

國立交通大學  
電信工程學系碩士班  
碩士論文

G.SHDSL 之可適應性等化器及格狀編碼解碼器

的硬體實現



Hardware Architecture Design of  
Adaptive Equalizer and TCM Decoder for G.SHDSL

研究生：李佳勳

指導教授：紀翔峰 博士

中華民國九十三年九月

G.SHDSL 之可適應性等化器及格狀編碼解碼器  
的硬體實現

Hardware Architecture Design of  
Adaptive Equalizer and TCM Decoder for G.SHDSL

研究生：李佳勳

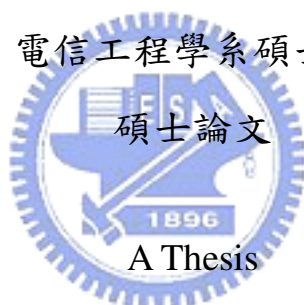
Student: Jia Xun Lee

指導教授：紀翔峰 博士

Advisor: Dr. Hsiang-Feng Chi

國立交通大學

電信工程學系碩士班



Submitted to Department of Communication Engineering  
College of Electrical Engineering and Computer Science

National Chiao-Tung University

for the Degree of Master

in

Communication Engineering

September 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年九月

# G.SHDSL 之可適應性等化器及格狀編碼解碼器 的硬體設計

學生：李佳勳

指導教授：紀翔峰 博士

國立交通大學電信工程學系碩士班



單銅絞線高速數位用戶迴路 (Single-pair High Speed Digital Subscriber Loop, SHDSL)，為新一代的對稱式 DSL 技術。可提供對稱的上下傳速率，最高可達 2.3Mbps。然而，在通訊系統中常見的 ISI (InterSymbol Interference)問題，同樣的在 SHDSL 系統中也會遇到，為了使傳輸效能不受 ISI 影響，我們必須設計一個可適應性的等化器解決 ISI 問題。常用的決策回授等化器有錯誤蔓延等的缺點存在，且不容易加上 TCM 解碼器以增加通道編碼增益。我們想要一個可以解決 ISI 問題同時容易得到通道編碼增益的等化器，而 Tomlinson-Harashima precoder(THP)系統有這項優點。所以我們以 G.SHDSL 規範，針對 THP 系統與 TCM 解碼，從演算法設計及電腦模擬開始，最後設計出符合成本效益之硬體架構，所得到的硬體以 50MHz 的操作頻率即可以符合最大傳輸速率的條件，同時以 FPGA 模擬板驗證了硬體功能。


# Hardware Architecture Design of Adaptive Equalizer and TCM Decoder for G.SHDSL

student : Jia Xun Lee

Advisor : Dr. Hsiang-Feng Chi

Department of Communication Engineering  
National Chiao Tung University

## ABSTRACT

The logo of National Chiao Tung University is a circular emblem. It features a central shield with a book and a torch, surrounded by the university's name in Chinese and English, and the year 1989. The shield is set against a background of a gear-like border.

The Single-pair High Speed Digital Subscriber Loop (SHDSL) is the new generation symmetric DSL technique which could supply at most 2.3 Mbps downlink and uplink data rate symmetrically to subscribers on a long loop. However, the InterSymbol Interference (ISI) is severe especially duration the transmission over a long loop. Normal data transmission is impossible without properly taking care of the ISI problem. To assure SHDSL transceivers to provide full-rate transmission, we need a powerful adaptive equalizer to ease the ISI problem. The decision feedback equalizer is the most often used equalizer for solving the ISI problem. However, it has the error propagation problem, which will degrade the system performance. To improve the performance, the joint equalization and channel decoding is necessary. Nevertheless, combining the trellis decoder in the decision feedback equalizer will result in high complexity hardware. A powerful equalizer called

Tomlinson-Harashima precoder (THP) system was proposed to solve this problem. By use of the precoding technique, the joint equalization and channel decoding can be accomplished by cascade a linear equalizer and a TCM decoder for channel coding. In this thesis, starting from algorithm design and computer simulation, we design the THP system and TCM decoder hardware architectures according to the G.SHDSL recommendation. The resulting hardware could achieve the maximum 2.3 Mbps data rate under the 50 MHz operation clock. The hardware was verified on the FPGA development board.



## 致謝

本論文的以順利的完成，第一個要感謝我的指導教授紀翔峰博士。在我的碩士兩年時間，給我適時的課業指導與細心的叮嚀。論文的進行過程中給予我明確的觀念，培養我獨立思考與研究的精神，一路的學習讓我受益良多，正確的建立研究生應有的態度。

還有碩士班的同窗同學賴昭宏、鍾文狀、李昭宏、王志軒等人，一起在課業上的奮鬥與生活上的相處，感到兩年碩士班過的很充實。

最後，一直在我求學過程中默默支持我的母親與家人，姑姑，我的女友，涂琬亭，還有涂爸爸與涂媽媽，謝謝你們的照顧、關心與鼓勵。我終於完成碩士班兩年的求學階段，能有今天都是你們給我的，謝謝你們。



致謝.....	iv
圖目錄.....	vii
表目錄.....	ix
第一章 緒論.....	1
1.1 論文簡介.....	1
1.2 研究動機與目標.....	2
1.3 論文組織.....	2
第二章 G.SHDSL PMD 層規範簡介.....	3
2.1 PMD Preactivation.....	4
2.2 PMD Activation Sequence.....	4
2.3 資料模式(Data Mode)運作.....	5
2.3.1 TCM 編碼器.....	5
2.3.2 Channel Precoder.....	7
第三章 Single Carrier Equalizer 與 Tomlinson-Harashima Precoder System.....	8
3.1 等化器基本原理.....	8
3.1.1 線性等化器(Linear Equalizer , LEQ).....	10
3.1.2 Fractional Spaced Equalizer (FSE).....	12
3.1.3 決策回授等化器(Decision Feedback Equalization , DFE).....	12
3.1.4 適應性等化器.....	14
3.1.5 適應性決策回授等化器.....	15
3.2 Tomlinson-Harashima Precoder (THP).....	16
3.2.1 THP 基本原理.....	16
3.2.2 資料跳動(data flipping)現象.....	18
3.2.3 資料跳動(data flipping)現象之解決方法.....	19
第 4 章 格狀編碼調變(TCM)解碼器.....	21
4.1 格狀編碼調變(TCM).....	21
4.2 TCM 解碼器.....	22
4.2.1 Viterbi 演算法.....	23
4.2.2 Viterbi 解碼方法.....	25
4.2.2.1 Branch Metric Unit (BMU).....	25
4.2.2.2 Add、Compare and Select Unit.....	26
4.2.2.3 Survivor Memory Unit.....	26
4.3 結合 THP 系統與 TCM 解碼器.....	27
第 5 章 THP 系統及 TCM 解碼器之硬體架構.....	29
5.1 THP 系統硬體架構.....	29
5.1.1 訓練模式硬體架構.....	30
5.1.1.1 錯誤估測量化器與移位乘法器.....	31
5.1.1.2 前饋濾波器.....	35

5.1.1.3 回授濾波器.....	37
5.1.1.4 決策裝置(slicer) .....	39
5.1.2 資料模式硬體架構.....	39
5.2 TCM 解碼器硬體架構.....	41
5.2.1 Branch Metric Unit (BMU) .....	43
5.2.2 Add、Compare and Select Unit (ACSU) .....	45
5.2.2.1 記憶體位址產生器.....	46
5.2.2.2 Radix-4 Butterfly .....	50
5.2.2.3 找尋最佳狀態計量與狀態計量正規化.....	55
5.2.3 Survivor Memory Unit (SMU) .....	55
5.2.3.1 位址產生器.....	56
5.2.3.2 交換器架構與最佳狀態解碼.....	56
5.2.3.3 未經編碼位元 $\{y_1, y_2\}$ 之解碼 .....	57
5.3 結論.....	57
第六章 硬體之模擬、設計與驗證.....	59
6.1 THP 系統與 TCM 解碼器之模擬 .....	59
6.1.1 THP 系統之軟體模擬.....	59
6.1.1.1 THP 系統訓練模式.....	60
6.1.1.2 THP 系統資料模式.....	65
6.1.2 TCM 解碼器.....	66
6.2 硬體實作與量測.....	67
6.2.1 硬體設計流程.....	67
6.2.2 ASIC 流程.....	67
6.2.3 FPGA 流程 .....	68
6.3 結論.....	70
第七章 總結與未來展望.....	71
參考文獻.....	72



# 圖目錄

圖 2-1 : Preactivation 參考模型 .....	4
圖 2-2 : Activation 參考模型 .....	4
圖 2-3 : PMD 層資料模式參考模型.....	5
圖 2-4 : 格狀編碼調變(TCM)編碼器方塊圖.....	6
圖 2-5 : Channel Precoder 方塊圖.....	7
圖 3-1 : 線性等化器 .....	10
圖 3-2 : 通道與 Zero-forcing 等化器.....	11
圖 3-3 : 決策回授濾波器 .....	13
圖 3-4 : 雜訊模型 .....	13
圖 3-5 : 線性等化器之 LMS 硬體架構.....	15
圖 3-6 : THP 系統.....	17
圖 3-7 : 資料跳動現象 .....	18
圖 3-8 : 4-PAM 星雲圖.....	19
圖 3-9 : 擴展後之 4-PAM 星雲圖.....	20
圖 4-1 : 8-PSK 訊號集合劃分.....	22
圖 4-2 : 迴旋編碼器為 $K=10$ , $R=1/2$ 之 TCM 方塊圖 .....	23
圖 4-3 : 16-PAM 信號星雲圖.....	23
圖 4-4 : 512 個狀態之格子圖 .....	25
圖 4-5 : ACS 遞迴式運算 .....	26
圖 4-6 : THP 系統與 TCM 解碼器.....	28
圖 5-1 : 訓練模式方塊圖 .....	30
圖 5-2 : 錯誤估測量化圖 .....	33
圖 5-3 (a) : 錯誤估測量化器.....	33
圖 5-3 (b) : 錯誤估測量化器真值表.....	34
圖 5-4 : 移位乘法器 .....	35
圖 5-5 : 前饋濾波器硬體架構 .....	36
圖 5-6 : 回授濾波器硬體架構圖 .....	39
圖 5-7 : TH Precoder 方塊.....	39
圖 5-8 : THP 濾波器硬體架構圖.....	41
圖 5-9 : TCM 解碼器方塊圖.....	42
圖 5-10 : Radix-2 與 Radix-4 格子圖 .....	43
圖 5-11 : Radix-4 格子圖 .....	44
圖 5-12 : 擴展後之星雲圖 .....	45
圖 5-13 : ACSU 方塊圖.....	46
圖 5-14 : 狀態 $N=16$ 之 Radix-2 格子圖 .....	47
圖 5-15 : (a)大矩陣運算(b)分解成三個小矩陣運算 .....	48

圖 5-16 : $i:[a, b] \xrightarrow{S_{3,5}} j:[b, a]$ 圖示 .....	49
圖 5-17 : 資料位址產生器方塊圖 .....	50
圖 5-18 : Singal ACS of Radix-4 Butterfly 硬體 .....	51
圖 5-19 : Radix-4 Butterfly 之時序圖 .....	53
圖 5-20 : (a)(b)狀態計量記憶體之讀寫時序圖，(c)(d)讀寫之分配時序圖 .....	54
圖 5-21 : SMU 方塊圖 .....	56
圖 6-1 : 設計流程 .....	59
圖 6-2 : 訓練模式之模擬方塊圖 .....	60
圖 6-3 : AWGN，通道模型一之收斂速度 .....	61
圖 6-4 : AWGN，通道模型二之收斂速度 .....	61
圖 6-5 : AWGN，通道模型一—訓練模式收斂至 24dB 時之眼圖 .....	62
圖 6-6 : AWGN，通道模型二—訓練模式收斂至 23dB 時之眼圖 .....	62
圖 6-7 : Cross talk，通道模型一之收斂速度 .....	63
圖 6-8 : Cross Talk，通道模型二之收斂速度 .....	64
圖 6-9 : Cross Talk，通道模型一—訓練模式收斂至 25dB 時之眼圖 .....	64
圖 6-10 : Cross talk，通道模型二—訓練模式收斂至 25dB 時之眼圖 .....	65
圖 6-11 : 資料模式 16PAM 眼圖 .....	66
圖 6-12 : TCM 解碼器與未編碼 8-PAM 位元錯誤率之比較 .....	67
圖 6-13 : 電路驗證環境 .....	70



## 表目錄

表 2-1：啓動連結流程 .....	3
表 2-2：16-PAM 訊號對映表.....	6
表 5-1：訓練模式記憶體容量 .....	31
表 5-2：前饋濾波器之係數記憶體分配 .....	36
表 5-3：回授濾波器之係數記憶體分配 .....	38
表 5-4：THP 濾波器記憶體配置.....	40
表 5-5：THP 係數記憶體分配.....	40
表 5-6：Radix-2 <sup>k</sup> 硬體架構比較 .....	42
表 5-7：TCM 解碼器之記憶體需求.....	42
表 5-8：SMU 架構之比較.....	58
表 6-1：THP 系統係數與資料位元寬度.....	59
表 6-2：THP 系統輸出結果.....	65
表 6-3：TCM 解碼器與 THP 系統硬體之邏輯閘數目 .....	68



---

# 第一章 緒論

---

## 1.1 論文簡介

單銅絞線高速數位用戶迴路 (Single-pair High Speed Digital Subscriber Loop, SHDSL), 為新一代的對稱式 DSL 技術, 因統一標準 G.991.2(G.SHDSL)[1] 的確立及成熟, 逐漸成為目前用於企業用戶之對稱式 DSL 市場的主流。利用 SHDSL 傳輸技術, 電信及網路服務業者只需在現存的電話網路架構下, 利用一條銅絞線提供高達 2.3Mbps 的雙向對稱、多重速率的高速傳輸服務。目前普及率相當高的 ADSL 是特別為住戶的應用而設計的, ADSL 的特性為非對稱的上下傳速度, 其中下傳(downstream)速度大於上傳(upstream)速度。不同於 ADSL, SHDSL 提供了以對稱方式上下傳, 速度介於 192 kbps 至 2.3Mbps, 且距離可達到 10kft(3km)至 20kft(6km)。此外, 還提供了 4-wire mode 可選擇, 使傳輸速度提升至 4.6Mbps, 或者以 2.3Mbps 速度傳輸達到 16kft(5km)距離。此特性非常適合在商業上的應用, 如視訊會議的時候會需要同時上下傳, 如果以 SHDSL 而言將可以提供此功能, 讓視訊會議達到暢通且無延遲。

然而, 在通訊系統中通道造成的碼際干擾(InterSymbol Interference, ISI)是很常見的問題, 通常用來抵抗碼際干擾的方法是使用等化器, 若加上通道編碼的話還可以再改善一些效能。DFE(Decision Feedback Equalization) [2,3]是很常被使用的等化器, 不過 DFE 有錯誤蔓延(error propagation)的現象與決策延遲(decision delay)兩個缺點, 兩者都會影響整體效能。這樣的問題必須被改善。

G.SHDSL 採用 Tomlinson-Harashima precoder (THP) [4,5]系統與格狀編碼調變 (Trellis code modulation, TCM)[6,7,8]來做等化器與通道編碼的工作。在技術上, Tomlinson-Harashima precoder 具有 DFE 的功能, 既可以抵抗碼際干擾同時也沒

有錯誤蔓延與決策延遲的缺點。而且可以很容易的串接(cascade)TCM 解碼器以增進效能，使用 TCM 編碼可以增進 3-6dB 之編碼增益。

## 1.2 研究動機與目標

為了克服 ISI 問題，我們必須實現一個強大的等化器，Tomlinson-Harashima precoder 相較於其他線性等化器與決策迴授等化器，它沒有放大雜訊、錯誤蔓延等問題存在，可以很容易的與 TCM 解碼器配合，增加編碼增益。本論文的目的是在 G.SHDSL 的規範下，設計出以 THP 系統作為等化器、以 TCM 作通道編碼並以 Viterbi decoder 做解碼工作的架構，解決碼際干擾問題及提升編碼增益，使用 THP 將等化器與 TCM 解碼器結合[3,9,10,11]。在研究的過程中，先以 MATLAB 做演算法的模擬，決定了演算法後，使用 C 語言進行定點數運算模型之撰寫，作為定點數的模擬及分析。有了定點數分析之模型，我們根據此模型設計出硬體架構。硬體之實現為最後的目標，我們完成符合低成本與規格之需求的硬體設計，最後以 FPGA (Field-Programmable gate arrays)做電路驗證並量測。

## 1.3 論文組織

本論文章節總共分七章，其內容敘述如下：

第一章是緒論。在第二章，我們把 SHDSL PMD 層的運作方式作了簡單的說明。並簡單的說明了 PMD 層的規範，PMD 層中包含了 TCPAM、Channel precoder 與 TCM 解碼器，它們是作等化器及通道編碼的工作，其相關的基本原理介紹將在第三章與第四章介紹到。在第三章，介紹等化器的基本原理，常見的等化器等，包括本論文所討論的 THP 系統。在第四章，介紹 TCM 解碼器。關於硬體的架構會在第五章講到，包括 THP 系統與 TCM 解碼器之硬體架構。接下來的第六章會說明硬體的模擬之結果與 ASIC、FPGA 的實做。第七章為未來展望與總結。

## 第二章 G.SHDSL PMD 層規範簡介

本章介紹了 G.SHDSL 規範在 PMD 層的簡介，針對 SHDSL 的規範做完善的說明，而等化器與 TCM 解碼器將是本論文的實現重點。在 PMD 層的主要功能有：

- 符元時序(timing)產生與恢復。
- 編碼與解碼。
- 調變與解調。
- 回音(echo)消除。
- 等化(line equalization)。
- 啟動連結(link startup)。

	1. Preactivation	2. Activation	3. Data Mode
訊號	G.994.1 => probe signal => G.994.1	Preact. => act. => Preact.	Data Signal
必須完成之任務	a. Data rate negotiation b. Power backoff c. Timing recovery	a. Echo canceller、precoder coefficient training b. Precoder、CRC、encoder coeff. transfer	a. Timing tracking、THP coefficient tracking

表 2-1：啟動連結流程

本論文主要工作在第二項及第五項，編碼與解碼與等化的硬體實現兩個目標上。在文中 Central office 端之收發機以 STU-C(SHDSL Transceiver Unit at the Central Office)簡寫，Remote 端之收發機以 STU-R(SHDSL Transceiver Unit at the Remote end)簡寫。當開始啟動 STU-C 與 STU-R 之連結時，PMD 層先啟動 G.994.1、

Preactivation 與 Activation 程序，完成啟動連結會正式進入資料模式，數據傳輸才可開始運作。啟動連結完成後將進入資料模式，之前經過訓練的 precoder、encoder、CRC 即可進行數據傳輸。其啟動連結流程如表 2-1：

## 2.1 PMD Preactivation

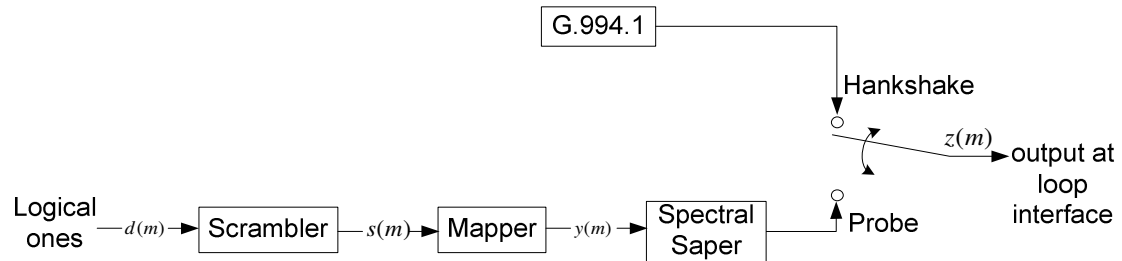


圖 2-1：Preactivation 參考模型

如圖 2-1 為 STU-C 或 STU-R 發射器之 preactivation 模型的參考圖[1]。符號  $m$  代表 symbol time，由於探測訊號使用 2-PAM 調變，所以位元時間就等於 symbol time。而符碼率，spectral shape，持續時間與 power backoff 會在 G.994.1[12]時選定。G.994.1 是 ITU 為 xDSL 而定的建議文件，當 xDSL 啟動時必須交換彼此接收機之溝通訊息，有關傳輸速率、環境、功率、條件等的參數。

## 2.2 PMD Activation Sequence

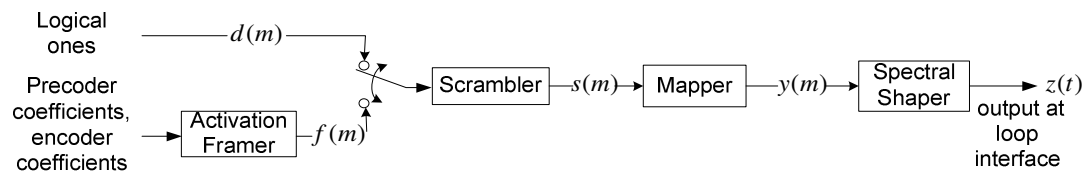


圖 2-2：Activation 參考模型

圖 2-2 為 activation 的模型圖。在 PMD 資料模式可以正式運作之前，需要做 activation 的程序[1]。STU-C 與 STU-R 會做一些溝通來達成 activation 模式運作。Activation 要進行 echo canceller、precoder coefficient 等訓練，做完了訓練必須把

precoder、encoder、CRC 等係數轉移才算完成 activation。會將訊號  $f(m)$  送出，將可完成轉移工作。

## 2.3 資料模式(Data Mode)運作

接下來為資料模式介紹，SHDSL 是以 TC-PAM (Trellis coded pulse amplitude modulation) 作為調變形式。如圖 2-3 所示，為 PMD 層資料模式的參考模型。被傳輸的資料皆使用此模型，當 STU-x 得到 precoder、encoder 等的係數後，資料模式就可以運作。

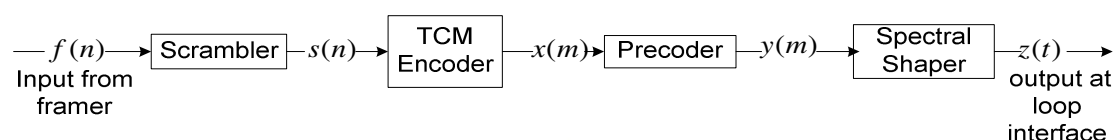


圖 2-3：PMD 層資料模式參考模型

圖中符號  $n$  代表著位元時間(bit time)， $m$  代表著符元時間(symbol time)， $t$  則是表示類比時間(analog time)。 $f(n)$  為 scrambler 的輸入，經過 scrambler 後輸出為  $s(n)$ 。 $s(n)$  為 PMD layer 要傳送的訊號，它會經過 TCM(Trellis Coded Modulation) 編碼器，產生  $x(m)$  訊號。 $x(m)$  會經過 THP 系統作通道預編碼處理，產生  $y(m)$  訊號。最後才經過 spectral shaper 將類比訊號  $z(t)$  經 loop interface 送出。

以下兩節說明 TCM 解碼器與 precoder，關於 scrambler、spectral shaper 之 PSD 可參考[1]。

### 2.3.1 TCM 編碼器

圖 2-4 是格狀編碼的方塊圖，一連串的 stream 從 scrambler 的輸出到 TCM 的輸入，經過了串聯序列轉平行列序器(S/P)會以每  $K$  個位元的串聯序列被平行輸出為一個符元，這  $K$  個位元中有一位元經過碼率為  $1/2$  的迴旋碼編碼器，以  $K+1$



位元經過 16-PAM 的 Mapper 的調變輸出。

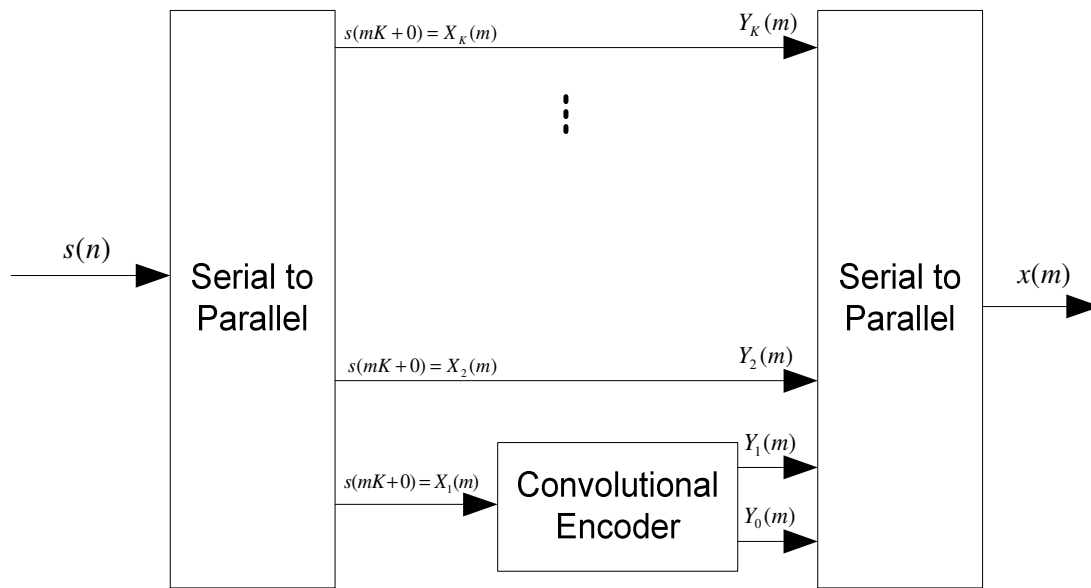


圖 2-4：格狀編碼調變(TCM)編碼器方塊圖

$Y_3(m)$	$Y_2(m)$	$Y_1(m)$	$Y_0(m)$	$x(m)$ 16-PAM
0	0	0	0	-15/16
0	0	0	1	-13/16
0	0	1	0	-11/16
0	0	1	1	-9/16
0	1	0	0	-7/16
0	1	0	1	-5/16
0	1	1	0	-3/16
0	1	1	1	-1/16
1	0	0	0	1/16
1	0	0	1	3/16
1	0	1	0	5/16
1	0	1	1	7/16
1	1	0	0	9/16
1	1	0	1	11/16
1	1	1	0	13/16
1	1	1	1	15/16

