

國立交通大學

電信工程研究所

碩士論文

無合成器之 LINC-OFDM 系統:硬體設計與 FPGA 實現

A Combinerless LINC-OFDM System:
Hardware Design and FPGA Implementation

A large, light blue watermark of the Tsinghua University logo is centered behind the English title. The logo is circular with a gear-like border and contains the university's name in Chinese characters and the year 1896.

研究生：林家彰

指導教授：吳文榕 博士

中華民國 102 年 7 月

無合成器之 LINC-OFDM 系統:硬體設計與 FPGA 實現

A Combinerless LINC-OFDM System:
Hardware Design and FPGA Implementation

研究生：林家彰

Student : Chia-Chang Lin

指導教授：吳文榕 博士

Advisor : Dr. Wen-Rong Wu



A Thesis

Submitted to Institute of Communications Engineering
College of Electrical and Computer Engineering

National Chiao Tung University

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science

In

Communications Engineering

July 2013

Hsinchu, Taiwan, Republic of China

無合成器之 LINC-OFDM 系統: 硬體設計與 FPGA 實現

研究生：林家彰

指導教授：吳文榕 博士

國立交通大學

電信工程研究所碩士論文

摘要

正交分頻多工(Orthogonal Frequency Division Multiplexing, OFDM)系統的傳送訊號有很高的 PAPR(peak-to-average power ratio, PAPR)值, 因此其功率放大器(power amplifier, PA)需有較大的線性放大區域使系統能正常操作, 造成 PA 為射頻(radiofrequency, RF)電路中最消耗功率的元件。非線性元件之線性放大(Linear amplification with nonlinear component, LINC)技術為解決此問題的方法之一, LINC 系統可使用效率較好的非線性放大器進行線性放大。然而, 在 LINC 系統中有一關鍵的元件即功率合成器(power combiner), 其設計及實現有很高的難度, 為了克服此問題有研究者提出無合成器之 LINC 系統。本篇論文利用現場可程式規劃閘陣列(Field Programmable Gate Array, FPGA)實現無合成器之 LINC-OFDM 系統。我們先為無合成器之 LINC-OFDM 系統傳送機與接收機設計低複雜度的架構。接著我們以硬體描述語言 VHDL 實現傳收機之運作模式, 並以暫存器轉移層級(register transfer level, RTL)模擬驗證設計的正確性。產生的電路再進一步放置和繞線(place and route), 並下載至 FPGA 平台。最後將 LINC-OFDM 訊號透過射頻(Radio Frequency, RF)傳出與收回, 確認我們的設計可以正常運作。

A Combinerless LINC-OFDM Systems: Hardware Design and FPGA Implementation

Student: Chia-Chang Lin

Advisor: Dr. Wen-Rong Wu

Institute of Communications Engineering
National Chiao Tung University

Abstract

It is well known that the power amplifier (PA) of OFDM systems must operate in a wide linear region to cope with the high PAPR problem, making it the most power-hungry device in the RF circuit. The linear-amplification-with-nonlinear-component (LINC) technique, allowing the use of nonlinear PAs, has been developed to solve the problem. However, a critical component in LINC transmitter, named power combiner, is difficult to design and implement. A combinerless LINC system is recently proposed to avoid the problem. This thesis considers the implementation of a combinerless LINC-OFDM system with FPGA. We first design low-complexity architectures for a combinerless LINC-OFDM transmitter and receiver. Then, we implement the operations of the transceiver with the VHDL and conduct register transfer level (RTL) simulation to verify the correctness of the design. The resultant circuit is further placed and routed, and downloaded to a FPGA platform. Finally, experiments are conducted to transmit and receive LINC-OFDM signal through radio frequency (RF), confirming that our design can function properly.

誌謝

本篇論文得以順利完成，首先我要感謝我的指導教授吳文榕博士，在研究期間細心耐心的指導我們做研究，從老師身上學習到了很多做研究的方法與態度，讓我在研究期間獲益良多。也謝謝口試委員蘇育德教授、李大嵩教授、吳仁銘教授所提供之寶貴意見，讓本篇論文更加完整。

其次，要感謝寬頻通訊與訊號處理實驗室的鈞陶學長、勝隆學長、汀華學姊和碩二的同儕運凱、偉豪、侑君、萊恩，在研究及課業學習上不吝指導與鼓勵；同時也感謝實驗室學弟妹柏豪、佳瑩、家妤、軒平的幫忙，讓這兩年的碩士生活精彩難忘。

最後我要感謝我的父母與哥哥，有你們的關心支持與栽培照顧，讓我在研究期間無後顧之憂，得以順利完成研究。

感謝這一路上所以給予過幫助以及鼓勵的人。



目錄

摘要	i
誌謝	iii
目錄	iv
表目錄	vi
圖目錄	vi
一、簡介	- 1 -
二、LINC 系統介紹	- 3 -
2.1 調變訊號在非線性功率放大器下的失真現象	- 3 -
2.2 傳統 LINC 系統	- 4 -
2.2.1 LINC 系統之原理與架構	- 4 -
2.2.2 LINC 訊號拆解	- 6 -
2.3 無合成器之 LINC 系統	- 9 -
2.4 偵測方法	- 10 -
2.4.1 強制歸零(Zero forcing, ZF)等化器	- 10 -
2.4.2 修正 ZF 等化器	- 11 -
三、LINC-OFDM 傳送機與接收機設計	- 13 -
3.1 設計流程與傳收機架構	- 13 -
3.2 重要功能區塊	- 17 -
3.2.1 CORDIC(Coordinate Rotation Digital Computer)演算法	- 17 -
3.2.2 快速傅立葉轉換(Fast Fourier transform, FFT)	- 21 -
3.3 傳送機設計	- 22 -
3.3.1 提高取樣(Up-Sampling)	- 22 -
3.3.2 振幅截切(Amplitude Clipping)	- 25 -
3.3.3 訊號拆解器(Signal Component Separator, SCS)	- 28 -
3.3.4 前導訊號(Preamble)	- 31 -
3.4 接收機設計	- 32 -
3.4.1 降低取樣(Down-Sampling)	- 32 -
3.4.2 封包偵測(Packet Detection)	- 33 -

3.4.3 符元起點估計(Symbol Timing Estimation)	- 35 -
3.4.4 頻率偏移估計(Frequency Offset Estimation)	- 36 -
3.4.5 通道響應估測(Channel Estimation).....	- 39 -
3.4.6 MZF 偵測(MZF Detection).....	- 40 -
四、模擬	- 42 -
4.1 系統效能模擬	- 42 -
4.2 硬體設計模擬	- 47 -
五、FPGA 實現與測量	- 53 -
5.1 平台介紹	- 53 -
5.2 FPGA 實現.....	- 55 -
5.2.1 LINC-OFDM 傳送機實現.....	- 55 -
5.2.2 LINC-OFDM 接收機實現.....	- 60 -
5.3 測量結果	- 64 -
六、結論	- 70 -
參考文獻	- 72 -



表目錄

表 3.1 二階多項式逼近：係數表	- 30 -
表 4.1 LINC-OFDM 系統模擬環境參數	- 42 -

圖目錄

圖 2.1 LINC 系統架構圖	- 4 -
圖 2.2 Hybrid power combiner	- 5 -
圖 2.3 Chireix power combiner	- 6 -
圖 2.4 訊號拆解示意圖	- 7 -
圖 2.5 IQ 法訊號拆解圖	- 8 -
圖 2.6 2X1 LINC 系統架構	- 9 -
圖 2.7 2X2 LINC 系統架構	- 9 -
圖 3.1 設計流程圖	- 14 -
圖 3.2 傳送機架構方塊圖	- 15 -
圖 3.3 接收機架構方塊圖	- 16 -
圖 3.4 遞迴 CORDIC 架構圖	- 20 -
圖 3.5 平行 CORDIC 架構圖	- 20 -
圖 3.6 Xilinx FFT v7.1 模組	- 21 -
圖 3.7 Xilinx FFT v7.1 操作時序圖	- 22 -
圖 3.8 傳送機架構方塊圖	- 22 -
圖 3.9 時域平方根升餘弦濾波器	- 23 -
圖 3.10 頻域平方根升餘弦濾波器	- 23 -
圖 3.11 (a) 直接型式，(b) 轉置型式，(c) 混合形式	- 24 -
圖 3.12 多項濾波器概念圖	- 25 -
圖 3.13 多項濾波器架構圖	- 25 -
圖 3.14 振幅縮減器方塊圖	- 26 -
圖 3.15 範圍(Segment)判定區塊架構圖	- 27 -
圖 3.16 二分法線性逼近架構圖	- 27 -
圖 3.17 SCS:相位法方塊圖	- 28 -

圖 3.18 SCS:IQ 法方塊圖	- 28 -
圖 3.19 OFDM 符元之振幅統計分析	- 29 -
圖 3.20 SCS 錯誤向量曲線圖	- 30 -
圖 3.21 錯誤因子內插法架構圖	- 30 -
圖 3.22 SCS :改良後的 IQ 法方塊圖	- 31 -
圖 3.23 前導訊號	- 31 -
圖 3.24 接收機架構方塊圖	- 32 -
圖 3.25 降低取樣架構圖	- 33 -
圖 3.26 封包偵測基本架構圖	- 34 -
圖 3.27 移動窗戶架構圖	- 34 -
圖 3.28 改良後之封包偵測架構圖	- 35 -
圖 3.29 符元起點估計架構圖	- 36 -
圖 3.30 頻率偏移補償方塊圖	- 38 -
圖 3.31 Coarse/Fine Estimation 硬體架構圖	- 38 -
圖 3.32 Angle Accumulator 架構圖	- 39 -
圖 3.33 通道響應估測架構圖	- 40 -
圖 3.34 訊號等化架構圖	- 40 -
圖 4.1 DMA 架構圖	- 43 -
圖 4.2 LINC-OFDM 系統 ZF 與 MZF 等化器效能(PER)	- 44 -
圖 4.3 系統封包遺失率	- 44 -
圖 4.4 LINC-OFDM 系統 ZF 與 MZF 等化器效能(SER)	- 45 -
圖 4.5 MSE of CFO Estimation	- 46 -
圖 4.6 LINC-OFDM 系統效能: 頻率偏移效應	- 46 -
圖 4.7 MSE of CORDIC Vector mode	- 47 -
圖 4.8 MSE of CORDIC Rotation mode	- 48 -
圖 4.9 LINC-OFDM 系統效能: Amplitude Clipping	- 49 -
圖 4.10 傳送端原始訊號星座圖	- 49 -
圖 4.11 傳送端訊號縮減星座圖(浮點數運算)	- 50 -
圖 4.12 傳送端訊號縮減星座圖(定點數運算)	- 50 -
圖 4.13 錯誤訊號之訊號與近似誤差比	- 51 -
圖 4.14 理想浮點數 SCS 輸出	- 51 -
圖 4.15 近似定點數 SCS 輸出	- 51 -
圖 4.16 LINC-OFDM (floating-point v.s. fixed-point)	- 52 -

圖 5.1 完整平台	- 53 -
圖 5.2 連結介面	- 53 -
圖 5.3 FPGA 開發版	- 54 -
圖 5.4 RF 模組	- 55 -
圖 5.5 傳送機控制狀態	- 56 -
圖 5.6 傳送機 RTL 架構圖	- 57 -
圖 5.7 傳送機 mapping-report	- 58 -
圖 5.8 傳送機 timing-report	- 59 -
圖 5.9 傳送機 static timing	- 59 -
圖 5.10 傳送機 RTL 模擬測試	- 59 -
圖 5.11 RTL 模擬: 傳送機	- 60 -
圖 5.12 接收機 RTL 架構圖	- 61 -
圖 5.13 接收機 mapping-report	- 62 -
圖 5.14 接收機 timing-report	- 63 -
圖 5.15 接收機 static timing	- 63 -
圖 5.16 ping-pong buffering	- 63 -
圖 5.17 接收機 RTL 模擬測試	- 64 -
圖 5.18 RTL 模擬:接收機	- 64 -
圖 5.19 傳送機電纜測試	- 65 -
圖 5.20 傳送機天線測試	- 65 -
圖 5.21 傳送機 TX1 訊號(電纜)	- 66 -
圖 5.22 傳送機 TX2 訊號(電纜)	- 66 -
圖 5.23 傳送機訊號(天線)	- 67 -
圖 5.24 天線擺設位置	- 68 -
圖 5.25 接收機控制模組 flag 訊號	- 68 -
圖 5.26 接收機偵測訊號星狀圖	- 69 -



一、簡介

隨著無線多媒體的蓬勃發展，行動裝置對資料傳輸率的需求大幅增加，頻譜的使用效率對一個無線通訊系統而言變得格外重要，然而無線通訊系統的頻寬卻是有限的，為了有效的使用頻寬，新一代無線通訊系統大都採用正交分頻多工(Orthogonal Frequency Division Multiplexing, OFDM)的調變技術來取代原有的單載波系統，相較於單載波系統，OFDM 在有限的頻寬下，將資料同時載在多個子載波(sub-carrier)上，每個子載波彼此互相正交，頻譜使用效率較高。在 OFDM 調變技術中因為將通道分為多個子載波，因此可以將一頻率選擇性(frequency-selective)衰落通道轉換成多個平坦(flat)衰落通道的集合，在接收端僅需針對每個子載波做簡單的等化，降低接收端通道等化器之複雜度。

在無線通訊系統中，系統的功率消耗是很重要的設計考量，而在射頻(Radio Frequency, RF)電路中的功率放大器往往是最消耗功率的元件。功率放大器的操作點必須落在線性區才能使的訊號不會產生失真的現象，但功率放大器其線性放大區域有一定範圍，當訊號振幅大於某一範圍就進入飽和區，在飽和區訊號會因非線性放大而失真，為了不讓訊號失真，線性區域就需放大，因而造成功率放大器的損耗功率變大。

雖然 OFDM 可實現大量資料傳輸，但是因為 OFDM 調變技術是由多個正交的子載波訊號所疊加而成，使得 OFDM 發射訊號之峰均功率比(Peak-to-Average Power Ratio, PAPR)較單載波訊號大很多，高 PAPR 值會降低功率放大器的效能，使得訊號失真。若要避免訊號失真則需要高線性度的功率放大器，高線性度的功率放大器會使得損耗功率變大，相較於單載波系統，OFDM 的功耗高出許多，因此 OFDM 的功耗問題一直是個有待解決的難題。

LINC(Linear amplification with nonlinear component)[1]為解決高 PAPR 系統功耗問題的一種技術，它可以藉由高效率的非線性功率放大器來進行訊號的線性放大，使得發射器能同時達到高效率及高線性度的效果，LINC 的基本原理是將其輸入訊號分解成兩個

(或兩個以上)的固定波包(constant envelope)的訊號，因為是固定振幅所以這些訊號可以經由非線性功率放大器來放大，放大後的訊號再經由功率合成器(power combiner)加以合成，由天線發送出去。功率合成器是影響 LINC 系統整體效能一個很重要的因素。在 LINC 系統中常用的功率合成器形式有混合合成器[2]、Chireix 合成器[3]、Wilkinson 合成器[4]等架構。然而這些架構在實際的實現上都有缺點，造成 LINC 系統所減少的功率耗損不如預期的多。

LINC 系統最大的挑戰即為功率合成器的實現，Abdelaal[5]因此提出了一種合成的方法能夠完成避免使用合成器，主要的想法是讓兩個非線性功率放大器後的訊號在空中結合，使接收端收到的是結合後的 LINC 訊號。為了訊號的接收，Abdelaal 做了一個假設，他假設兩個傳送天線非常接近，因此兩個通道的響應可以視為相同，如此一來接收到的訊號即為兩個訊號相加，但是在實際的情況，通道是不會完全相同的。而在 OFDM 系統中，由於多載波的特性，使得在時域的通道響應的一些微差異，也會造成子載波通道的響應產生很大的不同。為了克服通道的差異，[6]提出利用空時編碼的作法將拆解後的兩個訊號分做兩個時間在兩根天線傳送，如此一來即可解決[5]中通道必須相同的限制，但其代價是傳輸率減半。[7]則針對無合成器 LINC-OFDM 系統，利用最大相似(Maximum Likelihood, ML)的偵測方法，克服通道差異所造成的效能損耗，[8]提出編碼的 LINC-OFDM 系統架構以及相關的偵測演算法，能改善[7]複雜度太高的缺點。本論文針對無合成器之 LINC-OFDM 系統，設計並實現傳送機與接收機，我們將對各個功能方塊的硬體架構提出構想，以硬體描述語言實現並使用電腦輔助設計軟體來模擬之。

本論文的章節編排如下，第二章介紹 LINC 系統，第三章為 LINC-OFDM 系統傳送機與接收機的設計，第四章為模擬結果，包含了浮點數系統模擬以及硬體設計的模擬，第五章為 FPGA 實現與測量結果，第六章結論。

二、LINC 系統介紹

2.1 調變訊號在非線性放大器下的失真現象

考慮一典型的通訊系統，其調變訊號一般可表示如下

$$x(t) = a(t) \cos[\omega_c t + \varphi(t)] \quad (2.1)$$

其中 $a(t)$ 為訊號波包(envelope)，若 $a(t)$ 為常數 A ，則為固定波包， $\omega_c(t)$ 為訊號頻率(frequency)， $\varphi(t)$ 為訊號相位(phase)。當調變訊號經過非線性功率放大器，變動波包與固定波包會得到不一樣的結果，我們可以使用三階多項式來近似一非線性功率放大器，假設其輸入輸出關係表示如下

$$v_{out} = G_1 v_{in}^3(t) + G_2 v_{in}^2(t) + G_3 v_{in}(t) + G_4 \quad (2.2)$$

其中 $v_{out}(t)$ 為輸出訊號， $v_{in}(t)$ 為輸入訊號， G_1 到 G_4 為放大器係數。將固定波包調變訊號 $x(t)$ 帶入此功率放大器，可得輸出訊號 $y(t)$ 如下

$$\begin{aligned} y(t) &= G_1 x^3(t) + G_2 x^2(t) + G_3 x(t) + G_4 \\ &= G_1 A^3 \cos^3(\omega_c t + \varphi(t)) + G_2 A^2 \cos^2(\omega_c t + \varphi(t)) + G_3 A \cos(\omega_c t + \varphi(t)) + G_4 \\ &= \frac{G_1 A^3}{4} (\cos(3\omega_c t + 3\varphi(t)) + \cos(\omega_c t + \varphi(t))) + \frac{G_2 A^2}{2} (1 + \cos(2\omega_c t + 2\varphi(t))) \\ &\quad + G_3 A \cos(\omega_c t + \varphi(t)) + G_4 \\ &\xrightarrow{LPF} \left(\frac{G_1 A^3}{4} + G_3 A \right) \cos(\omega_c t + \varphi(t)) + \frac{G_2 A^2}{2} + G_4 \end{aligned} \quad (2.3)$$

若此功率放大器的特性曲線通過原點，則 $G_4 = 0$ ，將經過功率放大器得到的訊號再通過一低通濾波器(low pass filter, LPF)，將高頻訊號濾掉，從(2.3)我們可以知道，輸出與輸入訊號為線性的關係，因此固定波包的訊號經過非線性的放大不會有失真。

若調變訊號為變動波包，則經過功率放大器的輸出 $y(t)$ 如下

$$\begin{aligned}
y(t) &= G_1 x^3(t) + G_2 x^2(t) + G_3 x(t) + G_4 \\
&= G_1 a^3(t) \cos^3(\omega_c t + \varphi(t)) + G_2 a^2(t) \cos^2(\omega_c t + \varphi(t)) + G_3 a(t) \cos(\omega_c t + \varphi(t)) + G_4 \\
&= \frac{G_1 a^3(t)}{4} (\cos(3\omega_c t + 3\varphi(t)) + \cos(\omega_c t + \varphi(t))) + \frac{G_2 a^2(t)}{2} (1 + \cos(2\omega_c t + 2\varphi(t))) \quad (2.4) \\
&\quad + G_3 a(t) \cos(\omega_c t + \varphi(t)) + G_4 \\
&\xrightarrow{LPF} \left(\frac{G_1 a^3(t)}{4} + G_3 a(t) \right) \cos(\omega_c t + \varphi(t)) + \frac{G_2 a^2(t)}{2} + G_4
\end{aligned}$$

從(2.4)我們可以知道，訊號經過低通濾波器後，輸出訊號包含 $a^3(t)$ 項，與輸入訊號呈現非線性的關係，造成輸出訊號失真。由以上兩種輸入訊號性質可知，固定波包的訊號可以使用非線性功率放大器，而變動波包的訊號則不可。

2.2 傳統 LINC 系統

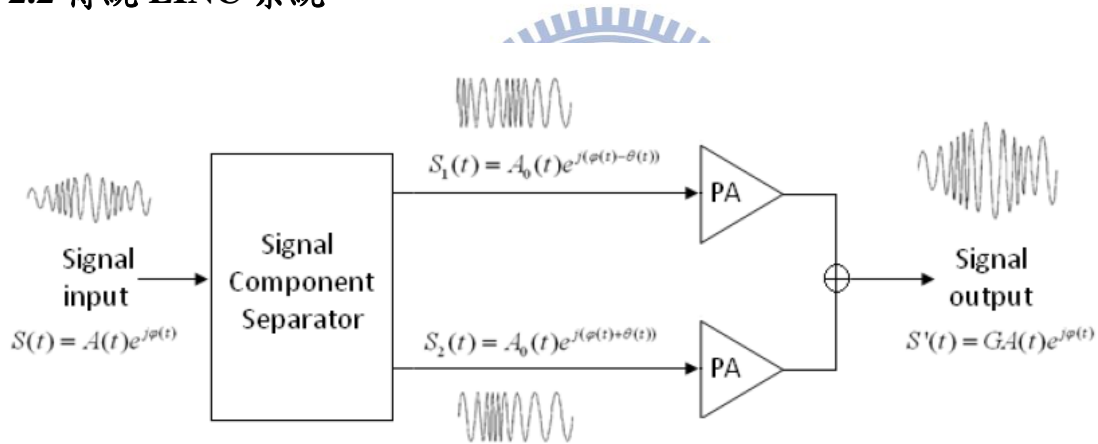


圖 2.1 LINC 系統架構圖

2.2.1 LINC 系統之原理與架構

傳統的 LINC 架構圖如圖 2.1 所示。基本原理[1]是將一個波包及相位都為時變(time varying)的輸入訊號，經過訊號分解器(Signal Component Separator, SCS)拆解成兩個固定波包之相位調製訊號，兩個固定波包的訊號之間僅存在一個相位差。因此這兩個相位調製訊號可以經由非線性功率放大器(Non-linear Power Amplifier)進行放大，最後兩個經過功率放大器的訊號再由功率合成器(Power Combiner)結合，重建出功率放大後具有時變波包之射頻調製訊號。其中輸出的訊號視為輸入訊號的線性放大結果。所使用的兩個功

率放大器為非線性放大器，利用非線性放大器而得到線性放大的輸出結果，可以有效提升系統的效能。

LINC 系統的架構主要分為四個部分，第一部分為訊號分解器，將輸入的基頻訊號拆解成兩個具有固定波包性質之相位調製訊號。第二部分是對相位調製訊號做調變，將原本拆解後基頻(baseband)或中頻(IF)的訊號調變至射頻(RF)訊號。第三部分為功率放大器，傳統的功率放大器是線性，但是效能較差，由於第一部分已將訊號拆解為固定波包，所以在此可以使用非線性的功率放大器來提升整個系統的功率效能。第四部份為功率合成器，將功率放大器的輸出由射頻電路做相加以重建訊號，若直接將兩路功率合成器的輸出做連接，當其中一路的電壓改變時，造成另一路所看到的負載阻抗(load impedance)隨之改變，因此兩路訊號必須能正確地相互抵銷，才得以重建出輸入的訊號，以下將介紹不同的功率合成器。

第一種功率合成器為複合式功率混合器(Hybrid power combiner)[2]，如圖 2.2 所示，此類型混合器讓負載阻抗維持固定，分隔(isolate)兩個功率放大器。將兩個相位調製訊號間的不同項合成至天線輸出，而相差項傳送至終端的負載電阻器(load resistor)以熱能的形式釋放。雖然此混合器可以保證輸出訊號的振幅有良好的線性度，但是因為部分能量的損失而降低了功率效能，此類型功率合成器稱為有耗能(lossy)合成器。

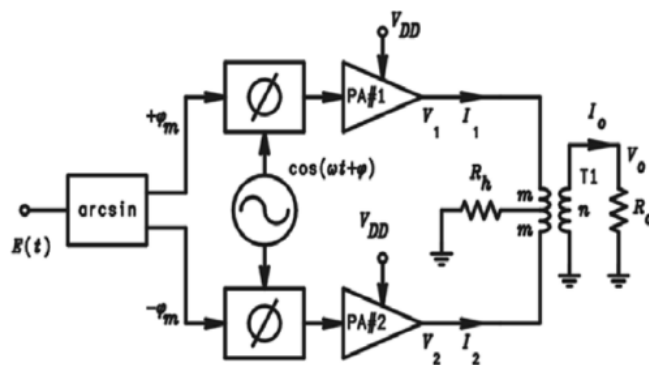


圖 2.2 Hybrid power combiner

第二種功率合成器為 Chireix power combiner[3]，如圖 2.3 所示，在合成前加入並聯電抗(shunt reactance)用來避免在特定振幅發生漏電抗(leakage reactance)的情形。此機制可以提升功率效能，又稱無耗能(lossless)合成器。但是此類型的功率合成器的兩個輸入沒有隔離電阻，會造成耦合(couple)，使得輸出訊號線性度較複合式混合器來得差。

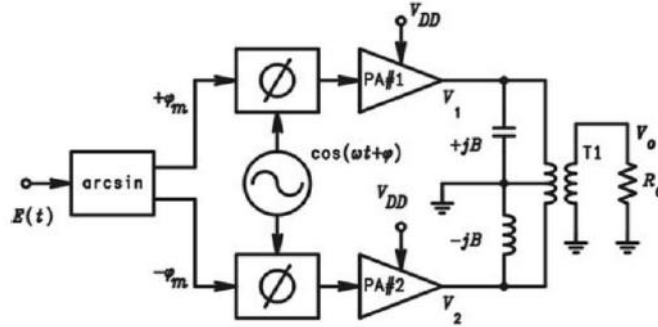


圖 2.3 Chireix power combiner

雖然上述的幾種合成器各有其優點，但仍然存在不可忽視的缺陷例如電路損耗(circuit losses)以及不匹配損耗(mismatch losses)的問題，要設計出高功率效能以及高線性度的合成器仍然是一個困難的課題。因此，本篇論文討論使用無合成器(combinerless)的 LINC 系統來解決合成器的問題。

