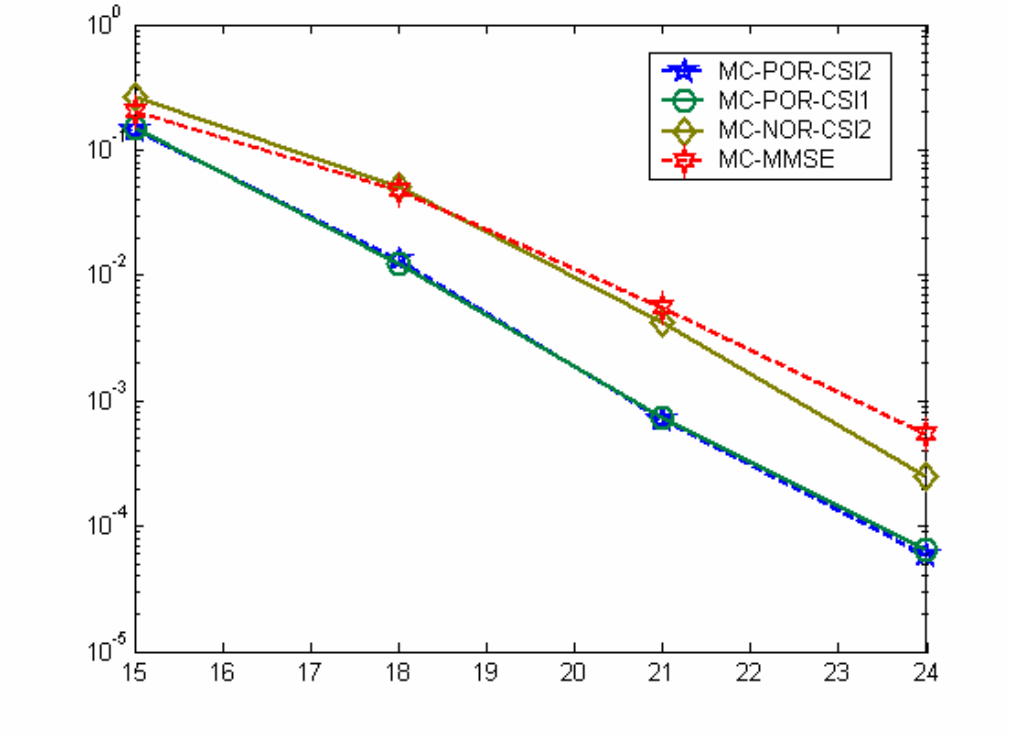

圖 3-17 Pre-ordering v.s In-ordering SIC for 64QAM

第4章硬體實現

4.1 設計流程

吾人使用電腦輔助工具來做模擬硬體實現，設計流程如圖 4-1 所示。設計流程主要分成軟體及硬體二部分，軟體使用 Matlab 模擬而硬體模擬使用 Modelsim 及 ISE 6 模擬。Matlab 模擬包括了浮點系統的模擬及定點(fixed-point)系統的模擬。浮點系統的模擬可以檢查各函數的正確性和設定參數，也可以評估系統的效能。而定點系統的模擬，用來決定各個訊號的所需要的表示位元數(word length)。各個信號表示位元數確定以後，便進入硬體實現模擬。在硬體實現模擬中，吾人使用 VHDL 作為硬體描述語言做程式編寫，且使用 Modelsim 作硬體的行為模擬來驗證 VHDL 程式碼的正確性，用 ISE 6 作為合成軟體。

硬體實現模擬流程中，首先使用硬體描述語言編輯(HDL Editor)模擬電路輸入。將電路輸入後，接下來可做行為模擬(Beaver simulation)，這可以快速的了解所設計電路正確與否，並做修改更正。在這一步驟時，會參照由 Matlab 定點系統模擬所產生的數據來幫助驗證函數功能正確。當硬體的行為模擬正確之後，便使用合成軟體 ISE 6 加入 VirtexE xcv-2000e 的 library 將 VHDL 程式轉為 RTL(register transfer level)，合成完之後，就可做時序模擬(Timing simulation)，模擬所設計電路在燒入 FPGA 晶片後，所造成的邏輯延遲及繞線延遲是否符合所需，在此便完成了所有的硬體模擬。

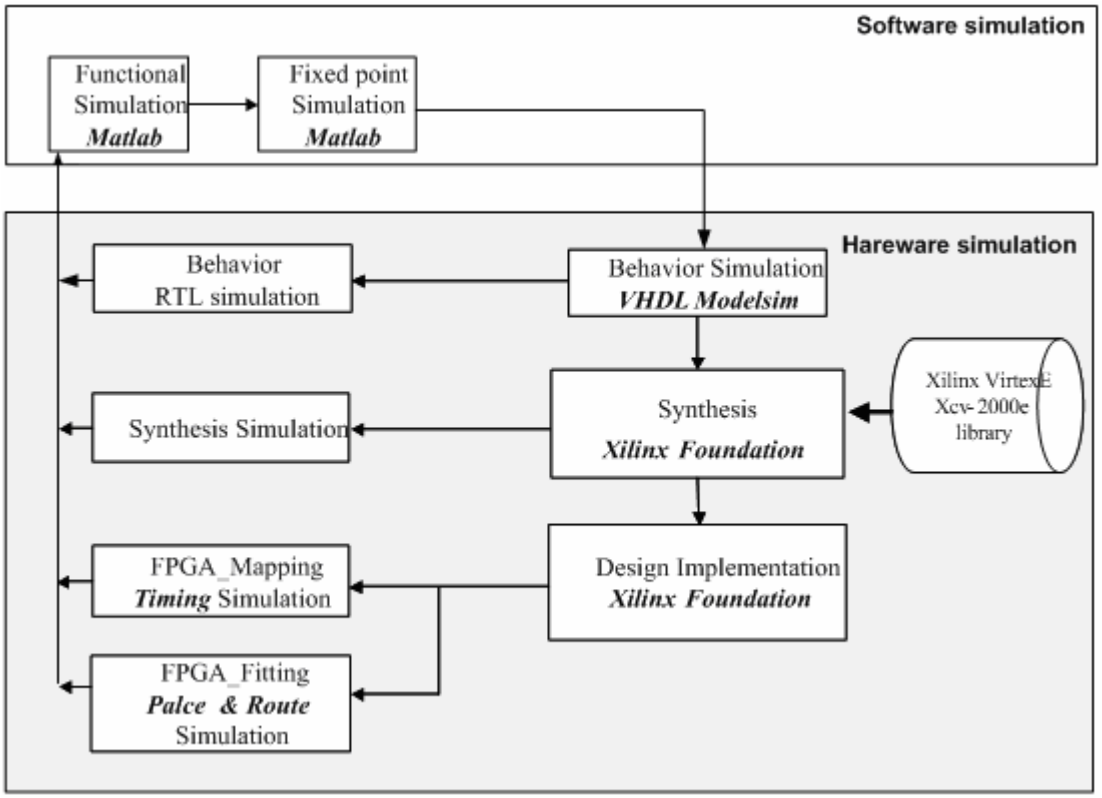

圖 4-1 硬體實現流程圖

4.2 組成元件介紹

吾人實作的接收機便如圖 4-2 所示，這是在 2.3.2 軟性輸入軟性輸出之MMSE接收機那一節中所介紹的接收機的簡化版，主要實作元件包括了最小均方差估測(MMSE)、軟性反對映(Soft demapping)、反交錯器(De-interleaver)及 Viterbi 解碼器。這樣的模式是在 11n 提案中基本的模式(2 根接收天線、20MHZ、編碼率 1/2、64QAM)，訊號流程大致和 2.3.2 中所推導的相同。底下則對這種情況下的訊號處理及在為了實作上的需求而做的修改做詳細介紹。在 4.2 中的安排中，會先對所要實作的接收機訊號流程做介紹，再分別介紹各元件如何實作。

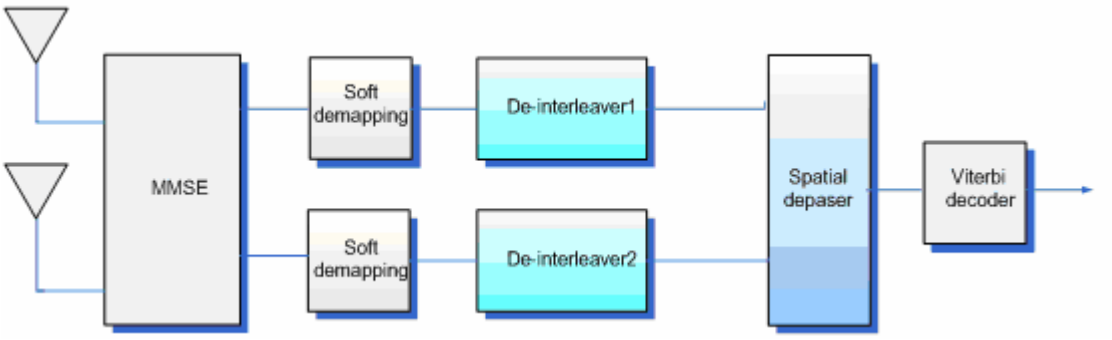

圖 4-2 實作之接收器方塊圖

4.2.1 接收機訊號流程

如同在 2.3.2 推導的，接收訊號在2×2 的情況下可以表示為:

$$Y_k = \begin{bmatrix} h_{1,k} & h_{2,k} \end{bmatrix} \begin{bmatrix} s_{1,k} \\ s_{2,k} \end{bmatrix} + N_k \tag{4-1}$$

$h_{i,k}$ 是 $s_{i,k}$ 的通道影響，也就是 H 的第 i 列。而 MMSE 的等化器則可以表示為

$$W = (H_k^H H_k + \frac{2}{\rho} I)^{-1} H_k^H = \begin{bmatrix} w_1^T \\ w_2^T \end{bmatrix} \tag{4-2}$$

ρ 是 SNR: $\frac{E_S}{N_0}$,則此等化過後的訊號即變成 $Z_k = WY_k$ ，假設 w_i^T 是 W 的第 i 行，則對第 i 根天線而言，等化後的訊號即可表示為:

$$Z_{i,k} = w_i^T h_{i,k} s_{i,k} + w_i^T h_{j \neq i,k} s_{j,k} + w_i^T N_k \tag{4-3}$$