表 2-2：16-PAM 訊號對映表

$K+1$  個位元  $Y_k(m)$ ， $\dots$ ， $Y_1(m)$ ，和  $Y_0(m)$  會被對映到一個準位  $x(m)$ 。如表 2-2 為一個 16-PAM 的對映表。

### 2.3.2 Channel Precoder

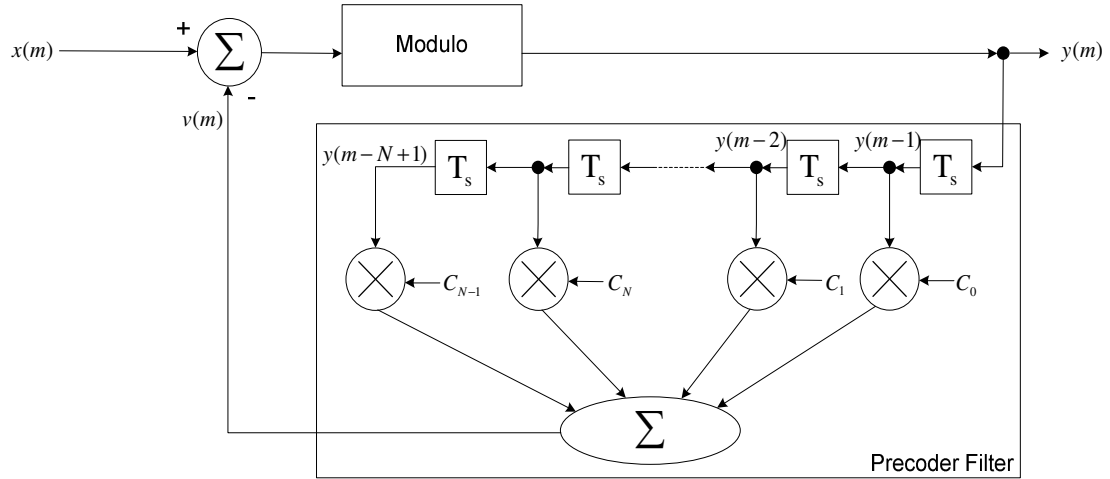


圖 2-5：Channel Precoder 方塊圖

圖 2-5 為 channel precoder 的方塊圖，包括 Modulo 與 Precoder 濾波器，其中  $T_s$  是每個符元延遲的時間，為 baud rate。其濾波器的係數經過訓練模式得到後會被轉移進 channel precoder。

---

## 第三章 Single Carrier Equalizer 與 Tomlinson-Harashima Precoder System

---

在通訊系統中，資料的傳送必須透過波道傳輸，例如電話纜線或無線波道這樣的不完美通道，在傳輸時因為多重路徑的關係，會產生碼際干擾(Intersymbol Interference, ISI)問題[13]，ISI 將嚴重的干擾訊號。當接收機收到訊號後將難以重建原本的傳輸訊號，效能嚴重的降低，此時需要一個功能強大的等化器來消除碼際干擾問題。然而在通訊中不可能會使用固定的通道，為了因應通道的改變，則等化器必須是適應性的，能夠隨著通道的改變而做調整，以應付通道的變化。關於等化器的技術已有相當多的研究，在本章我們將做快速的介紹幾種常見的技術，如線性等化器[3,13]、決策回授等化器[2,3,13]與在 G.SHDSL 系統中會用到的 Tomlinson-Harashima Precoder (THP) 系統等[4,5,9,10]。

### 3.1 等化器基本原理

等化器被設計來補償不完美通道特性，使的被傳送的訊號可以減少失真程度，其中最嚴重的干擾為 ISI 的現象，本節就從 ISI 的現象開始說明等化器的功能。

以  $I_k$  表示所要傳的離散訊號，經過傳輸濾波器  $g(t)$ ，其中  $G(f)$  為  $g(t)$  的傅立葉轉換，當  $|f| > W$  時  $G(f)=0$ ，也就是  $G(f)$  為有限頻寬。我們送出訊號

$$v(t) = \sum_{n=0}^{\infty} I_n g(t - nT) \quad (3-1)$$

以通道  $C(f)$  傳輸，這個通道為有限頻寬通道，在  $|f| > W$  時  $C(f)=0$ 。收到的訊號如下式(3-2)表示

$$r(t) = \sum_{n=0}^{\infty} I_n h(t - nT) + z(t) \quad (3-2)$$

其中

$$h(t) = \int_{-\infty}^{\infty} g(\tau) c(t - \tau) d\tau \quad (3-3)$$

$z(t)$  為可加成性的高斯白雜訊。取樣速率為  $1/T$  sample/s。現在有一個接收濾波器  $p(t)$ 。將收到的訊號通過濾波器  $p(t)$ ，得到

$$y(t) = \sum_{n=0}^{\infty} I_n x(t - nT) + z'(t) \quad (3-4)$$

$x(t)$  為  $g(t)$ 、 $c(t)$  與  $p(t)$  之迴旋積(convolution)， $z'(t)$  是雜訊  $z(t)$  與  $p(t)$  之迴旋積(convolution)。現在必須將  $y(t)$  以  $1/T$  sample/s 的取樣速度做取樣。即在  $t = kT + \tau_0$ ， $k = 0, 1, \dots$ ，做取樣， $\tau_0$  是取樣誤差。將得到：

$$y(kT + \tau_0) \equiv \sum_{n=0}^{\infty} I_n x(kT - nT + \tau_0) + z'(kT + \tau_0) \quad (3-5)$$

或將式(3-5)可表示為

$$y_k \equiv \sum_{n=0}^{\infty} I_n x_{k-n} + z'_k, \quad k = 0, 1, \dots \quad (3-6)$$

再將式(3-6)改寫為

$$y_k = x_0 \left( I_k + \frac{1}{x_0} \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} \right) + z'_k, \quad k = 0, 1, \dots \quad (3-7)$$

我們可以把  $x_0$  設為 1，得到：

$$y_k = I_k + \sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} + z'_k, \quad k = 0, 1, \dots \quad (3-8)$$

在式(3-8)我們最後看到等號的右邊有三項，其中  $I_k$  為欲得到的訊號資訊，第二

項為  $\sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n}$  稱為 ISI。第三項則是雜訊項。

我們觀察 ISI 項，發現這個干擾的訊號是來自之前所傳的訊號，以前的訊號會干擾後面要傳的訊號，也是為什麼稱之為碼際干擾(ISI)。然而要去除 ISI 項，

必須想辦法令  $\sum_{\substack{n=0 \\ n \neq k}}^{\infty} I_n x_{k-n} = 0$ ，如果令：

$$x(t = kT) \equiv x_k = \begin{cases} 1, (k = 0) \\ 0, (k \neq 0) \end{cases} \quad (3-9)$$

就可以使 ISI 項為零了。這個稱為奈奎斯(Nyquist)準則[3,13]。根據這個準則我們求得  $p(t)$  可以補償通道造成的 ISI 影響。

### 3.1.1 線性等化器(Linear Equalizer, LEQ)

現在我們介紹一個線性的等化器，如圖所示，它是由一個 tapped delay line 構成的濾波器，為一個有限脈衝響應濾波器。

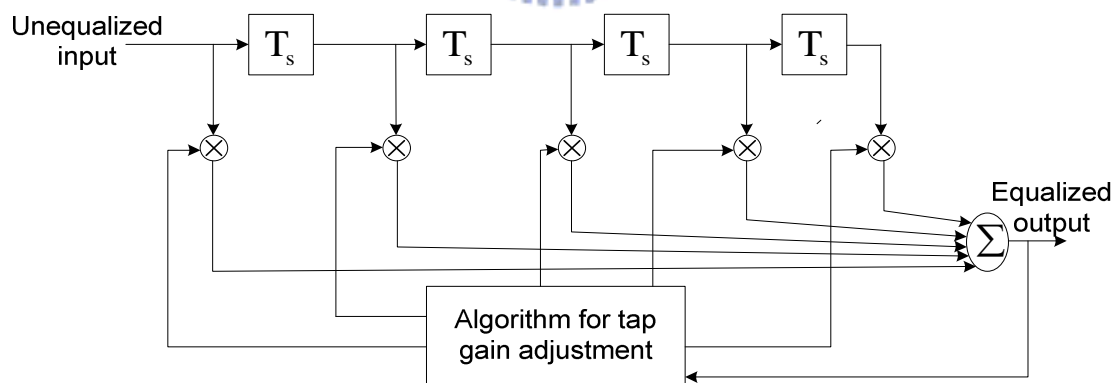


圖 3-1：線性等化器

圖 3-1 中線性等化器的輸入為接收訊號  $r(t)$ ，經過等化器後的輸出應為  $I_k$ 。而等化器的係數就是我們必須求出來的。基本上，線性等化器有兩種求法，zero-forcing[3,13]與均方誤差(Mean-Square-Error, MSE)準則[3,13]。我們將對這

兩種做法做簡單的介紹。

- Zero-Forcing :

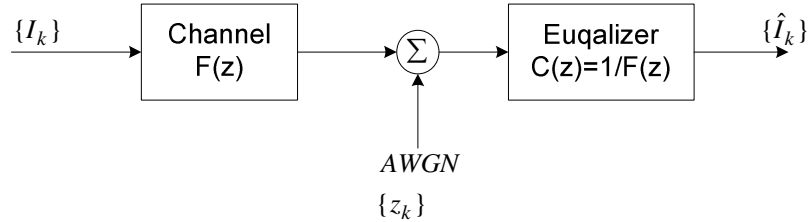


圖 3-2：通道與 Zero-forcing 等化器

如圖 3-2，通道  $F(z)$  及等化器  $C(z)$  已被合併成一個濾波器，得到下式脈衝響應

$$q_n = \sum_{j=-\infty}^{\infty} c_j f_{n-j} \quad (3-10)$$

根據奈奎斯準則如欲消除 ISI 項時，則必須令

$$q_n = \sum_{j=-\infty}^{\infty} c_j f_{n-j} = \begin{cases} 1, (n = 0) \\ 0, (n \neq 0) \end{cases} \quad (3-11)$$

將上式(3-11)做  $z$  轉換得到

$$Q(z) = C(z)F(z) = 1 \quad (3-12)$$

所以得到圖 3-2 中的等化器  $C(z) = \frac{1}{F(z)}$ ，得到了等化器  $C(z)$ ，因為它強制令所有的 ISI 項為零故以 Zero-forcing 稱之。在頻譜上這個等化器是  $F(z)$  的倒數，所以在通道有很大衰減的頻帶上將會有雜訊放大的缺點存在。

- MSE 準則：

在 MSE 準則中，我們欲得到等化器的係數  $\{c_k\}$ ，使的估計誤差  $\varepsilon_k = I_k - \hat{I}_k$  有最小均方誤差。令均方誤差為  $J$ ，

$$J = E\left|\varepsilon_k\right|^2 \quad (3-13)$$

我們欲得到最小的均方誤差為  $J$ ，根據 MSE 準則，得到了等化器  $C(z)$  的轉移函數[3,13]：

$$C(z) = \frac{F^*(z^{-1})}{F(z)F^*(z^{-1}) + N_0} \quad (3-14)$$

其中  $N_0$  為雜訊頻譜密度。如果  $N_0$  非常小時，比較 Zero-forcing 準則與 MSE 準則的等化器，兩者是非常相似的。相反的，如果  $N_0$  不為零時，MSE 準則的等化器的輸出會有殘餘的 ISI 項，在這裡我們並不討論這些殘餘量。

### 3.1.2 Fractional Spaced Equalizer (FSE)

前面敘述的線性等化器都是 baud-spaced 的等化器，現在我們說明 FSE 的一些基本原理[13]。FSE 的架構與 baud-spaced 等化器很相似，差別在於其取樣頻率的不同處。其 delay line taps 之間隔小於 baud-spaced  $T_s$ ，在數位濾波器容易實現的條件下通常取其間隔為  $T_f = T/M$ ， $M$  為如 2、3、4 等之整數。

傳送的訊號經過不完美通道後以 Baud-space 速率取樣會有交疊現象，FSE 利用  $T_f = T/M$  的速率作過度取樣是為了避免交疊現象(alias phenomenon)。

### 3.1.3 決策回授等化器(Decision Feedback Equalization, DFE)

圖 3-3 中為一決策回授等化器[3,13]，由前饋濾波器(前饋濾波器  $a(z)$  由 ZF-LEQ  $c(z)$  與 Whitening  $l+p(z)$  濾波器構成)、回授濾波器和決策裝置組成。由於 Zero-forcing 等化器與 MSE 準則之線性等化器有放大雜訊及殘餘 ISI 的缺點，而非線性的決策回授等化器可以克服這個缺點。如下式(3-15)為接收訊號，

$$r_k = \sum_{n=0}^{\infty} I_n h_{k-n} + z_k \quad (3-15)$$

$c(z)$  為 Zero-forcing 等化器，其輸出為：

$$r'_k = \hat{I}_k + z''_k \quad (3-16)$$

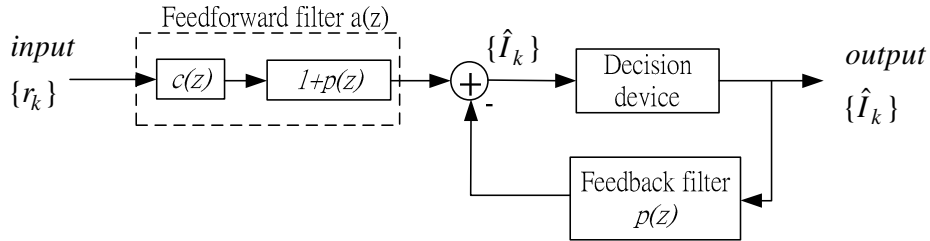


圖 3-3：決策回授濾波器

式(3-16)中  $r'_k$  為經等化後的傳送訊號項與雜訊項，也就是 ISI 項已經被消去了。其中  $z''_k$  是 Zero-forcing 等化器  $c(z)$  的輸出為一彩色雜訊(color noise)。現在的目標是把彩色雜訊做白化(whitening)的動作，決策回授濾波器使用 predictive 濾波器來完成白化動作，可以將彩色雜訊白化成白雜訊。在  $a(z)$  的輸出得到

$$r''_k = \hat{I}_k + \sum_{n=0}^{\infty} \hat{I}_k p_{k-n} + z'''_k \quad (3-17)$$

Predictive 濾波器  $1+p(z)$  已經雜訊白化成白雜訊  $z'''_k$ ，實際上不能完全的做到 whitening，因為決策回授濾波器的輸入雜訊  $v_k$  是由匹配濾波器、線性等化器塑造出的[3]如圖 3-4，如果這個整個的模型是 AR model，也就是說、匹配濾波器、線性等化器的組合是 all pole 系統，雜訊才有可能用 linear predictive 方式完全 whitening 雜訊，否則 prediction 濾波器只能提供部分 spectral flatening 的效果。

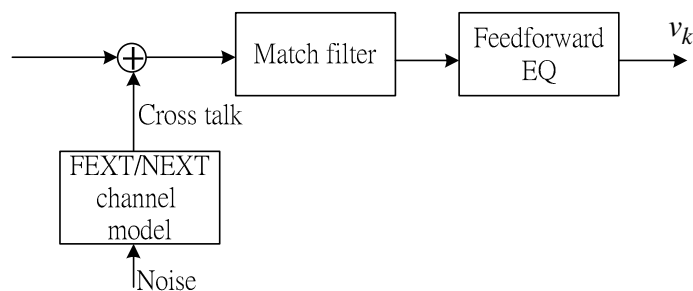


圖 3-4：雜訊模型



而式(3-17)還有  $\sum_{n=0}^{\infty} \hat{I}_k p_{k-n}$  的不預期項，必須將此項消去，利用回授濾波器

$p(z)$  可以將這個不預期項消去，剩下  $r_k'' = \hat{I}_k + z_k''$  為 decision device 的輸入，傳輸訊號已經被等化了，只剩下白雜訊的影響。對白雜訊造成的錯誤可以以通道解碼器增加編碼增益，使效能最佳化。但是決策回授濾波器存在著錯誤繁延的問題，一旦決策裝置做了錯誤決策，會造成這個現象發生，接著編碼增益的編碼增益就會降低。還有一個把決策裝置以通道解碼器實現的方式，因為解碼器做在回授迴路上，所以這個方式必須付出複雜度較高的解碼器的代價。

如果我們可以不用回授濾波器，就不會出現錯誤繁延與決策延遲的問題。有研究提出以發射器端實現回授濾波器的方法為解決方案。在 3.2 節我們將會做仔細的敘述。

### 3.1.4 適應性等化器



前面所敘述的線性等化器與決策回授等化器都是假設已知通道響應，然而，在實際的通訊系統應用中，通道的響應通常是無法預先知道的，所以有適應性等化器的產生，等化器的係數必須是可調整的能夠適應不同的通道而補償 ISI 問題。本小節將針對適應性的方法做簡單介紹。

Steepest-decent Method[13]：

為使式 3-13 中 MSE  $J$  最小化，我們將等化器係數向量表示如下式(3-18)

$$C_{k+1} = C_k - \Delta G_k, k = 0, 1, 2, \dots \quad (3-18)$$

其中 gradient vector：

$$G_k = \frac{1}{2} \frac{dJ}{dC_k} = -E(\epsilon_k V_k^*) \quad (3-19)$$

$C_k$  表示係數在第  $k$  次的疊代， $\epsilon_k = I_k - \hat{I}_k$  是第  $k$  次疊代的估計錯誤， $V_k$  則是接收訊號向量， $\Delta$  則是一個夠小的正數可以用來使上式(3-18)的疊代過程收斂。承

式(3-18)，如果最小 MSE 出現在  $k = k_0$  時， $C_{k_0} = 0$ ，此時即已達到最佳收斂值，式中的係數就不會繼續在做更新的動作了。

Least Mean Square(LMS) [13]：

$\hat{C}_k$  表示  $C_k$  的估測值， $\hat{G}_k$  代表  $G_k$  的估測值且  $\hat{G}_k = -\varepsilon_k V_k^*$ ，我們可以得到下式

$$\hat{C}_{k+1} = \hat{C}_k + \Delta \varepsilon_k V_k^* \quad (3-20)$$

此為基本的 LMS，遞迴的調整等化器之係數，LMS 的方法常常應用在商業上，如高速數據機的適應性等化器上。其基本的架構圖如下圖所示。

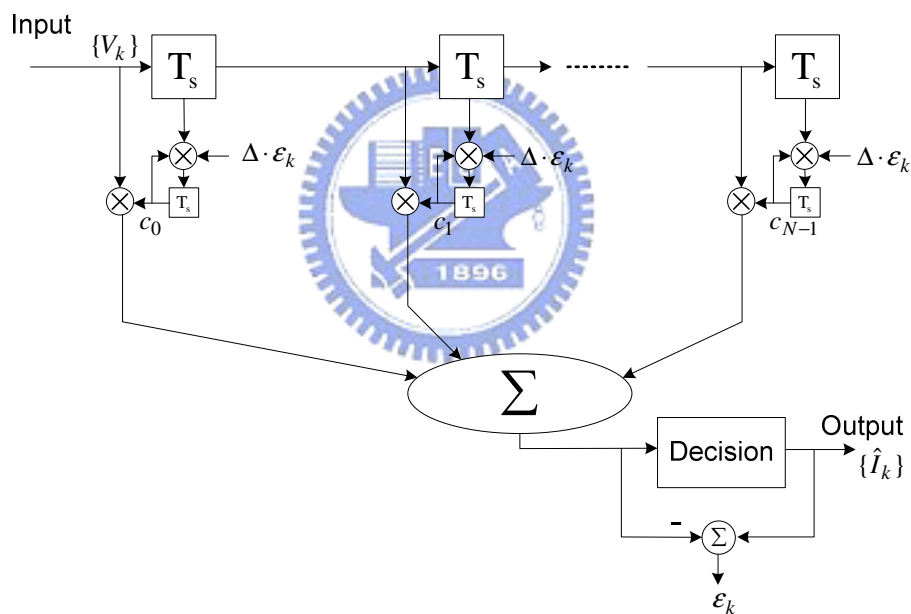


圖 3-5：線性等化器之 LMS 硬體架構

### 3.1.5 適應性決策回授等化器

誠如線性適應性等化器，決策迴授等化器的前饋濾波器及回授濾波器同理可以應用 LMS 同樣的方法求出係數。

$$\hat{C}_{k+1} = \hat{C}_k + \Delta \varepsilon_k V_k^* \quad (3-21)$$

$\hat{C}_k$  表示第 k 次的疊代的前饋及回授濾波器的係數向量， $\epsilon_k = I_k - \hat{I}_k$  是第 k 次疊代的估計錯誤， $V_k$  則是前饋濾波器及回授濾波器之輸入訊號向量  $\{v_k v_{k+1} \dots v_{k+N-1} I_k I_{k+1} \dots I_{k+M-1}\}$ ， $\Delta$  則是一個用來使上式的疊代過程可以收斂的正數。

## 3.2 Tomlinson-Harashima Precoder (THP)

決策回授等化器(DFE)比線性等化器具有較強消除碼際干擾效應之能力，然而卻存在著錯誤決策發生的可能性，造成整體效能下降，還有回授濾波器造成的決策延遲(decision delay)使的解碼器製作上的困難存在。如果我們可以預先得知通道響應(channel response)的情況下，就有可能把等化器的回授濾波器部分做在發射器(transmitter)端，在通道響應不會變化的很快應用就有可以採用這樣的做法，例如有線的傳輸(xDSL)。Tomlinson-Harashima Precoding 就是把等化器以發射器端實現的系統，可以等效於 DFE 的功能。它在 1968-1969 年同時間被 Tomlinson 及 Harashima 不約而同發明的，有人稱為 Tomlinson Precoding，也有人稱它 Tomlinson-Harashima Precoding(THP)，在本論文中我們以 THP 簡稱。

### 3.2.1 THP 基本原理

THP 的架構如圖 3-6 所示。等效的離散通道響應為  $h(z)$ ，假設已經知道通道響應  $h(z)$  並且 THP 的係數  $p(z)$  已經求得。令  $i_k$  為發射器(Tx)端要傳的訊號，會先經過 Precoder 的處理減去  $\sum_j p_j x_{k-j}$  項，由於回授濾波器(Precoder)會使傳送訊號的能量無限制的增大，所以有一個做模數 2L(Module 2L)運算的電路將傳送訊號限制在 -L 到 L 之間的範圍，其中 L 為訊號  $i_k$  的訊號最大正值。

因此，可以在 Mod2L 的電路後得到，其中 N 為 precoder 的係數長度

$$x_k = i_k - \sum_{j=0}^{N-1} p_j x_{k-j} \text{ module } 2L \quad (3-22)$$

$$x_k \in (-L, L]$$

我們可以找到一個整數  $z_k$  使的  $x_k$  滿足上式(3-22)的限制範圍，成為

$$x_k = i_k - \sum_{j=0}^{N-1} p_j x_{k-j} + 2Lz_k \quad (3-23)$$

把  $x_k$  限制在  $-L$  至  $L$  之間的範圍內，傳送訊號能量就會限制在固定範圍內。將  $x_k$  經過通道的傳輸後，接收訊號經過 ZF-LEQ  $c(z)$  得到之輸出為：

$$x_k = i_k - \sum_{j=0}^{N-1} p_j x_{k-j} + 2Lz_k + n'_k \quad (3-24)$$

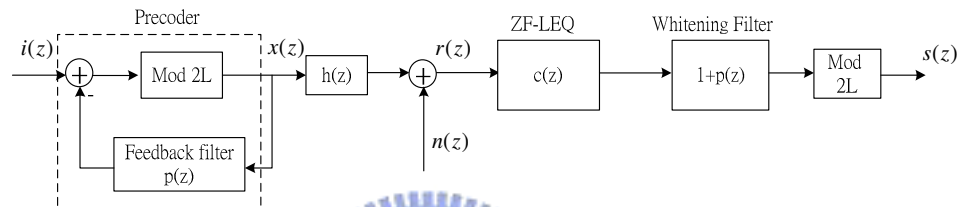


圖 3-6：THP 系統

與上式比較多出了最後一項雜訊  $n'_k$  且仍為一彩色雜訊(color noise)序列。再經過 whitening 濾波器  $1+p(z)$  將彩色雜訊轉換成白雜訊，我們可以得到  $1+p(z)$  的輸出為，

$$v_k = x_k + \sum_{j=0}^{N-1} p_j x_{k-j} + n''_k \quad (3-25)$$

其中：

$$n''_k = z'_k + \sum_{j=0}^{N-1} p_j z'_{k-j} \quad (3-26)$$

經過了 whitening 濾波器，彩色雜訊已經被轉換成了白雜訊， $n''_k$  則為白雜訊序列。

由前面(3-23)、(3-25)兩式可以解出  $v_k$ ：

$$v_k = i_k + 2Lz_k + n''_k \quad (3-27)$$

觀察  $v_k$ ，它是一個週期為  $2L$  的展開的數值再加上白雜訊項，經過模數  $2L$  電路消去  $2Lz_k$  項後得到：

$$S_k = \text{mod}[i_k + n_k''] \quad (3-28)$$

$S_k$  是 THP 系統最後欲得到的項，它單純的只有看到白雜訊及傳輸的  $i_k$  兩項。如果只是處理白雜訊，接上 TCM 解碼器後就可以得到直接的編碼增益，這是因為在統計上白雜訊與訊號是無相關性的，在分析不但容易且 TCM 解碼器對白雜訊也有較佳的抵抗能力。

