

國防部一一三年度「補助軍事院校教師（官）從事學術研究」

研究計畫名稱：相控陣列天線教學展示系統
之設計與教學

委 託 單 位：國防部

研 究 單 位：陸軍軍官學校

研究計畫主持人：陳宏圖

協 同 研 究 人：

中華民國 一一三年 十二 月 二十七 日

目次

摘要.....	1
第一章序論.....	2
1.1 前言.....	2
1.2 研究背景與目的.....	4
第二章研究內容與成果.....	8
2.1 微波震盪與功率放大電路.....	11
2.2 功率分配網路.....	12
2.3 相移器及其控制電路.....	16
2.4 天線單元與陣列天線.....	26
2.5 電源供應電路.....	31
2.6 接收指示器.....	32
第三章結論.....	36
參考文獻.....	37
附錄一、樹莓派控制器之 Python 程式.....	39
附錄一、接收指示器之 Arduino 程式.....	43

摘要

雷達(RADAR)是現代電子科技和電腦應用技術快速發展的重要成就之一。雷達的英文原意是無線電偵測及定距(RAdio Detection And Ranging)。最初在1930年代初期開始發展時，僅是一個簡單的系統，用於偵測敵方飛機。研發至今，雷達已演變成一個整合雷達偵測器(radar sensor)與電腦(computer)的自動化系統。應用範圍也不再侷限於軍事用途。舉凡太空飛行物速度和軌道的量測、精密跟踪、導航、測繪攝影、空中交通管制、氣象預報、資源探勘等，均是現代雷達的應用領域。

申請人在112年度執行「相控陣列天線之設計與教學」研究計畫，過程中當我們嘗試將各部組合成一個完整的「相控陣列天線」系統時，發現其波束偏轉現象只能透過場型量測系統才能觀察到。也就是說雖然波束的偏轉功能是有了，但如果每次在教學時都得在無反射室內講解，就很難讓學生產生具體的觀念，教學成效將大打折扣。因此本年度的計畫，我們完成了一套「相控陣列天線教學展示系統」，透過簡單的微波電路設計及微電腦控制系統，可以讓學生更直覺地看到陣列主波束因為天線單元間相位差而偏轉，更能實地瞭解「相控陣列天線」在相控雷達系統中角色與其應用。

第一章

序論

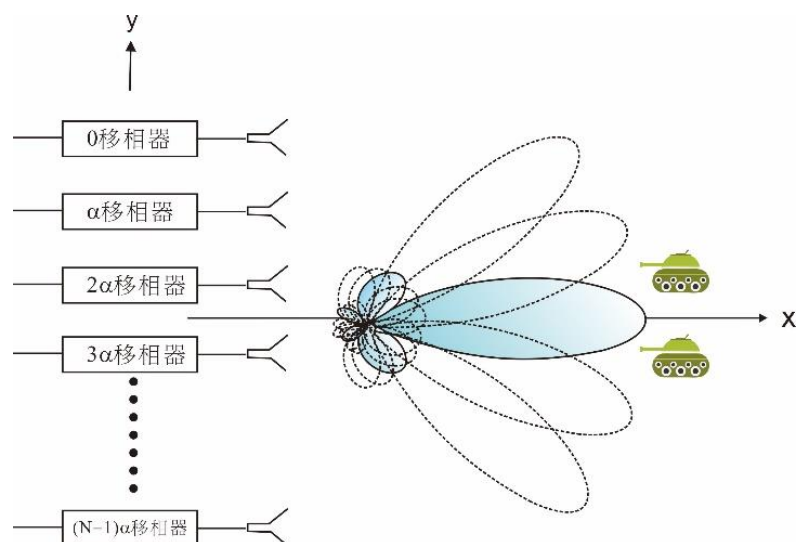
1.1 前言

連續波雷達乃透過持續發射電磁波，由雷達回波的有無來判斷偵測範圍內是否有目標物。連續波雷達所發射的電磁波，其頻率可以是固定或調變兩種方式。當發射頻率固定時，接收並分析反射波的頻率差，可以算出目標與雷達的相對速度。若發射的雷達波屬頻率調變的方式，分析雷達波與回波的頻率偏移(Frequency Shift)，即可算出目標物與雷達間的距離[1-5]。

陸軍軍官學校電機系依據國軍資通發展趨勢，規劃「電波通訊」、「計算機控制」、及「光電與電子」為本系發展重點。就「電波通訊」領域而言，申請人曾獲國防部補助，於107年執行「ISM頻段雷達之設計與教學」、109年執行「6GHz頻段合成孔徑雷達之設計與教學」、110年執行「滑軌式合成孔徑雷達成像系統之設計與教學」、111年度執行「切換天線式陣列雷達成像系統之設計與教學」、及112年度執行「相控陣列天線之設計與教學」等五項研究計畫。透過此五項計畫之執行，成功地將教學內容往「雷達工程」、「合成孔徑雷達」乃至「相控陣列天線」領域推進。過程中我們一直精進雷達設計，從手工移動式掃描、機械化自動掃描、乃至111年度以4×4切換天線式陣列來達到電子式掃描的功能。而112年度則更進一步以「相控陣列天線」為教學研究主體，以精進本校電機系「電波專題」課程，期使學生嫻熟雷達系統原理。各年度的計畫成果都已經上繳，在此不多贅述。

事實上，雷達成像品質的好壞，取決於雷達的「距離解析度(range resolution)」及「方位解析度(azimuth resolution)」。「距離解析度」是指在相同的角度下，雷達能分辨出不同距離目標物的能力；而「方位解析度」是指在相同距離下，雷達能分辨出不同角度目標物的能力[1-6]。在前幾年執行有關「合成孔徑雷達」計畫時，無論是人工定點掃描、或是機械載台掃描、乃至切換陣列式的掃描，

其原理都是讓雷達天線的主波束在偵測空間中做掃描。相較於比較緩慢、耗時、且不精準的機械式掃描，電子式掃描才是目前雷達系統的主流技術。而以目前的技術而言，如圖一所示的「相控陣列雷達 (Phased-Array Radar)」是目前電子式掃描技術中，最尖端也最具代表性的技術。

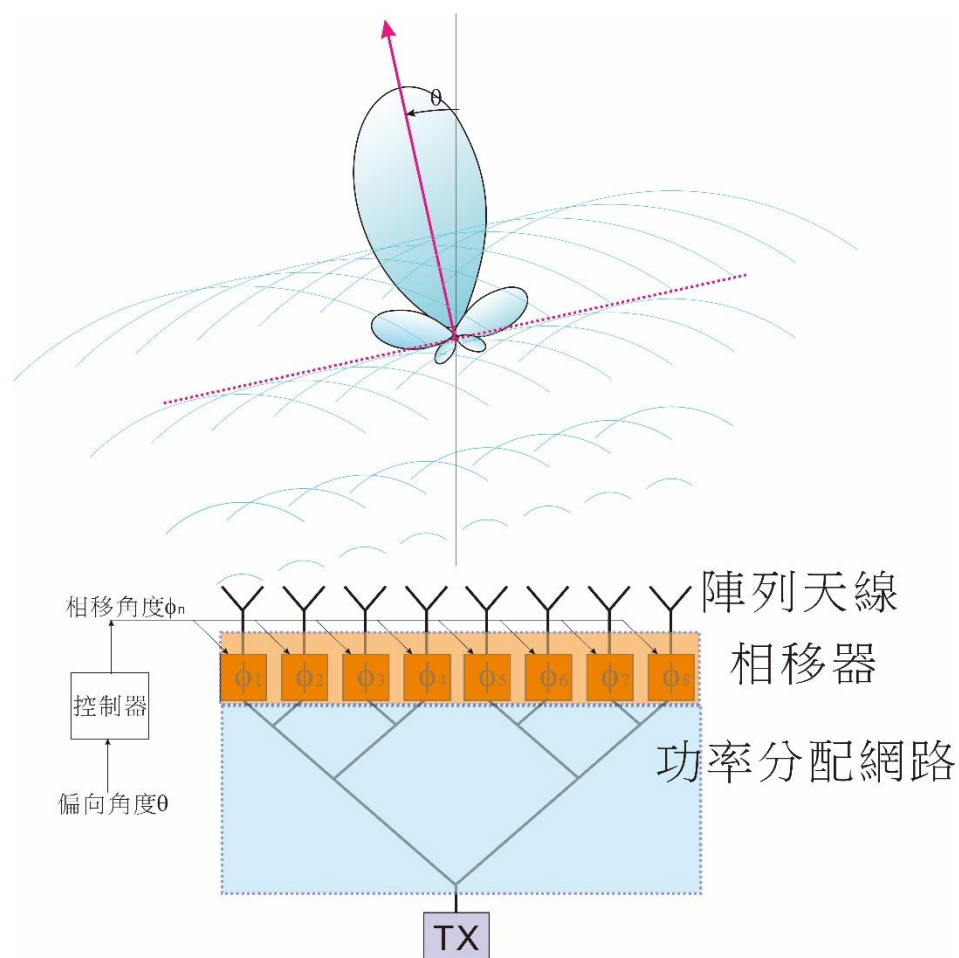


圖一、「相控陣列雷達」是最具代表性的電子式掃描技術

「相控陣列雷達」就是以「相控陣列天線 (Phased-Array Antennas)」進行電子式掃描的雷達系統。而「相控陣列天線」則是利用大量且可個別控制的天線單元排列而成的陣列天線。透過控制並調整各天線單元間之相位差，來達到使主波束轉向的目的[7-13]。「相控陣列天線」不但是「相控陣列雷達」的關鍵技術，更可應用在新一代的行動通訊系統中。例如第五代行動通訊所使用的巨量多入多出(Massive MIMO)的通訊技術，就是利用「相控陣列天線」以達到快速定位用戶並提供多個數據流，進而提升吞吐量的目的。

1.2 研究背景與目的

圖二所示為「相控陣列天線」的構成方塊圖：輻射單元排成陣列形式，以微控制器(micro-controller)控制相移器(phase shifter)改變饋入到個別天線單元的相位，由各單元輻射的電磁波會在特定方向產生建設性干涉，進而實現主波束在空間中定向或掃描的目的。



