

國立交通大學

電信工程學系碩士班

碩士論文

IEEE 802.11n 基頻接收機設計與實現



Design and Implementation of IEEE 802.11n

Baseband Receiver

研究生：莊秉卓

指導教授：吳文榕 博士

中華民國九十四年七月

IEEE 802.11n 基頻接收機設計與實現

Design and Implementation of IEEE 802.11n

Baseband Receiver

研 究 生：莊秉卓

Student：Bing-Juo Chuang

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



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

IEEE 802.11n 基頻接收機之設計與實現

Design and Implementation of IEEE 802.11n

Baseband Receiver

研究生：莊秉卓

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

IEEE 802.11n 被視為是下一代高速無線通訊的規格。其基頻的主要特性是利用多重輸入輸出正交分頻多工(MIMO-OFDM)技術。在本論文中，針對 IEEE 802.11n 接收機做設計與實現(以 TGn Sync 的提案為準)。我們將接收機分為前端與後端兩部分。前端的部分包括了封包偵測、自動增益控制、頻率偏移估計、碼框偵測、通道估計與快速傅利葉轉換等模組。而後端的部分則包括了最小均方誤差檢測(MMSE Detection)，軟性位元反對應(soft-bit demapper)與維特比解碼器(Viterbi Decoder)。在此我們設計了接收機的前端並對整個接收機做系統模擬。根據 FPGA 的設計流程，我們實現了 2x2 的前端接收機。在這個設計之中，我們用了座標旋轉數位電腦(CORDIC)等技術來做相位的估計與旋轉，也針對封包偵測與頻率偏移估計提出了有效的架構。由模擬圖可看出我們的在隨機產生的 MIMO 通道之下表現正確。

Design and Implementation of IEEE 802.11n Baseband Receiver

Student: Bing-Juo Chuang

Advisor: Dr. Wen-Rong Wu

Department of Communication Engineering
National Chiao-Tung University

Abstract

IEEE 802.11n is known as the specification for the next generation high-speed WLAN systems. The distinct baseband feature is the use of multi-input multi-output (MIMO) OFDM technology. In this thesis, we consider the design and implementation of an IEEE 802.11n baseband receiver (with the TGn Sync proposal). We divide the receiver into the front-end and the back-end receiver. The front-end receiver includes modules of packet detection, automatic gain control, frequency offset estimation, frame detection, channel estimation, and fast Fourier transform (FFT). The back-end receiver includes modules of the minimum mean square error (MMSE) signal estimator, the soft-bit demapper, and the Viterbi decoder. We first design the front-end receiver and perform system simulations for the whole receiver. Using the FPGA design flow, we then implement the front-end receiver for a 2x2 system. In the design, we use the CORDIC algorithm for phase estimation and rotation and propose efficient structures for packet detection and frequency offset estimation. Simulations show that our design perform properly in random generated MIMO channels.

誌 謝

本篇論文得以順利完成，首先要特別感謝我的指導教授 吳文榕博士，在課業學習與論文研究上不厭其煩的引導我們正確的方向。同時感謝口試委員李大嵩教授、鐘嘉德教授與紀翔峰，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。

另外，我要感謝謝雨濤學長、陳仁智學長、楊華龍學長、李彥文學長、許兆元學長與李俊芳學長等在研究上不吝指導，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，最後感謝我家人及女友宜汝，給予我在精神上最大的鼓勵與支持，使得我可以順利地完成碩士學位。



內容目錄

中文摘要.....	I
ABSTRACT.....	II
誌謝.....	III
內容目錄.....	IV
表目錄.....	VI
圖目錄.....	VII
第 1 章 緒論.....	1
第 2 章 無線區域網路 802.11N 標準介紹.....	3
2.1 前言.....	3
2.2 802.11N 中前導訊號(PREAMBLE)形式.....	4
2.2.1 前導訊號簡介.....	4
2.2.2 前導訊號詳情.....	6
2.3 其他規格說明.....	9
2.3.1 主要架構介紹.....	9
2.3.2 空間資料流對應天線轉換 (Antenna Map Transformation).....	11
第 3 章 系統同步與檢測演算法.....	14
3.1 封包偵測(PACKET DETECTION).....	14
3.1.1 Double Sliding Window Detection (DSWD).....	14
3.1.2 Delay, Correlated and Normalized Detection (DCND).....	16
3.1.3 模擬結果與討論.....	17
3.2 頻率偏移估計(FREQUENCY OFFSET ESTIMATION).....	19
3.2.1 Time Domain Approach for Frequency Synchronization.....	20
3.2.2 模擬結果與討論.....	22
3.3 通道估測(CHANNEL ESTIMATION).....	26
3.4 最小均方誤差檢測(MMSE DETECTION).....	28
3.5 模擬(SIMULATION).....	30
第 4 章 硬體設計與架構.....	38
4.1 設計流程.....	38
4.2 重要功能區塊 (FUNCTIONAL BLOCKS).....	39
4.2.1 CORDIC(Coordinate Rotation Digital Computer)演算法.....	39
4.2.2 快速傅利葉轉換(FFT).....	48
4.3 封包偵測 (PACKET DETECTION).....	55

4.4 自動增益控制 (AGC , AUTO GAIN CONTROL)	57
4.5 碼框偵測 (FRAME DETECTION)	59
4.6 頻率偏移估計 (FREQUENCY OFFSET ESTIMATION)	60
4.7 通道估計 (CHANNEL ESTIMATION).....	63
4.8 模擬結果	64
4.9 802.11N 接收機實現.....	66
4.10 實作心得	70
第 5 章 結論.....	71
參考文獻.....	73



表目錄

表格 2-1 Tone partitioning into sets for 20MHz(56 tones)	9
表格 2-2 PHY Feature Set	9
表格 3-1 MSE of perfect CSI and Estimated CSI (C: Channel Estimate using (3-14) equation。 I : Channel Estimate with Interpolation).....	35
表格 4-1 硬體粗略複雜度計算	67



圖目錄

圖 2-1 802.11a/g and 802.11n preamble format	4
圖 2-2 Legacy Preamble	6
圖 2-3 L-SIG field bit assignment	7
圖 2-4 20MHz, $N_t = 2$ 下 HT-LTF 前導訊號.....	8
圖 2-5 MIMO datapath	10
圖 2-6 convolutional encoder ($k = 7$).....	10
圖 2-7 Puncturing pattern for code rate = $5/6$	10
圖 2-8 兩根天線下, 11n 交錯器形式	11
圖 2-9 Time domain implementation of spatial spreading cyclic delay	12
圖 2-10 Spatial Spreading via \mathbf{Q}	13
圖 3-1 Double Sliding Window	15
圖 3-2 DSWD 中, 不同 SNR 之下比較 P_f 與 P_m	18
圖 3-3 DCND 中, 不同 SNR 之下比較 P_f 與 P_m	19
圖 3-4 MSE of normalized frequency offset for 1x1 and 2x2 ($f_o = 100\text{KHz}$)	23
圖 3-5 MSE of normalized frequency offset ($f_o = 100\text{kHz}$, $D=16$).....	24
圖 3-6 MSE of normalized frequency offset ($f_o = 100\text{kHz}$, $D=16$).....	24
圖 3-7 MSE of normalized frequency offset (2x2, $D=16$).....	25
圖 3-8 MSE of normalized frequency offset(2x2, $D=16$).....	26
圖 3-9 HT-LTF(TGn Sync 三月版).....	27
圖 3-10 HT-LTF(TGN Sync 五月版).....	28
圖 3-11 2x2 下通道估計示意圖	28
圖 3-12 多重路徑衰落通道示意圖	31
圖 3-13 Average Power of Exponential Decay Multipath Channel.....	32
圖 3-14 MIMO-OFDM floating point performance (no impairment).....	33
圖 3-15 Frequency offset impairment.....	34
圖 3-16 MIMO-OFDM performance (Perfect vs. Channel Estimate).....	35
圖 3-17 Perfect vs. Channel Estimation with interpolation	36
圖 3-18 802.11n 傳送接收機	37
圖 4-1 硬體設計流程圖	38
圖 4-2 向量旋轉示意圖	39
圖 4-3 CORDIC Algorithm with Vector Rotate (Rotating angle = $3\pi/2$)	44
圖 4-4 CORDIC Algorithm with Vector Rotate (Rotating angle = 4.93)	45
圖 4-5 CORDIC Algorithm with Angle Calculation ($1 - i$).....	46
圖 4-6 Iterative CORDIC structure.....	47
圖 4-7 Pipeline CORDIC structure	48
圖 4-8 八點 FFT(1).....	50

圖 4-9 八點 FFT(2).....	50
圖 4-10 Radix-2 與 Radix-4 基本元件比較	51
圖 4-11 PE of Radix- 2^2	51
圖 4-12 PE of Radix- 2^3	52
圖 4-13 16-points pipeline FFT	52
圖 4-14 Memory-Based FFT 示意圖	53
圖 4-15 64 點 Radix- 2^3 pipeline FFT	53
圖 4-16 PE of BF2	54
圖 4-17 Bit-Reversal output.....	54
圖 4-18 FFT Block(虛線部分)	55
圖 4-19 DCND 基本架構圖	56
圖 4-20 S 區塊示意圖	56
圖 4-21 SOP 整體架構與定點數(fix point)配置.....	57
圖 4-22 Power fluctuation of HT-LTF w.r.t. data.....	58
圖 4-23 AGC 硬體架構圖	58
圖 4-24 (a)Direct form (b) Transposed form (c) Hybrid form (with length 3 subfilters)	60
圖 4-25 Frame Detection 架構圖	60
圖 4-26 頻率偏移補償方塊圖	61
圖 4-27 Angle Generator 硬體架構圖 1.....	61
圖 4-28 Angle Generator 硬體架構圖 2.....	62
圖 4-29 Fix Angle Accumulator.....	63
圖 4-30 通道估計硬體架構圖	64
圖 4-31 MIMO-OFDM (floating point vs. fix point)	65
圖 4-32 MIMO-OFDM (AGC performance).....	66
圖 4-33 接收機 mapping report.....	68
圖 4-34 接收機 timing report	68
圖 4-35 接收機 RTL 架構圖	69

第1章 緒論

WLAN 近年來在市場上取得很大的成功，從 IEEE 802.11b 到 11g 和 11a，在傳輸率的提升也讓人相當肯定，進一步穩固了 802.11 系列在市場上的地位。然而，使用者對於頻寬的需求尚未滿足。多重輸入與輸出(Multiple-Input Multiple-Output, MIMO)結合正交分頻多工(Orthogonal Frequency Division Multiplexing, OFDM)被視為是下一代無線通訊最主要的技術，而目前 IEEE 進行制定的下一代標準 802.11n 也如火如荼的進行當中，其目標是將傳輸率由目前最快 54Mbps 一舉提高至 100Mbps 以上。

多重輸入輸出(MIMO)指的是一種在傳送端與接收端放置多根天線的無線電傳輸系統(radio communication system)。傳送端與接收端的多根天線可用來增加傳輸品質(減低錯誤率)或是提高傳輸速率。目前 MIMO 被視為能夠提供高資料速率的先進技術，這是因為它能夠增加系統的負載(capacity)而不需要付出額外的頻寬與能量。

正交分頻多工技術(OFDM)在 1960 年代中期即已提出，但在當時由於正交多載波硬體的製作相當困難且昂貴，而且當時的數位信號處理並不發達的，此技術沒有被廣泛的使用。直至 1971 年 Weistein 及 Ebert 在[10]提出利用離散時間富利葉轉換(DFT)來取代類比多載波的設計，此技術得以再度受到重視。在 1980 年 Peled 及 Ruiz 提出利用前置循環信號(Circular Prefix)來解決在多重路徑通道下的符際干擾(Inter Symbol Interference, ISI)問題。

在正交分頻多工系統中，輸入信號須以區塊處理，經過快速富利葉反轉換後，再經過一連串的处理傳送出去。而接收端必須經過區塊的同步後才能將正確的區間取出。如果區間取錯了，不但使多載波間的正交性被破壞，而且造成載波間彼此干擾，進而影響系統效能。其次因為接收端振盪器相對於傳送端的振盪器，通常無法完全的匹配，而有一頻率偏移(Frequency Offset)存在，造成載波間的正交性被破壞。尤其是在高階調變上，頻率偏移所造成的相位旋轉更是不容忽

視，因此在接收機的頻率偏移估算及其補償是維持系統正常運作的重要工作。在接收端除了頻率偏移估算之外，準確的通道估算(Channel Estimation)及通道效應補償的重要性亦是不容小覷。

而本論文將針對目前較受矚目之 TGn Sync 的提案，設計並實現接收機。我們將接收機分為前後兩個部分，接收機的前半部的功能包括封包偵測，頻率偏移補償，碼框偵測，自動增益控制，通道估計與快速傅利葉轉換，而接收機的後半部則包括 MMSE 偵測演算法與 Viterbi Decoder 的部分。本篇論文主要針對接收機前半部做研究與討論，並對其硬體架構提出一些構想，接著以電腦輔助設計軟體來模擬之，而接收機的後半部則由本實驗室另一位同學李峰宇同學所完成。論文主要採用 TGn Sync 系統中，2x2 根傳送接收天線，20MHz 與 64QAM 下做分析與硬體設計。下列概述各章節的內容：

第一章：簡述 MIMO-OFDM 技術之由來並概述本論文

第二章：無線區域網路 802.11n 提案介紹

第三章：同步與訊號檢測演算法的介紹

第四章：硬體架構設計

第五章：將本論文所提出的方法及效能分析作一個總結。

第2章 無線區域網路 802.11n 標準介紹

2.1 前言

在上一回 802.11g 制定時，就因為技術作法上的爭議而延遲了標準的推出。而這次參與制定 802.11n 提案的廠商更多了，去年六月 IEEE 802.11 Task Group N (TGn) 共收到了 62 件提案。在眾多提案當中，又以 WWiSE(World Wide Spectrum Efficiency) 和 TGn Sync 兩大陣營的提案最受重視，而這兩大陣營的提案也確定能夠出線，現在的問題只在需要花多久的時間來整合兩大陣營的差異，以達到大家都同意的標準。IEEE 802.11n 制定的進度希望在 2005 年 7 月投票決定第一個版本，並預計在 2007 年 3 月公布最終版本，而分屬兩陣營的領導廠商 Atheros 與 Airgo 都同意加速該標準的通過，也就是最終版本可望在 2006 年第二季確定，將比 IEEE 所設定的時程快上 9 個月。

WWiSE 陣營由 Airgo、Bermar、Conexant、TI 等廠商所組成，它們強調的重點是利用已經核准，現存且在全球適用的 20MHz 通道寬度上，確保任何電信法規都能立即使用及部署。在 2x2 組態配置對 20MHz 最低頻寬要求下，最高可達到 135Mbps。而在主管單位允許之下，利用 4x4 的架構和 40MHz 的通道寬度之下，則可達到將近 540Mbps 的資料傳輸速率。

TGn Sync(Task Group 'n' synchronization)陣營則是由 Atheros，Agere，Intel，Nokia，Cisco 等重量級廠商所組成，TGn Sync 的提案目標是到達最高傳輸速率 600Mbps，在普通雙天線組態下，也可以達到 243Mbps 的速度，但都得在 40MHz 的頻寬組態下運作。

其實由兩大陣營提出的提案看來，差別並不大，其基本的技術架構也很類似，都是採用 MIMO OFDM 技術，也都提供 2x2/4x4，20MHz/40MHz 等組態選擇，所以只要不過於堅持己見，應可以加快審核的進度。

本篇論文的有關於規格的部分，是以 2005 年三月 TGn Sync 改版的提案為主。

2.2 802.11n 中前導訊號(preamble)形式

2.2.1 前導訊號簡介

在 TGn Sync 所提的 802.11n proposal 中，高速前導符元(HT Preamble, High Throughput Preamble)主要可以分成兩部分，第一部分與 802.11a/g 的 preamble(Legacy Preamble, 或稱 Legacy PPDU)一樣，第二部分則為 802.11n 另外設計的 preamble，這樣設計的原因，是為了讓 802.11n 的系統也能夠跟舊有的 802.11a 與 802.11g 系統相容，增加其使用的彈性。

下圖我們可以很清楚的了解 802.11a/g 與 802.11n 中 preamble 的形式。

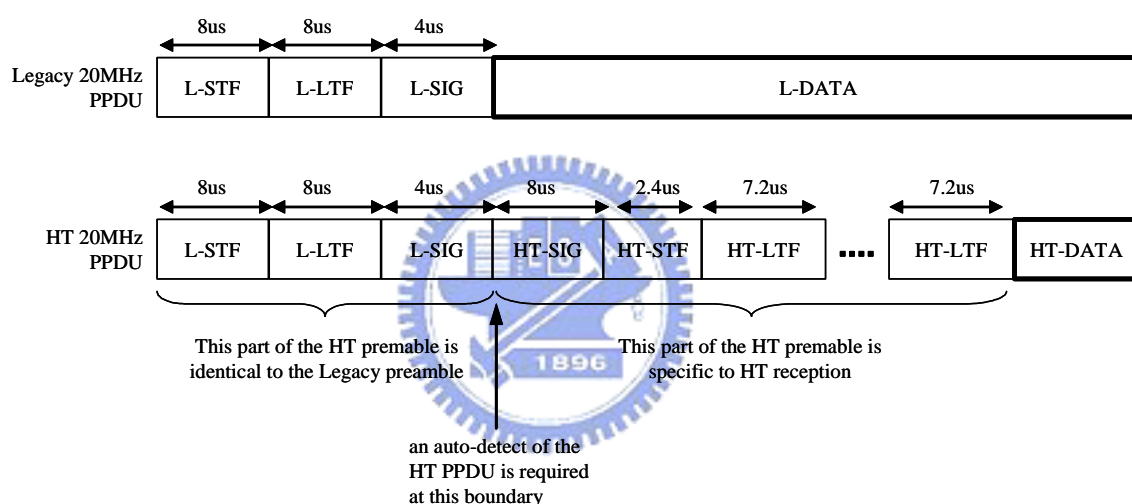


圖 2-1 802.11a/g and 802.11n preamble format

由上圖中，我們可以看到 HT preamble 是由上面所述的兩部分所構成，但是 HT 接收機(即 802.11n receiver)事前並不知道收到的是 legacy PPDU 或是 HT PPDU。因此，我們在 HT 接收機上發展一種自動偵測的機制放在 legacy signal(L-SIG)之後，讓 HT 接收機能夠判別出 L-SIG 之後接的是 HT signal (HT-SIG) 還是 legacy data(L-DATA)。其實現的方法很簡單，我們只要由以下兩個規定來自動偵測 HT-SIG:(a)以 BPSK 的訊號傳送 HT-SIG 在交軸(quadrature axis)而取代原本放在直軸(direct axis)上。(b)將 HT-SIG 上面的嚮導訊號(pilot)與 L-SIG 反向。經由這以上的兩個規定，在接收端接收時，只要在收完 L-SIG 之後，看看是否在直軸的能量明顯變少，而在交軸上的能量明顯增加，就能知道收的訊號是否為

HT 系統所傳送的了。

一般說來，preamble 的功能，大致分為下列幾種：

- (1) 封包偵測(Start-of-Packet detection):當天線在接收訊號時，我們無從得知是否傳送端傳送的訊號已經到達接收端，這時需要做封包偵測，或稱為粗略時序同步估計(coarse timing synchronization)，來判定訊號已經進到接收端了。
- (2) 自動增益控制(AGC ,Auto Gain Control):當傳送的訊號經過通道衰減之後，可能會因訊號的變小或變大，造成接收下來之後，經過 A/D resolution 不夠或是 overflow 太多的問題，所以我們利用 preamble 來估計接收下來的 power，再來調整之後接收訊號的大小。
- (3) 粗略頻率偏移估計(Coarse Frequency Offset Estimation):當傳送端與接收端的載波頻率有不匹配的情形時，便會發生頻率偏移，而造成接收訊號產生相位旋轉的現象。粗略頻率偏移估計，可以粗估出一個範圍較大但較不準的 frequency offset，再由之後估計範圍較小但較精準的的精確頻率偏移估計(Fine Frequency Offset Estimation)再補償一次，達到補償頻率偏移的目的。
- (4) 精準時序估計(Fine Timing Offset Estimation):或稱為碼框偵測(frame detection)。當接收端收到訊號之後，preamble 估通道的部分，還有後面資料的地方，必須先轉到頻率軸上(即做快速傅利葉轉換 FFT)，這時候的碼框偵測就是要抓準每一個做傅利葉轉換的碼框，這時候就會用到精準時序估計。通常碼框偵測的方法，都是用一個已知的訊號(如前導訊號)和接收到的訊號做比對，來得到碼框應有的位置。
- (5) 精確頻率偏移估計(Fine Frequency Offset Estimation):請參考(3)的粗略頻率偏移估計(Coarse Frequency Offset Estimation)。
- (6) 通道估計(Channel Estimation):通道的估測是 preamble 中相當重要的一環，

估計的準確與否，也直接影響了最後的效能，所以在 TGn Sync 的 proposal 中，用來估測通道的前導訊號佔整個前導訊號的比例最高，其符元數目也隨著空間流(Spatial Stream)的數目不同而有所不同。

2.2.2 前導訊號詳情

(1) L-STF(Legacy Short Training Field):短訓練符元，其時域的訊號是由十個相同的區塊所組成，如下圖 2-2 所示，前面幾個區塊負責做封包偵測(Packet Detection)和自動增益控制(AGC) 粗調。在 L-STF 所做的自動增益控制區塊中，只是粗略的估測訊號的強弱，因為通常 L-STF 的部分，我們只會用第一根天線傳送出去。(註:TGn Sync proposal 規範 L-STF 至 HT-SIG 的部分可以只由第一根天線傳送)。後面幾個區塊，則利用區塊重複的特性，來做粗略頻率偏移補償。我們也可以將 L-STF 的後面幾個區塊和 L-STF 前面 CP 的部分，來做為符元時序控制(Symbol Timing)或碼框偵測(Frame Detection)。

(2) L-LTF(Legacy Long Training Field):長訓練符元，利用較長的重複區塊，來做精確頻率偏移估計(Fine Frequency Offset Estimation)。另一個用途則是初步的估計通道，用來解出之後的 L-SIG 與 HT-SIG，以便了解所接收封包的調變資訊(MCS, Modulation Coding Scheme)。

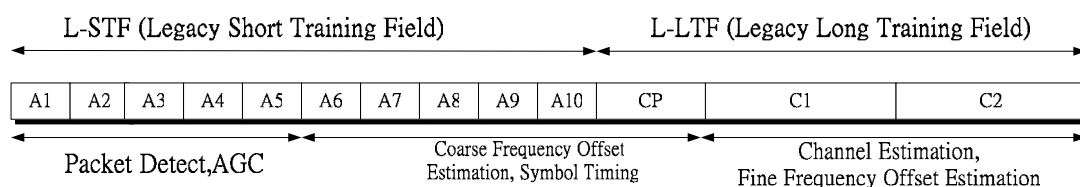


圖 2-2 Legacy Preamble

(3) L-SIG：在 20MHz 模式之下，L-SIG field 與 802.11a 規格中的 SIGNAL field 相同(定義在[1]的 17.3.4, figure 111)。如下圖 2-3 所示，其中 RATE

與 LENGTH 的組合可以代表在 HT 環境下的傳送的長度 (duration)。雖然如此，在 L-SIG 中的 RATE 與 LENGTH 並不需要與真實傳送的長度一樣，通常會使其長度大於在 HT-SIG 中的真實長度，這可做為在 MAC 層的一個保護機制，避免它誤認為只是單純的 802.11a/g 的封包。

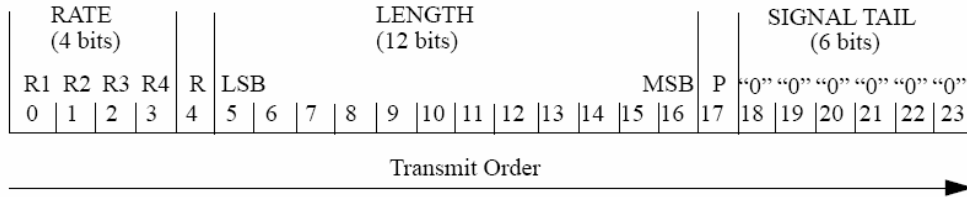


圖 2-3 L-SIG field bit assignment

(4)HT-SIG : HT-SIG 緊隨在 L-SIG 之後，它包括了更多的 HT PHY 的資訊，由於並不是本論文的重點，故在此不贅述，若有興趣可參考文獻[2]。

(5)HT-STF(HT-Short Training Field):高速短訓練符元，HT-STF 這個區塊，純粹是用來做自動增益控制(AGC)的微調，讓接收訊號能坐落在準確的 A/D 範圍之內，以確保在之後的 HT-LTF 能夠估出準確的通道，和準確的解出 HT-LTF 之後的傳送資料。為求估計出來的功率夠準確，我們會以頻率交錯(tone-interleaving)的方式，在不同的傳送天線之間，傳送在頻域上互不干擾的訊號，這樣的方法可以大幅的減少功率波動 (Power Fluctuation)，也就是能夠更準確的估出訊號的功率。更詳盡的討論將會在後面介紹。

(6)HT-LTF(HT-Long Training Field):高速長訓練符元，主要的用途是做通道的估計 (Channel Estimation)，也可以用來做頻率偏移的補償。與 HT-STF 相同的地方，在於估計通道的時候，傳送端是採用頻率交錯 (tone-interleaving)的方式，一方面除了要降低功率波動外，另一方面也能達到「完全空間與頻率通道估計」(Full Space-Frequency Observability)。以 20MHz， $N_{tx} = 2$ 為例子，其中其 preamble 定義為 $HTL_{-28:28} = [-1, -1, 1, \dots, 1, 1]$ ([2]中的 sequence 2)，而 $Set0_{-28:28}$ 的偶數項