### 3.2.2 資料跳動(data flipping)現象

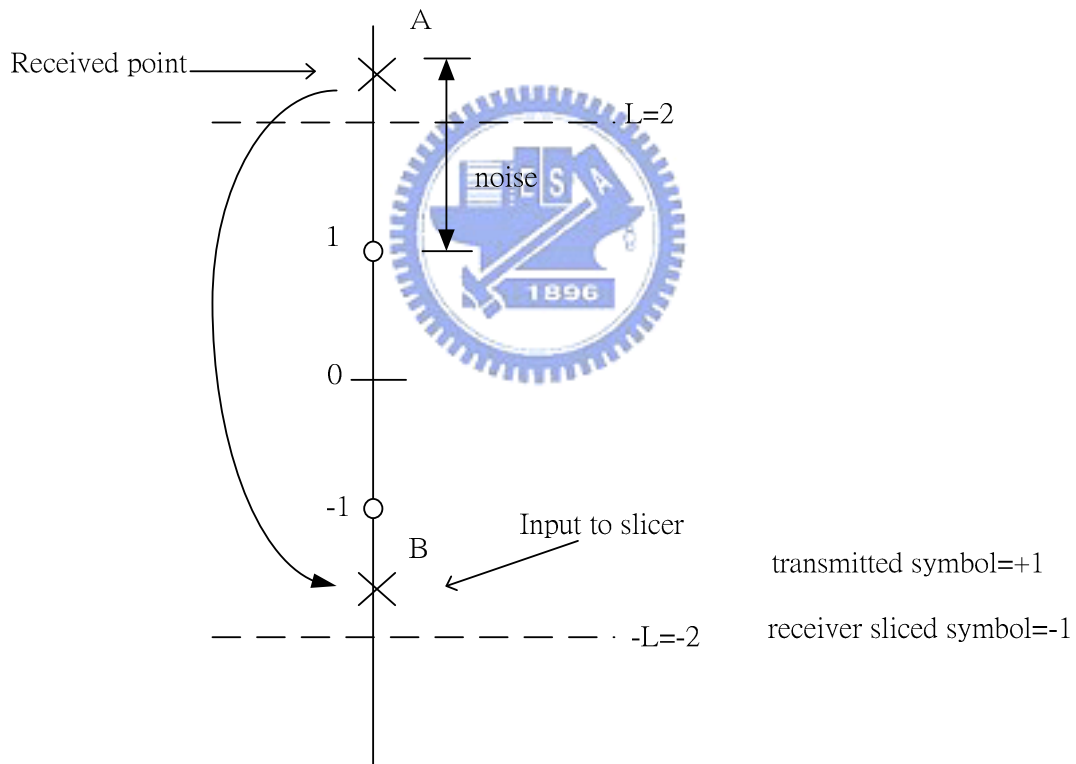


圖 3-7：資料跳動現象

承式(3-28)

$$S_k = \text{mod}[i_k + n_k''] \quad (3-29)$$

假如  $i_k + n_k''$  落在  $(-L, L]$  範圍之外的條件成立的話，也就是

$$|i_k + n_k''| > L \quad (3-30)$$

取模數  $2L$  的電路就會把序列  $i_k + n_k''$  對映至  $(-L, L]$  範圍內，此時錯誤就會發生了。這個現象稱為資料跳動(data flipping)[10]。資料跳動容易造成誤判，如此一來會使的錯誤率增加，因為 THP 使用取模數(module) $2L$  造成資料跳動的現象，所以在解碼器或決策裝置的設計上必須要克服這個問題。

現在簡單的說明資料跳動如何的造成錯誤。我們看到圖 3-7 所示，假設我們傳送訊號為+1，接收時因為有雜訊干擾，所以接收到的訊號為 A。因為雜訊干擾使它的大小值超出了  $(-L, L]$  的範圍之外必須取模數  $2L$  將訊號 A 對映至  $(-L, L]$  之內，訊號 A 取模數  $2L$  之後我們得到了訊號 B，訊號 A 被已經對映進  $(-L, L]$  的範圍之內了。如果將訊號 B 送進決策裝置，得到之輸出為-1。最後的解為-1，與原始傳送的值不同，所以誤判的錯誤在這個時候發生了。

資料跳動現象會造成 THP 系統及通道編碼兩者的效能下降，必須予以補償，才不致使資料跳動現象影響 THP 系統及通道編碼兩者效能。3.2.3 節將敘述解決此問題的方法。



### 3.2.3 資料跳動(data flipping)現象之解決方法

前面敘述過資料跳動會使 THP 系統及通道編碼兩者效能下降，現在我們敘述一個解決此問題的方法，有效的避免了資料跳動的問題。以 4-PAM 的星雲圖為例子，如圖 3-8 所示，我們看到之前所述資料跳動現象發生， $r$  表示接收到的訊號，因為有雜訊影響使它超出  $(-L, L]$  的範圍外，將  $r$  取模數  $2L$  之後得到訊號  $r'$ ，訊號  $r$  則被對映至  $(-L, L]$  內了。

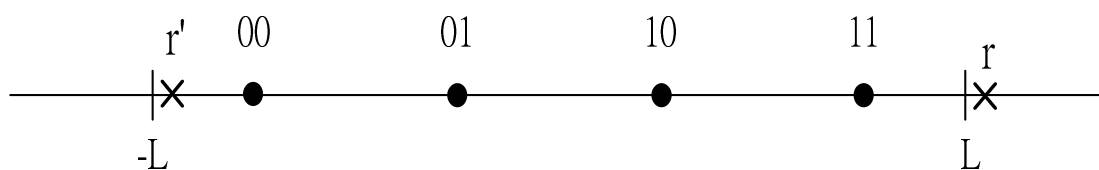


圖 3-8：4-PAM 星雲圖

如果要解決資料跳動問題必須把原始的訊號星雲圖擴展，所以我們複製 4-PAM 星雲圖{00, 11}的兩個訊號，星雲圖擴展成如下圖 3-9。

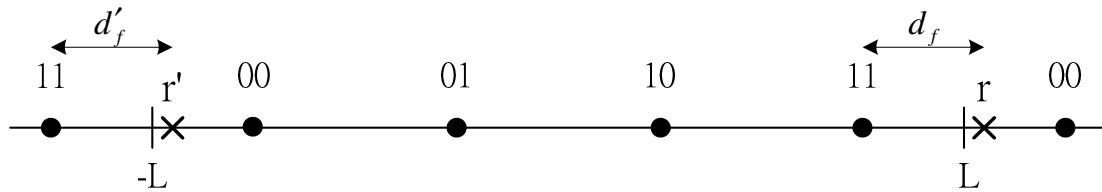


圖 3-9：擴展後之 4-PAM 星雲圖

現在星雲圖已經被擴展多了最外圍的訊號，做法為將訊號{11, 00}分別複製在右邊、左邊位移  $2L$  的位置，就可以得到擴展的星雲圖。很清楚了看到原本接收的訊號  $r$  與訊號 11 的 transition metric 是  $d_f$ ，如果訊號  $r$  取了模數  $2L$  之後的 transition metric  $d_f'$  也等於  $d_f$ ，資料跳動的錯誤就不會發生了。如圖 3-9，我們已經把訊號星雲圖擴展多了外圍的兩個訊號{11, 00}。算出  $r$ ' 的 transition metric 為  $d_f'$ ，應該會等於  $d_f$ 。

如果  $d_f = d_f'$ ，即 transition metric 的計算無誤，剩下計算最大可能性的工作就是維特比解碼器的工作，維特比演算法將在第四章介紹。根據擴展星雲圖的方法，TCM 解碼器計算 transition metric 的設計將需要做修正。

---

## 第 4 章 格狀編碼調變(TCM)解碼器

---

由於在傳輸環境受限制下，所有的訊號都要盡可能的以較小的能量傳輸，以達到所需要的效能。在傳輸功率受限制下並且要達到制定的效能，使用通道編碼則是一種有效的方法，藉由將傳輸的資料序列中加入冗餘(redundancy)的位元，用來保護需要傳輸的資料序列，可以增加編碼增益，此時的功率效益就可以提升。Viterbi[14,15]演算法是一種最大可能性法則的解碼方式，已經被應用在許多通訊系統中，如行動通訊、數位視訊廣播系統與數據機等。如果在低位元速率的應用中，Viterbi 解碼器可以 DSPs 實現。相反的，如果在高速的應用時，Viterbi 解碼器需以 VLSI 技術實現專門處理 Viterbi 解碼的硬體。在本章會介紹 TCM[6,7,8]與 Viterbi 演算法，可以了解 TCM 編碼如何運作與 Viterbi 演算法的解碼方法。

### 4.1 格狀編碼調變(TCM)

由於通道編碼在編碼時，常會加入許多多餘的位元(redundancy)來做保護，以碼率  $R$  的編碼器而言，假如未編碼時的傳輸速率為  $R_s$ ，如果必須傳輸相同的資料量時，編碼後則比未編碼時需要  $1/R$  倍的頻寬，在有限頻寬條件下時是不可能達到通道編碼的目標。

而 TCM 則提供了編碼及調變的技術，可以在有限頻寬的限制下提供如未編碼時相同的傳輸速度，如此一來可以避免增加頻寬。下列三步驟為 TCM 編碼及調變：

1. 增加一位元編碼的 Redundancy 至每  $m$  個位元。
2. 將訊號星雲圖從  $2^m$  個訊號增加至  $2^{m+1}$  個訊號。
3. 利用集合劃分方式將  $m+1$  位元訊號星雲圖上的每個訊號做編碼。

第三步驟中利用集合劃分(set partition)的技術將訊號分成多個不同的子集合，如圖 4-1 為 8-PSK 的一個例子，此目的為增加訊號的最小尤拉距離(minimum



Euclidean distance)以增進效能。

迴旋碼常被應用在 TCM 上，它提供通道編碼的功能，藉由加入某些位元 (redundancy)，促使接收器有偵測錯誤的可能性。

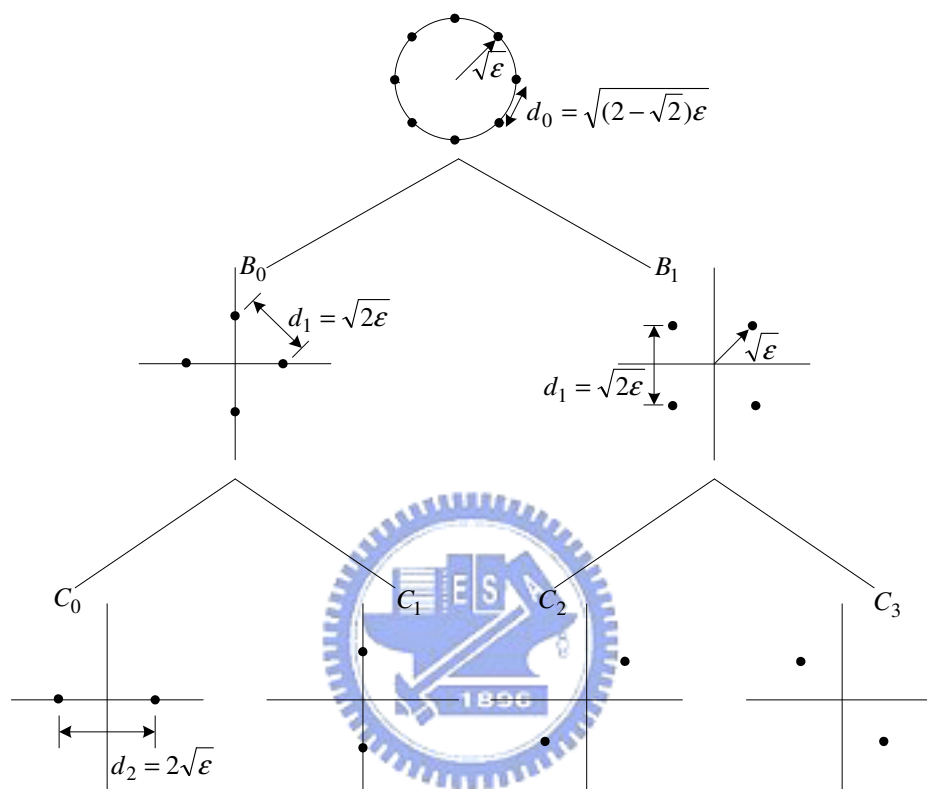


圖 4-1：8-PSK 訊號集合劃分

## 4.2 TCM 解碼器

圖 4-2 為 G.SHDSL 採用之格狀編碼調變器，TCM 之編碼器為碼率  $R=1/2$ ，限制長度  $K=10$  之迴旋碼編碼器，產生多項式(generator polynomials)分別為  $G1=\{0101101110\}$ 、 $G0=\{1100110001\}$  的迴旋編碼器做通道編碼(channel coding)，訊號為 16-PAM 調變傳送。如圖 4-3 所示，這是 16-PAM 的訊號星雲圖(signal constellation)。16-PAM 的訊號被分成 4 個訊號集合  $C0$ 、 $C1$ 、 $C2$  與  $C3$ ，每個訊號集合中都有四個訊號。訊號集合之選擇由  $c_1c_0$  兩位元決定，當  $c_1c_0=00$  時將選擇集合  $C0$ ， $c_1c_0=01$  時將選擇集合  $C1$ ， $c_1c_0=10$  時將選擇集合  $C2$ ， $c_1c_0=11$  時將

選擇集合 C3。

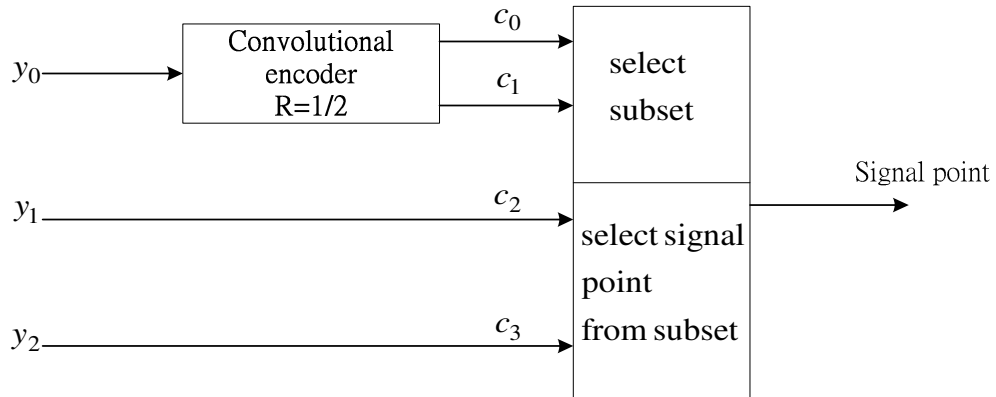


圖 4-2：迴旋編碼器為 K=10，R=1/2 之 TCM 方塊圖

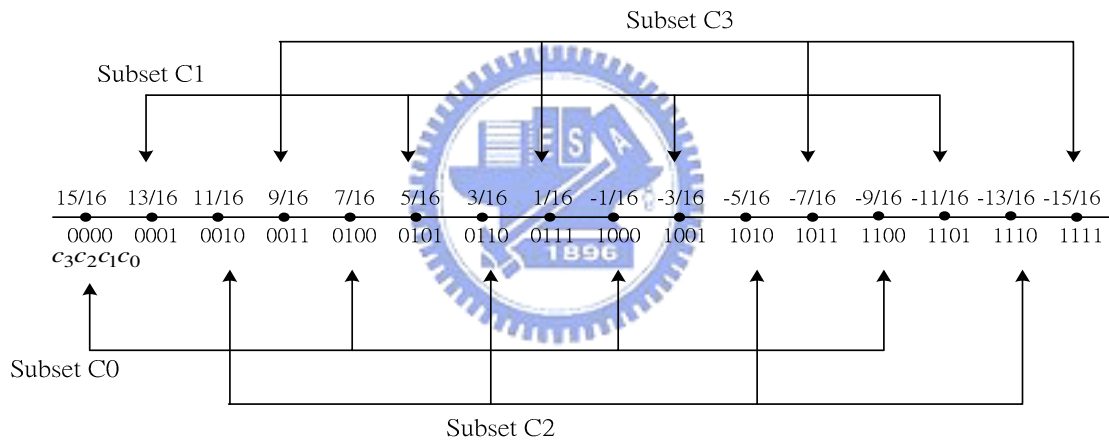


圖 4-3：16-PAM 信號星雲圖

### 4.2.1 Viterbi 演算法

Viterbi 演算法早在 1967 就被提出用來做迴旋碼(Convolutional code)解碼之用。Viterbi 演算法在 VLSI 上設計也可以很容易的被實現，所以在通訊系統中已成為一個不可缺少的裝置，在許多通訊標準中已經是標準配備。然而 SHDSL 中使用的 TCM 中也有用到迴旋碼，同樣 Viterbi 也適合用來解 TCM。以下我們介紹 Viterbi 演算法。我們令解碼器接收到訊號：

$$y_k = c_k + n_k \quad (4-1)$$

$c_k$  為經迴旋碼編碼後欲傳送訊號， $n_k$  為可加成性高斯白雜訊。要從接收訊號序列  $Y$  求出最佳傳送序列  $\hat{C}$ ，可利用最大可能性序列估測(Maximum Likelihood Sequence Estimation)法則求出。假設傳送序列  $C$  收到訊列  $Y$  時的最大可能性函數為  $P(Y|C)$ ，則可得  $\hat{C}$ ：

$$\hat{C} = \arg \left\{ \max_{\text{all sequences } C} P(Y|C) \right\} \quad (4-2)$$

因為雜訊樣本為統計獨立，所以可以分解最大可能性函數為：

$$P(Y|C) = \prod_{k=0}^{D-1} P(y_k | c_k) \quad (4-3)$$

$P(y_k | c_k)$  為在傳送  $c_k$  下接收到  $y_k$  的條件機率密度函數，而我們必須求出  $P(y_k | c_k)$  的值，先取對數(log)可簡化如：

$$BM_k = \log(P(y_k | c_k)) \sim |y_k - c_k|^2 \quad (4-4)$$

我們稱  $BM_k$  為分支計量(branch metric)，它是  $y_k$  與  $c_k$  之間的歐式距離(Euclidean distance)。我們藉由算出最小歐式距離找出最大可能性序列：

$$\hat{C} = \arg \left\{ \min_{\text{all sequences } C} \sum_{k=0}^{D-1} |y_k - c_k|^2 \right\} \quad (4-5)$$

Viterbi 演算法藉由遞迴的算出最小歐式距離可以找出最大可能性序列  $\hat{C}$ 。此演算法根據最大可能性(maximum likelihood)的法則，算出最大可能性的決策(decision)，計算出每個訊號的每個決策可能性(likelihood)的值或者距離(distance)，這個值或距離定義為 branch metric。然後循著可能路徑作累加，會得到每一個可能路徑的 metric，從多個路徑中找出最大可能性的一條路徑，而這條路徑稱為存活路徑(survivor path)，沿著這條存活路徑的尋找(trace)回去則可以找出我們要所要解的最佳決策(decision)。

以下整理出 Viterbi 演算法的解碼程序：

1. 接收到訊號。
2. 根據格子圖(trellis diagram)算出訊號與每一條可能走的分支(branch)的距離

(branch metric)。(BMU)。

3. 在每一個狀態(state)上比較出可能的生存路徑(survivor path)。(ACSU)。
4. 比較所有的狀態的 metric，找出最大可能性的生存者(survivor)。(SMU)。
5. 根據步驟四，找出的生存路徑解出最佳決策(decision)。

在步驟 2、3 與 4 後面註明了該步驟會由那一個執行單元運算，將在下節說明。

## 4.2.2 Viterbi 解碼方法

Viterbi 演算法可以分成 BMU、ACSU 與 SMU 三個部分的運算。

### 4.2.2.1 Branch Metric Unit (BMU)

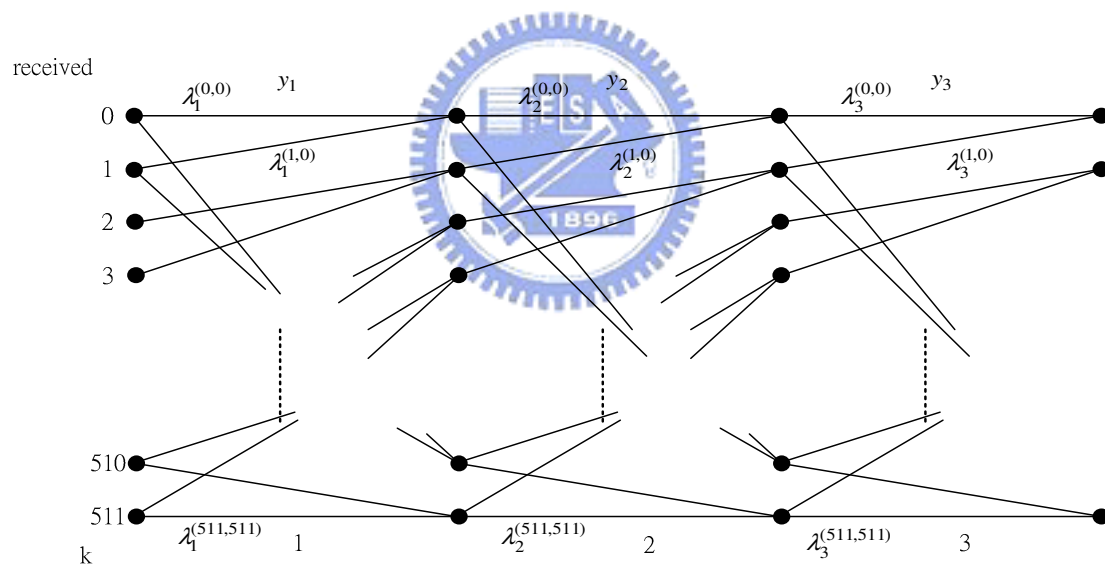


圖 4-4：512 個狀態之格子圖

圖 4-4 為圖 4-2 TCM 方塊之狀態  $N=512$  對映格子圖，頂端標示了接收到的訊號  $y_k$ ，底端  $k$  表示時間。當我們每次接收到取樣訊號都要計算出 1024 條分支的分支計量  $\lambda_k^{(i,j)}$ ， $(i,j)$  表示由狀態  $i$  至狀態  $j$  的分支。如圖 4-4 所標示符號。當計算出分支計量後必須送至 ACSU 求狀態計量(state metric)之用。

### 4.2.2.2 Add、Compare and Select Unit

當 ACSU 收到 BMU 的分支計量之後，還要算出狀態計量(或稱路徑計量，path metric)。每個狀態會有兩條分支的路徑合併進來，如圖 4-5，而 ACSU 的動作就是比較這兩條狀態計量的大小，選出較小的狀態計量，並把選擇的結果紀錄下來，紀錄在存活記憶體(survivor memory)，也就是存在 SMU 中。做了選擇後還必須把選擇的狀態計量存在記憶體。如圖所示為兩條路徑合併的是意圖。下面的路徑為存活路徑，所以求出了狀態  $j$  在時間  $k+1$  的狀態計量：

$$\gamma_{j,k+1} = \min(\gamma_{k+1}^{(i_0,j)}, \gamma_{k+1}^{(i_1,j)}) = \gamma_{k+1}^{(i_1,j)} \quad (4-6)$$

還會再求出此狀態的選擇訊號，參考圖 4-5，當上面的路徑為存活則  $decision=0$ ，反之  $decision=1$ 。ACSU 會將  $decision$  送至在 SMU 中做解碼之用。

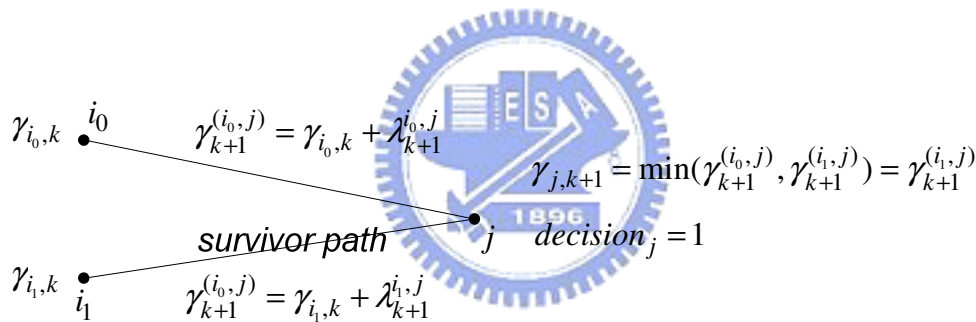


圖 4-5：ACS 遞迴式運算

ACSU 還要將最佳狀態(最小之狀態計量)求出，將狀態值告訴 SMU，可做追蹤存活路徑之用。

### 4.2.2.3 Survivor Memory Unit

要說明 SMU 的解碼功能之前我們先說明存活路徑深度的觀念(survivor depth)，可以幫助對 SMU 的了解。解碼器有實現終止處(termination)與不實現終止處兩種做法。有終止處時，格子圖的起始與終止狀態都是已知，這個時候的解碼長度(survivor depth)決定於起始與終止間的長度，而 SMU 的硬體大小會正比於這個長度。但是在有些應用會有連續序列，不知道何時才會有終止，如廣播、