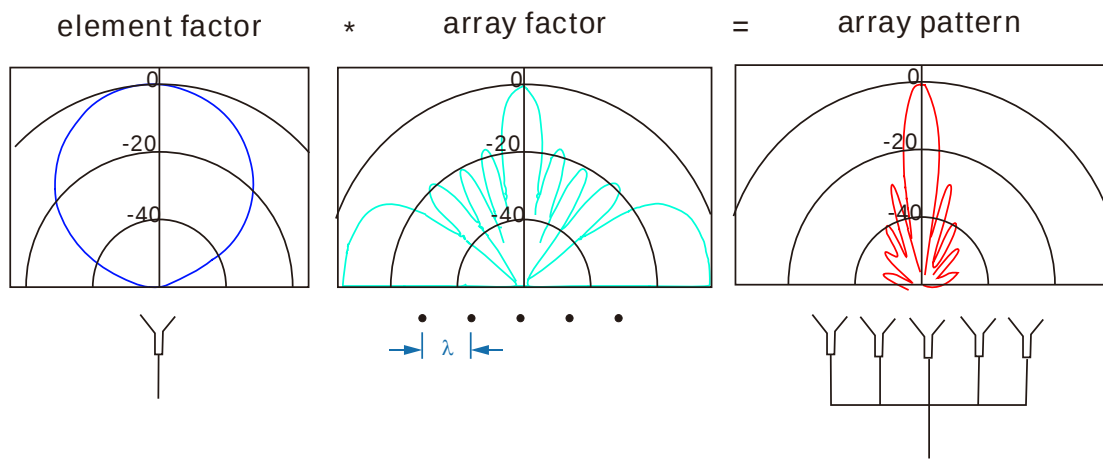
圖二、改變饋入到個別天線單元的相位以使主波束偏移

天線陣列最大的功能就是將發射/接收的能量集中於特定的方向，縮小主波束寬以提高增益。以一維的線性陣列為例，若將各元件的E-plane對齊在一個平面上，稱為E-plane linear array，則將縮小E-plane的主波束寬；同樣的，若將各元件的H-plane對齊在一個平面上，稱為H-plane linear array，則將縮小H-plane的主波束寬。一般來說，天線陣列元件數增加一倍，增益最多可以提高3dB。

分析天線陣列，最重要的是「場形相乘原理(principle of pattern multiplication)」。
假設天線陣列是由相同類型的天線所組成並且面向相同方向，則天線陣列的輻射場型可以由「元件因子(element factor)」及「陣列因子(array factor)」相乘而得(參見圖三)：

$$F(\theta, \phi) = F_e(\theta, \phi) * F_a(\theta, \phi) \quad (1)$$

其中，元件因子 $F_e(\theta, \phi)$ 為單一天線元件的場型函數；陣列因子 $F_a(\theta, \phi)$ 則是以等向性(isotropic)輻射體排列成相同陣形時的場型函數。



圖三、場形相乘原理

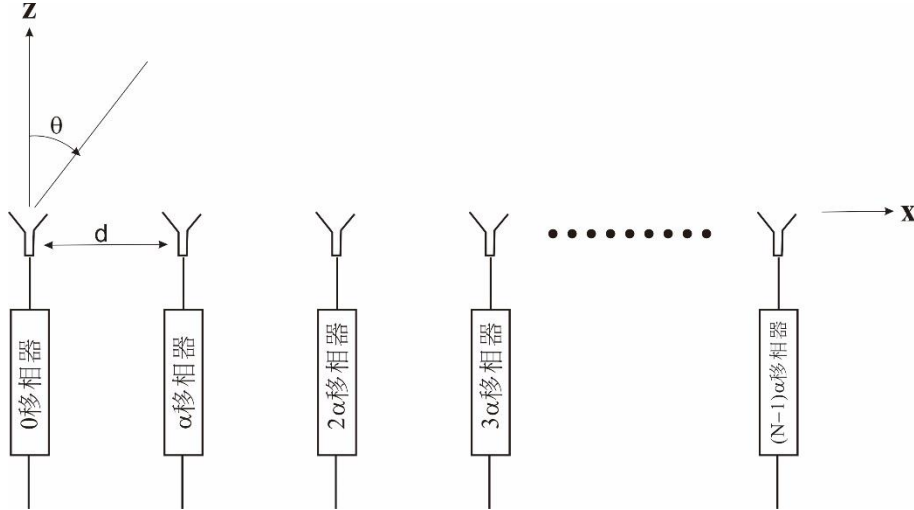
雖然在實際應用上，主流的做法是以二維的相控陣列讓主波束轉向，達到電子式掃描的目的。但考量以大學專題程度，本計畫僅以一維的「均勻、等距之線形相控陣列天線」為教學研究主題，使學生透過簡單的微波電路設計及微電腦控制系統，來實現一維的「相控陣列天線」。如圖四所示為 N 個天線單元、元件間距為 d 之均勻等距的線形相控陣列，其規一化陣列因子(normalized array factor)為

$$|A(\theta)| = \frac{1}{N} \left| \frac{\sin[\frac{N}{2}(\beta d \sin \theta + \alpha)]}{\sin[\frac{1}{2}(\beta d \sin \theta + \alpha)]} \right| \quad (1)$$

其中， $\beta = 2\pi/\lambda$ 為波數， N 為元件個數， α 為元件間相位差。陣列最大增益的方向 θ_0 ，必須滿足

$$\beta d \sin \theta_0 + \alpha = 0 \quad (2)$$

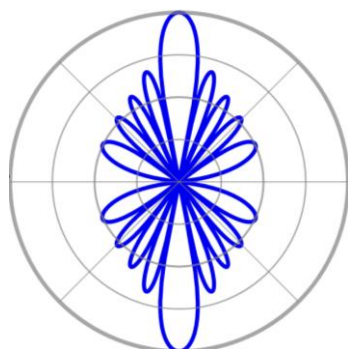
的條件。因此調整相鄰兩元件間的激發相位差(α)，可以改變陣列最大增益的方向。



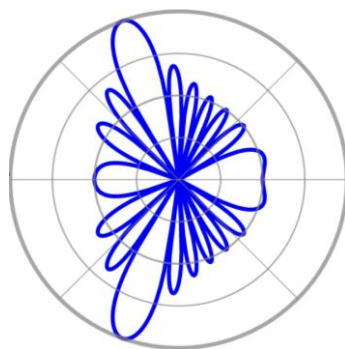
圖四、N 個天線單元之均勻等距線形相控陣列

如圖五所示，以元件數 10 個、間距半波長的線形陣列為例：當每個元件以相同功率及相位激發時($\alpha=0$ 度)，陣列在正面($\theta=0$ 度)方向有最大增益；當相位差 $\alpha=60$ 度時，陣列最大輻射方向約在 $\theta=-20$ 度；當相位差 $\alpha=120$ 度時，陣列最大輻射方向約在 $\theta=-40$ 。反之，當相位差 $\alpha=-60$ 度時，陣列最大輻射方向約在 $\theta=20$ 度；當相位差 $\alpha=-120$ 度時，陣列最大輻射方向約在 $\theta=40$ 。而當相位差 $\alpha=\pm 180$ 度時，為端射式陣列，其最大輻射方向在 $\theta=90$ 度。