我們假設 MMSE 等化效果很好，也就是假設 $w_i^T h_{i,k} \cong 1$ ，且 $w_i^T h_{j \neq i,k} s_{j,k} \cong 0$ ，如此我們便可以將(4-3)化簡為:

$$Z_{i,k} = s_{i,k} + \underbrace{w_i^T N_k}_{N_{eff}} \tag{4-4}$$

如此一來，$\sigma_{N_{eff}}^2 = \|w_i\|^2 N_0$ 且 $H_{eff}=1$，由等化過後的在第 i 根天線下第 k 個子載波的 QAM symbol($Z_{i,k}$)要反對映到位元($v_{k,i}^n$)時，代入(3-1):

$$v_{k,i}^n = \mathrm{CSI}_{\mathrm{k,i}} \times D(Z_{i,k}, n) \tag{4-5}$$

n 代表著第幾個位元，在吾人的實作中使用 64QAM，所以 $n=1\cdots6$ 。而 CSI 則可以代入其定義:

$$CSI_{k,i} = \frac{2\left|(H_{eff})_{k,i}\right|^2}{\sigma_{eff}^{\ 2}} = \frac{2}{\|w_i\|^2 N_0} \tag{4-6}$$

而 $D(Z_{i,k},n)$ 在 64QAM 如同在 2.3 節中推導可表示成:

$$D(Z_{i,k},n)=\begin{cases} R\{Z_{i,k}\} & n=1 \\ -\left|R\{Z_{i,k}\}\right|+4 & n=2 \\ -\left|\left|R\{Z_{i,k}\}\right|-4\right|+2 & n=3 \\ I\{Z_{i,k}\} & n=4 \\ -\left|I\{Z_{i,k}\}\right|+4 & n=5 \\ -\left|\left|I\{Z_{i,k}\}\right|-4\right|+2 & n=6 \end{cases} \tag{4-7}$$

$R\{Z_{i,k}\}$ 是指 $Z_{i,k}$ 的實部，而 $I\{Z_{i,k}\}$ 則是指 $Z_{i,k}$ 的虛部。再來位元訊號經過反交錯器及 Viterbi 解碼器後便完成了整個訊號處理。

我們整理上述的訊號處理流程，接收訊號首先會經過 MMSE 將混合在一起的訊號分開二個獨立處理的資料流，因此必須利用(4-2)算出由在頻域的通道係數所組成的 MMSE 等化矩陣，再來便是使用(4-6)算出由 MMSE 等化矩陣係數所組成的 CSI。因此在吾人的實作中，圖 4-2 中的 MMSE 方塊，即是先算出由 52 組 (11n 提案中所定義的一個 OFDM symbol 中的資料個數) 2×2 頻域通道係數所組成的 52 組 2×2 MMSE 等化器係數，及 52 組 2×1 的 CSI，然後將這 52 組等化器係數及 CSI 放在 RAM 中等待之後的接收訊號使用。而圖 4-2 中的 Soft Demapping

方塊便是接收由 MMSE 方塊算出的 CSI 及等化過後訊號再利用(4-5)及(4-7)算出位元訊號v。最後再經過反交錯器及解碼器便完成了整個接收機了。我們將於 4.2.2-4.2.4 中詳細介紹 MMSE 方塊、反交錯器及 Viterbi 解碼器是如何實作出來的。

4.2.2 最小均方差估測

我們令在第 k 個子載波上的通道矩陣係數H_k為:

$$H_k = \begin{bmatrix} h_{k,1} & h_{k,2} \\ h_{k,3} & h_{k,4} \end{bmatrix} \tag{4-8}$$

因為底下的推導都是在同樣的子載波上，為了表示方便，我們令$h_{k,m} = h_m$，則通道矩陣可以改寫為:

$$H_k = \begin{bmatrix} h_1 & h_2 \\ h_3 & h_4 \end{bmatrix}$$

且令$\alpha = \frac{2}{\rho}$，如此由(4-3)，所要求的 MMSE 等化器的矩陣係數則可寫成:

$$\begin{aligned} W &= \begin{bmatrix} w_1^T \\ w_2^T \end{bmatrix} \\ &= (H_k^H H_k + \frac{2}{\rho} I)^{-1} H_k^H \\ &= \left(\begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix} \begin{bmatrix} h_1 & h_2 \\ h_3 & h_4 \end{bmatrix} + \alpha \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix} \\ &= \begin{pmatrix} h_1^* h_1 + h_3^* h_3 + \alpha & h_1^* h_2 + h_3^* h_4 \\ h_2^* h_1 + h_4^* h_3 & h_2^* h_2 + h_4^* h_4 + \alpha \end{pmatrix}^{-1} \begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix} \end{aligned} \tag{4-9}$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \frac{1}{AD - BC} \begin{bmatrix} D & -B \\ -C & A \end{bmatrix} \tag{4-10}$$

利用(4-10) 2×2 的反矩陣公式則(4-9)可改寫成(4-11)及(4-12)

$$W=\begin{pmatrix} h_1^*h_1+h_3^*h_3+\alpha & h_1^*h_2+h_3^*h_4 \\ h_2^*h_1+h_4^*h_3 & h_2^*h_2+h_4^*h_4+\alpha \end{pmatrix}^{-1}\begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix}$$
$$=\frac{1}{g}\begin{bmatrix} h_2^*h_2+h_4^*h_4+\alpha & -(h_1^*h_2+h_3^*h_4) \\ -(h_2^*h_1+h_4^*h_3) & h_1^*h_1+h_3^*h_3+\alpha \end{bmatrix}\begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix}$$
$$=\frac{1}{g}\left(\begin{bmatrix} h_1^*(h_4^*h_4)-h_4(h_2^*h_3^*) & h_3^*(h_2^*h_2)-h_2(h_1^*h_4^*) \\ h_2^*(h_3^*h_3)-h_3(h_4^*h_1^*) & h_4^*(h_1^*h_1)-h_1(h_2^*h_3^*) \end{bmatrix}+\alpha\begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix}\right)$$

(4-11)

$$g=(h_1^*h_1)(h_4^*h_4)+(h_2^*h_2)(h_3^*h_3)-(h_1^*h_4^*)(h_2h_3)-(h_2^*h_3^*)(h_1h_4)+\alpha(h_1^*h_1+h_2^*h_2+h_3^*h_3+h_4^*h_4)+\alpha^2$$

(4-12)

然而在(4-11)中，W 要算出來需要四個複數除法，因此我們讓 MMSE 矩陣係數整個放大 g 倍，而避免除法:

$$W'=\begin{bmatrix} w'^T_1 \\ w'^T_2 \end{bmatrix}=gW=\begin{bmatrix} h_1^*(h_4^*h_4)-h_4(h_2^*h_3^*) & h_3^*(h_2^*h_2)-h_2(h_1^*h_4^*) \\ h_2^*(h_3^*h_3)-h_3(h_4^*h_1^*) & h_4^*(h_1^*h_1)-h_1(h_2^*h_3^*) \end{bmatrix}+\alpha\begin{bmatrix} h_1^* & h_3^* \\ h_2^* & h_4^* \end{bmatrix}$$

(4-13)

使用 W' 取代 W 會使得訊號經過 MMSE 等化過後放大 g 倍，然而我們發現之後的軟性反對映中，由只需要修改(4-7)為(4-14)便不影響其機率值。

$$D(Z_{i,k},n)=\begin{cases} R\{Z_{i,k}\} & n=1 \\ -\left|R\{Z_{i,k}\}\right|+4g & n=2 \\ -\left|\left|R\{Z_{i,k}\}\right|-4g\right|+2g & n=3 \\ I\{Z_{i,k}\} & n=4 \\ -\left|I\{Z_{i,k}\}\right|+4g & n=5 \\ -\left|\left|I\{Z_{i,k}\}\right|-4g\right|+2g & n=6 \end{cases}$$

(4-14)

由(4-13)算出放大的 MMSE 矩陣後，則第一根天線的 CSI_1 及第一根天線的 CSI_2 便可由(4-6)算出:

$$CSI_1 = \frac{\left|(H_{eff})_1\right|^2}{\sigma_{eff}^{\ 2}} = \frac{2}{\left\|w'_1\right\|^2 N_0}$$

$$CSI_2 = \frac{\left|(H_{eff})_2\right|^2}{\sigma_{eff}^{\ 2}} = \frac{2}{\left\|w'_2\right\|^2 N_0}$$

將 CSI 中的 2 及 N_0 正規化後，我們可以得最後 CSI 的計算方式:

$$CSI_1 = \frac{1}{w'^{\ 2}_{11} + w'^{\ 2}_{12}}$$

$$CSI_2 = \frac{1}{w'^{\ 2}_{21} + w'^{\ 2}_{22}} \qquad (4\text{-}15)$$

如果 $W' = \begin{bmatrix} w'_{11} & w'_{12} \\ w'_{21} & w'_{22} \end{bmatrix}$ 。

MMSE 方塊圖如圖 4-3 所示: 在 MMSE 這個方塊中，一開始由頻域上通道係數計算 52 組的 W'、CSI 及 g 並將這些值存在 RAM 中，而之後資料訊號 Y 進來時便從 RAM 中讀出 W' 由 $Z' = W'Y$ 做矩陣相乘而可得到 Z' 。而 Z' 便可進入軟性反對映了。值得注意的是，MMSE 這樣子的計算複雜度實在太高，於是在實現時吾人提高系統的時脈變為原來的四倍，共用乘法器而能有效減小面積。