($[-28:2:-2], [2:2:28]$)與 $HTL_{-28:28}$ 的偶數項相同， $Set0_{-28:28}$ 的奇數項 ($[-27:2:-1], [1:2:27]$)則全部放 0;相反的， $Set1_{-28:28}$ 的奇數項與 $HTL_{-28:28}$ 的奇數項相同，而偶數項則全部放 0。之後再將 $Set0$ 與 $Set1$ 經過快速傅利葉反轉換，依圖 2-4 的排列方式由天線發射出去，其中每根天線的導訊號(如下圖中的 HT-LTF0 與 HT-LTF1)重複了兩次，是為了在接受端能夠對兩次估到的相同通道做平均，以期能達到最佳的通道。由圖 2-4，我們可以清楚的看到，在同一時間區塊裡，不同天線所傳送的頻率位置會錯開，也就是可以在不受其它根天線干擾的情形下估出完整的通道，而另一方面，也會因為傳送天線的增加，HT-LTF 的數目也會跟著增加，為了避免前導訊號數目增加太多，在 $Set1$ 之後的前導訊號都只重複一次而已。另外三根天線時，如表格 2-1，分為 $Set0$ 、 $Set1$ 與 $Set2$ ，分別傳送 $(-28, -25, -22...26)$ 、 $(-27, -24, -21...27)$ 與 $(-26, -23, -20...28)$ 的方式做 Tone Interleaving。而四根傳送天線時則以此類推。

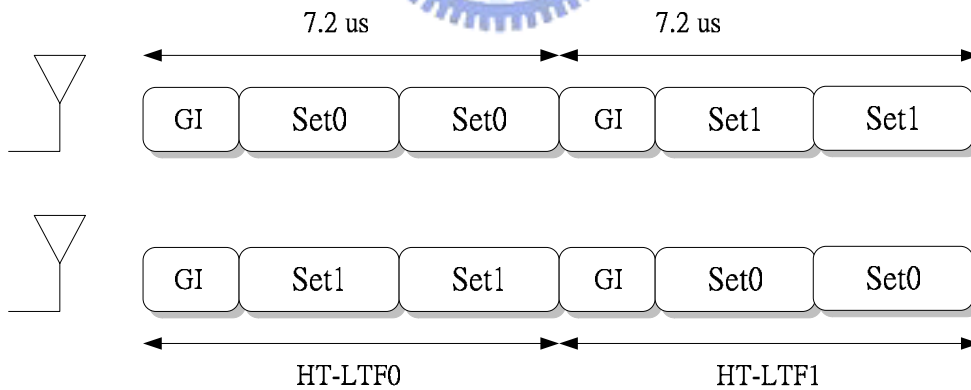


圖 2-4 20MHz， $N_t = 2$ 下 HT-LTF 前導訊號

Nss	Set 0	Set 1	Set 2	Set 3
1	[-28:1:-1] [1:1:+28]			
2	[-28:2:-2] [2:2:28]	[-27:2:-1] [1:2:27]		
3	[-28:3:-1] [2:3:26]	[-27:3:-3] [3:3:27]	[-26:3:-2] [1:3:28]	
4	[-28:4:-4] [1:4:25]	[-27:4:-3] [2:4:26]	[-26:4:-2] [3:4:27]	[-25:4:-2] [4:4:28]

表格 2-1 Tone partitioning into sets for 20MHz(56 tones)

2.3 其他規格說明

2.3.1 主要架構介紹

有關 TGn Sync 的主要實體層的規格如下所示。

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
Code Rate	1/2, 2/3, 3/4, 5/6	
Guard Interval	800ns	400ns
Convolutional Coding	R=1/2, K=7, ($g_1=133_8$, $g_2=171_8$)	

表格 2-2 PHY Feature Set

由表格 2-2 可以看出，與之前 802.11a 不同的地方，在於除了天線數的增加外，頻寬由 20MHz 到 40MHz，調變等級由原本最高 64QAM 到 256QAM，Code

Rate 也從 3/4 升高到 5/6，同時守護頻帶(Guard Interval)也選擇性的由 800ns 降至 400ns，所有的改變都為了提高無線網路的資料吞吐量。

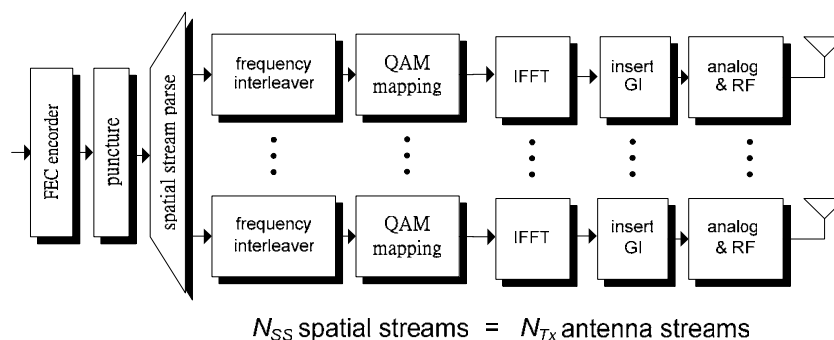


圖 2-5 MIMO datapath

圖 2-5 是 MIMO OFDM 傳送端的資料流程，先由上端 MPDU(Mac Protocol Data Unit)的資料經過編碼器(FEC encoder)，這個編碼器為編碼速率 1/2 的迴旋碼(convolutional code，圖 2-6)。

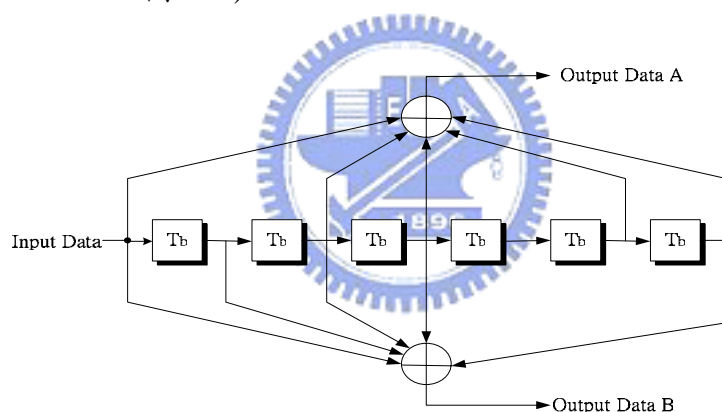


圖 2-6 convolutional encoder ($k = 7$)

經過編碼的位元，可再由打洞(puncture)的方式，提高編碼速率至 2/3，3/4 或 5/6(參考圖 2-7)。

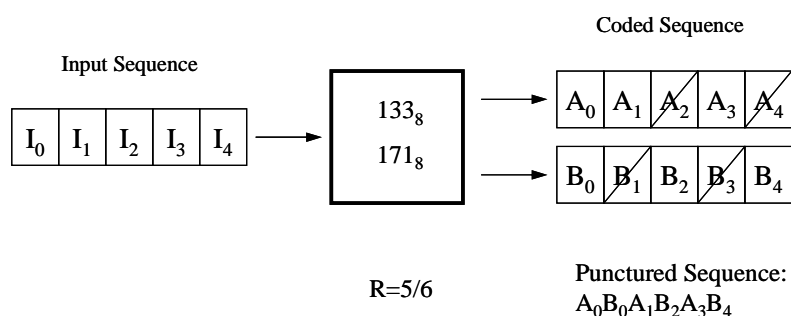


圖 2-7 Puncturing pattern for code rate = 5/6

接著，為了避免迴旋碼容易產生一連串錯誤的情形(burst errors)，我們需要做資料交錯器(Data interleaving)。這裡的資料交錯器又分為空間與頻率上的交錯(space-frequency interleaving)，即空間分配(Spatial Parsing)與頻率交錯器(Frequency Interleaver)。空間分配器的公式如(2-1)

$$s = \max\{N_{BPSC} / 2, 1\} \quad (2-1)$$

其中 N_{BPSC} 代表 Bits per subcarrier，例如 64QAM 時，BPSC=6，此時 $s=3$ ，假如這時的傳送資料流為 $2(N_{ss}=2)$ ，表示分 3 個位元給第一個資料流，接下來 3 個位元給第二個，再三個位元給第一個...以此類推。

而在頻率交錯器方面，包含三種交錯方式(圖 2-8)，也就是分成三個步驟(three-step permutation)。第一個步驟是將相臨的編碼位元(coded bits)分散到不相臨的載波之上，此舉是避免當某一個載波受到不好通道影響時，確保不會有連續的錯誤發生。第二個步驟則是將編碼位元輪流出現在最高位元(MSB)或最低位元(LSB)，避免最高位元一錯，就造成星狀圖的位置差距很大。第三個步驟為頻率旋轉(Frequency rotation)，就是在不同的空間流(spatial streams)上，做不同大小的載波旋轉，這可以避免在某個很差的頻率上，每個空間流都很差的機會。詳細交錯器的介紹，請參考[2]。

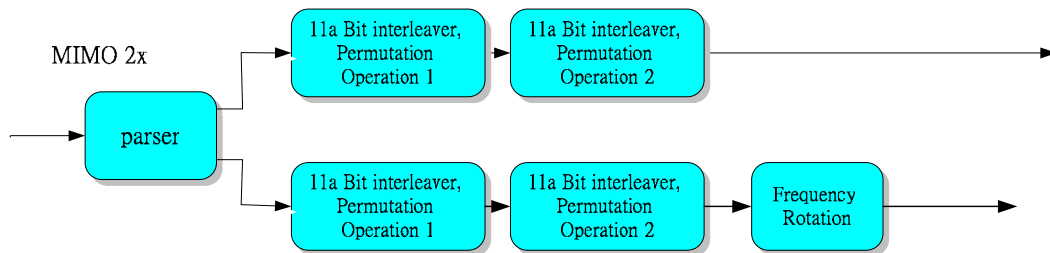


圖 2-8 兩根天線下，11n 交錯器形式

2.3.2 空間資料流對應天線轉換 (Antenna Map Transformation)

在 MIMO 當中，空間流的個數(N_{ss})不見得要與傳送天線的個數(N_{Tx})一樣，而是遵循著 $N_{ss} \leq \min\{N_{Tx}, N_{Rx}\}$ (其中 N_{Rx} 為接收天線的個數)。這個目的是為

了增加整個系統的彈性，例如當 $N_{SS} = 2$ 時，如果硬體設備有 3 根傳送天線的話，可以多利用一根天線來增加 transmit diversity，或是用更多的接收天線來做 receive diversity，這種技術就叫做 Spatial Spreading，如圖 2-9 所示。

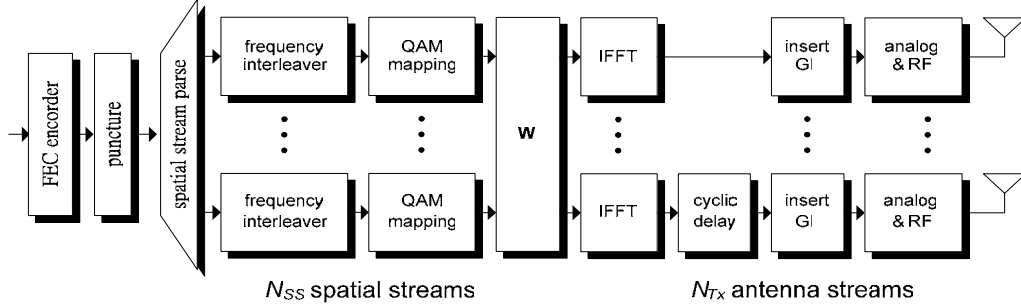


圖 2-9 Time domain implementation of spatial spreading cyclic delay

圖 2-9 中的 \mathbf{w} 是從正交矩陣(unitary matrix) \mathbf{W} (大小為 $N_{Tx} \times N_{Tx}$)取出 N_{SS} 行出來，而通常在 $N_{Tx} = 2$ 與 $N_{Tx} = 4$ 使用 Walsh matrix 如下式

$$\mathbf{W}_{2 \times 2} = \frac{1}{\sqrt{2}} \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix} \text{ 和 } \mathbf{W}_{4 \times 4} = \frac{1}{2} \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix} \quad (2-2)$$

而在 $N_{Tx} = 3$ 時使用 Fourier matrix 如下

$$\mathbf{W}_{3 \times 3} = \frac{1}{\sqrt{3}} \begin{bmatrix} +1 & +1 & +1 \\ +1 & e^{j2\pi/3} & e^{-j2\pi/3} \\ +1 & e^{-j2\pi/3} & e^{j2\pi/3} \end{bmatrix} \quad (2-3)$$

如果只單純的經過 \mathbf{w} 的話，會產生不可預期的 beamforming 效應，這是因為若不同的 spatial streams 之間的相關性太高的話，容易形成一個有方向性的 beam，而使得別的方向的接收機無法接收到訊號。所以我們可以在時域上天線與天線之間，做循環延遲(Cyclic Delay Diversity, CDD)，便可以解決這個問題(圖 2-9)。我們也可以把 CDD 放到頻域上來做並與 \mathbf{w} 做結合為 \mathbf{Q} ，如(2-4)。

$$\begin{cases} \mathbf{Q} = \Phi \cdot [\mathbf{W}_{N_{Tx} \times N_{Tx}}]_{N_{SS}} \\ \text{where } \Phi = \text{diag}(1, \exp(-j2\pi k \Delta_F D), \dots, \exp(-j2\pi(N_{Tx} - 1)k \Delta_F D)) \end{cases} \quad (2-4)$$

另外有時 \mathbf{Q} 也會拿來做 Transmit Beamforming, 即讓 \mathbf{Q} 等於估測到的通道(\mathbf{H})做 SVD(singular value decomposition)後的 \mathbf{V} 矩陣如下。

$$\mathbf{H}_{N_r \times N_t} = \mathbf{U}_{N_r \times N_r} \mathbf{D}_{N_r \times N_t} \mathbf{V}_{N_t \times N_t}^H \Rightarrow \mathbf{Q} = \mathbf{V}_{N_t \times N_t} \quad (2-5)$$

$\mathbf{D} = \text{diag}\{\sigma_0, \sigma_1 \dots \sigma_{N-1}\}$ 為以奇異數組成的對角矩陣, 而 \mathbf{U} 與 \mathbf{V} 皆為正交矩陣。做 SVD 分解的目的, 是可讓通道間不互相干擾, 而變成互相平行的傳輸, 這麼一來, 我們可以視不同通道的 SNR 給予不同的資料, 或是依通道狀況的不同給予不同的傳輸速率, 例如在[2]中的 Extended MCS 裡, 就會在不同的空間流(spatial stream)上, 給予不同大小的 QAM 調變。我們將傳送訊號 \mathbf{x} 乘上 \mathbf{V} 矩陣之後, 在接收端收到的訊號如下：

$$\begin{aligned} \mathbf{y}_{N_r \times 1} &= \mathbf{H}_{N_r \times N_t} \mathbf{V}_{N_t \times N_t} \mathbf{x}_{N_t \times 1} + \mathbf{v}_{N_r \times 1} \\ &= \mathbf{U}_{N_r \times N_r} \mathbf{D}_{N_r \times N_t} \mathbf{V}_{N_t \times N_t}^H \mathbf{V}_{N_t \times N_t} \mathbf{x}_{N_t \times 1} + \mathbf{v}_{N_r \times 1} \\ &= \mathbf{U}_{N_r \times N_r} \mathbf{D}_{N_r \times N_t} \mathbf{x}_{N_t \times 1} + \mathbf{v}_{N_r \times 1} \end{aligned} \quad (2-6)$$

其中的 \mathbf{v} 為雜訊。接著再乘上 \mathbf{U}^H , 並將 \mathbf{x} 解回來, 其式子如下：

$$\mathbf{U}_{N_r \times N_r}^H \mathbf{y}_{N_r \times 1} = \mathbf{D}_{N_r \times N_t} \mathbf{x}_{N_t \times 1} + \mathbf{U}_{N_r \times N_r}^H \mathbf{v}_{N_r \times 1} \quad (2-7)$$

統整以上 spatial spreading 的方法, 我們可以用一個簡單的 \mathbf{Q} 矩陣來當作廣義的 spatial spreading, 如圖 2-10。

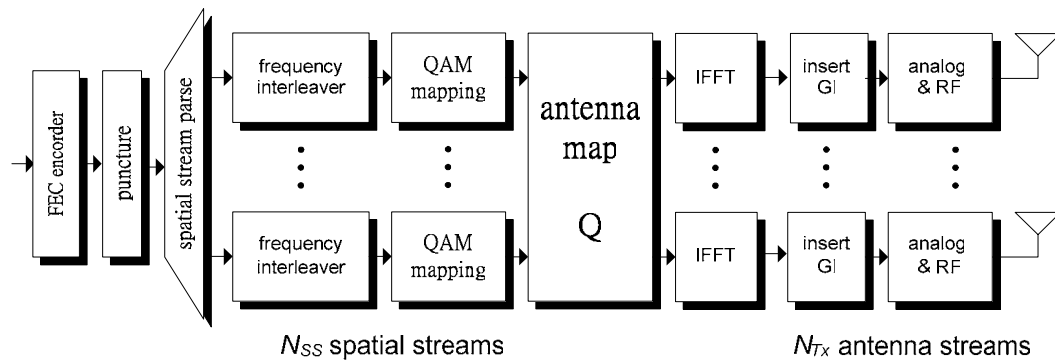


圖 2-10 Spatial Spreading via \mathbf{Q}

第3章系統同步與檢測演算法

正交分頻多工(Orthogonal Frequency Division Multiplexing)技術解決了在多重路徑通道(Multipath Channel)中，因通道產生的延遲所造成的困擾，但在系統同步方面，卻衍生出一些新的問題。除了封包偵測之外，免除內部符元干擾(ISI-free)的偵測，可以確保我們將正確而不受其他符元干擾的取樣值做快速傅利葉轉換(FFT)。而為了保持載波之間的正交性，達到無載波干擾(ICI-free)，傳送與接收機之間的頻率不匹配，須得設法估計與補償。

MIMO(Multi-Input, Multi-Output)系統讓充分地利用空間上的特性,以提高傳輸速度與頻譜使用率。不可避免的，在接收端的天線上，會有著不同傳送天線間的互相干擾，如何將這些干擾分離開來，是檢測演算法中的一個重要課題。本章主要是介紹一些常用的演算法，來解決同步與檢測的問題。

3.1 的封包偵測與 3.2 的頻率偏移估計在最後都會針對這兩個主題做簡單的模擬說明，而 3.3 的通道估計與 3.4 的最小均方誤差檢測則與整個系統效能有顯著的影響，我們將在 3.5 針對 3.3 與 3.4 做整體的模擬與說明。

3.1 封包偵測(Packet Detection)

由於無線網路技術的進步，硬體的複雜度也隨之提高，功率的節省也就越來越重要了。而封包偵測，不但可以初步的估測前導訊號的起始位置，也可以在還沒接收到封包時，讓之後的硬體處於休止的狀態，以節省功率。與封包偵測有關的演算法相當多，但在考慮硬體實現的複雜度與面積的前提之下，在此我們僅介紹兩種常用的演算法。

3.1.1 Double Sliding Window Detection (DSWD)

這是在[3]中所提出的演算法。此種演算法的是建立在訊號雜訊比(SNR)夠大的假設之下，不需要借助於前導訊號特別的訊號安排，也能夠執行封包偵測。其原理很簡單，如下圖 3-1 所示，A 與 B 為兩個緊連的窗戶(windows)，這兩個窗

戶分別計算通過訊號所具有的能量，我們分別以 a_n 及 b_n 表示之，並令 m_n 為一個決定變數(decision variable)。其數學式子如下所示。

$$\begin{cases} a_n = \sum_{m=0}^{M-1} r_{n-m} r_{n-m}^* = \sum_{m=0}^{M-1} |r_{n-m}|^2 \\ b_n = \sum_{l=0}^{L-1} r_{n+l} r_{n+l}^* = \sum_{l=0}^{L-1} |r_{n+l}|^2 \\ m_n = a_n / b_n \end{cases} \quad (3-1)$$

我們先假設 A 與 B 的窗戶大小相同，雜訊能量(noise power)固定，即訊號雜訊比(SNR)夠大。在還沒有封包進來時，A 與 B 皆充斥著 noise 的能量，又因為雜訊的能量是固定的，所以 a_n 及 b_n 的大小會差不多，即 m_n 在這段其間內，會維持一定的大小(約等於 1)。當封包慢慢進入 A 窗戶後， a_n 的能量慢慢增加， b_n 則維持不變，即 m_n 會慢慢的上升，而在封包抵達 A 與 B 的交界處時， m_n 會達到最大。當封包進入 B 之後， b_n 的能量也會隨之提升， m_n 則慢慢的下降，到最後封包完全充斥著 A 與 B 時， m_n 又會回到一個定值(約等於 1)。

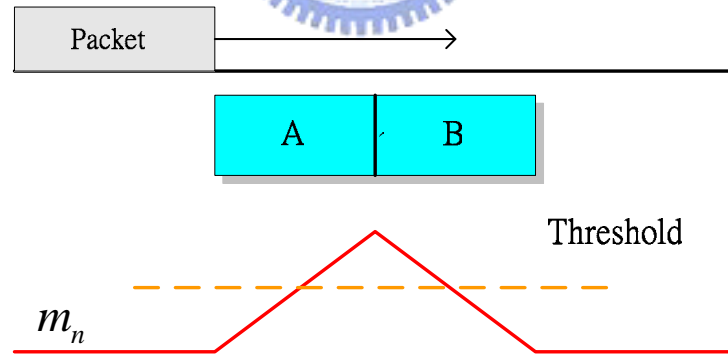


圖 3-1 Double Sliding Window

由 m_n 的曲線看來，在封包進來後，會出現一個最大值(maximum)，但是用最大值來判定訊號是否進來是困難的，因為我們永遠不知道下一刻是否會出現一個更大的值。所以一般的作法，都會採用定立一個門檻(Threshold)，當 m_n 超過門檻值的時候，我們就判定是否收到封包，此時我們通常會定義假警報(false alarm)與遺失封包(miss)兩種機率來判定這個演算法的好壞。

3.1.2 Delay, Correlated and Normalized Detection (DCND)

在 TGn Sync 所提的 802.11n proposal([1])所定義的 Legacy Short Training Field(L-STF)的時域上，是由 10 個大小同為 16 的相同區塊所組成，我們可以將這個重要的特性用在封包偵測上。我們將介紹兩種觀念類似，且各有優缺點的方法。

最直覺的想法，是設計兩個大小同為 16(定義為 D)且相連的窗戶，當訊號通過時，兩個窗戶就互相做相似運算(correlation)，其數學式子如下(3-2)。在封包尚未進來之前， m_n 只會是一個很小的值，而當封包一進到窗戶之中，藉由前導訊號所存在的相關性， m_n 也會隨著變大，當大於我們所設的門檻值時，就判定已接收到封包。這種方法簡單且易實現，不過會有 m_n 的分布範圍過大的問題，也就是說門檻值的設定會更加的困難。

$$\begin{cases} m_n = c_n = \sum_{k=0}^{D-1} r_{n+k} r_{n+k+D}^* \\ D = 16 \end{cases} \quad (3-2)$$

為了解決門檻值的設定變動過大的問題，在[4]中所提出改善的方法，就是對之前求到的 c_n 做正規化(normalization)。如(3-3)，將 c_n 除上窗戶上的總能量，再乘上四倍後，我們嚴格的控制 m_n 在 0 與 1 之間，在設定門檻值時就方便多了。不過在硬體方面，卻得付出多出除法器的代價才行。

$$\begin{cases} c_n = \sum_{k=0}^{D-1} r_{n+k} r_{n+k+D}^* \\ p_n = \sum_{k=0}^{2D-1} r_{n+k} r_{n+k}^* = \sum_{k=0}^{2D-1} |r_{n+k}|^2 \\ m_n = \frac{4|c_n|^2}{p_n^2} \end{cases} \quad (3-3)$$

3.1.3 模擬結果與討論

由以上介紹的兩個方法，我們可以了解到，方法的本身並不困難，而如何定訂一個好的門檻值，顯得更重要一些。

影響門檻值的決定，是在於要讓假警報(false alarm)與封包遺失(miss)的機率下降，但我們知道，這兩種機率值是相衝突的。當門檻值過高時，要被判定會封包的難度增加，假警報機率下降，但封包遺失的機會跟著上升。當門檻值過低時，判定為封包的難度降低，封包遺失的機會下降，但假警報的機率則上升。如何取得一個折衷平衡點(trade off)，在後面將做更深入的探討。

為了比較這三種方法，我們先定義一個共同的環境。我們採用[1]所定義的前導訊號，為配合 L-LSF 前導訊號的重複周期為 16，我們固定 DSWD 的 A 與 B 窗戶大小為 16，DCND 的 D 皆等於 16，通道則設定為五根多重路徑(rayleigh fading)，總能量(σ_H^2)為 1，在接收端用上述介紹的兩種方法，分別計算其 P_f (false alarm)及 P_m 的(miss)機率。

(1) DSWD:首先，我們先探討功率雜訊比的不同是否對 DSWD 有所影響。圖 3-2

為 P_f 和 P_m 與門檻值及 SNR 的模擬結果，由圖中可知， P_f 並不會因為訊號雜訊比(SNR)的不同而有明顯的改變，而 P_m 卻會隨著 SNR 的增加有往下掉的趨勢，這道理很簡單，在討論假警報的情況下，並沒有收到封包，即 A 與 B 窗戶裡面都是雜訊，而 A 與 B 的能量並不會因為雜訊的變大變小而已變。但在討論封包遺失時，當 SNR 一上升，A 與 B 的能量也會跟著上升，這時即使門檻值很高，封包遺失的機會也不大。所以我們看到當 SNR=30 時， P_m 幾乎貼在 0 的位置。