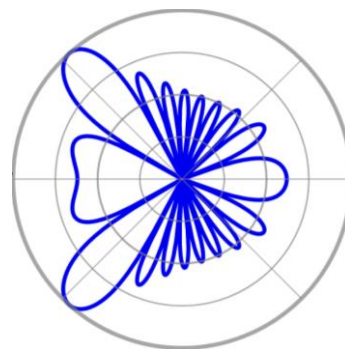
(a) $\alpha = 0$ 度



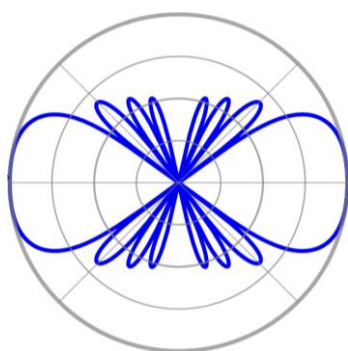
(b) $\alpha = 60$ 度



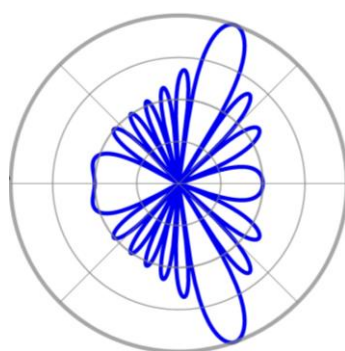
(c) $\alpha = 120$ 度



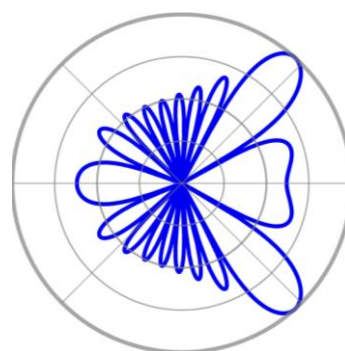
(d) $\alpha = \pm 180$ 度



(e) $\alpha = -60$ 度



(f) $\alpha = -120$ 度



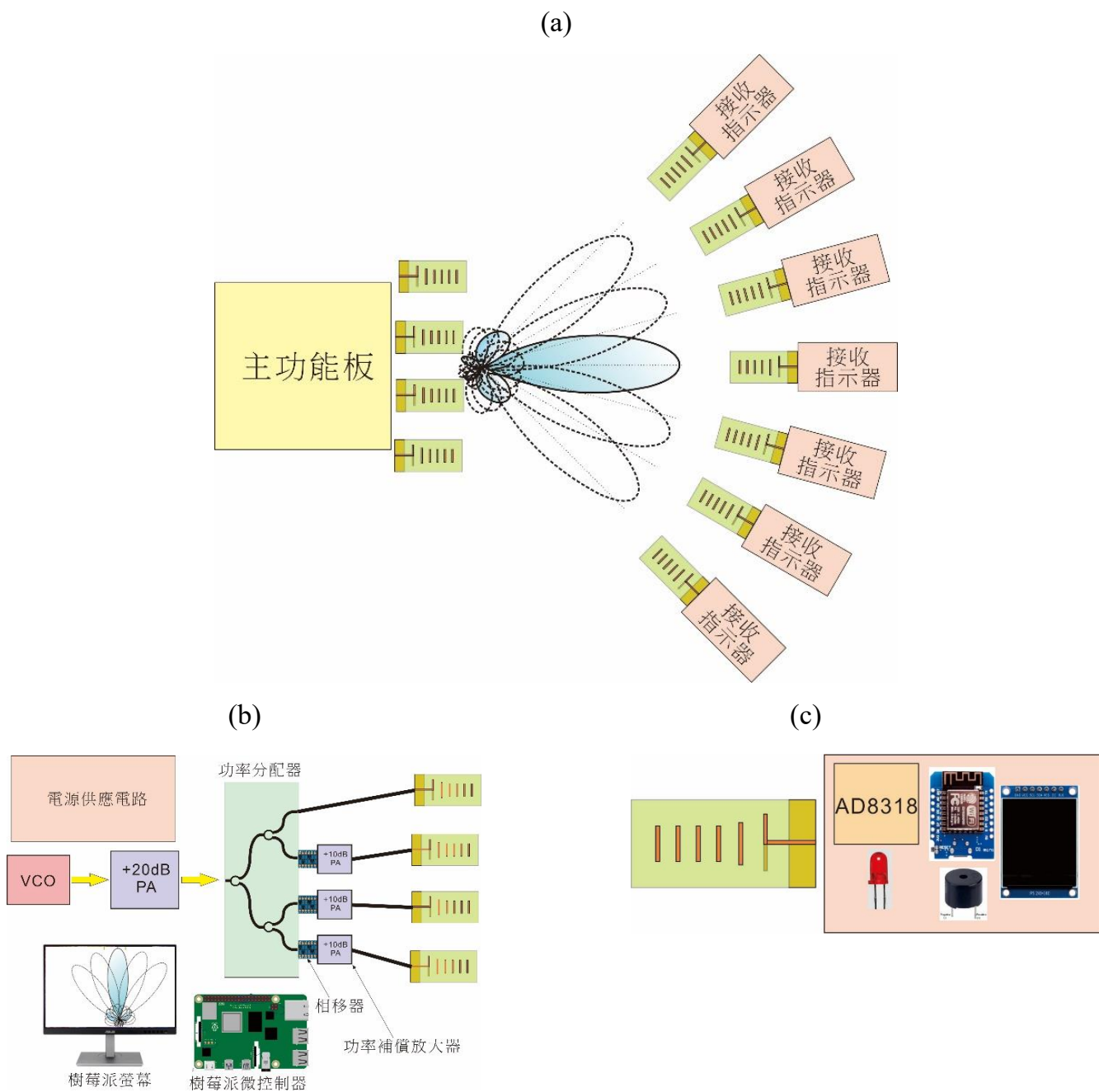
圖五、元件數為 10、間距半波長的線形陣列，不同相位差時之陣列因子場型

第二章

研究內容與成果

「專題製作」課程是本校（陸軍軍官學校）電機系高年級學生的必修課程，每週上課時數三小時，兩學期合計108小時。本課程區依據主題分組，「電波專題」為其中一個主題，本學年度修課學生人數共12員。透過本計畫之執行，教導學生熟悉天線工程、微波工程、雷達工程等科目及相關量測技術。此外，以「相控陣列天線」實現主波束在空間中定向或掃描的功能，也是教學重點之一。

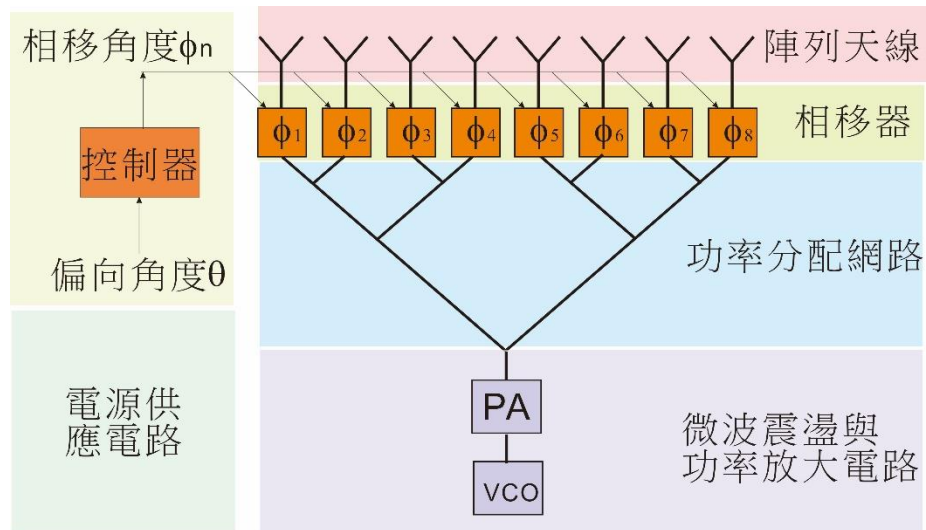
本年度的研究計畫中，我們設計並完成一套「相控陣列天線教學展示系統」的雛型。展示系統整體結構圖如圖六(a)所示，主要由圖六(b)的主功能板及圖六(c)的接收指示器所構成。如圖六(a)所示，在與主功能板固定距離的扇形位置上擺放數個接收指示器，當主功能板上的「相控陣列天線」主波束偏轉時，相對應角度上的接收指示器會接收到足夠強的功率，進而驅動包括：點亮LED燈、蜂鳴器發出“嗶”聲、並在OLED螢幕上顯示接收到的訊號強度等動作。如此即可立即得知目前主波束所偏轉的角度，可以給學生對「相控陣列天線」建立具體的觀念，有效提昇教學成效。



圖六、相控陣列天線教學展示系統

- (a) 系統整體結構圖；
- (b) 主功能板各部示意圖；
- (c) 接收指示器各部示意圖。

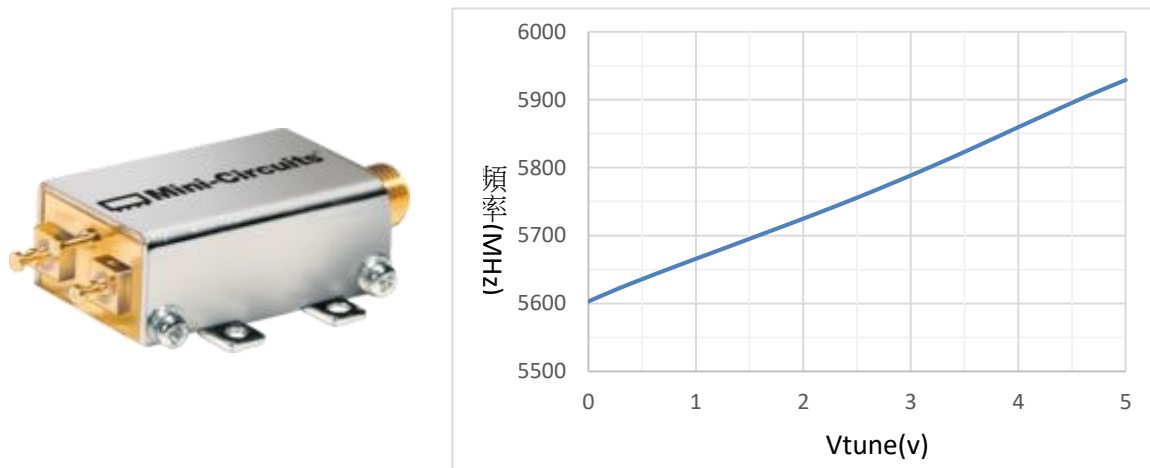
為了能系統化地進行「相控陣列天線教學展示系統」的教學工作，如圖七所示，我們將主功能板的系統區分為：（一）微波震盪與功率放大電路、（二）功率分配網路、（三）相移器及其控制電路、（四）天線單元與陣列天線、（五）電源供應電路；而接收指示器則在第（六）小節中說明。



圖七、將「主功能板」分為五部份進行製作

2.1 微波震盪與功率放大電路

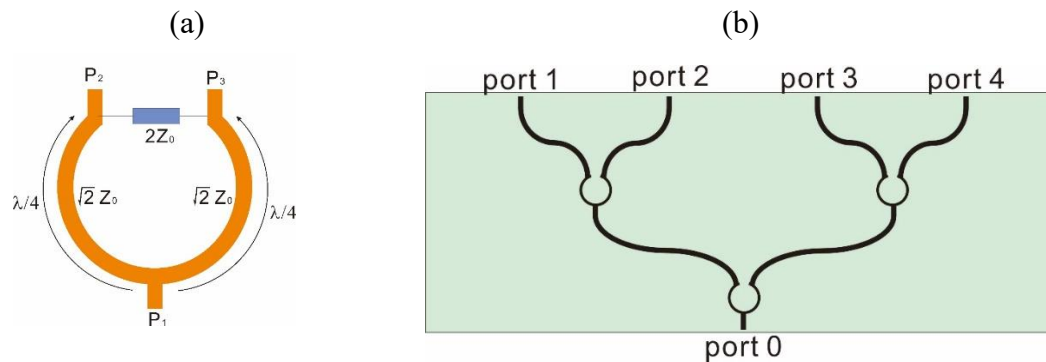
如圖七紫色區塊部份所示，為微波震盪與功率放大電路。本計畫中的訊號源由壓控震盪器震盪出微波功率，並經低雜訊功率放大器將功率提升。計畫中採用的壓控震盪器為Mini-Circuits ZX95-5776，其外觀及特性曲線如圖九所示。當Vtune電壓5V時，可獲得5.9 GHz、功率約-2 dBm的微波訊號。低雜訊功率放大器採用的型號為Mini-Circuits ZX60-V63，該放大器操作在6 GHz時，增益約為15 dB。因此壓控震盪器的輸出經由低雜訊功率放大器之後，可將功率提升至13 dBm。