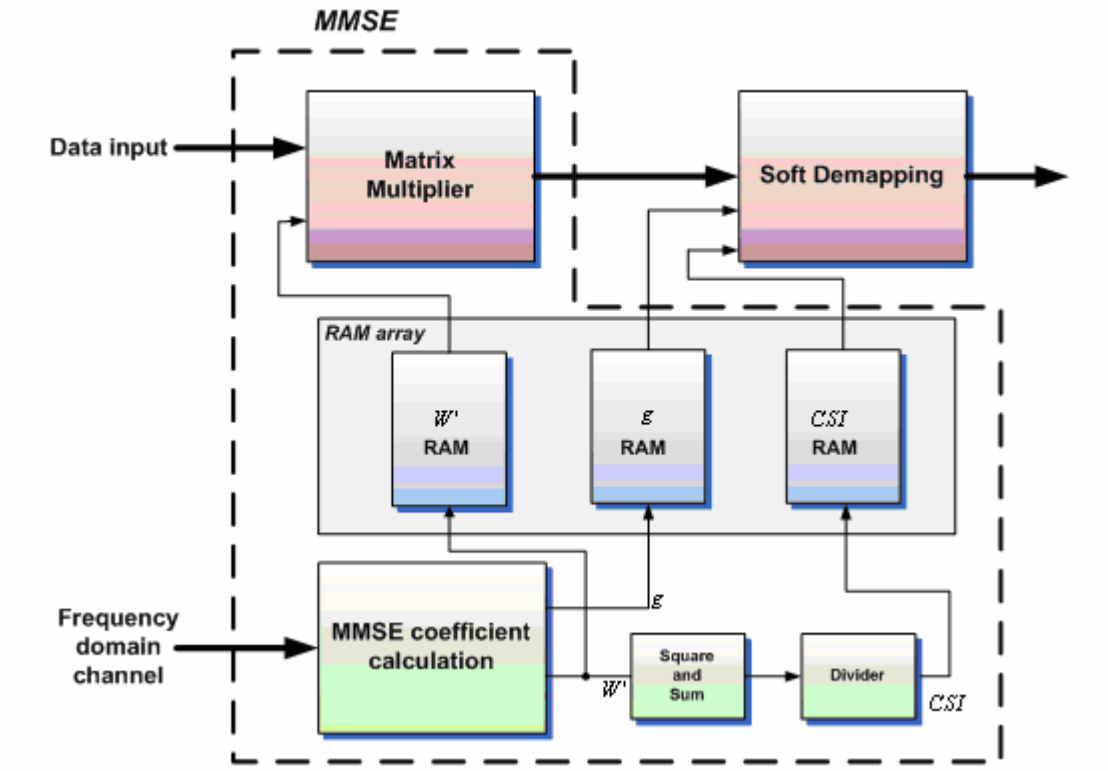

圖 4-3 MMSE 結構方塊圖

4.2.4 反交錯器

如同 2.4.2.3 中所描述，反交錯器是由(2-36)至(2-34)所定義的三次對調所組成。假設在第一次對調之前資料的索引(index)為 r ，而在第一次對調之後第二次對調之前的索引為 j ，i 則為第二次對調及第三次對調之間的索引，k 是第三次對調之後 QAM 的反對映之前的索引。三次對調如下所示:

$$j=\left(r+\left(\left(2\times i_{ss}\right)\bmod 3+3\times \text{floor}(i_{ss}/3)\right)\times N_{rot}\times N_{BPSC}\right)\bmod N_{CBPS},\qquad r=0,1,\ldots,N_{CBPS}-1$$

$$i=s\times\text{floor}\left(j/s\right)+\left(j+\text{floor}\left(N_{column}\times j/N_{CBPS}\right)\right)\bmod s,\qquad j=0,1,\ldots,N_{CBPS}-1$$

$$k=N_{column}\times i-\left(N_{CBPS}-1\right)\text{floor}\left(i/N_{row}\right),\qquad i=0,1,\ldots,N_{CBPS}-1$$

我們使用二塊 RAM 來完成上述的對調。然而在吾人的實作中，使用 64QAM 及 2 根接收天線及 20MHZ 所以上述三次對調所需要的參數便可以決定為:

$$N_{BPSC} = 6$$

$$N_{CBPS} = 52 \times N_{BPSC} = 312$$

$$s = \max(\frac{N_{BPSC}}{2}, 1) = 3$$

$$N_{row} = 13$$

$$N_{colmn} = 4 \times N_{BPSC} = 24$$

$$N_{rot} = 11$$

如此，我們便可以將二根天線個別經過三次對調的索引值存在 ROM 中，當作所使用的 RAM 的位址。如圖 4-4 所示。

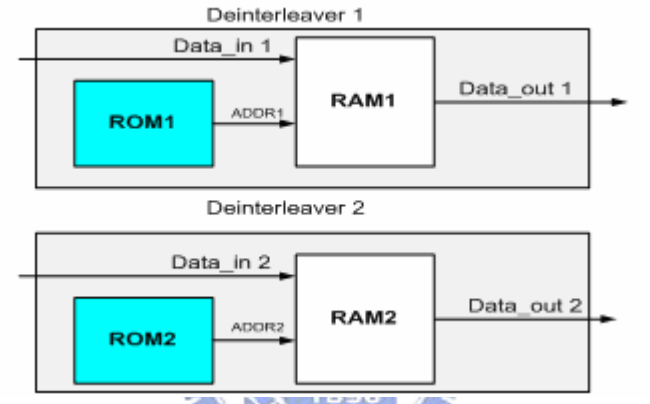

圖 4-4 反交錯器結構方塊圖

4.2.5 Viterbi 解碼器

在吾人的實作當中，吾人選擇了 Viterbi 演算法當作在 802.11n 中所使用的迴旋編碼器的解碼器。一般而言，為了達到模組化及管線式處理，可以將解碼器的電路分成四個主要的處理單元，如圖 4-5 所示。其中包括:分支計量值單元 BMU(Branch Metric Unit ;BMU)、相加-比較-選擇單元(add-Compare-Select Unit ; ACSU)、路徑計量值記憶單元(Path Metric Memory Unit ; PMMU)及存活路徑記憶單元(Survivor Memory Unit ; SMU)。各單元的功能簡述如下:

I. 分支計量值單元(BMU):根據所收到的值計算在格狀圖(trellis)每一級的分支計量值。

II. 相加-比較-選擇單元(ACSU):對每個狀態執行相加-比較-選擇運算，用來計算所有狀態的存活計量值和產生決策位元(或存活路徑)

III. 路徑計量值記憶單元(PMMU):用來儲存存活計量值，以供下一級在 ACSU 中計算路徑計量值及存活計量值。

IV. 存活計憶單元(SMU):負責記錄 ACSU 所產生的決定位元，並找出存活路徑中最早時間點的位元，而當作解碼輸出。

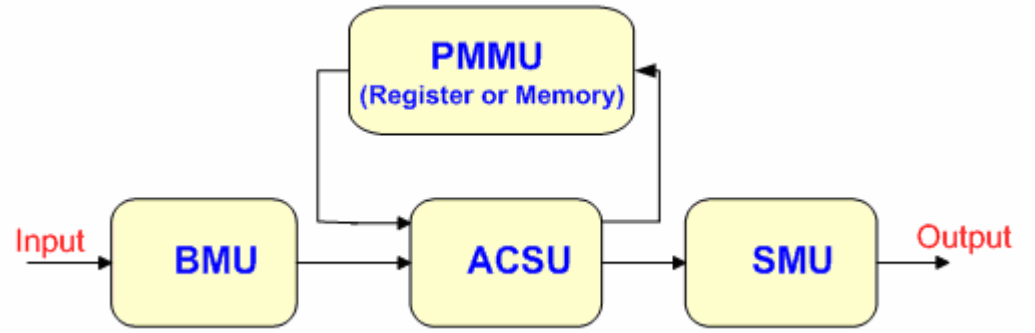

圖 4-5 Viterbi 解碼器方塊圖

在本節的安排中，我們將先對 Viterbi 的原理做介紹，再將上述的四個單元一一做較詳細的介紹。

4.2.5.1 Viterbi 演算法

Viterbi 演算法是在 1967 年由 Viterbi 所提出來的[23]來對迴旋碼進行解碼的，近年來，Viterbi 演算法則有非常廣泛的應用，例如數位通訊系統、語音辨識等。此演算法為最大近似解碼(Maximum Likelihood Decoding ; MLD)，簡單地說便是依據接收訊號來尋找在格狀圖上所有可能的路徑，最後選擇一條最短路徑來當作最佳解碼路徑。演算法中包括了計算各個路徑與所收到訊號之間的相似度，並且去除掉一些不可能成為最佳選擇的路徑。當幾個路徑進入同一狀態時，在這幾個路徑之間的其中一個會因有「適當」的計量值(metric)而被選擇，而這被選擇的路徑也就是存活路徑(survivor path)。所謂的「適當」則要根據計量值的定義。在格狀圖中的所有狀態都要進行這樣的存活路徑選擇。在接下來的時間也是根據這樣的法則來去除不可能的路徑，如此，在收集到一定長度的訊號後便可以將解碼位元輸出了。底下我們用一個例子來說明這演算法。

為了描述 Viterbi 演算法是如何動作之前，我必需先定義出格狀圖，這是根據編碼器而產生的。如圖 4-6 所示，我們用一個簡單的$(n,k,m)=(2,1,2)$的迴旋碼編碼器來舉例說明，它由 2 個移位暫存器，2 個模 2(Modulo-2)加法器所組成，而模 2(Modulo-2)的加法器可以由 XOR 閘來實現。迴旋碼編碼器基本上是一個有限狀態機(Finite state Machines)，我們可以用狀態圖來描述其輸入輸出的關係，由之編碼器可以建立如圖 4-7 所示的狀態圖。其中每一狀態點為編碼器內暫存器之值，輸入與輸出位元則表為$v^{(1)}v^{(2)}/u$。通常我們會將這狀態圖轉成格狀圖來說明，因為對於時間與狀態之間的轉換有較好的表達能力。如圖 4-8 所示，實線代表輸入 0，虛線代表輸入 1。對圖 4-8 來說，因為編碼器有二個移位暫位器，所以共會有 4 個狀態，分別為 $S_0(0,0)$，$S_1(1,0)$，$S_2(0,1)$，$S_3(1,1)$，每一個狀態有 2 條路徑指向下一個狀態，一條為當輸入位元 0 時，另一條則是輸入位元為 1 時。