由圖 3-2 可看出，在 SNR = 5 時，我們可以決定出一個最佳的門檻值 2.8，這時的 P_m 與 P_f 約為 24%，但在 SNR = 10 時，最佳門檻值為 4.5，這個差距是很大的，也造成我們設定門檻值時的困擾，不過如果我們限定封包在接收時一定要有 SNR=10 以上的品質時，一

方面這樣的訊雜比下，可以達到一定的效能(錯誤率)，一方面也方便我們設定門檻值。如果在這個假設之下，由圖我們可以看出當 $\text{SNR} > 10\text{dB}$ 的時候，建議的門檻值可設定在 4~5 之間，約能夠達到 P_m 與 P_f 小於 2%。

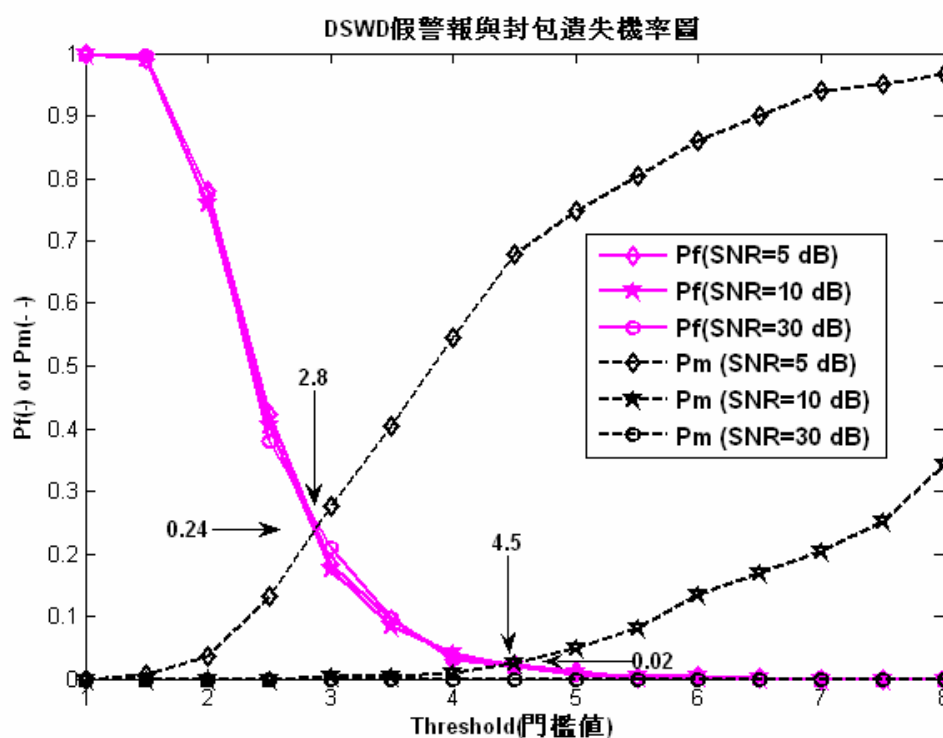


圖 3-2 DSWD 中，不同 SNR 之下比較 P_f 與 P_m

- (2) DCND：圖 3-3 則為 DCND 中 P_f 和 P_m 與門檻值及 SNR 的模擬結果。由圖可看出，與 DSWD 相同的地方，是隨著 SNR 的提高， P_f 的曲線並不會因此而有所改變，但是 P_m 的曲線會逐漸往後退。由圖中， $\text{SNR} = 5$ 時，可找到一個最佳門檻值等於 0.42，這時的 P_f 與 P_m 約為 12%，而當 $\text{SNR} = 10$ 時，最佳門檻值等於 0.58。比照在 DSWD 中的方法，我們只討論 $\text{SNR} > 10$ 的狀況時，大約可以訂出一個門檻值約介於 0.5 到 0.6 之間，而這時的 P_m 或 P_f 可以壓在 1% 以下甚至更小。

由以上的這兩種不同的方法，看出 DCND 在效能方面比 DSWD 好上一倍左右(0.24→0.12, 0.02→0.01)，在門檻值的設定上面也更加簡單且範圍更小(0.6 至

0.8 之間)，所以在封包偵測方面，建議採用 DCND 的方法，之後我們也會以這個方法來做硬體實現。

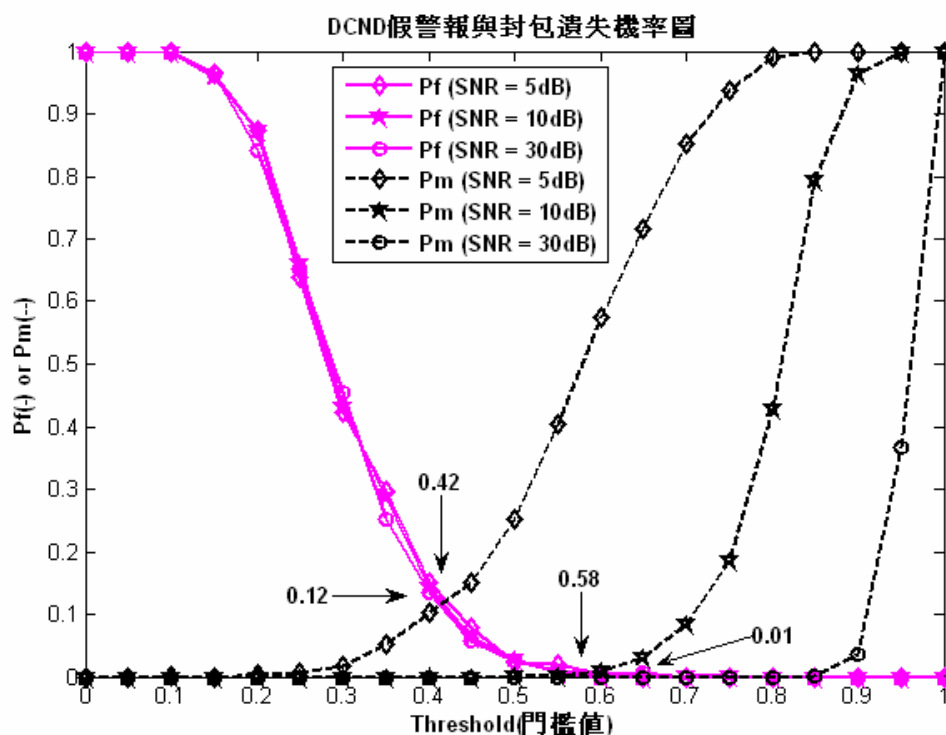


圖 3-3 DCND 中，不同 SNR 之下比較 P_f 與 P_m

3.2 頻率偏移估計(Frequency Offset Estimation)

正交分頻多工技術是採用多個重疊的子載波來傳輸訊號，在碼框已經同步的情形下，各個子載波是彼此正交的。但如果在傳送端與接收端的振盪器(Crystal Oscillator)不匹配時，若不進行頻率偏移補償，將使子載波間的正交性被破壞掉，因而造成子載波間的相互干擾(Inter-carrier Interference, ICI)，不僅信號的能量會下降，相鄰的子載波也會對本身的載波造成干擾，大大的影響了整體的效能。

在[3][5][6]中，提到頻率偏移估算的方法，基本上這些方法的原理，都是利用相隔為 D 的間距上，在複數平面上同值同相位的兩個點，當頻率偏移時，該兩點會因為時間的差異而產一個相位差。於是由已知的間距 D ，可進一步的估算傳送與接收端的頻率偏移，這個方法稱為最可能頻率偏移估算(Maximum

Likelihood Frequency Offset Estimation)方法。

在[3]中，以「最大可能頻率偏移估算」的方法為基礎，提出分別在時域與頻域上的方法，而頻域上的方法主要是用在長訓練符元上，因為長訓練符元能在頻域上估計通道，同時也用來做頻率偏移的補償。而本論文所討論的 802.11n 的通道估計可在之後的前導訊號 HT-LTF 完成，故在此只討論時域上的方法。

3.2.1 Time Domain Approach for Frequency Synchronization

在時域上補償頻率偏移比起在頻率上更為直覺，對 OFDM 系統而言，能在短訓練符元就做能快速(不需經過 FFT)的做好粗略的頻率補償是件好事。

令傳送的時域訊號為 s_n ，載到載波上之後的複數基頻訊號 y_n 為

$$y_n = s_n e^{j2\pi f_{tx} n T_s} \quad (3-4)$$

f_{tx} 為傳送端的載波頻率， T_s 為取樣頻率，而接收端則以一個載波頻率為 f_{rx} 做為降到基頻的頻率，在忽略雜訊的情況下，則解調訊號為

$$\begin{aligned} r_n &= s_n e^{j2\pi f_{tx} n T_s} \cdot e^{-j2\pi f_{rx} n T_s} \\ &= s_n e^{j2\pi (f_{tx} - f_{rx}) n T_s} \\ &= s_n e^{j2\pi \Delta f n T_s} \end{aligned} \quad (3-5)$$

其中 $\Delta f = f_{tx} - f_{rx}$ 為傳送與接收端載波頻率的差值。令 D 為兩個重複區塊間兩點相同取樣值的延遲，則令估計子(estimator) z 為

$$\begin{aligned} z &= \sum_{n=0}^{L-1} r_n r_{n+D}^* \\ &= \sum_{n=0}^{L-1} s_n e^{j2\pi \Delta f n T_s} (s_{n+D} e^{j2\pi \Delta f (n+D) T_s})^* \\ &= e^{-j2\pi \Delta f D T_s} \sum_{n=0}^{L-1} |s_n|^2 \end{aligned} \quad (3-6)$$

我們由上式可發現 Δf 與 z 的相位成正比關係，所以頻率偏移估計子如下

$$\hat{\Delta f} = -\frac{1}{2\pi DT_s} \angle z \quad (3-7)$$

由於 z 的角度 $-2\pi\Delta fDT_s$ 必需明確的定義在 $[-\pi, \pi)$ 之間，所以頻率偏移所能估測的範圍為

$$|\Delta f| \leq \frac{\pi}{2\pi DT_s} = \frac{1}{2DT_s} \quad (3-8)$$

在 WLAN 中，定義了載波頻率偏移量為載波頻率的 20ppm，在 11n 中的載波頻率為 2.4GHz，5GHz，與 4.9GHz，以 5GHz 為例，能容許的最大頻率偏移量為 $\Delta f = 40 \cdot 10^{-6} \cdot 5 \cdot 10^9 = 200kHz$ 。而以短訓練符元(L-STF， $D=16$)所能估測的最大頻率為

$$\Delta f_{\max} = \frac{1}{2 \cdot 16 \cdot 50 \cdot 10^{-9}} = 625kHz \quad (3-9)$$

而長訓練符元(L-LTF， $D=64$)能估計的最大頻率為

$$\Delta f_{\max} = \frac{1}{2 \cdot 64 \cdot 50 \cdot 10^{-9}} = 156.25kHz \quad (3-10)$$

也就是說單靠長訓練符元來估測頻率偏移並不一定可靠，還要搭配上短訓練符元的粗估來修正才行。

頻率偏移補償在 MIMO 上的作法並沒有太大的改變，只是在每根天線上做與 (3-6) 相同的事之後，再做相加的動作。其公式如下。

$$\begin{aligned} z &= \sum_{q=1}^{N_r} \sum_{n=0}^{L-1} r_{q,n} r_{q,n+D}^* \\ &= \sum_{q=1}^{N_r} \sum_{n=0}^{L-1} s_{q,n} e^{j2\pi\Delta f n T_s} (s_{q,n+D} e^{j2\pi\Delta f (n+D) T_s})^* \\ &= e^{-j2\pi\Delta f D T_s} \sum_{q=1}^{N_r} \sum_{n=0}^{L-1} |s_{q,n}|^2 \end{aligned} \quad (3-11)$$

Alert van Zelst 在[7]中說明了天線根數和重複取樣值延遲大小(D)與效能的關係。令頻率偏移估計子的正規化變異數為 K 如下

$$K = \text{var}\left(\frac{\hat{\Delta f} - \Delta f}{\Delta F}\right) = E\left(\frac{\hat{\Delta f} - \Delta f}{\Delta F}\right)^2 = \frac{N_{FFT}}{(2\pi)^2 N_r D^3 \rho} \quad (3-12)$$

此公式在高 SNR 時有較好的準確性。其中 $\hat{\Delta f}$ 代表估計的頻率偏移量， Δf 代表真正的頻率偏移量， ΔF 則代表載波間的距離(11n 中 20MHz 下為 20MHz/64)， ρ 是每根接收天線的訊號雜訊比， N_{FFT} 為 subcarrier 的數目(11n 中 20MHz 下為 64) 而 N_r 則為接收天線數。由這個公式，我們可以了解到效能與接收天線和重複取樣值延遲大小(D)的三次方成正比的關係，但接收天線越多，重複區塊越長，效能就越好。

3.2.2 模擬結果與討論

在此我們以 802.11n 所給的前導訊號來做為送出的訊號，傳送天線數 N_t ，接收天線數 N_r ，通道為五根多重路徑，若傳送天線數大於一根時，則所有傳送天線傳送相同的前導訊號。 $D=16$ 時，我們送出短訓練符元(L-STF)， $D=64$ 時，我們送出長訓練符元(L-LTF)。圖 3-4 至圖 3-8 繪出了 K 在不同 SNR 之下效能，其中 Δf 代表頻率偏移量，而 K 在 (3-12) 已經定義過，我們稱他為 MSE(Mean Square Error) of normalized frequency offset。我們以此來定義效能的好壞，當 K 越小，表示與真正的頻率偏移越接近，效能就越好。

在 (3-12) 中，效能(與 K 成反比)與 $N_r \cdot D^3$ 成正比。圖 3-4 繪出在 1x1 與 2x2 根傳送與接收天線時，為別在 $D=16$ 與 $D=64$ 時的情況，兩個實線(1x1， $D=16$ 與 $D=64$)相差了約 18dB($10\log_{10}(4^3)=18.0618\text{dB}$)，兩個虛線亦然。而在較上方的實線與虛線(1x1， $D=16$ 與 2x2， $D=64$)則差了約 3dB($10\log_{10}(2)=3\text{dB}$)，這與 (3-12) 的公式符合。

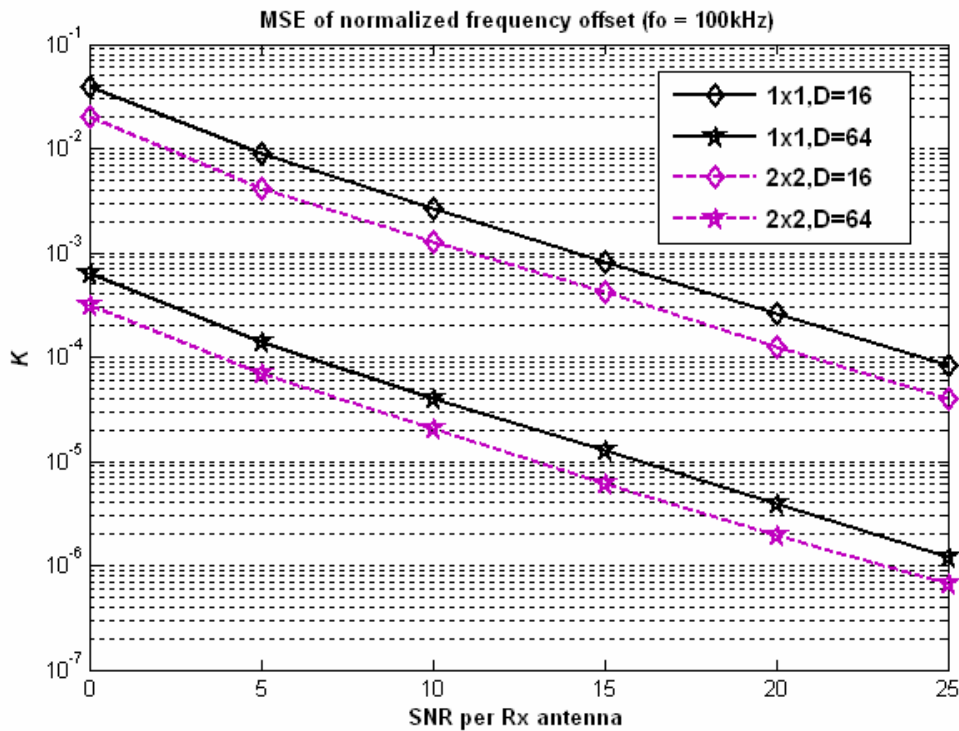


圖 3-4 MSE of normalized frequency offset for 1x1 and 2x2 ($f_o = 100\text{kHz}$)

這個現象在圖 3-5 中看得更清楚，它們分別在 1x1、1x2、1x3 與 1x4 之下的理論值(由 (3-12) 而來)與模擬值，可以看出效能與接收天線數成正比的關係。注意 (3-12) 與傳送天線的多寡並沒有關係，而我們模擬的結果，如圖 3-6 所示，分別是 1x2、2x2、3x2 與 4x2 根傳送接收天線，明顯的效能幾乎一樣。這是因為，不管有幾個傳送天線，一般說來，我們都會讓全部的傳輸功率為 1，所以傳送天線個數並不會反應在效能之上。

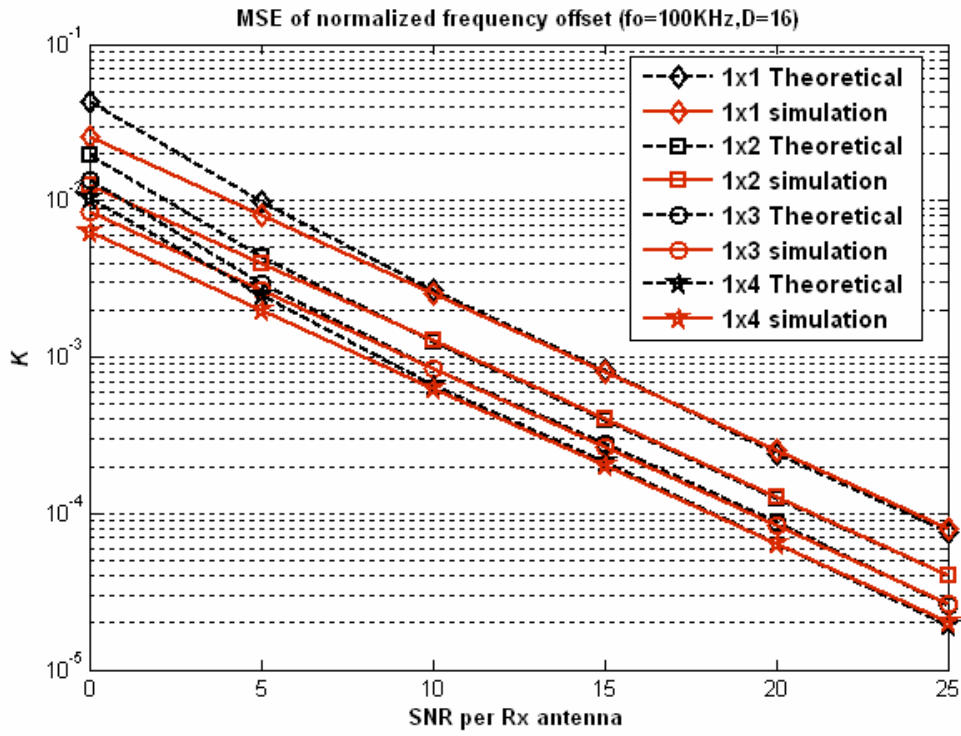


圖 3-5 MSE of normalized frequency offset ($f_o = 100\text{kHz}$, $D=16$)

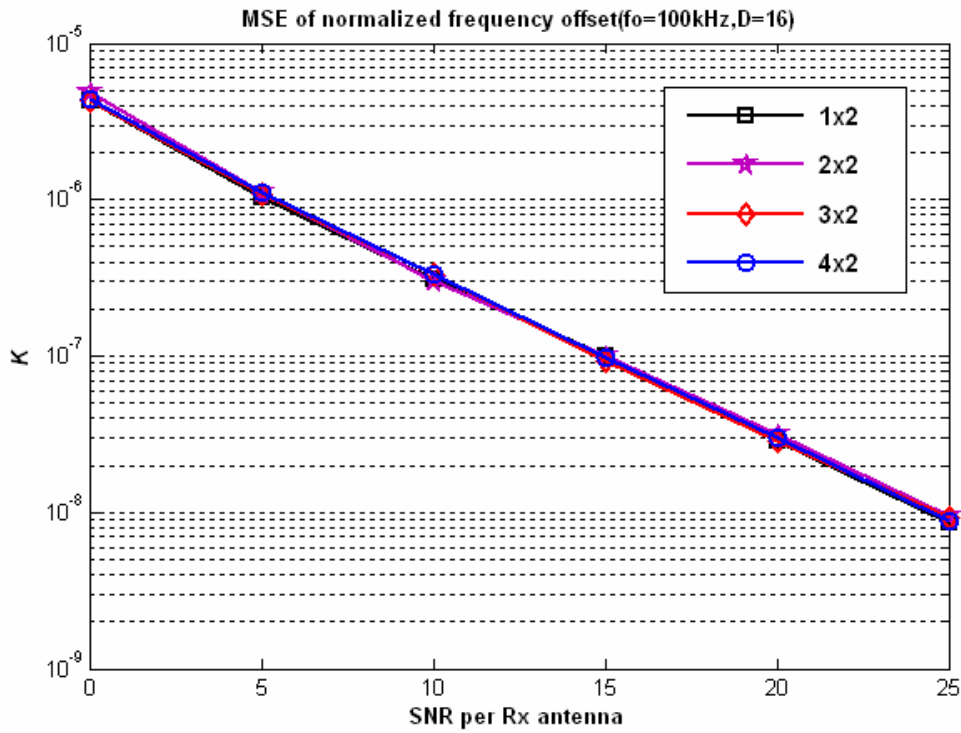


圖 3-6 MSE of normalized frequency offset ($f_o = 100\text{kHz}$, $D=16$)

在(3-9)中，我們提到利用短訓練符元所能估計頻率偏移的最大範圍為625kHz，在圖 3-7 中，當 $f_o = 800\text{kHz}$ 時，可以看到 K 值高達 16;而在 $f_o = 630\text{kHz}$ 時，理論上是完全估不準的，但是在低 SNR 的情況下，有可能會因為雜訊過大的影響，讓某些取樣點的頻率偏移落在 625kHz 以內。而在高 SNR 時，則由估不準的頻率偏移所主導，因而有在低 SNR 比高 SNR 效能還好的可能。當 $f_o = 625\text{kHz}$ 時，是能估準的極限，但因為一點點雜訊的影響，就可能估出的角度由 π 變 $-\pi$ ，而造成頻率偏移估不準。

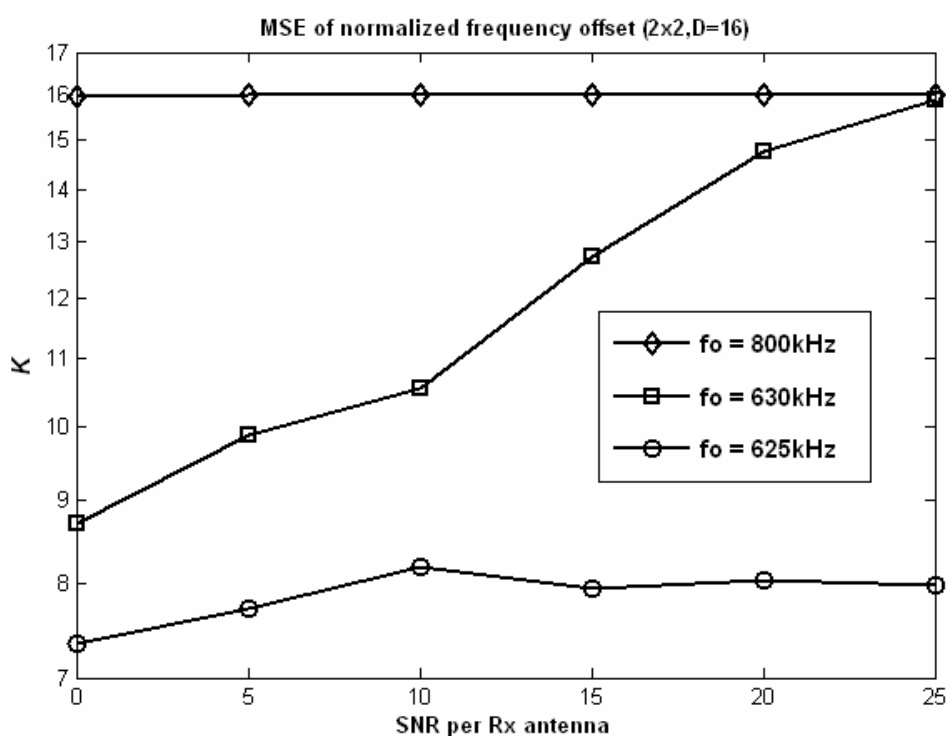


圖 3-7 MSE of normalized frequency offset (2x2, $D=16$)

接著圖 3-8 中， $f_o = 625\text{kHz}$ 的曲線剛剛已經提過。我們再取一個與 625kHz 接近但略小的頻率偏移量 $f_o = 600\text{kHz}$ ，明顯的在低 SNR 與高 SNR 之間會有一個瀑布形狀(waterfall)，因為低 SNR 因為雜訊過大，導致估計角度超出應有的範圍，而在高 SNR 時，效能較不受雜訊過大的影響，而接近當 $f_o = 100\text{kHz}$ (可完全估的準)時的曲線。