圖八、Mini-Circuits ZX95-5776 外觀及特性曲線

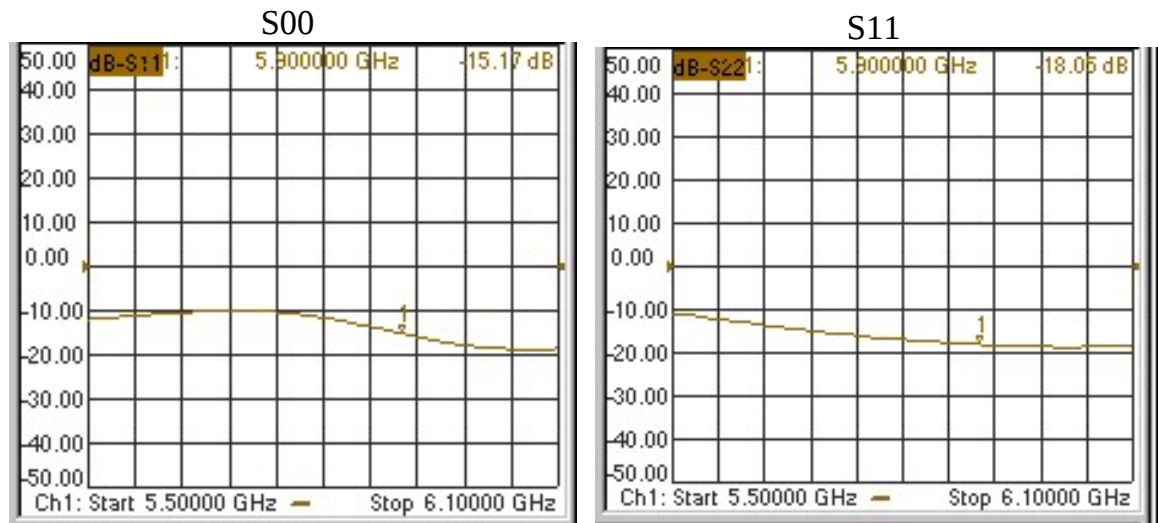
2.2 功率分配網路

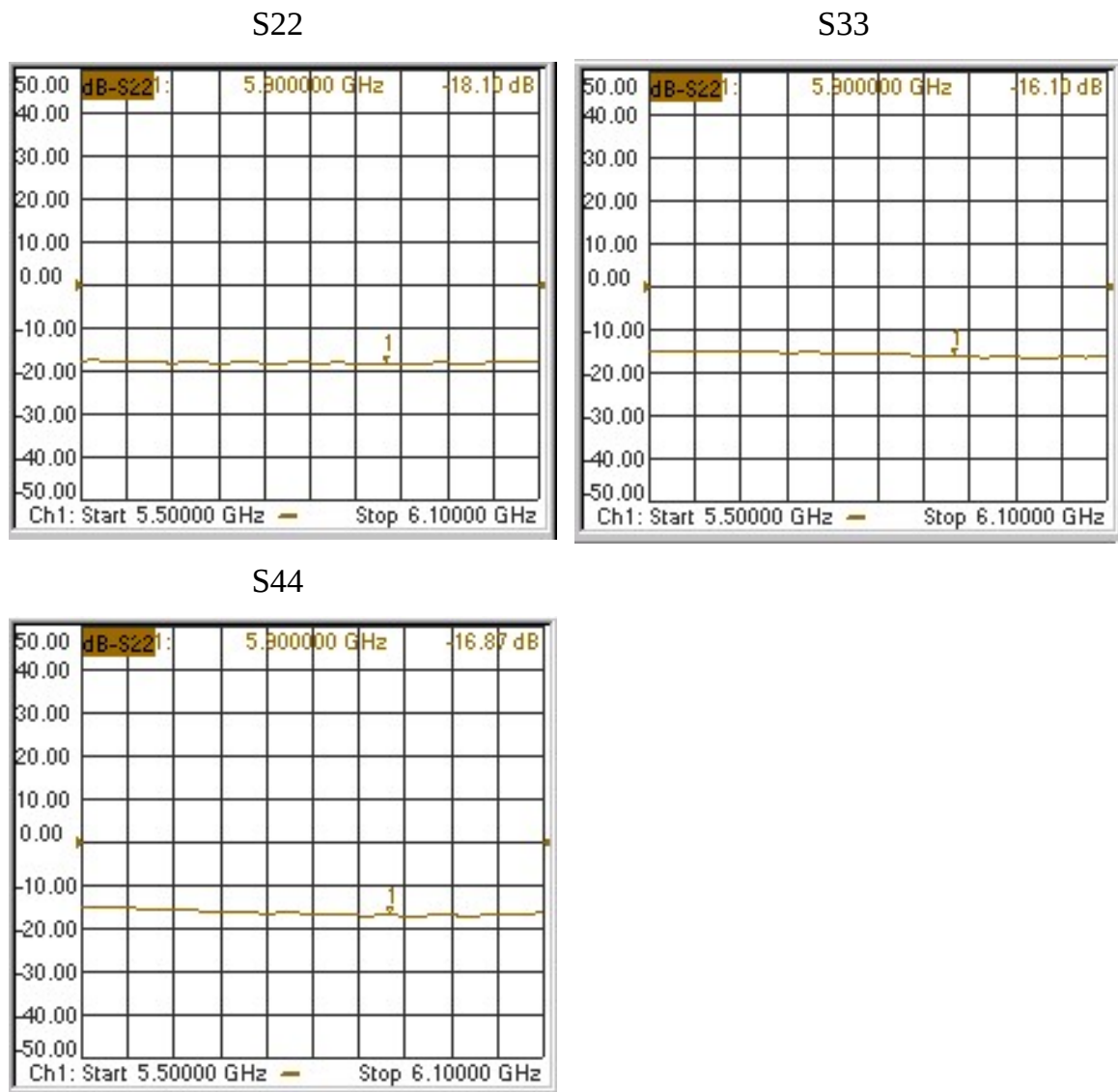
如圖七藍色區塊部份所示，為功率分配網路。功率分配器(power divider)的功能為將輸入功率分為功率相當的兩部份。功率分配器屬被動式元件，在製作上比較簡單，本計畫採用如圖九(a)所示之威爾金森功率分配器(Wilkinson power divider)。威爾金森功率分配器為威爾金森(Ernest J. Wilkinson)於1960年提出設計[14]，其結構簡單且輸出埠間彼此互相隔離、又同時各埠皆阻抗匹配的一種功率分配網路。本計畫採用堆疊式設計，以獲得如圖九(b)所示之一分四的功率分配網路。



圖九、威金森功率分配器

圖十所示為一分四威爾金森功率分配器各埠匹配的量測結果。 S_{xx} 代表port x 的阻抗匹配程度，當測量某一埠的匹配時，其他埠則連接50歐姆終端電阻。由結果可以看出，操作在5.9 GHz時，各埠的 S_{xx} 都在-15dB以下，有良好的阻抗匹配。



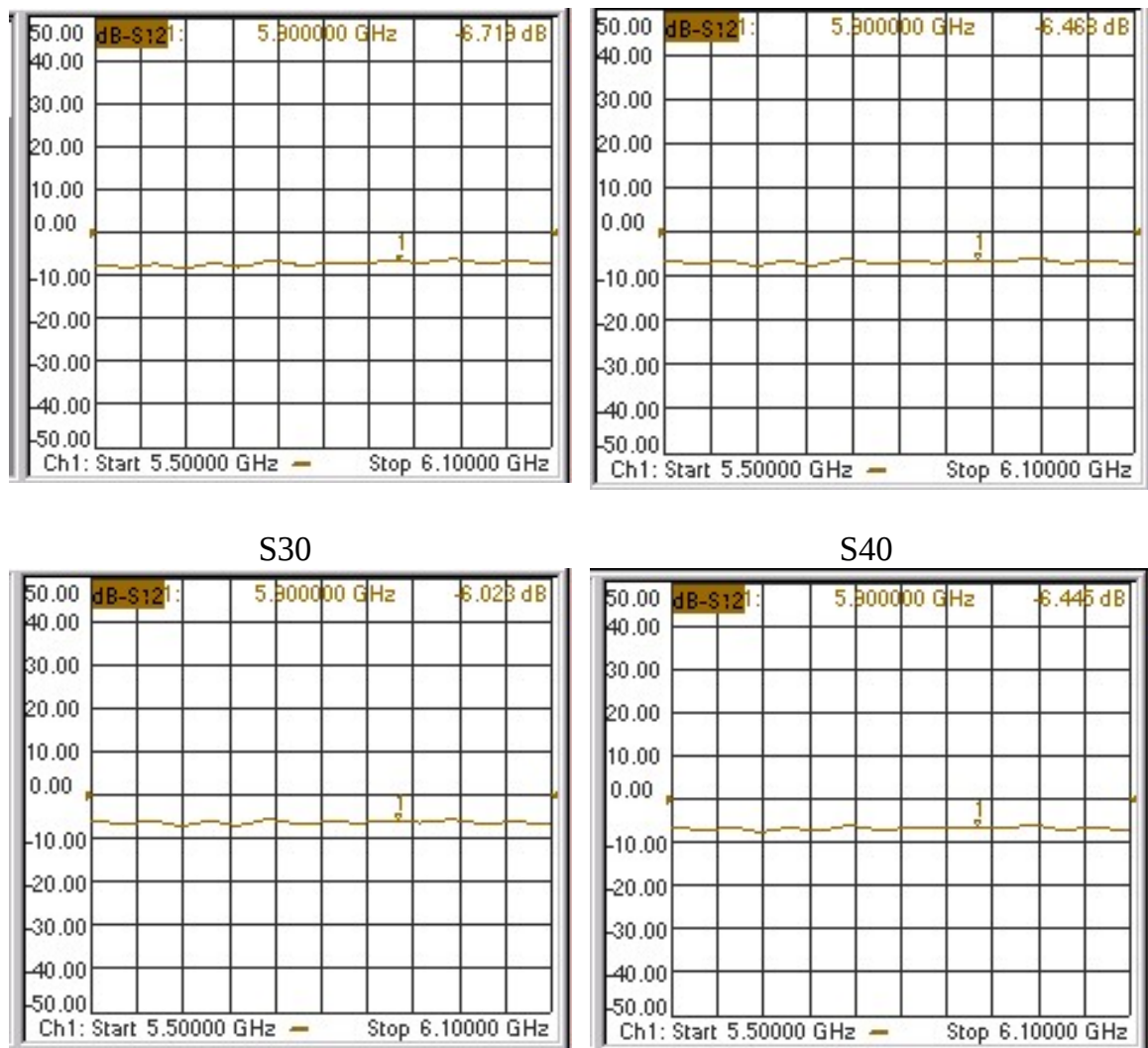


圖十、一分四威爾金森功率分配器各埠阻抗匹配的量測結果。

圖十一所示為一分四威爾金森功率分配器各埠分配功率的量測結果。 S_{x0} 代表從 port 0 傳遞到 port x 過程的饋入損失(insertion loss)。當測量 port 0 與某一埠的饋入損失時，其他埠則連接 50 歐姆終端電阻。由結果可以看出，從 port 0 傳遞到各埠的饋入損失 S_{x0} 都在 -6dB 左右，也就是說此一威爾金森功率分配器有良好的「一分為四」的效果。