SHDSL 等，它們沒有終止處，所以不知道終止狀態為何，本論文就是這個做法。然而存活長度必須被決定，假如有兩條在時間  $k$  的最佳狀態合併進來的路徑，若追蹤回去到時間  $k-D$  時，這兩條路徑會有相當大的機率合併在一起。此時的  $D$  稱為存活深度，這個特性可以保證我們的最佳狀態解碼的正確性，而一般存活深度大都取 5 倍的限制長度即  $D=5K$ [8]。回到 SMU 解碼程序，我們從 ACSU 得到每一級的 *decision* 與最佳狀態，而 SMU 已經紀錄了每一級的 *decision*，這些 *decision* 代表了所有的存活路徑，我們只要知道了最佳狀態就可以往回追蹤出解碼資訊。

SMU 常見的做法有暫存器交換(Register Exchange Algorithm, REA)與 Trace-back 方法兩種做法。我們以下針對此兩種不同方法做介紹。

- TBA：TBA 相較於 REA 因為需要往回追蹤(trace-back)則會有較大的潛伏期為  $D+M$  其  $M$  為往回追蹤時所需的額外時間，所以  $M$  與  $D$  有直接關係，一般  $M=D$ 。通常 TBA 不需要較大的記憶體存取頻寬，所以相對地較節省功率。
- REA：通常暫存器交換的做法是採用大量的暫存器，這些暫存器用來儲存生存路徑的資訊，而暫存器大小與存活深度  $D$  與迴旋編碼器之限制長度  $K$  有關，當  $D$ 、 $K$  決定之後則需要  $D \cdot 2^K$  位元的暫存器大小。使用 REA 的優點在於其解碼速度較快，其潛伏期(latency)為  $D$ 。

本論文採用了 REA 做為 SMU 之硬體架構，因為 REA 具有低潛伏期之優點，且硬體設計上也具有規則性，有容易設計之優點。我們修改 REA，不使用暫存器架構，以 memory-based 實現，避免用大量暫存器實現。只需規劃記憶體之擷取位址就可以做到 REA。

### 4.3 結合 THP 系統與 TCM 解碼器

第三章說明了 THP 系統的基本原理，當等化器把通道造成的 ISI 問題解決了之後，剩下白雜訊，同樣是造成錯誤率增加的原因，所以將 THP 系統與 TCM

解碼器結合起來可以同時得到 ISI 的補償與通道編碼的增益。其運作模型如下。

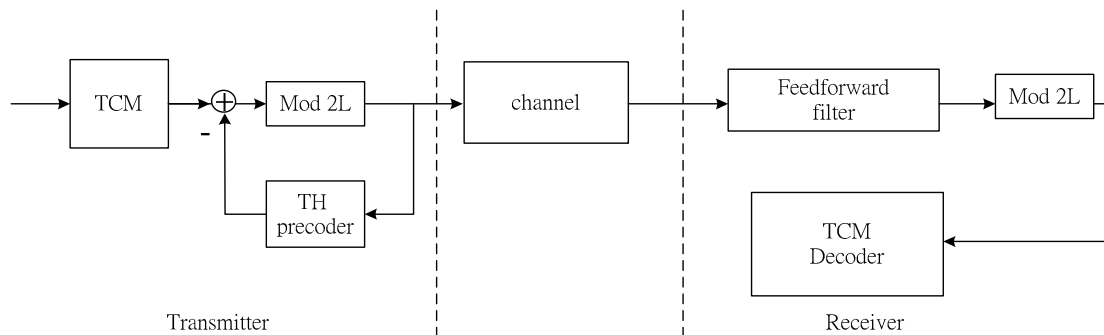


圖 4-6：THP 系統與 TCM 解碼器

發射器端有格狀編碼調變器與 TH precoder，要傳送的訊號會先經過編碼和調變後，再以 TH precoder 預先處理才經傳送至接收機。接收機可以看到一個前饋濾波器，經過這個濾波器是做線性等化及將彩色雜訊白化的動作，再經模數 2L 的電路後就是 TCM 解碼器。TCM 解碼器則是以 Viterbi 演算法實現的一個解碼器。

圖 4-6 為等化器與 TCM 解碼器的結合，可以很容易的得到通道編碼增益。而一般的決策回授等化器要與解碼器結合，勢必要把解碼器做在決策裝置上，但是解碼器有潛伏期(latency)的問題存在仍待解決，增加了解碼器實現之困難度。將 THP 系統與常見的決策回授等化器比較，差別在 THP 系統之接收機端少了回授濾波器，它被做在發射機端。在接收機端只有一個前饋濾波器和 Mod2L 電路，將可不費工夫就可在等化器後面接上解碼器。

---

## 第 5 章 THP 系統及 TCM 解碼器之硬體架構

---

本章要介紹 THP 系統及 TCM 解碼器的硬體架構。主要是根據 G.SHDSL PMD 層的規定而定製的一個硬體架構。

### 5.1 THP 系統硬體架構

承第二章所敘述，在 activation 模式會把 Precoder 的係數送給 Precoder 濾波器使用(STU-R 傳至 STU-C 和 STU-C 傳至 STU-R)。本節中我們將 THP 系統區分成兩個部分做說明，分別為訓練模式和資料模式。

我們設計的 THP 系統是以 DFE(Decision Feedback Equalizer) 及 LMS-based 為架構做係數訓練(training)的基本硬體架構，分別有前饋濾波器及回授濾波器兩組係數。以最小均方(least mean square, LMS)的方法求出前饋濾波器及回授濾波器的係數。所有濾波器之係數及輸入資料的儲存都會使用到記憶體，因此記憶體使用上的配置需要適當的設計才不致於有衝突(指記憶體位址)發生。

進入資料模式前，發射器端會先收到由接收機送來的 precoder 濾波器的係數，完成了係數轉換後即可進入資料模式正式運作。THP 系統在資料模式下開始運作，被傳送的訊號都會經過 TC-PAM 與 channel precoder 的通道編碼與調變後送出，在接收器端有一個前饋濾波器，此前饋濾波器功能等效於 DFE 之前饋濾波器功能，此濾波器為 fractionally spaced，可使用 poly phase 方式實現省下運算量[16]。本論文的數位濾波器硬體之實現方法是由係數記憶體、資料記憶體和數個乘法累加器(Multiplier Accumulation, MAC)構成，為 memory-based，不用大量的乘法器與暫存器。至於乘法累加器的數目的決定端視資料速率(R)、系統操作頻率(M)和濾波器係數長度(N)決定。所以乘法累加器個數 $=NR/M$ 。5.1 節要說



明 THP 系統硬體架構，其設計有以下三點考量：

- 必須能夠達到 G.SHDSL 最大的傳輸速率 2.3Mbps。
- 前饋濾波器係數長度為可變。
- 前饋濾波器最大係數長度為 64，回授濾波器固定係數長度為 128。

而 SHDSL 之 STU-C 與 STU-R 的硬體設計將 THP 的硬體與 DFE 的回授濾波器要被整合在一起為一個收發機，在硬體的使用上可以做分享，如係數記憶體。

### 5.1.1 訓練模式硬體架構

圖 5-1 為 THP 系統以 DFE(Decision Feedback Equalizer) 及 LMS-based 為架構做為係數訓練(training)的基本硬體架構。圖中的前饋濾波器及回授濾波器將以適應性濾波器的架構實現，將使用 SRAM 儲存係數，係數為可調整的。

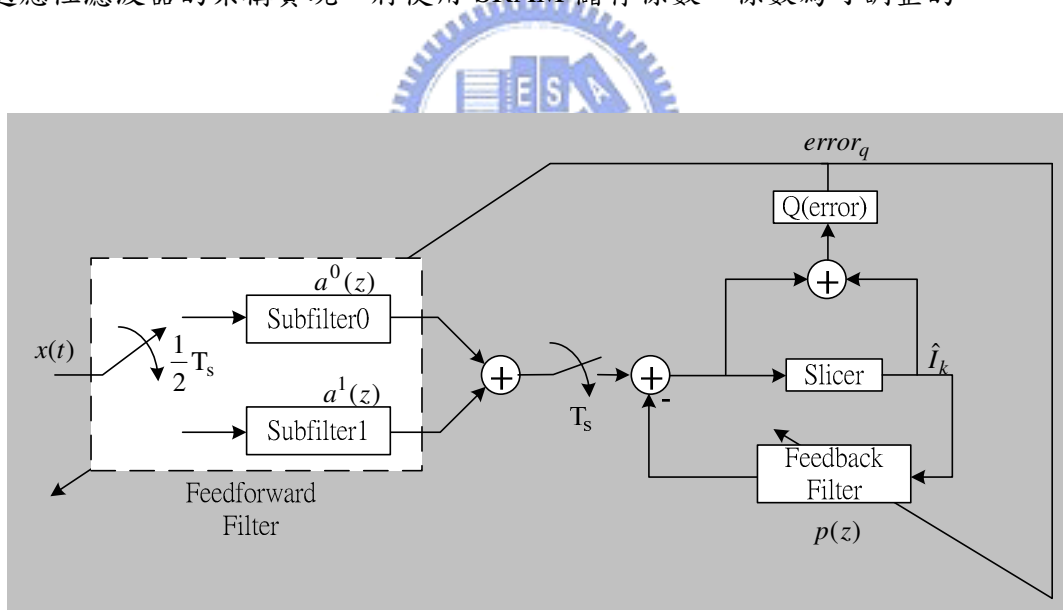


圖 5-1：訓練模式方塊圖

前饋及回授濾波器係數位元寬度將以 16 位元 2 補數法表示之，前饋濾波器係數長度為可變，最大值為 64。回授濾波器則使用 128 係數長度。輸入資料以 14 位元表示。有關濾波器係數的位元寬度與輸入資料的位元寬度，我們將以第六章的硬體模擬為依據，決定了位元寬度。係數與輸入資料所需要用到的記憶體

大小如下表 5-1：

	Feedforward Filter	Feedback Filter
Coefficient & Data Memory	16x64 & 14x64 bits	16x128 bits (Coefficient only)
Total Capacity	1,920 bits	2,048 bits

表 5-1：訓練模式記憶體容量

LMS 之係數更新方法如式 5-1：

$$W_{k+1} = W_k + \mu \cdot error_{qk} \cdot X_k \quad (5-1)$$

$W_k$ ：為前饋濾波器與回授濾波器的係數向量  $W = [\bar{w}^f \ \bar{w}^b]$ 。

$\mu$ ：為 step size，為 2 的幕次方之正數。

$error_q$ ：為量化後的錯誤估測(error estimation)，為 2 的幕次方之數值。

$X_k$ ：為前饋濾波器與回授濾波器的輸入向量， $X = [\bar{v}^f \ \bar{v}^b]$ 。

THP 系統訓練模式之硬體架構大致分成三個部分個別說明，包括(1)錯誤估測量化器與移位乘法器、(2)前饋濾波器和(3)回授濾波器。

### 5.1.1.1 錯誤估測量化器與移位乘法器

本量化器之量化位階如圖 5-2 所示，x 軸表示錯誤估測  $error$ ，y 軸表示量化後之錯誤估測  $error_q$ 。x 軸為被量化的  $error$ ，當  $x=0$  至 1 時量化結果  $error_q$  為 1。

x 軸的範圍會呈倍數 2 的方式遞增，且量化結果  $error_q$  也呈倍數 2 的方式遞增。

量化時錯誤估測  $error$  很大時其量化後的  $error_q$  值相對於  $error$  有較大的量化誤

差，相反的當錯誤估測  $error$  較小時其量化後的  $error_q$  值相對於  $error$  有較小的量

化誤差。所以在當  $error$  值很小時代表估測值較正確時而量化後的量化誤差也很

小條件下，*error* 的量化誤差並不會影響 THP 係數的收斂趨勢。



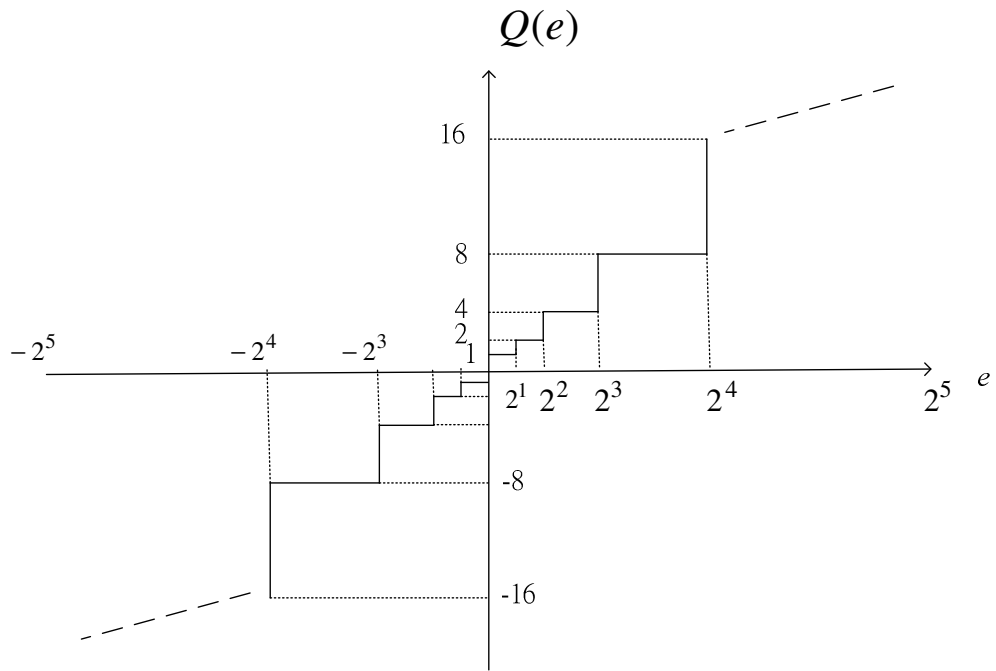


圖 5-2：錯誤估測量化圖

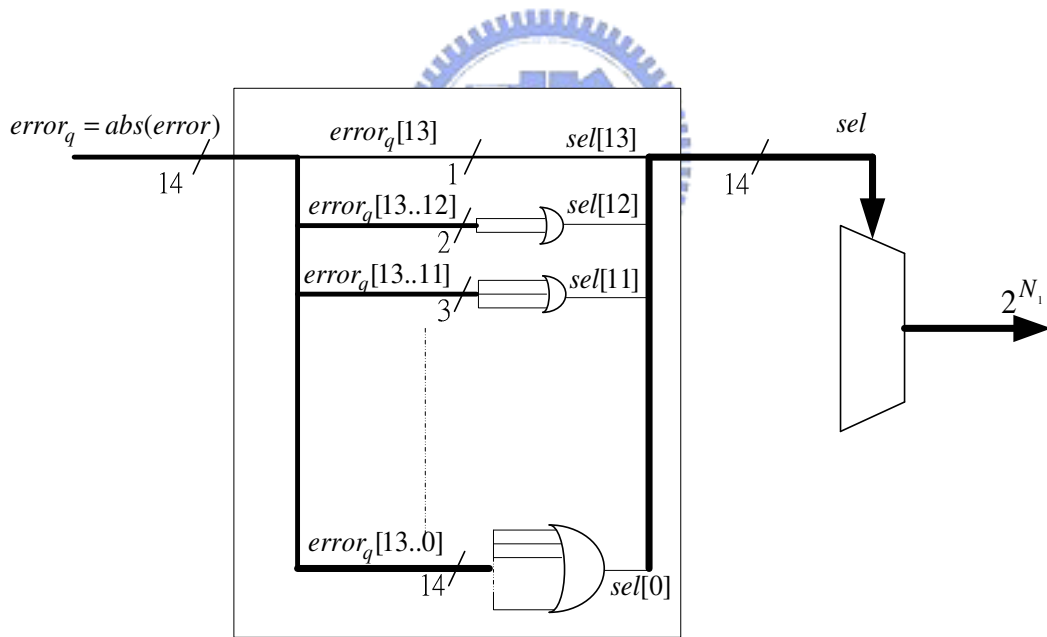


圖 5-3 (a)：錯誤估測量化器

<i>sel</i>	$2^{N_1}$
0000_0000_0000_00	1
0000_0000_0000_01	1
0000_0000_0000_11	2
0000_0000_0001_11	4
0000_0000_0011_11	8
0000_0000_0111_11	16
0000_0000_1111_11	32
0000_0001_1111_11	64
0000_0011_1111_11	128
0000_0111_1111_11	256
0000_1111_1111_11	512
0001_1111_1111_11	1024
Else	4096

圖 5-3 (b)：錯誤估測量化器真值表

根據圖 5-2 的量化位階， $error$  被量化成  $2^{N_1}$ ， $error_q = Q[error] = 2^{N_1}$ 。其硬體架構如圖 5-3，硬體架構用了 13 個或閘與一個多工器， $N_1$  為由訊號  $sel$  決定之常數。例如當  $error=811_{10}=error_{abs}=00\_0011\_0010\_1011_2$  經過圖 5-3 電路可以得到  $sel=00\_0011\_1111\_1111_2$ ， $N_1=9$ ，也就是得到量化後  $error_q=2^9$ 。

現在移位乘法器取代了式 5-1 中的乘法器，其硬體架構如圖 5-4。因為  $error_q$  與  $\mu$  皆為 2 的冪次方，所以  $\mu \cdot error_q$  的指數為  $N = N_1 + N_2$ 。  $X_k$  乘以  $\mu \cdot error_q$  的運算變成了一個移位器就可實現。原本三個數的連乘需要兩個乘法器，這需要較大的硬體面積且其 critical path 較長。現在只有一個移位乘法器與一個量化器來完成，其面積與 critical path 都小於兩個乘法器。

根據式 5-1，前饋濾波器與回授濾波器都需要做係數調整，所以會用到兩個移位乘法器，其架構如圖 5-4。

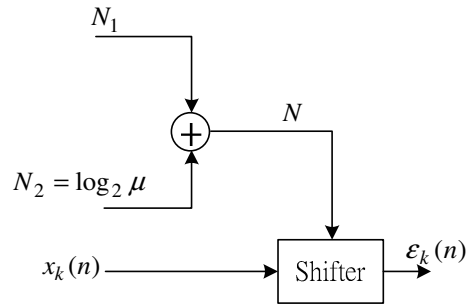


圖 5-4：移位乘法器

### 5.1.1.2 前饋濾波器

圖 5-5 前饋濾波器硬體架構圖，為 poly-phase 架構[16]，此硬體架構包括了一個資料記憶體、兩組係數記憶體，記憶體總容量如表 5-1 所示為 1920 bits。使用一組乘法累加器作迴旋積(convolution sum)運算之用。還有一個加法器作為係數更新(update)之用。

前面說過了乘法累加器的個數決定有三個要素，既然我們在前饋濾波器只使用了一個乘法累加器，既要符合最大傳輸速率 2.3Mbps(symbol rate=2.3/3 M·sym/sec)與最大係數長度 64，所以系統操作頻率至少為

$$64 \div \frac{3}{2.3} = 49.07 \text{ MHz}。$$

現在說明記憶體分配工作，分成資料記憶體及係數記憶體。資料記憶體分配比較簡單，考慮到前饋濾波器為一個 fractional space equalizer 其間隔為 T/2，而我們只需要把一塊資料記憶體空間依記憶體位址區分成兩部分，分為單數位址和雙數位址給兩個子濾波器個別使用，如資料記憶體有 64 個位址，對映之位址 0、2、4 至 62 供 subfilter0 存放 32 筆的輸入資料，對映之位址 1、3、5 至 63 供 subfilter1 存放 32 筆輸入資料。

至於係數記憶體，我們已經將其分成兩組係數記憶體 M0、M1，每組 32 個

位址係數儲存空間，如圖 5-1 示。令前饋濾波器的兩個子濾波器  $a^0(z) = \sum_{j=0}^{31} a_j^0 z^{-j}$

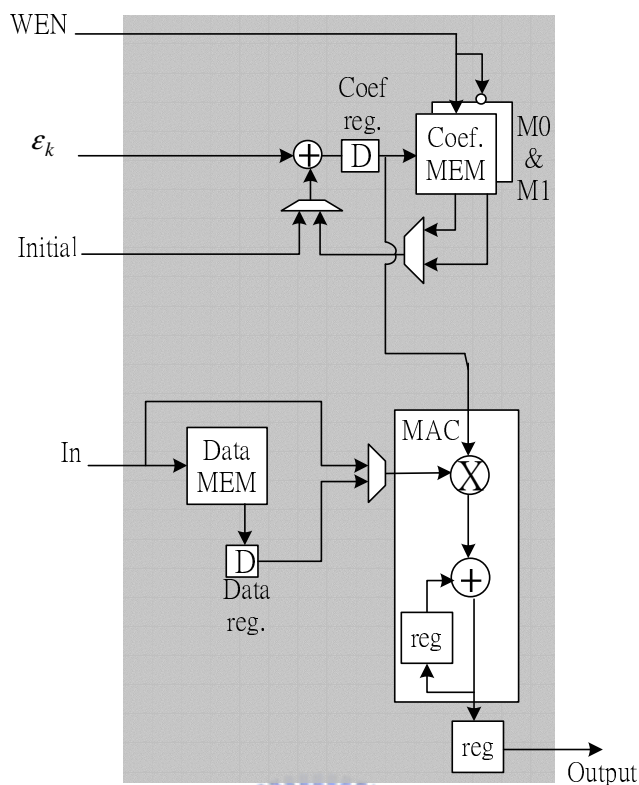


圖 5-5：前饋濾波器硬體架構

與  $a^1(z) = \sum_{j=0}^{31} a_j^1 z^{-j}$ ，前饋濾波器之係數為  $a_j^0$  與  $a_j^1$ 。如表 5-2 所分配，每個子濾

波器的 32 係數被分成兩部分分別存在記憶體 M0、M1 中。記憶體 M0 中存放  $j$  為偶數的係數，M1 則存放  $j$  為奇數的係數。

M0	M1
$a_j^0, j = 0, 2, \dots, 30$	$a_j^0, j = 1, 3, \dots, 31$
$a_j^1, j = 0, 2, \dots, 30$	$a_j^1, j = 1, 3, \dots, 31$
$a_j^0$ 儲存在 M0、M1 之雙數記憶體位址， $a_j^1$ 儲存在 M0、M1 單數記憶體位址	

表 5-2：前饋濾波器之係數記憶體分配

為什麼係數記憶體會不同於資料記憶體要分成兩組不同塊的記憶體。因為受限於記憶體的存取機制，必須是使用單埠的記憶體，這種記憶體在每個時脈週期只能做讀取或者寫入其中一種動作，如果我們除了每次讀出係數做乘法累加後還要做更新係數的動作的話就沒有多餘的時間寫入更新後的係數了。將記憶體分成兩組

後會有兩倍的存取頻寬，當一組記憶體正在讀出係數時，另一組記憶體則正在做寫入新係數的動作。當係數  $a_0^0$  在第 1 個時脈從被記憶體 M0 被讀出並做了係數更新與第一個 MAC 的運算，在第 2 個時脈時係數  $a_0^1$  從 M1 被讀出而係數  $a_0^0$  被寫回 M0，同樣的係數  $a_0^1$  被更新與用來做第二個 MAC 的運算。

可變係數長度做法只要將輸出暫存器做些限制即可，如果我們今天需要使用  $L$  個係數長度(其中  $0 < 2L \leq 64$  代表每個子濾波器係數長度)，只要每個子濾波器的乘法累加器的累加次數只要做  $L$  次後就停止累加，該累加暫存器就不會被累加。當訓練模式被啟動時，係數記憶體必須被初始化，initial 就會開始有初始值，

使的  $\varepsilon_k(n) = 0$ 、 $\begin{cases} a^0(j=l)=1 \\ a^0(j \neq l)=0 \end{cases}$  與  $\begin{cases} a^1(j=l)=1 \\ a^1(j \neq l)=0 \end{cases}$ ，代表在第  $l$  個係數被初使化為 1。

訓練開始時，如式  $\varepsilon_k(n) = \text{shift}[x_k(n), N]$ ， $n=0,1,\dots,31$ ，等於式 5-1 等號右邊的第二項，用來做係數微調運算。而新係數成為  $w_{k+1}(n) = w_k(n) + \varepsilon_k(n)$ ，只需要圖 5-5 係數記憶體旁的加法器就可以完成係數更新運算。

### 5.1.1.3 回授濾波器

圖 5-6 中為回授濾波器的硬體架構圖，此硬體包括了兩個資料暫存器(暫存器總容量為 128bits)、四個係數記憶體(記憶體大小為 2048bits，其記憶體的容量配置如表)、兩個乘法累加器和兩個加法器。前面說過了乘法累加器的個數決定有三個要素，既然我們在前饋濾波器只使用了兩個乘法累加器，既要符合最大傳



輸速率 2.3Mbps 與最大係數長度 128，所以系統頻率必須至少為 49.07MHz。

回授濾波器使用了兩個 MAC 才能完成 128 係數長度的運算，每個 MAC 個別需要兩組係數記憶體和一個 64 位元的資料暫存器，所以每次取樣每個 MAC 能夠處理 64 次的乘法累加運算。

$$\text{令回授濾波器 } w^b(z) = \sum_{j=0}^{127} p_j z^{-j} \text{，其係數 } p_j \text{，} j=0 \text{ 至 } 63 \text{ 儲存在 MAC0 的兩}$$