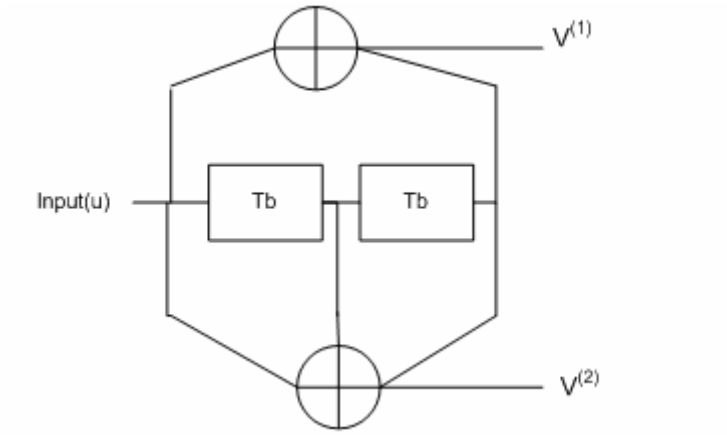

圖 4-6 (2,1,2)之迴旋碼編碼器

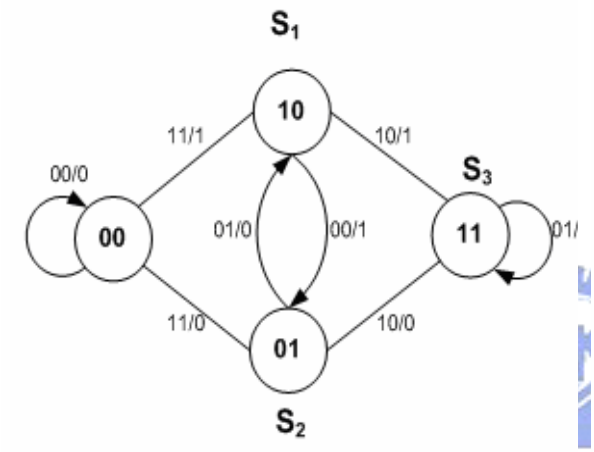

圖 4-7 (2,1,2)迴旋碼編碼器之狀態圖

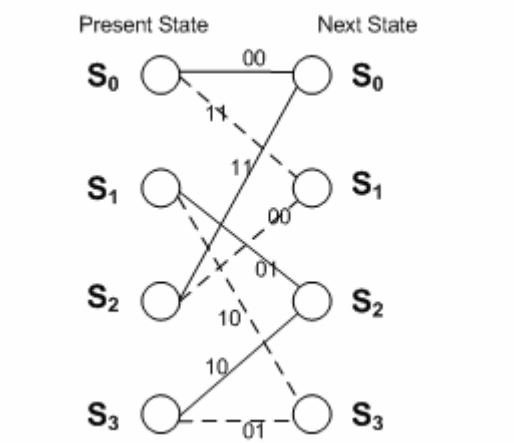

圖 4-8 (2,1,2) 迴旋碼編碼器之格狀圖

為了說明方便，我們使用 Hard Decision 的 Viterbi 來說明 Viterbi 是如何運作的，然而在吾人的實作中是使用 Soft Decision 的 Viterbi，而二者的差別只需將底下所使用的漢明距離改為歐幾里德(Euclidean)距離即可。在說明 Viterbi 演算法之前，我們先定義一些名詞:

- 碼句(code word) $Cw(i, j)$:由狀態 S_i 轉移到狀態 S_j 所對應的編碼器輸出。
- 分支計量值(Branch metric) $Bm(i, j, t)$:在時間 t，接收到的訊號($r(t)$)和碼句之間的相似度(Likelihood)，可用漢明距離表示。
- 路徑計量值(Path metric) $Pm(i, j, t)$:在時間 t，接收到的碼和由狀態 S_i 轉

移到狀態 S_j 之間的相似度。

- 存活計量值(Survivor metric) $Sm(j,t)$:在時間 t，進入狀態 S_j 的最小路徑計量值。

$Cw(i,j)$ 、 $Bm(i,j,t)$ 、 $Pm(i,j,t)$ 、 $Sm(j,t)$ 之間可表示如下:

$$Bm(i,j,t) = Cw(i,j) ** r(t)$$

$$Pm(i,j,t) = Bm(i,j,t) + Sm(j,t-1) \quad (4\text{-}16)$$

where ** is hamming distance calculation

假設有一全零資料序列 u =(0000000)，經過編碼後，我們可以得到編碼後的輸出序列為 v =(00,00,00,00,00,00,00)，而 v 序列在經過通道的輸輸過程中受到雜訊的干擾，使得第 1 及第 3 位元由 0 變成 1，而解碼器所收到的序列為 (10,00,00,00,10,00,00)，經由以下的 Viterbi 演算法，可求得解碼之結果為 $\hat{u}$ =(0000000)= u 。

I. 如圖 4-9 所示，在 t = 0 時，從狀態 S_0 開始，其存活計量值為 0。若輸入 0，則路徑向上，在 t = 1 時，狀態 S_0 輸出 00。若輸入 1，則路徑向下，在 t = 1 時，狀態點 S_1 ，輸出為 11。若輸出位元與 r 之所對應的位元相同時，則我們給此位元之計量值為 0，反之為 1。因此，我們可以算出分支計量值 $Bm(0,0,1)=1$ 及 $Bm(0,1,1)=1$。因為在 t = 1 時，每一狀態只有一條路徑進入，所以存活計量值 $Sm(0,1)=1$ 、 $Sm(1,1)=1$ 。

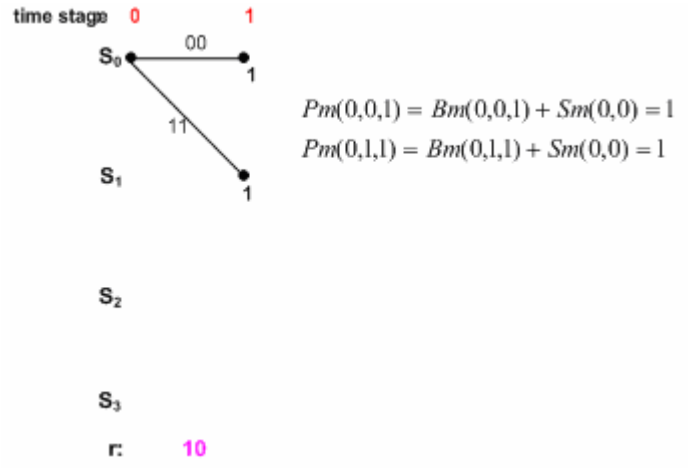

圖 4-9 Viterbi 演算法步驟 I

II. 重覆 I。如圖 4-10 所示，在 $t = 3$ 時，每一狀態都有二條路徑進入，我們選擇路徑較小的為存活計量值，此一路徑稱為存活路徑，同時把另一較大的路徑刪除。若兩路徑計量值相同時，則任取一條路徑為存活路徑。因此，S_0、S_0、S_0、S_3 的存活計量值 $Sm(0,3)$、$Sm(1,3)$、$Sm(2,3)$、$Sm(3,3)$ 分別為 1、2、3、3。