2.2.2 LINC 訊號拆解

將 LINC 訊號分解器的輸入訊號假設為一實數訊號如下

$$s(t) = A(t) \cos(\omega_c t) \quad (2.5)$$

其中波包 $A(t) \geq 0$ ，將 $A(t)$ 以下式代換

$$A(t) = A_m \sin \varphi(t) \quad (2.6)$$

$$\varphi(t) = \sin^{-1} \left(\frac{A(t)}{A_m} \right) \quad (2.7)$$

其中 A_m 為 $A(t)$ 的最大值，因此決定了 $\varphi(t)$ ，代換之後可拆解如下

$$\begin{aligned}
s(t) &= \frac{A_m}{2} [\sin(\omega_c t + \varphi(t)) - \sin(\omega_c t - \varphi(t))] \\
&= s_1(t) - s_2(t) \\
s_1(t) &= \frac{A_m}{2} \sin(\omega_c t + \varphi(t)) \\
s_2(t) &= \frac{A_m}{2} \sin(\omega_c t - \varphi(t))
\end{aligned} \tag{2.8}$$

經過拆解後的 $s_1(t)$ 及 $s_2(t)$ ，可以表示為固定波包的向量，並且在順時針及逆時針的方向分別旋轉一個角度 $\varphi(t)$ ，如圖 2.4 所示，因為訊號已拆解成兩固定波包訊號 $s_1(t)$ 及 $s_2(t)$ ，所以可以分別使用非線性放大器。假設放大器的放大增益為 G ，則輸出結果可表示如下

$$Gs_1(t) - Gs_2(t) = GA_m \sin \varphi(t) \cos(\omega_c t) = Gs(t) \tag{2.9}$$

如(2.9)所示，可以得到線性放大的結果。

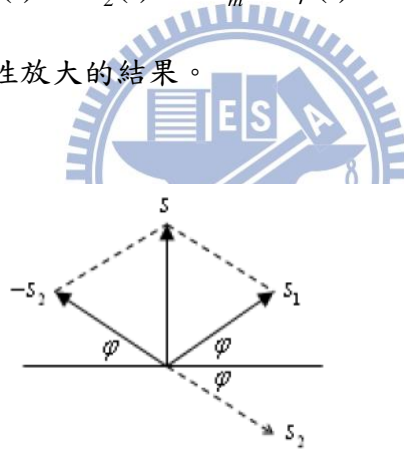


圖 2.4 訊號拆解示意圖

若將輸入訊號寫成複數訊號的通式如下

$$s(t) = A(t) \cos(\omega_c t + \theta(t)) \tag{2.10}$$

其中波包 $A(t) \geq 0$ ，將 $A(t)$ 利用(2.6)及(2.7)代換後可得

$$\begin{aligned}
s(t) &= \frac{A_m}{2} [\sin(\omega_c t + \theta(t) + \varphi(t)) - \sin(\omega_c t + \theta(t) - \varphi(t))] \\
&= s_1(t) - s_2(t) \\
s_1(t) &= \frac{A_m}{2} \sin(\omega_c t + \theta(t) + \varphi(t)) \\
s_2(t) &= \frac{A_m}{2} \sin(\omega_c t + \theta(t) - \varphi(t))
\end{aligned} \tag{2.11}$$

因為輸入為複數訊號，(2.10)之輸入訊號可以改寫如下

$$\begin{aligned} A(t) &= \sqrt{I^2(t) + Q^2(t)} \\ \theta(t) &= \tan^{-1} \frac{Q(t)}{I(t)} \end{aligned} \quad (2.12)$$

上式中 $I(t)$ 為訊號的實部， $Q(t)$ 為訊號的虛部。在座標平面中，對任一複數基頻訊號 $s(t) = I(t) + jQ(t)$ ，一樣可拆解為兩個複數訊號 $s_1(t)$ 和 $s_2(t)$ ，兩個複數訊號有相同的振幅 $\frac{A_m}{2}$ ， $\frac{A_m}{2}$ 為一個常數值，且 $s(t) = s_1(t) + s_2(t)$ 。拆解訊號的方法有相位調變(phase modulation)法以及 IQ(in phase/quadrature phase)法。相位調變法即利用(2.12)求得，由(2.7)求得相位 $\varphi(t)$ ，拆解後兩訊號的相位為 $\theta(t) + \varphi(t)$ 和 $\theta(t) - \varphi(t)$ 。而 IQ 法是先計算錯誤訊號 $e(t)$ ，再求 $s_1(t)$ 和 $s_2(t)$ ， $e(t)$ 、 $s_1(t)$ 、 $s_2(t)$ 之間的關係如圖 2.5 所示，推導結果如下

$$e(t) = js(t) \sqrt{\frac{A_m^2}{|s(t)|^2} - 1} \quad (2.13)$$

拆解後的兩訊號可以表示為

$$\begin{aligned} s_1(t) &= I_1(t) + jQ_1(t) = \frac{1}{2}(s(t) + e(t)) \\ s_2(t) &= I_2(t) + jQ_2(t) = \frac{1}{2}(s(t) - e(t)) \end{aligned} \quad (2.14)$$

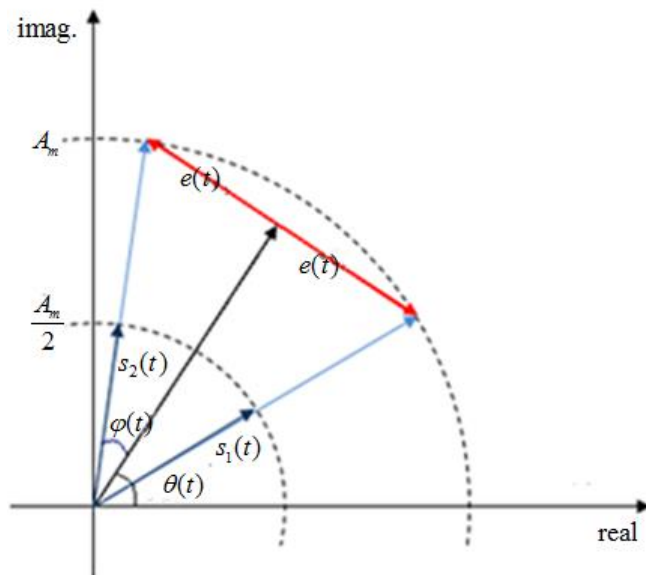


圖 2.5 IQ 法訊號拆解圖

2.3 無合成器之 LINC 系統

傳統的 LINC 系統是將輸入訊號拆解成兩個固定波包的訊號以克服線性功率放大器的低效能度問題，因此 LINC 使用兩個非線性功率放大器以達到較高效能，得到線性放大的輸出。但 LINC 系統的問題在於高效能的功率合成器不容易實現。因此[5]提出無功率合成器的方法，在此稱之為無合成器之 LINC 系統。此系統將功率放大器的輸出視為傳送端，兩個固定波包訊號透過兩根天線來傳送，經過兩個通道由接收端接收，因此兩個拆解後的訊號在空間中自然合成。假設兩根天線距離非常靠近，兩個通道可視為相同，接收到的訊號即可視為功率放大器輸出的兩個訊號經合成後再經過通道，如此可以免去傳統 LINC 系統在功率合成器設計上的問題。在無合成器的系統中，傳送之前必須要加入濾波器，因為兩個固定波包的訊號不一定符合傳送頻帶的限制。

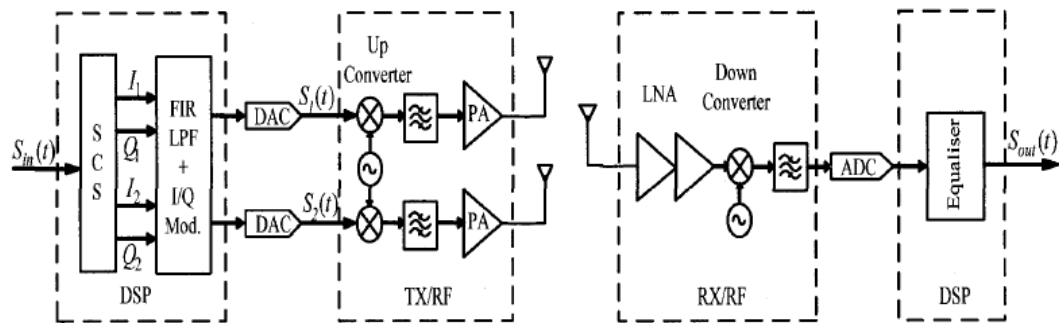


圖 2.6 2x1 LINC 系統架構

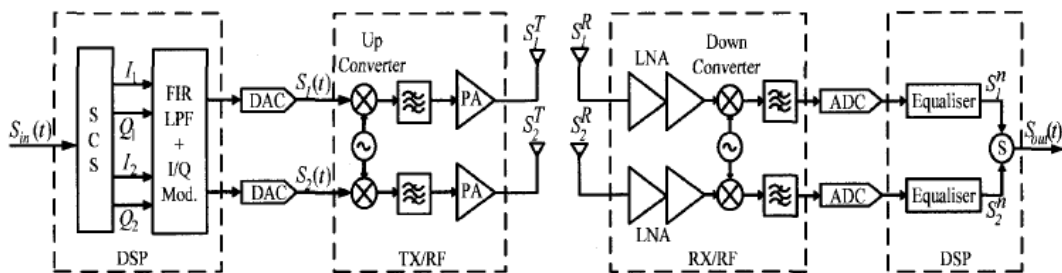


圖 2.7 2x2 LINC 系統架構

圖 2.6 為 2x1 無合成器 LINC 系統架構，數位訊號模組(Digital Signal Processing, DSP)將傳送的訊號拆解後經過濾波器/調變器，再經過功率放大器放大後傳送。[5]也提出極化天線的系統架構如圖 2.7 所示，接收端為兩根天線，形成一類似 2x2 的 MIMO 系統，

如此一來可以避免兩根傳送天線之間的干擾，並且可以允許兩個通道不必相同。[5]的模擬結果證實 2×1 和 2×2 無合成器 LINC 的系統效能較比起傳統 LINC 系統有大幅的提升。

2.4 偵測方法

2.4.1 強制歸零(Zero Forcing, ZF)等化器

令 \mathbf{s} 表示一 OFDM 符元在時域(time domain)上的訊號，如前所述，LINC 系統將傳送的訊號 \mathbf{s} 在時域拆解成 \mathbf{s}_1 及 \mathbf{s}_2 如下

$$\begin{aligned}\mathbf{s}_1 &= \frac{1}{2}(\mathbf{s} + \mathbf{e}) \\ \mathbf{s}_2 &= \frac{1}{2}(\mathbf{s} - \mathbf{e})\end{aligned}\quad (2.15)$$

$$\mathbf{e} = \left[js(0) \sqrt{\frac{A_m}{|s(0)|^2} - 1}, \dots, js(N-1) \sqrt{\frac{A_m}{|s(N-1)|^2} - 1} \right]^T \quad (2.16)$$

其中 N 為子載波個數，在 LINC-OFDM 系統中，因為 OFDM 的符元定義在頻域(frequency domain)，因此在接收端會將接收到的訊號轉至頻域，令 $\tilde{s}_1(p)$ 及 $\tilde{s}_2(p)$ 為一子載波的兩個頻域訊號， $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ 為相對應的通道頻率響應， $\tilde{y}(p)$ 為接收訊號， p 為載波的指標(index)，則接收訊號表示如下

$$\tilde{y}(p) = \tilde{h}_1(p)\tilde{s}_1(p) + \tilde{h}_2(p)\tilde{s}_2(p) + n(p) \quad (2.17)$$

等化器的目的是求出原始的傳送訊號 $\tilde{s}(p) = \tilde{s}_1(p) + \tilde{s}_2(p)$ ，ZF 等化器目的是要完全補償通道效應，在此我們假設 $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ 很相近但不完全一樣，因此我們取 $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ 的平均值做為等效的通道響應 $\tilde{h}_{ZF}(p) \equiv (\tilde{h}_1(p) + \tilde{h}_2(p))/2$ ，利用此通道響應來作等化可得等化後的訊號如下

$$\hat{\tilde{s}}(p) = \left(\tilde{h}_{ZF}(p) \right)^{-1} \tilde{y}(p) \quad (2.18)$$

ZF 等化器作通道補償因不考慮雜訊因素，所以等化後有雜訊放大的問題，另外因為 $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ 不完全相等，等化後會產生一干擾項，表示如下

$$\begin{aligned} \tilde{y}(p) = & \left(\tilde{h}_{ZF}(p) \right) \times (\tilde{s}_1(p) + \tilde{s}_2(p)) \\ & + \underbrace{\left(\frac{\tilde{h}_1(p) - \tilde{h}_2(p)}{2} \right) \times \tilde{s}_1(p) + \left(\frac{\tilde{h}_2(p) - \tilde{h}_1(p)}{2} \right) \times \tilde{s}_2(p)}_{\text{Interference}} + \tilde{n}(p) \end{aligned} \quad (2.19)$$

值得注意的是此干擾項與雜訊無關，拆開後的訊號轉至頻域後 $\tilde{s}_1(p)$ 及 $\tilde{s}_2(p)$ 可能有很大的振幅，意即 $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ 的差異即使很小也可能產生很大的干擾。

2.4.2 修正 ZF 等化器

為了避免(2.19)所提到的干擾項嚴重影響到接收訊號，[8]提出一修正的 ZF 等化器，將(2.16)代入(2.15)得到訊號在時域的表示式如下

$$\begin{aligned} \mathbf{s}_1 &= \frac{1}{2} (\mathbf{I} + j\mathbf{C}(\mathbf{s}))\mathbf{s} \\ \mathbf{s}_2 &= \frac{1}{2} (\mathbf{I} - j\mathbf{C}(\mathbf{s}))\mathbf{s} \end{aligned} \quad (2.20)$$

$$\mathbf{C}(\mathbf{s}) = \text{diag} \left[\left[\sqrt{\frac{A_m}{|s(0)|^2} - 1}, \dots, \sqrt{\frac{A_m}{|s(N-1)|^2} - 1} \right]^T \right]$$

其中 N 為子載波個數， $\mathbf{C}(\mathbf{s})$ 為一對角矩陣，將一個 OFDM 符元訊號表示成矩陣的型式，接收到的訊號(2.17)可以表示如下

$$\begin{aligned} \tilde{\mathbf{y}} &= \tilde{\mathbf{H}}_1 F \mathbf{s}_1 + \tilde{\mathbf{H}}_2 F \mathbf{s}_2 + \tilde{\mathbf{n}} \\ &= \underbrace{\frac{1}{2} (\tilde{\mathbf{H}}_1 + \tilde{\mathbf{H}}_2)}_{\text{signal}} \tilde{\mathbf{s}} + \underbrace{\frac{j}{2} (\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2) F \mathbf{C}(\mathbf{s}) F^H}_{\text{interference}} \tilde{\mathbf{s}} + \tilde{\mathbf{n}} \end{aligned} \quad (2.21)$$

(2.21)中 F 及 F^H 分別代表離散傅利葉轉換 (Discrete Fourier Transform, DFT) 和反離散傅利葉轉換 (Inverse Discrete Fourier Transform, IDFT)， $\tilde{\mathbf{H}}_1$ 和 $\tilde{\mathbf{H}}_2$ 為對角矩陣，對角線上的值為 $\tilde{h}_1(p)$ 及 $\tilde{h}_2(p)$ ，從(2.21)可得知，干擾項在兩個通道相同 $\tilde{\mathbf{H}}_1 = \tilde{\mathbf{H}}_2$ 時為 0，當 $\tilde{\mathbf{H}}_1 \neq \tilde{\mathbf{H}}_2$

時，干擾項會存在，因為 $\mathbf{C}(s)$ 項的值恆正，因此可得知(2.21)干擾項的平均值為正。當 $\tilde{\mathbf{H}}_1$ 與 $\tilde{\mathbf{H}}_2$ 是兩個高相關性通道時，(2.21)有高的訊號對干擾比值(Signal to Interference Ratio, SIR)。當 $\tilde{\mathbf{H}}_1$ 與 $\tilde{\mathbf{H}}_2$ 相關性降低時，(2.21)的訊號對干擾比值亦漸降低。若能將(2.21)干擾項的平均值扣除，可以讓干擾項變成零平均(zero-mean)訊號，則干擾項的影響將可降低。[8]的做法是使用一參數 $\mu > 0$ ，假設此參數可以近似 $\mathbf{C}(s)$ 的平均值，(2.21)可改寫如下

$$\begin{aligned}
\tilde{\mathbf{y}} &= \frac{1}{2}(\tilde{\mathbf{H}}_1 + \tilde{\mathbf{H}}_2)\tilde{\mathbf{s}} + \frac{j}{2}(\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2)F\mathbf{C}(s)F^H\tilde{\mathbf{s}} + \tilde{\mathbf{n}} \\
&= \frac{1}{2}(\tilde{\mathbf{H}}_1 + \tilde{\mathbf{H}}_2)\tilde{\mathbf{s}} + \frac{j}{2}(\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2)F(\mu\mathbf{I})F^H\tilde{\mathbf{s}} \\
&\quad + \frac{j}{2}(\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2)F(\mathbf{C}(s) - \mu\mathbf{I})F^H\tilde{\mathbf{s}} + \tilde{\mathbf{n}} \\
&= \underbrace{\left[\frac{1}{2}(\tilde{\mathbf{H}}_1 + \tilde{\mathbf{H}}_2) + \frac{j\mu}{2}(\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2) \right]}_{\text{new signal}}\tilde{\mathbf{s}} + \underbrace{\frac{j}{2}(\tilde{\mathbf{H}}_1 - \tilde{\mathbf{H}}_2)F(\mathbf{C}(s) - \mu\mathbf{I})F^H}_{\text{new interference}}\tilde{\mathbf{s}} + \tilde{\mathbf{n}}
\end{aligned} \tag{2.22}$$

我們將修正(modified)的 ZF 等化器縮寫為 MZF，因此定義 MZF 為

$$\tilde{h}_{MZF}(p) \equiv \frac{1}{2}(\tilde{h}_1(p) + \tilde{h}_2(p)) + \frac{j\mu}{2}(\tilde{h}_1(p) - \tilde{h}_2(p)) \tag{2.23}$$

以 MZF 等化後的訊號可表示如下

$$\hat{\tilde{s}}(p) = \left(\tilde{h}_{MZF}(p) \right)^{-1} y(p) \tag{2.24}$$

三、LINC-OFDM 傳送機與接收機設計

3.1 設計流程與傳收機架構

我們使用電腦輔助工具來模擬硬體實現，設計流程如圖 4-1 所示。設計流程主要分成硬體和軟體二部分，軟體使用 Matlab 模擬而硬體使用 ModelSim 及 Xilinx ISE 模擬。Matlab 模擬包含了浮點數(floating point)的模擬及定點數(fixed point)的模擬。浮點數的系統模擬可以檢查各功能方塊的正確性和設定參數，也可以評估系統的效能，而定點數的系統模擬，用來決定各個訊號所需要的表示位元數，並試著將位元降低至容許的效能，各個訊號表示位元數確定之後，便進入硬體實現模擬。

硬體實現模擬流程中，要先確定每個功能方塊的硬體架構，我們以硬體描述語言 VHDL 描述這些功能方塊，使用 Modelsim 或 Xilinx Isim 做硬體的行為模擬(Behavior simulation)，此一步驟會參照由 Matlab 定點數的系統模擬所產生的數據，來幫助驗證 VHDL 功能方塊的正確性。當硬體的行為模擬正確之後，使用 Xilinx ISE 14.2 加入 Spartan6 XC6SLX150 的 library 將 VHDL 程式碼轉為 RTL (register transfer level)，合成完之後再做轉換(translate)、對應(map)、布置及繞線(place and route)，接下來就可做時序模擬(timing simulation)，模擬電路所造成的邏輯延遲及繞線延遲是否達到我們要求的操作速度。在此便完成了所有的硬體模擬。

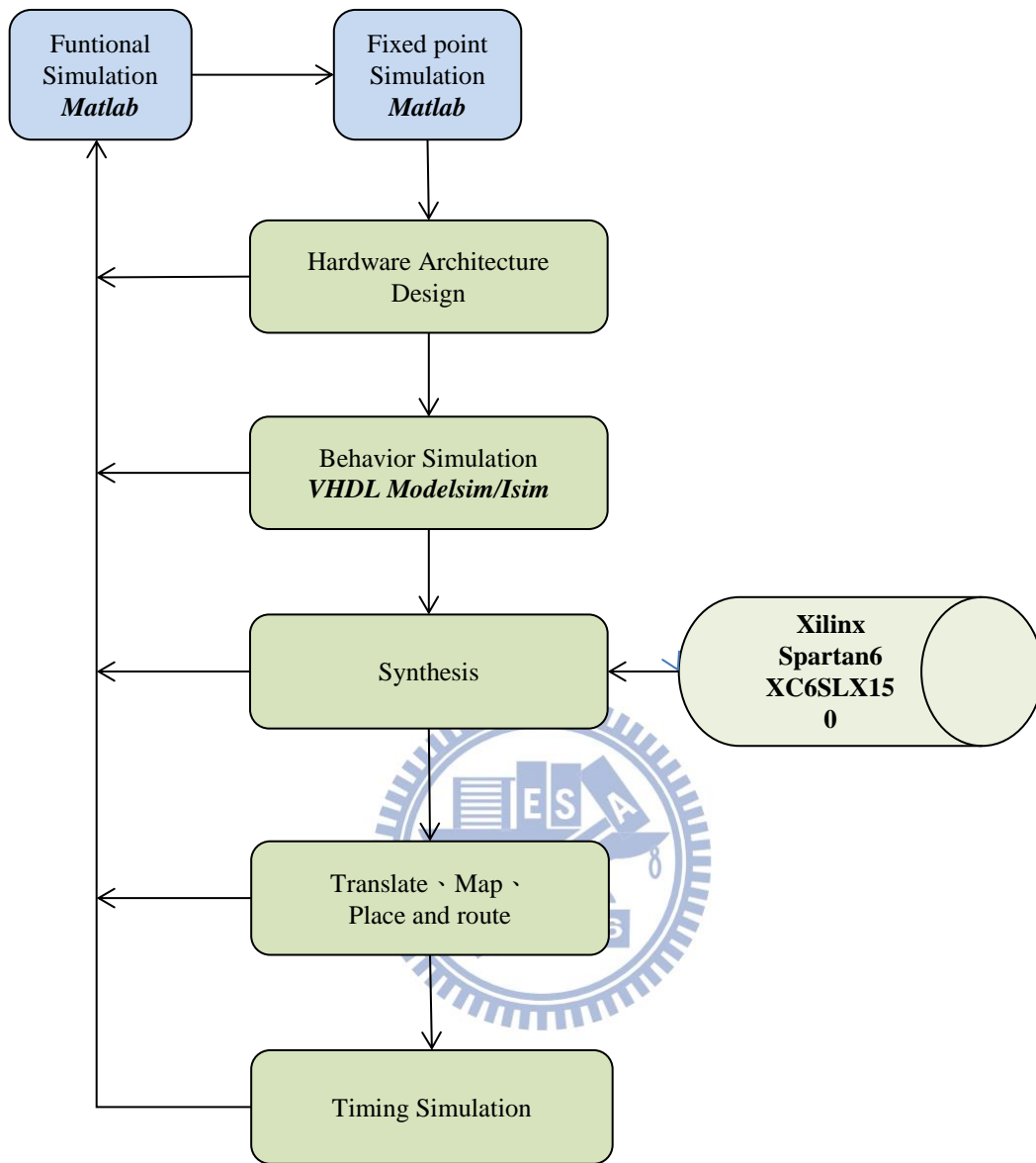


圖 3.1 設計流程圖

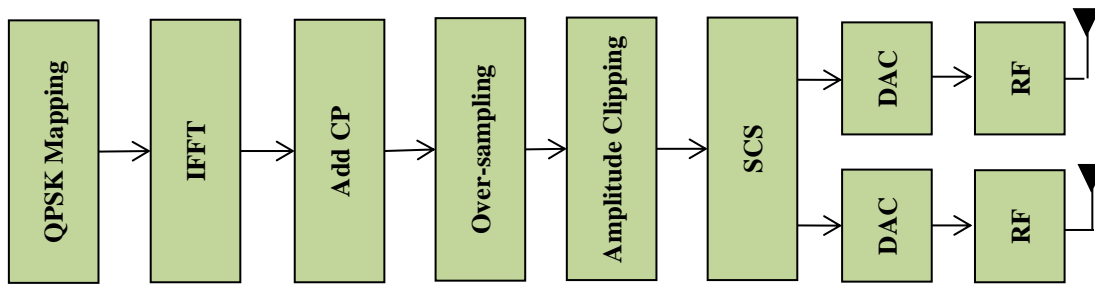


圖 3.2 傳送機架構方塊圖

傳送機架構如圖 3.2 所示，接下來會先對傳送機架構做簡述。一開始要將位元(bit)映射(mapping)到符元(symbol)，產生 OFDM 頻域的子載波訊號，接下來做 IFFT 運算，將訊號轉變為時域的 OFDM 符元，並在前端加上循環前綴(Cyclic Prefix, CP)，以避免符元間互相干擾(Inter Symbol Interference, ISI)。接著，透過將訊號補零(pad zero)和濾波(filtering)來提高取樣頻率。在訊號進入 SCS 之前，我們必須訂出一個訊號的最大振幅，讓訊號進入 SCS 運算後可正常運作，超過最大振幅即保留訊號相位並縮減至規範的最大振幅，如此也可以降低峰均功率比(Peak-to-Average Power Ratio, PAPR)，因此我們需要振幅縮減器。最後將訊號通過 SCS，將訊號由變動波包拆解成兩個固定波包訊號。由 SCS 輸出的訊號經過數位轉類比轉換器(Digital to Analog Converter, DAC)，然後再將基頻訊號載至射頻，由天線送出。