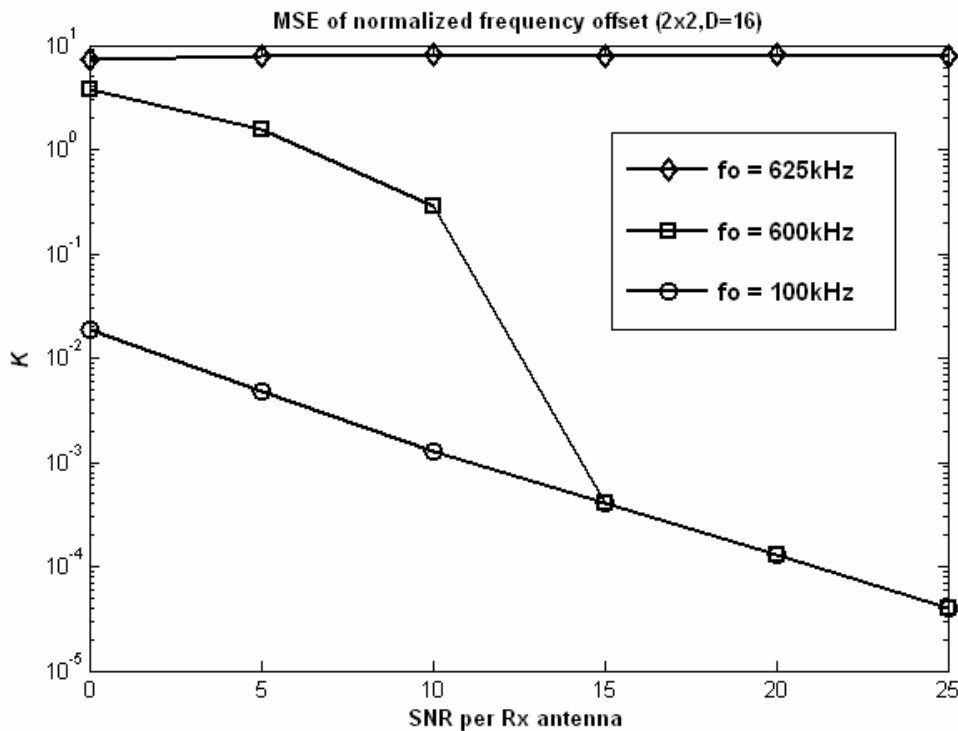


圖 3-8 MSE of normalized frequency offset(2x2 , $D=16$)

3.3 通道估測(Channel Estimation)

在 802.11n 中，TGN Sync 所訂定估測通道的前導訊號，因為利用了 Tone Interleaving 的技術，使得在頻域上通道估測與 802.11a 並沒有什麼兩樣。而為了達到「完全空間與頻率通道估計」(Full Space-Frequency Observability)，通常會因為天線數的增加，如圖 3-9 所示，這也使得前導訊號(preamble)的長度跟著增加，這一直是 TGN Sync 被大家垢病的地方，這也是為什麼，TGN Sync 的提案在五月新的改版中，將 HT-LTL 的長度變小，在 2x2 的情況下，如圖 3-10 中所示，不管幾根天線，都只有第一組的前導訊號有重覆，而之後的都沒有，不像之前的版本中，都會重覆兩個。在這邊只有第一組重覆的原因，主要並不是像舊提案中用來提高估計效能，而是用來做頻率偏移的微調。

對於 OFDM 的通道估計，我們可以假設在一個 OFDM 信號方塊中，或是一個封包中，通道的變化是靜態的。並且可以進一步假設在 WLAN 系統應用中其

通道是緩慢變化通道(slow fading channel)。估測通道的方法有很多種，本論文為了顧及在實作上的複雜度與實現的可能性下，我們選擇在頻域上估測通道。由於前導訊號採用 Tone Interleaving 的方式，故可推導其數學式子如下：

$$R_{l,k} = H_k X_k + W_{l,k} \quad (3-13)$$

$R_{l,k}$ 代表在第 l 根天線，第 k 個 tone 上面收到的訊號， $W_{l,k}$ 則是其上的雜訊，

上式中的 $R_{l,k}$ 只受到一個通道上的一個頻率影響。我們由

$HTL_{-28:28} = [-1, -1, 1, \dots, 1, 1]$ ([2] 中的 sequence 2) 可知道，HTL 上的訊號都為 -1 或是 1，由於 HTL 為已知的訊號，所以只要做下面的動作，就可以估出通道

$$\begin{aligned} \hat{H}_{l,k} &= R_{l,k} X_k^* = (H_k X_k + W_{l,k}) X_k^* \\ &= H_k |X_k|^2 + W_{l,k} X_k^* = H_k \pm W_{l,k} \\ &\text{where } X_k = \pm 1 \end{aligned} \quad (3-14)$$

由上式中我們可以看出前導訊號($HTL_{-28:28}$)設計的概念，其分散的放置 1 與 -1，除了讓 $HTL_{-28:28}$ 的平均功率為 1 之外，也可免去在硬體設計上需要除法器的困擾。

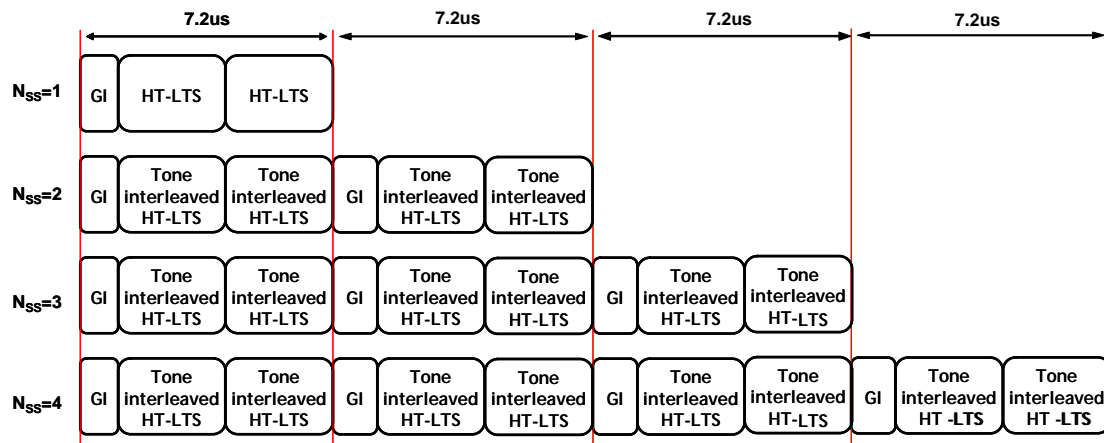


圖 3-9 HT-LTF(TGN Sync 三月版)

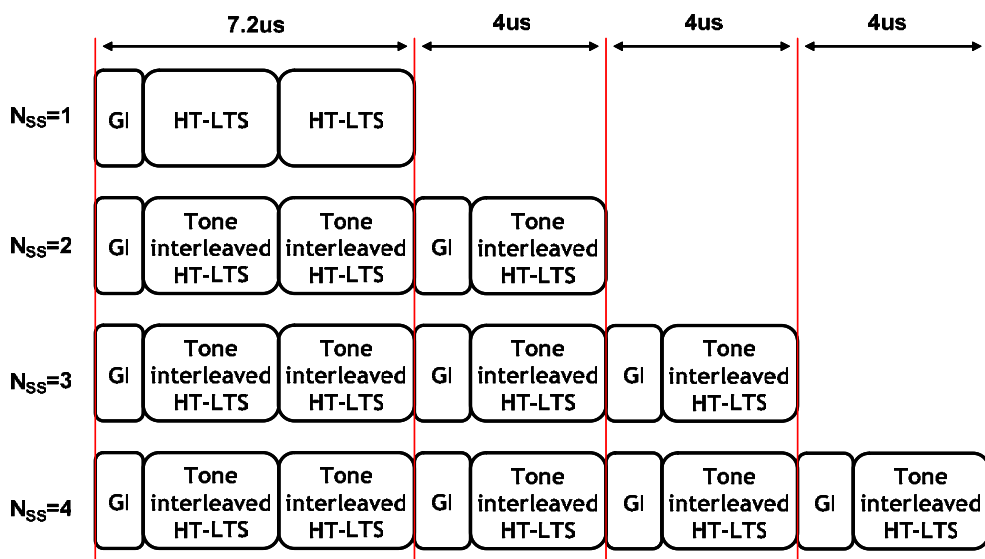


圖 3-10 HT-LTF(TGN Sync 五月版)

在 2×2 的組態之下，估測通道的方式如下圖 3-11 所示，在傳送端上，Set0(偶)代表 Set0 只傳 $HTL_{-28:28}$ 的偶數頻率上的訊號，其他則放 0，其他的以此類推。而在接收端，方塊中的 H11(偶)+H12(奇)表示這個方塊上的偶數頻率上是 H11 通道的偶數頻率加上 H12 的奇數頻率，餘下類推。由此示意圖可知，我們在接收端只要稍微做個重組，就可以還原出所有通道的估計值了。

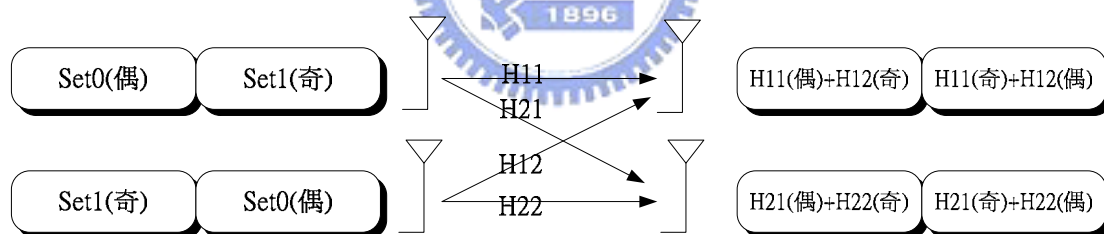


圖 3-11 2×2 下通道估計示意圖

以上所介紹的方法是最直覺也最方便的，但是隨著天線數的增加，通道估計的效能也會跟著變差，這點可以用內插的方法加以改善，我們將在 3.5 的模擬中有所著墨。

3.4 最小均方誤差檢測(MMSE Detection)

在下一代的無線通訊系統的主要目標，是增加連結速率(link throughput)和網路的承載量(network capacity)，而在傳送與接收端放置多根天線是達到此目的的一個很重要的技術，特別是在豐富的散射環境當中(rich scattering

environment)。此多重輸入、多重輸出(multiple-input multiple-output, MIMO)技術主要可以分成兩大類：Space Time Coding (STC)與 Space Division Multiplexing (SDM)。STC 主要是利用傳送端不同的分枝做空間上的編碼技術進而提高接收的效能，而 SDM 主要是藉由傳送互相獨立的資料串(data streams)，搭載在同一個載波，但在不同的傳送天線上，以達到更高的吞吐量(throughput)。在本論文中的主要訴求是增加位元速率，所以我們將重點擺在 SDM 上面。

與 SDM 有關的演算法有很多，像是 Zero Forcing、Maximum Likelihood Estimation、MMSE、V-BLAST 和 D-BLAST 等等，由於本論文的系統還有搭配 Viterbi Decoder 的緣故，所以我們採用 MMSE (Minimum Mean Square Error)濾波器。我們發現，用 MMSE 配上 Viterbi soft-Decoder，可以達到不錯的效能。

假設 MIMO OFDM 通道為頻率選擇通道(Frequency Selective Fading)，並有適當的 CP 與準確的同步之下，在頻域下的訊號模型，可視為平衰落(flat fading)，其式子表示(3-15)



$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{n}$$

(3-15)

若考慮在數位的情況下，觀察取樣點 k ，則可以改寫為 $\mathbf{Y}(k) = \mathbf{H}\mathbf{X}(k) + \mathbf{n}(k)$ ，然後定義一個錯誤向量 $\mathbf{e}(k)$ ，代表傳送訊號與乘上 MMSE 濾波器的接受訊號之間的誤差，定義為：

$$\mathbf{e}(k) = \mathbf{x}(k) - \mathbf{G}^H \mathbf{y}(k)$$

(3-16)

其中 \mathbf{G} 為 MMSE 濾波器。依照 MMSE Criterion，我們要最小均方差 J ，其定義如下：

$$\begin{aligned} J &= E \left\{ \mathbf{e}^H(k) \mathbf{e}(k) \right\} \\ &= tr \left[E \left\{ \mathbf{e}(k) \mathbf{e}^H(k) \right\} \right] \end{aligned}$$

(3-17)

為了使 J 最小，我們對其微分，另其等於 0，可以推導出 Wiener-Hopf equation:

$$\mathbf{G}^H \mathbf{R}_{yy} = \mathbf{R}_{xy} \quad (3-18)$$

其中

$$\mathbf{R}_{yy} = E\{\mathbf{y}(k)\mathbf{y}^H(k)\} \quad \mathbf{R}_{xy} = E\{\mathbf{x}(k)\mathbf{y}^H(k)\} \quad (3-19)$$

\mathbf{R}_{yy} 為接收訊號向量的協方差(Covariance)，而 \mathbf{R}_{xy} 為傳輸和接收向量的交互相關(Cross-Correlation)。由(3-18)，我們可以推導出 MMSE 濾波器如下：

$$\mathbf{G} = [\mathbf{H}\mathbf{H}^H + \alpha \mathbf{I}_{N_r \times N_r}]^{-1} \mathbf{H} \quad (3-20)$$

上式可等效為下式：

$$\mathbf{G} = \mathbf{H} [\mathbf{H}^H \mathbf{H} + \alpha \mathbf{I}_{N_t \times N_t}]^{-1} \quad (3-21)$$

其中 α 為每根接收天線上 SNR 的倒數。在 SDM 之下，傳送天線數必需小於接收天線個數，即 $N_t \leq N_r$ ，由此看來，(3-21)的計算量會小於或等於(3-20)，因此，一般說來，我們較習慣採用(3-21)來做為最小均方差檢測的公式。最後一步，我們解得的訊號為：

$$\hat{\mathbf{x}}(k) = \mathbf{G}^H \cdot \mathbf{y}(k) \quad (3-22)$$

3.5 模擬(Simulation)

最後，我們針對整個大系統，做整個浮點數(floating point)的模擬，包括了封包偵測，粗調頻率偏移補償，細調頻率偏移補償，碼框偵測，通道估計等主要在前導訊號需要完成的部分。在接收端則採用 MMSE Detection 配合通道資訊(CSI, Channel State Information)再加上 Soft-Input Viterbi Decoder，整個傳送與接收端的方塊圖如下圖 3-18。

傳送端採用的規格為 N 串空間流(spatial streams)， N 根傳送天線，其中 N 為 1、2、3、4，頻寬為 20MHz，每個封包的長度為 12 個 OFDM symbol。FEC encoder 採用圖 2-6 的 Convolutional Encoder ($k=7$)，coding rate 為 1/2，即沒有經過打洞器(puncture)。而空間分配(Spatial Parsing)與頻率交錯器(Frequency Interleaver)在 2.3.1 中都有介紹。QAM mapping 部分採用 64QAM(請參考[1])，保護符元(Guard Interval)長為 800ns，而頻率偏移量是在 3.2.1 中所介紹過的最大容忍頻率偏移 200kHz。

通道的部分，我們使用的是多重路衰落通道模型。在無線通信系統中，信號經由多個路徑到達接收端。如下圖 3-12 所示

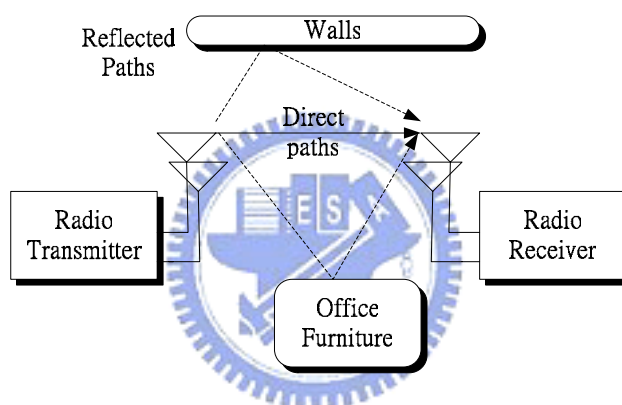


圖 3-12 多重路徑衰落通道示意圖

在無線區域網路的傳送端傳送信號經過不同的反射物到達接收端。接收機在前端的天線除了接收到一個直接到達的路徑之外，還收到經過不同路徑的衰減信號。隨著路徑長短與到達的時間不同，於是在不同的環境下定義用來描述通道，經由不同路徑延遲的方根延遲擴展(Root Mean Square Delay Spread, T_{RMS})。在我們的通道模型中，假設不同天線之間的通道沒有關連性，而每一個通道路徑的數目為 5，平均振幅隨著延遲的增加而成一個指數衰減(Exponential Decay)。其中定義第 k 個路徑為一個複數，而實數及虛數部分是由平均值為 0 且變異數為 $\sigma_k^2/2$ 的高斯隨機變數所產生。也就是任一個路徑上的振幅為 Rayleigh Distribution，相位為 Uniform Distribution 的隨機變數，通道模型中定義第 k 個路徑的數學示如下

$$\begin{aligned}
h_k &= N(0, \frac{1}{2}\sigma_k^2) + j \cdot N(0, \frac{1}{2}\sigma_k^2) \\
\sigma_k^2 &= \sigma_0^2 \cdot \exp(-kT_s / T_{RMS}) \\
\sigma_0^2 &= 1 - \exp(-T_s / T_{RMS})
\end{aligned}
\tag{3-23}$$

其中 T_s 為 50ns， T_{RMS} 在此設為 150ns，而訊號雜訊比(SNR)是定義在每根接收天線上，以一整個封包為單位加上雜訊。下圖為每個通道路徑的平均功率圖。

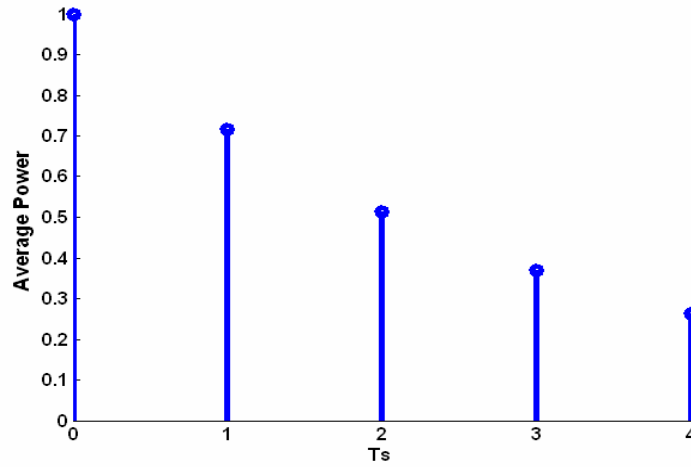


圖 3-13 Average Power of Exponential Decay Multipath Channel

在接收端的部分，圖 3-18 為整個傳送與接收的系統圖，其中 Synchronization，Channel Estimation 與 MMSE Detection 以之前提過的方法加以模擬，而 QAM(soft) Demapping 方法，由於不是本論文所探討的重點，故請參考 [14]。另外我們畫出了圖 3-14 至圖 3-17 四張模擬圖，代表在不同傳送天線與接收天線下，SNR 與 BER 的關係。

首先我們來看看，在頻率偏移完全補償，通道係數已知和時序完美的情況下， $N \times N$ ($N=1,2,3,4$)，所跑出來的效能圖 3-14，此圖代表在本論文模擬的環境之下的最好的情況。我們可以看到，隨著天線數的增加，天線之間的干擾也越來越大，所以效能會跟著下降，這 1×1 (即 SISO 之下)，由於沒有其他天線的干擾，效能比起其他三條線要好上許多。

接著我們在 2×2 與 3×3 ，已知通道係數及時序完全已知的情況下，有頻率

偏移並補償和已知頻率偏移的比較。從圖 3-15 中，可以看出經過補償後的頻率偏移，與已知頻率偏移的情況下，差不到 1dB。

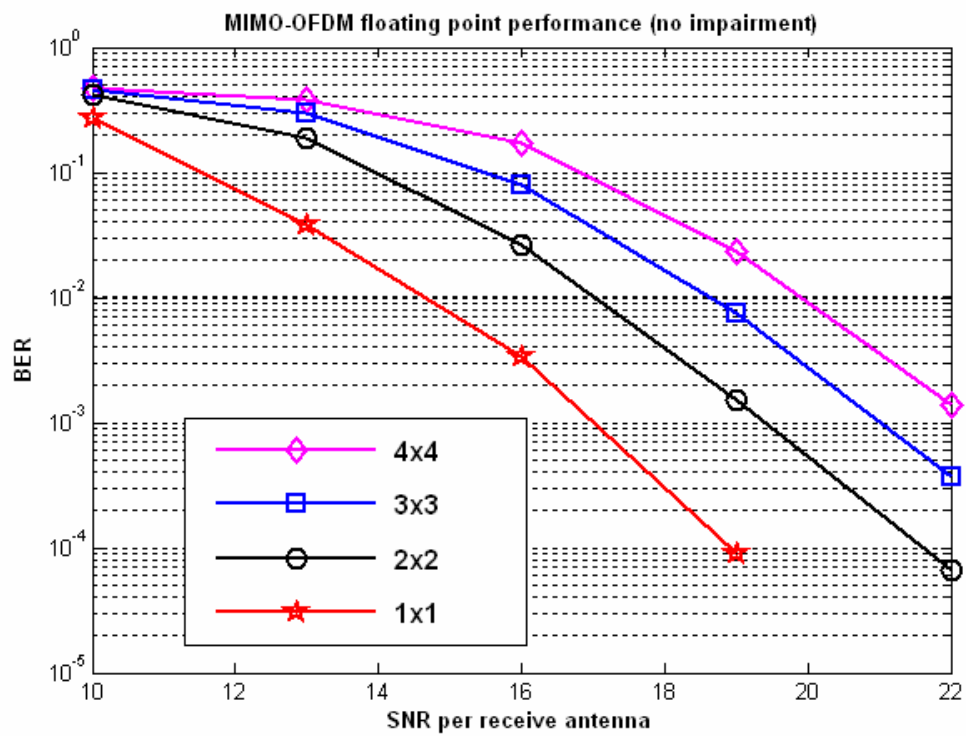


圖 3-14 MIMO-OFDM floating point performance (no impairment)

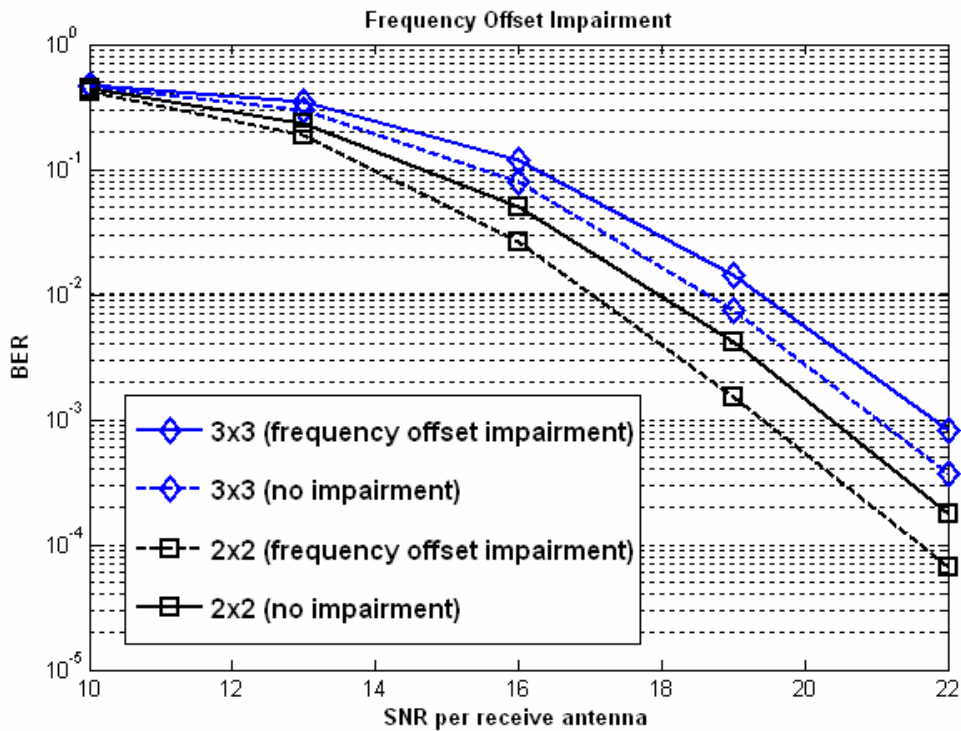


圖 3-15 Frequency offset impairment

在通道估計方面(這裡的通道估計所用的是 TGn Sync 三月提案中的前導訊號，請參考圖 3-9)，如果單純的用(3-14)中所介紹的方法的效能如圖 3-16，在 BER 為 10^{-1} 時，2x2 天線下，已知通道係數與估測通道的差距為 4dB，在 3x3 天線下為 5dB，在 4x4 天線下則為 6dB，我們將正確的通道資訊(CSI)與估出的通道資訊做均方差(MSE)，所算出的數據如表格 3-1。