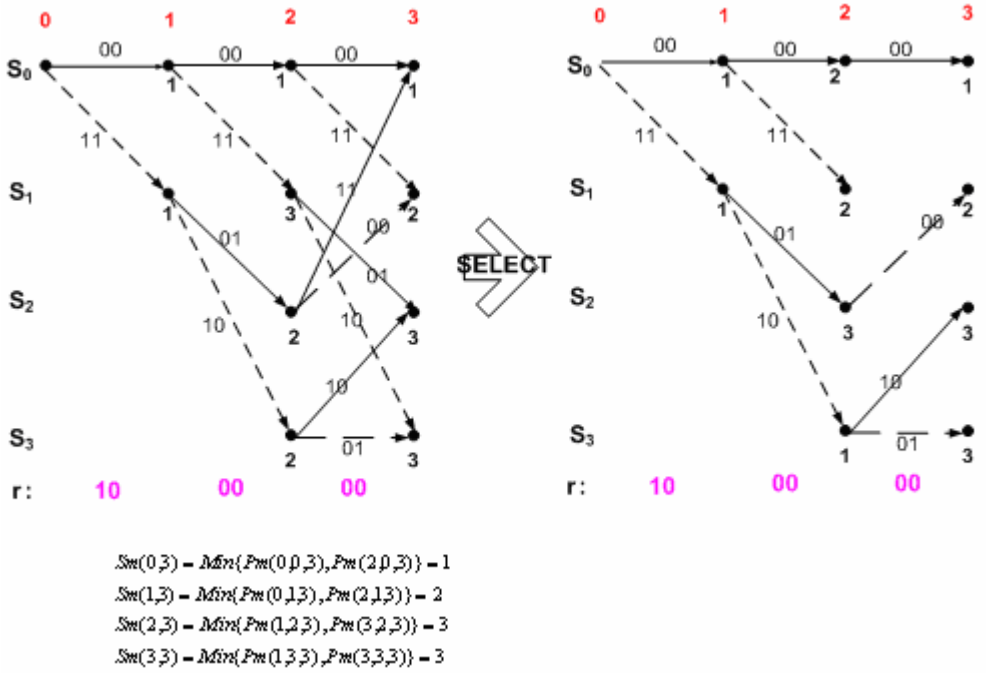

圖 4-10 Viterbi 演算法步驟 II

重覆 II，經過時間 $t = 4 \sim 7$。如

III. 圖 4-11 所示，在最後 $t = 7$ 時，我們挑選有存活計量值的狀態所走的路徑為

存活路徑，由存活路徑，也就是在圖中線條較粗者的，得到解碼之結果為 $\hat{u}$=(0000000)= u ，在此路徑中，實線為 0 而虛線為 1。

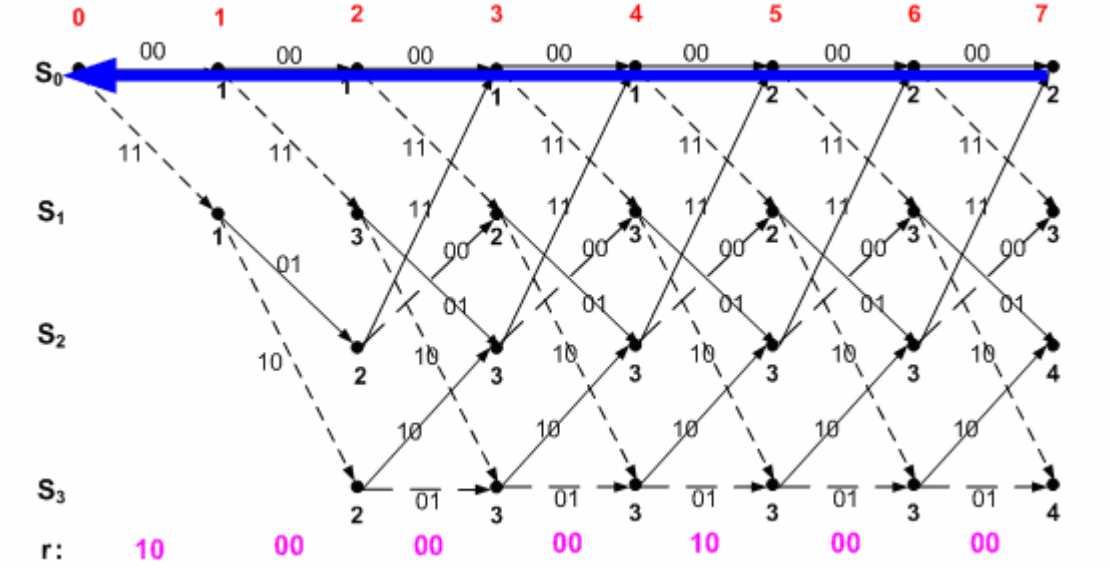

圖 4-11 Viterbi 演算法步驟 III

最後 Vitebi 演算法總結如下:

I. 從單元時間 t=1 開始，計算進入每一狀態之單一路徑的路徑計量值(path metric) $Pm(i, j, t)$ ，並記錄每一狀態之存活路徑和存活計量值(survivor metric) $Sm(i, j, t)$ 。

II. t 增加 1，對於每一個狀態中，比較進入此狀態的路徑，且將含有最小路徑計量值的路徑和其值記錄起來，而這個記錄起來的計量值便是存活計量值，而其它路徑便給予刪除。

III. 如果 $t<L+m$，則重覆步驟 2，否則停止。其中 L 為資料的長度，m 為編碼器中移位暫存器的個數。最後，比較在所有狀態的存活路徑計量值，有最小計量值的存活路徑便可以當作解碼路徑。

在 Viterbi 演算法中，我們必須將所有狀態及其對應的存活路徑儲存起來，若路徑很長(也就是訊號序列長)，則需要一個很大的記憶體。因此，我們必須在

路徑到達一定長度時加以截斷。當我們要截斷最前面的路徑時，必須要將其資料取出(解碼)。一般而言，若路徑之截斷長度(也就是被截斷存活路徑的長度)為 5.8*m* 時，不管任何一種路徑取出資料，所增加的解碼錯誤率是可以忽略的，也就是說，其最後面的一個分支都會重合成一條，如圖 4-12 所示。

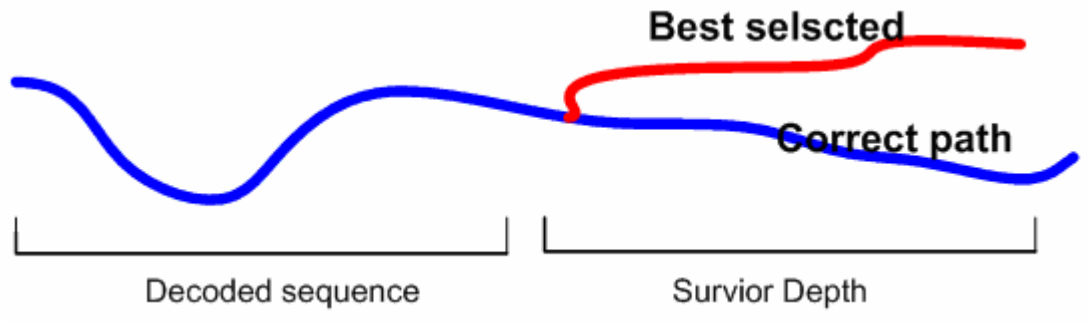

圖 4-12 Viterbi 演算法之截斷長度示意圖

4.2.5.2 分支計量值單元

分支計量值單元(BMU)，是根據所收到的值計算在格狀圖(trellis)每一級的分支計量值，也就是在計算在格狀圖上任二個節點之間的計量值。舉例說明，在圖 4-11 中，由節點 S_1 走到節點 S_2 的路徑是 01，所以假設我們收到的訊號是 *r(t)*=01，則此二節點的計量值便是 0，如果 *r(t)*=11，則此二節點的計量值便是 1。而在這個單元中的目的便是要計算出任二節點之間的計量值，而在圖 4-11 中，任二節點之間的路徑只有 00、01、10、11 四種，所以只要計算所收到的訊號跟 00、01、10、11 之間的距離，而任二個節點之間的計量值，便是這四個值的其中一個。在 VLSI 中，BMU 主要有二個實現的架構: (1)由記憶體產生，也就是說 BMU 是以查表的方式由 ROM 或 RAM 實現，而節省硬體面積。(2)由運算單元產生，將接收訊號直接運算出在格狀圖中所需的計量值，而這種方法的硬體複雜度則根據計量值的定義。在吾人的實作中是以絕對值，做為計算計量值的方法，且採用(2)由運算單元產生計量值的架構。圖 4-13 便是吾人使用的 BMU 架構。

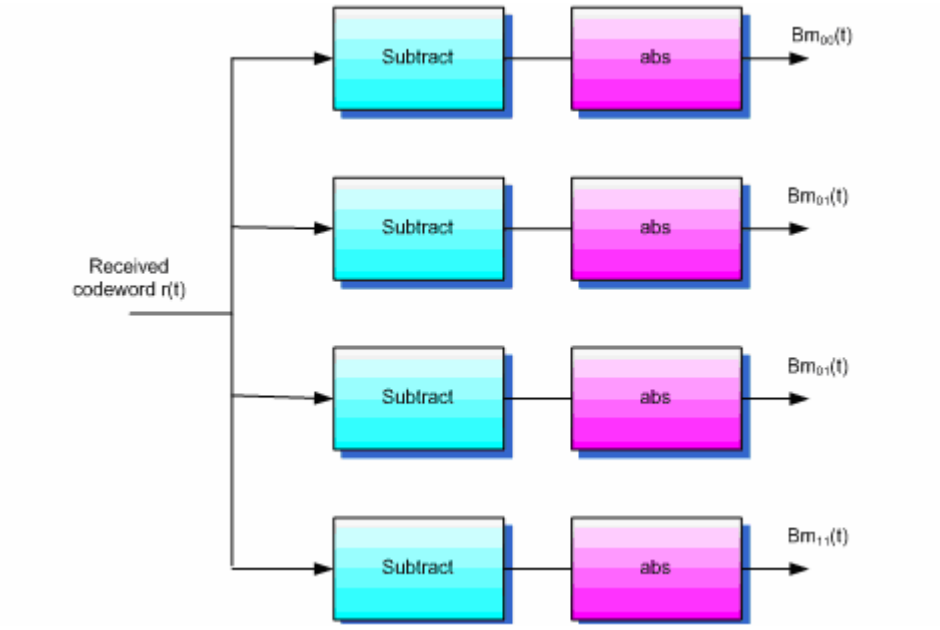

圖 4-13 BMG 方塊圖

4.2.5.3 相加-比較-選擇單元(ACSU)

ACSU 是整個 Viterbi 解碼器的核心，在高速傳輸的應用中(如 WLAN)，平行架構的 ACSU 則是必要的，因此在格狀圖的每一節點，便代表著一個 ACS 處理單元。

圖 4-14 為 ACSU 架構的例子，這是在 4.2.5.1 中所用使(2,1,2)的架構。這 ACS 處理單元有四個輸入(2 個分支計量值及 2 個存活計量值)和 2 個輸出(下一級的存活計量值及決定位元)。我們先由格狀圖畫分出有幾個蝴蝶模組，且決定各蝴蝶模組的輸入輸出關係，而一個蝴蝶模組是由二個 ACS 所組成。在 ACS 中，其主要功能有二:

1. 在路徑計量值中選擇最小的當作存活路徑值，並儲存之。
2. 產生存活路徑之決定位元，並送至存活路徑記憶單元(SMU)。

決定位元是在 ACS 中所產生最重要的資訊。它暗示了下一級的存活計量值是由哪一組存活計量值及分支計量值之和所產生的。假設決定位元為 0，代表著在 ACS 中上面那條是被選擇為下一級的存活路徑。反之決定位元為 1 的話，存活路徑便是挑下面那條。代表著所有節點的 ACS 所產生的決定位元，將會被送

到 SMU 中做解碼而產生出最後的輸出。

然而在吾人的實作中，是使用 6 個移位暫存器的編碼器，這代表了格狀圖會有 64 個狀態節點，便有 32 個蝴蝶模組、64 個 ACS，蝴蝶模組的目的為指示 ACS 應該抓取怎樣的路徑計量值及分支計量值，其架構如圖 4-15 所示。