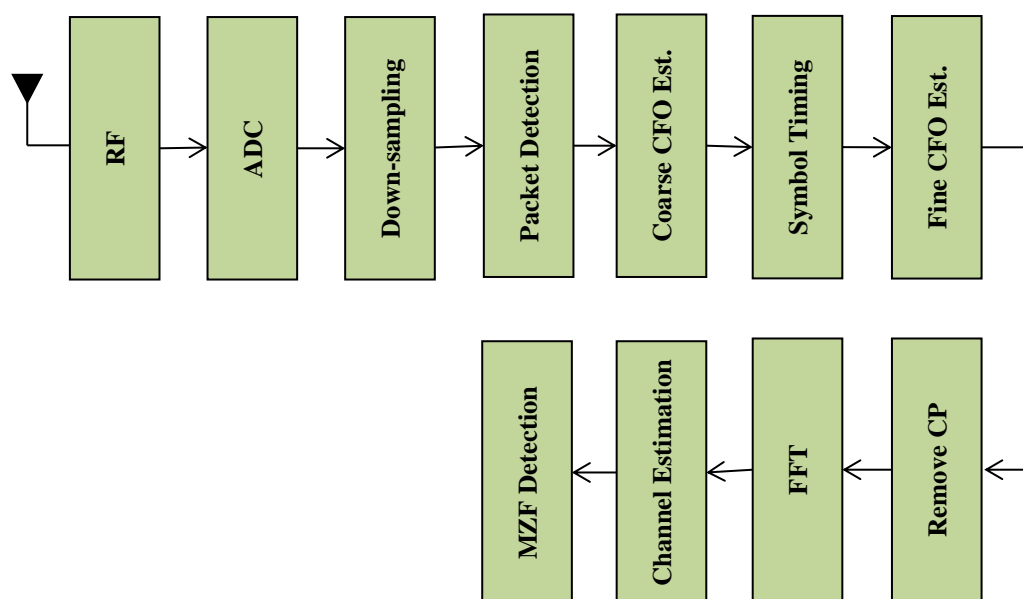


圖 3.3 接收機架構方塊圖

本篇論文的接收機主要針對訊號同步的問題做處理，架構流程參考 802.11a 的系統架構[9]，接下來會先對接收機架構做簡述。OFDM 技術解決了訊號在多重路徑通道 (multipath channel) 中，因通道產生的延遲所造成的困擾。但在通道效應的處理之前，我們需要先作時間和頻率的同步。時間同步除了封包偵測之外，還需要做符元時序的估計，確保我們將正確而不受其他符元干擾的取樣值做 FFT。而傳送機與接收機之間的頻率不匹配，會造成載波間的相互干擾 (Inter Carrier Interference, ICI)，為了保持載波間的正交性，需做頻率同步將偏移的頻率估計與補償。做完同步運算後接下來是利用 FFT 將訊號轉至頻域，以進行通道效應估測與訊號偵測，本論文的偵測方法使用了 2.4.2 所提到改良後的 MZF 偵測來實現。

接下來我們將介紹無合成器之 LINC-OFDM 傳收機(2x1)的硬體設計，章節 3.2 我們會先對傳送機與接收機皆有使用到的重要功能區塊做介紹，章節 3.3 介紹傳送機架構的硬體設計，章節 3.4 則介紹接收機架構的硬體設計。

3.2 重要功能區塊

3.2.1 CORDIC(Coordinate Rotation Digital Computer)演算法

CORDIC 是一種旋轉相關的低複雜度演算法[10]，它可以用來計算三角函數、乘法、除法等等。在本篇論文中，主要是拿來計算反正切函數(Arctangent)和相位的旋轉。此演算法利用反覆做角度的向量旋轉，最後以移位器和加法器來實現。CORDIC 演算法的運作，一般說來可以分為兩種模式，一種叫做旋轉模式(Rotation Mode)，另一種則是向量模式(Vector Mode)。這兩種模式在本論文的實現中都會用到。旋轉模式是將輸入向量旋轉一個給定的角度，向量模式則是將原本輸入的向量，反覆旋轉至 x 軸上，藉此得到原本向量的角度。

➤ 旋轉模式

在旋轉模式中，我們先讓角度累加器 z 的初始值為欲旋轉的角度，並在每次的疊代當中，試圖讓角度累加器的值為零。旋轉模式的數學式如下

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

where

$$d_i = -1 \text{ if } z_i < 0, d_i = +1 \text{ if } z_i > 0 \quad (3.1)$$

本論文中我們會利用旋轉模式來做為頻率補償及震幅縮減的旋轉。當疊代的次數夠多時，上式可轉變為

$$\begin{cases} x_n = A_n(x_0 \cos(z_0) - y_0 \sin(z_0)) \\ y_n = A_n(x_0 \sin(z_0) + y_0 \cos(z_0)) \\ z_n = 0 \\ A_n = \prod_n \sqrt{1 + 2^{-2i}} \end{cases} \quad (3.2)$$

➤ 向量模式

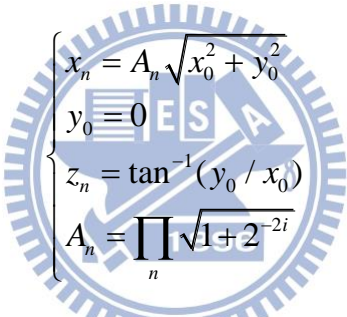
在向量模式中，CORDIC 試著將輸入訊號旋轉至 x 軸上，我們可以透過這樣的機制，得到原本輸入訊號的相位與振幅。與旋轉模式不一樣的地方在於疊代 y_i 的值會趨近於 0，因此會由 y_i 來決定旋轉的方向。向量模式的數學式如下，

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

where

$$d_i = +1 \text{ if } y_i < 0, d_i = -1 \text{ if } y_i > 0$$
(3.3)

當疊代的次數夠多時，上式可轉變為



$$\begin{cases} x_n = A_n \sqrt{x_0^2 + y_0^2} \\ y_n = 0 \\ z_n = \tan^{-1}(y_0 / x_0) \\ A_n = \prod_n \sqrt{1 + 2^{-2i}} \end{cases}$$
(3.4)

由(3.4)式中可知道，當我們讓 z_0 為零時，可以利用 z_n 來算出向量的角度。

不管是旋轉模式或向量模式，在每次的疊代中，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，最大旋轉角度的限制如下

$$-\tan(2^{-i}) \leq \text{rotating angle per iteration} \leq \tan(2^{-i})$$
(3.5)

也就是當 i 起始為 0 時，每次旋轉的角度會介於 $-\frac{\pi}{4}$ 與 $\frac{\pi}{4}$ 之間。當總旋轉角度超過最大能旋轉的值時，就會發生錯誤。因此我們必須知道最大可旋轉的角度為多少，假設每次疊代時的 d 都為 1，且 i 的起始值為 0，不斷相加的結果，可旋轉最大的角度為

$$\sum_{i=0}^{\infty} \tan^{-1}(2^{-i}) \cong 1.7433$$
(3.6)

也就是能旋轉的角度範圍在 -1.7433 與 1.7433 之間，大約在正負 100 度之間。若超過這

個角度，我們必須先旋轉 $-\pi$ 或 π 將旋轉角度限制在能運作的範圍之內，其公式如下

$$\begin{cases} x_0 = d_0 \cdot x_{-i} \\ y_0 = d_0 \cdot y_{-i} \\ z_0 = z_i \text{ if } d_0 = 1, \text{ or } z_{-i} - \text{sign}(y_0) \cdot \pi \text{ if } d_0 = -1 \end{cases} \quad (3.7)$$

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

其中 x_{-i} 、 y_{-i} 和 z_{-i} 表示初始值， $\text{sign}(\cdot)$ 代表該括號中的正負號。做完初步的旋轉之後，再把 x_0 、 y_0 、 z_0 與 d_0 帶入先前的公式即可。經過這個處理之後，總旋轉角度範圍擴大為 $-\pi$ 到 π 之間了。

另外，從(3.6)可看出，最大能旋轉的角度為 1.733，由於之後我們可以透過(3.7)預先旋轉角度 π ，所以最大能旋轉的角度為

$$\pi + 1.7433 = 4.8849 = 279.884^\circ \quad (3.8)$$

如果旋轉角度大於 4.8849，我們可以先扣掉 2π ，也就是將範圍限定在 $-\pi$ 與 π 之間，即可做正確的 CORDIC 運算。

CORDIC 演算法的架構[12]主要有兩種，遞迴(iterative)架構與平行(parallel)架構。遞迴 CORDIC 架構如圖 3.4 所示，每次遞迴的反正切函數值由唯讀記憶體(Read Only Memory, ROM)讀出，只需要使用同一組電路，較節省面積，但所需運算的時脈週期(clock period)隨遞迴次數增加而遞增，因此運算所需的時間較長為此架構的缺點。平行 CORDIC 架構如圖 3.5 所示，將所有的遞迴運算層級堆疊，Arctangent 值由 ROM 取出，對每個層級而言可視為一個常數，可在一個時脈週期完成運算，但是要考量到單一時脈周期的時間長短，且也耗費較多硬體面積。因此，我們希望取得兩種架構優點的平衡，將兩種架構結合使用，每次的遞迴操作二到三個層級的運算，在面積與運算時脈週期間取得平衡。

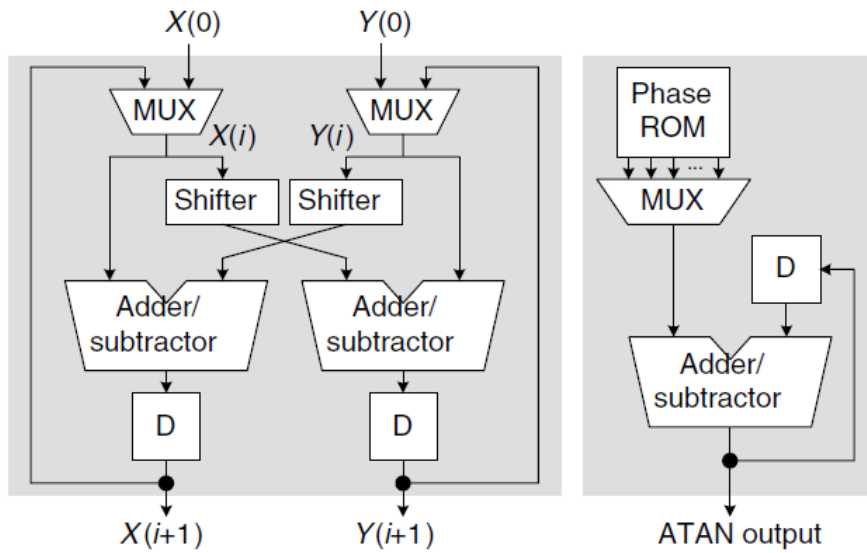


圖 3.4 遞迴 CORDIC 架構圖

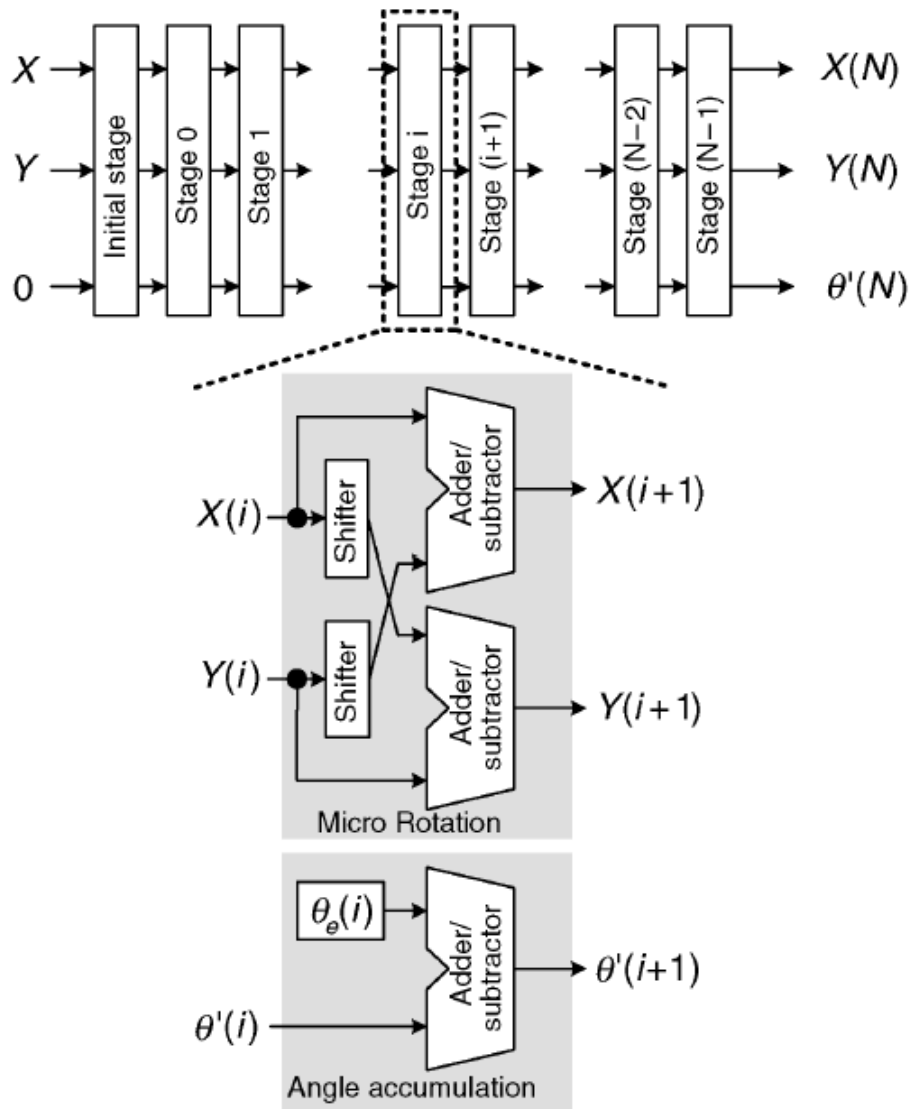


圖 3.5 平行 CORDIC 架構圖

3.2.2 快速傅立葉轉換(Fast Fourier transform, FFT)

本篇論文中，FFT 運算直接使用由 Xilinx IPCore 產生的模組，因此將參照 Xilinx 所提供的數據表(Data Sheet)[13-14]加以說明。

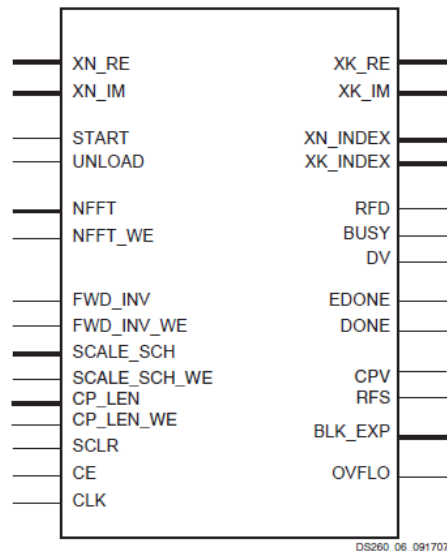


圖 3.6 Xilinx FFT v7.1 模組

圖 3.6 是在實際電路中所使用的 FFT 模組，接下來將對模組中一些重要的輸入與輸出訊號做說明。在輸入的訊號中，START 用來控制 FFT 運算的起始點，當此訊號升起 (rising) 時，開始輸入要運算的資料，XN_RE 為輸入的訊號實部，XN_IM 為輸入的訊號虛部，START 訊號升起只需維持一個時脈周期的時間，因此載完一個完整 OFDM 符元的資料後，會繼續等待下一次 START 訊號的升起，才會繼續讀取下一個 OFDM 符元來做 FFT 運算。運算過程中 BUSY 訊號會升起直到運算結束，當運算結束後，DONE 訊號會升起，此時可以將我們所需要的輸出資料取出，XK_RE 為輸入的訊號實部，XK_IM 為輸入的訊號虛部，由於後續的處理需要一些緩衝時間，因此我們先將這些資料儲存於自行建立的隨機存取記憶體(Random Access Memory, RAM)空間，以增加彈性做後續的處理。圖 3.7 為此模組的操作時序圖。

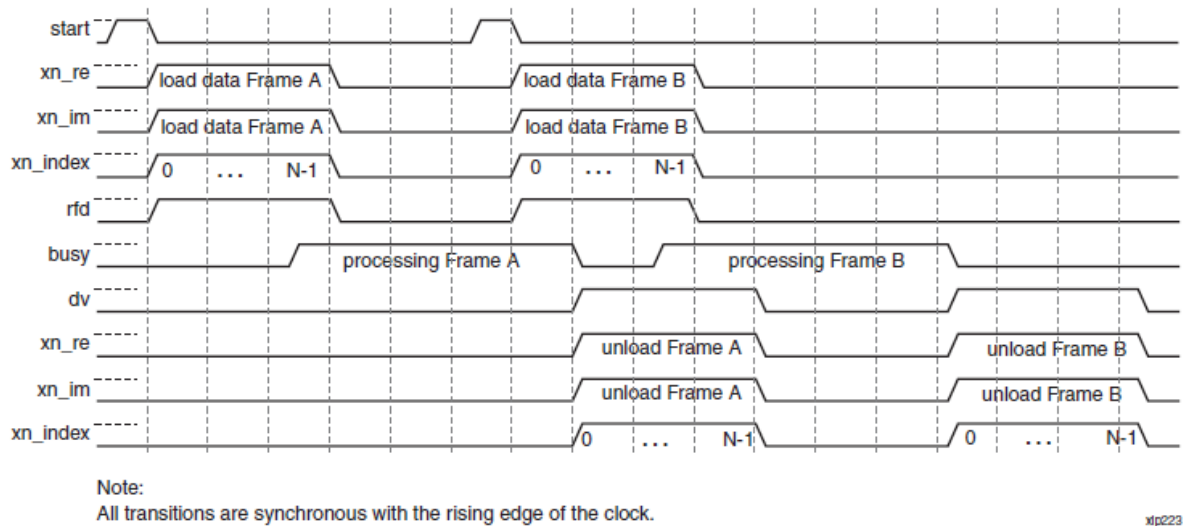


圖 3.7 Xilinx FFT v7.1 操作時序圖

3.3 傳送機設計

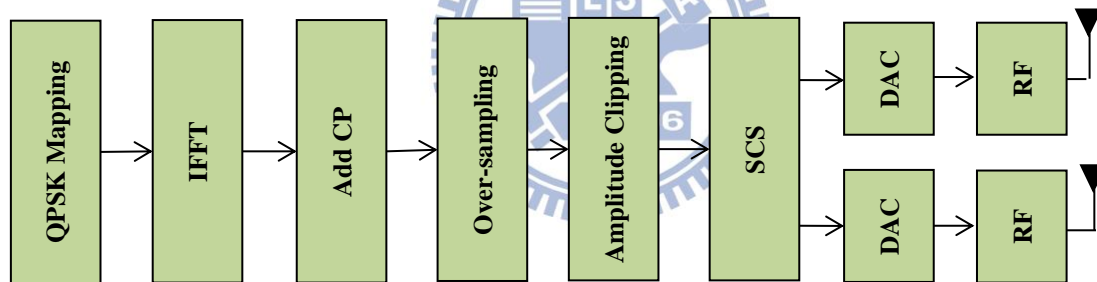


圖 3.8 傳送機架構方塊圖

3.3.1 提高取樣(Up-Sampling)

為了使類比端的低通濾波器更好設計，我們通常會先在數位端做提高取樣的動作。提高取樣可分為兩個步驟，首先將數位訊號依照超取樣因子(over-sampling factor)在訊號點之間補零，接下來再透過數位低通濾波器濾波，如此便可提高訊號的取樣頻率，本論文的數位低通濾波器我們使用平方根升餘弦(square root raised cosine, SRRC)濾波器，如圖 3.9 與圖 3.10 所示，為超取樣因子等於 4 的平方根升餘弦濾波器。

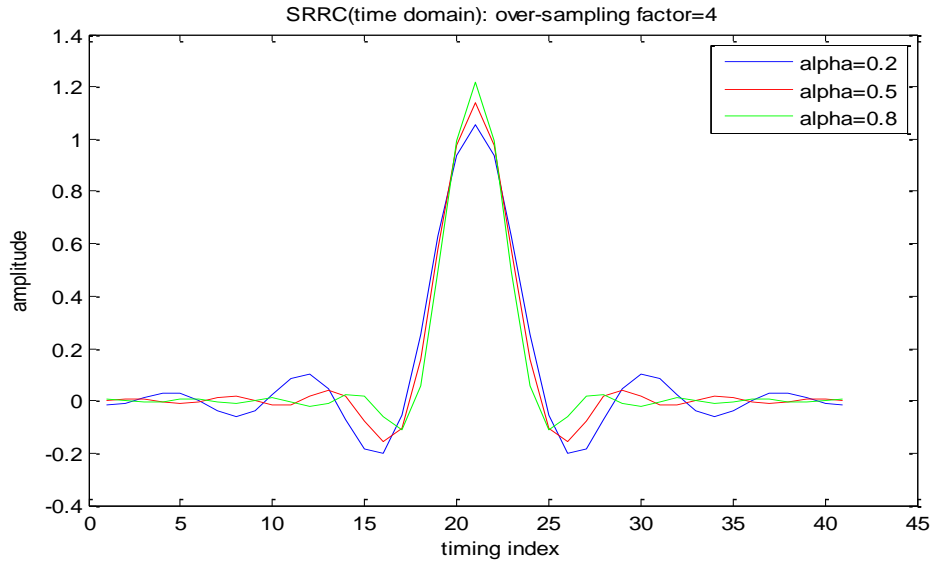


圖 3.9 時域平方根升餘弦濾波器

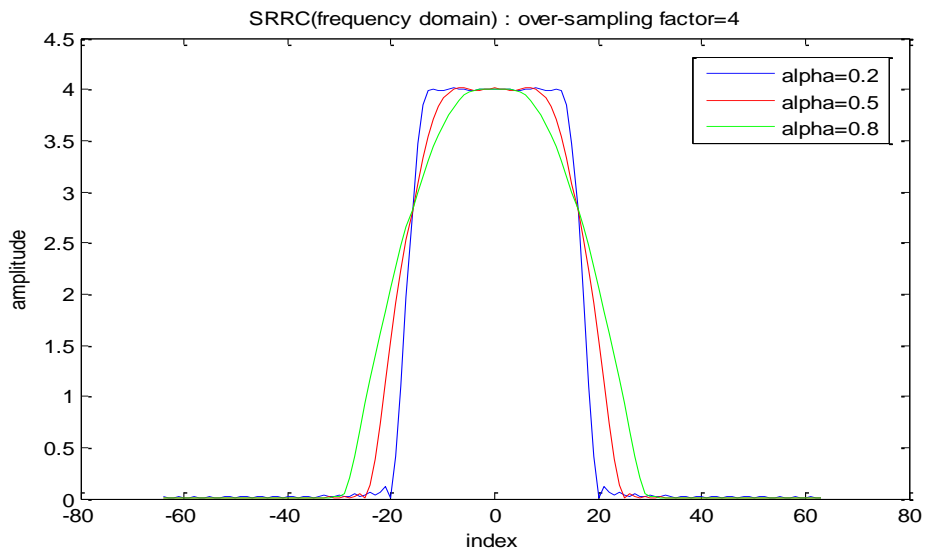


圖 3.10 頻域平方根升餘弦濾波器

一般來說，數位濾波器的硬體架構有以下三種[10]，如圖 3.11 所示。圖(a)為最常見的架構，但是有最長的關鍵路徑(critical path)，濾波器係數越多其關鍵路徑越長，此為其缺點。而圖(b)雖然重要路徑較短，但卻有扇入(fan-in)太大的缺點，需要較大的驅動能力。圖(c)可以改良圖(a)和圖(b)的缺點，在兩者之間取得一個平衡。

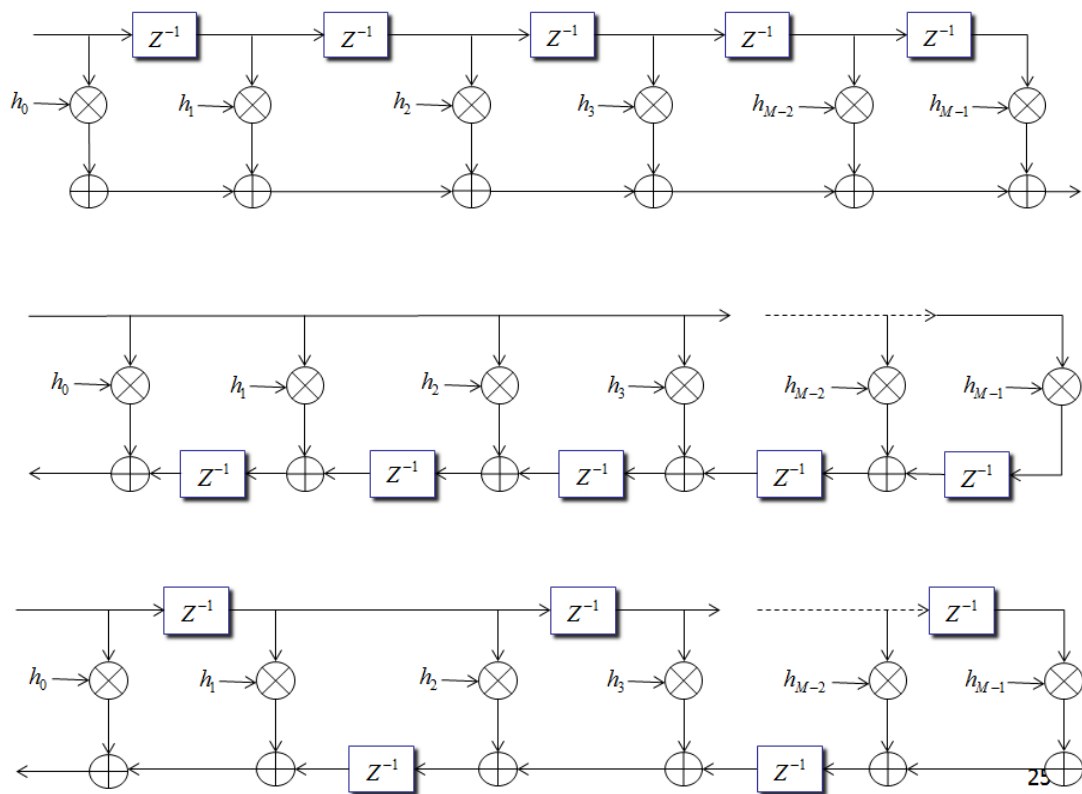


圖 3.11 (a) 直接型式，(b) 轉置型式，(c) 混合形式

由於提高取樣的輸入訊號的性質，我們可以將訊號補零的部分與係數相乘的乘法器省略。假設超取樣因子為 N ，我們可以將原本的係數拆成 N 組，亦即有 N 組子濾波器 (sub-filter)，而形成一多相位 (poly-phase) 濾波器，如圖 3.12 與圖 3.13 所示，如此只需一個多工器選擇不同時脈週期所需要的濾波器係數，所使用的乘法器可降為原本 $\frac{1}{N}$ ，關鍵路徑也因此縮短，由於圖 3.13 的每個子濾波器的係數較少，我們可以直接使用圖 (a) 的架構即可，這就完成提高取樣的硬體架構了。