組係數記憶體中，其餘係數  $p_j$ ， $j=64$  至 127 儲存在 MAC1 的兩組係數記憶體中。

因為每個 MAC 的係數記憶體被分成了兩組各深度為 32 的係數記憶體，要把係數  $p_j$ ， $j=0$  至 63 與係數  $p_j$ ， $j=64$  至 127 再做分配。如表 5-3 所示：

M2	M3	M4	M5
$p_j$ ， $j = 0, 2, \dots, 62$	$p_j$ ， $j = 1, 3, \dots, 63$	$p_j$ ， $j = 64, 66, \dots, 126$	$p_j$ ， $j = 65, 67, \dots, 127$
32 taps	32 taps	32 taps	32 taps

表 5-3：回授濾波器之係數記憶體分配

每組係數記憶體均可存放 32 個係數，有了如此分配之後便可很容易的解決單埠記憶體存取頻寬不足的問題，係數更新之後可以在另一個時脈週期儲存回記憶體。

圖 5-6 硬體架構的最後輸出如式：

$$eq_{fb}(n) = \sum_{j=0}^{127} \text{sign}(\hat{I}_j) \cdot p_{n-j} \quad (5-2)$$

如式 5-2，乘上  $\text{sign}(\hat{I}_k)$  等於是做改變正負號的動作，而  $\text{sign}(\hat{I}_k)$  只需要一位元就可表示，1 代表負號，0 代表正號，所有的輸入資料  $\text{sign}(\hat{I}_k)$  都會被儲存在圖 5-6 之資料暫存器中。MAC 中的乘法器也只需使用多工器選擇是否改變正負

號就可以實現。

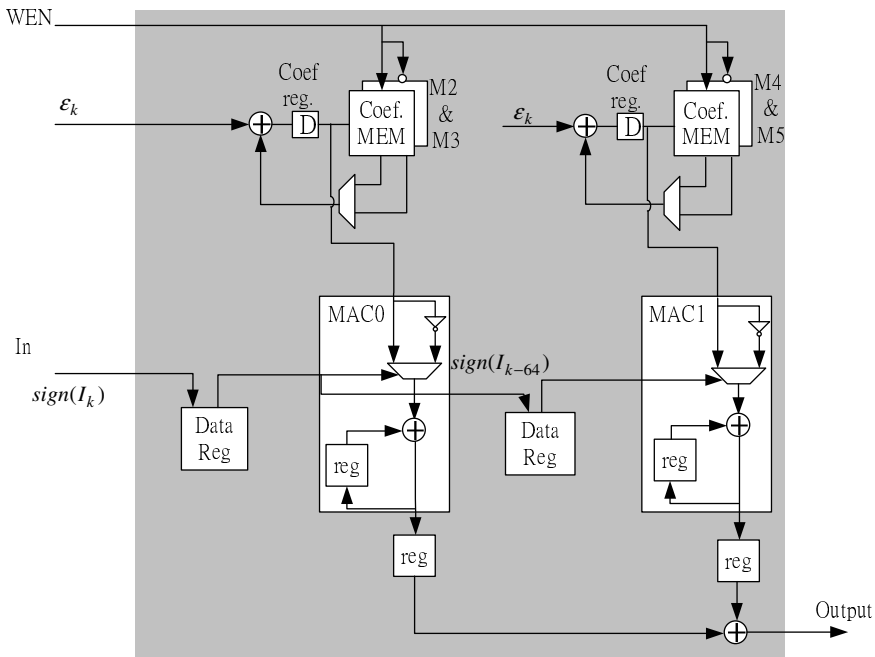


圖 5-6：回授濾波器硬體架構圖

#### 5.1.1.4 決策裝置(slicer)

由於在訓練模式時，訊號皆以 2-PAM 調變方式傳送，所以圖 5-1 中的決策裝置只需要判別兩種準位，+A 或者 -A，A 表示大小值。因為只需要判斷正負號，圖 5-1 的決策裝置輸出只有一位元，輸出 0 代表決策為 +A，1 則代表決策為 -A。

#### 5.1.2 資料模式硬體架構

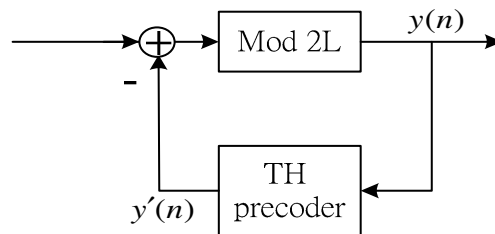


圖 5-7：TH Precoder 方塊

當訓練模式結束後，接收機端會把回授濾波器的係數轉移至發射機端的 THP 濾波器，此時發射機端會利用記憶體把係數儲存下來（此係數記憶體與圖 5-6 的係數記憶體相同）。係數準備好了之後，這個 THP 濾波器就可以順利的在資料模式下運作。THP 濾波器的方塊如圖 5-7。

THP 硬體架構如圖 5-8 所示。其濾波器硬體包括兩個係數記憶體、兩個資料記憶體和兩個乘法累加器構成。記憶體容量的配置表 5-4 如下：

	THP Filter
Coefficient & Data Memory	16x128 bits & 14x128 bits
Total	3840 bits

表 5-4：THP 濾波器記憶體配置

圖 5-8 為 THP 濾波器的硬體架構圖，因為這個濾波器係數為訓練模式後接收機端轉移給發射器端的，在硬體架構上也相當類似於圖 5-6 的硬體架構。只有資料記憶體容量與 MAC 有些微的差別。

由圖 5-8 的濾波器可以得到輸出

$$y'(n) = \sum_{j=0}^{63} y(j)p(n-j) + \sum_{j=64}^{127} y(j)p(n-j) \quad (5-3)$$

上式中可以看到兩個做累加的運算，由兩個 MAC 可以負責完成。而輸入資料  $y(n)$  會存在資料記憶體中，在第 65 的取樣點時第一個資料記憶體會被存滿，此時最舊的輸入資料  $y(n-64)$  將會被移出存到另一個資料記憶體中。而讀出係數時只要依序從 M6、M7 兩個係數記憶體中讀出即可，M6、M7 儲存的係數分配如下表：

M6	M7
$p(j)$ , $j=0,1,\dots,63$	$p(j)$ , $j=64,65,\dots,127$
64 taps	64 taps

表 5-5：THP 係數記憶體分配。

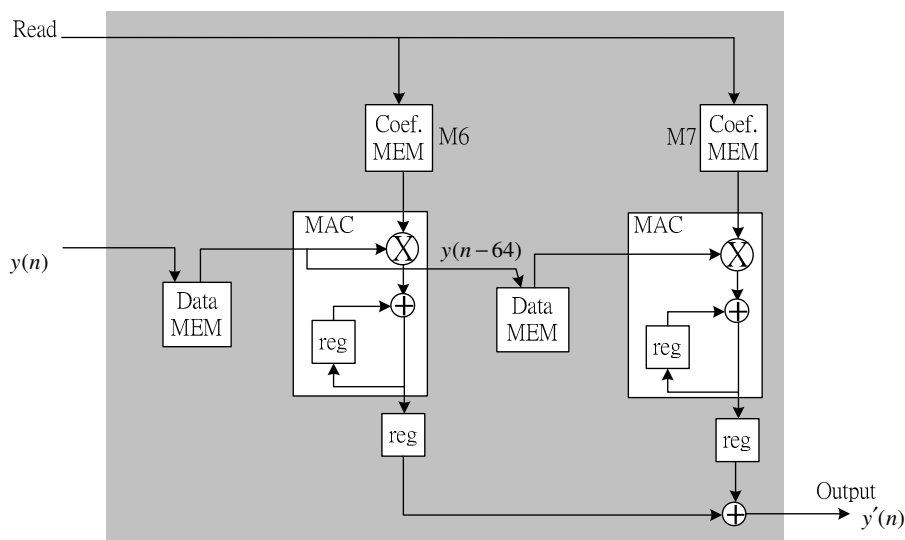


圖 5-8：THP 濾波器硬體架構圖

## 5.2 TCM 解碼器硬體架構

TCM 解碼器的解碼方式為大家熟知的 Viterbi 演算法，硬體的主要架構共分為三個運算單元，分別是(1)Branch Metric Unit(BMU)、(2)Add、Compare and Select Unit(ACSU)、(3)Survivor Memory Unit(SMU)。其中 BMU 硬體之設計較為簡單，其餘的 ACSU 與 SMU 之硬體都是 Memory-based 的設計，且以節省記憶體與符合 G.SHDSL 規格為訴求。TCM 解碼器之硬體設計考量：

- 必須達到 G.SHDSL 最大的傳輸速率 2.3Mbps。
- BMU 須針對 THP 系統資料跳動現象做修正。
- ACSU 及 SMU 採用記憶體作儲存裝置。
- 迴旋碼編碼器限制長度  $K=10$ ，意即每一個取樣訊號(a symbol)必須處理多達  $512(2^{K-1})$ 個狀態。

這四點考量不外乎都是以速度為主的訴求，針對這四點訴求我們必須制定出硬體架構以符合要求。先考慮記憶體的因素，(1)必須採用最少的記憶體 (in-place)、(2)單埠的記憶體，表 5-7 分配了 TCM 解碼器的容量配置。我們先把 512 個狀態分成 8 組(8 banks)，這樣可以避免記憶體的衝突，後面將會說明如何

分配與如何產生記憶體位址。ACS 要一次從記憶體抓取八個狀態並做運算和存回，所以 ACSU 必須是可以每次做八個狀態 ACS 運算 butterfly 的基本架構。512 個狀態將需分 64 次 butterfly 運算始可處理完畢，因為記憶體為單埠的，每次還要乘上 2 個時脈週期為 128 個時脈週期，所以我們每筆取樣訊號需要 128 個時脈週期處理。

Radix	k	Ideal Speedup	Complexity Increase	Area Efficiency	System Speed
2	1	1	1	1	98.133MHz
4	2	2	2	1	49.066MHz

表 5-6：Radix- $2^k$  硬體架構比較

	ACSU	SMU
Total memory	7168 bits	25600 bits
Banks	8	8

表 5-7：TCM 解碼器之記憶體需求

如表 5-6[18]，有兩種不同的實現架構，如果以 radix-2 實現的架構需要 98.133MHz ASIC 的速度，若 radix-4 的架構需要 49.066MHz 的速度。因為採用 radix-4 架構 ASIC 不需較高的電路操作頻率，可比 radix-2 架構快一倍速度，radix-4 只需較高複雜度的代價，所以我們採用 radix-4 的架構實現 TCM 解碼器。

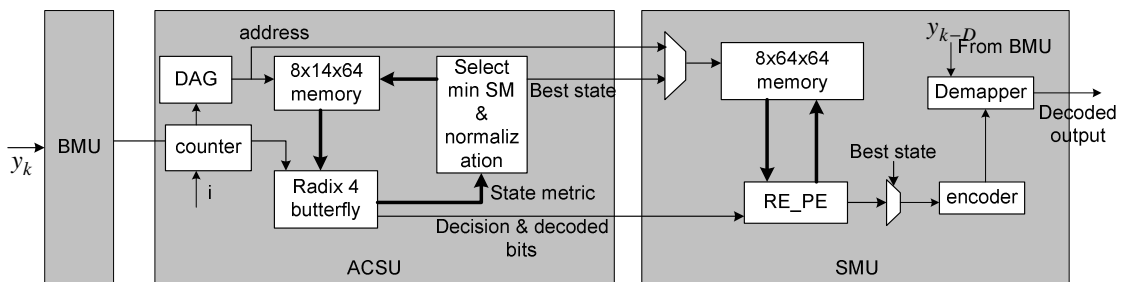


圖 5-9：TCM 解碼器方塊圖

如圖 5-9 為 TCM 解碼器硬體方塊圖，主要區分為 BMU、ACSU 和 SMU 三個部分實現，將在 5.2.1、5.2.2 和 5.2.3 三節中說明硬體架構。

### 5.2.1 Branch Metric Unit (BMU)

BMU 主要功能為計算 branch metric。BMU 是 TCM 解碼器的第一級，輸出的 branch metric 送至 ACSU 計算 path metric 之用。由於本論文中採用 radix-4 的架構，所以一次需要處理兩筆取樣訊號，而不同於 radix-2 的架構一次只需處理一筆取樣訊號。

參考圖 5-10，上為 radix-4 butterfly，下為 radix-2 butterfly。這是四個狀態的迴旋編碼器的格子圖，radix-4 butterfly 為一個 4 輸入與 4 輸出的運算單元，根據格子圖，radix-4 butterfly 每取樣兩次輸入訊號才能得到其分支計量。另一個 radix-2 butterfly 為一個 2 輸入 2 輸出的運算，而每次取樣的訊號都要算出分支計量。

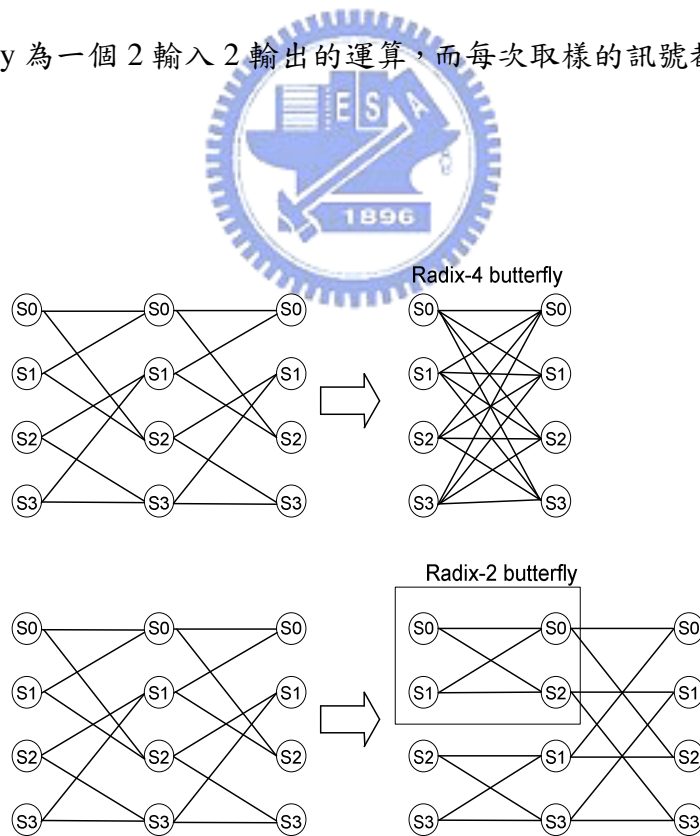


圖 5-10：Radix-2 與 Radix-4 格子圖

圖 5-11 之左圖為一狀態  $N=4$  的格子圖，其中每個狀態的分支上標示的符號代

表其 radix-2 架構之分支計量。由左圖可以等效出右圖 radix-4 的格子圖，同樣標上了其分支計量。BMU 就是要求出 radix-4 架構的分支計量。

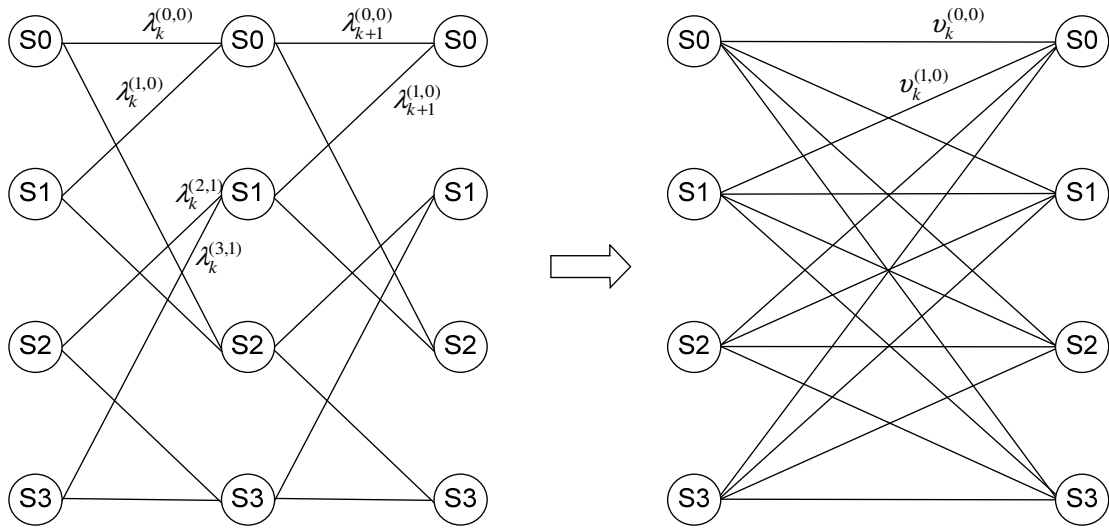


圖 5-11：Radix-4 格子圖

以  $N=4$  為例，我們首先計算出  $N=4$  狀態格子圖之 radix-2 架構的分支計量如式：

$$\lambda_k^{(i,j)} = \text{dis}(r_k, c_k) \quad (5-4)$$

其中  $i, j = 0, 1, 2, 3$  代表四個狀態。 $r_k$ ， $c_k$  分別為接收訊號與狀態轉移可能傳送的訊號。 $\text{dis}(x, y)$  表示為求  $x$  與  $y$  的距離。 $\lambda_k^{(i,j)}$  則表示當時間  $k$  時，狀態  $i$  轉移至狀態  $j$  的分支計量。

BMU 必須能夠把 radix-4 butterfly 架構中每一條分支的分支計量(branch metric)算出來。參考圖 5-11。由 radix-2 得到的分支計量可以再求出 radix-4 的分支計量：

$$v_{k+1}^{(i,j)} = \lambda_k^{(i,m)} + \lambda_{k+1}^{(m,j)} \quad (5-5)$$

為時間  $k$  與  $k+1$  兩兩 radix-2 的分支計量相加之和。

計算分支計量時必須針對擴展之星雲圖作適當的修改，如圖 5-12 為擴展後的星雲圖，圖中虛線所指為擴展的信號，當分支的訊號子集合為  $C_0$  與  $C_3$  時，

必須考慮與訊號-17/16、17/16 的歐氏距離，這樣才不會受到資料跳動現象的影響。

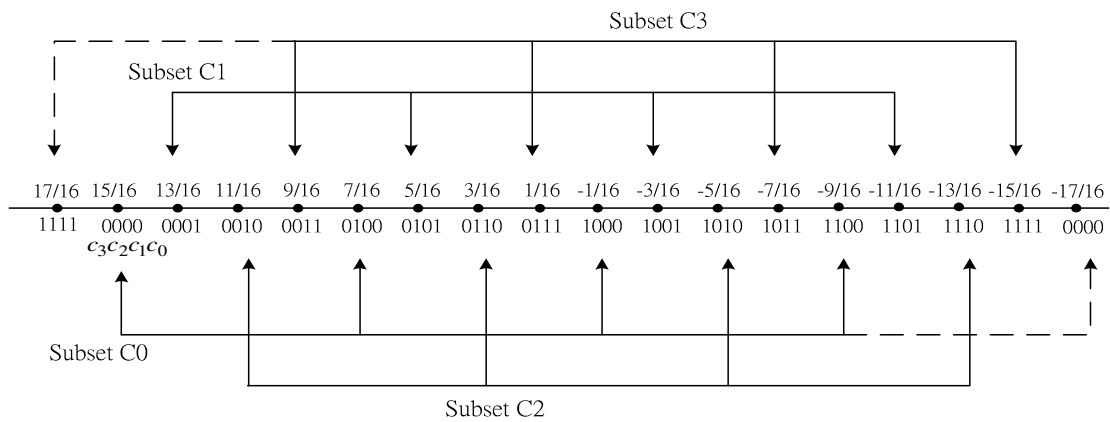


圖 5-12：擴展後之星雲圖

## 5.2.2 Add、Compare and Select Unit (ACSU)

BMU 把分支計量送至 ACSU 後，ACSU 會遞迴的作 ACS 運算。以狀態 N=4 的格子圖為例其運算如下：

$$[\gamma_{j,k}, sel_{j,k}] = \text{Min}(\gamma_{0,k-2} + v_{k-2}^{(0,j)}, \gamma_{1,k-2} + v_{k-2}^{(1,j)}, \gamma_{2,k-2} + v_{k-2}^{(2,j)}, \gamma_{3,k-2} + v_{k-2}^{(3,j)}) \quad (5-6)$$

其中  $j=0,1,2,3$  代表狀態值， $k$  代表時間，radix-4 的架構是每兩次取樣才做一次 ACS 運算。 $\gamma_{j,k}$  是狀態計量， $sel_{j,k} = 0,1,2,3$  表示選擇的存活路徑。 $v_k^{(0,0)}$  表示當時間  $k$  時，狀態 0 至狀態 0 這條分支的分支計量。 $\text{Min}(x, y, w, z)$  為找出  $x$ 、 $y$ 、 $w$  或  $z$  中最小的四個分支計量值。

如式 5-6 所示，ACSU 運算核心為一個 radix-4 butterfly。前面敘述過解碼器每次需解出兩個取樣訊號，每次必須在 128 個時脈週期解完，所以每個時脈週期需處理四個狀態的 ACS 運算。由於每次(兩個時脈週期)要做八個狀態的運算，所以會有八個來自不同記憶體位置 state metric，如果要做到毫無記憶體衝突且能節省記憶體的狀況。在使用單埠記憶體時，必須把記憶體分為八組(banks)，就有辦法做到良好的分配及管理。

在計算 state metric 時，由於是有限位元數表示法，必須注意到累加時會有溢位



(overflow)的可能性，所以每次在運算時，都要做正規化的處理程序。

$$\gamma_{j,k} = \gamma_{j,k} - \min(\gamma_{\min,k-2}) \quad (5-7)$$

如式 5-7，每次算出的狀態計量都要減去前一次找出的最小狀態計量。

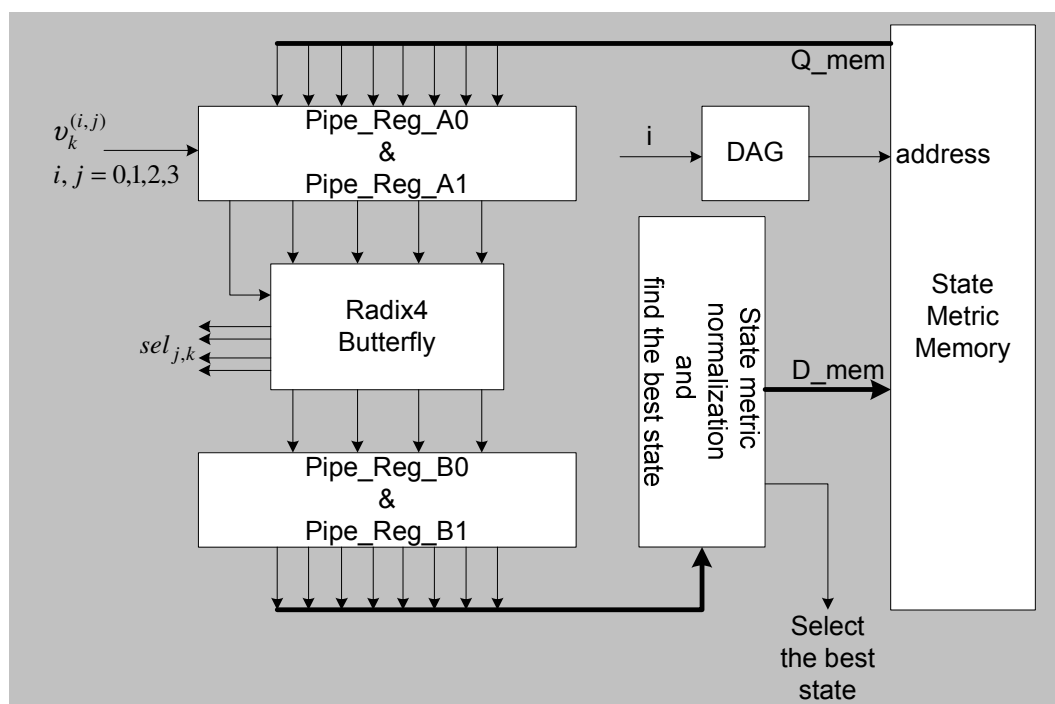


圖 5-13：ACSU 方塊圖

我們將 ACSU 的硬體架構分為記憶體位址產生器、radux4 butterfly 與找尋最佳狀態與狀態計量正規化三部分說明。

### 5.2.2.1 記憶體位址產生器

狀態計量的儲存方式為 in-place，只能使用 14x512 bits 容量的記憶體，每個狀態計量用 14 位元表示，為無符號(unsigned)表示法，總共需存 512 個狀態計量。如果要充分使用這些記憶體空間，需要做到能夠從同一個記憶體抓出資料算完之後再存回同一個位址，這樣的機制需要在記憶體的位址產生方法上下功夫。

如果  $N=2^4$  的格子圖依 Radix-2 butterfly 可以排成 4 個 stages，如圖 5-14 所示，

很容易的可以排出其記憶體位址的產生方式。因為將  $N=2^9$  的格子圖無法依 radix-4 butterfly 排成如上那樣會重複循環的 stages，因為  $N$  不是 4 的幕次方。