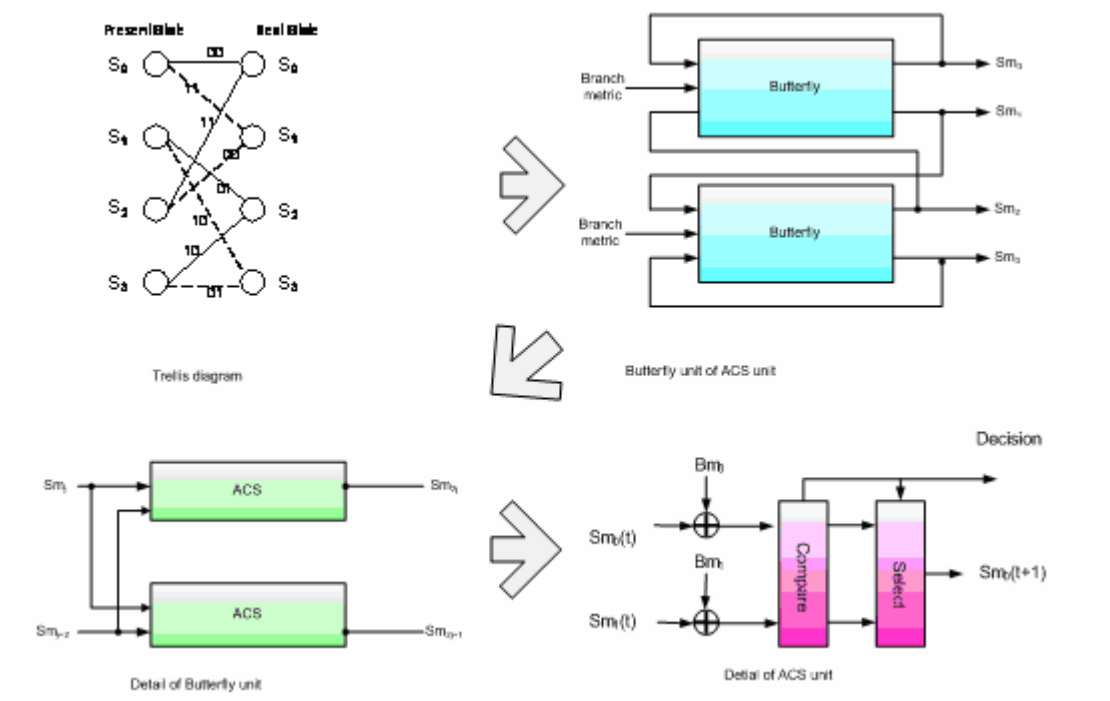

圖 4-14 ACS 設計流程圖

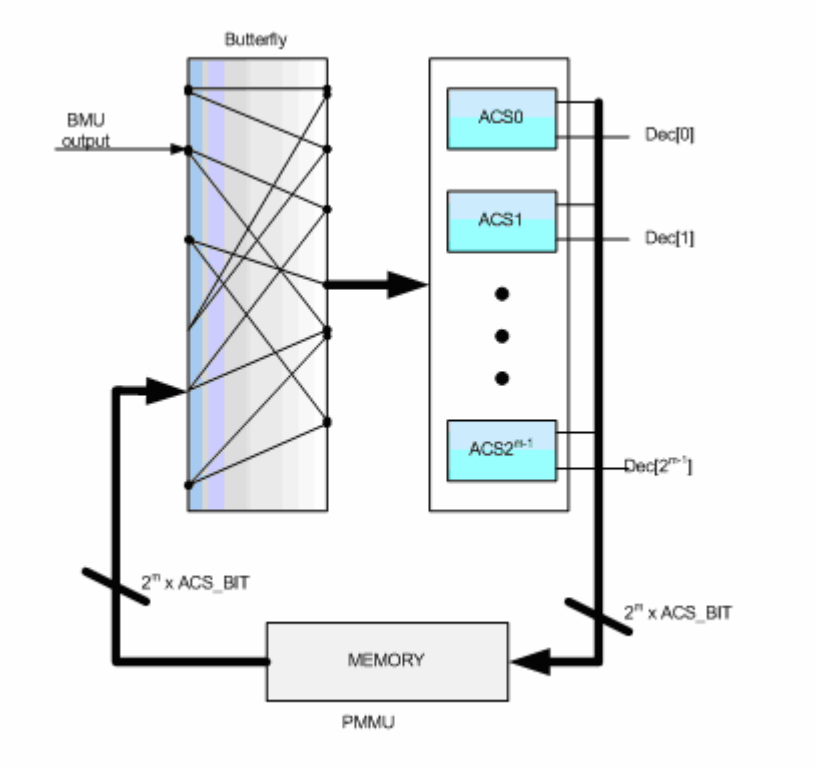

圖 4-15 平行 ACS 架構方塊圖

然而當資料不斷的進來，因為存活計量值會一直累加，為避免溢位(overflow)所造成的錯誤，我們必須對存活計量值做正規化，而正規化的方法有很多，最直覺的想法是將所有狀態之存活計量值同減去一個固定的值，但這造成了額外的硬體面積，於是在吾人的實作中採用[24]的修改過的模正規化(Modified Modulo Normalization)，其架構如圖 4-16 所示。

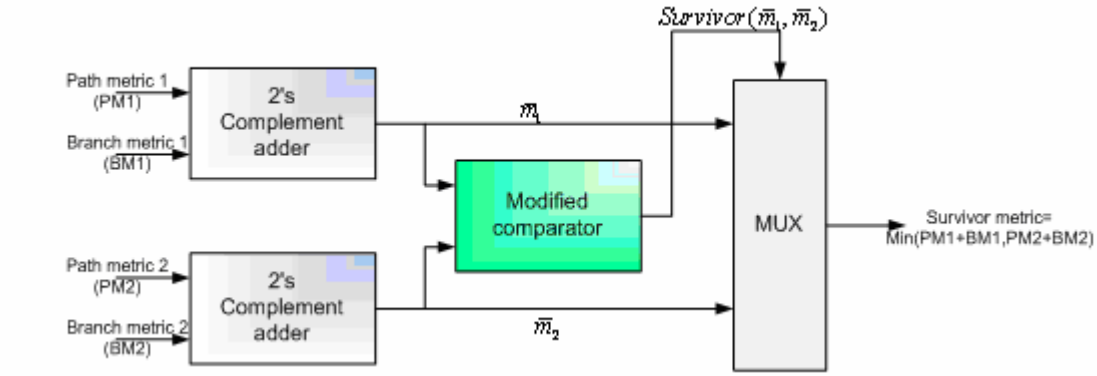

圖 4-16 模正規化架構方塊圖

在模(modulo)的技術中，所有的計量值(metric) m_j 都要依(4-17)被轉換為正規化過的計量值 $\bar{m}_j$ 。

$$\bar{m}_j = (m_j + \frac{C}{2})(\text{mod } C) - \frac{C}{2} \tag{4-17}$$

C 表示由路徑計量值所能表示的最大值。而比較法則決定了要給 SMU 的存活位元的值。這法則表示如下:

$$Survivor(\bar{m}_1, \bar{m}_2) = \bar{m}_{1p} \oplus \bar{m}_{2p} \oplus y(\bar{m}_1, \bar{m}_2) \tag{4-18}$$

$y(\bar{m}_1, \bar{m}_2)$ 表示 $\bar{m}_1$ 及 $\bar{m}_2$ 的非負(unsigned)比較。$\bar{m}_{1p}$ 及 $\bar{m}_{2p}$ 分別表示 $\bar{m}_1$ 及 $\bar{m}_2$ 的最高位元。也就是說 $Survivor(\bar{m}_1, \bar{m}_2) = y(\hat{m}_1, \hat{m}_2)$ 當 $\bar{m}_1$ 及 $\bar{m}_2$ 有同樣的正負號。

4.2.5.4 存活記憶單元

存活記憶單元(SMU)的目的最主要就是記錄從 ACS 中送來的決定位元或存活路徑。再根據這些資料進行解碼。這是在 Viterbi 解碼器中，最具有討論設計空間的地方。現存有二種著名的 SMU 做法，分別是暫存器交換方法[25](register-exchange method ; REM)及後向追溯方法[26]-[28](trace back method ; TBM)。暫存器交換方法有著架構簡單、快速的優點，但使用的暫存器所增加的面積卻是在設計中最需考量的地方。然而在另一方面後向追溯方法使用記憶體來儲存由 ACS 送來的存活路徑，而使得所須的面積較暫存器交換方法小，但有著較長的延遲及較複雜的控制電路，且所需要的功率(power)也較大。在吾人的實作中，綜合考量高系統時脈(clock)及所需面積大小，吾人採用在[28]中所提出的以記憶體實作 TBM 的其中一種，稱為“k pointer even”。底下則介紹這種方法。

首先我們先介紹在 TBM 中一定會被定義的三種動作:

1. 寫入(WR):從 ACSU 中所送過來的存活路徑將會被會寫入與狀態互相對應的記憶體中，比如在吾人的實作中，ACSU 共有 64 個 ACS，代表著 64 狀態，

相對的也有 64 組存活路徑產生而需要被記錄。

2. 回溯讀取(Trace-back Read ; TB):TB 這動作主要是將讀取寫在記憶體中的存活路徑，將讀出的值當作一個指標，而這指標指向上個狀態，而能找到上一個狀態數。這動作的目的為希望能讓所有路徑重合。
3. 解碼讀取(Decode Read ; DC):這動作和 TB 相同，同樣讀出一個指標而找到上個狀態。但是這個動作的對像是更舊的資料，而且是從已經作過完整的 TB 之後所找到的狀態數開始做 DC 這動作。在 DC 中，讀出的指標值除了指向上一個狀態數之外，在搭配上現在的狀態數，便可以得到當初輸入而解碼輸出了。

k=3 的 k point even 方法如圖 4-17 所示。在這方法中使用了 *k* 個讀取指標(包括回溯讀取指標及解碼讀取指標)，由 2*k* 塊大小為 M/2 的記憶體所組成，*M* 是指截斷長度。寫入的指標是從左到右的，而讀取指標是由右而左的。在一個時間點中，六塊記憶體中有一個是執行寫入、一個解碼讀取、二個回溯讀取及二個閒置。第一個位元輸出是在當寫入指標寫到整個記憶體的最後時，所以全部的延遲為 3*M*。然而在因為解碼讀取是由右而左的，所以位元是以一個倒反的順序輸出的，這需要另一塊記憶體當作 LIFO(Left In First Out)而讓輸出順序正確。