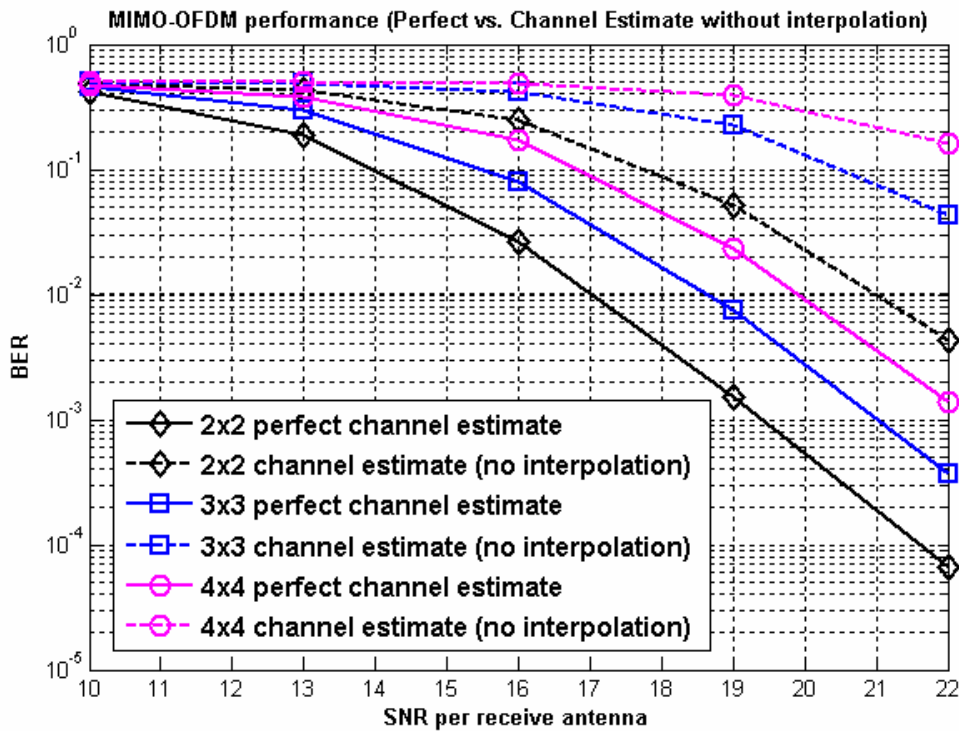


圖 3-16 MIMO-OFDM performance (Perfect vs. Channel Estimate)

SNR		10dB	13dB	16dB	19dB	22dB
$N \times N$						
2x2	C	0.4178	0.2123	0.1068	0.0540	0.0269
	I	0.1964	0.0995	0.0513	0.0268	0.0142
3x3	C	0.6399	0.3239	0.1633	0.0821	0.0410
	I	0.2684	0.1416	0.0751	0.0434	0.0267
4x4	C	0.8706	0.4379	0.2205	0.1097	0.0548
	I	0.4726	0.2673	0.1195	0.0596	0.0299

表格 3-1 MSE of perfect CSI and Estimated CSI (C: Channel Estimate using (3-14) equation。I : Channel Estimate with Interpolation)

其中的 C 代表用(3-14)中介紹的方法估計通道，而 I 則是用內插的方式的方式先內插出每個 HT-HTF 符元裡未傳送的通道，再加以平均當做估到的 CSI。由表格 3-1 看出 2x2，3x3 與 4x4 之間，MSE 與傳送天線數(N_t)成正比，例如 2x2 的 C 列與 3x3 的 C 列就差了 1.5 倍，這是因為在同一個頻率上，雖然沒有別根傳

送天線的資料干擾，但因為傳送天線瓜分了傳送的功率，所以在每個頻率上的訊號雜訊比相對下降了。除了通道估不準造成了效能下降，另外在 MMSE Detection 方面，CSI 的不準度，也會隨著天線數的增加，而讓 MMSE 的效能下降，所以這兩方面都造成效能隨天線數的增加而變差。

為了改善通道的估不準，我們以通道資訊內插(matlab 內建函數)的方式來改善效能，其結果如圖 3-17，同樣在 BER 為 10^{-1} 時，2x2 天線下，已知通道係數與估測通道的差距由 4dB 減少到 2.5dB，3x3 天線下則從 5dB 減少到 3.5dB，4x4 天線由 6dB 減少至 4dB。由表格 3-1 我們也可以看出，經過內插之後(I)，大約比沒內插時好上 2 倍。我們也發現，在 2x2 天線與 3x3 天線下，所選用的 linear 內插與 spline 內插相差無幾，但在 4x4 下則需要用 spline 才能達到較好的效果。

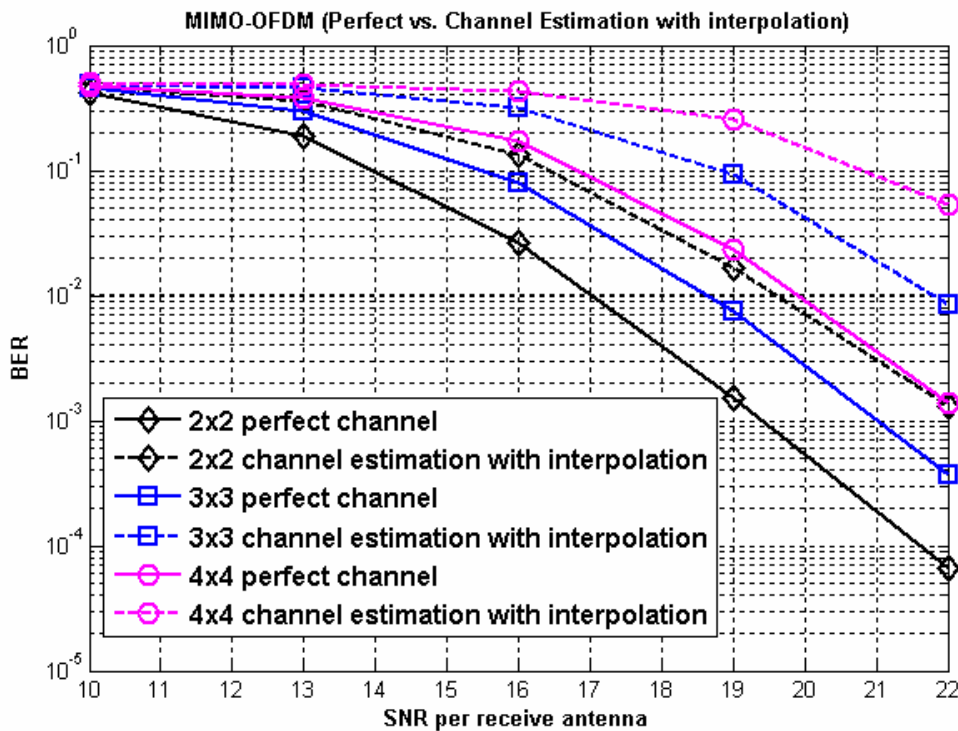


圖 3-17 Perfect vs. Channel Estimation with interpolation

第4章 硬體設計與架構

4.1 設計流程

在硬體設計方面，我們使用電腦輔助工具來做硬體實現，包括 Xilinx ISE 3.0，ModelSim，而其中的合成軟體則使用 Xilinx ISE 中內建的 XST Synthesizer 來做合成。我們針對 TGn Sync 所提的 proposal，在 2x2 根傳送與接收天線，64QAM 之下，模擬封包偵測、頻率偏移估計與補償、自動增益控制和通道估計等等與接收機前導訊號有關的功能方塊。

流程圖如下圖 4-1，我們先以 Matlab 的浮點數運算(floating point)驗證我們的功能方塊與整個系統的效能，同樣再由 Matlab 做定點數運算(fix point)調整各個元件所需有位元數並試著將位元降低至容許的效能。接下來確定每個功能方塊的硬體架構，以硬體描述語言 VERILOG 描述這些功能方塊，並以 Xilinx ISE 內建的 XST 合成器進行合成，轉成 RTL，合成完之後，再做 place and route 的 timing simulation，確認是否達到我們所需要的 clock 速度。

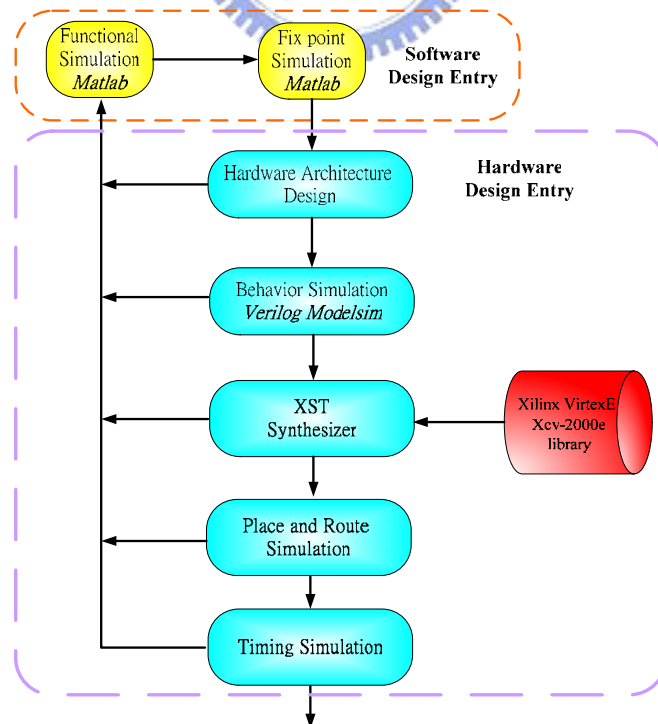


圖 4-1 硬體設計流程圖

經過定點數的模擬，我們發現訊號在接收天線收下來之後，10 位元的 A/D 已經足夠符合我們的需求，之後硬體的實現將以這些訊號做訊號處理，來達到我們想要的效能。

4.2 重要功能區塊 (Functional blocks)

4.2.1 CORDIC(Coordinate Rotation Digital Computer)演算法

✚ CORDIC 介紹

在西元1959 年 Jack E. Volder 提出CORDIC(Coordinate Rotation Digital Computer)演算法是，是用來做一些數值運算，例如三角函數、乘法、除法、二進位到固定基底數值轉換等等。在本篇論文中，主要是拿來計算Arctangent和相位的旋轉。此演算法利用反覆做任意角度的相量旋轉，最後以移位器與加法器來實現。CORDIC演算法的向量旋轉可寫成(4-1)。

$$V' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (4-1)$$

上式為 $[x \ y]^T$ 經旋轉角度 ϕ 之後得到 V' 。圖 4-2為向量旋轉示意圖。

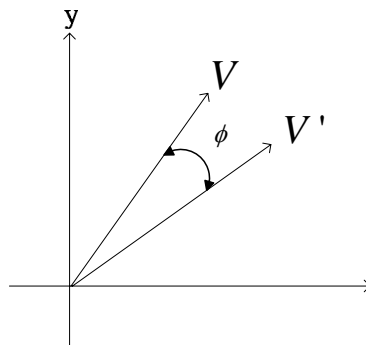


圖 4-2 向量旋轉示意圖

假如我們需要旋轉多次的角度去達成上式的運算，則累加的角度

$$\phi = \sum \phi_i \quad (4-2)$$

接著將(4-1)改寫為

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \prod_i \begin{bmatrix} \cos \phi_i & -\sin \phi_i \\ \sin \phi_i & \cos \phi_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (4-3)$$

把連續角度的旋轉以矩陣相乘表示，接著再將 $\cos \phi_i$ 提出來，改為下式

$$\begin{aligned} \begin{bmatrix} x' \\ y' \end{bmatrix} &= \left\{ \prod_i \cos \phi_i \begin{bmatrix} 1 & -\tan \phi_i \\ \tan \phi_i & 1 \end{bmatrix} \right\} \begin{bmatrix} x \\ y \end{bmatrix} \\ &= \left\{ \prod_i \cos \phi_i \right\} \prod_i \begin{bmatrix} 1 & -\tan \phi_i \\ \tan \phi_i & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \end{aligned} \quad (4-4)$$

再來我們限制 ϕ_i 使得

$$\tan \phi_i = \pm_i 2^{-i} \quad (4-5)$$

一旦用成這種形式，我們就可以輕易的用移位器與加法器來實現。上式中的正負號是為了讓(4-2)中的 ϕ 能夠不斷地做修正與微調。

$$\begin{aligned} \begin{bmatrix} x' \\ y' \end{bmatrix} &= \left\{ \prod_i \cos(\tan^{-1} 2^{-i}) \right\} \prod_i \begin{bmatrix} 1 & \mp_i 2^{-i} \\ \pm_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \\ &= \left\{ \prod_i \frac{1}{\sqrt{1+2^{-2i}}} \right\} \prod_i \begin{bmatrix} 1 & \mp_i 2^{-i} \\ \pm_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \\ &= A_n \cdot \prod_i \begin{bmatrix} 1 & \mp_i 2^{-i} \\ \pm_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \end{aligned} \quad (4-6)$$

上式中最 A_n 在疊代個數固定的時候是一個定值，可以事先算出來，在做後做完疊代之後再乘上即可，當 i 由0開始時，此值在 i 到夠大的時候大約在0.6073時達到收斂。而後面矩陣的連續相乘，也只是做簡單的移位與加法(或減法)而已。我們再將該式改為疊代的形式

$$\begin{cases} x_{i+1} = K_i [x_i - y_i \cdot d_i \cdot 2^{-i}] \\ y_{i+1} = K_i [y_i - x_i \cdot d_i \cdot 2^{-i}] \end{cases}$$

where

$$K_i = \cos(\tan^{-1} 2^{-i}) = \frac{1}{\sqrt{1+2^{-2i}}}$$

$$d_i = \pm 1$$

(4-7)

由上式中，在坐標旋轉的運算裡，為了得到每次旋轉的方向，必需設計一個角度累加器去得到方向序列輸出 d_i ，此角度累加器(Angle Accumulator)數學方程式可表示為下式：

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

(4-8)

在[8]中，CORDIC 演算法的運作，一般說來可以分為兩種模式(mode)，一種叫做 Rotating Mode，另一種則是 Vectoring Mode。這兩種模式在本論文的實現中都會用到。Rotating Mode 是給將輸入向量旋轉一個給定的角度，再 Vectoring Mode 則是將原本輸入的向量，經由不斷的旋轉並轉至 x 軸之上。

➤ Rotating Mode

在頻率估測當中，在已經估出頻率偏移角度的情況下，我們要對之後的每一個取樣值估頻率補償，也就是對取樣值做旋轉。在 Rotating Mode 中，我們先讓角度累加器的初始值為欲旋轉的角度，並在每個疊代當中，試圖讓角度累加器的值為 0。Rotating Mode 的公式如下

$$\begin{cases} x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} = y_i - x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i}) \end{cases}$$

where

$$d_i = -1 \text{ if } z_i < 0, +1 \text{ otherwise}$$

(4-9)

我們便利用此式來做為頻率補償的旋轉。當疊代次數夠多時，上式可變為


$$\begin{cases} x_n = A_n [x_0 \cos z_0 - y_0 \sin z_0] \\ y_n = A_n [y_0 \cos z_0 + x_0 \sin z_0] \\ z_n = 0 \\ A_n = \prod_n \frac{1}{\sqrt{1+2^{-2i}}} \end{cases} \quad (4-10)$$

➤ Vectoring Mode

在 Vectoring Mode 中，CORDIC 試著將輸入訊號旋轉至 x 軸上，我們可以經由這樣的機制，得到原本輸入訊號的角度(phase)與大小(magnitude)。與 Rotating Mode 不一样的地方，在於它會試著讓疊代 y_i 的值為 0，以便讓旋轉後的向量對齊 x 軸，換句話說，我們由 y_i 來決定旋轉的方向。Vectoring Mode 公式如下：

$$\begin{cases} x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i} \\ y_{i+1} = y_i - x_i \cdot d_i \cdot 2^{-i} \\ z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i}) \end{cases}$$

where

$$d_i = +1 \text{ if } y_i < 0, -1 \text{ otherwise}$$

(4-11)

當疊代次數夠多時，上式可變為

$$\begin{cases} x_n = \sqrt{x_0^2 + y_0^2} / A_n \\ y_n = 0 \\ z_n = z_0 + \tan^{-1}(y_0 / x_0) \\ A_n = \prod_n \frac{1}{\sqrt{1+2^{-2i}}} \end{cases} \quad (4-12)$$

明顯的，由上式中，當我們讓 z_0 為 0 時，可以利用 z_n 來算出頻率補償的角度，又因為我們並不需要算出輸入訊號的大小(magnitude)，所以並不需要計算 A_n 。

不管是 Rotating Mode 或是 Vectoring Mode，在每次疊代中，旋轉角度都會有其最大限制，這個限制與 i 的起始位置有關，其最大旋轉角度的限制如下：

$$-\tan^{-1}(2^i) \leq \text{rotating angle per iteration} \leq \tan^{-1}(2^i) \quad (4-13)$$

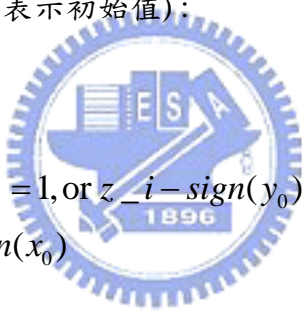
也就是當 i 起始為 0 時，旋轉角度會介於 $-\frac{\pi}{2}$ 與 $\frac{\pi}{2}$ 之間，當總旋轉角度超過最大能旋轉的值時，就會發生錯誤。那最大可旋轉的角度是多少呢？我們假設每次疊代時的 d 都為 1，不斷相加的結果，可旋轉最大的角度為(先假設 i 起始值為 0)

$$\sum_{i=0}^{\infty} \tan^{-1}(2^{-i}) \cong 1.7433 \quad (4-14)$$

也就是能旋轉的總最大角在 -1.7433 與 1.7433 之間，大約在正負 100 度之間。若要超過這個角度的話，解決的辦法先旋轉 $-\pi$ 或 π 將旋轉角度限制在能運作的範圍之內，其公式如下(其中 $_i$ 表示初始值)：

$$\begin{cases} x_0 = d_0 \cdot x_{-i} \\ y_0 = d_0 \cdot y_{-i} \\ z_0 = z_{-i} \text{ if } d_0 = 1, \text{ or } z_{-i} - \text{sign}(y_0) \cdot \pi \text{ if } d_0 = -1 \end{cases}$$

where $d_0 = \text{sign}(x_0)$



(4-15)

其中 $\text{sign}()$ 代表該括弧中的正負號。做完初步的旋轉之後，再把 x_0 、 y_0 、 z_0 與 d_0 代入之前介紹的公式中即可，經過這個處理之後，我們所能得到總旋轉角度範圍為擴大至 $-\pi$ 到 π 之間了。

CORDIC 模擬結果

針對 CORDIC 演算法，我們以 Matlab 模擬了圖 4-3 到圖 4-5 共三張圖，代表著隨著疊代次數(n)的增加， x 、 y 與 z 與理論上當 n 很大時的 x 、 y 與 z 的趨近狀況。首先我們針對 Rotating Mode 來做模擬，並針對論文主題中的頻率補償方面，也就是向量的旋轉做模擬。假設輸入訊號為 $1+i$ ，旋轉角度為 $3\pi/2$ ，那麼輸出訊號應該為 $1-i$ 才對。由圖 4-3 中，我們可以看到， $x(i)$ 與 $y(i)$ 隨著 i 的增加慢慢的逼進 1 與 -1，這邊注意 x 與 y 的起始值 $x(0)$ 、 $y(0)$ ，原本應該都是 1 才對，

但在圖上為-1，這是因為旋轉的角度大於 π (為 $3\pi/2$)，所以我們已經先旋轉了一個 π 角度了，變為 $x(0)=-1$ ， $y(0)=-1$ 。而在 $z(i)$ 方面也逐漸變為0，印證之前在 Rotating Mode 中所說的 $z(n)$ 會趨近於0。

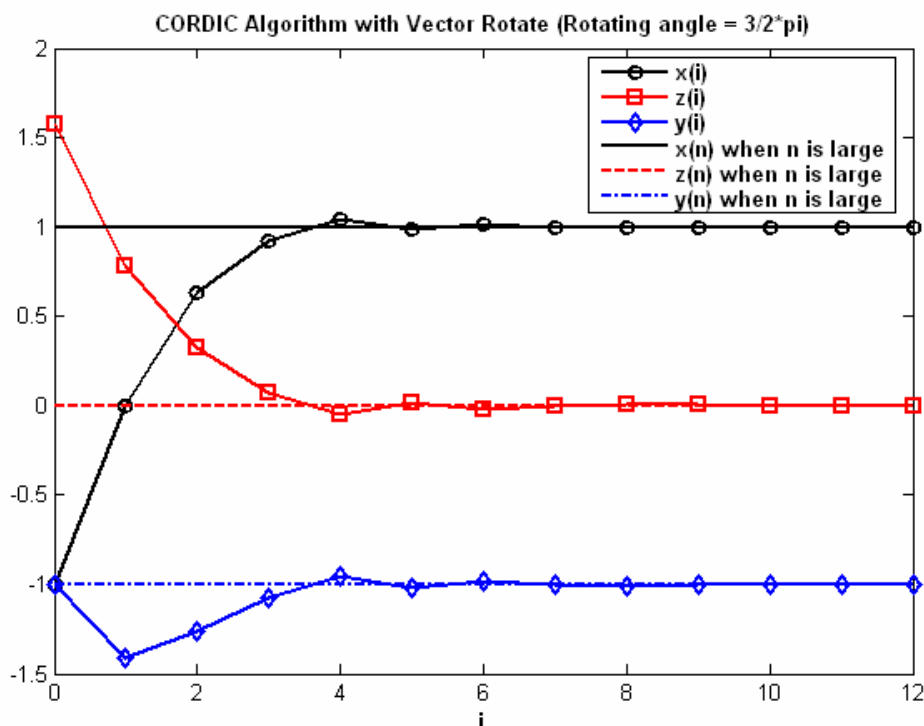


圖 4-3 CORDIC Algorithm with Vector Rotate (Rotating angle = $3\pi/2$)

假若輸入訊號仍為 $1+i$ ，如果旋轉角度為 4.93 (約為 282.468°)，會發生什麼情形呢?我們從圖 4-4 中，可以看到當 i 很大時，不論是 $x(i)$ 、 $y(i)$ 或 $z(i)$ 都無法貼上它們應有的值，但是還蠻接近的，這是因為這個角度稍稍超過了所能旋轉的臨界值。這個臨界值是從可(4-14)中看出，最大能旋轉的角度為 1.7433 ，由於之後我們可以透過預先旋轉角度 π ，所以最大能旋轉的角度為 $\pi + 1.7433 = 4.8849 = 279.884^\circ$ 。不過這個問題不會造成我們的困擾，因為大於這個角度的話，我們可以扣掉 2π ，也就是將範圍限定在 $-\pi$ 與 π 之間即可，或是改變 i 的初始值(如0改為-1或-2)等。

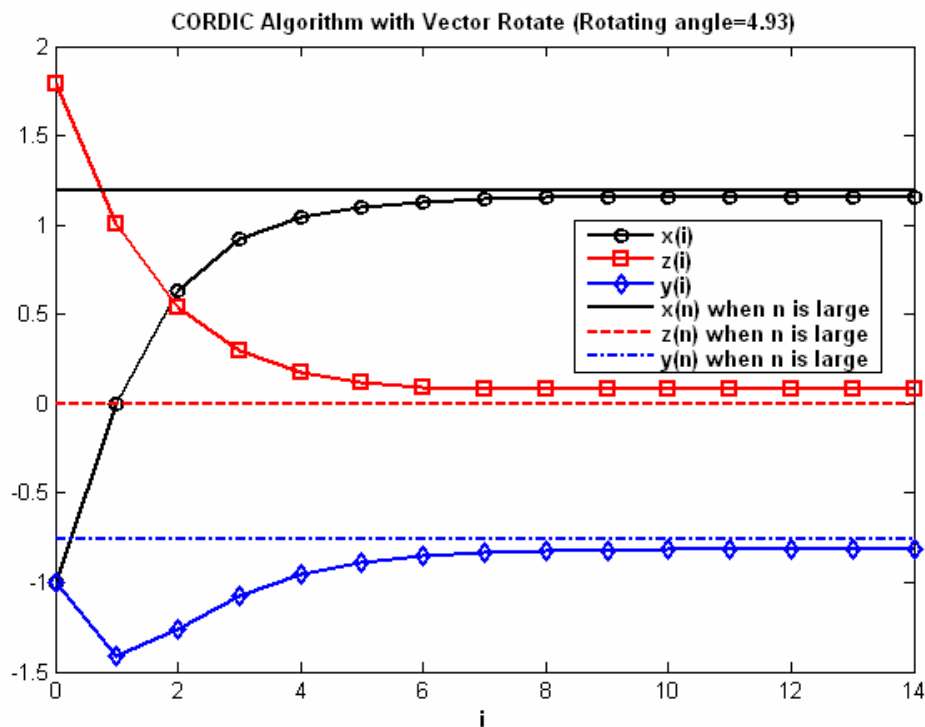


圖 4-4 CORDIC Algorithm with Vector Rotate (Rotating angle = 4.93)

接著討論在 Vectoring Mode，並針對論文主題中，估測頻率補償時的角度來做模擬。假設輸入訊號為 $1-i$ ，預期估出的角度應該是 $-\pi/4$ (-0.7854)。我們由圖 4-5 可以看到，隨著 i 的增加，預求的角度 $z(i)$ 也慢慢貼近 -0.7854 ，而 $y(i)$ 也如同之前所提到的，在 i 很大時會趨近於 0。在 Vectoring Mode 並不像 Rotating Mode 中旋轉角度要小於 4.8849 的限制，因為這邊所算出來的角度，都定義在 $-\pi$ 與 π 之間。

由圖 4-3 與圖 4-5 可以看出，CORDIC Algorithm 的收斂速度還蠻快的，大概在 $i=6\sim 8$ 之間就可達到收斂的效果，我們就以這個經驗法則來做為之後實作的參考。