S10

S20

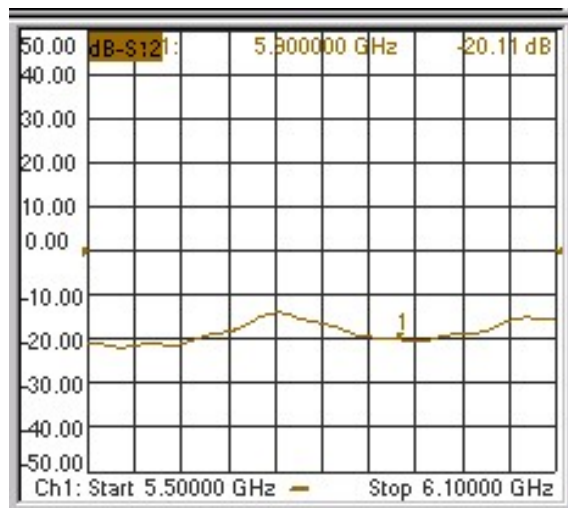


圖十一、一分四威爾金森功率分配器各埠分配功率的量測結果。

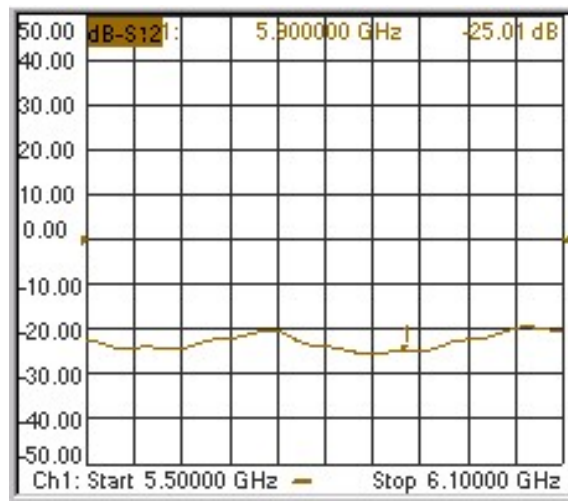
圖十二所示為一分四威爾金森功率分配器 port 1、port 2、port 3、port 4 各埠隔離度的量測結果。 $|S_{xy}|$ 代表 port x 與 port y 的隔離度。當測量 port x 與 port y 之隔離度時，port 0 與其他埠則連接 50 歐姆終端電阻。相鄰兩埠的隔離度 S12、S23、S34，其隔離度都在 20 dB 以上。其他非相鄰兩埠的隔離度 S13、S24、S14，其隔離度都在 24 dB 以上。由結果看出各埠之間隔離效果良好。

S12

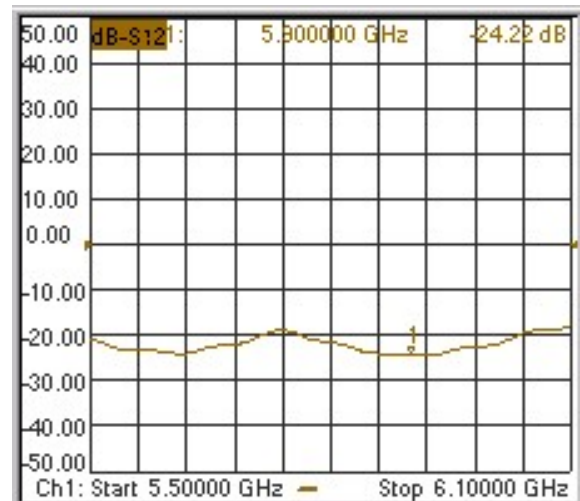
S13



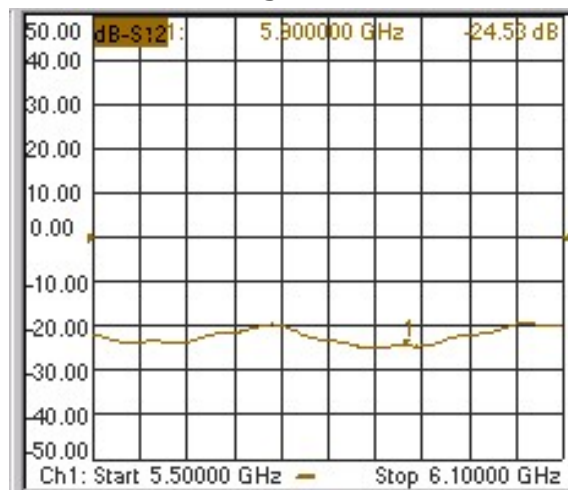
S14



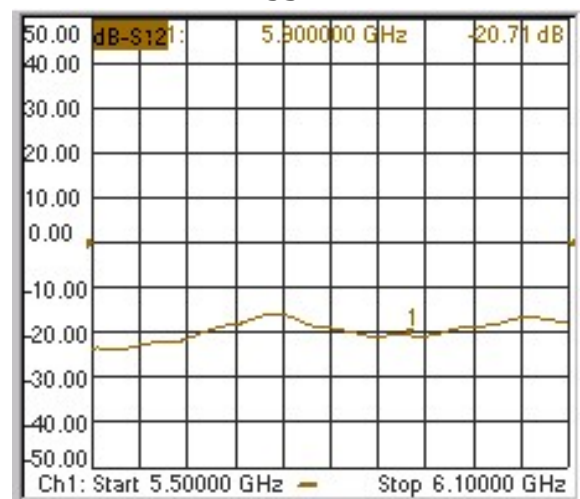
S23



S24



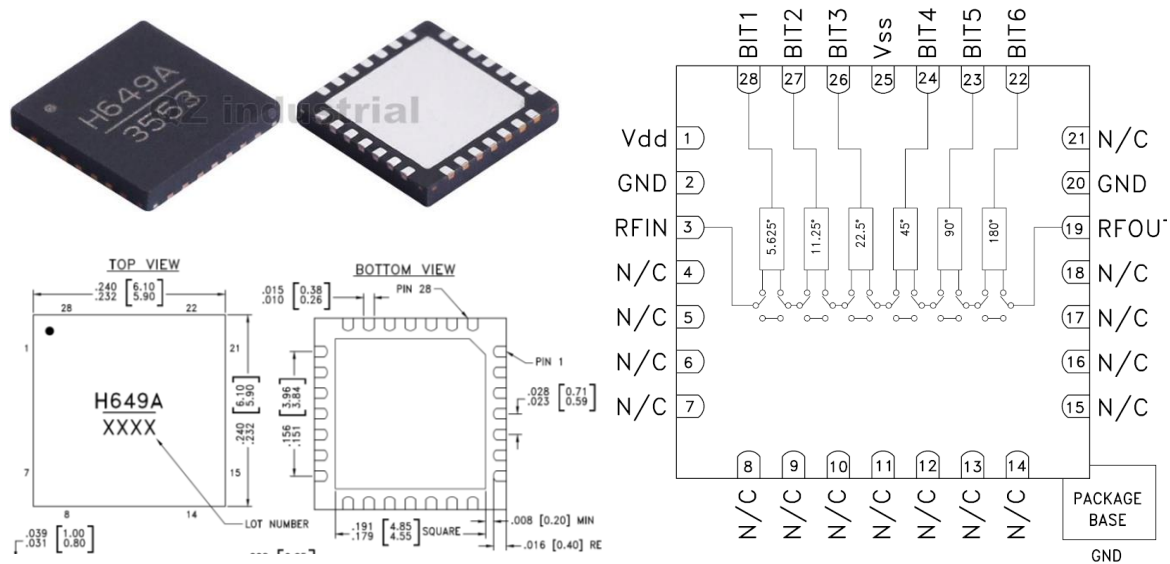
S34



圖十二、一分四威爾金森功率分配器各埠隔離度的量測結果。

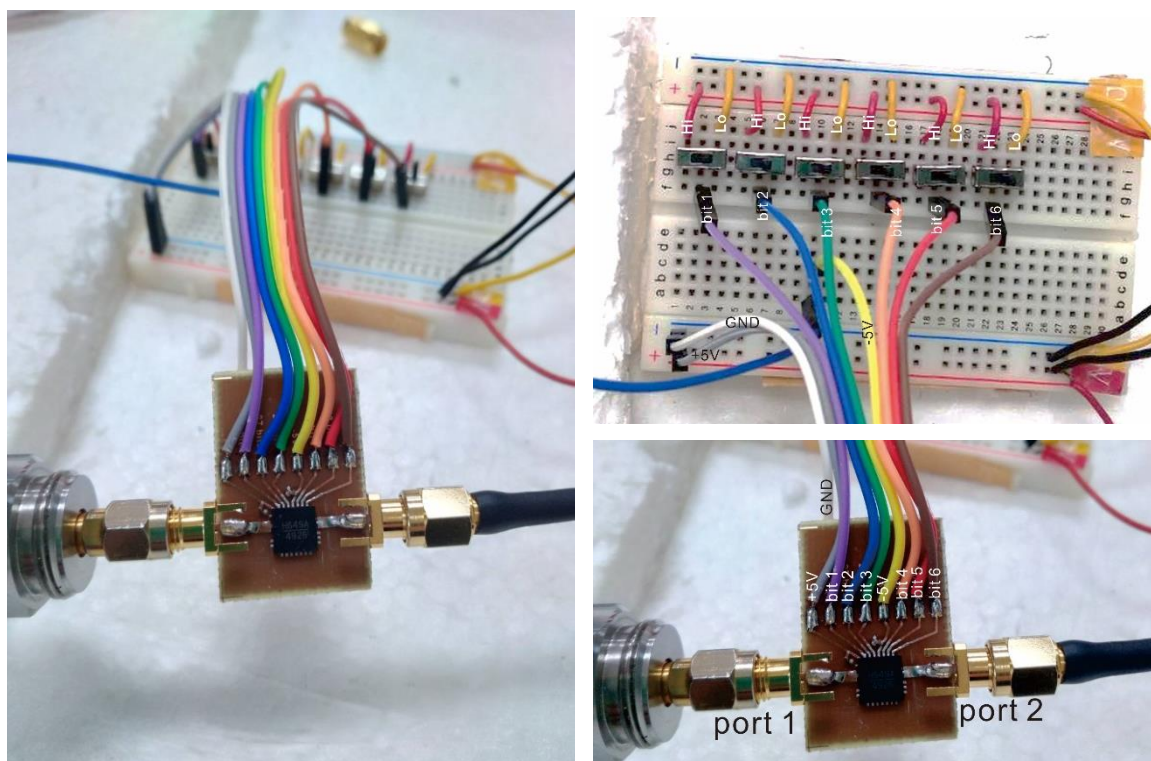
2.3 相移器及其控制電路

如圖七所示黃底色方框部分為相移器及其控制器電路，為簡化教學及製做流程，計畫中以市售現有的 IC 及微控制器單元，進行區塊模組化的講解及實驗。計畫中採用亞德諾半導體公司 (Analog Devices Inc.) 所設計生產之 HMC649A 相移器 IC，其外觀與接腳如圖十三所示。HMC649A 相移器 IC 共有 6 個控制腳位，也就是將 360 度相角量化成 64 等分，每一等分間隔 5.625 度。對於大學專題層級的實驗來說，此一切割精度已然足夠。