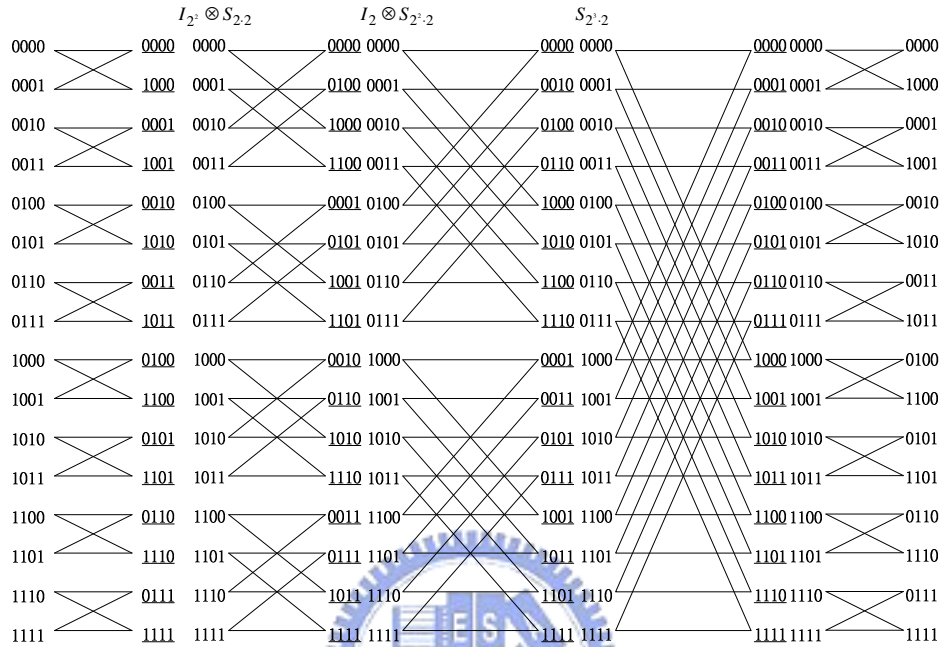


圖 5-14：狀態  $N=16$  之 Radix-2 格子圖

所以必須修改為把兩個 radix-4 butterfly 綁在一起，成了一個 8 輸入 8 輸出的 butterfly。再把這個架構的 512 個狀態的格子圖可以規律地排成 9 個 stages 的格子圖。現在只要找出每個 stage 記憶體位址的重排矩陣即可得到位址產生方式，藉由重排矩陣使一個有順序的序列重排出所要的記憶體位址。要了解重排矩陣之前，先簡單介紹 Kronecker Product  $\otimes$  的運算方式[19]。

令矩陣  $M$  與  $N$ ：
$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \text{ 和 } N = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

其矩陣  $M$  與  $N$  之 Kronecker Product 運算為：
$$M \otimes N = \begin{bmatrix} aA & aB & bA & bB \\ aC & aD & bC & bD \\ cA & cB & dA & dB \\ cC & cD & dC & dD \end{bmatrix} \quad (5-8)$$

當矩陣  $M$  為單位矩陣  $I_B$  時，我們可以得到一個稱為 normal factor 形式的特殊 Kronecker Product，式(5-9)為一個 normal factor 形式的例子。

$$I_3 \otimes N = \begin{bmatrix} A & B & & & & \\ C & D & & & & \\ & & A & B & & \\ & & C & D & & \\ & & & & A & B \\ & & & & C & D \end{bmatrix} = \begin{bmatrix} N & 0 & 0 \\ 0 & N & 0 \\ 0 & 0 & N \end{bmatrix} \quad (5-9)$$

此形式的好處在於，可將複雜地大矩陣運算分解成數個獨立且相同的小矩陣來運算，如式子(5-10)所示，若有一向量  $\bar{w}$  為向量  $\bar{v}$  經過矩陣  $K$  運算後所得到之向量，且矩陣  $K$  為 normal factor 形式時，則圖 5-15(a)將化簡成圖 5-15(b)。

$$\bar{w}^t = K\bar{v}^t = (I_3 \otimes N)\bar{v}^t \quad (5-10)$$

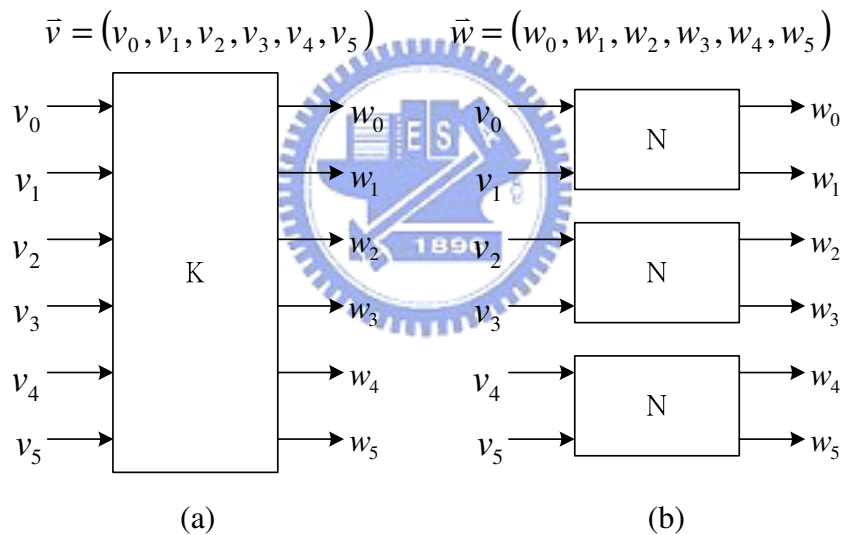


圖 5-15：(a)大矩陣運算(b)分解成三個小矩陣運算

而  $S_{B_0, B_1}$  為重排矩陣，也就是說一個含有  $B_0 \times B_1$  個元素之向量  $\bar{v}$ ，經過此矩陣運算後，其內部元素將會互相調動重新排列成為一個相樣含有  $B_0 \times B_1$  個元素的新向量  $\bar{v}'$ 。此重排矩陣  $S_{B_0, B_1}$  可用另一種較簡單的方式來描述其重排方式，假設一個具有 15 個元素的向量  $\bar{v}$ ，其向量元素所在位置分別為  $i = 0$  至  $i = 14$ ，經重排矩陣  $S_{3,5}$  運算後向量  $\bar{v}$  內的元素將重新排列成為向量  $\bar{v}'$ 。如圖 5-16 所示，原本在向量

$\bar{v}$  中  $i=2$  位置的元素經由  $S_{3,5}$  運算後，將移至新向量  $\bar{v}'$  中的  $j=6$  位置。我們可將元素所在位置  $i$ ，根據重排矩陣  $S_{3,5}$  的下標  $[3,5]$  作為其  $i$  的基底來表示，經由  $S_{3,5}$  轉換可得重排後由基底  $[5,3]$  所表示的位置  $j$ ，如下式所示：

$$i:[a,b] \xrightarrow{S_{3,5}} j:[b,a] \quad (5-11)$$

藉由此式子，我們可以知道經過重排後位置之間的相對關係。

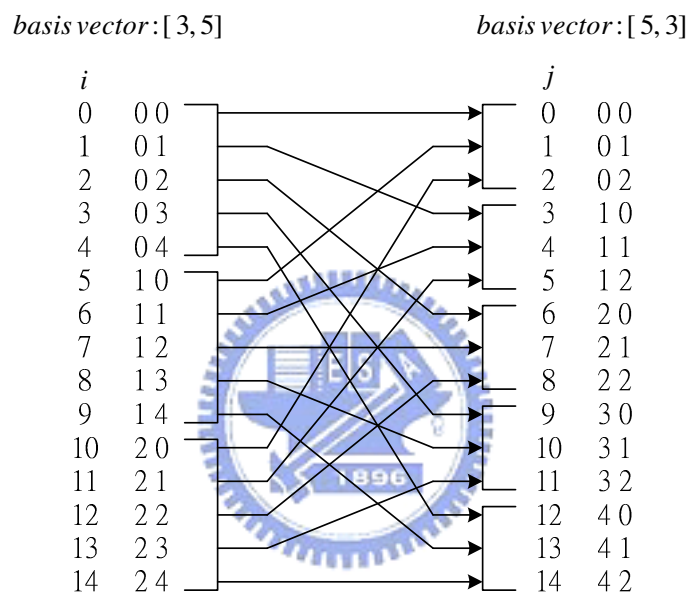


圖 5-16 :  $i:[a,b] \xrightarrow{S_{3,5}} j:[b,a]$  圖示

最後，我們可將之前所提到的 Kronecker Product 與 shuffle permutation 結合在一起運用。若有一重排矩陣為  $I_{B_0} \otimes S_{B_1, B_2}$  形式，則重排後位置的對應關係如式子(5-12)所示，其中  $i$  為轉換前之位置且基底為  $[B_0, B_1, B_2]$ ， $j$  為轉換之後的位置，其基底為  $[B_0, B_2, B_1]$ 。

$$i:[a,b,c] \xrightarrow{I_{B_0} \otimes S_{B_1, B_2}} j:[a,c,b] \quad (5-12)$$

由之前的流程圖可知，一個  $N = 2^M$  點的 radix-2 分時快速傅立葉轉換共有  $M$  個 stages，且每個 stage 對應到不同形式的 shuffle permutation，而第  $i$  個 stage 的 shuffle

permutation 為：

$$I_{2^{M-i}} \otimes S_{2^i,2} \quad i = 0,1,\dots,M-1 \quad (5-13)$$

故我們可由上式，來得知 butterfly 運算器所欲擷取資料的記憶體對應位址。  
藉由將計數器所計數出來的值  $k$  ( $k = 0,1,\dots,N-1$ )，經過第  $i$  個 stage 的 shuffle permutation ( $I_{2^{M-i}} \otimes S_{2^i,2}$ ) 運算後，如下式(5-14)所示，即可產生 butterfly 運算器的輸入資料擷取位址。圖 5-17 為 DAG 之方塊圖。

$$k : [a,b,c] \xrightarrow{\text{shuffle}} \text{address}[a,c,b] \quad (5-14)$$

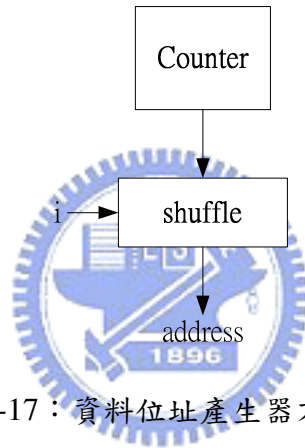


圖 5-17：資料位址產生器方塊圖

針對本論文  $N=512$  狀態的特殊例子，我們找到其重排矩陣如式 5-15：

$$\begin{cases} I_{2^M} & i = 0 \\ S_{2^{2i} \cdot 2^{M-2i}} & , i = 1,2,3,4, M = 9 \\ S_{2^{2i-M} \cdot 2^{2(M-i)}} & i = 5,6,7,8 \end{cases} \quad (5-15)$$

根據重排矩陣其位址的產生方式，stage 從  $i=0,1,2,\dots,8$ ，共有 9 級。如圖 5-17，有一個 9 位元的 counter，每個 stage 都會從 0 數到 511。輸入 shuffle 後會把 9 位元的順序序列重新排列成一個新序列，這新序列就是我們要的記憶體位址。

### 5.2.2.2 Radix-4 Butterfly

如圖 5-13 所示，ACSU 只使用單一 radix-4 butterfly 做 ACS 運算，每個時脈

週期處理四個狀態。這個 butterfly 有 16 個分支計量輸入與 4 個狀態計量輸入，根據式 5-6 而每個狀態需比較 4 個路徑找出最小的狀態計量即為存活的路徑，存活路徑被選擇後會有一個指示訊號  $sel_{j,k}$  將會送至 SMU 供解碼之用。如圖 5-18 為單一個狀態 ACS 的運算硬體，一個 radix-4 butterfly 需要圖 5-18 四個相同的硬體組成。

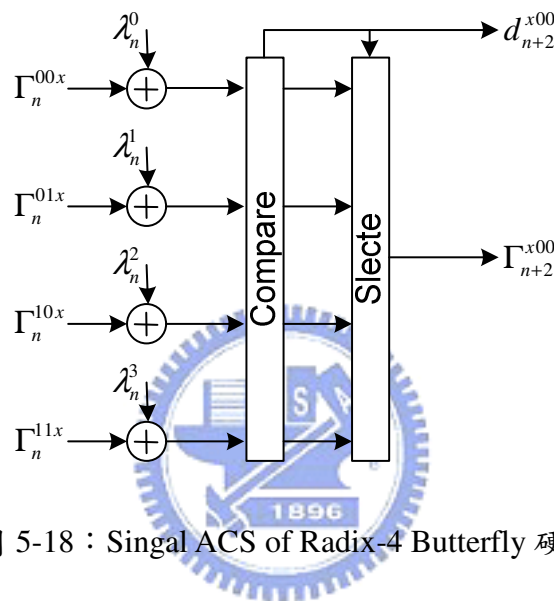


圖 5-18：Singal ACS of Radix-4 Butterfly 硬體

我們在圖 5-19 把 radix-4 butterfly 的運算核心之時序圖畫出，前面也強調過本論文之 TCM 解碼器之 ACSU 只使用了一個 radix-4 butterfly 的運算核心。此運算核心在每個時脈必須處理四個狀態 ACS 運算，也就是我們的 radix-4 butterfly 運算核心並不會閒置(idle)，在時序的安排上沒有浪費一點時間。

因為記憶體需分成八組，承圖 5-19，每兩個時脈週期會被讀出一次，每次都有 8 個狀態計量被讀出存在管線暫存器 A 中(Pipeline Register)，經 ACS 運算後之狀態計量也會被存在管線暫存器 B 中，將管線暫存器 B 中的狀態計量經正規化後以兩個時脈週期為一個週期寫入記憶體。

圖 5-19 中 Pipe\_RegA0 與 Pipe\_RegA1 代表管線暫存器 A，為 radix-4 butterfly 之輸入，Pipe\_RegA0 與 Pipe\_RegA1 分別都存了 4 個狀態計量，共存了 8 個狀態

計量，它們必須是同時從記憶體被讀出與寫入的。Pipe\_RegB0 與 Pipe\_RegB1 代表管線暫存器 B，為 radix-4 butterfly 之輸出，Pipe\_RegB0 與 Pipe\_RegB1 分別都存了 4 個狀態計量。在第一個時脈時 radix-4 butterfly 會將暫存器 Pipe\_RegA0 內之狀態計量讀出做 ACS 運算，完成 ACS 後把結果存在暫存器 Pipe\_RegB0。第二個時脈時 radix-4 butterfly 會將暫存器 Pipe\_RegA1 內之狀態計量讀出做 ACS 運算，完成 ACS 後把結果存在暫存器 Pipe\_RegB1。如此的遞迴下去，直到 512 個狀態都做完 ACS 運算，共需分 64 次的記憶體存取。

Pipe\_Reg\_dec0 與 Pipe\_Reg\_dec1 為輸出暫存器，儲存 radix-4 butterfly 每次 ACS 運算後產生存活路徑之決策(decision)。其時序與 Pipe\_RegB0 與 Pipe\_RegB1 兩暫存器相同。

記憶體都是正緣觸發，也就是讀出和寫入的動作都發生在正緣觸發後。承圖 5-20 為記憶體讀寫之時序圖，清楚的說明了記憶體在讀出與寫入時序之分配，這樣良好的時序分配使的 radix-4 butterfly 可以容易在時間上做資源共享(resource shared)，不用額外使用第二個 radix-4 butterfly 處理 ACS 運算。

圖 5-20(a)、(b)清楚的標示了從記憶體依序讀出 512 個狀態計量，完成了 ACS 運算與正規化後再被寫回記憶體。第一個時脈要先從記憶體中讀出狀態計量，而 Write\_en 設為 1 表示讀出動作，Q\_mem 是記憶體資料輸出，在第一個時脈的正緣後 8 個狀態計量被穩定的讀出，符號 sm0-7 表示 512 個狀態計量中前 8 個狀態計量。在第二個時脈之負緣時，Pipe\_Reg\_A 暫存器讀取 Q\_mem，此 8 個狀態計量將被存在暫存器中。ACS 運算在前面已經敘述過其時序圖，如圖 5-19。

做 ACS 運算時為了確保功能之正確性，記憶體讀寫動作必須務必注意到不可抹寫到尚未做 ACS 之狀態計量。圖 5-19 中 Chip\_en 為一位元之記憶體開關控制訊號，0 表示關掉記憶體不做任何讀寫動作，1 則可以執行讀寫。注意圖 5-20(a) 中 D\_mem 訊號為記憶體之輸入資料，在第二個時脈週期時 Chip\_en 被設置為 0，因為此時並未有任何 ACS 運算完成，D\_mem 之資料不會被寫進記憶體。這動作確保了記憶體內之狀態計量未做 ACS 前不會被抹寫。

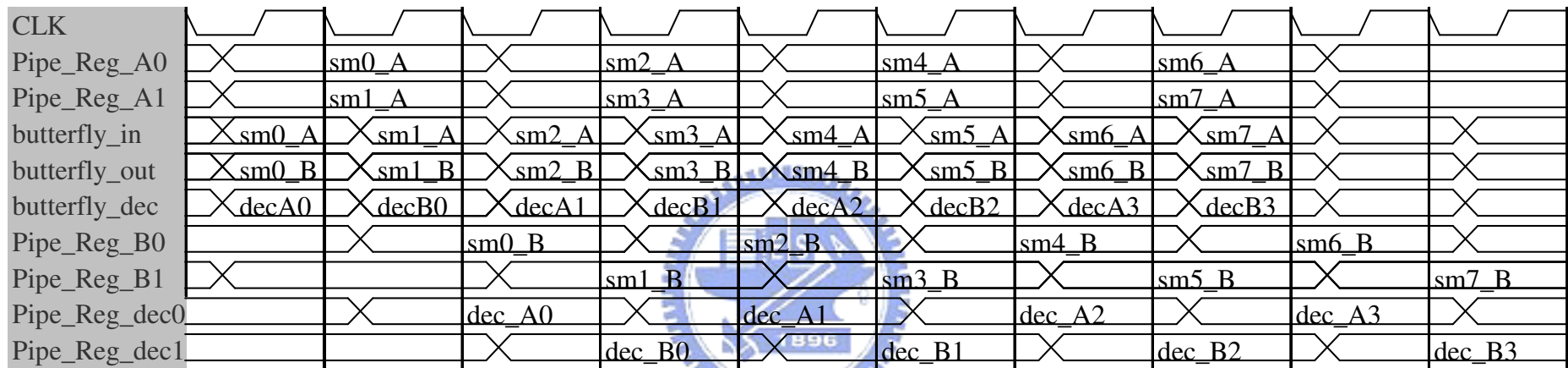


圖 5-19：Radix-4 Butterfly 之時序圖



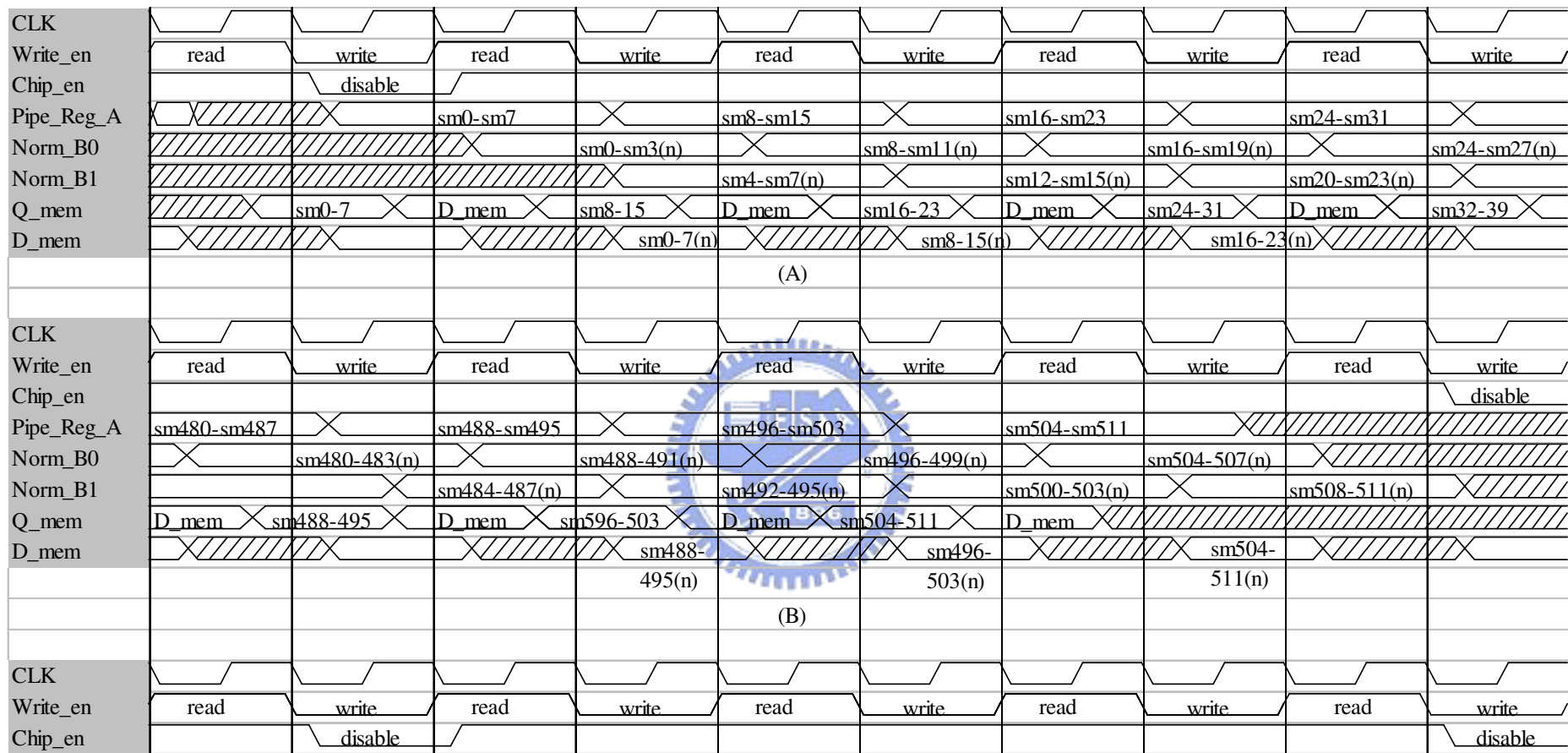


圖 5-20：(a)(b)狀態計量記憶體之讀寫時序圖，(c)(d)讀寫之分配時序圖

最後在圖 5-20(c)、(d)要說明狀態計量從記憶體被讀出做 ACS 運算後寫回記憶體，這個動作需要兩個時脈週期完成。Address 為記憶體之位址，標示的號碼 0、1、2 ....、63 共需 64 次的讀出與寫入，號碼表示第 n 組記憶體位址，由記憶體位址產生器產生。

### 5.2.2.3 找尋最佳狀態計量與狀態計量正規化

512 個狀態計量以每次 8 個狀態計量的數量送到此方塊，圖 5-13，在此方塊要完成找尋最佳狀態與狀態計量正規化兩個任務。

- 找尋最佳狀態：只要找出 512 狀態中最佳狀態的記憶體位址，即最小狀態計量，這個記憶體位址會送到 SMU 供解碼之用。找尋最小狀態計量只需要減法器就可以比較出大小，每次比較八筆狀態計量使用暫存器將最小的狀態計量存下來，每 512 個狀態分成 64 次比較，最後可以找出 512 個狀態中最小之狀態計量。每個最小狀態計量都有個記憶體位址，因為 SMU 的記憶體位址與 ACSU 的記憶體位址相同，所以只要把最小狀態計量的記憶體位址送至 SMU 即可提供解碼，參考圖 5-21 中 Best state 就是 ACSU 找出最佳狀態後，在 SMU 記憶體中擷取解碼位元(decoded bit)。
- 狀態計量正規化：承式 5-5，每做完 512 狀態的 ACS 運算後，可以先找出這 512 狀態中最小的狀態計量。下一次要做另一個 512 狀態之 ACS 運算時，會把運算後的狀態計量減去前一次找到的最小狀態計量。

### 5.2.3 Survivor Memory Unit (SMU)

SMU 根據 ACSU 提供的決策訊號找出存活路徑，利用交換器(Exchange Element)電路實現，以存活深度(survivor depth) $D=5K=50$  為長度找出其存活路徑。如圖 5-21 為 Memory-based SMU，我們在接下來 5.2.3.1 與 5.2.3.2 兩小節說明。