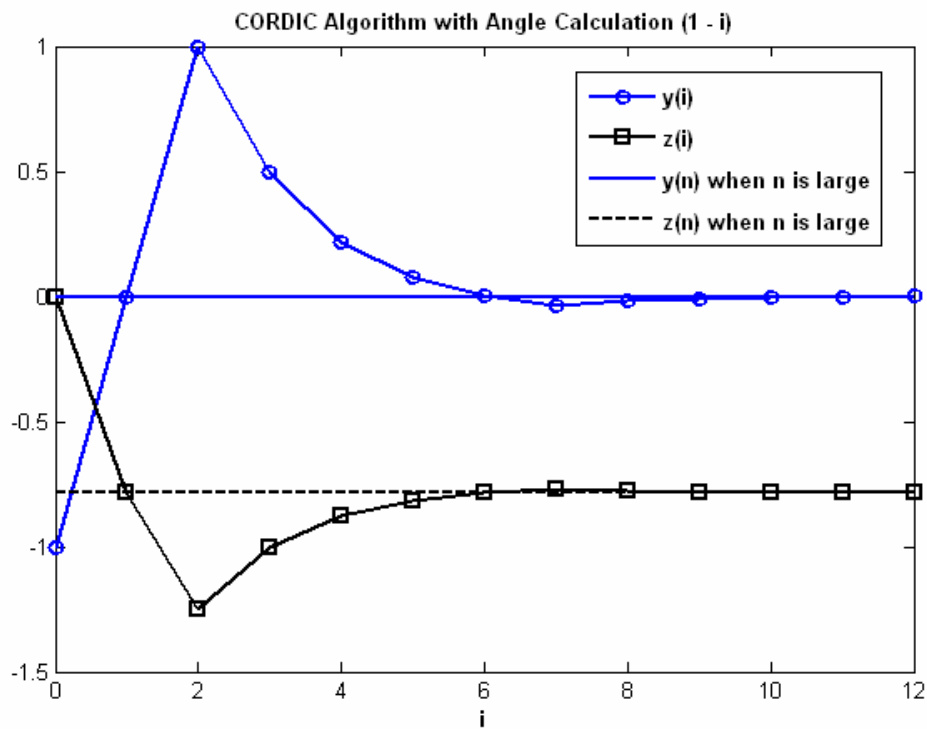


圖 4-5 CORDIC Algorithm with Angle Calculation (1 - i)

實作電路設計

在頻率估計中，我們必須先算出需要補償的角度，圖 4-6 為其架構圖，當 i 為 0 時，多工器(multiplexer)選擇初始值 x_0 、 y_0 與 z_0 ，當 i 大於 1 之後，多工器選擇由加法器運算出來的結果 x_i 、 y_i 與 z_i 。 d_i 由 y_i 的正負號來決定，進而影響 z_i ，在這裡我們關心的只有 z_i 的最終值，依照之前的討論，我們取出 z_6 做為最後的角度。

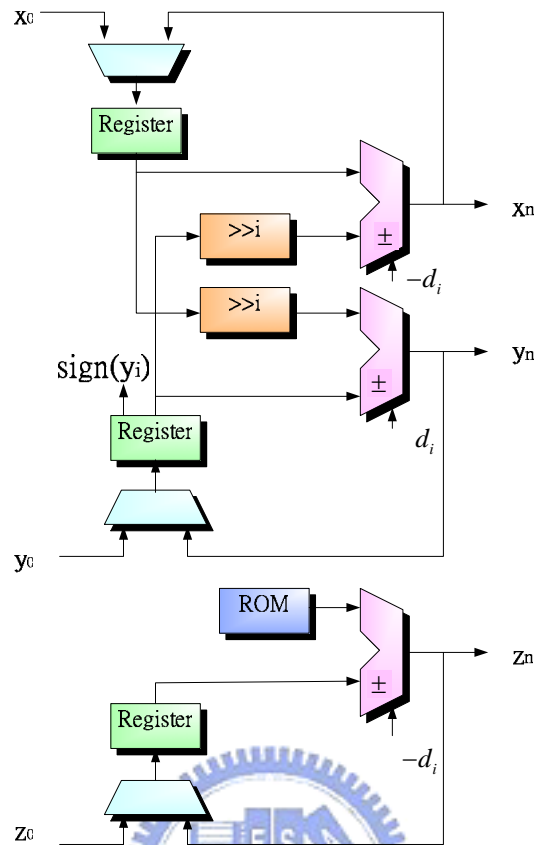


圖 4-6 Iterative CORDIC structure

算出補償的角度之後，就是以補償的角度來對輸入訊號做旋轉。如果依照圖 4-6 的架構的話，必需提高七倍的 clock rate 才行，這時我們可以利用 pipeline 的方式來設計，如圖 4-7 所示， z_0 為旋轉的角度，而 x_0 與 y_0 是欲旋轉向量的實部與虛部。 z_i 的運算由查表(LUT, Look Up Table)即 ROM 而來，而其正負號來用來控制每個加(減)法器該加還是該減。 x_0 與 y_0 再經過七級的旋轉之後，即完成了頻率補償的動作了。

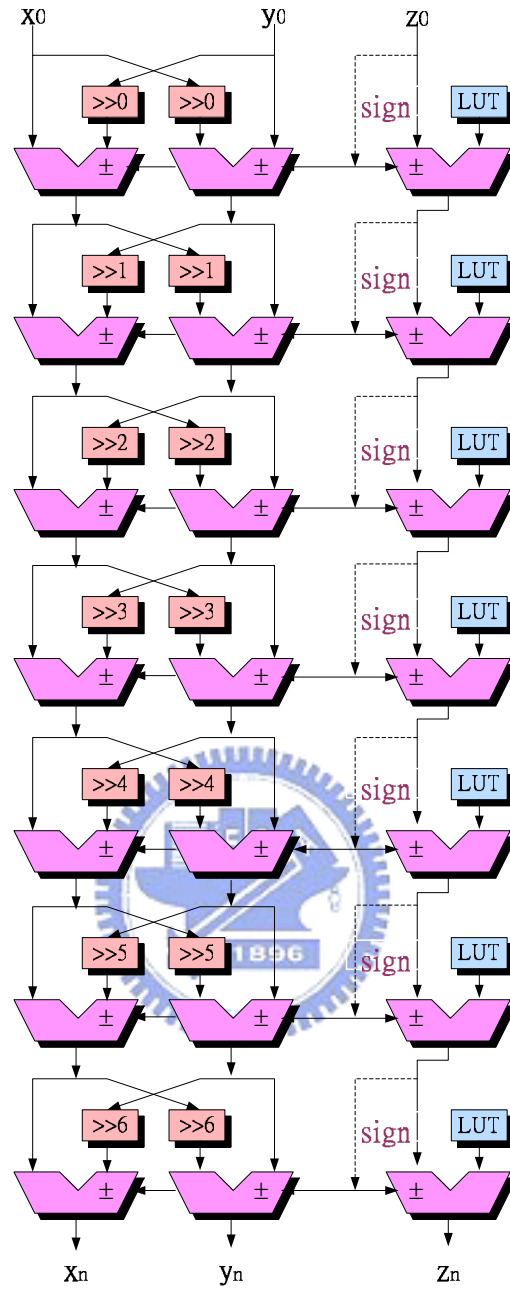


圖 4-7 Pipeline CORDIC structure

4.2.2 快速傅利葉轉換(FFT)

FFT 基本介紹

快速傅利葉轉換的運算是根據離散傅利葉轉換(DFT)的數學而來，其利用離散傅利葉轉換的複數乘法在複數平面上的對稱性質，將具有對稱性質的多個乘法合併成一項，因此可以大大地減少數學運算量，在不變更原數學模型架構之下，

能獲得較有效率的運算。該方法是在 1965 由 Cooley and Tukey 所提出的，此即為現在著名的快速傅利葉轉換(FFT)[11]，其數學模型大致上可分為兩種不同的架構，一種為時間點分組架構(Decimation In Time)，而另外一種則是頻率點分組架構(Decimation In Frequency)。本論文採用頻率點分組架構進行實現。DIF 架構([12])的推導如下：

DFT 為 $X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, k = 0, 1, \dots, N-1$ ，其中 $W_N = \exp(-j2\pi/N)$ 。將 k 點分組為偶數 k 點與奇數 k 點。觀察偶數 k 點的組合，令 $k = 2r$ ，其中 $r = 0, 1, 2, \dots, (N/2)-1$

$$\begin{aligned}
 X[2r] &= \sum_{n=0}^{N-1} x[n] W_N^{2rm} \\
 &= \sum_{n=0}^{(N/2)-1} x[n] W_N^{2rm} + \sum_{n=N/2}^{N-1} x[n] W_N^{2rm} \\
 &= \sum_{n=0}^{(N/2)-1} x[n] W_N^{2rm} + \sum_{n=0}^{(N/2)-1} x[n + (N/2)] W_N^{2r[n + (N/2)]} \\
 &= \sum_{n=0}^{(N/2)-1} (x[n] + x[n + (N/2)]) W_{N/2}^m
 \end{aligned} \tag{4-16}$$

由上面公式可知，其運算為前半部和後半部的訊號相加後做 $N/2$ 點的 DFT，在奇數點方面，則令 $k = 2r+1$ ，其中 $r = 0, 1, 2, \dots, (N/2)-1$ ，其式改寫為：

$$\begin{aligned}
 X[2r+1] &= \sum_{n=0}^{N-1} x[n] W_N^{(2r+1)n} \\
 &= \sum_{n=0}^{(N/2)-1} x[n] W_N^{(2r+1)n} + \sum_{n=N/2}^{N-1} x[n] W_N^{(2r+1)n} \\
 &= \sum_{n=0}^{(N/2)-1} x[n] W_N^{(2r+1)n} + \sum_{n=0}^{(N/2)-1} x[n + (N/2)] W_N^{(2r+1)[n + (N/2)]} \\
 &= \sum_{n=0}^{(N/2)-1} (x[n] - x[n + (N/2)]) W_N^n W_{N/2}^{nr}
 \end{aligned} \tag{4-17}$$

此即為將輸入資料的前半部與後半部資料相減後乘上轉動因素(twiddle factor)然後做 $(N/2)$ 點的 DFT。其示意圖如圖 4-8，再將第二級中 $(N/2)$ 點 DFT，以相同的原理進行分解，就可以獲如圖 4-9，著名的 DIF butterfly。由於此架構是利用在

複數平面 0° 與 180° 的對稱性，將乘法合併，因此運算可以由原先每一級 N 次複數運算降為每一級 $N/2$ 次複數運算，這種方法稱為 radix-2 架構。

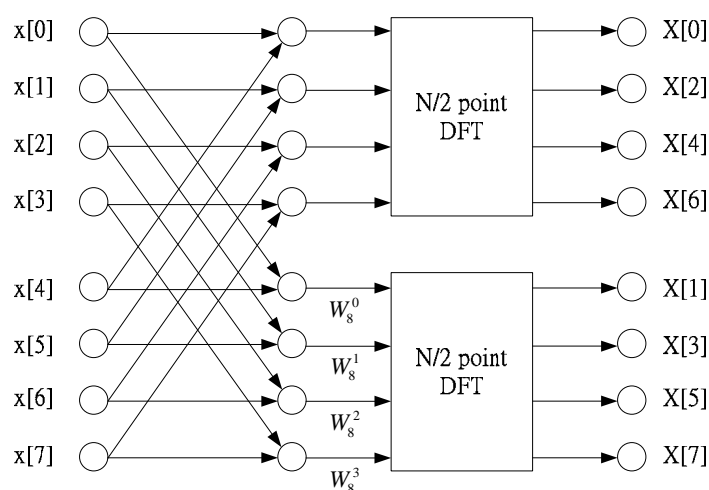


圖 4-8 八點 FFT(1)

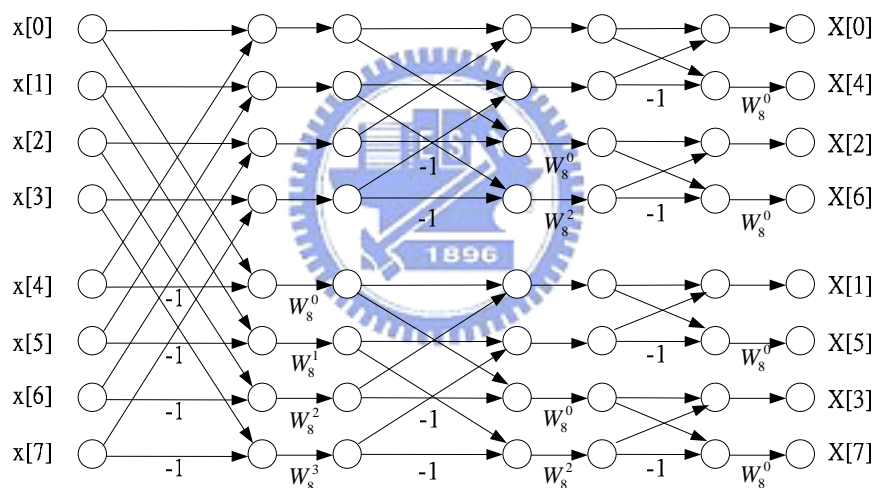


圖 4-9 八點 FFT(2)

如果將 radix-2 的觀念推廣到 0° 、 90° 、 180° 與 270° ，可以發現在複數平面上所對應的值是 1 、 $+j$ 、 -1 與 $-j$ ，而當一個複數輸入訊號乘上 $+j$ 或 $-j$ 時，其結果只是等效於實虛部交換，再將其中一項變號而已，也因為如此，原本所乘上的 twiddle factor 就成為不主要的複數乘法運算(trivial multiplication)，這也就是 radix-4 比 radix-2 運算更有效率的地方(約可減少運算量 25%)。

我們可由 (4-16) 的 $X[2r]$ 與 (4-17) 的 $X[2r+1]$ ，分別以 r 代入 $2s$ 與 $2s+1$ ，得到 radix-4 的式子如下所示：

$$\begin{aligned}
X[4s] = X[2r] \quad \Bigg|_{r=2s} &= \sum_{n=0}^{(N/4)-1} \left(x[n] + x\left[n + \frac{N}{2}\right] \right) W_{N/4}^{ns} + \sum_{n=0}^{(N/4)-1} \left(x\left[n + \frac{N}{4}\right] + x\left[n + \frac{3}{4}N\right] \right) W_{N/4}^{ns} \\
X[4s+1] = X[2r+1] \quad \Bigg|_{r=2s} &= \sum_{n=0}^{(N/4)-1} \left(x[n] - x\left[n + \frac{N}{2}\right] \right) W_{N/4}^{ns} - j \sum_{n=0}^{(N/4)-1} \left(x\left[n + \frac{N}{4}\right] - x\left[n + \frac{3}{4}N\right] \right) W_{N/4}^{ns} \\
X[4s+2] = X[2r] \quad \Bigg|_{r=2s+1} &= \sum_{n=0}^{(N/4)-1} \left(x[n] + x\left[n + \frac{N}{2}\right] \right) W_{N/4}^{2n} - \sum_{n=0}^{(N/4)-1} \left(x\left[n + \frac{N}{4}\right] + x\left[n + \frac{3}{4}N\right] \right) W_{N/4}^{2n} \\
X[4s+3] = X[2r+1] \quad \Bigg|_{r=2s+1} &= \sum_{n=0}^{(N/4)-1} \left(x[n] - x\left[n + \frac{N}{2}\right] \right) W_{N/4}^{3n} + j \sum_{n=0}^{(N/4)-1} \left(x\left[n + \frac{N}{4}\right] - x\left[n + \frac{3}{4}N\right] \right) W_{N/4}^{3n}
\end{aligned} \tag{4-18}$$

Radix-4 除了運算量減少，整體運算更有效率，也可以節省功率，實在是不錯的方式，不過缺點就是硬體設計複雜度提高，圖 4-10 為 radix-2 與 radix-4 的基本元件比較。

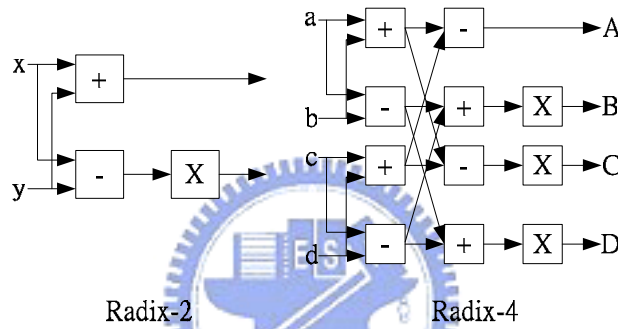


圖 4-10 Radix-2 與 Radix-4 基本元件比較

用上圖中可看出，一個 Radix-2 基本元件要兩個複數加法器與 1 個複數乘法器，而 Radix-4 則需要 8 個複數加法器及 3 個複數乘法器，硬體複雜度增加了三倍，為了改善這個問題，可採用 Radix-2²，和 Radix-4 一樣有減少運算複雜度的優點，而且仍可保持 Radix-2 的 butterfly 架構，其原理也很簡單，只是把一個 Radix-4 的運算，拆成兩級來做而已，Radix-2² 的 PE(Processing Element)如下圖。

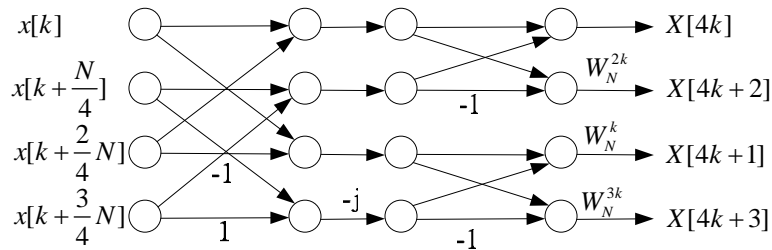


圖 4-11 PE of Radix-2²

由數學分析可以得知，FFT 之 radix 的級數越高，運算量越少，計算更有效率，但硬體的複雜度會提高，在此，我們根據 Radix-2² 推廣到 Radix-2³ (由 Radix-8

演算法推得)，將 3 個 radix-2 串接，具有 radix-2 簡單架構的優點及低複雜乘法運算量的特點，有關 Radix-8 的公式請參考[13]，Radix- 2^3 的 PE 如下圖

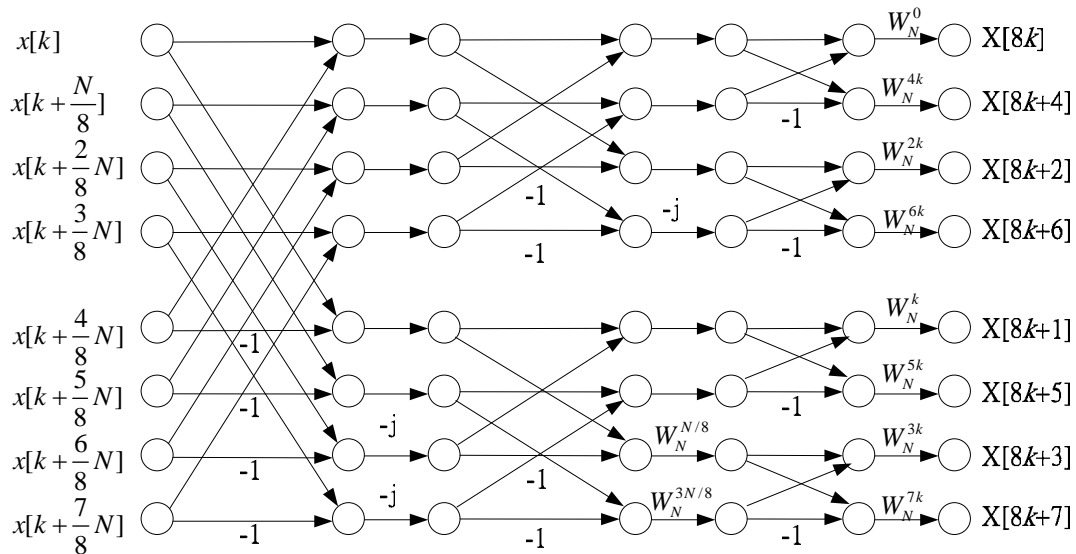


圖 4-12 PE of Radix- 2^3

FFT 硬體設計

在 FFT 上的硬體架構，主要分為兩種，一種稱為單一路延遲迴授系統 (single path delay feedback system, SDF) 或稱為 pipeline FFT，另一種為 Memory-based FFT。下圖為 Radix-2 16-points pipeline FFT 的基本架構圖。

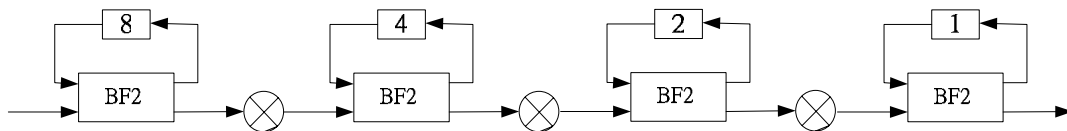


圖 4-13 16-points pipeline FFT

由上圖中，我們可以看到，當 FFT 運算進行穩定狀態後，每個暫存器的使用率將為 100%，因此系統運作正好是一級接著一級，故可以達到較高的吞吐量，而使用的記憶體大小為 $N-1$ (N 代表 N 點 FFT)，由於記憶體是屬於分散記憶體 (Distributed Memory)，因為記憶體會分散在每個 butterfly 架構之上，較單一塊的記憶體會占上較大的面積。而圖 4-14 為 memory-based FFT 架圖，一開始先由 $x(n)$ 將欲做 FFT 的訊號存到記憶體 X 上，經過 FFT CORE 一級的處理器後，再回授到記憶體 X 上，有幾級的 FFT，就會繞幾次。最後一級經過 FFT Core 出來

之後，將資料存到記憶體 Y 上，做 bit-reverse 的動作之後將結果輸出至 X(k)。

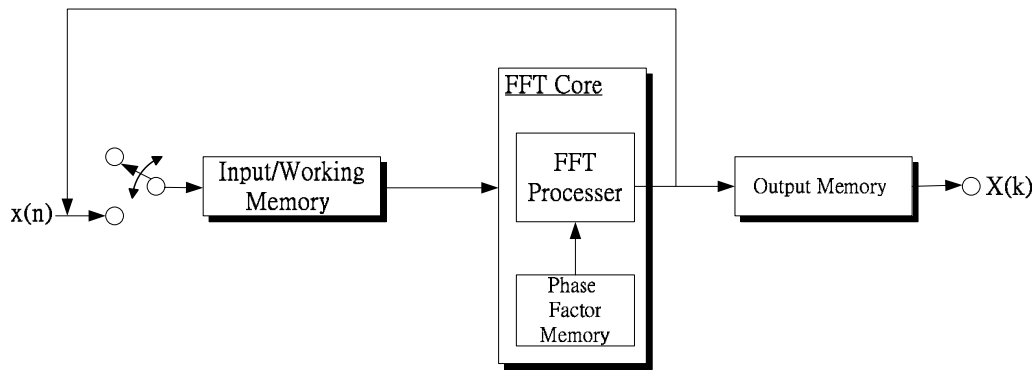


圖 4-14 Memory-Based FFT 示意圖

在 11n 的系統中，20MHz 之下，64 點的 FFT 並不大，在 pipeline FFT 架構中並不會占去太多的面積，而且其複雜度和控制方面，都比 Memory-Based FFT 簡單許多，而其管線的特性，除了讓 OFDM 在一長串連續資料做 FFT 時更有效率，也讓乘法器及暫存器的使用率達到 100%，本篇論文中，主要以 pipeline FFT 來做為硬體上的架構。

之前我們提過，Radix-8 在運算複雜度方面會比 Radix-2 簡單，因為其利用對稱性，將更多重要乘法以不重要(trivial)乘法取代，計算量減少，也更有效率，但在 FPGA 的實現上，則需考量扇入(fan-in)與扇出(fan-out)的問題，因此 radix-2 的架構反而比較好，因為其扇入，扇出數都很少，可適應 FPGA 上頻寬的限制。整合上面的優點，即可構成 Radix- 2^3 FFT。64 點 Radix- 2^3 pipeline FFT 的架構如下圖：

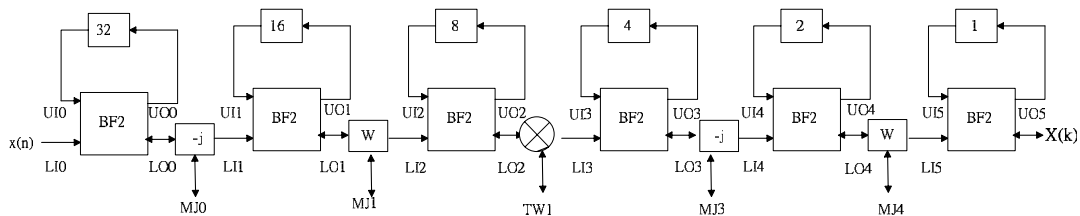


圖 4-15 64 點 Radix- 2^3 pipeline FFT

其中的 BF2 架構也很簡單，可分為兩種 mode，一種為 Bypass mode，一種為 Normal mode，如下圖 4-16

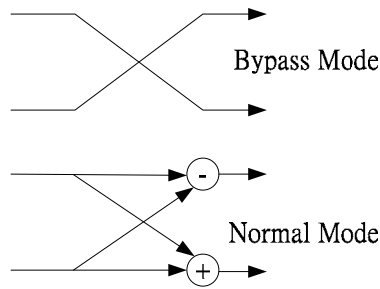


圖 4-16 PE of BF2

其中 Bypass Mode 主要是讓資料直接通過，經過一段時間的暫存並回授之後，再經由切換至 Normal Mode 中，與之後的資料做相加減。

在圖 4-15 中所示，我們只需在 TW1 的地方放上一個乘法器即可，其它的地方都只是不重要乘法，由於在之前天線收上來之後，經過 A/D 的部分為 10 個 bits，故我們也讓 $x(n)$ 為 10 個 bits，每經過一級，訊號的動態範圍就會變大兩倍，所以在每一級之後做 scale 或是 truncate 是必要的。 $X(k)$ 輸出也是 10 個 bits，而 $X(k)$ 輸出並不是以 $X(0)$ ， $X(1)$ 的依順序出來，而是 $X(0)$ ， $X(32)$ ， $X(16)$... 等等，bit-reversal 的方式輸出。為了讓輸出是依序的，我們需要兩個記憶體來將 $X(0)$ ， $X(32)$ ， $X(16)$... 對應到記憶體中存起來，再依序讀出來，其架構如下圖 4-17。用兩個記憶體的主要目的，是當第一串 FFT 資料來時，由 RAM1 做 Bit-Reversal 的方式存入時，RAM2 則依序輸出，第二串 FFT 資料由 RAM2 做 Bit-Reversal 的方式存入，由 RAM1 依序輸出。