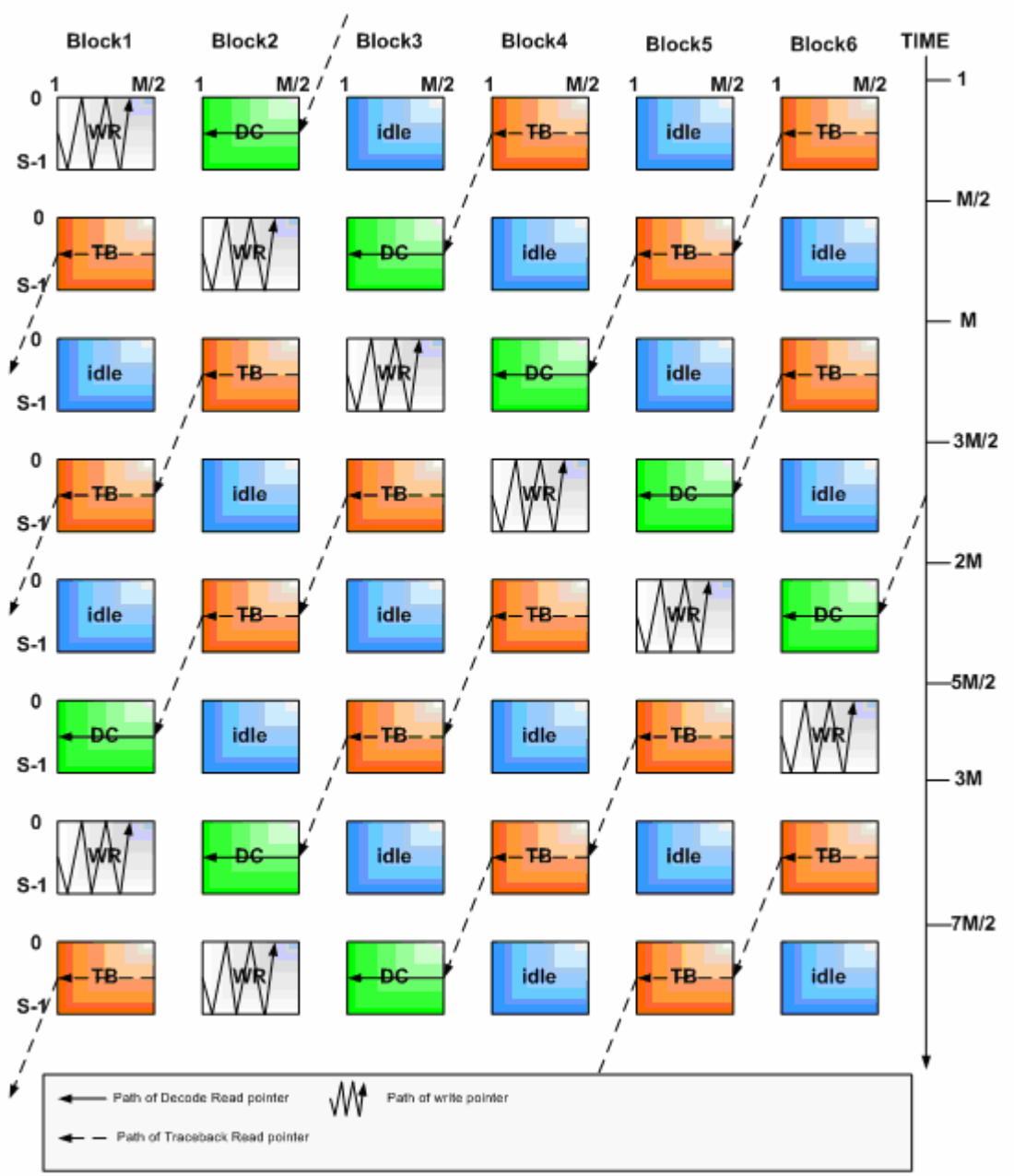

圖 4-17 TBM k point even 流程圖

4.3 硬體模擬結果

在本節中我們依照4.1所描述的設計流程將所設計之接收機用VHDL語言寫成行為模式(behavior mode)的硬體架構，流程主要有以下幾點:

1. 使用 MATLAB 做定點模擬，在和浮點模擬有可接受的效能衰減下，決定各個單元中變數的位元數，而定點模擬和浮點模擬之比較如圖 4-22 所示。
2. 以 Modelsim 軟體模擬驗證此行為模式是否正確,再和步驟 1 中的 MATLAB 定點模擬比較來驗證。
3. 行為模式正確後就可將此VHDL程式輸入Xilinx的ISE 6軟體就可得到合成後的 RTL 巢狀(netlist)架構檔，此是根據所撰寫行為模式下 VHDL 程式所做合成後的輸出檔案， 而所設計之接收機之 RTL 架構圖如圖 4-18 及圖 4-19 所示。

然而吾人在硬體實現中並沒有做 FPGA 驗証，因此其它模擬並無進行。最後由 ISE 6 所產生的 Mapping report 及 Timing report 如圖 4-20 及圖 4-21 所示。

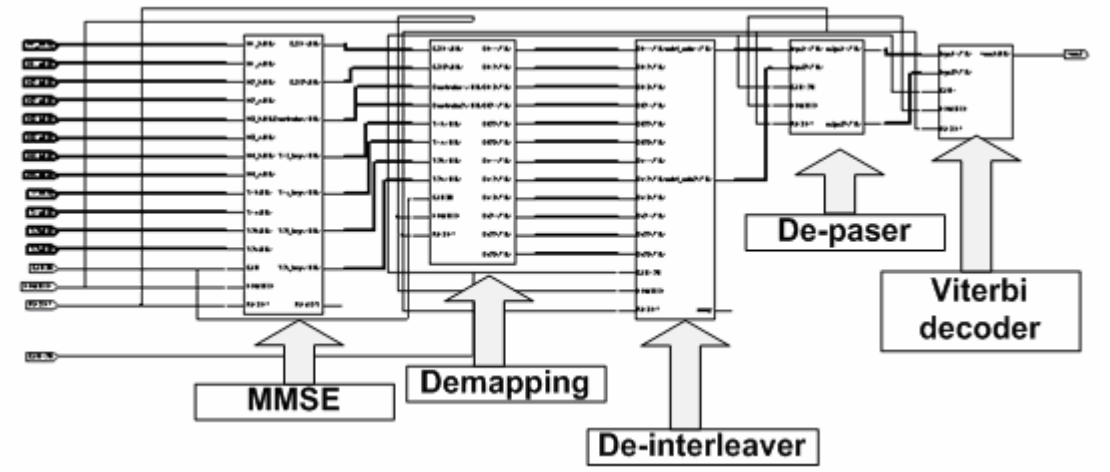

圖 4-18 接收機之 RTL 架構圖

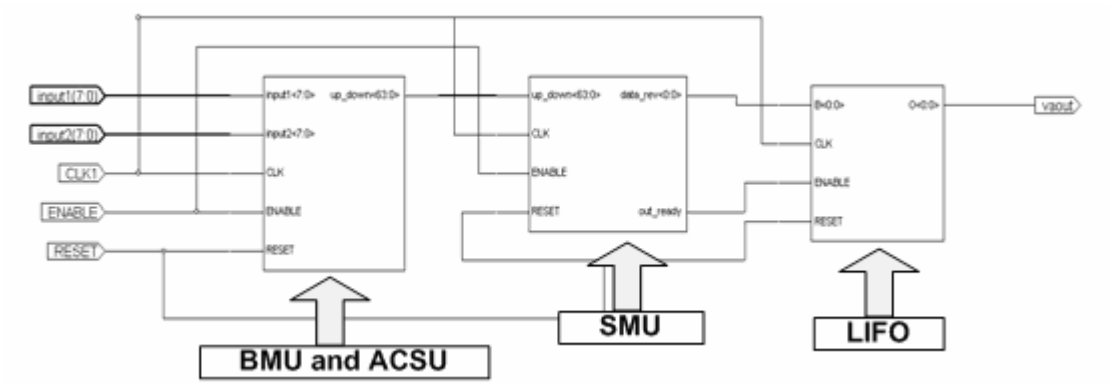

圖 4-19 Viterbi 解碼器之 RTL 架構圖

```
Design Summary
--------------
Number of errors:      0
Number of warnings:    0
Logic Utilization:
  Total Number Slice Registers:   1,896 out of 38,400    4%
    Number used as Flip Flops:                1,510
    Number used as Latches:                     386
  Number of 4 input LUTs:         7,884 out of 38,400   20%
Logic Distribution:
    Number of occupied Slices:                        4,562 out of 19,200   23%
    Number of Slices containing only related logic:   4,562 out of  4,562  100%
    Number of Slices containing unrelated logic:          0 out of  4,562    0%
        *See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:        7,976 out of 38,400   20%
      Number used as logic:                     7,884
      Number used as a route-thru:                 92
   Number of bonded IOBs:           138 out of    404   34%
      IOB Latches:                                  16
   Number of Block RAMs:             19 out of    160   11%
   Number of GCLKs:                   2 out of      4   50%
   Number of GCLKIOBs:                2 out of      4   50%

Total equivalent gate count for design:  421,193
Additional JTAG gate count for IOBs:  6,720
Peak Memory Usage:  221 MB
```

圖 4-20 接收機之 Mapping report

```
Timing Summary:
---------------
Speed Grade: -8

   Minimum period: 13.644ns (Maximum Frequency: 73.292MHz)
   Minimum input arrival time before clock: 47.616ns
   Maximum output required time after clock: 7.316ns
   Maximum combinational path delay: No path found
```