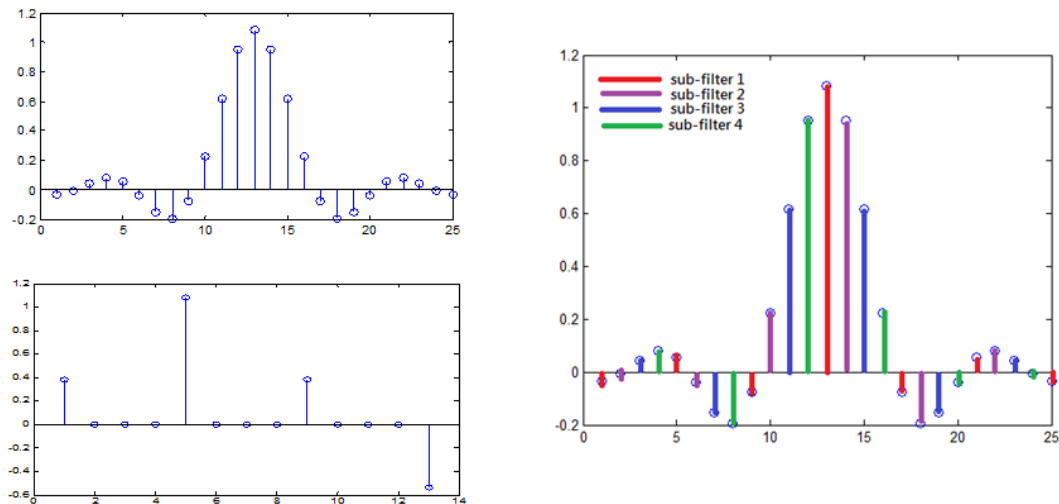


圖 3.12 多相位濾波器概念圖

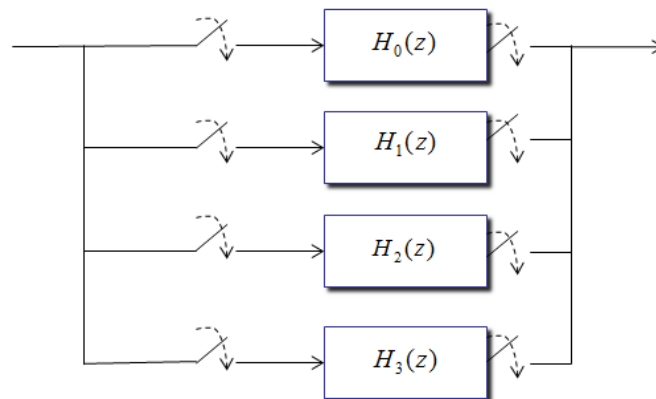


圖 3.13 多相位濾波器架構圖

3.3.2 振幅裁切(Amplitude Clipping)

在訊號進入 SCS 拆解成兩個固定波包訊號前，我們要先定義出最大振幅 A_m ，如果訊號振幅大於 A_m ，因為我們使用的 SCS 為 IQ 法，會造成計算錯誤訊號的根號內有負值，SCS 運算會發生問題，所以我們需要先對振幅超過 A_m 的訊號做處理。另外，由於我們計算錯誤訊號時使用多項式內插法(下一小節 3.3.3 會再做說明)，為了減少儲存係數，必須將內插曲線規範在一個限定範圍內，因此也需要定義一個振幅最小值 a_m 。在此我們的方法為直接將訊號的振幅做縮減到一定的範圍內，如(3.9)所示。振幅大於 A_m 則保留訊

號相位，並將訊號振幅縮減至 A_m 。如果振幅小於 a_m 則保留訊號相位，並將訊號振幅拉長至 a_m 。

$$x(t) = \begin{cases} x(t), & a_m \leq |x| \leq A_m \\ A_m e^{j\varphi(t)}, & |x| > A_m \\ a_m e^{j\varphi(t)}, & |x| < a_m \end{cases} \quad (3.9)$$

where $\varphi(t) = \angle x(t)$

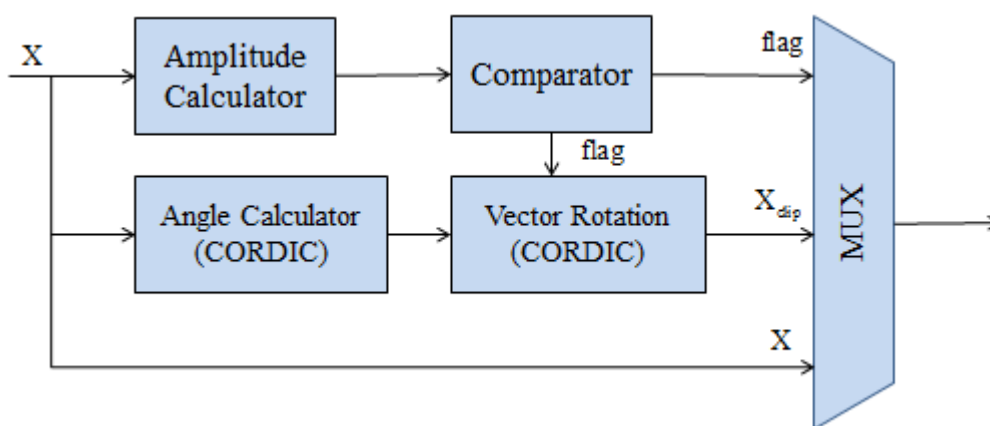


圖 3.14 振幅縮減器方塊圖

圖 3.14 為振幅裁切器方塊圖。我們要先算出訊號的振幅 $\sqrt{I^2 + Q^2}$ ，為了避免硬體上開根號的複雜運算，參考[12]以二分法線性逼近來取代根號的運算，此方法的最大誤差約為 2.2%，為可容許範圍。二分法線性逼近可以表示為

$$\sqrt{X^2 + Y^2} \approx \begin{cases} a_0 X + b_0 Y, & 0 \leq \frac{Y}{X} \leq \frac{1}{2} \\ a_1 X + b_1 Y, & \frac{1}{2} \leq \frac{Y}{X} \leq 1 \end{cases} \quad (a_0, b_0, a_1, b_1) = \left(1, \frac{3}{16}, \frac{3}{4}, \frac{11}{16}\right) \quad (3.10)$$

由(3.10)可觀察到，所有係數的分母皆為 2 的幕次方，這讓我們在硬體設計上可避免使用乘法器，只需用到簡單的移位器與加法器。圖 3.15 與圖 3.16 為二分法線性逼近(3.10)的架構圖。

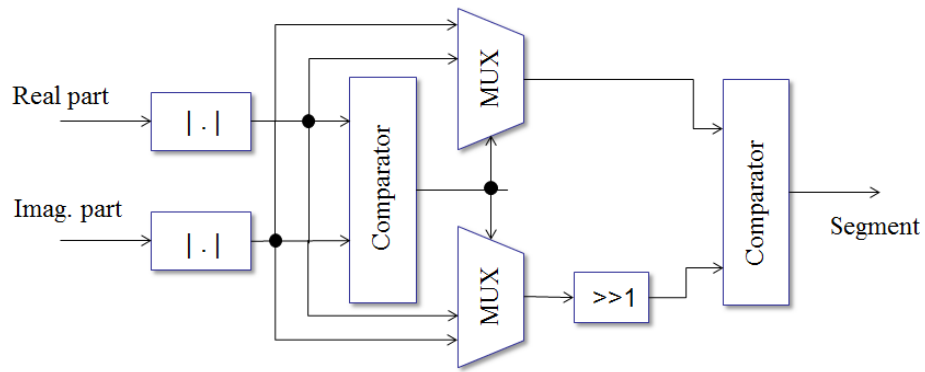


圖 3.15 範圍(Segment)判定區塊架構圖

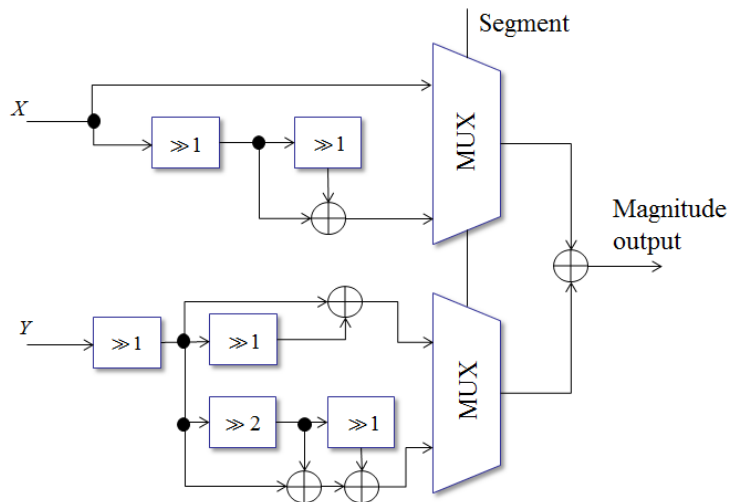


圖 3.16 二分法線性逼近架構圖

算出訊號振幅之後，經由比較器判別振幅是否大於 A_m 或小於 a_m ，如果沒有則不做任何動作就將原本的訊號輸出，反之則將處理過的訊號輸出。訊號處理前要先求出訊號角度，我們以 CORDIC 向量模式來實現此區塊，接下來再將純量的振幅旋轉前一步驟所求出向量訊號的角度，此處需使用到 CORDIC 旋轉模式，旋轉完後即得到我們要的訊號。

3.3.3 訊號拆解器(Signal Component Separator, SCS)

在 2.2.2 我們有提到，SCS 的實現主要有相位法[15]以及 IQ 法[16]，其方塊圖如圖 3.17 與圖 3.18。由於相位法需要多次的角度運算與旋轉，以及除法和反正弦(Arcsine)運算。因此在本論文中，我們以 IQ 法做為基本的架構。

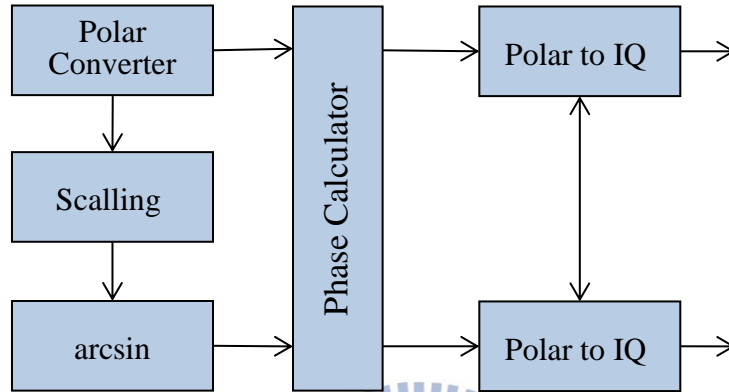


圖 3.17 SCS:相位法方塊圖

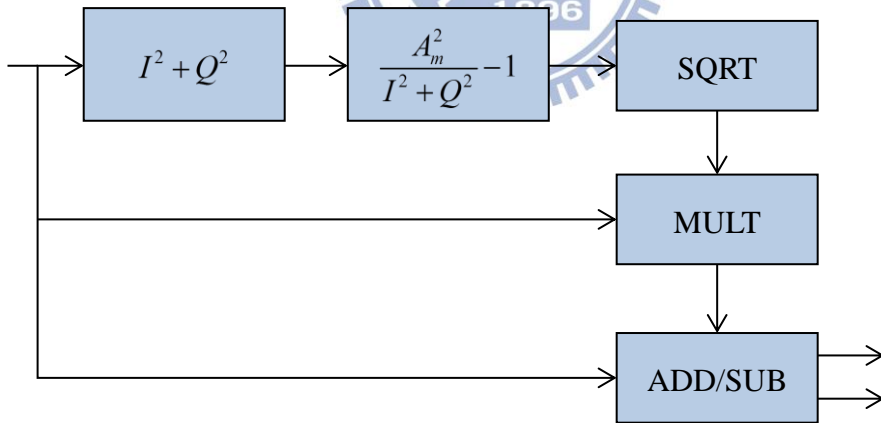


圖 3.18 SCS:IQ 法方塊圖

由圖 3.18 可知，為了求出錯誤訊號，我們需要除法器以及開根號，這都是在硬體時做我們想避免的。因此我們想簡化求出錯誤訊號的區塊。我們將錯誤訊號的根號項稱作錯誤因子(error factor)，其數學式表示如下

$$e_f(t) = \sqrt{\frac{A_m^2}{|s(t)|^2} - 1} \quad (3.11)$$

其中 $e_f(t)$ 為純量，在此我們將問題轉變成要簡化錯誤因子的運算，錯誤因子的唯一變數為訊號振幅，可以將此看成一對一的多項式對應，以高階多項式分段逼近錯誤因子的曲線[17-18]。為了節省硬體資源，我們需要先對輸入的訊號振幅做統計分析，決定逼近取線的上界與下界，並於訊號進入 SCS 前做前置的處理，此部分上一小節 3.3.3 已介紹。訊號振幅統計分析如圖 3.19 所示，我們將上界設定在 1.85，約為訊號統計變異數的兩倍左右，下界在訂在 0.1，由之後第四章的模擬可知，訂定此振幅範圍並不會對系統效能造成顯著的影響。逼近曲線如圖 3.20 所示，我們使用二階多項式來逼近此曲線，經模擬分析顯示我們需將此線分割成六個區塊，因此儲存的係數較多，不過其好處是使用的乘法器較少，關鍵路徑也相對較小，錯誤因子內插法的架構如 3.21 所示。六個區塊的二階多項式的係數如表 3.1 所示。

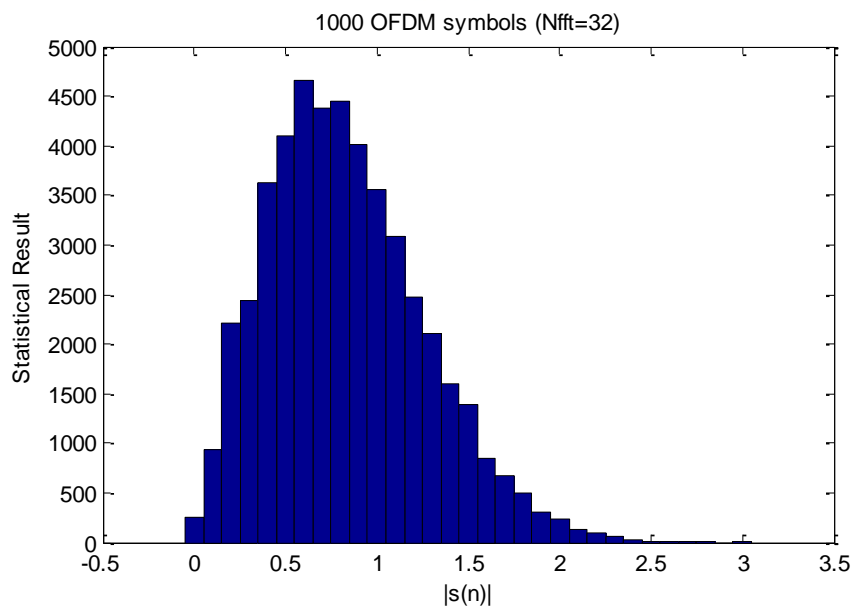


圖 3.19 OFDM 符元之振幅統計分析

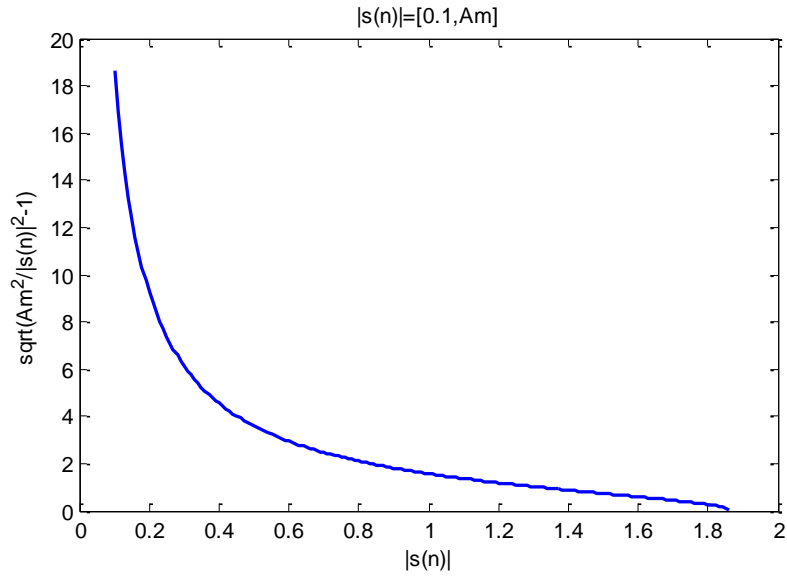


圖 3.20 SCS 錯誤向量曲線圖

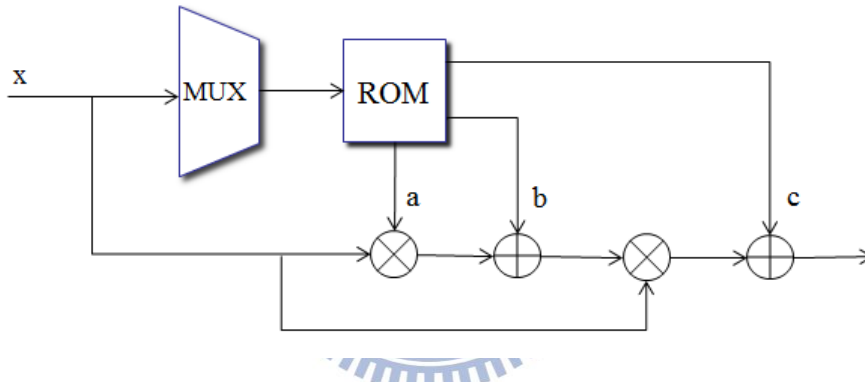


圖 3.21 錯誤因子內插法架構圖

表 3.1 二階多項式逼近： $aX^2 + bX + c$ 係數表

	a	b	c
0.1~0.2	623.000	-280.350	40.410
0.2~0.4	77.700	-70.190	20.195
0.4~0.75	10.314	-18.364	10.246
0.75~1.4	1.477	-5.324	5.437
1.4~1.8	-0.745	0.858	1.137
1.8~1.85	-53.842	62.161	-17.833

經過改良後，我們的 SCS 架構如圖 3.22 所示，由(3.11)可知，在進入多項式內插前，仍需求出訊號振幅，這部份我們於 3.3.1 已提到，可以使用二分法線性逼近來完成，避開了乘法器與開根號的使用，僅需簡單的移位器與加法器。而錯誤因子我們使用多項式逼近，需要用到乘法器與少量的記憶空間儲存係數，取代了原本複雜的除法器以及開根號。最後再將我們求到的錯誤因子與訊號相乘並做加減法的處理，如此就完成了我們 SCS 的架構設計。

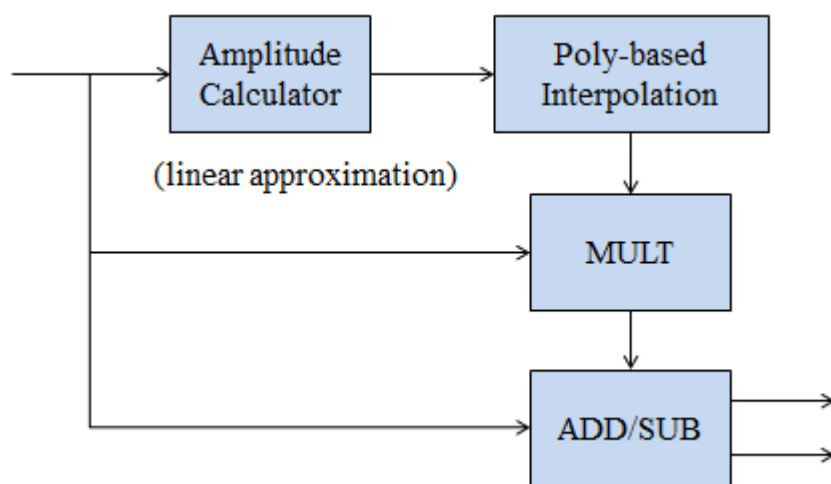


圖 3.22 SCS:改良後的 IQ 法方塊圖

3.3.4 前導訊號 (Preamble)

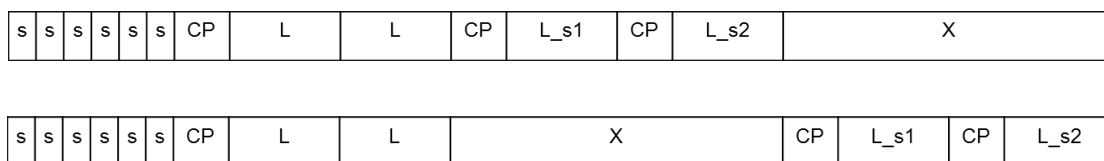


圖 3.23 前導訊號

前導訊號主要用來做接收端的同步處理，需要根據目的及演算法的不同，來訂定出符合要求的前導訊號，本論文的前導訊號為參考 802.11a 的規範[9]後自訂。如圖 3.23 所示本論文使用的前導訊號包含六個週期的短訓練符元(training symbol)，每個週期的短訓練符元包含 8 個符元長度，主要利用其重複性，來進行封包偵測以及粗略頻率偏移估測。接在後面的為長訓練符元，每個週期的長訓練符元包含 32 個符元長度，長訓練符元前

會加一段 CP，長訓練符元主要用來做符元起點估計以及精細頻率偏移估計。最後，為了做通道估測時避免兩根天線互相干擾，我們將 SCS 拆開後的兩個長訓練符元在不同的時間傳送，並於接收端將拆開後的長訓練符元相加並還原，如此便可有效估測出兩根傳送天線所產生不同的通道效應。

3.4 接收機設計

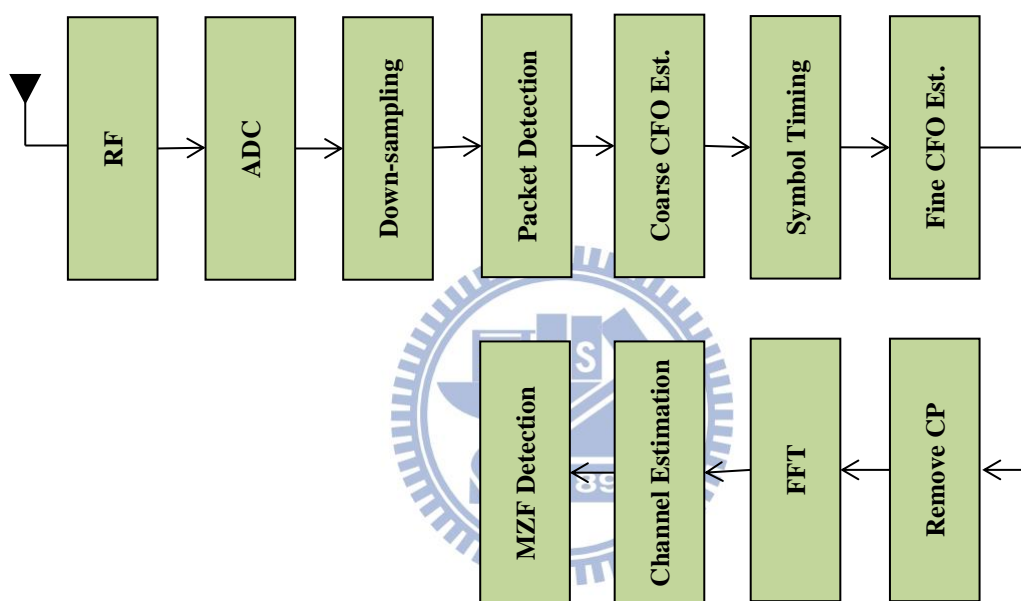


圖 3.24 接收機架構方塊圖

3.4.1 降低取樣(Down-Sampling)

在降低取樣之前，我們會先將訊號經過平方根升餘弦濾波器，目的是為了與傳送端提高取樣時的濾波做匹配，兩端濾波的效應合成即為升餘弦(Raised Cosine, RC)濾波器，也能將訊號頻寬外的雜訊濾掉。接下來才降低取樣頻率，根據降取樣因子 M ，我們只需每 M 個時脈週期輸出一個訊號，為了節省乘法器的使用，我們可利用 3.3.3 提高取樣所提到的多相位濾波器的概念，將濾波器拆成 M 部分， M 個時脈週期驅動一次訊號移位暫存器，且一次移動 M 個暫存器的位置，與提高取樣不同的是最後要將 M 個子濾波器輸出結果相加，得到輸出結果，本論文的 M 定義為 4。架構圖如下 3.25 所示。

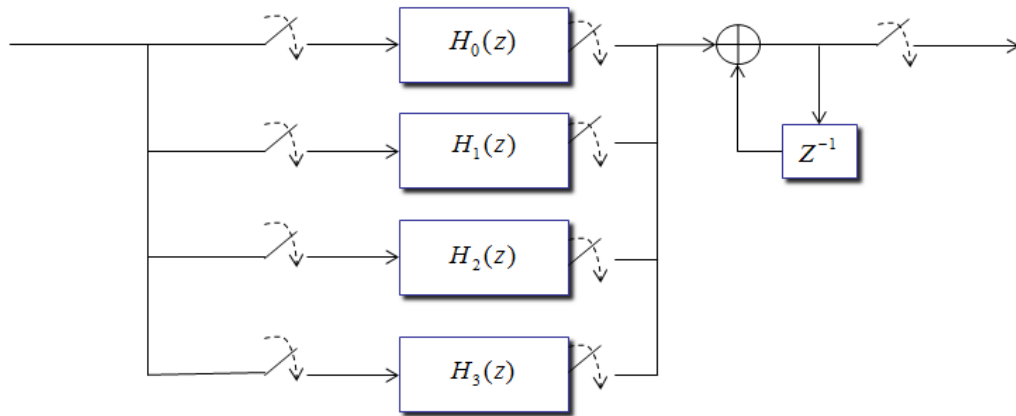


圖 3.25 降低取樣架構圖

3.4.2 封包偵測 (Packet Detection)

封包偵測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前，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有很多，在此我們是利用前導訊號重覆的特性來進行封包偵測。本論文的短訓練符元包含 6 個大小同為 8 的相同區塊所組成，因此我們可以設計兩個大小同為 16(定義為 L)的視窗，兩個視窗的間隔為 8(定義為 D)，當訊號通過時，兩個視窗就互相做相關性(correlation)運算，其數學式如下

$$\begin{cases} m_n = c_n = \sum_{k=0}^{L-1} r_{n+k} r_{n+k+D}^* \\ D = 8, L = 16 \end{cases} \quad (3.12)$$

從(3.12)可知，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，借由前導訊號所存在的關連性， m_n 也會隨著變大，當大於我們所設的門檻值時，則判定已接收到封包。這種方法簡單且易實現，不過會有 m_n 分佈範圍過大的問題，造成門檻值的設定相當困難。