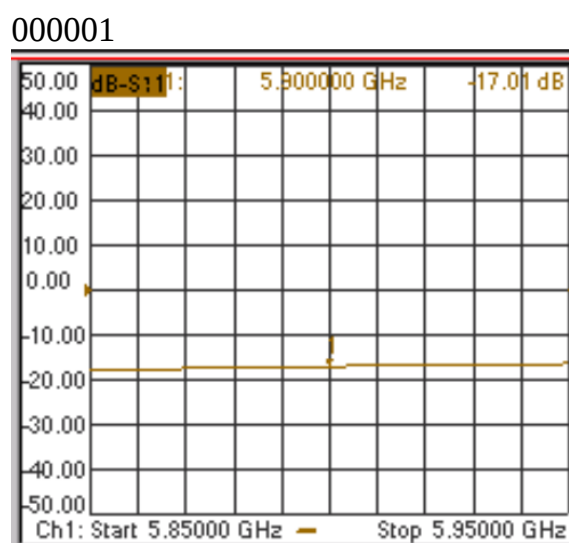
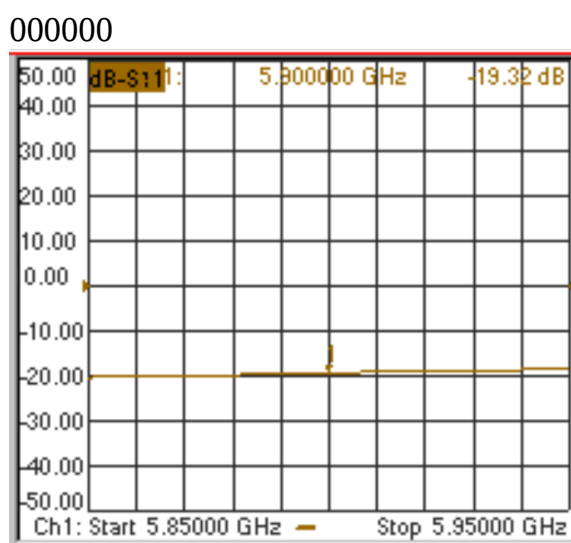
圖十三、亞德諾半導體 HMC649A 相移器 IC 之外觀及接腳圖

圖十四所示為測量 HMC649A 相移器 IC 時的接線配置圖。HMC649A 相移器 IC 為 QFN28 封裝，其接點間距僅 0.65mm，因此必須先蝕刻線寬極細的印刷電路板將 IC 焊接上去，並接上 SMA 接頭之後方能使用。如圖十四所示，以六個開關在 5V 及 0V 之間切換，以分別代表 Bit 1~Bit 6 的 HIGH 或 LOW。將準備妥當的測試板連接網路分析儀，以網路分析儀測試包括：PORT 1 的匹配 (S11)、PORT 2 的匹配 (S22)、PORT 1 到 PORT 2 的饋入損耗 (S21)、以及 PORT 1 到 PORT 2 的相位延遲 (phase of S21) 等。

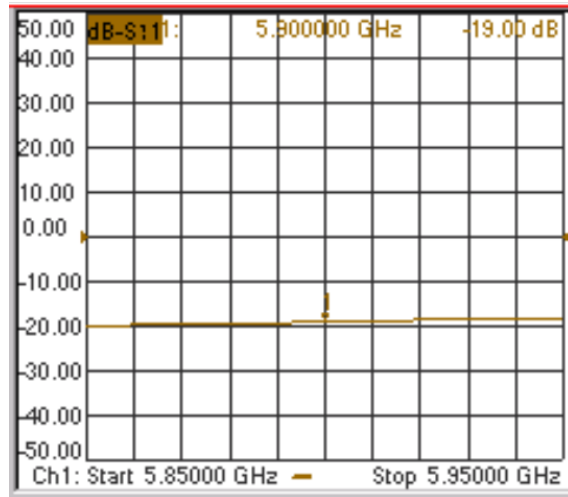


圖十四、測量 HMC649A 相移器 IC 時的接線配置圖。

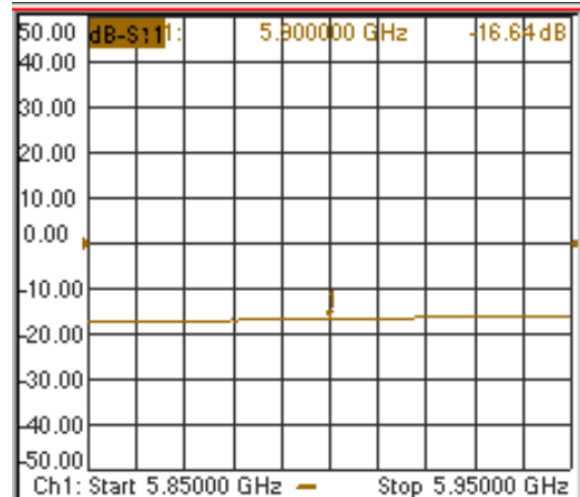
圖十五所示為 PORT 1 匹配程度(S11)的量測結果。圖中展示了八種相位切換時的組合，以確保相移器電路在相位切換時，依舊能保持匹配狀態。切換開關的組合，以 0 代表 LOW，以 1 代表 HIGH，並以(Bit 6, Bit 5, Bit 4, Bit 3, Bit 2, Bit 1)的順序數列來代表。由結果可以看出，操作在 5.9 GHz 時，S11 都能保持在 -15dB 以下，代表有良好的阻抗匹配。