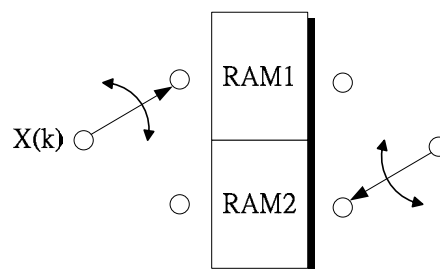


圖 4-17 Bit-Reversal output

一般來說，pipeline 的設計都是針對連續需要做 FFT 的區塊做設計最為容易，控制訊號方面也很容易實現。但在 11n 當中，HT-data 並非連續的區塊做 FFT，而是中間有著一定的間隔(即 CP=16)，再加上之前的通道估計，也會用到 FFT，

使得時序上及控制訊號上的安排更加的困難，我們可以參考下圖 4-18

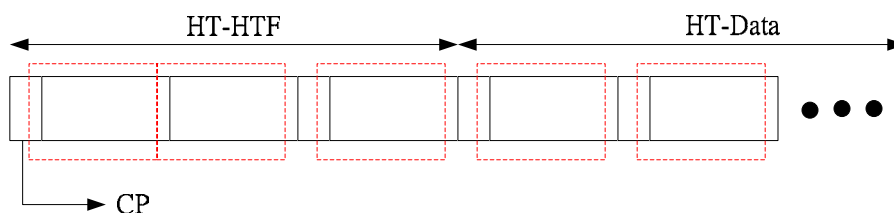


圖 4-18 FFT Block(虛線部分)

由上圖我們可以看到，執行 FFT 的區塊並不是連續的，而且其中帶有一點不規則性(HT-HTF 的部分)，這時我們需要一些計數器，來設法跳過不做 FFT 的部分。如要避開這個不規則性，我們也可以把第一個 FFT Block 拿掉(如上圖 4-18 第一個虛線框框)，因為這裡重覆的 HT-HTF 區塊主要是用來做頻率補償的。這樣一來，就可以更加的簡化我們的設計了。

4.3 封包偵測 (Packet Detection)

由 3.1 的討論當中，我們得知由 DCND 的方法可以得到較好的效能，在此，我們也針對這個方法以硬體來實現。最基本的架構圖，如圖 4-19 所示，此架構圖由(3-3)而來。我們曾在圖 3-3 中討論過，當 $SNR > 10dB$ 時，此時最佳的門檻值(m_n)可選擇介於 0.5 至 0.6 之間，這時假警報或是封包遺失的機率都小於 1% 以下。可是由下圖 4-19 中可看到，除了若干乘法器外，我們還需要一個準確的除法器，這是我們所不樂見的。除法器除了會占去不少的面積，也會造成數十個週期的延遲，這對封包偵測來說，是很沒有效率的，因為做這個封包偵測，已經讓我們浪費了 32 個短訓練符元(L-STF)($D=16$)，在不想浪費記憶體存著的情況下，是不容許再有更多的延遲。

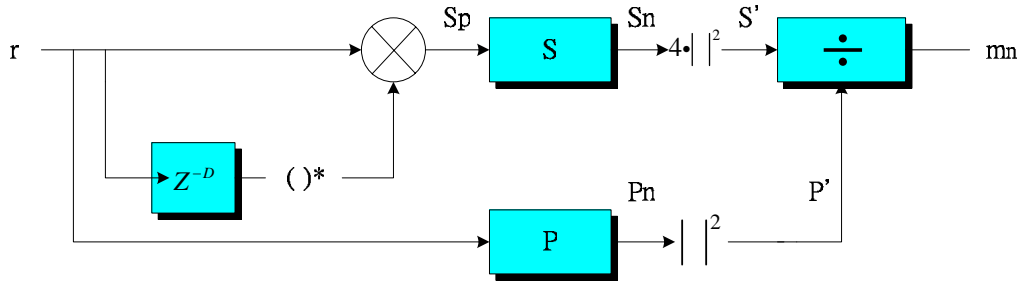


圖 4-19 DCND 基本架構圖

為了兼顧到面積與時序，我們必須做出一些讓步。在圖 3-3 中，假如將門檻值選在 0.5，僅會稍稍的提高假警報的機會(約 3%)，但是封包遺失的情形幾乎不會發生，我們可以容許較高的假警報機會，是因為可以由後面 HT-SIG 中的 CRC 解回來，但如果封包遺失的機會過高時，則需要做重傳的動作，相當浪費頻寬。而門檻值設為 **0.5** 能夠大大的增加我們在硬體上的便利性，我們只需要將圖 4-19 中的 S' 乘上兩倍之後，經過一個比較器，當 $2S' > P'$ 時，我們判定偵測到封包。

而另一個設計的重點則是圖 4-19 中 S 區塊的部分，當 S 區塊前面的複數乘法做完後(即圖 4-19 中 S_p)，會在 S 區塊中做累加的動作，但因為是以滑動視窗(Sliding Window)的方式，只有累加 16 個值，也就是說累加完現在的值之後，要把 16 個取樣值前的 S_p 扣掉才行，故 S 區塊的設計如下圖 4-20 所示，而 P 區塊原理則相同。

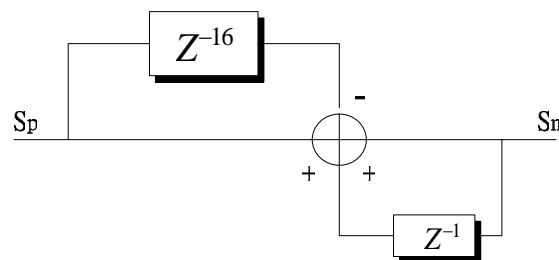


圖 4-20 S 區塊示意圖

在定點數模擬方面(fix point simulation)，假警報的機率可以降到約 7%(較浮點數模擬 3% 來的差)，而封包遺失的機率可降至 0(與浮點數模擬相同)。圖 4-21 為整體的架構圖與定點數配置圖。

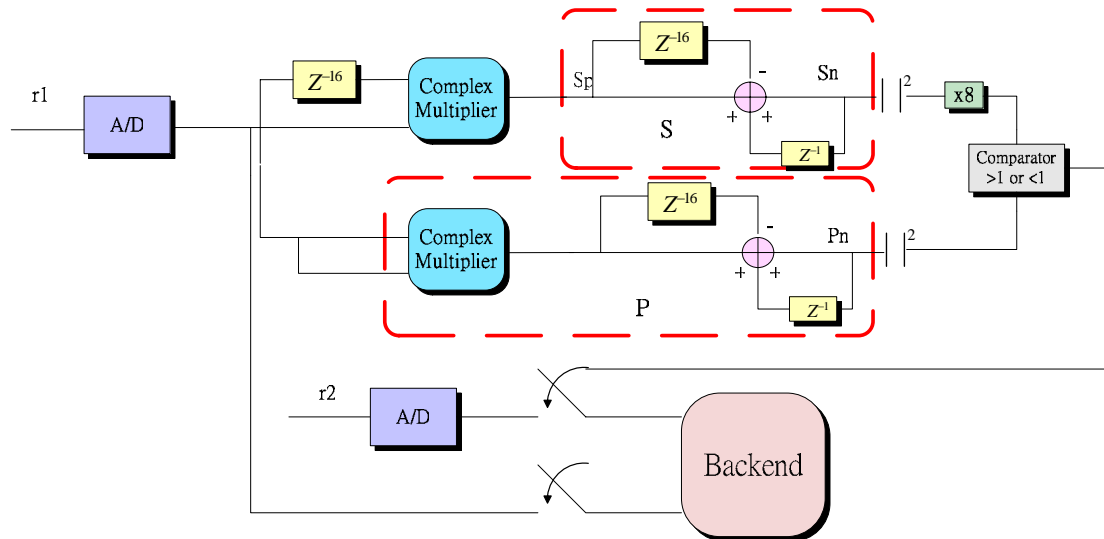


圖 4-21 SOP 整體架構與定點數(fix point)配置

4.4 自動增益控制 (AGC, Auto Gain Control)

自動增益控制是用來偵測原本過弱或過強的訊號，並將其補償放大或縮小到 A/D 正常的操作區域，以避免過多的溢出(overflow)或是解析度不夠。舉個例子，假如我們的 A/D 是將 1 與 -1 間的訊號做 8 位元(256 個 level)的取樣，而如果收下來的訊號功率只有 0.25，則訊號很可能只集中在 LSB 附近，這時則需要 AGC 偵測並調整放大兩倍。假如收到的訊號功率為 4，則訊號容易超過 1 或 -1 而被砍掉造成失真，則這時 AGC 可以偵測並調整減小兩倍。

而在 11n 中有前導訊號有兩部分是在做自動增益補償的，分別的 L-STF 與 HT-STF，其中 HT-STF 必需能夠較準確的估算訊號的功率，因為後面的通道估測影響整體效能很大，而 L-STF 只需做粗步的估計即可。如何使 HT-STF 有較準確的估計功率呢，TGn Sync 在[2]中特別以頻率交錯(Tone Interleaving)的方式來安排 HT-STF，這是因為 HT-STF 是一個固定的訊號，如果沒有做頻率交錯，在接收天線端估測功率會容易受到各個不同通道的影響，因而造成功率率的估不準。我們來看看有做頻率交錯與沒有做頻率交錯的比較如下圖 4-22，表示在頻率交錯與不做頻率交錯之下，用 HT-HTF 估出的功率與資料功率(Data power)的比值(dB)與累積分佈函數(cumulative distribution function, CDF)關係的比較圖。可以看出在頻率交錯之下其分布範圍與資料功率差不多，但在沒有頻率交錯時，估出功率

的範圍則明顯增大不少。由此可知，頻率交錯能幫助我們增加在估訊號功率上的準確性。

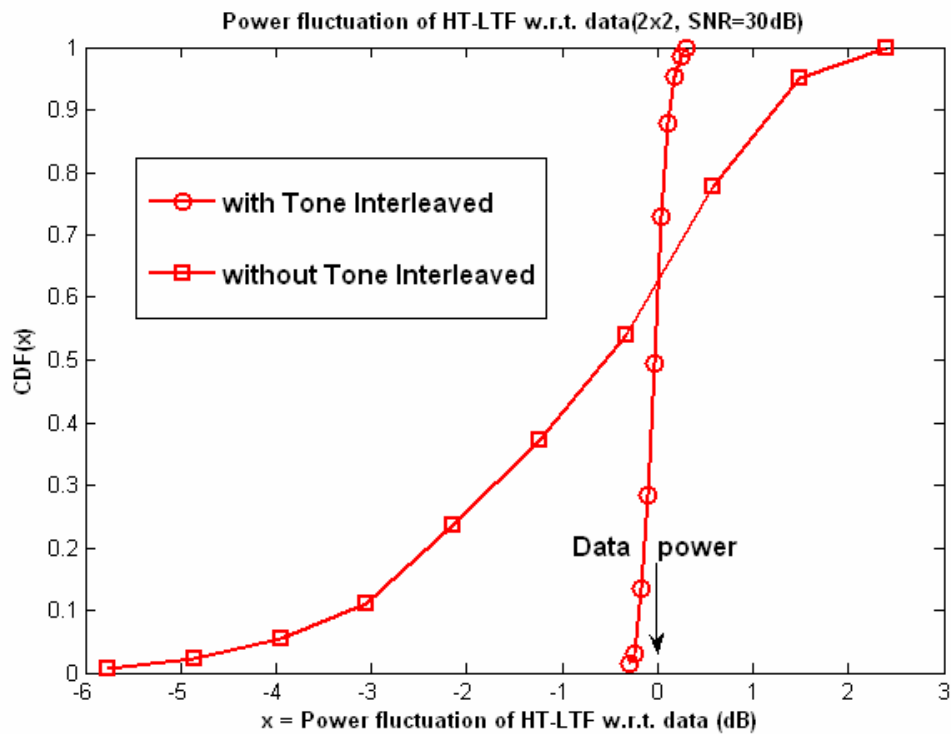


圖 4-22 Power fluctuation of HT-LTF w.r.t. data

硬體設計方面相當簡單，由於計算功率只要每個取樣值的絕對值平方相加即可，故只需要一個複數乘法器和加法器即可，其架構如下圖所示：

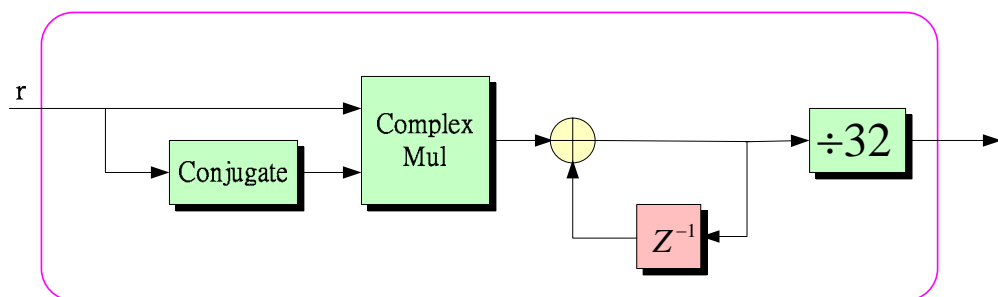


圖 4-23 AGC 硬體架構圖

不同的接收天線收下來的訊號(上圖中的 r)各自估算自己的功率，並將其輸出信號回授給前端放大器做放大和或是縮小，而 AGC 所影響的效能，我們將在 4.7 中做說明。

4.5 碼框偵測 (Frame Detection)

在 OFDM 接收機中，通常要有一個好的碼框偵測，才能在後面做快速整傅利葉轉換(FFT)時，取到正確的值。一般說來碼框偵測的做法，都是利用匹配濾波器(Matched Filter)，將接收到的訊號與已知的前導訊號做匹配，與封包偵測不同的地方是，在做完匹配之後，不是經過一個門檻值，而是取出一個最大值，這樣才有辦法做更準確的時序控制，當然前提是在一定數目的值之中挑選一個最大值出來。

而我們選用來匹配接收訊號的前導訊號取自L-STF的最後四個取樣點與L-LTF的最前面四個取樣點，因為這兩組前導訊號的相關性不大，有較為明顯的界線(Boundary)，會更容易偵測出碼框的位置。而一般說來，匹配濾波器的架構有以下三種，如圖 4-24所示。圖(a)為最基本常見的架構，但是有最長的重要路徑(critical path) 需要經過七加法器和一個乘法器，所以重要路徑太長是其缺點，而圖(b)的缺點則是扇入(fan-in)太大，需要大的驅動能力，圖(c)可以改良(a)和(b)的缺點，在兩者之間取得一個平衡，所以我們採用(c)的架構，設計如圖 4-25的改良式的匹配濾波器(Hybrid form with length 2 subfilters)，並在輸出的地方做一個取最大值位置(index)的電路，這就完成我們碼框偵測的硬體架構了。

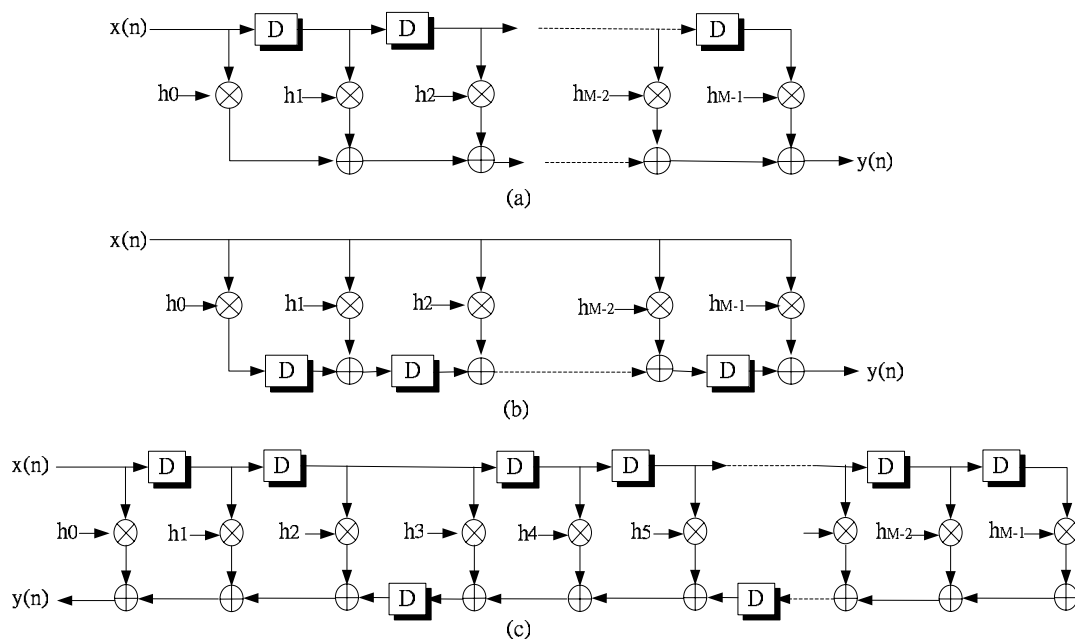


圖 4-24 (a)Direct form (b) Transposed form (c) Hybrid form (with length 3 subfilters)

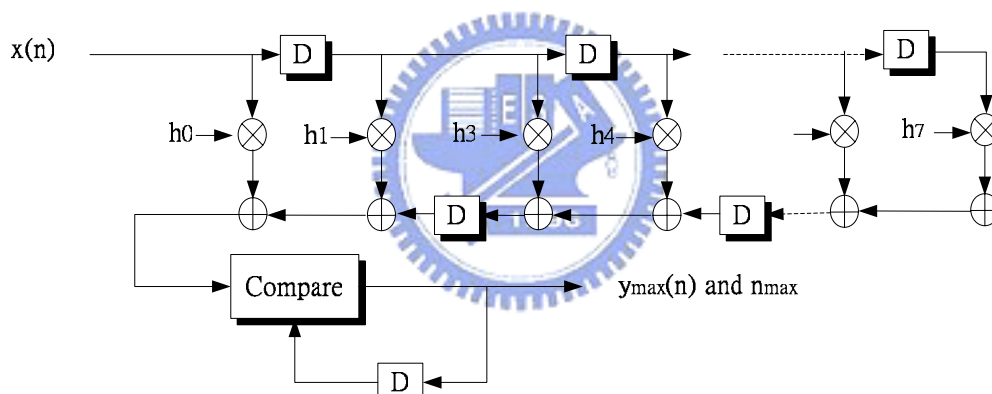


圖 4-25 Frame Detection 架構圖

4.6 頻率偏移估計 (Frequency Offset Estimation)

在 3.2 中曾提過頻率偏移估計的方法，是由兩個相隔 D 的區塊向量，做內積之後求出其角度，再將此頻率偏移補償回來，這個做法與封包偵測還蠻類似的。又因為我們會在兩個地方做頻率補償，一個是在 L-STF 做粗調而在 L-LTF 的地方做細調。另外在頻率偏移估計中的另一個重點就是角度的估計與向量的旋轉，這在 4.2.1 中有詳細的介紹。

由 3.2 可知，我們將頻率偏移補償分為兩個階段，一個是頻率偏移粗調，一

個則是頻率偏移細調，也就是補償頻率偏移時，我們必需對接收到的訊號做兩次的旋轉才行。其基本的架構圖如下圖 4-26。

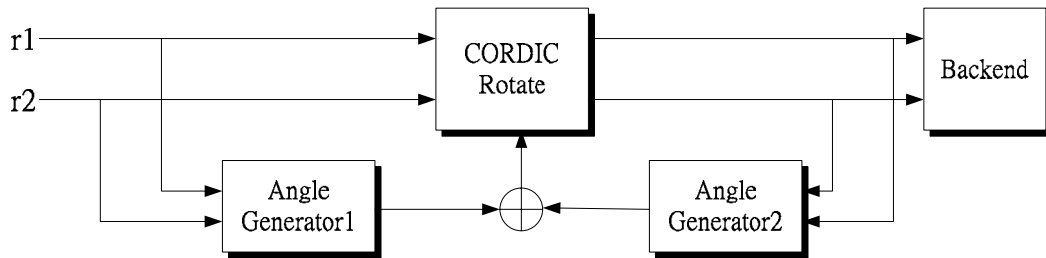


圖 4-26 頻率偏移補償方塊圖

圖 4-26 中， $r1$ 與 $r2$ 代表接收到的兩個資料串。Angle Generator1 與 Angle Generator2 分別是由 L-STF 與 L-LTF 產生的旋轉角度，而 Angle Generator2 的輸出，必需等 L-HTF 經由 CORDIC Rotate 旋轉 Angle Generator1 所產生的角度之後，才由 Angle Generator2 計算其第二次的旋轉角度，並與第一次旋轉角度相加之後，算出所需旋轉的總角度。有關 CORDIC Rotate 的硬體設計，可參考圖 4-7。

接下來則是 Angle Generator 的硬體設計。依照 (3-11) 及 (3-7)，我們的硬體架構如下圖 4-27:

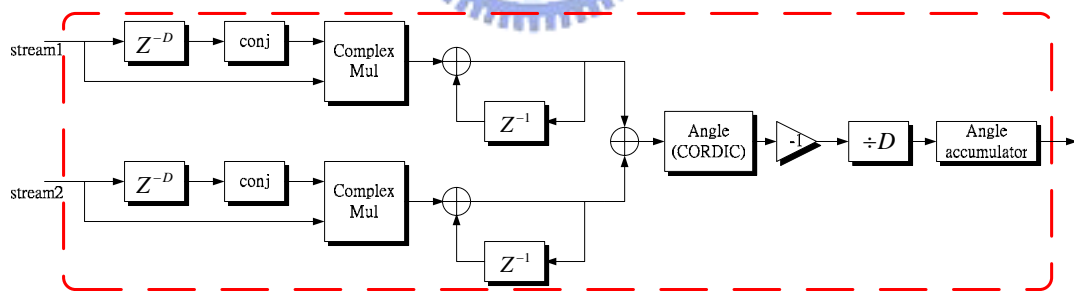


圖 4-27 Angle Generator 硬體架構圖 1

其中的 D 在粗調時(利用前導訊號 L-STF)為 16(Angle Generator1 架構)，在細調時(利用前導訊號 L-LTF)為 64(Angle Generator2 架構)。而 Fix Angle accumulator 是一個固定角度的累加器(有別於在 3.2.1 中介紹的角度累加器)，其功能為對每個取樣值旋轉固定角度的倍數，此方塊稍候會再介紹，至於 $\div D$ 的方塊，因為 D 是 2 的次方，所以只要做小數點的向右位移即可。而由圖 4-27 看出，當 D 為 16 時，共需要 $16 \times 2 \times 2$ 個($D=16$ ，兩根天線與實部虛部)register，而當 D 為 64($D=64$ ，

兩根天線與實部虛部)時，也要浪費 $64 \times 2 \times 2$ 個 register，和兩個複數的乘法器，這是相當浪費的，那還有減少硬體的空間嗎?答案是肯定的，我們只需將(3-11)做一點小修改即可，如下式

$$\begin{aligned}
 z &= \sum_{n=0}^{L-1} \left[\left(\sum_{q=1}^{N_r} r_{q,n} \right) \left(\sum_{q=1}^{N_r} r_{q,n+D}^* \right) \right] \\
 &= \sum_{n=0}^{L-1} \left[(N_r \cdot r_n) \cdot (N_r \cdot r_n^*) \right] \text{ if } r_{q,n} = r_n \\
 &= N_r^2 \cdot \sum_{n=0}^{L-1} \left[s_n e^{j2\pi\Delta f n T_s} \cdot (s_{n+D} e^{j2\pi\Delta f (n+D) T_s})^* \right] \\
 &= e^{-j2\pi\Delta f D T_s} \cdot N_r^2 \cdot \sum_{n=0}^{L-1} |s_n|^2
 \end{aligned} \tag{4-19}$$

再利用(3-7)估出頻率偏移即可。在不考慮雜訊的情況下，依照 11n 的 spec，傳送端的前導訊號是固定的，假設通道只有一條路徑並且都是 1(實際上這種通道會造成在 MMSE 解不出來，這裡只是方便做導證)，則可令 $r_{q,n} = r_n$ 。請注意(3-11)與(4-19)的不同，(3-11)是將各個天線的相關性算出後再相加，而(4-19)則是相將天線間的取樣值相加後，再做相關性的運算。這樣一來，不管幾根接收天線，我們永遠只需要 $D \times 2 \times 2$ 個 register 與一個複數乘法器，不過這種架構比起圖 4-27 的架構在效能上稍差，這點將在 4.7 中有模擬與說明。