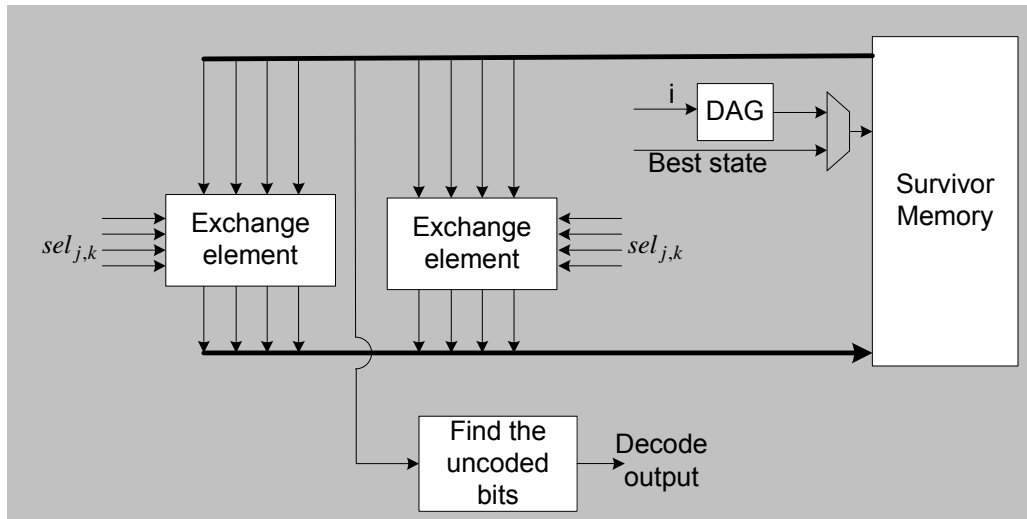


圖 5-21：SMU 方塊圖

### 5.2.3.1 位址產生器

在 SMU 會用到記憶體儲存每條存活路徑資料。如要使用最少的記憶體空間，採用 in-place 將是最佳方式，而其記憶體位址產生方式會與 ACSU 的記憶體相同，所以使用同一個位址產生器。其位址產生方式可參考 5.2.2.1 節。

### 5.2.3.2 交換器架構與最佳狀態解碼

圖 5-21 中之交換元件(exchange element)以式 5-16 敘述交換元件之行為，將存活路徑  $p_{j,k}$  做交換， $i=0,1,2,3$ ，此交換元件的功能可找出 512 個狀態的每一條存活路徑，每次可以找出四條存活路徑，只需要使用多工器就可實現。根據 ACSU

$$N=512, \begin{cases} 0 \leq j < N/4, m = j \\ N/4 \leq j < N/2, m = j - N/4 \\ N/2 \leq j < 3N/4, m = j - N/2 \\ 3N/4 \leq j < N, m = j - 3N/4 \end{cases}$$

Case( $sel_{j,k}$ )

$$00 : p_{j,k} = \{ p_{m,k} \gg 2, b_j \}$$

$$01 : p_{j,k} = \{ p_{m+1,k} \gg 2, b_j \}$$

$$10 : p_{j,k} = \{ p_{m+2,k} \gg 2, b_j \}$$

$$11 : p_{j,k} = \{ p_{m+3,k} \gg 2, b_j \} \quad (5-16)$$

給的決策訊號  $sel_{j,k}$ ，交換元件就可選出每個狀態的存活路徑。如果 ACSU 決定

了存活路徑，舊的存活路徑會被砍掉 2 位元 LSB 後，再補上兩位元的解碼位元  $b_j$  後全部複製至  $j$  路徑(狀態  $j$  的存活路徑)，就可以得到在時間  $k$  狀態  $j$  的存活路徑。

有了 512 條的存活路徑，ACSU 還必須提供了 best state 的訊息，將最佳存活路徑從 SMU 記憶體中擷取出。從 SMU 記憶體讀出之最佳路徑，其 MSB 的那兩位元即是最佳解，為時間  $k-D$  時的解。

### 5.2.3.3 未經編碼位元 $\{y_1, y_2\}$ 之解碼

在格子編碼調變(TCM)中，參考圖 4-6，有兩個位元的資料沒有經過編碼，而 Viterbi 演算法只解出已經迴旋編碼器所編碼的位元  $y_0$ ，因此當 SMU 解出經過迴旋編碼器所編碼的那一個位元後，還需把剩下之兩個未編碼位元位元資訊解出  $\{y_1, y_2\}$ 。我們只要根據 TCM 的 Mapper 製作 demapper 解就行了，根據以下步驟即可找出未編碼位元，如圖 5-21 中最底下方塊功能。

1. 將解出之已編碼位元  $y_0$  輸入原迴旋編碼器可以得到兩位元的編碼資訊  $\{c_0, c_1\}$ 。
2. 承 1，利用這兩位元資訊  $\{c_0, c_1\}$  選出訊號星雲圖中的訊號子集合。
3. 利用過去接收到的訊號  $r_{k-D}$  找出與訊號子集合中最為接近的訊號，對映此訊號所編的二進位值即可解出未編碼位元。

所以每個取樣訊號將會解出三位元的解碼資訊  $\{y_0, y_1, y_2\}$ 。

## 5.3 結論

前 5.1 與 5.2 兩節介紹過了 THP 系統與 TCM 解碼器的硬體架構。已達到以下的功能需求：

- 必須能夠達到 G.SHDSL 最大的傳輸速率 2.3Mbps。
- 前饋濾波器係數長度為可變。
- 前饋濾波器最大係數長度為 64，回授濾波器固定係數長度為 128。
- BMU 須針對 THP 資料跳動現象做修正。
- ACSU 及 SMU 採用記憶體作儲存裝置。
- 迴旋碼編碼器限制長度  $K=10$ ，意即每一個取樣訊號(a symbol)必須處理

$512(2^{K-1})$ 個狀態。

最後我們比較本論文設計之 memory-based SMU 硬體架構與常見的 TBM 架構之不同處。整理由表 5-8 詳述。

	功率消耗	潛伏期	記憶體用量
REA (using memory instead of register)	需求每個狀態之 survivor path，需較大的記憶體存取頻寬所以功率消耗較大	潛伏期較小，需與存活深度 D 相同的潛伏期	記憶體使用量較少，只需要儲存與存活深度 D 的記憶體，約需 DN 位元，N 為狀態數
TBM	只需從最佳 survivor path 追蹤出最佳解，記憶體存取頻寬小，則功率消耗小	較大，需 D+M 的潛伏期，其中 M 表示解碼深度	記憶體使用量較大，需要存 D+M 的深度，需 (D+M)N 位元，N 為狀態數

表 5-8：SMU 架構之比較

最後在第六章會敘述我們在 ASIC 與 FPGA 上實現硬體、效能分析與硬體驗證的結果。

## 第六章 硬體之模擬、設計與驗證

第五章介紹過了 THP 系統與 TCM 解碼器的硬體架構，本章將對第五章所提之硬體架構作實現，依據圖 6-1 之設計流程，利用軟體 Matlab 與 C 語言先做硬體架構模擬分析，在了解效能的可行性後才進一步的把硬體以 RTL 實現，最終實現的硬體會在 FPGA 模擬板上驗證其功能，並使用邏輯分析儀測量結果。

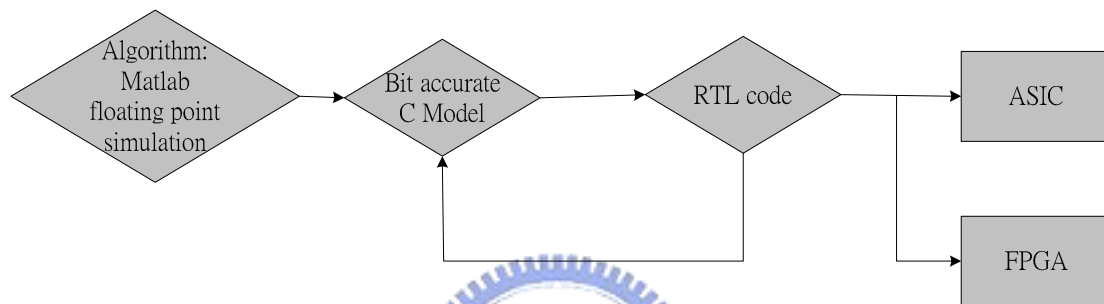


圖 6-1：設計流程

### 6.1 THP 系統與 TCM 解碼器之模擬

為了驗證第五章所提的硬體架構的效能，我們以 Matlab 與 C 語言把硬體架構的 bit accurate 模型以程式語言的方式描述，這個步驟可以確保硬體架構能正確運作後才進行 RTL 之硬體實現。

#### 6.1.1 THP 系統之軟體模擬

	FSE	Feedback Filter	TH Precoder Filter
Coeff. Word-length	16	16	16
Input data word-length	14	1	14
Tap length	64(max)	128	128

表 6-1：THP 系統係數與資料位元寬度

本論文設計之 Tomlinson-Harashima Precoder 系統之係數與資料等之量化位元

寬度如表 6-1 所示，14 位元為資料之量化寬度。THP 系統皆以 16 位元的量化寬度實現，並以 C 語言描述硬體行為，做 THP 系統之模擬。

### 6.1.1.1 THP 系統訓練模式

使用兩種不同通道模型傳輸訊號，在接收端有 AWGN 或者 cross talk 的干擾源，其模擬模型方塊圖如圖 6-1 所示。模擬之離散通道模型為以下兩種：

- 通道一：傳輸速率 1.544MHz，迴路長度 6400 呎，無 bridge taps。
- 通道二：傳輸速率 1.544MHz，迴路長度 6400 呎，有 bridge taps。

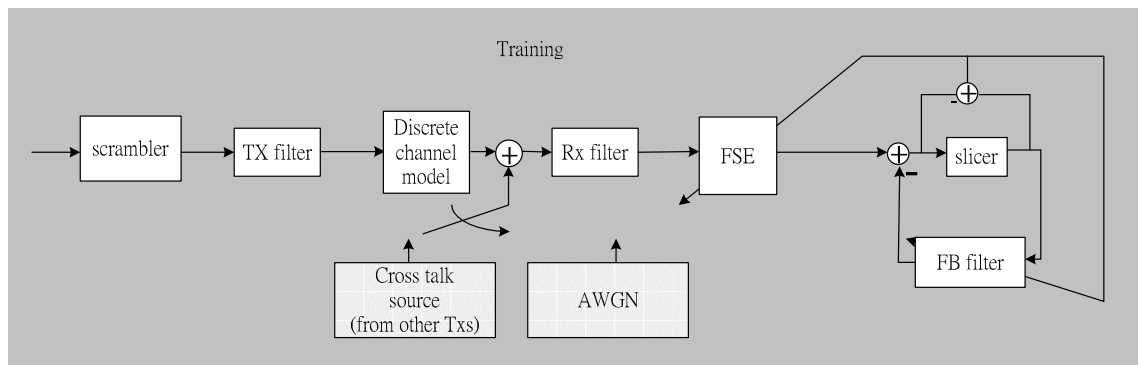


圖 6-2：訓練模式之模擬方塊圖

- AWGN：

圖 6-3、6-4 是以通道一與通道二傳輸之訓練模式，雜訊為 AWGN，其 SNR 為 20dB 到 23dB 的環境。

在通道一與通道二的環境模擬得到了，圖 6-3 與 6-4 各有 4 條曲線。同樣都以 AWGN SNR 23dB 的雜訊干擾下，通道一的收斂行為會比通道二的收斂較好，這是由於通道環境條件不同。

要觀察經等化器後的結果，通常以眼圖為依據，如果眼睛張的越開，表示 SNR 值較佳且不容易受取樣影響。圖 6-5、6-6 分別為模擬通道一與通道二傳輸模型在訓練模式之眼圖，為 slicer 之輸入訊號是 2-PAM 格式。兩張圖分別為收斂至 24dB 與 23dB，可以看出眼睛的形狀，表示 ISI 影響已經被降低，只剩白雜訊 SNR 為 24、23dB。

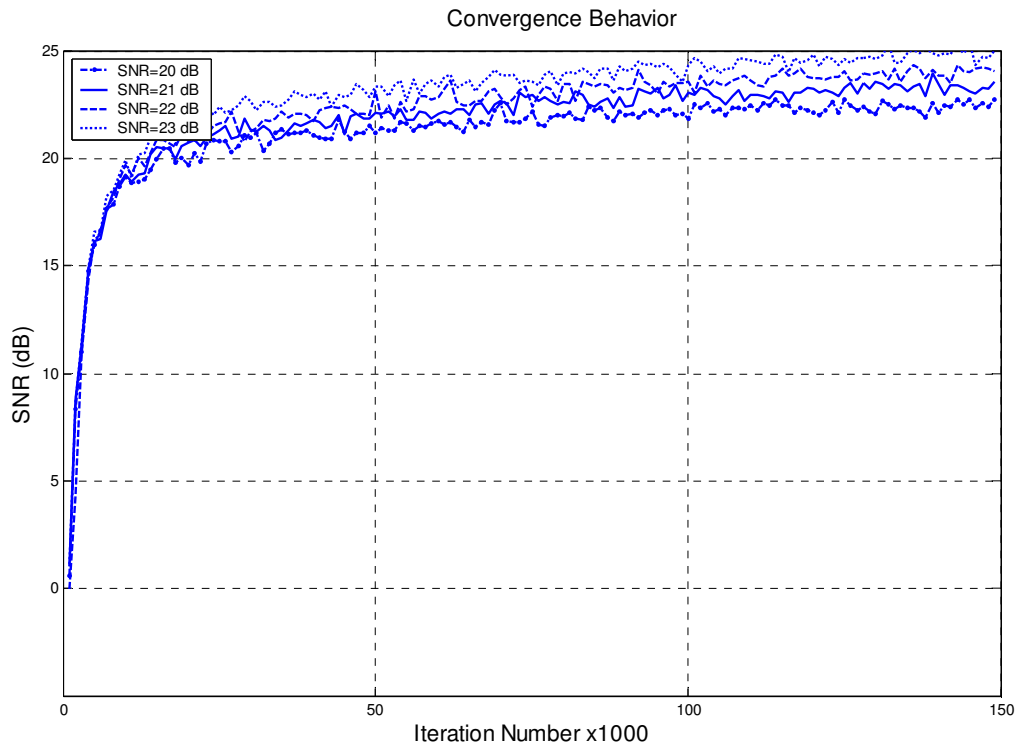


圖 6-3：AWGN，通道模型一之收斂速度

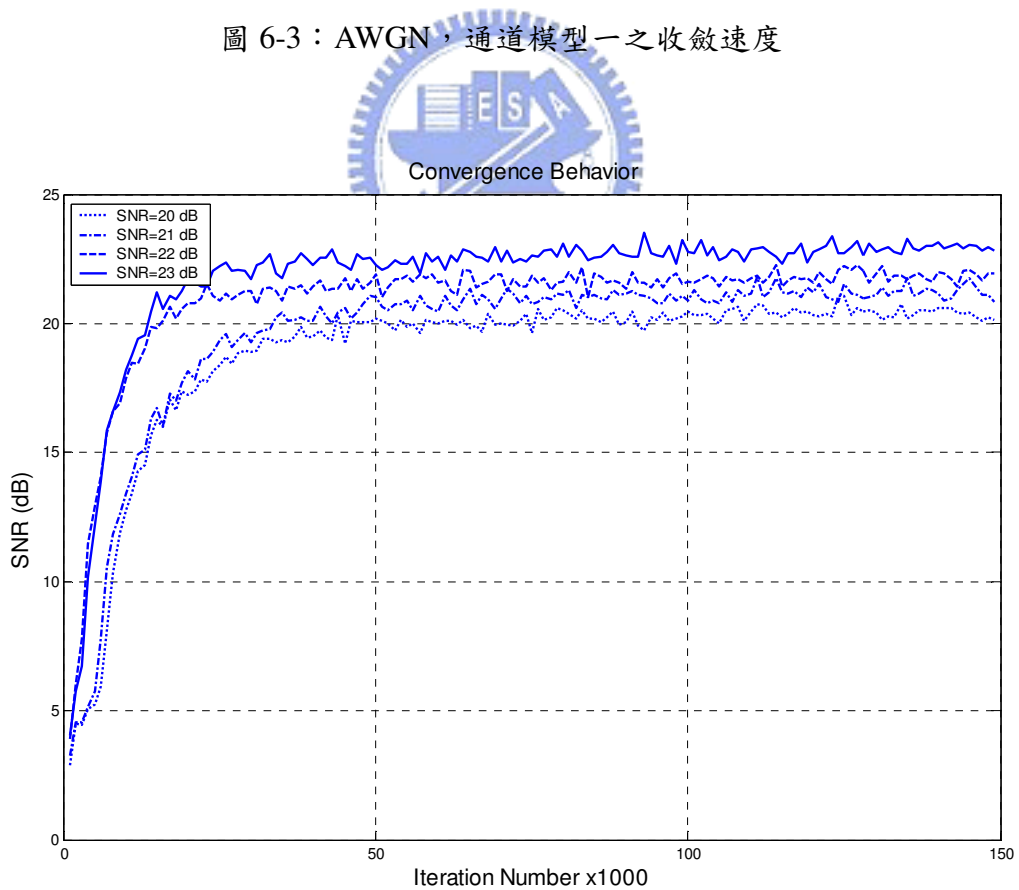


圖 6-4：AWGN，通道模型二之收斂速度



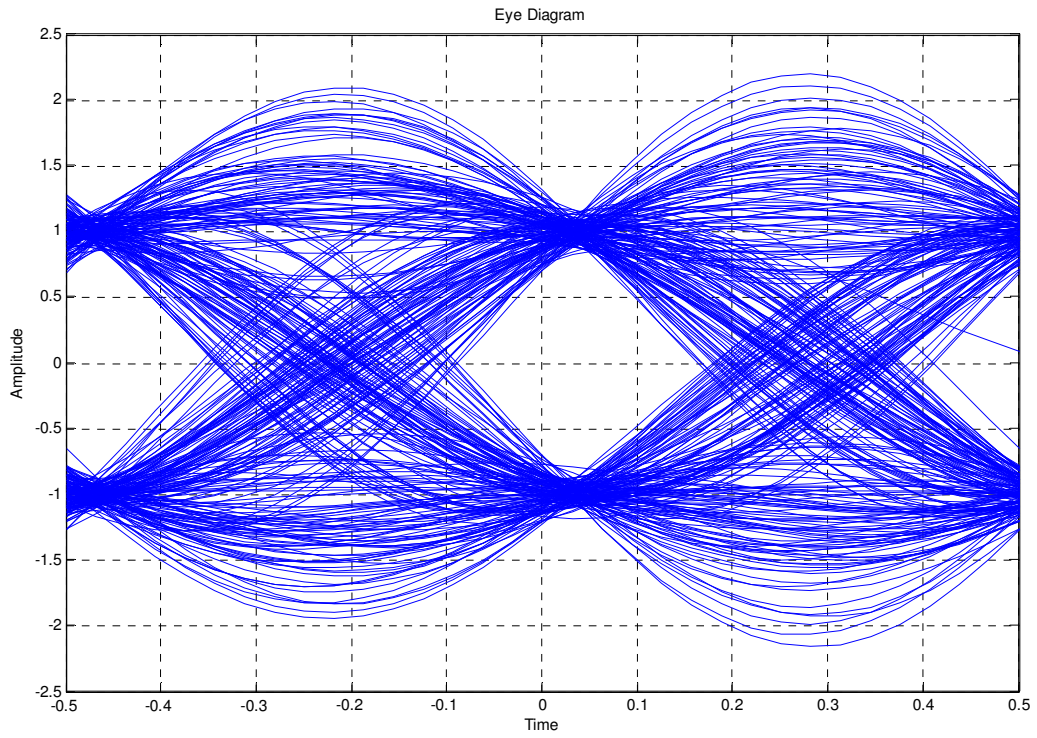


圖 6-5：AWGN，通道模型一—訓練模式收斂至 24dB 時之眼圖

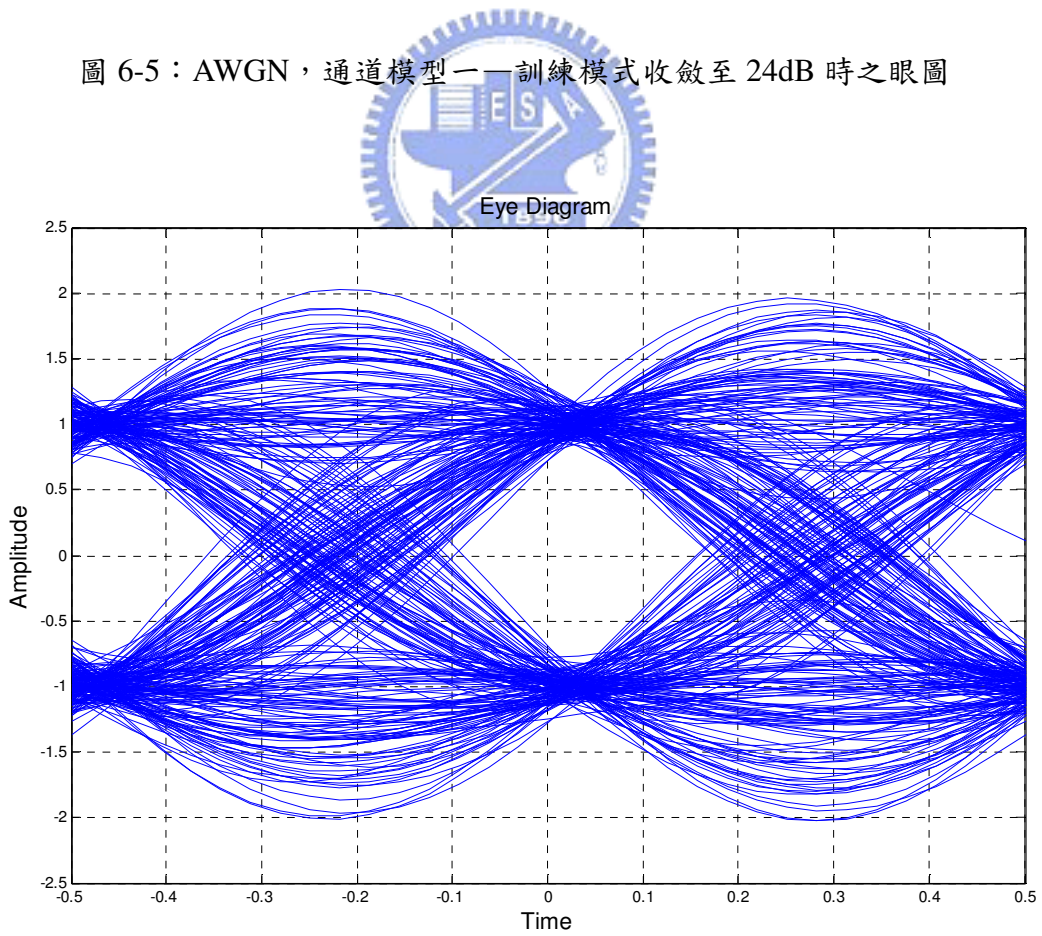


圖 6-6：AWGN，通道模型二—訓練模式收斂至 23dB 時之眼圖

- Cross talk :

圖 6-7、6-8 是以通道一與通道二傳輸之訓練模式，雜訊為來自其他發射機之 cross talk，這個發射機與圖 6-2 相同，只是被傳送的訊號與使用的 scrambler 不同，雜訊 SNR 為 20dB 到 25dB 的環境。

圖 6-7、6-8 模擬在通道一與通道二環境，雜訊 cross talk 之 SNR 由 20 到 25dB，得到六條不同收斂速度與趨近值。收斂後六條曲線的輸出 SNR 接近 SNR 20、21、22、23、24、25dB。

圖 6-9、6-10 分別為模擬通道一與通道二傳輸模型在訓練模式之眼圖，為 slicer 之輸入訊號是 2-PAM 格式。兩張圖分別為收斂至 24dB 與 23dB，可以看出眼睛的形狀，表示 ISI 影響已經降低，只有白雜訊 SNR 為 24、23dB。