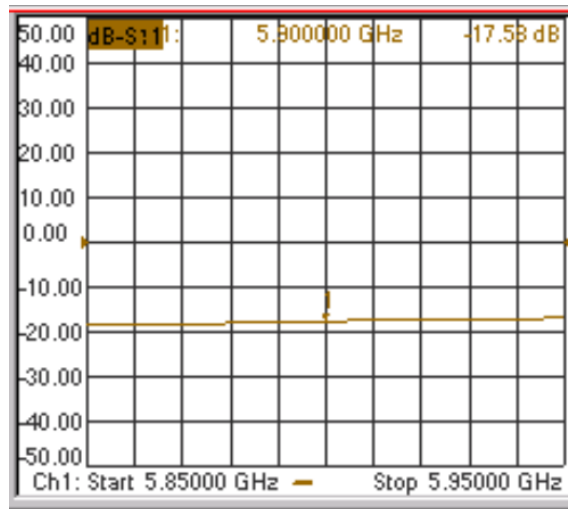
000010



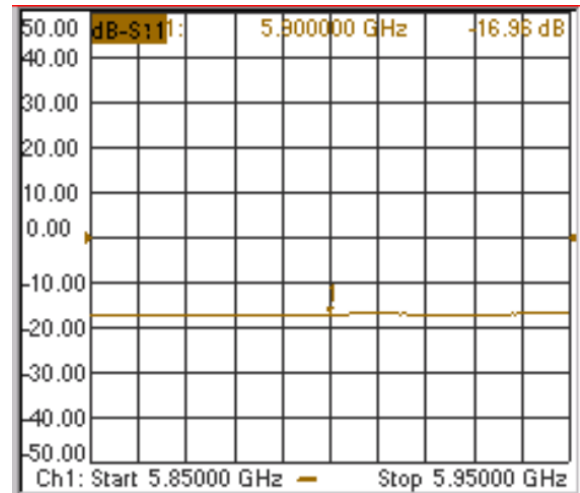
000100



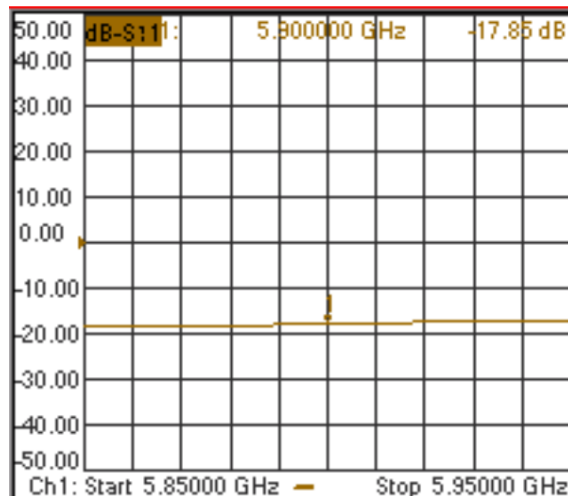
001000



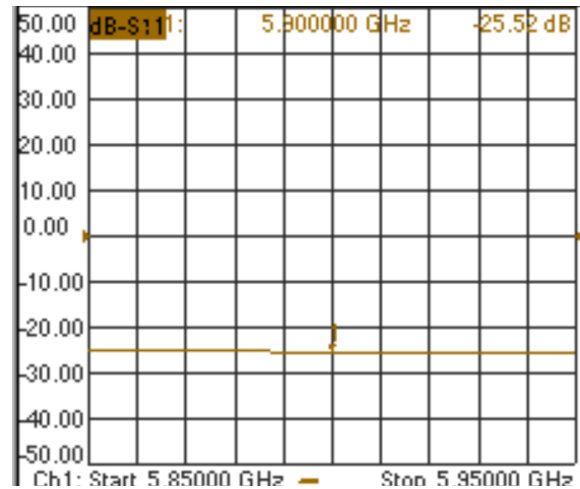
010000



100000



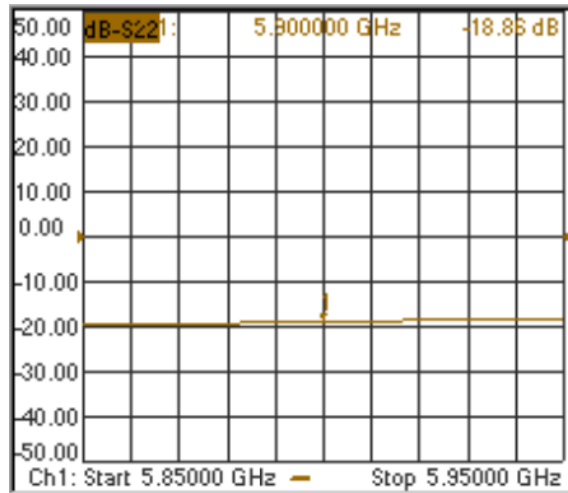
111111



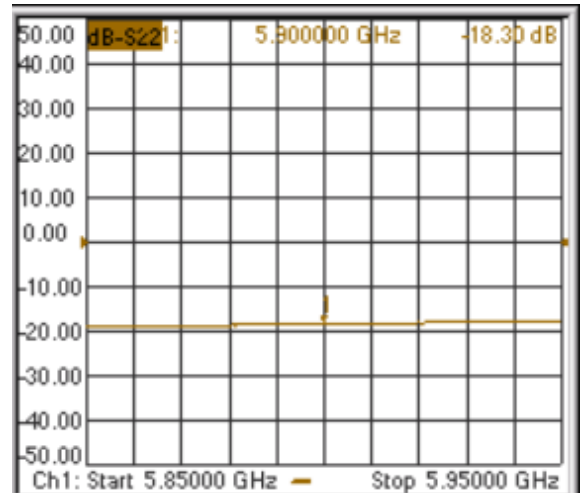
圖十五、PORT 1 匹配程度(S11)的量測結果。

圖十六所示為 PORT 2 匹配程度(S22)的量測結果。圖中展示了八種相位切換時的組合，以確保相移器電路在相位切換時，依舊能保持匹配狀態。切換開關的組合，以 0 代表 LOW，以 1 代表 HIGH，並以(Bit 6, Bit 5, Bit 4, Bit 3, Bit 2, Bit 1)的順序數列來代表。由結果可以看出，操作在 5.9 GHz 時，S22 都能保持在 -15dB 以下，代表有良好的阻抗匹配。

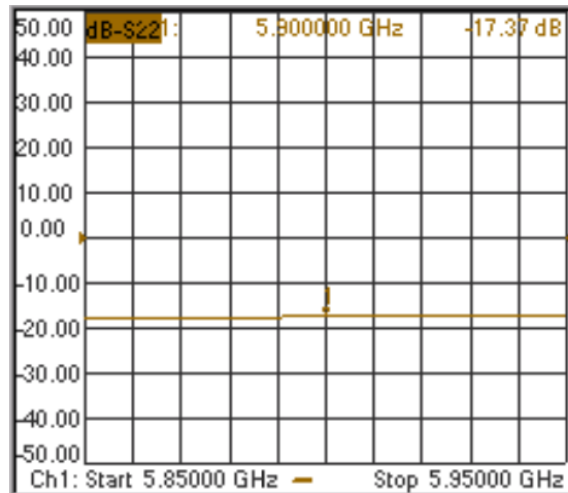
000000



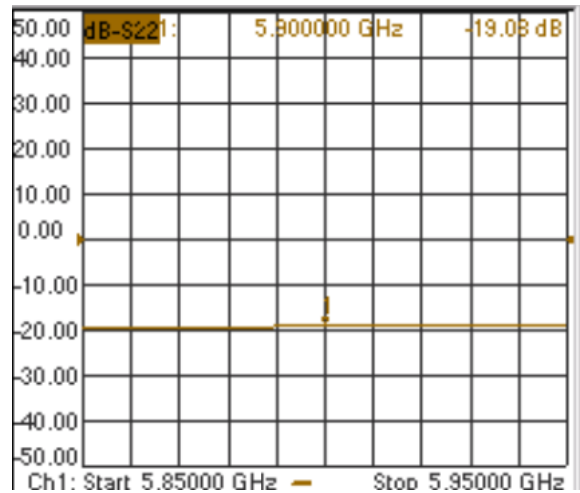
000001



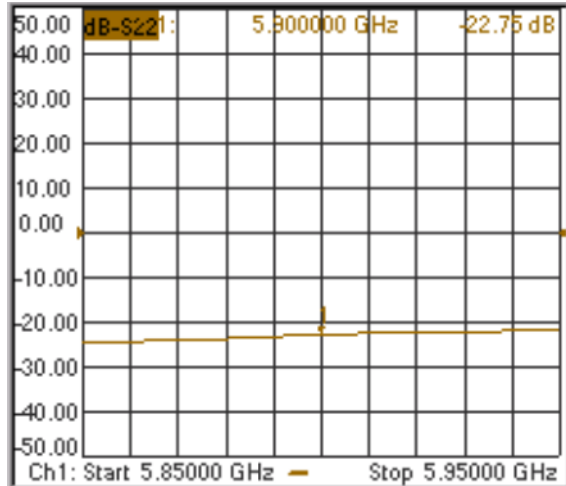
000010



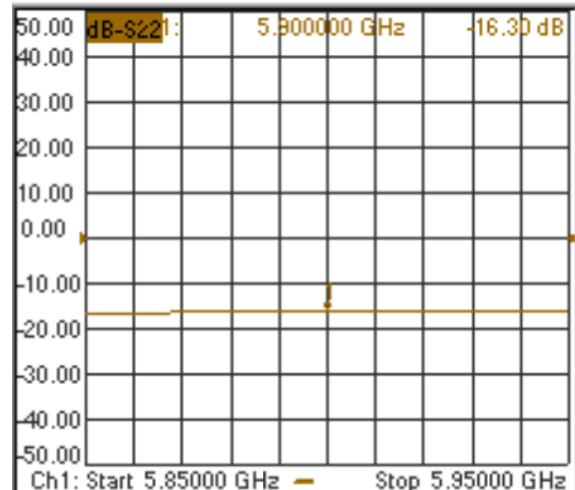
000100



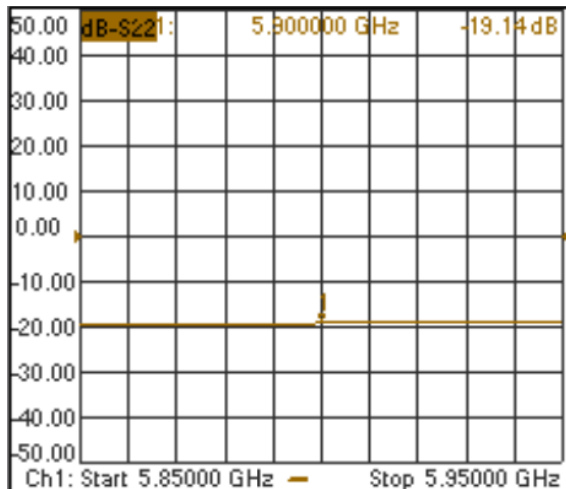
001000



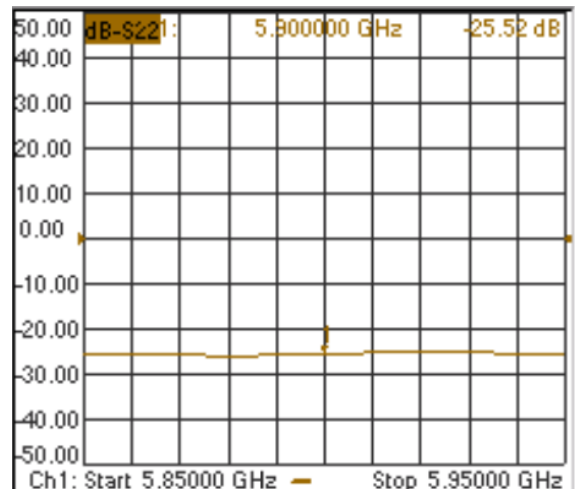
010000



100000



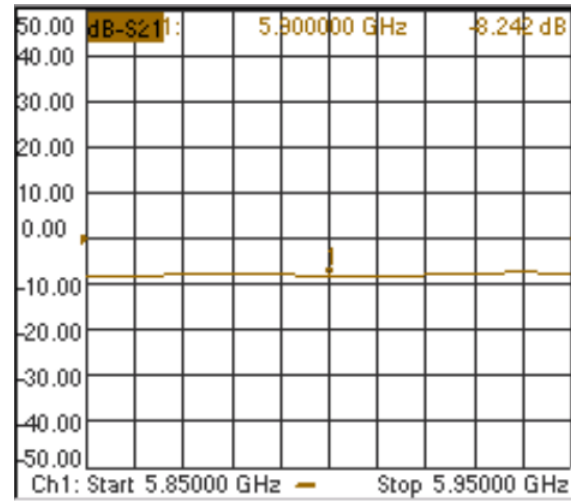
111111



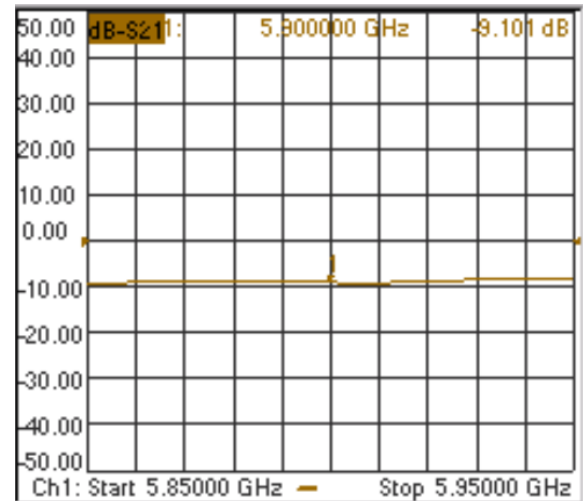
圖十六、PORT 2 匹配程度(S22)的量測結果。

圖十七所示為 PORT 1 到 PORT 2 的饋入損耗(S21)之量測結果。圖中展示了八種相位切換時的組合，以了解相移器電路在相位切換時，饋入損耗的狀況。切換開關的組合，以 0 代表 LOW，以 1 代表 HIGH，並以(Bit 6, Bit 5, Bit 4, Bit 3, Bit 2, Bit 1)的順序數列來代表。由結果可以看出，操作在 5.9 GHz 時，PORT 1 到 PORT 2 的饋入損耗(S21)在 -6.5dB~-9.1dB 之間變動。也就是說如果我們的目標是均勻線型陣列，則必須在相移器之後再串接一將近 10dB 的微波放大器，以維持個天線元件之間有強度相近的饋入。