為了解決門檻值的設定變動過大的問題，改善的方法就是對之前求到的 c_n 做正規化(normalization)，數學式如下

$$\begin{cases} c_n = \sum_{k=0}^{L-1} r_{n+k} r_{n+k+D}^* \\ p_n = \sum_{k=0}^{D-1} r_{n+k+D} r_{n+k+D}^* \\ m_n = \frac{|c_n|^2}{p_n^2} \end{cases} \quad (3.13)$$

將 c_n 除上視窗的能量，我們就可以把 m_n 的範圍控制在 0 與 1 之間，在設定門檻值時就方便多了。不過正規化會造成在硬體設計方面，得多付出除法器的代價。

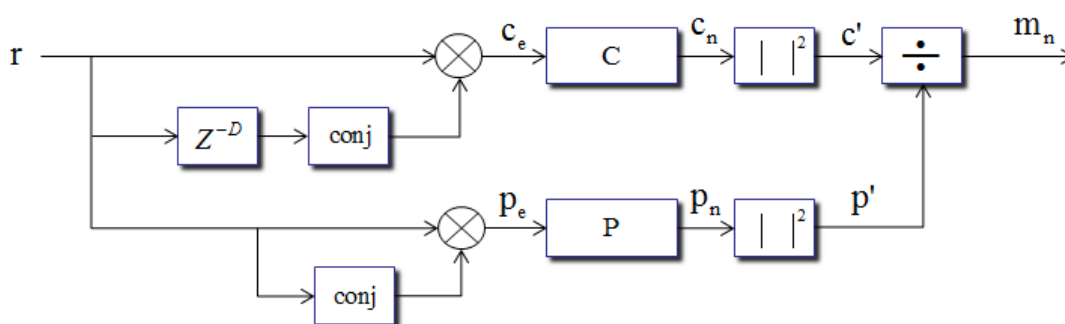


圖 3.26 封包偵測基本架構圖

如圖 3.26 所示，訊號進來後會分兩路進行，為了降低硬體之面積與複雜度，透過移動視窗(sliding window)遞迴算出 c_n 與 p_n ，其數學式表示如下

$$\begin{aligned} c_{n+1} &= c_n + r_{n+1} r_{n+1+D}^* - r_{n+L+1} r_{n+L+1+D}^* \\ p_{n+1} &= p_n + |r_{n+1+D}|^2 - |r_{n+L+1+D}|^2 \end{aligned} \quad (3.14)$$

移動視窗運算架構圖如下所示

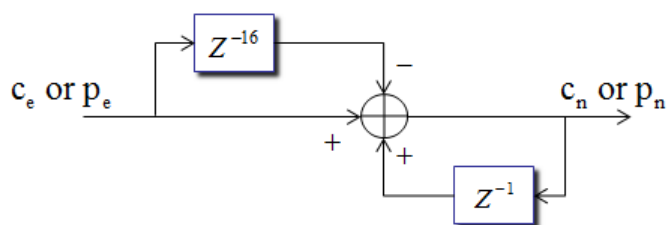


圖 3.27 移動窗戶架構圖

另外，由圖 3.26 可看到，除若干乘法器與加法器外，我們還需要一個除法器，這是我們所不樂見的。除法器除了會佔去不少的面積，也會有時間的延遲，為了避免除法器的使用，我們將門檻值訂亦為 0.75，此門檻所造成的封包遺失率為可接受範圍(見第四章模擬部分)，但可大大的增加在硬體上實作的便利性，我們只需將圖 3.26 中的 c' 透過移位器處理，再經過一個比較器，當 $c' > \frac{3}{4} p'$ ，我們判定偵測到封包。改良後的封包偵測硬體架構如圖 3.28 所示。

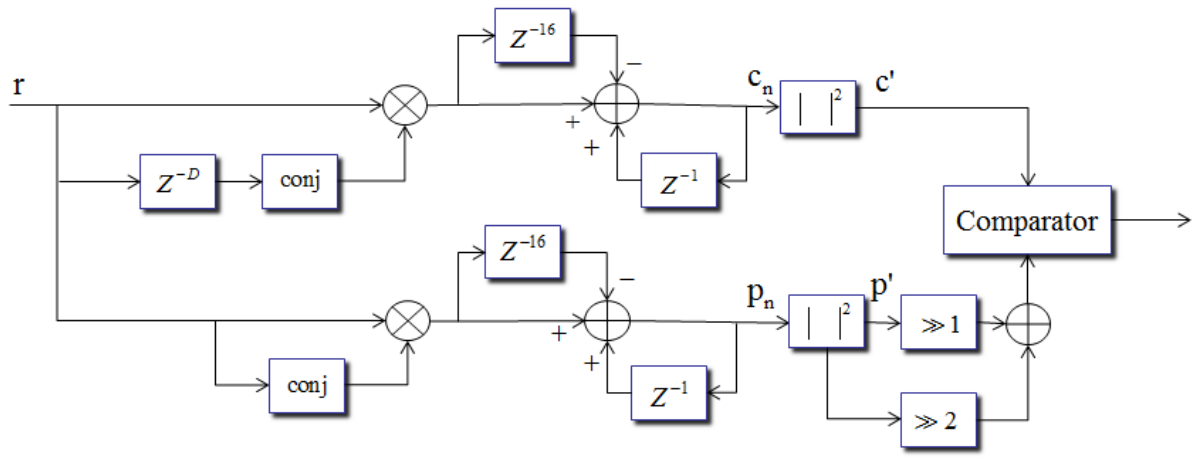


圖 3.28 改良後之封包偵測架構圖

3.4.3 符元時序估計 (Symbol Timing Estimation)

在 OFDM 接收機中，通常要有一個好的符元起點估計，才能取到正確的訊號做後面的 FFT。一般來說符元起點估計的做法，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，其數學表示如下

$$\hat{t}_s = \arg \max_n \left| \sum_{k=0}^{L-1} r_{n+k} t_k^* \right|^2 \quad (3.15)$$

其中 r_{n+k} 為接收訊號， t_k 為已知用來匹配的前導訊號，由上式可知，時序偵測與封包偵測不同，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，也就是說要在我們所決定的偵測時間長度內決定出一個最大值當作符元起點。

而我們選用來匹配接收訊號的前導訊號取自短訓練符元區塊的最後八個取樣點與長訓練符元區塊的最前面八個取樣點，因為這兩組前導訊號的相關性不大，有較為明顯的界線，會更容易偵測出符元起點的位置。在此我們採用 3.3.2 介紹過的混合形式的匹配濾波器，以有效的降低關鍵路徑之長度，並在輸出的地方比較並記錄最大值的位置 (index)，如圖 3.29 所示，這就完成了我們符元起點估計的硬體架構了。

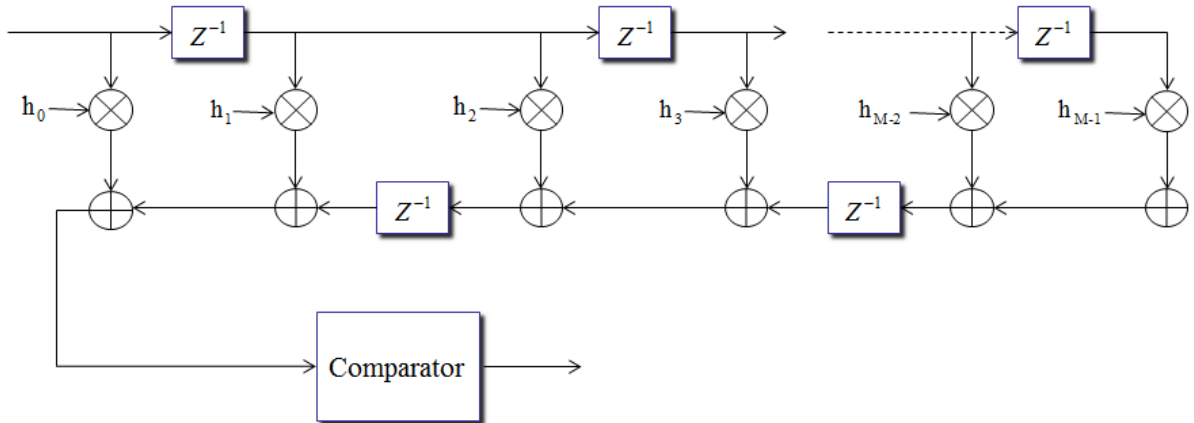


圖 3.29 符元起點估計架構圖

3.4.4 頻率偏移估計與補償 (Frequency Offset Estimation/Compensation)

傳送端與接收端的震盪器 (crystal oscillator) 頻率通常不會相同，若不進行頻率偏移補償，將使子載波間的正交性被破壞，造成子載波間的相互干擾，而對系統效能造成嚴重的影響。本論文使用 [9] 之頻率偏移估計的方法，主要是利用前導訊號的週期性特性 (令週期為 D)，對兩個時間不同但數值相同的複數訊號而言，當有頻率偏移時，兩點會因為時間的差異而產生一相位差，於是由已知間距是 D ，因此可以容易估算出傳送端與接收端的頻率偏移。

假設傳送的時域基頻訊號為 s_n ，載到載波上的訊號為 y_n ，則傳送端的訊號可以表示如下

$$y_n = s_n e^{j2\pi f_n n T_s} \quad (3.16)$$

其中， f_{tx} 為傳送端的載波頻率， T_s 為每個子載波的取樣週期，而接收端將射頻訊號降到

基頻的載波頻率為 f_{rx} ，不同震盪器所產生的載波頻率不會完全相同，也就是說 f_{tx} 與 f_{rx} 會有差異，在忽略雜訊的情況下，解調訊號可表示如下

$$\begin{aligned} r_n &= s_n e^{j2\pi f_{tx} n T_s} e^{-j2\pi f_{rx} n T_s} \\ &= s_n e^{j2\pi (f_{tx} - f_{rx}) n T_s} \\ &= s_n e^{j2\pi f_{\Delta} n T_s} \end{aligned} \quad (3.17)$$

其中 $f_{\Delta} = f_{tx} - f_{rx}$ ，為傳送端與接收端載波頻率的差值，D 為兩個重複區塊中兩點相同取樣值的延遲，令 z 為：

$$\begin{aligned} z &= \sum_{n=0}^{L-1} r_n r_{n+D}^* \\ &= \sum_{n=0}^{L-1} s_n e^{j2\pi f_{\Delta} n T_s} (s_{n+D} e^{j2\pi f_{\Delta} (n+D) T_s})^* \\ &= \sum_{n=0}^{L-1} s_n s_{n+D} e^{j2\pi f_{\Delta} n T_s} e^{-j2\pi f_{\Delta} (n+D) T_s} \\ &= e^{-j2\pi f_{\Delta} D T_s} \sum_{n=0}^{L-1} |s_n|^2 \end{aligned} \quad (3.18)$$

我們因此可得頻率偏移估計值如下

$$\hat{f}_{\Delta} = -\frac{1}{2\pi D T_s} \angle z \quad (3.19)$$

由於 z 的角度 $-2\pi f_{\Delta} D T_s$ 必須定義在 $-\pi$ 與 π 之間，所以頻率偏移所能估測的範圍有所限制：

$$f_{\Delta} \leq \frac{\pi}{2\pi D T_s} = \frac{1}{2D T_s} \quad (3.20)$$

由(3.20)可知，D 越大所能估測的範圍越小，因此我們先使用 D 較小的短訓練符元做粗略(coarse)估計，如此補償後所剩的頻率偏移角度變小，為了得到更可靠的結果，再以 D 較大長訓練符元做細部(fine)估計，也就是我們在補償頻率偏移時，必須對接收到訊號以兩次估計得到的角度和做倍數的旋轉。基本的架構圖如圖 3.30 所示。

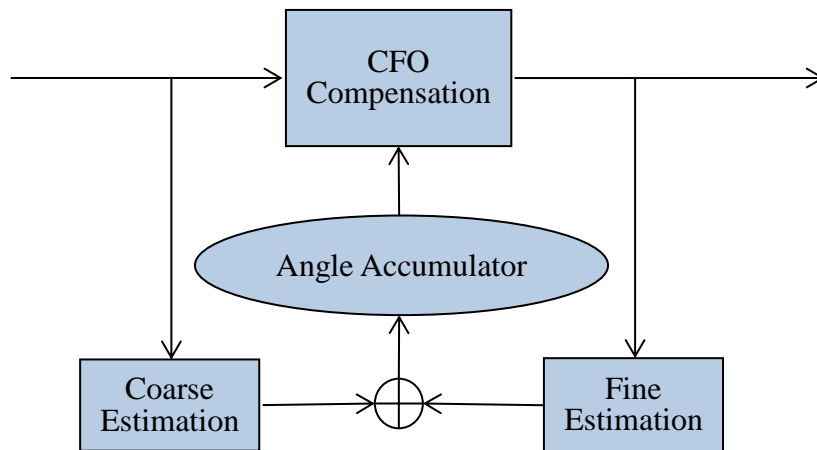


圖 3.30 頻率偏移補償方塊圖

圖 3.30 中，粗估(Coarse Estimation)與細估(Fine Estimation)兩區塊分別是由短訓練符元長訓練符元估出的旋轉角度，而細估區塊的輸出，必須等長訓練符元經由 CORDIC 旋轉粗估區塊所產生的角度之後，才能由細估區塊計算出細估的旋轉角度，並與粗估的旋轉角度相加之後，算出所需要旋轉的總角度。有關 CORDIC 的硬體設計，可參照 3.2.2。接下來則是頻率偏移估計的硬體設計。由(3.18)，我們可以將硬體架構可表示如圖 3.31。

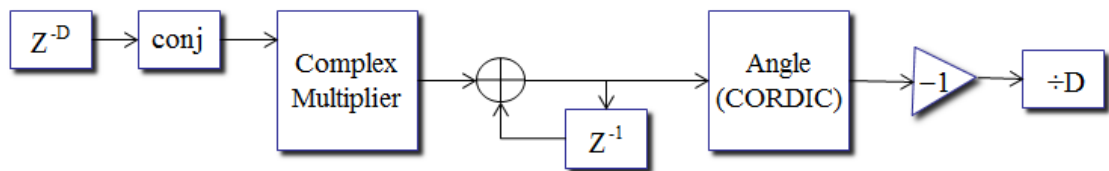


圖 3.31 Coarse/Fine Estimation 硬體架構圖

其中 D 在粗略估計時為 8，在細估計時為 32。角度的求取利用 CORDIC 向量模式得到，至於 $\div D$ 的方塊，因為 D 是 2 的次方，所以只需要一個移位器即可。而角度累加(Angle Accumulator)區塊為一固定角度的累加器，其功能為對每個取樣值旋轉固定角度的倍數，其架構如下

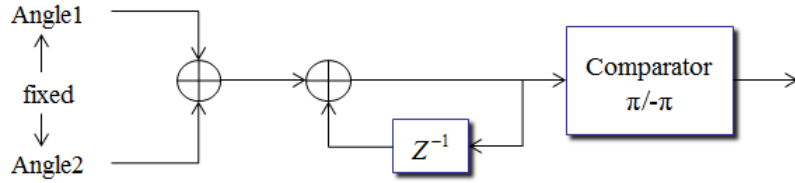


圖 3.32 Angle Accumulator 架構圖

上圖中的輸入只是一個最小的旋轉角度 a ，給第一個輸入的取樣值做旋轉，而之後的旋轉角度分別為 $2a, 3a\dots$ 。角度累加大於 π 或小於 $-\pi$ 時，為了避免超過 CORDIC 所能操作的範圍，所以要做減 2π 與加 2π 的動作，這是圖 3.28 中比較器(Comparator)的運作方式。將以上元件連結起來以後，就是我們所使用的頻率補償區塊了。

3.4.5 通道響應估測 (Channel Estimation)

在通道估測中，3.3.5 前導訊號已介紹，我們將由 SCS 拆開後的長訓練符元，由第一根天線先後傳送，第二根天線則不送資料，於接收端做 FFT 之前，先將拆開後的時域長訓練符元相加再做轉換，轉換後再透過運算得到第一根天線的通道響應，至於要求第二根天線的通道響應做法與第一根相同，只要在傳送端改成第一根天線不送訊號，第二根天線則傳送由 SCS 拆開後的時域長訓練符元。通道估測的數學表示式如下

$$R_k = H_k X_k + W_k \quad (3.21)$$

$$\begin{aligned} \hat{H}_k &= R_k X_k^* \\ &= (H_k X_k + W_k) X_k^* \\ &= H_k |X_k|^2 + W_k X_k^* \\ &= H_k + W_k X_k^* \end{aligned} \quad (3.22)$$

其中， R_k 為接收端接收到的子載波訊號， X_k 為傳送端傳送的子載波， H_k 為影響子載波的通道效應， W_k 為雜訊。 \hat{H}_k 則表示運算後估測出來的通道效應。由於長訓練符元頻域是由一堆 +1 與 -1 所組成，所以乘法器可用簡單的查表與判斷就可取代，在硬體實現上相當容易。

由於本論文的偵測方法使用 MZF 偵測，因此我們得到兩根天線的通道係數後，再做 MZF 通道係數的運算(2.23)，由[8]可知道，參數 μ 在 0.7 倍左右的最大振幅效能最好，我們選擇相差不多的 0.75，如此可以用移位器取代乘法器的使用。運算完後將通道係數存在記憶體中，當後端的訊號等化要執行時，就直接將通道係數輸出給 MZF 等化器作運算，進而解決訊號延遲的問題。通道響應估測的硬體架構如圖 3.33 所示。

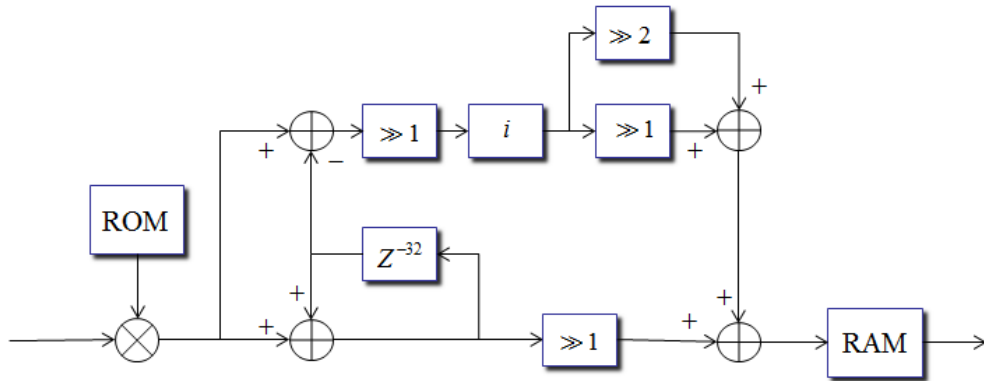


圖 3.33 通道響應估測架構圖

3.4.6 MZF 偵測 (MZF Detection)

訊號等化部分，我們利用 3.4.5 通道估測得到要用來做 MZF 偵測的通道係數做處理，因為我們使用 OFDM 技術傳輸，訊號的等化可簡化為對每個子載波分別做處理，也就是說通道效應對頻域上每個子載波可視為平緩衰減(flat fading)，如此我們只需把相對應子載波位置的通道效應除掉，即可完成訊號的等化。由於通道效應為複數，訊號等化的數學表示式如下

$$\frac{x}{a+bi} = \frac{x}{a+bi} \cdot \frac{a-bi}{a-bi} \quad (3.23)$$

架構如下圖 3.34 所示

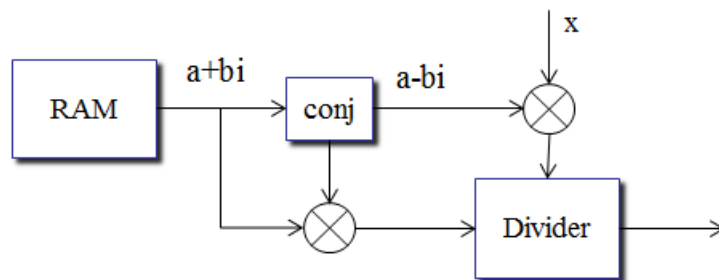


圖 3.34 訊號等化架構圖

由上圖可知，我們仍需要一個除法器，而此除法器是無法避免的，在此我們使用的除法器為恢復式除法(restoring division)[19]，概念上與直式除法相同，接下來我們以一個簡單的範例做說明，可以更容易理解。假設我們所要得到的商數整數位元為 5，小數位元為 3，範例如下

Example : $29 \div 3$

$$29 - 2^4 \cdot 3 = -17 \quad \boxed{0} \text{ restore}$$

$$29 - 2^3 \cdot 3 = 5 \quad \boxed{1}$$

$$5 - 2^2 \cdot 3 = -7 \quad \boxed{0} \text{ restore}$$

$$5 - 2^1 \cdot 3 = -1 \quad \boxed{0} \text{ restore}$$

$$5 - 2^0 \cdot 3 = 2 \quad \boxed{1}$$

$$2 - 2^{-1} \cdot 3 = \frac{1}{2} \quad \boxed{1}$$

$$\frac{1}{2} - 2^{-2} \cdot 3 = -\frac{1}{4} \quad \boxed{0} \text{ restore}$$

$$\frac{1}{2} - 2^{-3} \cdot 3 = \frac{1}{8} \quad \boxed{1}$$

$$\text{Ans. } 01001.101_{\text{dec.}} \rightarrow 9.625$$

首先，我們要先決定商數的位元大小 m ，接下來只需重複 m 次的移位減法與比較，即可完成除法的硬體架構。由於我們所設定的商數位元較大(為 12)，會造成關鍵路徑太長，影響系統最大操作頻率，因此我們將其做一次的管線化(pipelining)處理，以解決除法器的重要路徑太長的缺點。

四、模擬

4.1 系統效能模擬

本論文中的 LINC-OFDM 系統模擬使用 SUI[20]的相關通道模型，產生不同的相關係數(correlation coefficient) 之衰落通道(fading channel)，系統的參數設定如表 4.1 所示。其中訊號拆解器參數 A_m 我們設定為四倍標準差，因此系統中的 PAPR 值即可被限制在 6dB 以內。

表 4.1 LINC-OFDM 系統模擬環境參數

Parameter	Setting
Mapping	QPSK
OFDM size	32
Length of CP	16
Number of OFDM symbols /packet	3
Number of channel taps	Successive 3 taps
A_m	4 times of Standard Deviation
Correlation coefficient	0.96

由於傳收機為數位處理系統，但在環境中訊號需以類比的型態來傳送。因此在傳送端要先將傳送訊號由數位訊號轉成類比訊號才可送出，即需要數位轉類比轉換器(Digital to Analog Converter, DAC)。接收端要將環境中收到的類比訊號轉換成數位訊號，以做後續的處理，所以需要類比轉數位轉換器(Analog to Digital Converter, ADC)。但是Matlab模擬的環境皆為數位，我們想讓模擬更接近現實，因此藉著較高的取樣頻率，讓訊號接近類比型態，我們稱作以數位來模擬類比(Digital Modeled Analog; DMA)，一個數位模擬類比濾波器實際上仍為數位，不過它可以模擬類比的行為。本論文的模擬中所用的DMA架構如圖4.1所示。

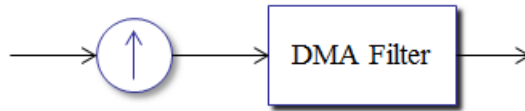


圖4.1 DMA架構圖

在此章節，我們先做整個系統的浮點數模擬，傳送機架構如圖 3.4 所示，接收機架構如圖 3.20 所示。由於訊號傳送是以封包為單位，當有封包遺失時，會使位元錯誤率(Bit Error Rate, BER)或符元錯誤率(Symbol Error Rate, SER)受到扭曲，因此如要計算 BER 或 SER，應要扣除封包遺失所造成的位元或符元錯誤，或者改以計算系統的封包錯誤率(Packet Error Rate, PER)，只要傳送封包有一個位元或符元發生錯誤，即表示此封包發生錯誤，計算封包錯誤率包含了訊號偵測錯誤以及封包偵測錯誤造成的封包遺失。本章節我們主要以 SER 來表示系統效能的好壞。

在章節 2.4.1 與 2.4.2 中介紹了兩種等化器 $\tilde{\mathbf{h}}_{ZF}$ 與 $\tilde{\mathbf{h}}_{MZF}$ ，如(2.19)所示使用 $\tilde{\mathbf{h}}_{ZF}$ 等化，會因兩個通道響應差異而產生大的干擾項，而 $\tilde{\mathbf{h}}_{MZF}$ 等化器在式(2.22)中已經扣除了干擾項的平均值，因此干擾項會變小，等化後的訊號失真較小。此兩種等化器的等化效能在通道相關係數為 0.96 時模擬結果如圖 4.2 所示，圖 4.2 是尚未加入頻率偏移的結果，從圖中可看出 $\tilde{\mathbf{h}}_{MZF}$ 確實可以有效的提升等化的效能。另外，圖 4.3 顯示系統的封包遺失率(packet loss rate)，封包遺失率僅計算封包偵測時封包遺失所造成的錯誤，我們設定一個視窗大小代表正確偵測到封包的範圍，在此範圍內偵測到的封包才能在後續的符元時序估計找到正確的峰值，若未在此範圍偵測到封包，則代表封包遺失，我們以三種不同門檻值做分析，從圖 4.3 可知，門檻值為 0.75 時效能最好，由圖 4.2 及圖 4.3 的比較，我們也可知此系統的封包遺失率相較於封包錯誤率而言小很多，因此不會支配 LINC-OFDM 系統的封包錯誤率曲線。圖 4.4 則以 SER 來表示系統的效能。

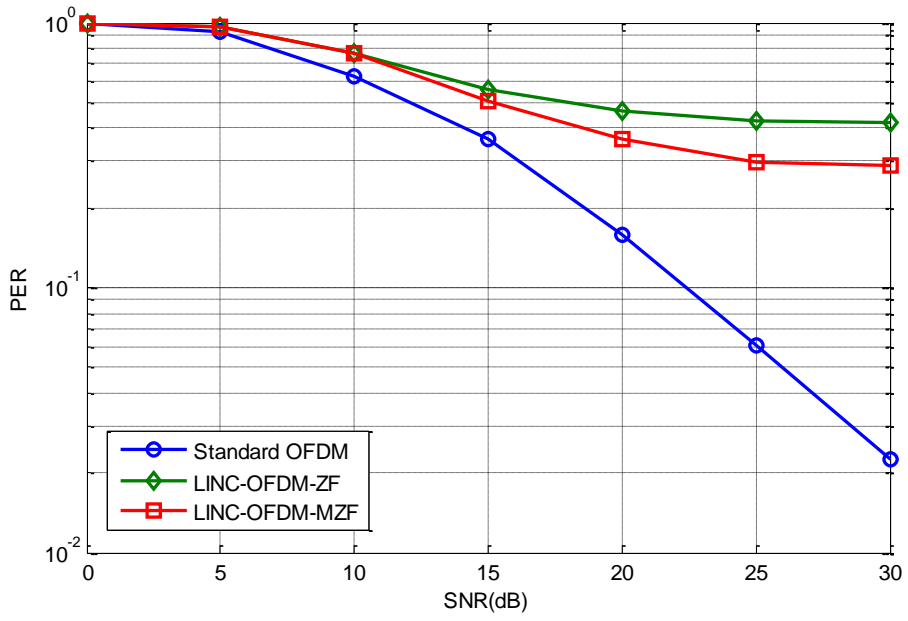


圖 4.2 LINC-OFDM 系統 ZF 與 MZF 等化器效能(PER)