圖 4-21 接收機之 Timing report

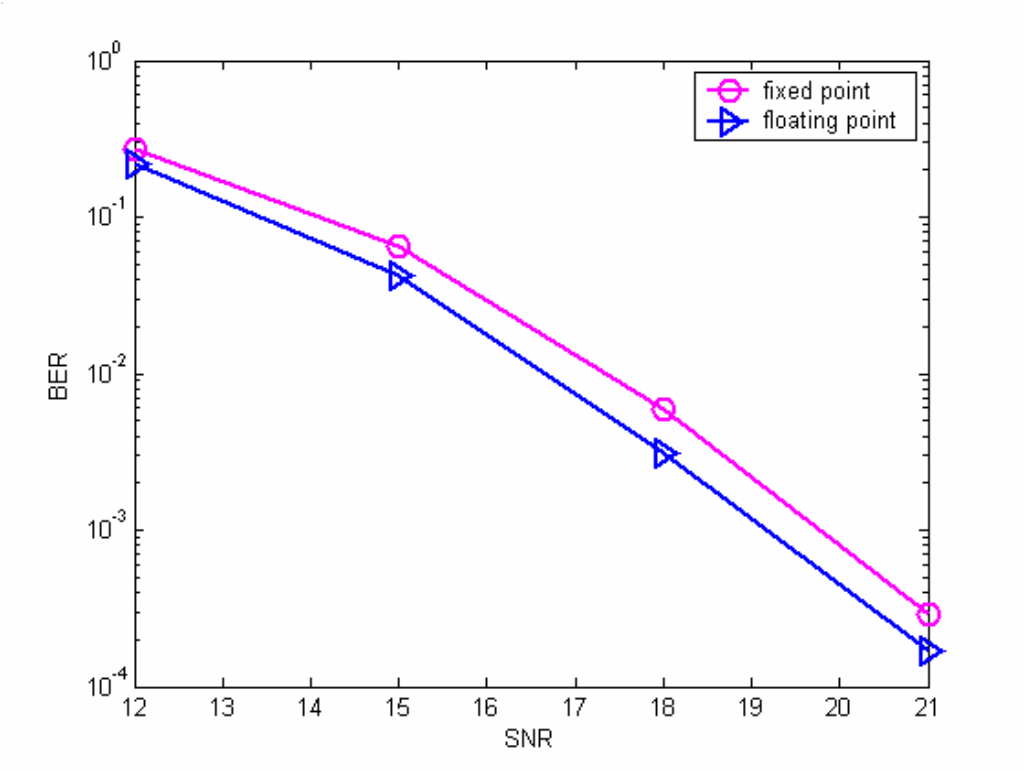

圖 4-22 接收機浮點模擬對定點模擬之比較圖

第5章結論

在本論文中，吾人考量如何在 BICM MIMO-OFDM 系統中結合 Viterbi 解碼器及偵測單元，而做一個整體的設計。論文主要有二個工作環境，第一個是在由 TGN Sync 所提出的 11n 草案中傳送架構，在這個傳送架構下，吾人的設計的重點包括了演算法的選擇、CSI 計算與選擇及軟性反對映。因此本論文的這個部分中首先介紹了在 MIMO-OFDM 中常用的偵測方法包括了 MMSE 及 V-BLAST 二種。接著介紹簡化之軟性反對映，並且在這樣的軟性反對映下，介紹軟性輸入軟性輸出之 MMSE 偵測單元。在這部分的最後，本論文介紹了 11n 草案中傳送端編碼器、壓縮器、交錯器等單元是如何定義的。

論文的第二個工作環境是平行編碼器的傳送架構，在這傳送架構下，每根傳送天線都有編碼器、壓縮器及交錯器。在這樣的環境中，提高效能是在這部分主要的設計考量。本論文以 Pre-ordering 及遞迴二種系統討論如何在 BICM MIMO OFDM 系統中結合 VBLAST 及 Viterbi 解碼器，其中由吾人所發展出來的 Pre-ordering 系統中，由於先在傳送端排序，因此可在接收端讓通道狀況最佳天線最先開始偵測且經過解碼後再重建回來幫助下一根天線的偵測，這在第三章最後的模擬結果發現，這樣的方法的確能有效提高效能表現。

在論文的最後，吾人依一般 FPGA 設計流程使用 VHDL 實現第二章所介紹的簡化之接收機，主要包 MMSE 偵測單元及 Viterbi 解碼器。然而 MMSE 偵測單元方面吾人所使用的實現方法所產生的乘法器個數過多，且沒有將各個單元之間的乘法器共用來降低複雜度，因此降低偵測單元的運算複雜度將是未來實現的主要改進空間。

參考文獻

[1] G.. J. Foschini and M.J. Gans, “On limits of wirelesss communications in a fading environment when using multiple antennas, ” *Wireless Pres. Commun.*, vol. 6, no. 3, pp.311-335, Mar. 1998

[2] A. van Zelst, “Space division multiplexing algorithms,” in *Proc. 10th Mediterranean Electrotech. Conf.*, vol. 44, pp. 744-756, Mar, 1998.

[3] A. van Zelst, R. van Nee, and G. A. Awater, “Space division multiplexing(SDM) for OFDM system,” In *Proc. IEEE Veh. Technol. Conf.*, May 200, pp. 1070-1074.

[4] B. Hassibi, “A fast square-root implementation for BLAST,” in Conf. Rec. Thirty-Fourth Asilomar Conf. Signals, Syst. Comput., 2000, pp. 1255–1259.

[5] P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela, “V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel,” in *Proc. ISSSE, 1998*, pp. 295–300.

[6] G. D. Golden, C. J. Foschini, R. A. Valenzuela, and P. W. Wolniansky, “Detection algorithm and initial laboratory results using V-BLAST space-time communication architecture,” *Electron. Lett.*, vol. 35, no. 1, pp. 14–16, Jan. 1999.

[7] V. Tarokh, N. Seshadri, and A. R. Calderbank, “Space-time codes for high data rate wireless communication: Performance criterion and code construction,” *IEEE Trans. Inform Theory*, vol. 44, pp. 744–756, Mar.1998.

[8] “TGn Sync proposal technical specification, ” *TGn Sync*, Mar. 2005.

[9] *IEEE 802.11a Stand., ISO/IEC 8802-11:1999/Amd 1:2000(E*)

[10] B. Wubben, R. Bohnke, J. Rinas, V. Kuhn, and K. D. Kammeyer, “Efficient algorithm for decoding layered space-time codes,” *Electron. Lett.*, vol. 37, no. 22, pp. 1348–1350, Oct. 2001.

[11] Choi , Jinho, “A bi-directional zero-forcing BLAST receiver” *IEEE transactions on Signal Processing*, vol. 52, no.9, pp.2670-2673, Sept. 2004.

[12] D., Wubben, , Bohnke, R., Kuhn, V., Kammeyer, K.-D.,“MMSE extension of V-BLAST based on sorted QR decomposition,” *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th* , vol. 1, Oct. 2003.

[13] Viterbo, E., Bouros, J., “A universal lattice code decoder for fading channels,” *IEEE Transactions on Information Theory*, vol. 45, pp. 1639 – 1642, July 1999.

[14] H. Vikalo and B. Hassibi, “The Expected Complexity of Sphere Decoding, Part I: Theory, Part II: Applications,” *IEEE transactions on Signal Processing, submitted for publication*, 2003.

[15] G. Caire, G. Taricco and E. Biglieri, “Bit-interleaved coded modulation,” *IEEE Trans. Info. Theory*, vol. 44, pp. 927-946, May 1998.

[16] F. Tosato and P. Bisaglia, “Simplified soft-output demapper for binary interleaved COFDM with application to HIPERLAN/2,” *Proc. IEEE Int’l. Conf. on Comm. 2002*, vol. 2, pp. 664-668 , 2002.

[17] K.-B., Song; Chan-Soo Hwang; Cioffi, J.M., “Rate-compatible punctured convolutionally (RCPC) space-frequency bit-interleaved coded modulation (SF-BICM),” *IEEE International Conference on Communications*, vol. 6, June, pp. 3284 – 3288, 2004

[18] A. van Zelst, “Per-antenna-coded schemes for MIMO OFDM,” in *Proc. IEEE Int. Conf. Commun.*, vol. 4, Anchorage, AK, May 2003, pp. 2832-2836.

[19] X. Li, A. Chindapol and J. A. Ritchey, “Bit-interleaved coded modulation with iterative decoding and 8 PSK signaling,” *IEEE Trans. Commun.*, vol. 50, pp. 1250-1257, 2002

[20] A. M. Tonello, “Space-time bit-interleaved coded modulation with an iterative decoding strategy,” *Proc. of IEEE Vehi. Tech. Conf. 2000 Fall*, Boston,

pp.473-478, Sep. 2000.

[21] K. B. Song and S. A. Mujtaba, “A low-complexity space-frequency BICM MIMO-OFDM system for next-generation WLANs” *Proc. IEEE Globecom*, vol. 2 pp. 1059 - 1063, 2003.

[22] O. Oteri, A. Paulraj, W. J. Chimitt, K. Holt, “SPACE-TIME-FREQUENCY CODING FOR OFDM-BASED WLANs,” http://whitepapers.zdnet.co.uk/0,39025945,60131031p-39000516q,00.htm

[23] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm” *IEEE Transactions on Information Theory* ,vol. 13, issue 2, Apr 1967, pp.260 – 269.

[24] C. B. Shung, Gottfrried Ungerboeck, H. K. Thapar, “VLSI Architectures for Metric Normalization in the Viterbi Algorithm”, *IEEE International Conference* vol.4, 1999, pp. 1723 –1728.

[25] Stephen B. Wicker, “Error Control Systems for digital communication and storage”, 1995 by Prentice-Hall, Inc.

[26] C. M. Rader, “Memory management in a Viterbi algorithm” *IEEE Trans. Commun*., vol. 29, Sept.1981, page(s) : 1399-1401.

[27] Gennady Feygin and P. G. Gulak, “Architectural Tradeoffs for Survivor Sequence Memory Management in Viterbi Decoders”, *IEEE Trans. Commun*, vol. 41. No. 3. March 1993.

[28] Michael Horwitz and Robin Braun, “A generalized Design Technique For Trace back Survivor Memory Management In Viterbi Decoders”, *IEEE*, 1997.

簡歷

姓　　名：　李峰宇

性　　別：　男

出生日期：　民國 70 年 7 月 6 日

出生地　：　雲林縣

學　歷　：

桃園縣立文安國小　(1987.9~1993.6)

桃園縣立大成國中　(1993.9~1996.6)

國立武陵高級中學　(1996.9~1999.6)

國立中正大學電機系　(1999.9~2003.1)

國立交通大學電信工程研究所碩士班(2003.2~2005.7)

公元 2005 年 7 月獲得碩士學位