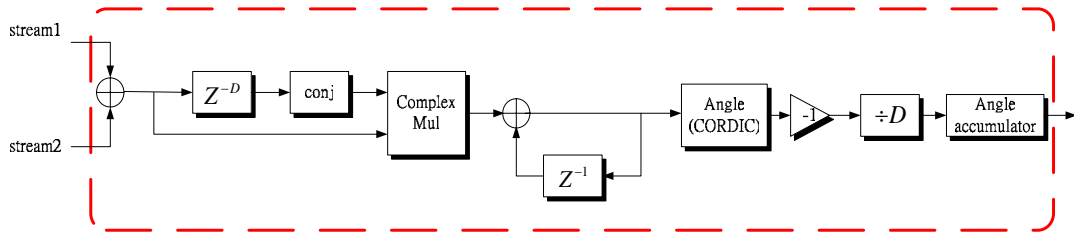


圖 4-28 Angle Generator 硬體架構圖 2

圖 4-27 中的 Fix Angle accumulator 是固定角度累加器，其架構如下：

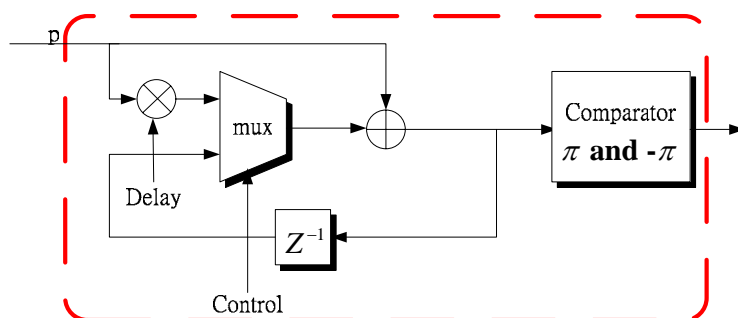
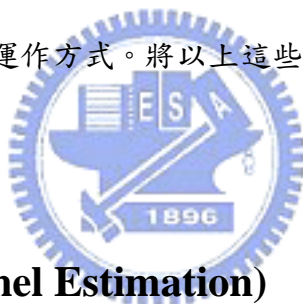


圖 4-29 Fix Angle Accumulator

上圖中的輸入只是一個最小的旋轉角度 p ，給第一個取樣值做旋轉，而之後的旋轉角度分別為 $2p$, $3p$...，但由於 Angle Generator 元件有延遲的關係，所以旋轉值應該要從(延遲時間 $\times p$)開始累加。由於角度累加大於 π 或小於 $-\pi$ 時，為了避免超過 CODIC Rotate 所能操作的範圍，所以要做減 2π 與加 2π 的動作才行，這是圖 4-29 中 Comparator 的運作方式。將以上這些元件兜起來後，就可當作我們的頻率偏移補償區塊了。



4.7 通道估計 (Channel Estimation)

在通道估計中，根據 TGn Sync 三月份的提案，每個 HT-LTF 都會重複一次，所以我們得將收到的訊號，做前後的的相加再做平均，如圖 4-30 所示， $r1$ 與 $r2$ (兩根接收天線下的兩個資料串) 會經過 64 個時間延遲，做相加後才經過乘法器。由於負責估計通道的前導訊號(HT-LTF)是由一堆 $+1$ 與 -1 所組成，所以在硬體實現上相當的簡單，比較困難的地方在於輸出通道係數時，由於傳送端有經過頻率交錯，所以在接收端估計的通道會發生通道係數交錯在一起的情形，為了分離出這些交錯在一起的係數，我們必需要把通道係數先用記憶體存起來，再依照一定的順序讀出來。在 2×2 根天線之下，總共會有四個通道，由於接收完前導訊號(HT-LTF)之後，經過一個 CP 長度(16 點取樣值)，緊接著就是資料進來，假如我們希望第一個 tone 的四個通道同時進到 MMSE Detector 的話，我們需要四個 64

大小的記憶體，因為通道包含實部與虛部，所以實際上需要八個 64 長度的記憶體。經過模擬，通道係數的位元數在 10 個位元左右，可以達到足夠的效能。依照(3-14)，通道估計的架構圖如下：

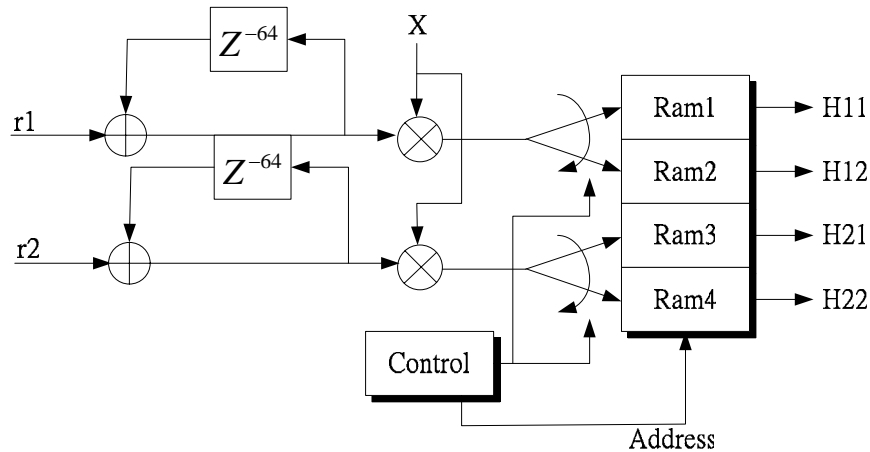


圖 4-30 通道估計硬體架構圖

r1 與 r2 代表輸入的兩個資料串(主要指 HT-HTF 區塊)。在上圖中，由於 X 只是 +1 與 -1 而已，所以乘法器可用簡單的查表與判斷就可以取代，而當所有的通道係數都存好在記憶體中，後面資料進來時，就可以四塊記憶體(包含實數與虛數其實共八塊)平行的輸出係數給 MMSE Detector 做運算(四個 RAM 依序由位址 1、位址 2...)，進而減少過多的延遲。

4.8 模擬結果

有了之前的功能方塊之後，我們對整個系統以 MATLAB 做定點數的模擬。採用定點數的部分主要是針對前面的前導符元做訊號處理，如封包偵測，自動增益控制，頻率偏移補償，通道估測等，而其他的環境與參數，請參考 3.5 節。

為了預留在系統末端檢測(Detection)的部分在定點數方面效能下降的空間，我們希望 BER 在 10^{-2} 時，浮點數與定點數的差別超過 1dB，在進行一些定點數的調整之後，我們跑出來如圖 4-31，這圖代表了浮點數與定點數上，SNR 與 BER 的關係，其中的 Preadd 代表圖 4-28 的架構，而 Postadd 代表圖 4-27 的

架構。可以看出在用 Postadd 的定點數頻率偏移架構之下，與浮點數相差無幾。接著再以面積上較為節省的 Preadd 架構，在 BER 為 10^{-2} 時，浮點數與定點數大約只相差不到 0.5dB。而 Preadd 的效能比較差的原因，是因為每個通道的脈衝響應差異大的情形下，在接收端先做加起來的動作時，會破壞掉用來做頻率偏移估的前導訊號(如 L-STF)的重覆性，效能也就跟著變差。在這個模擬圖上，為了避免因為封包偵測錯誤或是碼框偵測的錯誤而造成效能方面的嚴重下降，我們在此跳過這些封包在錯誤率上的計算。

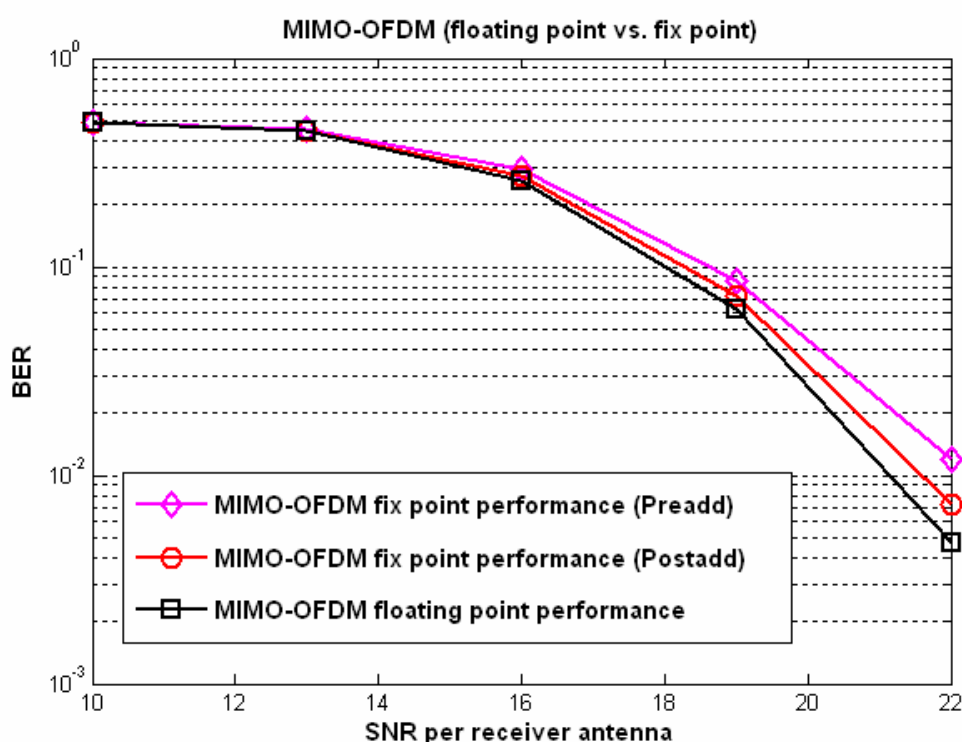


圖 4-31 MIMO-OFDM (floating point vs. fix point)

接著我們針對自動增益控制，來模擬其效能，我們使用了 L-STF 與 HT-STF 來做兩次的自動增益控制。由於之前定點數的模擬上，訊號與通道的振幅大小 (magnitude) 變化不大，很難看出自動增益控制的效能。所以在模擬上，必需稍微改變一下通道的振幅大小。我們假設了兩種情況，一種是通道放大五倍的情形，而另一種則是通道縮小五倍的情形。我們可以看出這兩條線在圖 4-32 上的效能，在通道放大五倍時，大部分的訊號在經過 A/D 時，都會被量化成最大值(即

溢出，Overflow)，使得效能大大的降低，而當通道減小五倍時，則對每個取樣值而言，很容易被量化掉很多資訊，也就是其解析度(resolution)不足的情況下，效能也跟著下降。接著，我們藉由自動增益控制來補償訊號的過大或是過小。我們先在正常的通道振幅下做統計，發現在 L-STF 區塊所收到的功率約為 0.0156。接著，當通道振幅有所改變時，我們把 RF 收下來的訊號提高 $\sqrt{0.0156 / power_est}$ 倍，再進行量化，其中 $power_est$ 是我們由 L-STF 所估計出來的能量，另外在 HT-STF 的前導訊號上，以此類推，所得到的效能如圖 4-32 所示，可以發現 5 倍與 1/5 倍的通道係數，在經過 AGC 的補償之後，都與正常的通道係數幾乎貼在一起，顯示自動增益補償發揮了效用。

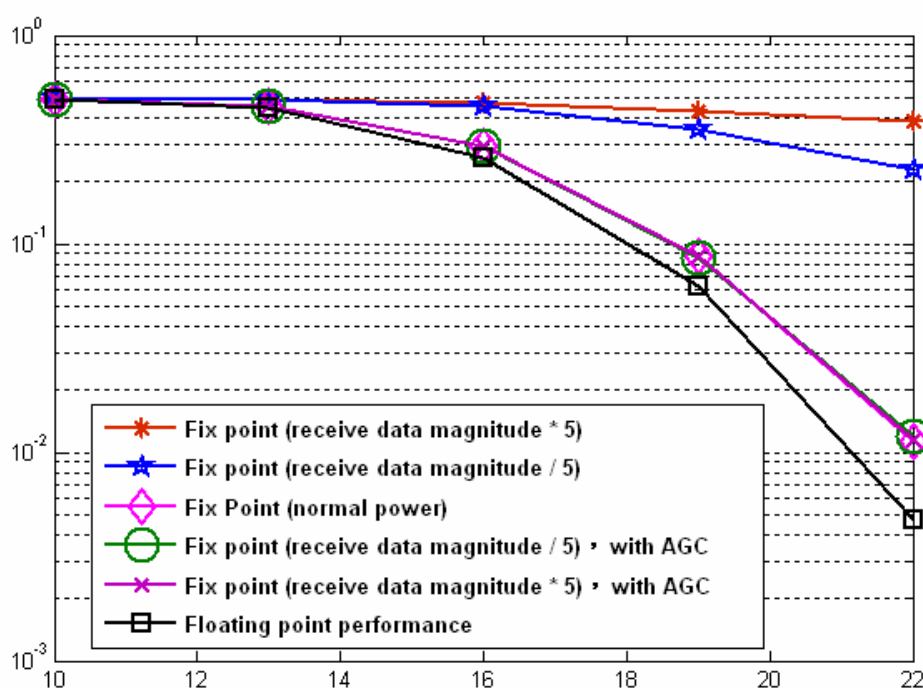


圖 4-32 MIMO-OFDM (AGC performance)

4.9 802.11n 接收機實現

我們所設計的 RTL 架構圖如圖 4-35 所示，請注意這裡的接收機並不包括之後檢測(Detection)的部分。我們先由 Packet Detection 方塊來偵測是否已收到封

包，再將此資訊傳給 Control 方塊，依序控制之後各個方塊的啟動時間與運作的時序。而接下來依序是 Coarse FO 方塊做初步頻率補償，Frame Detection 方塊做時序的對齊，和 Fine FO 方塊做精準的頻率偏移補償。之後兩個資料流各別通過一個 FFT 方塊做快速傅利葉轉換，再將輸出訊號通過 Channel Estimation 方塊做通道的估計，將通道估計參數與資料傳給下一級，這就完成了我們 802.11n 整個接收機的實現。

表格 4-1 為硬體上粗略複雜度的計算，可看出 FFT 與 Frequency Offset 所占的面積較大。FFT 面積較大的原因，是因為我們要放置兩個 FFT 方塊在硬體中。而 Frequency Offset 則是因為其暫存器與乘法器太多的關係。其中可以看出圖 4-33 與圖 4-34 為接收機的 Mapping Report 與 Timing Report，總共用去了約 40 萬閘數。而最大工作頻率為 40.323MHz，也符合我們在 11n 下 20MHz 頻寬的要求。



	乘法器	加法器	暫存器	Gate Count(萬)
Packet Detection	4	4	50	5
AGC	1	1	1	3
Frequency Offset	4	18	96	8
Frame Detection	8	8	8	5
FFT	1	12	68	16
Channel Estimation	0	0	128	6

表格 4-1 硬體粗略複雜度計算

```

Design Summary
-----
Number of errors:      0
Number of warnings:    0
Logic Utilization:
  Total Number Slice Registers:  9,126 out of 38,400   23%
    Number used as Flip Flops:    9,122
    Number used as Latches:        4
  Number of 4 input LUTs:        18,202 out of 38,400  47%
Logic Distribution:
  Number of occupied Slices:      9,751 out of 19,200   50%
  Number of Slices containing only related logic: 9,751 out of 9,751 100%
  Number of Slices containing unrelated logic:    0 out of 9,751   0%
    *See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:        18,758 out of 38,400  48%
  Number used as logic:            18,202
  Number used as a route-thru:      556
  Number of bonded IOBs:           178 out of 404     44%
    IOB Flip Flops:                 44
  Number of Block RAMs:             17 out of 160     10%
  Number of GCLKs:                   1 out of 4       25%
  Number of GCLKIOBs:                1 out of 4       25%

Total equivalent gate count for design: 402,158
Additional JTAG gate count for IOBs: 8,592
Peak Memory Usage: 294 MB

```

圖 4-33 接收機 mapping report

```

Design statistics:
  Minimum period: 24.800ns (Maximum frequency: 40.323MHz)

```

圖 4-34 接收機 timing report



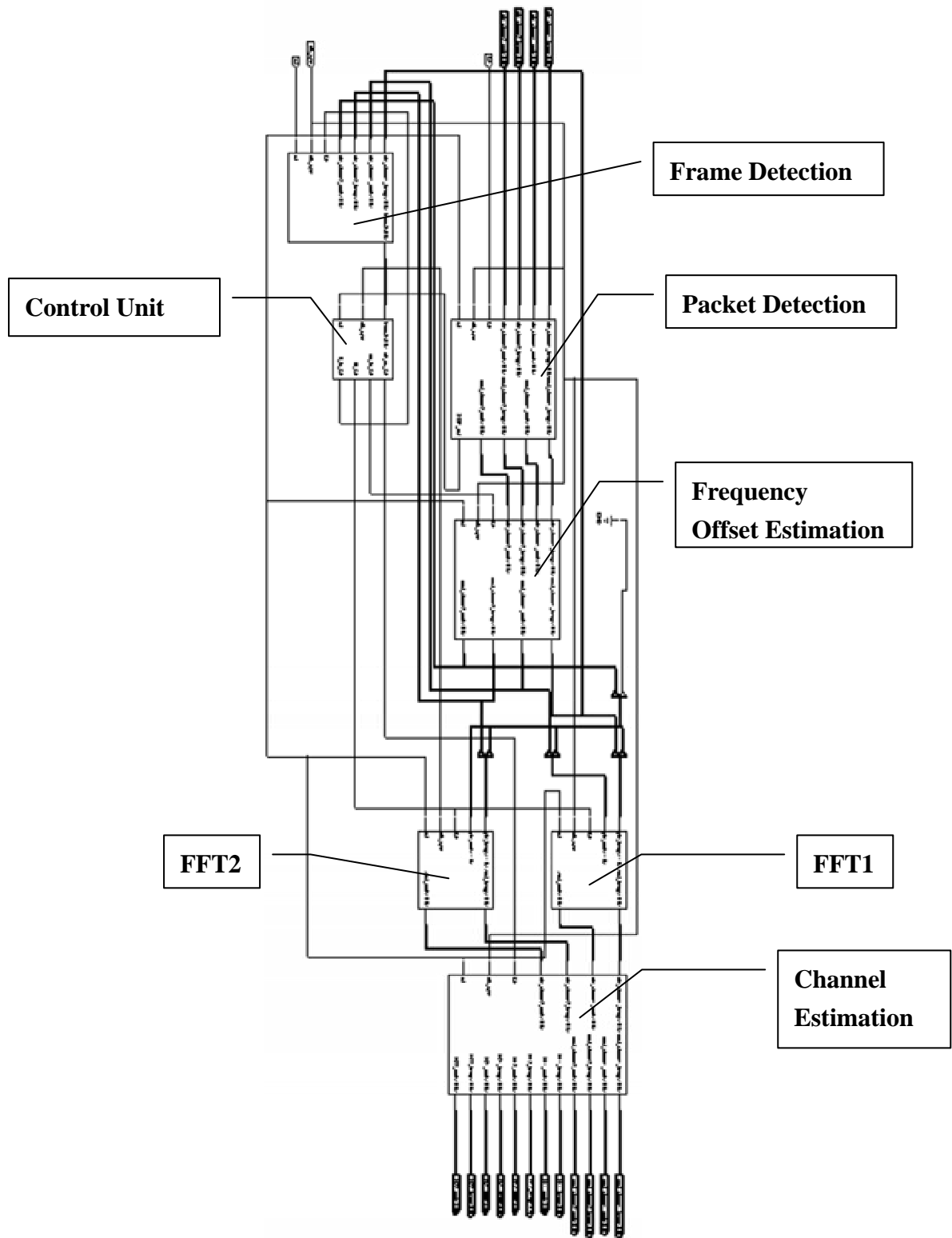


圖 4-35 接收機 RTL 架構圖

4.10 實作心得

由於本論文是做硬體的設計與實現，在硬體設計與撰寫程式的過程中，常常會遇到一些困難，必須要試著不斷的嘗試並解決問題，這也引發了我想將實作的一些心得寫在這篇論文當中。

我們都知道，要由原本習慣以演算法來架構整個系統的話，要再跨越到硬體設計這個領域上，是需要一些時間來適應的。在以演算法來架構系統上，通常我們會以 MATLAB 或是 C 來做程式的撰寫，程式碼是以循序(sequential)的方式執行的，對於方塊與方塊之間的連結相當的方便。但在硬體實現上，由於硬體語言的執行方式是並進的(concurrent)，所以在硬體方面，我們會需要相當多的控制訊號來控制整體的運作，隨著硬體的增大，這些控制訊號也會越來越複雜，這算是硬體設計上比較大的門檻。另一個門檻則是從原本在 MATLAB 上浮點數的模擬到 VERILOG 上所使用的必須為定點數來實現，我的做法則是在 MATLAB 上修改原本浮點數的程式，變成以定點數所組成的程式。定點數的模擬比浮點數困難許多，因為每一個乘法器，加法器，移位器的位元數都得考慮的很清楚才行。而在定點數方面，不見得要把所有的值都變成是整數，這樣會造成與原本浮點數的值相去過遠，在偵錯方面會太過困難。舉個例子，如果某數字為十進位數的 2.9，要將它以 7 個位元來表示，我們將其設定為 3.4 個位元(代表小數點前面有 3 個位元，後面有 4 個位元)，我的做法是先將 2.9 乘上 2 的 4 次方後，再將小數點省去，變為 46，再將 46 除回 2 的四次方，變為 2.875，而在二進位的表示上，則為 010.1110。以 2.875 來當作我們經過定點數運算的值，與原本浮點數的程式較為相近，在偵錯方面，就顯得容易多了。

以上實作心得只是在實作上的一個簡單的入門而已，硬體設計的技巧必須要靠經驗的累積，最好的方法，還是實際的動手去做，並適時的請教有經驗的人，才是最基本做實作的精神。

第5章 結論

在本篇論文中，首先對 TGn Sync 在 IEEE 802.11n 的提案中所介紹的規格做一個說明，接著針對 802.11n 中前導訊號(preamble)的部分，包括封包偵測、頻率偏移估計與補償、自動增益控制與通道估測等演算法做介紹，並將這些演算法發展其架構用到之後所要設計的硬體上面。

在封包偵測(Packet Detection)方面，我們除了針對舊有且較常用的演算法做介紹與比較，並針對其中一個較有效率的演算法發展其架構，並將其本來所需要的除法器替換成另一個更省面積的比較器，而效能不致於差太多。

在頻率偏移估計與補償(Frequency Offset Estimation and Compensation)上，將原本在單天線輸入輸出(Single Input Single Output, SISO)推廣到多根天線上面，並針對其效能做模擬。在實作上，我們也利用了 CORDIC 演算法中不同的模式(MODE)，來做頻率偏移角度上的估計，與補償時向量的旋轉，架構上也嘗試著犧牲一些效能來節省面積。

而在自動增益控制(Auto Gain Control)上，由於在數位上只能做功率的偵測，並回授給外面的類比放大器元件，所以我們只針對這部分做模擬的分析，並說明在頻率交錯(Tone Interleaving)上估測功率的效能，會比沒有做頻率交錯的效能來得好。而在整個系統加上 AGC 模擬方面，我們故意讓通道參數放大五倍與縮小五倍，來測試我們 AGC 對整個系統的影響。

最後的通道估測(Channel Estimation)運作在頻域之上，在此除了解釋，隨著天線數的增加，通道估計的效能也會越來越差，而可以以內插(Interpolation)的方式來增加效能，根據模擬的結果，在 N_t 與 N_r 分別在 2, 3 與 4 時，BER 在 10^{-1} 上，平均可以提高 1.5 至 2dB 的效能。

隨著 MIMO-OFDM 的發展，由於天線數的增加，演算法變複雜的情況下，計算量與硬體複雜度方面都大大的提高了，也造成功率節省上的不易。使用者對

於頻寬的需求是無止盡的，但我相信未來的目標除提高速度與頻寬之外，如何結省成本，將是這個產品是否能在這個市場上長久生存的重要關鍵。



參考文獻

- [1] IEEE Std 802.11®-1999 (Reaff 2003)
Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)
Specifications
- [2] Syed (Aon) Mujataba, IEEE802.11-04-0889, “TGn Sync Proposal Technical Specification”.
- [3] Juha Heiskala, and John Terry, “OFDM Wireless LANs,”2001.
- [4] Stefan H. Muller-Weinfurtner, “On the Optimality of Metrics for Coarse Frame Synchronization on OFDM: A Comparison,” 1998 IEEE.
- [5] Moose, P.H.,” A technique for orthogonal frequency division multiplexing frequency offset correction,” Communications, IEEE Transactions on Volume 42, Issue 10, Oct. 1994 Page(s):2908 – 2914.
- [6] van de Beek, J.J.; Sandell, M.; Borjesson, P.O.” ML estimation of time and frequency offset in OFDM systems,” Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on] Volume 45, Issue 7, July 1997 Page(s):1800 – 1805
- [7] Allert van Zelst, and Tim C. W. Schenk.”Implementation of a MIMO OFDM-Based Wireless,” IEEE Transaction on Signal Processing, VOL.52,No.2,February 2004.
- [8] Ray Andraka. “A survey of CORDIC algorithms for FPGA based computers”
- [9] Ken Turkowski, Apple Computer “ Fixed-Point Trigonometry with CORDIC Iterations”, Jan 17,1990.
- [10] Weinstein, S, Ebert, P. “Data Transmission by Frequency-Division Multiplexing Using the Discrete Fourier Transform,” Communications, IEEE Transactions on [legacy, pre - 1988]Volume 19, Issue 5, Oct 1971 Page(s):628 - 634

- [11] James W. Cooley and John W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comp., vol. 19, pp.297-301, April 1965.
- [12] Alan V. Oppenheim, Ronald W. Schaffer, and John R. Buck, "Discrete-time signal processing", second edition, PRENTICE HALL SIGNAL PROCESSING SERIES, 1989, 1999.
- [13] J. G. and Hannu Tenhunen, "Efficient VLSI implementation of radix-8 FFT algorithm," IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp.468-471, 1999.
- [14] Filippo Tosato and Paola Bisaglia, "Simplified Soft-Output Demapper for Binary Interleaved COFDM with Application to HIPERLAN/2", 2002 IEEE.



簡歷

姓 名：莊秉卓

性 別：男

出生日期：民國 70 年 6 月 13 日

出生地：高雄市

學 歷：

高雄市立五權國小 (1987.9~1993.6)

高雄市立五福國中 (1993.9~1996.6)

高雄市立中學 (1996.9~1999.6)

國立交通大學電信系 (1999.9~2003.6)

國立交通大學電信工程研究所碩士班(2003.9~2005.7)

公元 2005 年 7 月獲得碩士學位