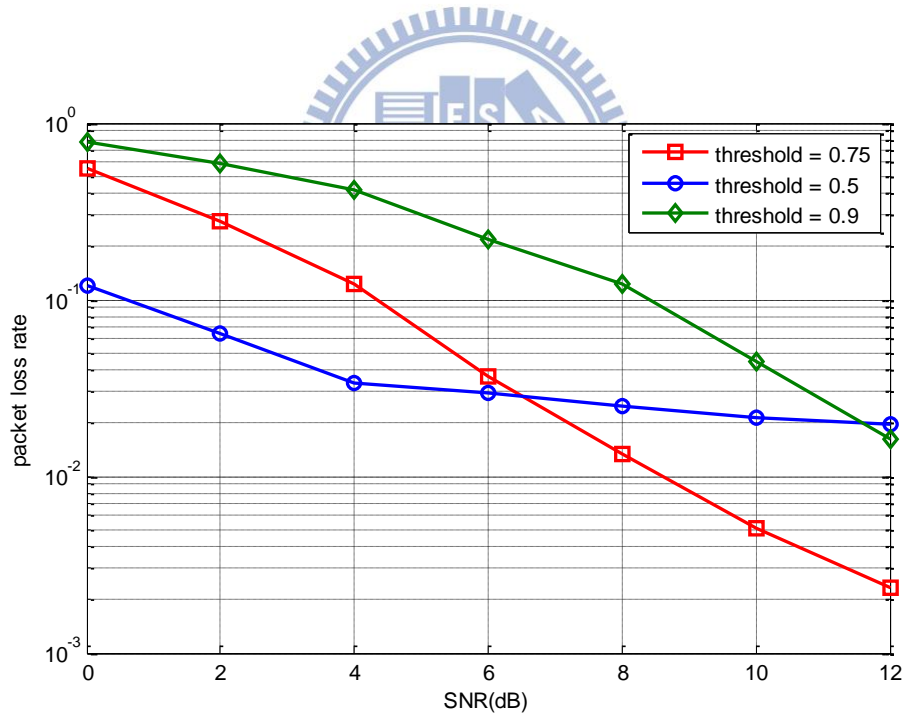


圖 4.3 系統封包遺失率

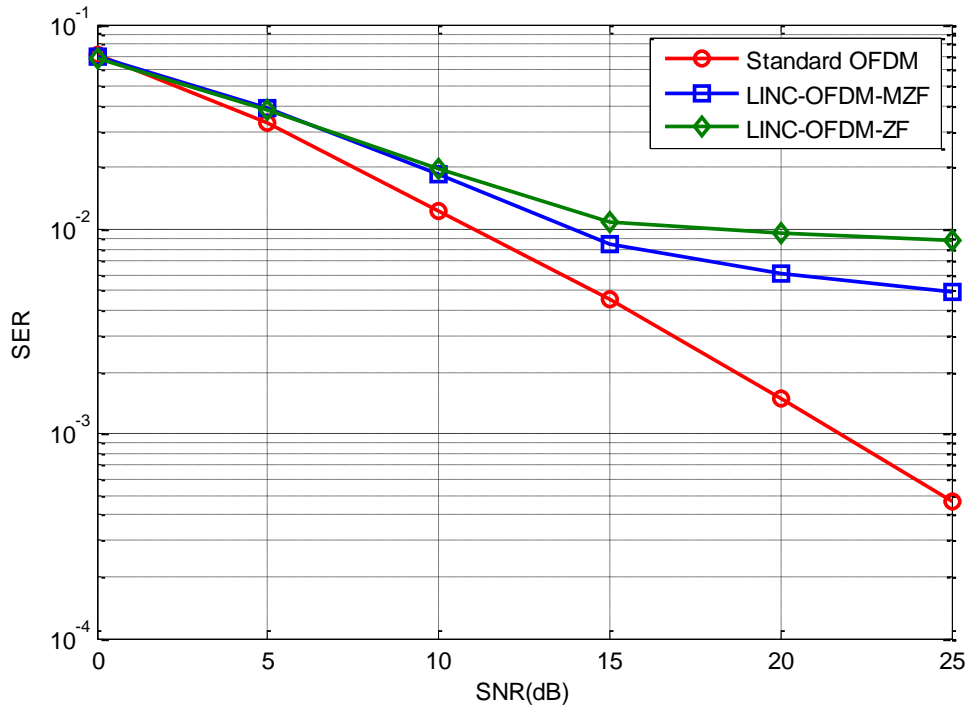


圖 4.4 LINC-OFDM 系統 ZF 與 MZF 等化器效能(SER)

接下來，我們要探討頻率偏移對系統效能的影響，章節 3.4.5 我們有提到，頻率偏移補償分為粗調與細調兩部分，圖 4.5 即為這兩個部分估計後的均方錯誤(Mean Square Error, MSE)，由於在低訊雜比(Signal to Noise Ratio, SNR)時雜訊影響太大，不管是粗估或細估都無法很準確的估計偏移的頻率，但高 SNR 時便能看出兩者有顯著的差異。圖 4.6 則是將頻率偏移效應加入我們的浮點數系統，觀察對效能的影響，所加入頻率偏移大小設定為 0.01 倍的子載波間距(subcarrier spacing)，我們可以觀察到當 SNR 低時，頻率偏移效應對系統效能有較大的影響，但到高 SNR 時，頻率偏移估計漸趨準確，且失真漸由等化干擾項主導，因此可看到有頻率偏移和沒有頻率偏移的系統的效能會趨於一致。

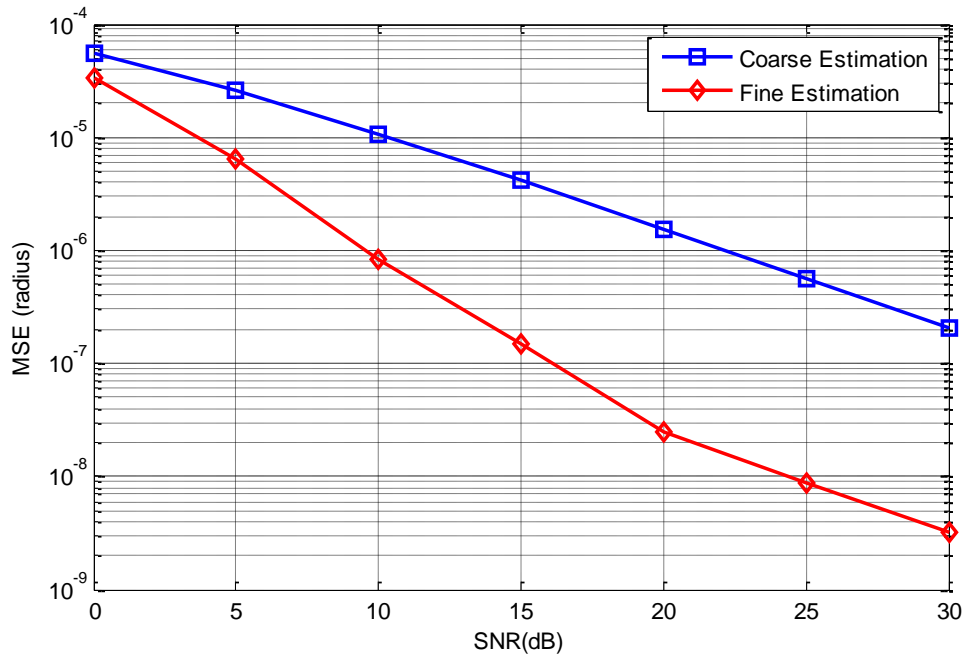


圖 4.5 MSE of CFO Estimation

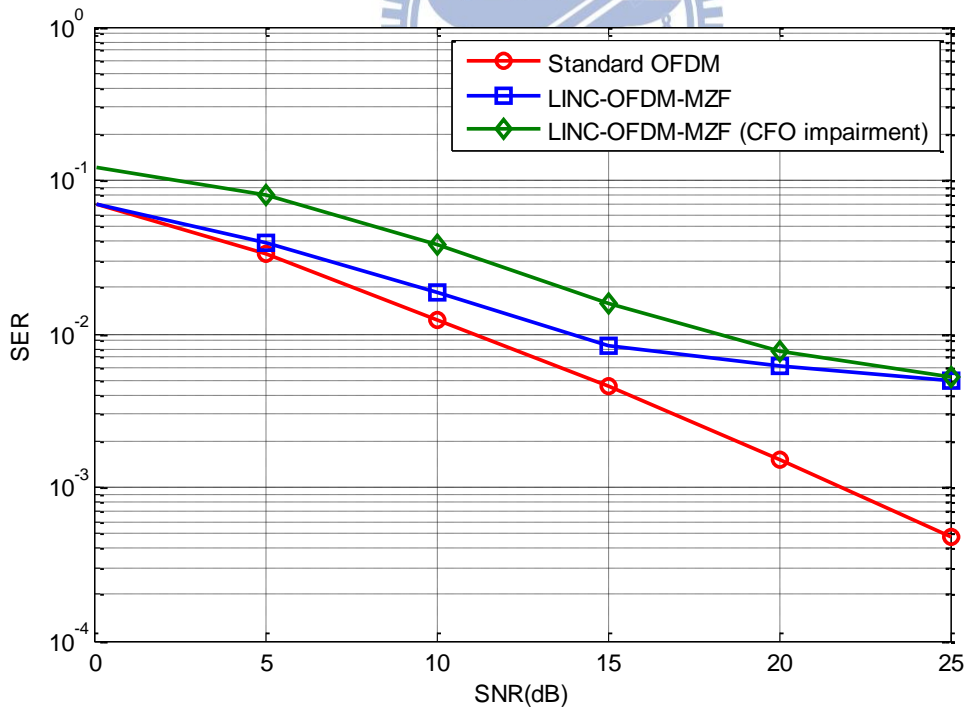


圖 4.6 LINC-OFDM 系統效能: 頻率偏移效應

4.2 硬體設計模擬

針對 CORDIC 演算法，我們以 Matlab 模擬了圖 5.6 及圖 5.7 兩張圖，代表隨著遞迴次數的增加，由向量模式所求出的角度誤差或旋轉模式所旋轉後的向量誤差會越來越小，也就是說遞迴次數越多，CORDIC 演算法越準確。但是在硬體設計面積與時序的考量上，我們不希望遞迴次數太多，由圖 5.6 及圖 5.7 可觀察到，當遞迴次數在 8 以後，CORDIC 演算法兩種模式的 MSE 曲線下降會趨於平緩，我們因此根據模擬的結果將遞迴次數定為 8，在遞迴次數為 8 的情況下，如果使用平行架構，需要較長的時間才能完成動作，會造成最大操作頻率下降，若使用遞迴架構，則需要多個時脈週期才能完成，造成過多的訊號輸出延遲，增加了模組間時序控制的麻煩。因此在章節 3.2.1 有提到，我們將 CORDIC 架構結合圖 3.2 的遞迴架構與圖 3.3 的平行架構，以此來兼顧面積與時序的考量。

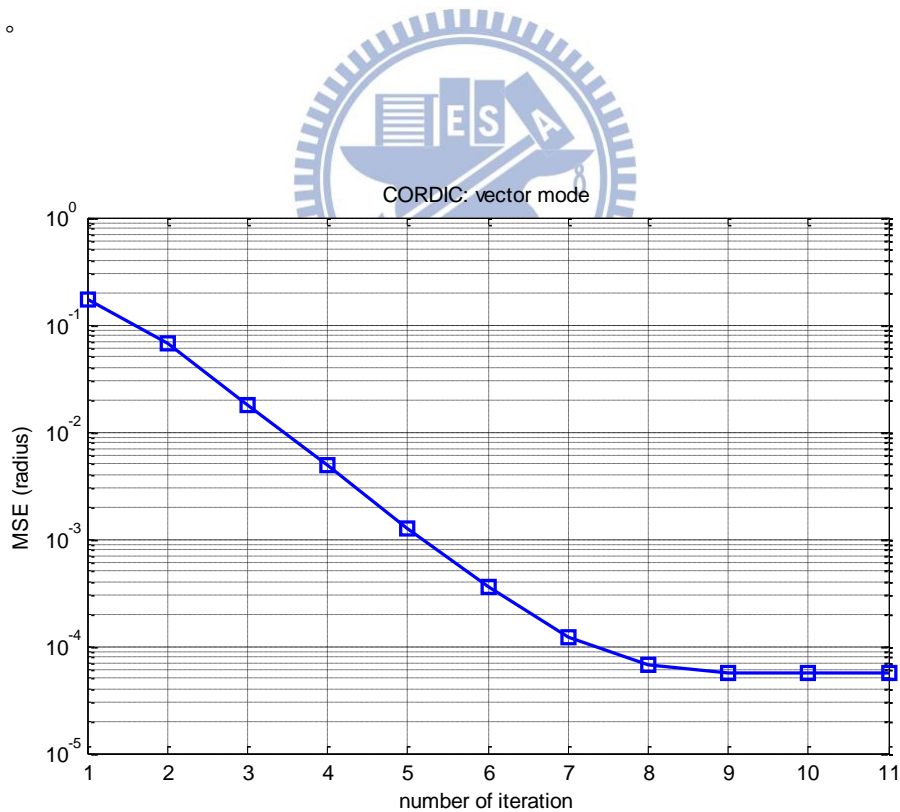


圖 4.7 MSE of CORDIC Vector mode

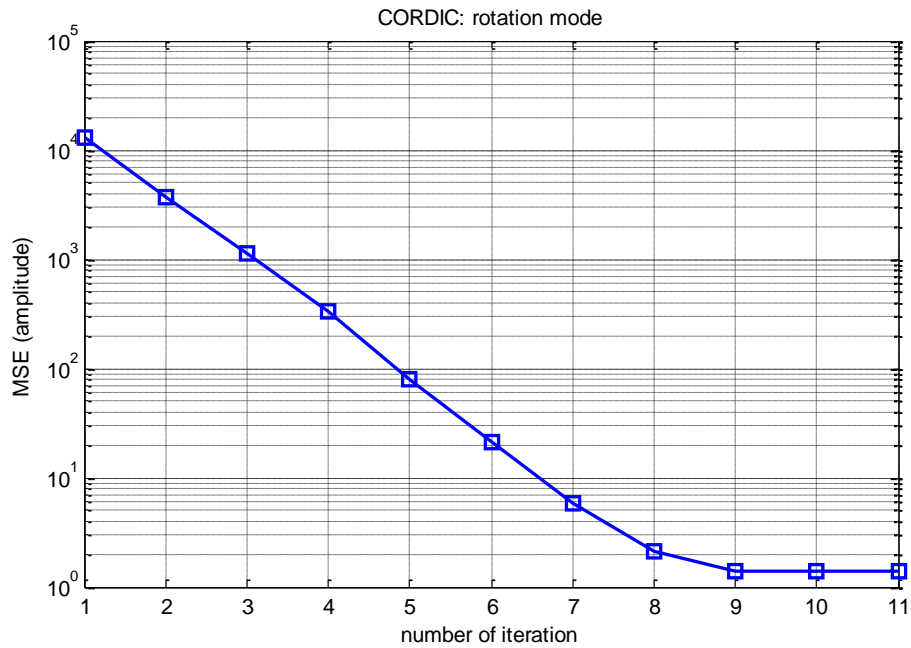


圖 4.8 MSE of CORDIC Rotation mode

接下來，我們將振幅裁切做定點模擬，振幅裁切要將訊號經過式(3.9)的運算處理，振幅最大值如前所述我們設定為訊號的四倍標準差，於章節 3.3.5 我們有提到，求取錯誤訊號時我們使用到多項式內插，考量到硬體的面積，需要將訊號振幅定一個下界，因為振幅太小會造成式(3.11)錯誤因子的動態範圍變很大，所需要用來儲存訊號的位元數以及多項式內差的係數組數都會變多。因此我們訂定了一個振幅的下界值 0.1，圖 5.8 即表示加上這個振幅下界的限制後對系統效能造成的影響，我們可以觀察到這個界線並未對效能產生顯著的影響。圖 5.9-圖 5.11 則為振幅裁切前後的浮點數運算與定點數運算的星狀圖。

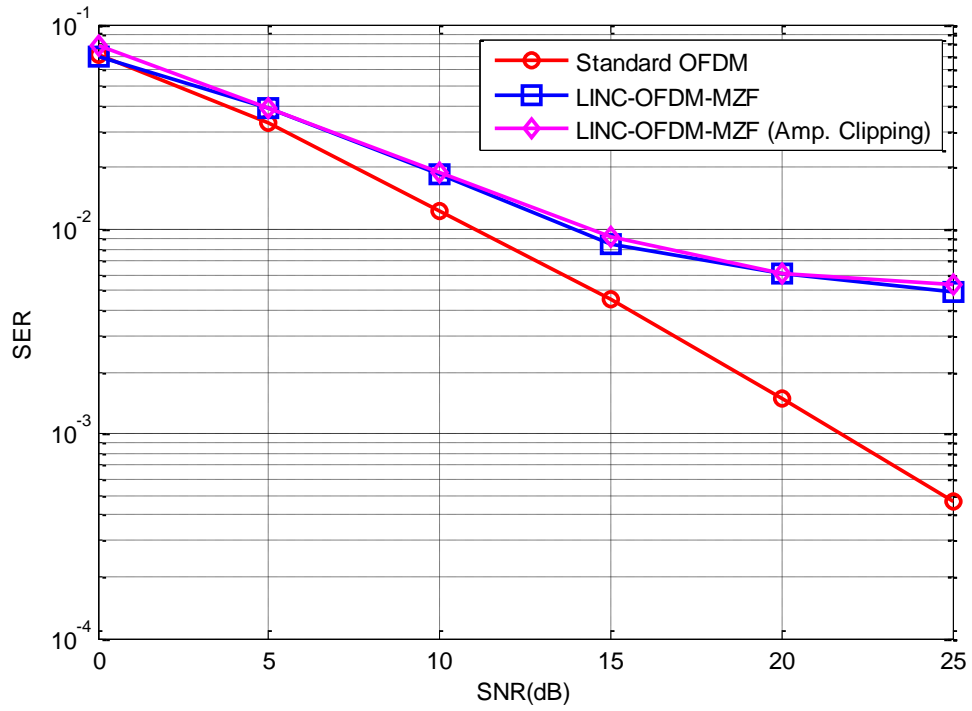


圖 4.9 LINC-OFDM 系統效能: Amplitude Clipping

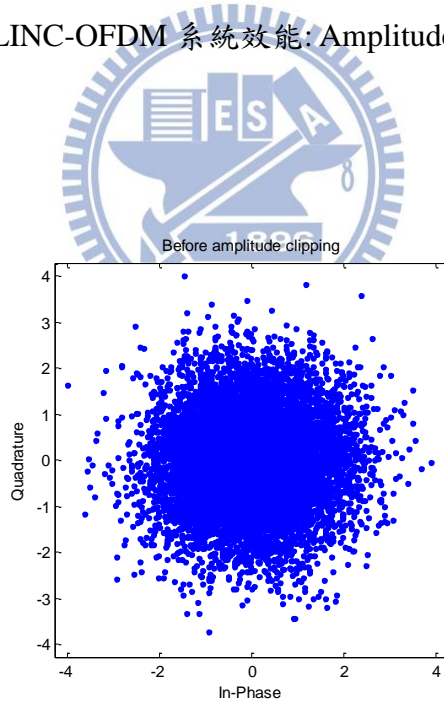


圖 4.10 傳送端原始訊號星座圖

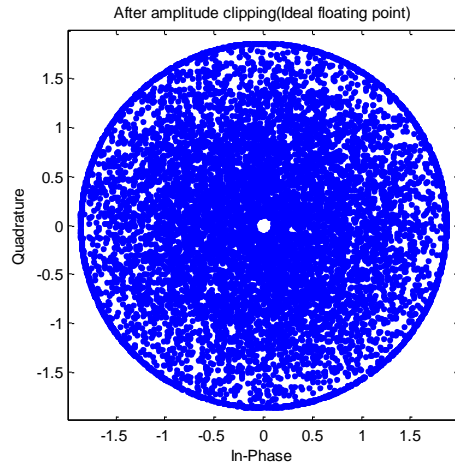


圖 4.11 傳送端訊號縮減星座圖(浮點數運算)

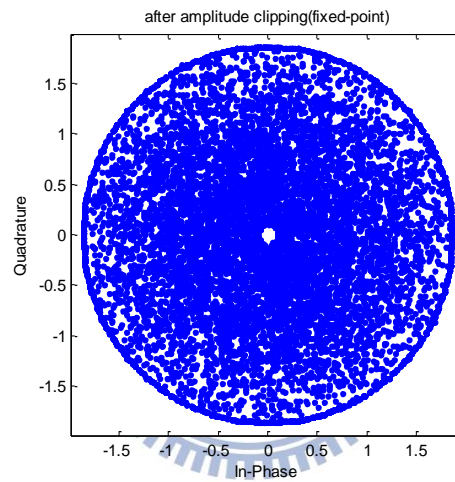


圖 4.12 傳送端訊號縮減星座圖(定點數運算)

接下來的部分，我們將針對 SCS 的設計做模擬，在章節 3.3.5 我們有提到，SCS 的架構我們主要使用多項式內插來逼近錯誤因子的曲線，因此我們需要了解每段用來逼近的多項式係數運算後所造成的誤差大小，圖 5.12 即為使用表 3.1 為係數所造成的訊號與近似誤差比(Signal to Approximation Error Ratio)，透過適當的設計調整，我們最後將錯誤因子曲線分成六部分，每個部份皆以一個二階多項式逼近，最後我們將訊號與近似誤差比控制在 30dB 以上。圖 5.13 為理想上 SCS 的輸出，沒有經過近似且為浮點數運算。圖 5.14 則為透過多項式近似錯誤因子後的 SCS 輸出，且為定點數運算。

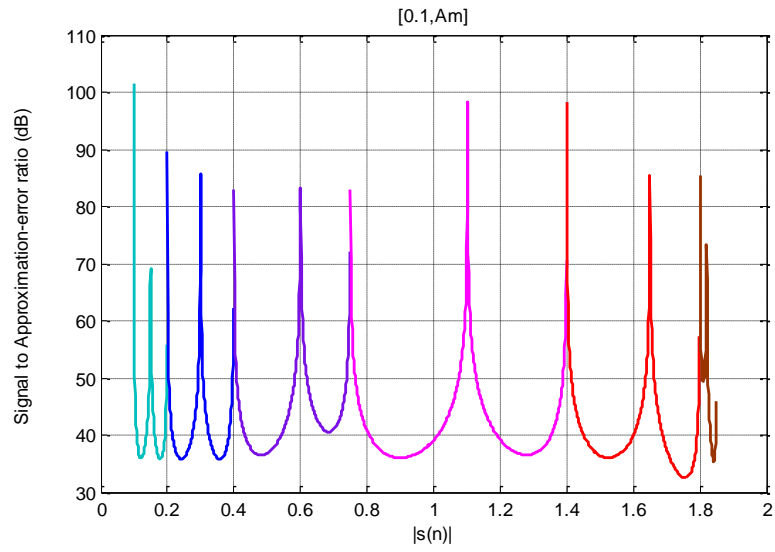


圖 4.13 錯誤訊號之訊號與近似誤差比

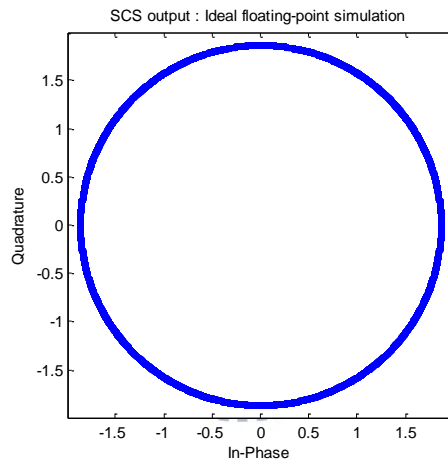


圖 4.14 理想浮點數 SCS 輸出

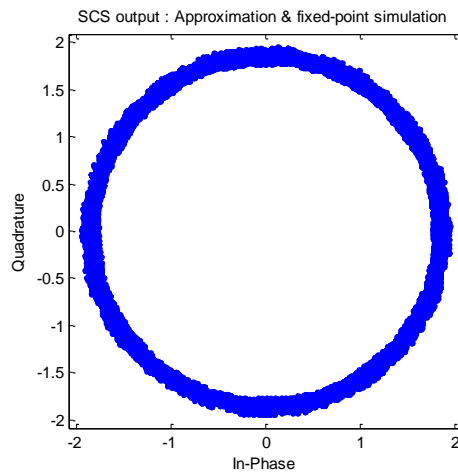


圖 4.15 近似定點數 SCS 輸出

最後，我們將整個 LINC-OFDM 系統做定點模擬，以了解定點數運算對整個系統效能造成的影響。定點模擬主要將所有浮點數轉換成定點數，定點設計的重點在於盡量將位元數降低但效能下降要控制在容許的範圍內，在硬體設計時再對每個訊號根據所設定的位元數做轉換。圖 4.16 為浮點數模擬與定點數模擬的系統效能比較。