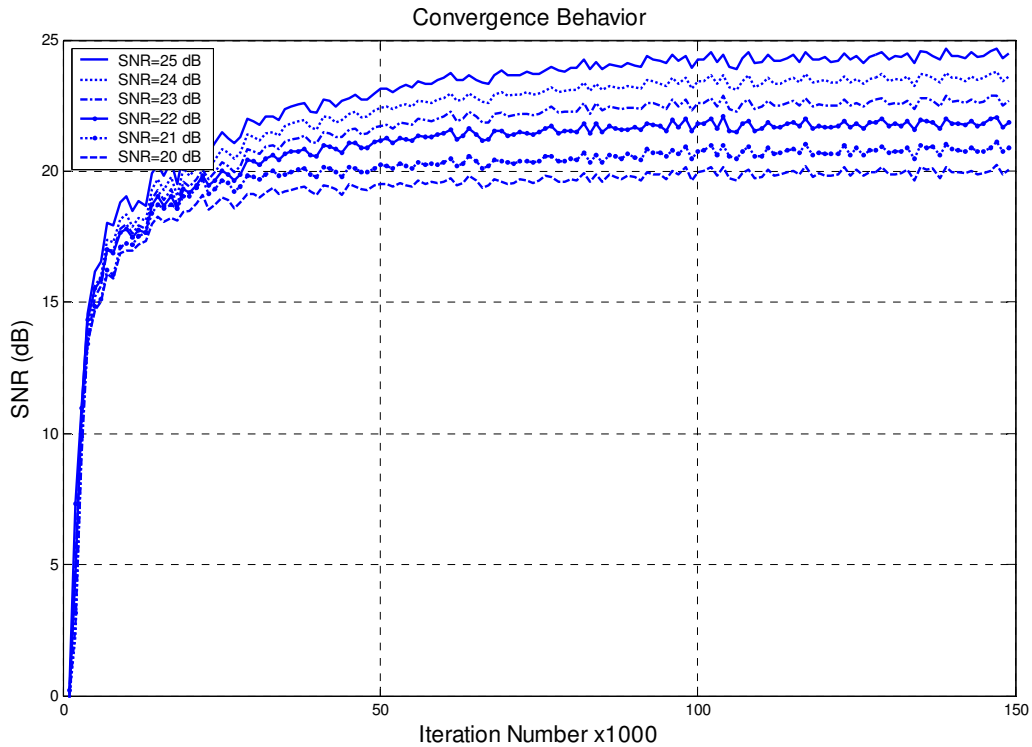


圖 6-7：Cross talk，通道模型一之收斂速度

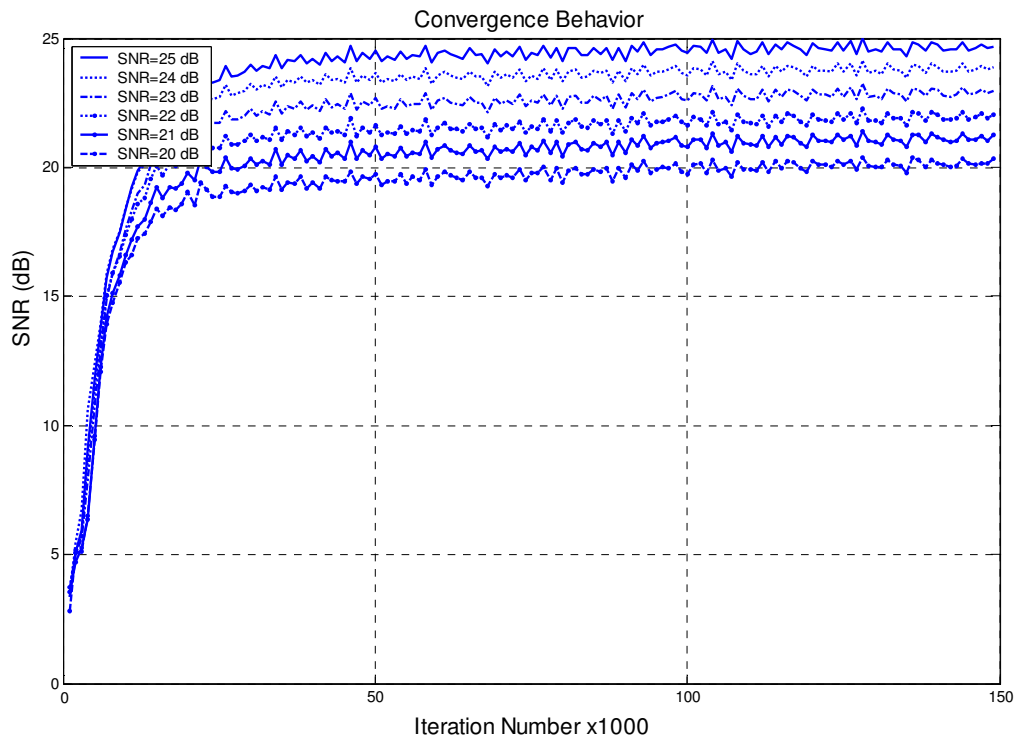


圖 6-8：Cross Talk，通道模型二之收斂速度

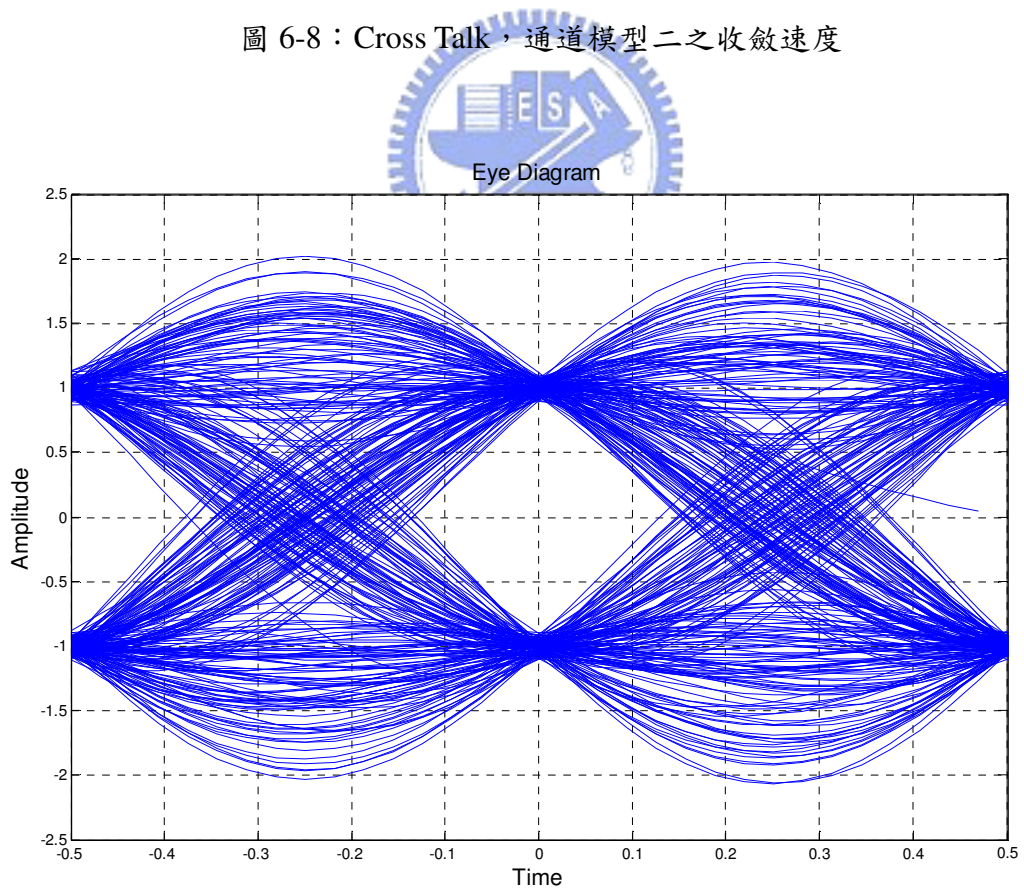


圖 6-9：Cross Talk，通道模型一—訓練模式收斂至 25dB 時之眼圖

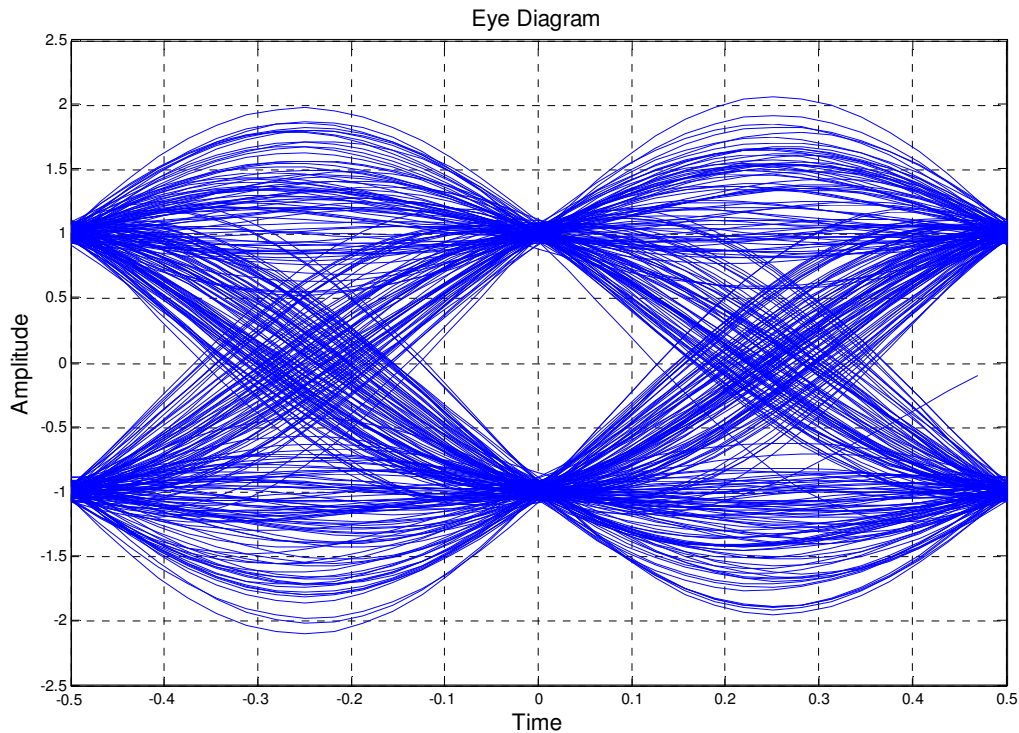


圖 6-10：Cross talk，通道模型二—訓練模式收斂至 25dB 時之眼圖

### 6.1.1.2 THP 系統資料模式

資料模式為 TC-16PAM 之訊號，在模擬時我們直接使用訓練模式時得到之 precoder 濾波器係數。下表為資料模式下等化器輸出結果。

	通道一 AWGN SNR=22dB	通道二 AWGN SNR=22dB	通道一 Cross talk SNR=23 dB	通道二 Cross talk SNR=23dB
THP 系統輸出 結果	24 dB	23 dB	23 dB	23 dB

表 6-2：THP 系統輸出結果

表 6-2 分別為在 AWGN 與 cross talk 不同雜訊下，以通道一、通道二傳輸的模擬結果。其傳輸環境與訓練模式時相同。圖 6-11 為資料模式之眼圖，雜訊為 AWGN，圖中可以看到 15 個眼睛。

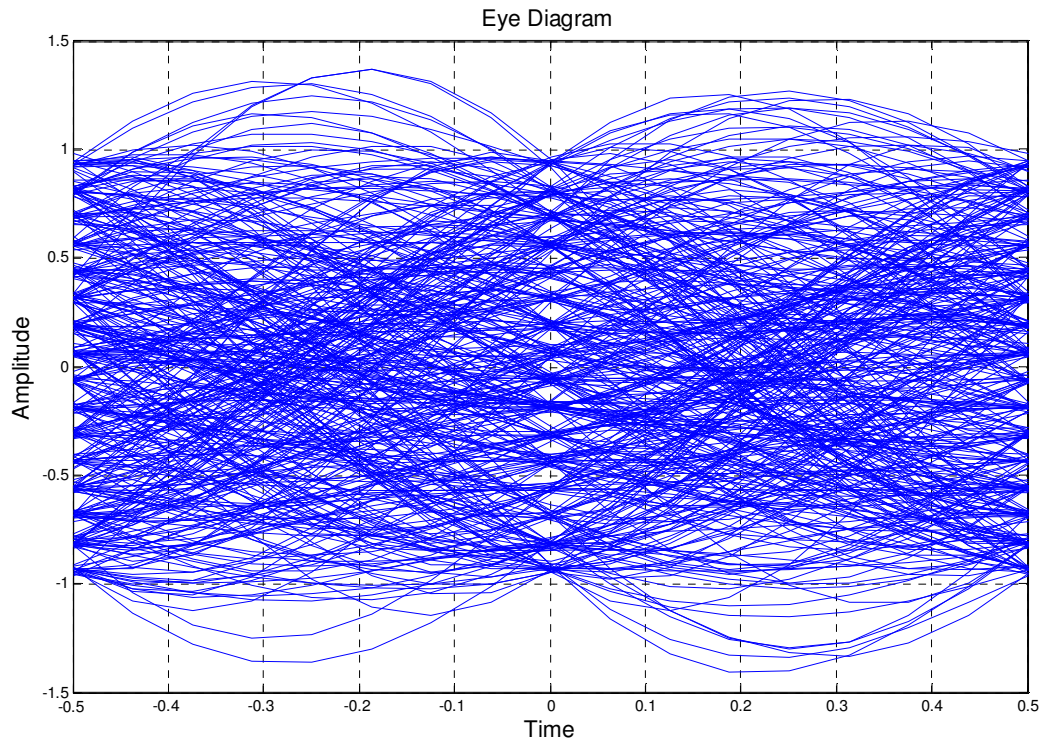


圖 6-11：資料模式 16PAM 眼圖

## 6.1.2 TCM 解碼器

TCM 解碼器之硬體架構在第五章已經清楚的介紹了。根據[8]，TCM 解碼器之存活深度  $D$  採用 5 倍的限制長度  $D=5K$ 。我們模擬在高斯通道下 TCM 解碼器之位元錯誤率，比較 SNR=18 至 28dB 的範圍，並以未編碼 8-PAM 的系統相較解碼後之位元錯誤率。理論上 TCM 系統可以達到 3-6dB 之編碼增益，圖 6-12 為本論文設計之 TCM 解碼器可以看到當位元錯誤率在  $10^{-7}$  附近時，與未編碼 8-PAM 比較，其編碼增益可達 4dB。

圖中兩條 TCM 解碼器的 BER 曲線 VD1、VD2 代表不同 word-length 的硬體架構，區分分支計量與狀態計量的量化位元數不同。VD1 硬體架構以 8 位元量化分支計量、12 位元量化狀態計量。VD2 硬體架構以 10 位元量化分支計量、14 位元量化狀態計量。圖中的 BER 曲線在 SNR=21.5 至 22.5 dB 時 VD2 表現略為優秀，在低 SNR 時 VD1 與 VD2 的增益都差不多，本論文之硬體實現則以 VD2 的 word-length 為藍本。

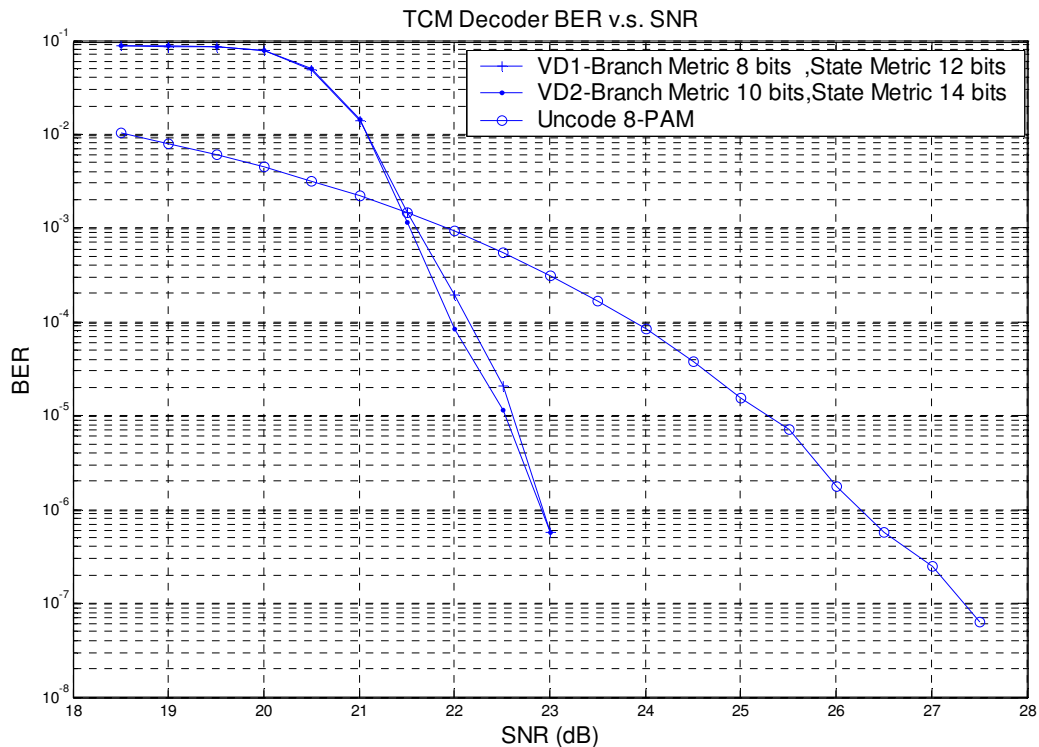


圖 6-12：TCM 解碼器與未編碼 8-PAM 位元錯誤率之比較

## 6.2 硬體實作與量測

### 6.2.1 硬體設計流程

本節的硬體實作將用 ASIC 流程與 FPGA 流程兩方式來驗證。其中關於 ASIC flow 方面，因為我們只想知道其硬體合成完後的邏輯閘數目 (gate counts)，所以我們只完成到 Design Compiler 合成 (synthesis) 步驟及 gate simulation，關於 Place & Route 等後端工作並不實現它。至於 FPGA flow，我們將利用 QuartusII 軟體模擬我們所設計的 RTL code，將其合成的電路下載至 FPGA 模擬板上做硬體驗證。

### 6.2.2 ASIC 流程

我們 ASIC 的設計流程如下：

Architecture design => RTL coding => RTL synthesis => Gate level simulation。

至於 physical design 與 verification 方面我們並沒有進行，這可以在未來實現。我們使用 Verilog Hardware Describe Language 撰寫 RTL，這是很多工程師普

遍使用的一種硬體描述語言，在這就不在多作介紹了。因為我們的硬體設計都是 memory-based，所以本論文設計的硬體會使用到記憶體，在 ASIC 流程我們利用 Artisan 公司設計的軟體可以編譯出我們要的記憶體，也就是 RASDxx.v 的檔案，這些都是 block boxes 我們只能利用其模擬記憶體的功能，並不能知道這些記憶體真正的內部 layout。

ASIC 方面都是以工作站做電路的模擬、驗證與合成等工作。RTL code 撰寫完成之後，我們先做功能的模擬，以 Debussy 觀察輸出波形，驗證功能正確。然後使用 Synopsys 的 DC(design compiler)將 RTL 合成電路，使用 TSMC 0.25 $\mu$ m 製程的 library，其合成電路的結果在表 6-3。合成電路經過 gate level 的模擬，在操作頻率 50MHz 的條件下，得到表 6-3 的結果。

	TCM decoder	THP	DFE
Gate counts	40,625	6,431 (only precoder filter)	11,612 (feedforward filter and feedback filter)

表 6-3：TCM 解碼器與 THP 系統硬體之邏輯閘數目

(不含記憶體，TSMC 0.25 $\mu$ m 製程，clock constraint 為 50MHz)

### 6.2.3 FPGA 流程

使用 FPGA 驗證板流程如下：

Architecture design => RTL coding => RTL synthesis (using Quartus II ver.2.2) => Gate level simulation => Programming => Verification (using Logic analyzer)。

當硬體架構設計完後，我們必須撰寫 RTL code，而 ASIC 與 FPGA 流程同樣都是利用相同 RTL code 作電路合成。在 FPGA 流程，我們利用電腦輔助設計軟體 QuartusII 將我們所設計的 RTL code 作電路合成與佈局。唯一不同 ASIC 流程的地方就是使用記憶體的方式，QuartusII 有提供記憶體的模組，所以必須使用 CAD 提供的記憶體。

當 RTL code 撰寫完成之後，再把 RTL 電路合成，使用 QuartusII 可以選擇被合成後要 programming 的 FPGA 晶片，我們會選擇 Stratix EP1S25F780C5 這塊

晶片，直接把電路的結果 programming 在晶片上，我們就可以馬上在 FPGA 發展板上做電路驗證的工作。基本上在 FPGA 上所得到的最大操作頻率會比在 ASIC 上差，因其 FPGA 內部的佈局並不是最佳狀態，故會有比較嚴重的 wire delay 發生。所以一般來說在 ASIC 上所得到的操作頻率會是在 FPGA 上的兩倍以上。

驗證環境：

1. CAD：Quartus II ver.2.2。
2. FPGA 模擬板：DSP development board, Stratix edition，  
Altera Stratix EP1S25F780C5 device。
3. 邏輯分析儀與資料產生器：Agilent 1692A 與 Acute PG2050。
4. 示波器：Tektronix TDS3032B。

硬體驗證方式：

1. 將輸入資料先行儲存於 FPGA 模擬板所提供的 ROM 裡面，當資料必須開始輸入時，會依序的從 ROM 裡面把資料讀出並將資料饋至我們所設計的電路輸入端。時脈訊號由 FPGA 模擬板所提供之 PLL 來產生，直接接至電路之 clock 的腳位。
2. 我們已經先行把所設計電路輸出之腳位分配好在板上提供的可測量腳位，在把邏輯分析儀的探針一一的接上這些腳位，即可測量輸出訊號。

如圖 6-13，為電路驗證環境的照片，圖中有邏輯分析儀、資料產生器、示波器、FPGA 發展板與個人電腦。





圖 6-13：電路驗證環境

## 6.3 結論

本論文實現了 SHDSL 規格的基本需求，如果最大傳輸速率 2.3Mbps 的條件下，本硬體架構只需要 49.06MHz 的操作頻率即可符合需求。在符合傳輸速度的條件下而不用太高的操作頻率，還可以節省功率消耗。雖然在 FPGA 上量測到的最高操作頻率為 42MHz 還差需求 49.06MHz 一些，不過在這裡 FPGA 模擬板是作功能驗證的工作，且沒有把電路佈局作最佳化，wire delay 很嚴重導致操作頻率的下降，在 ASIC 方面這些缺點將會改善，所以其操作頻率可以很容易的提升。

---

## 第七章 總結與未來展望

---

在本論文中，實現了結合 Tomlinson-Harashima precoder[4,5]與 TCM[6,7,8] 解碼器硬體架構，在 G.SHDSL 規範中採用了此種架構，對於 ISI 能夠有效的抵抗。TCM 解碼器可以提供約 4dB 的編碼增益。

THP 系統之數位濾波器以乘法累加器實現(MAC)，可以避免使用大量之乘法器來實現大量的係數長度，並以 embedded SRAM 儲存大量的資料及係數。其中接收機的前饋濾波器是 fractionally-spaced，並以 polyphase 方式[18]實現來省下大量的運算。

TCM 解碼器部分，如同第五章所述是 radix-4 memory-based 之硬體架構，以 radix-4 架構來得到比傳統 radix-2 架構快一倍的速度。只需要一個 radix-4 butterfly 就可以運算 512 個狀態之 ACS，以 50MHz 的操作頻率即可以達到 2.3Mbps 的產量(throughput)。

在第二章提到 SHDSL PMD 層的主要功能有：

- 符元時序(timing)產生與恢復。
- 編碼與解碼。
- 調變與解調。
- 回音(echo)消除。
- 等化(line equalization)。
- 啟動連結(link startup)。

本論文主要工作在實現第二項及第五項，我們已把編碼與解碼與等化的硬體實現實現，完整的 PMD 層還可以繼續被實現，如時序恢復等的研究。時序恢復在通訊系統同樣是系統效能重要的課題，能夠設計一個精確同步擷取的接收機可以使效能更好。未來將可研究此課題。除此之外，希望可以整合所有子系統實際的做出一顆積體電路，並作功能的驗證。

## 参考文献

- [1] ITU-T, "Single-Pair High-Speed Digital Subscriber Line (SHDSL) Transceivers," G.991.2, Huntsville, Ontario, Aug., 2000.
- [2] J.M. Cioffi, "MMSE decision-feedback equalization and coding-part I: general results," IEEE Trans. Commun., vol. 43, no. 10, pp. 2582-2594, Oct., 1995.
- [3] Robert F. H. Fischer, "Precoding and signal shaping for digital transmission," IEEE Inc., New York, 2002.
- [4] M. Tomlinson, "New automatic equalizer employing modulo arithmetic," Electron. Lett., vol. 7 nos. 5 and 6, pp. 138-139, Mar. 1971.
- [5] H. Harashima and H. Miyakawa, "Matched-transmission technique for channels with intersymbol interference," IEEE Trans. Commun., col. COM-20, pp. 774-780, Aug. 1972.
- [6] G. Ungerboeck, "Channel coding with multilevel/phase signals," IEEE Trans. Inform. Theory, vol. IT-28, pp.55-67, 1982.
- [7] S.B. Wicker, "Error control systems for digital communication and storage", Prentice-Hall Inc., 1995.
- [8] G. Clark and J. Cain, "Error correction coding for digital communications," New York: Plenum, 1981.
- [9] Gregory J. Pottie, M. Vedat Eyuboglu, "Combined Coding and Precoding for PAM and QAM HDSL Systems.", IEEE Journal on selected Areas in Commun., Vol. 9, No. 6, pp. 861-870, Aug. 1991.
- [10] Ahmad K. Aman, Robert L. Cipo, Nikolaos A. Zervos, "Combined Trellis Coding and DFE through Tomlinson Precoding.", IEEE Journal on selected Areas in Commun.,

Vol. 9, No. 6, pp. 876-884, Aug. 1991.

[11] G.D. Forney, Jr. , M.V. Eyuboglu, "Combined equalization and coding using precoding," IEEE Commun., Magazine, pp. 25-34, dec., 1991.

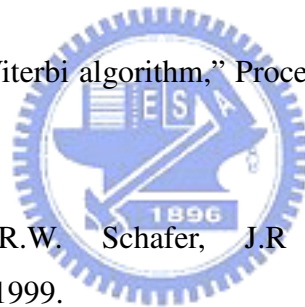
[12] ITU-T, "Handshake Procedures for Digital Subscriber Line (DSL) Transceivers," G.994.1, June, 1999.

[13] J.G. Proakis, "Digital Communications." New York, USA:MrGraw-Hill, 4<sup>th</sup> ed., 2000.

[14] A.J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Trans. On Information Theory, vol. IT-13, pp. 260-269, Apr. 1967.

[15] G.D. Forney, JR., "The Viterbi algorithm," Proceedings of the IEEE, vol. 61, pp. 268-278, Mar. 1973.

[16] A.V. Oppenheim, R.W. Schafer, J.R. Buck,"Discrete-Time Signal Processing."Pretice-Hall Inc.,1999.



[17] P. Black, T. Meng, "A 140MBit/s, 32-State, radix-4 Viterbi decoder," IEEE Journal of Solid-State Circuits, vol. 27, no. 12, pp. 1877-1885, dec. 1992.

[18] C.B. Shung, P.H. Siegel, G. Underboeck, H.K. Thapar, "VLSI Architectures for Metric Normalization in the Viterbi Algorithm," ICC'90, vol. 4, pp. 1723-1728, 1990.

[19] Marc Davio,"Kronecker Products and Shuffle Algebra," IEEE Trans. on Computers, Vol. C-30, No. 2, Feb. 1981.