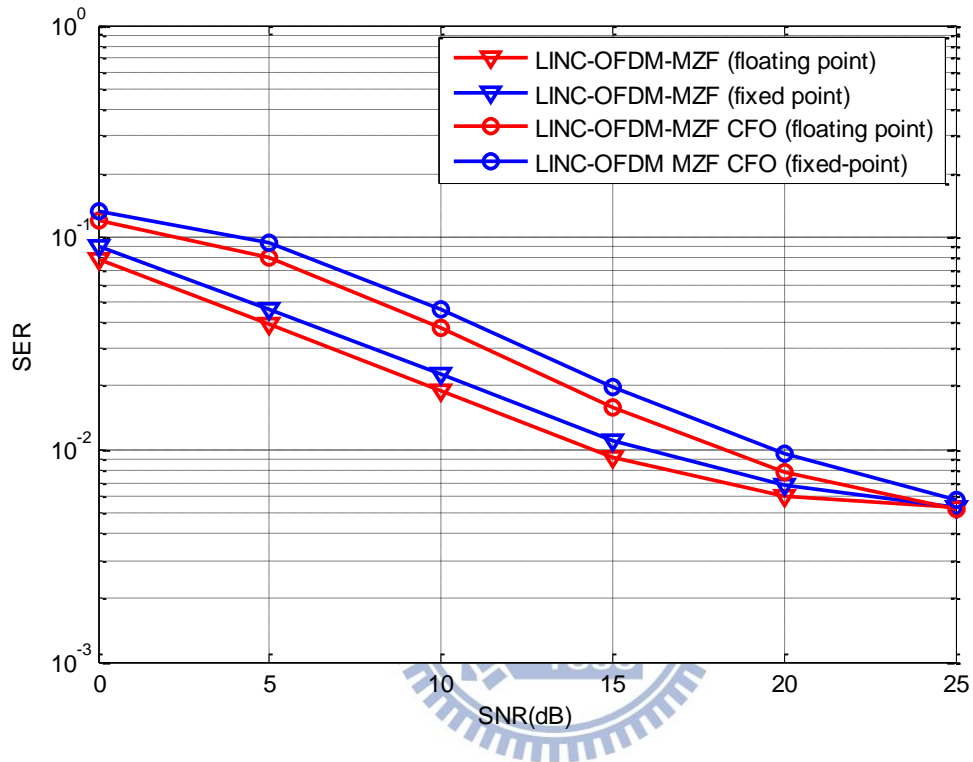


圖 4.16 LINC-OFDM 之浮點與定點模擬

五、FPGA 實現與測量

5.1 平台介紹

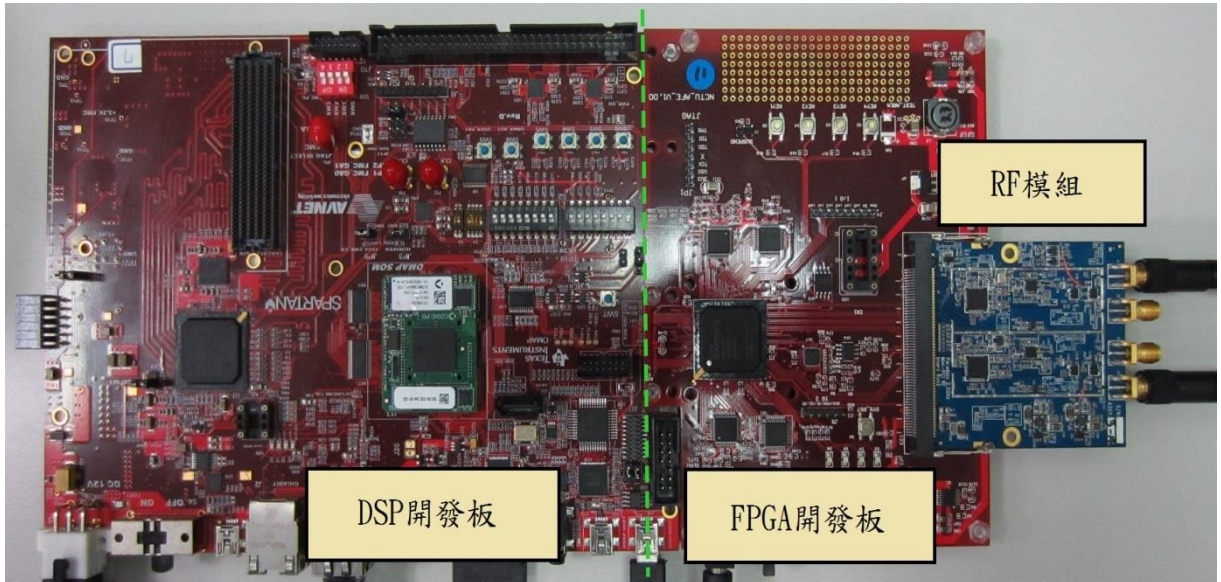


圖 5.1 完整平台

本篇論文所使用的硬體平台如圖 5.1 所示，此平台可分為三個部分，包含了 FPGA 開發板、RF 模組以及 DSP 開發板。圖 5.2 為連結介面，表示開發板或模組連結時所使用的連結介面種類。DSP 開發板為 AVNET S60MAP，由於 DSP 開發板在本論文僅用來將 FPGA 開發板上接收機偵測後的訊號傳送至電腦，再以 Matlab 讀檔繪出訊號的星座圖方便觀察，因此不多加敘述。接下來將針對 FPGA 開發板以及 RF 模組分別作介紹。

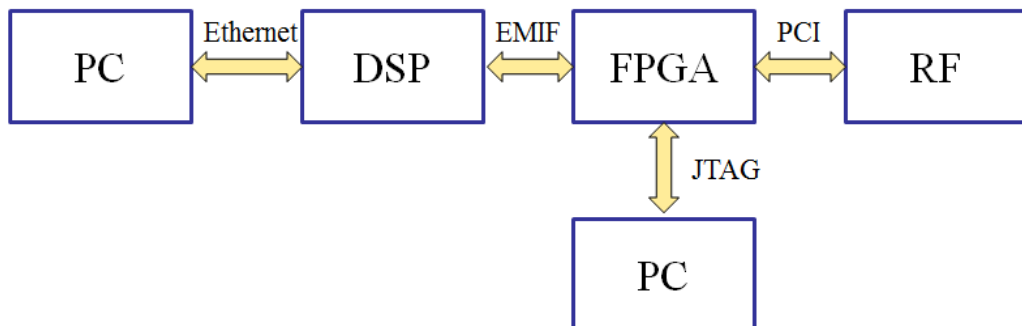


圖 5.2 連結介面

FPGA 開發板如圖 5.3 所示，主要包含了 Xilinx FPGA(Spartan6 XC6SLX150)、DAC (AD9763)、ADC(AD9238)，其中 DAC 為 10 位元的轉換，ADC 為 12 位元的轉換，時脈來源則來自於開發板上的震盪器，震盪頻率為 80MHz，為了產生其他頻率的時脈，我們使用了 Xilinx IPcore 來進行除頻的動作。如圖 5.2 所示，FPGA 與電腦透過 JTAG 連結，可進行電路下載與燒錄，或者在電路運作時以 ChipScope 觀察內部拉線的訊號，另外，開發板上有一 PCI 插座(socket)，用來連接 FPGA 開發板以及 RF 模組。

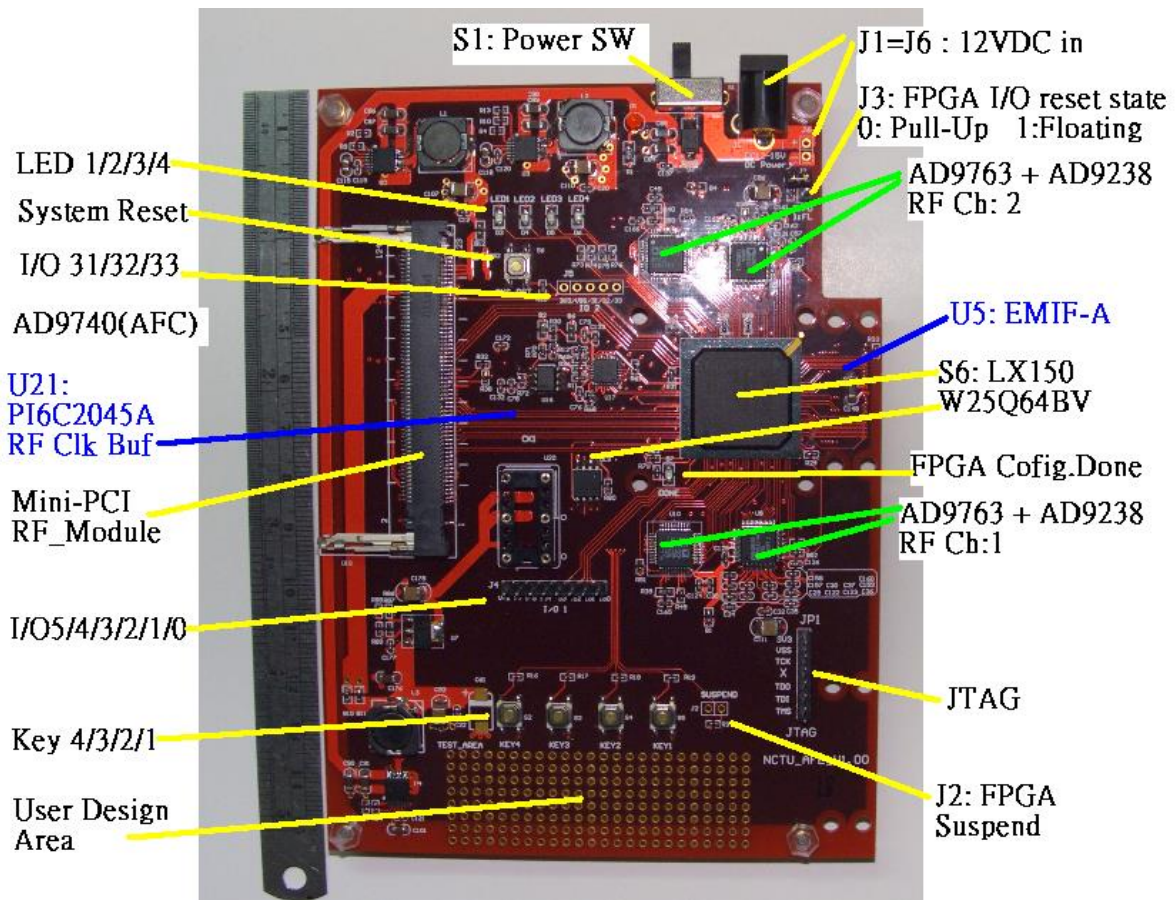


圖 5.3 FPGA 開發版

RF 模組如圖 5.4 所示，RF 模組使用了 MAXIM 的 Max2837，此模組為 2.3GHz 到 2.7GHz 的射頻收發機，包含了兩組傳輸通道，可以用來實現本篇論文 2x1 無合成器之 LINC-OFDM 系統。模組相關的參數設定由 FPGA 開發板控制，如前所述，FPGA 開發板與 RF 模組透過 PCI 介面連結。當電路開啟後，FPGA 開發板會先將 RF 模組的所有參數先設定好，資料要在參數設定好之後才能正確地由 FPGA 開發板通過 RF 模組傳送。



圖 5.4 RF 模組

5.2 FPGA 實現

5.2.1 LINC-OFDM 傳送機實現

我們所設計的傳送機 RTL 架構如圖 5.6 所示。除了章節 3.1 所提到傳送機相關的模組，我們還需要額外 RAM 當作訊號處理前的緩衝區，以及控制電路來控制所有模組的操作時序。控制電路主要用來控制訊號的發送順序，訊號傳送一開始要先將 ROM 裡面儲存的前導訊號依序送出，接下來才送資料，實際控制電路需要兩組，如圖 5.5 所示，第一組處理 SCS 前傳送訊號的排序，第二組主要用來處理 SCS 以後估測通道效應所使用長訓練符元的交錯傳送。

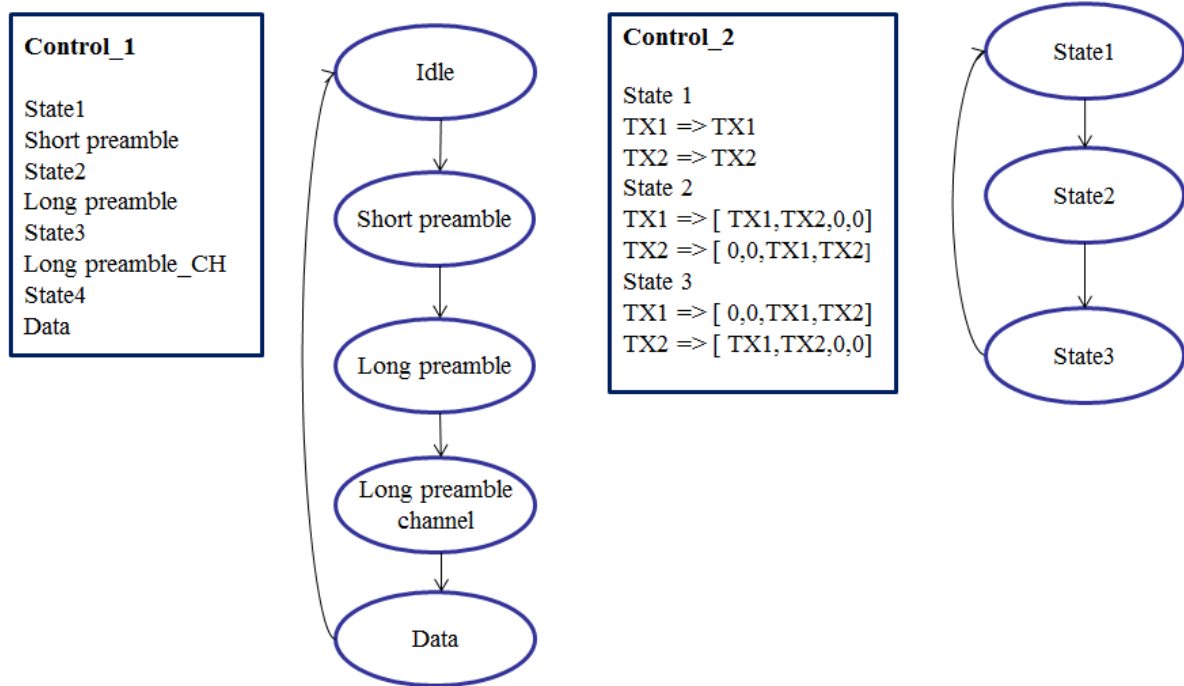


圖 5.5 傳送機控制狀態

圖 5.7 為傳送機的 mapping-report，從此報告中我們可以了解接收機總共使用了多少 FPGA(Spartan6 XC6SLX150)的資源。圖 5.8 為接收機的 timing-report，代表經過合成以後的時序報告，可以讓我們粗略知道合成電路的最大操作頻率，但此階段的報告並不考慮電路繞線後所造成的線路延遲，若將此因素考慮進去計算，即為此電路的 static timing，如圖 5.9 所示，我們可以發現加了繞線的延遲後，static timing 的最大操作頻率確實比 timing-report 小。


```

Slice Logic Utilization:
  Number of Slice Registers:          2,131 out of 184,304    1%
    Number used as Flip Flops:        2,091
    Number used as Latches:           1
    Number used as Latch-thrus:       0
    Number used as AND/OR logics:     39
  Number of Slice LUTs:              3,764 out of 92,152    4%
    Number used as logic:             3,339 out of 92,152    3%
      Number using O6 output only:    2,009
      Number using O5 output only:    140
      Number using O5 and O6:         1,190
      Number used as ROM:              0
    Number used as Memory:            303 out of 21,680    1%
      Number used as Dual Port RAM:    32
        Number using O6 output only:  0
        Number using O5 output only:  0
        Number using O5 and O6:       32
      Number used as Single Port RAM:  0
      Number used as Shift Register:   271
        Number using O6 output only:  57
        Number using O5 output only:  0
        Number using O5 and O6:       214
      Number used exclusively as route-thrus: 122
      Number with same-slice register load: 111
      Number with same-slice carry load: 10
      Number with other load:         1

Slice Logic Distribution:
  Number of occupied Slices:          1,239 out of 23,038    5%
  Number of LUT Flip Flop pairs used: 4,000
    Number with an unused Flip Flop:  2,417 out of 4,000    60%
    Number with an unused LUT:        236 out of 4,000    5%
    Number of fully used LUT-FF pairs: 1,347 out of 4,000    33%
  Number of unique control sets:      30
  Number of slice register sites lost to control set restrictions: 87 out of 184,304    1%

IO Utilization:
  Number of bonded IOBs:              63 out of 338    18%

Specific Feature Utilization:
  Number of RAMB16BWERS:              0 out of 268    0%
  Number of RAMB8BWERS:               1 out of 536    1%
  Number of BUFIO2/BUFIO2_CLKs:       0 out of 32    0%
  Number of BUFIO2FB/BUFIO2FB_CLKs:   0 out of 32    0%
  Number of BUFG/BUFGMUXs:            3 out of 16    18%
    Number used as BUFGs:              3
    Number used as BUFGMUX:            0
  Number of DCM/DCM_CLKGENs:          0 out of 12    0%
  Number of ILOGIC2/ISERDES2s:        0 out of 586    0%
  Number of IODELAY2/IODRP2/IODRP2_MCBs: 0 out of 586    0%
  Number of OLOGIC2/OSERDES2s:        0 out of 586    0%
  Number of BSCANs:                   0 out of 4    0%
  Number of BUFHs:                    0 out of 384    0%
  Number of BUFPLLs:                   0 out of 8    0%
  Number of BUFPLL_MCBs:               0 out of 4    0%
  Number of DSP48A1s:                  25 out of 180    13%
  Number of ICAPs:                     0 out of 1    0%
  Number of MCBs:                      0 out of 4    0%
  Number of PCILOGICSEs:               0 out of 2    0%
  Number of PLL_ADVs:                  0 out of 6    0%
  Number of PMVs:                      0 out of 1    0%
  Number of STARTUPS:                  0 out of 1    0%
  Number of SUSPEND_SYNCS:             0 out of 1    0%

Average Fanout of Non-Clock Nets:     2.73

Peak Memory Usage: 670 MB
Total REAL time to MAP completion: 3 mins 38 secs
Total CPU time to MAP completion: 3 mins 37 secs

```

圖 5.7 傳送機 mapping-report

```

Timing Summary:
-----
Speed Grade: -3

Minimum period: 23.164ns (Maximum Frequency: 43.170MHz)
Minimum input arrival time before clock: 9.130ns
Maximum output required time after clock: 3.597ns
Maximum combinational path delay: No path found

```

圖 5.8 傳送機 timing-report

```

Timing summary:
-----

Timing errors: 0 Score: 0 (Setup/Max: 0, Hold: 0)

Constraints cover 265828784197916 paths, 0 nets, and 15908 connections

Design statistics:
  Minimum period: 23.294ns{1} (Maximum frequency: 42.930MHz)
  Minimum input required time before clock: 10.491ns
  Minimum output required time after clock: 9.038ns

```

圖 5.9 傳送機 static timing

接下來我們針對傳送機做RTL模擬，模擬方法如圖 5.10 所示，傳送機電路經由 Xilinx ISim 模擬器執行模擬，然後將結果寫到檔案，並經由 Matlab 的浮點數運算的接收機進行處理，最後察看結果，我們假設傳送機送出的訊號並未經過通道和雜訊的干擾，結果如圖 5.11 所示，由於傳送機的訊號經過定點數運算與近似運算，造成接收機偵測後訊號的星座圖會有稍微散開的現象。

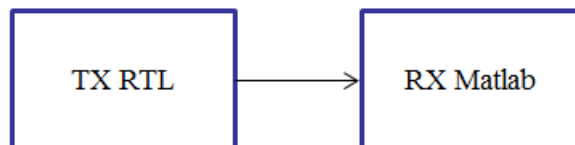


圖 5.10 傳送機 RTL 模擬測試

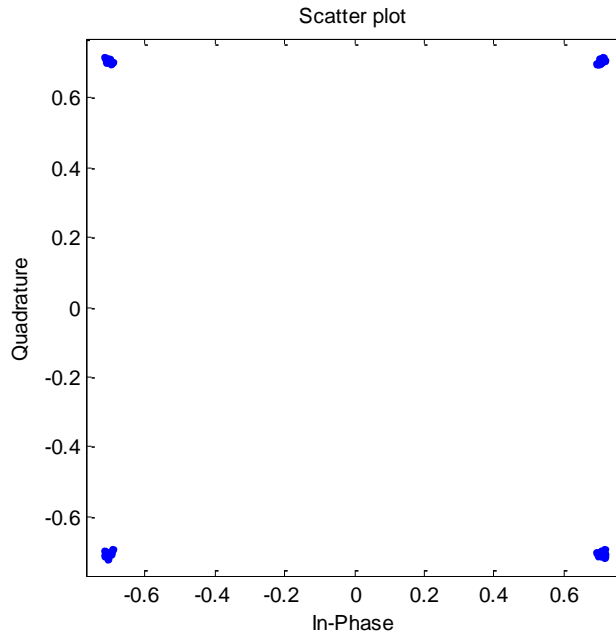


圖 5.11 RTL 模擬: 傳送機

5.2.2 LINC-OFDM 接收機實現

我們所設計的接收機 RTL 架構如圖 5.12 所示，圖 5.13 為 mapping-report，圖 5.14 及圖 5.15 則分別為 timing-report 與 static timing。接收機一樣需要一個控制電路以及 RAM 當緩衝的儲存空間。控制電路主要用來控制同步模組的運作時間，如封包偵測後，要進行粗略頻率偏移估測，接著進行角度補償，然後符元起點估計..等等。接收機的 RAM 主要分成兩部分，第一個部分為做 FFT 之前，這是由於用來通道估測的前導訊號需要做重組，但訊號又是不斷的進來，因此需要 RAM 來當緩衝空間。第二個 RAM 為做完 FFT 以後，通道估測後的通道係數有兩組，分別代表兩根天線的通道狀況，我們把係數及訊號存在第二個 RAM 的空間，等到 MZF 等化器係數運算完成後再進行等化。值得一提的是在傳送機與接收機所使用的 RAM，為了節省存取空間，我們使用了乒乓緩衝器 (ping-pong buffering) 的架構，將 RAM 一分為二，如圖 5.16 所示，讓訊號的讀取以及寫入可以同時進行。

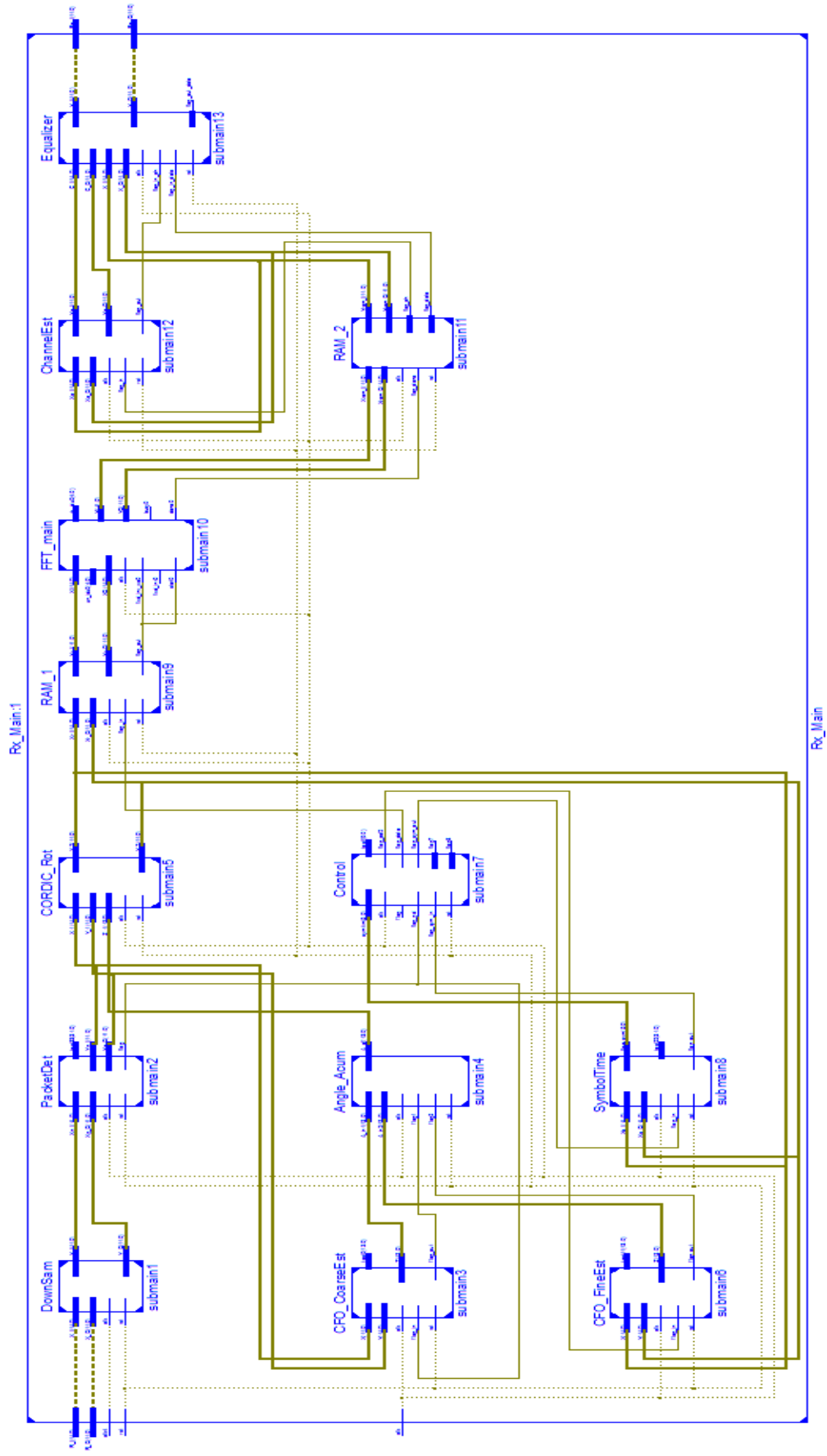


圖 5.12 接收機 RTL 架構圖

Slice Logic Utilization:

Number of Slice Registers:	2,931 out of 184,304	1%
Number used as Flip Flops:	2,683	
Number used as Latches:	0	
Number used as Latch-thrus:	0	
Number used as AND/OR logics:	248	
Number of Slice LUTs:	7,941 out of 92,152	8%
Number used as logic:	7,700 out of 92,152	8%
Number using O6 output only:	3,624	
Number using O5 output only:	630	
Number using O5 and O6:	3,446	
Number used as ROM:	0	
Number used as Memory:	50 out of 21,680	1%
Number used as Dual Port RAM:	0	
Number used as Single Port RAM:	0	
Number used as Shift Register:	50	
Number using O6 output only:	30	
Number using O5 output only:	0	
Number using O5 and O6:	20	
Number used exclusively as route-thrus:	191	
Number with same-slice register load:	145	
Number with same-slice carry load:	44	
Number with other load:	2	

Slice Logic Distribution:

Number of occupied Slices:	2,561 out of 23,038	11%
Number of LUT Flip Flop pairs used:	8,717	
Number with an unused Flip Flop:	6,324 out of 8,717	72%
Number with an unused LUT:	776 out of 8,717	8%
Number of fully used LUT-FF pairs:	1,617 out of 8,717	18%
Number of unique control sets:	35	
Number of slice register sites lost to control set restrictions:	103 out of 184,304	1%



IO Utilization:

Number of bonded IOBs:	157 out of 338	46%
------------------------	----------------	-----

Specific Feature Utilization:

Number of RAMB16BWERS:	0 out of 268	0%
Number of RAMB8BWERS:	0 out of 536	0%
Number of BUFIO2/BUFIO2_2CLKs:	0 out of 32	0%
Number of BUFIO2FB/BUFIO2FB_2CLKs:	0 out of 32	0%
Number of BUFG/BUFGMUXs:	2 out of 16	12%
Number used as BUFGs:	2	
Number used as BUFGMUX:	0	
Number of DCM/DCM_CLKGENs:	0 out of 12	0%
Number of ILOGIC2/ISERDES2s:	0 out of 586	0%
Number of IODELAY2/IODRP2/IODRP2_MCBs:	0 out of 586	0%
Number of OLOGIC2/OSERDES2s:	0 out of 586	0%
Number of BSCANS:	0 out of 4	0%
Number of BUFHs:	0 out of 384	0%
Number of BUFPLLs:	0 out of 8	0%
Number of BUFPLL_MCBs:	0 out of 4	0%
Number of DSP48A1s:	113 out of 180	62%
Number of ICAPs:	0 out of 1	0%
Number of MCBs:	0 out of 4	0%
Number of PCILOGICSEs:	0 out of 2	0%
Number of PLL_ADVs:	0 out of 6	0%
Number of PMVs:	0 out of 1	0%
Number of STARTUPs:	0 out of 1	0%
Number of SUSPEND_SYNCs:	0 out of 1	0%

Average Fanout of Non-Clock Nets: 2.42

Peak Memory Usage: 791 MB
 Total REAL time to MAP completion: 5 mins 27 secs
 Total CPU time to MAP completion: 5 mins 21 secs

圖 5.13 接收機 mapping-report


```

Timing Summary:
-----
Speed Grade: -3

Minimum period: 26.491ns (Maximum Frequency: 37.748MHz)
Minimum input arrival time before clock: 6.362ns
Maximum output required time after clock: 3.732ns
Maximum combinational path delay: No path found

```

圖 5.14 接收機 timing-report

```

Timing summary:
-----

Timing errors: 0 Score: 0 (Setup/Max: 0, Hold: 0)

Constraints cover 578313999376312 paths, 0 nets, and 32400 connections

Design statistics:
Minimum period: 34.125ns{1} (Maximum frequency: 29.304MHz)
Minimum input required time before clock: 11.577ns
Minimum output required time after clock: 11.849ns

```



圖 5.15 接收機 static timing

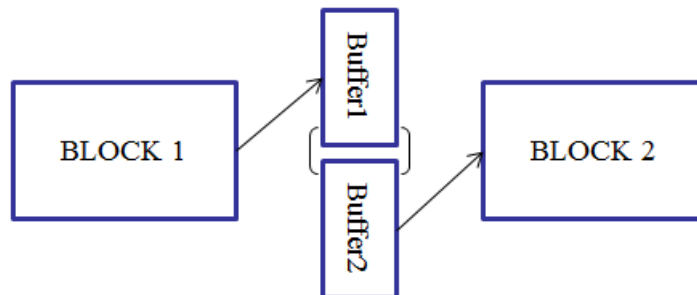


圖 5.16 ping-pong buffering

接下來我們針對接收機做 RTL 模擬，模擬方法如圖 5.17 所示，傳送機訊號我們由 Matlab 浮點數運算產生，並寫入檔案，讓接收機電路經由 Xilinx ISim 模擬器讀檔進行模擬，然後再將結果寫到檔案，並經由 Matlab 讀檔觀察結果，為了瞭解接收機各個同步模組是否正常運作，我們加入了多路徑通道以及頻率偏移效應，假設通道路徑為 3 且

相關係數為 1，頻率偏移為子載波間距的 0.01 倍，因為未加雜訊，若以浮點數系統模擬這個例子，會集中於 QPSK 四個星座點上，RTL 模擬結果如圖 5.18 所示，我們可以觀察到受定點數運算的影響，星座點並非完全集中於四點。

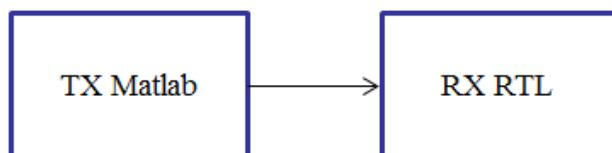


圖 5.17 接收機 RTL 模擬測試

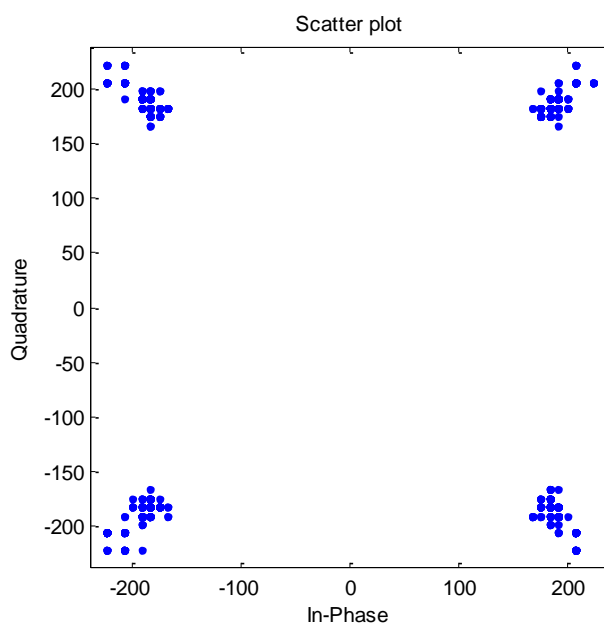


圖 5.18 RTL 模擬:接收機

5.3 測量結果

我們將傳送機與接收機的電路與 FPGA 開發板上的平台連接電路結合，並下載 (download)到 FPGA 開發板上，為了操作方便，傳送訊號位元我們直接存在 FPGA 的 ROM 上，封包要開始傳送時再去讀取。觀察的工具則包含了 Xilinx 電路設計軟體上的 Chip-Scope，直接將訊號拉線存進 FPGA 的 RAM 空間，再由該軟體將訊號輸出觀察，另外為了方便觀察偵測後接收訊號的星狀圖，我們將 FPGA 開發板透過 EMIF 介面連結另一套

DSP 開發板，透過該平台將訊號讀取至電腦，再直接用 Matlab 畫出星狀圖觀察，由於 DSP 開發板並非本論文重點，在此並不多加敘述。在此為了與 DSP 模組的速率配合，我們將操作頻率設定為 5M(提高取樣前)與 20M(提高取樣後)，此頻率皆小於傳送機與接收機電路的最大操作頻率。

首先為傳送機訊號觀測，我們將傳送機的訊號經由 RF 模組打出，將兩根天線的訊號由電纜(cable)分別接至接收機，並由接收機 RF 模組降頻處理，如圖 5.19 所示，我們在此以 ChipScope 觀察訊號，確定訊號通過 RF 模組的升頻與降頻後的可順利接收後再做天線的傳輸，圖 5.21 與圖 5.22 顯示傳送的訊號，我們可以看到兩天線所送的訊號在封包中間皆有一段區域不送訊號，該區域是為了估測到兩個通道響應而使訊號交錯的傳送，意即有一根天線送訊號，另一根則不送。接著我們將電纜換上天線，使用兩根天線打出訊號，讓訊號在空氣中自行結合，再由一根天線收回，如圖 5.20 所示，我們由圖 5.23 可以觀察到原本兩根天線所打出的訊號皆有一段前導訊號不送資料，訊號在空氣中結合收回後，兩天線用來估測通道效應的前導訊號區塊產生互補而將空區塊填滿。

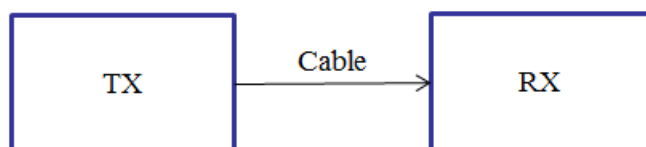


圖 5.19 傳送機電纜測試

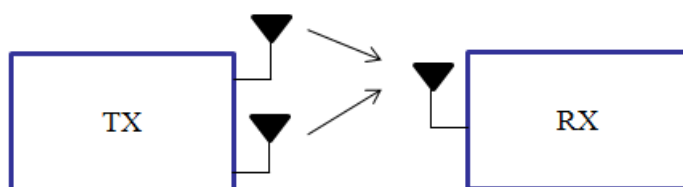


圖 5.20 傳送機天線測試

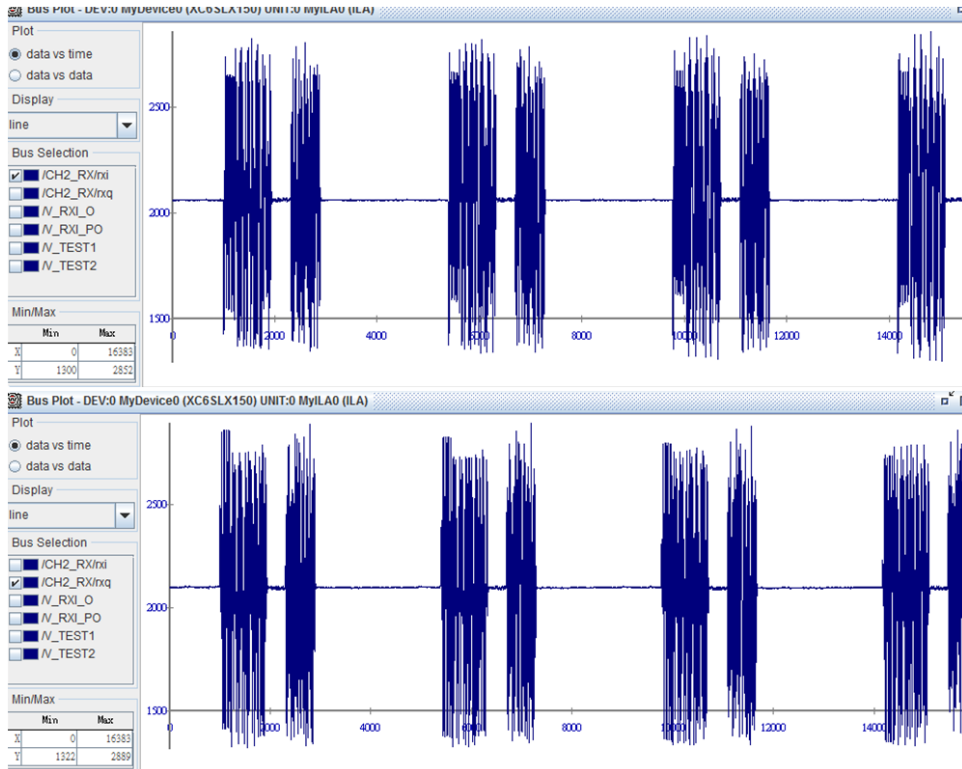


圖 5.21 傳送機 TX1 訊號(Cable)

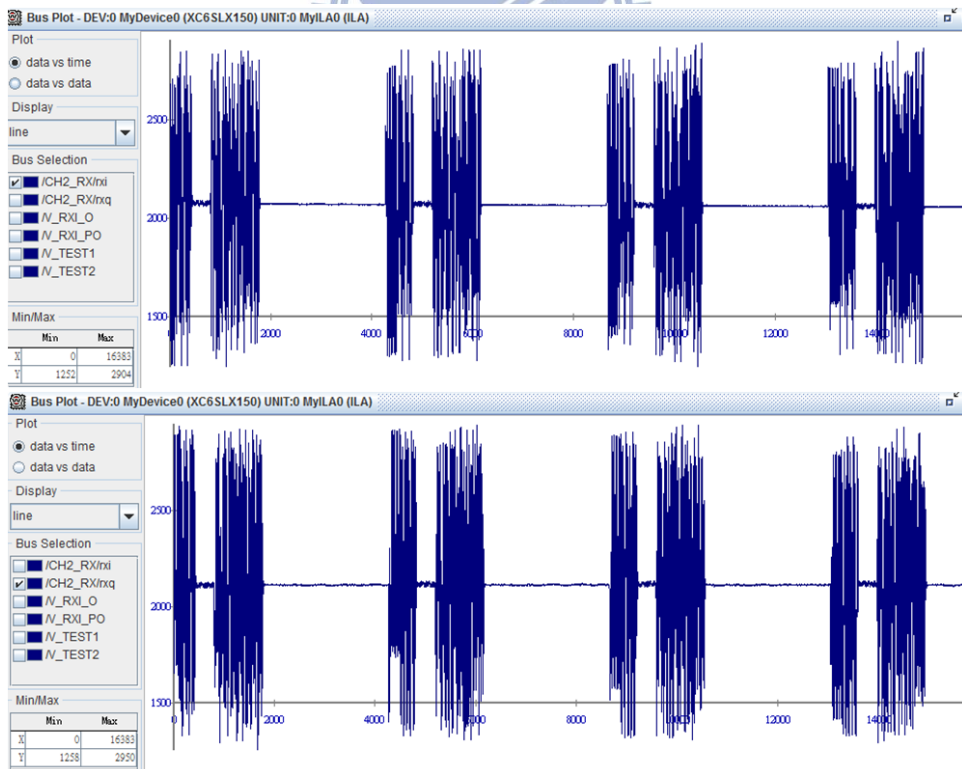


圖 5.22 傳送機 TX2 訊號(Cable)

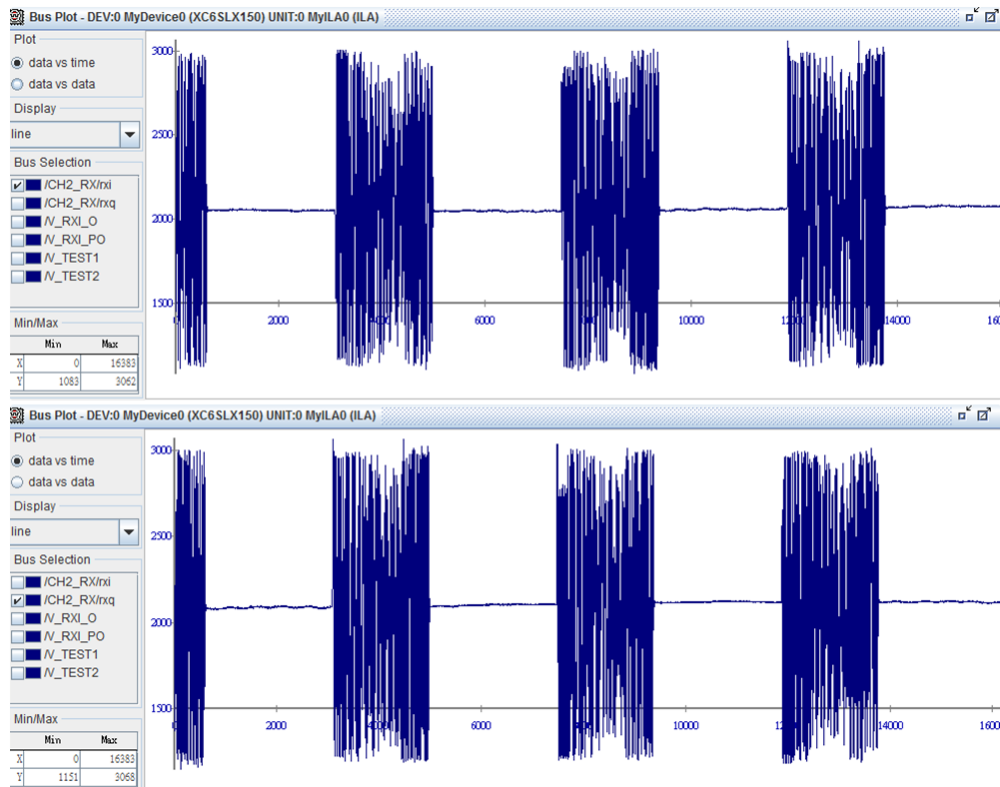


圖 5.23 傳送機訊號(Antenna)

接下來為接收機部分，圖 5.24 為傳送天線與接收天線的擺設位置。圖 5.25 為接收機控制模組的旗幟(flag)訊號，由此圖我們可以瞭解各同步模組是否運作，以及其運作的時間點。圖 5.26 為訊號經過 MZF 等化器偵測後的星狀圖，此圖與圖 5.18 及圖 5.11 的 RTL 傳送機與接收機的模擬比較，星座圖散開的程度大很多，但此現象其實是可以預見，因為經由天線將訊號以電磁波的形式送出，通道狀況與相關性並非我們所能控制，另一個因素為 RF 模組上類比元件的不穩定性以及其所造成的非理想效應(impairment)，例如 IQ 失調(imbalance)、混頻器(mixer)所產生的相位雜訊、DC 偏移(offset)..等等，都會對系統的效能造成影響。



圖 5.24 天線擺設位置

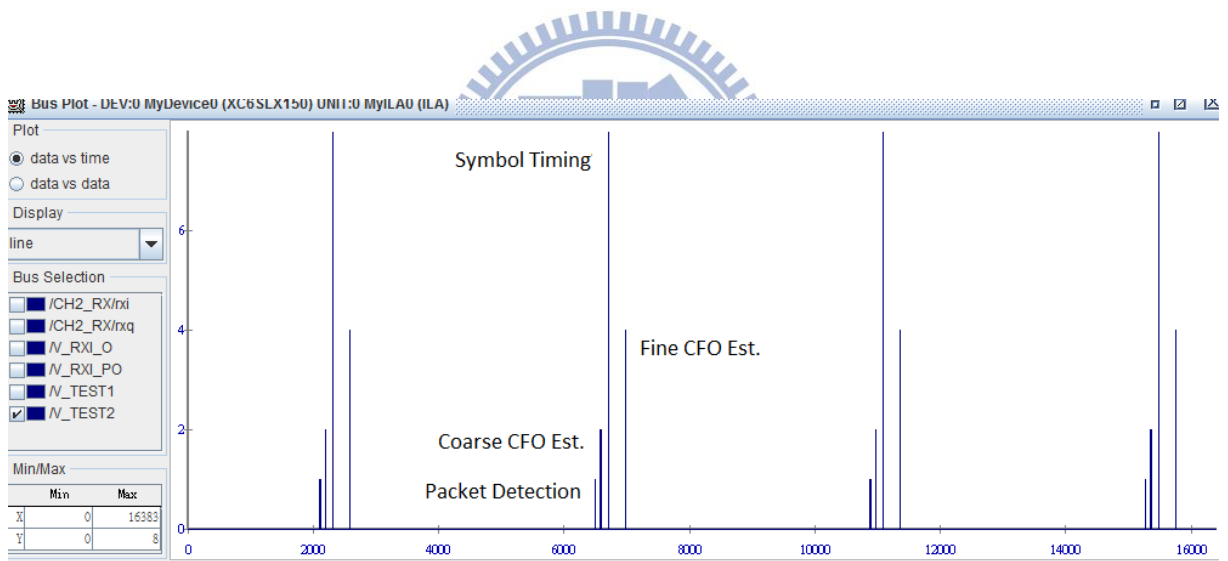


圖 5.25 接收機控制模組 flag 訊號

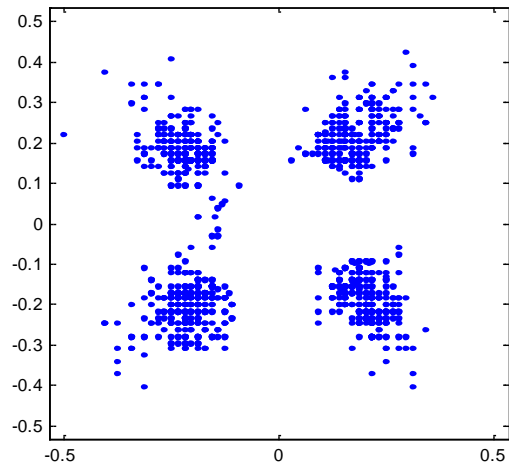
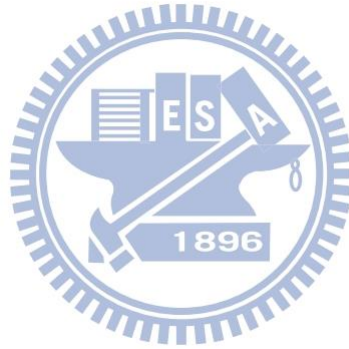


圖 5.26 接收機偵測訊號星座圖



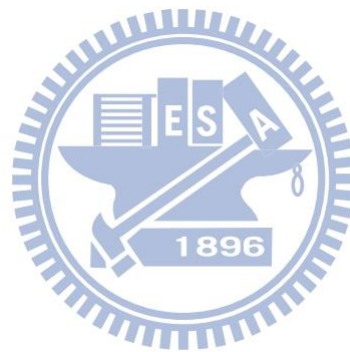
六、結論

LINC 系統可以解決 OFDM 調變技術中高 PAPR 的問題，而無合成器的設計可以避免傳統 LINC 系統中功率合成器的諸多缺點。本篇論文主要目的在討論無合成器之 LINC-OFDM 系統的硬體架構，並以 FPGA 設計流程使用 VHDL 做系統之實現，整個設計包含了傳送機與接收機。首先在第二章我們介紹了傳統 LINC 系統的架構以及無合成器 LINC 系統的架構，並介紹了無合成器 LINC-OFDM 系統的偵測方法。第三章則是先對硬體設計重要功能區塊做介紹，包含了 CORDIC 演算法以及 FFT 模組，接著介紹傳送機架構，主要包含了振幅裁切、取樣提高、SCS 以及前導訊號模組，最後我們討論了接收機的演算法與架構，接收機主要分為同步電路以及偵測電路兩部分，同步電路包含了封包偵測、符元起點估計、載波頻率估計以及通道估測，偵測電路的部份我們使用了較容易實現的 MZF 等化器。在第四章中，我們先以浮點數運算模擬了 LINC-OFDM 系統的系統效能，接著將重要模組的以定點運算做了系統模擬。在第五章中，我們先介紹了使用的硬體實現平台，接著將先前所介紹的硬體架構以硬體描述語言實現，最後將電路下載到 FPGA 開發板，藉由實際的天線發送，接收，測量。

從第五章的實測當中，我們可以發現，在數位域模擬與測試的電路和通過類比處理以及實際環境後的測試結果不盡相同，因為數位電路的模擬測試以電路邏輯與模組間操作時序的正確性為主，並未考慮很多類比環境造成的影響，這些影響包含了 RF 模組的類比電路造成的非理想效應，另外本論文假設 LINC 系統之通道相關性要很高，實際的通道不一定符合這個假設，如何在通道響應差異較大時仍然能保持效能是無合成器 LINC-OFDM 系統的重要的議題。[8] 提出了一個具編碼之 LINC-OFDM 系統，可以有效的維持 OFDM 系統效能至相關性為 0.96，如果要能夠應用在更低的相關性環境則需進一步的研究。

本篇論文我們實現了無合成器 LINC-OFDM 系統的傳收機，從浮點數系統效能模擬，到電路設計以及 RTL 模擬，最後下載到 FPGA 平台透過 RF 模組實際收發訊號。由於本

篇論文的接收機實現著重在前端的同步處理，訊號偵測部分僅使用 MZF 等化器，如前所述，無合成器之 LINC-OFDM 的效能受到兩根傳送天線的通道相關性影響甚大，[8] 提出了一個具編碼之 LINC-OFDM 系統，可以有效的解決的通道相關性較低時的問題，未來可以考慮將此系統實現。



參考文獻

- [1] D. C. Cox, "Linear Amplification with Nonlinear Components," *IEEE Transactions on Communications*, Vol. 22, pp. 1942- 1945, December 1974.
- [2] Birafane, A., Kouki, A.B., "Sources of linearity degradation in LINC transmitters for hybrid and outphasing combiners," *Electrical and Computer Engineering*, Vol. 1, pp.547-550, May 2004
- [3] F. H. Raab, P. Asbeck, S. Cripp et-al., "RF and Microwave power amplifier and technologies," *Summit Technical Media, LLC*, November 2003
- [4] D. M. Pozar, *Microwave engineering*, NJ: Wiley, pp. 318-323, 2005
- [5] Abdelaal, M.M.: "LINC based amplifier architectures for power efficient wireless transmitters," Ph. D. Thesis, 2009, Ecole Polytechnique (Montreal, Canada).
- [6] S. Ali, B. Adebisi, G. Markarian and E. Arikan, "Signal combining in LINC amplifier using Alamouti codes," *Electronics Letters*, Vol. 46, No. 18, September 2010
- [7] K.-S. Hsu, "Maximum Likelihood Detection for Combinerless LINC-OFDM Systems," M.S Thesis, NCTU, June 2011
- [8] Y.-P. Hsu, "Maximum Likelihood Detector for Coded Combinerless LINC-OFDM Systems," M.S Thesis, NCTU, July 2012
- [9] J. Terry, J. Heiskala, *Ofdm Wireless Lans: A Theoretical and Practical Guide*, Sams Indianapolis, IN, USA ,2001
- [10] B.-J. Chuang, "Design and Implementation of IEEE 802.11n Baseband Receiver," M.S Thesis, NCTU, July 2005
- [11] Z.-S. Lin, "Design and Implementation of the IEEE 802.11a Receiver with Channel Tracking," M.S Thesis, NCU, July 2005
- [12] T.-D. Chiueh, P.-Y. Tsai, *OFDM Baseband Receiver Design for Wireless Communications*, John Wiley & Sons (Asia) Pte Ltd. ,2007

- [13] “DS260 LogiCORE IP Fast Fourier Transform v7.1,” Xilinx Product Specification, DS260, March 1, 2011
- [14] “ISim Hardware Co-Simulation Tutorial: Accelerating Floating Point FFT Simulation,” Xilinx Product Specification, UG817 (v14.1) ,April 24, 2012
- [15] K.-Y. Jheng, Y.-J. Chen, and A.-Y. Wu, “Multilevel LINC System Designs for Power Efficiency Enhancement of Transmitters,” *IEEE Journal of Selected Topics in Signal Processing* ,Vol. 3,No. 3, June 2009
- [16] O. Hammi, S. Boumaiza, F. Ghannouchi, “Design and Optimization of Digital Signal Components Separator of LINC Transmitters Using FPGA Processors,” *IEEE International Conference on Signal Processing and Communications*, 24-27 November 2007
- [17] C. W. Farrow, “A Continuously Variable Digital Delay Element,” *IEEE International Symposium on Circuits and Systems* , 1988
- [18] F. Harris, “Performance and Design of Farrow Filter Used for Arbitrary Resampling,” *IEEE Digital Signal Processing Proceedings*, Vol. 2, 1997
- [19] V. A. Pedroni, *Circuit Design With VHDL*, The MIT Press ,August 2004
- [20] Y. S. Cho, J. Kim, W. Y. Yang, C. G. Kang, *MIMO-OFDM Wireless Communications with Matlab*, Wiley, pp.84-89, 2010