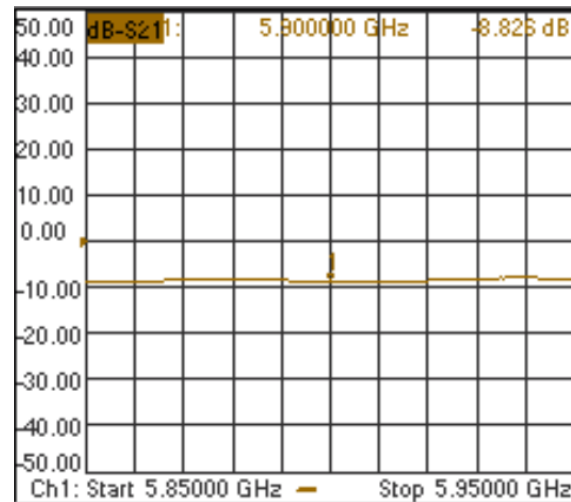
000000



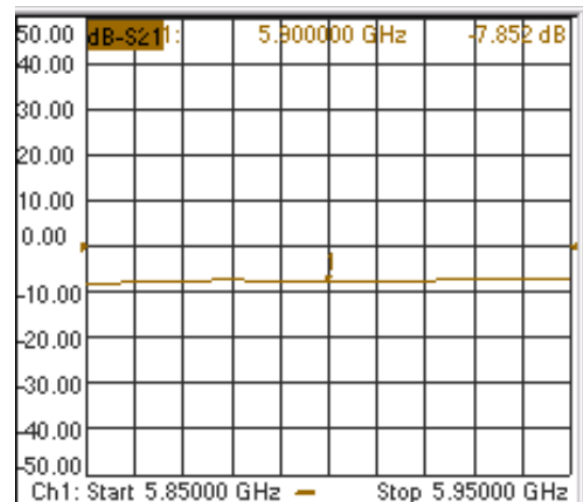
000001



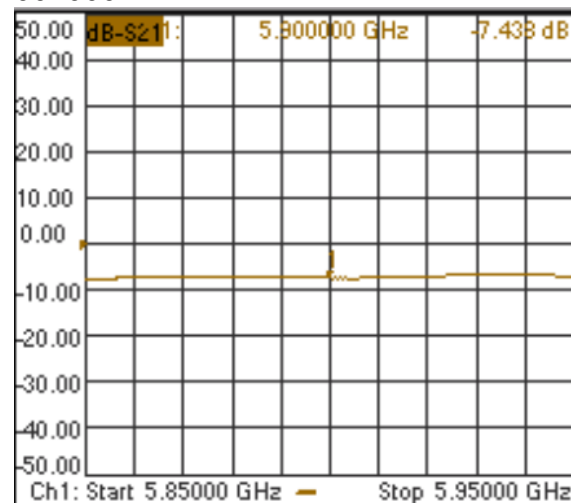
000010



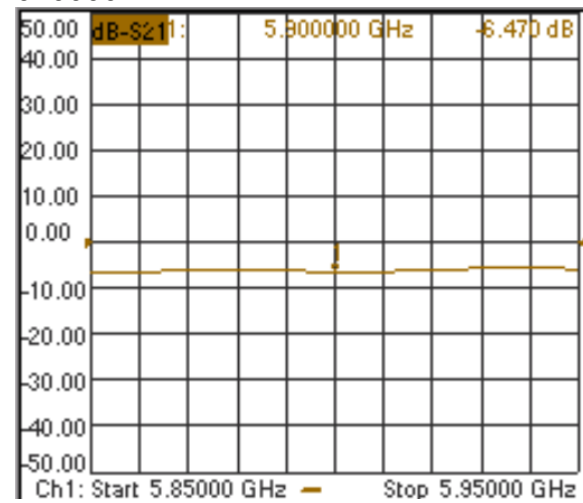
000100

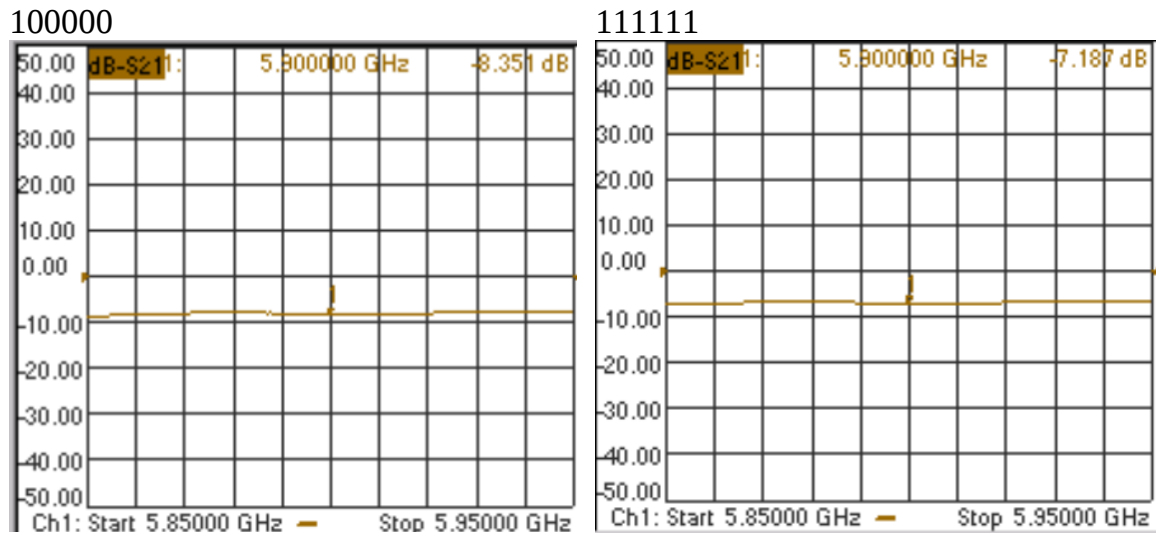


001000



010000

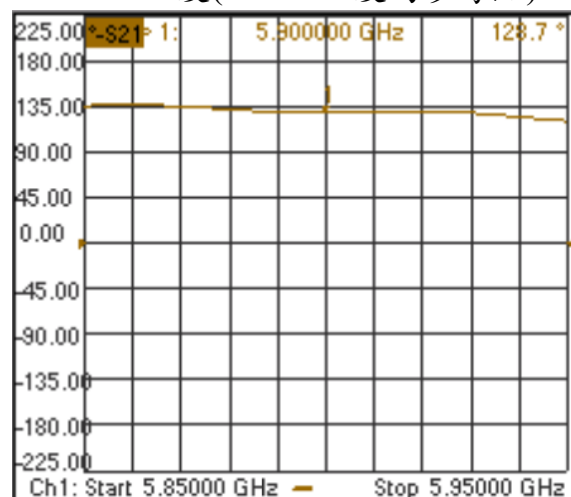




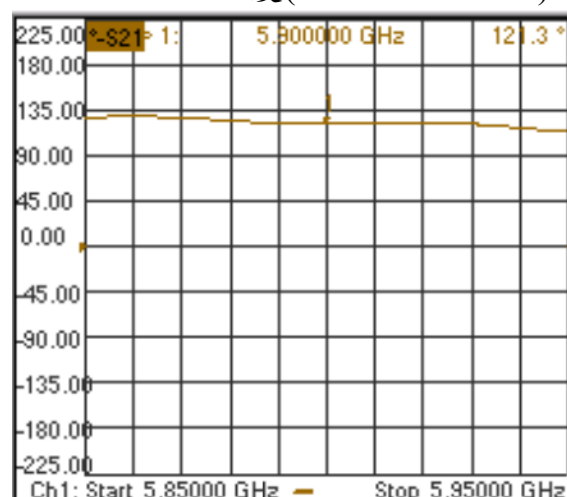
圖十七、PORT 1 到 PORT 2 的饋入損耗(S21)之量測結果。

圖十八所示為 PORT 1 到 PORT 2 的相位延遲(phase of S21)之量測結果。圖中展示了八種相位切換時的組合，以了解相移器電路在相位切換時，饋入損耗的狀況。切換開關的組合，以 0 代表 LOW，以 1 代表 HIGH，並以(Bit 6, Bit 5, Bit 4, Bit 3, Bit 2, Bit 1)的順序數列來代表。亦即透過這 6 個控制腳位，將 360 度相角量化成 64 等分，每一等分間隔 5.625 度。例如若(Bit 6, Bit 5, Bit 4, Bit 3, Bit 2, Bit 1)的數列組合為 000100，則其理論上的相位延遲應為 $4 * 5.625 = 22.5$ 度。圖中亦將理論值列出作為比較。由結果可以看出，操作在 5.9 GHz 時，PORT 1 到 PORT 2 的相位延遲，量測值與理論值差異不大，都在可接受的誤差範圍內。

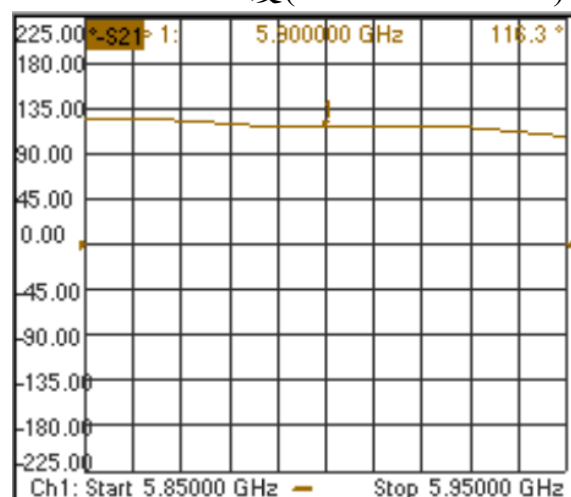
000000→0 度(以 128.7 度為參考點)



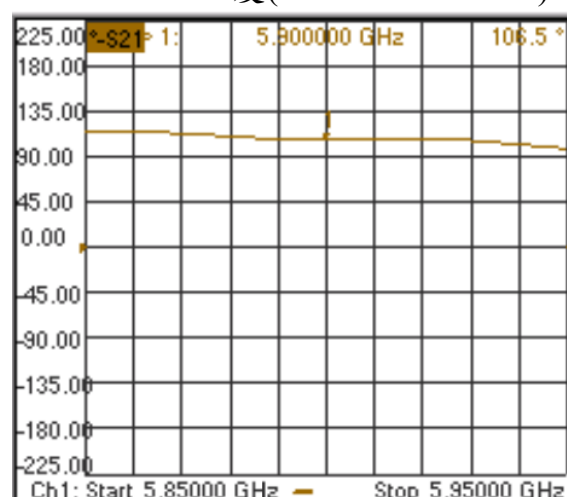
000001→-5.625 度(121.3-128.7=-7.4)



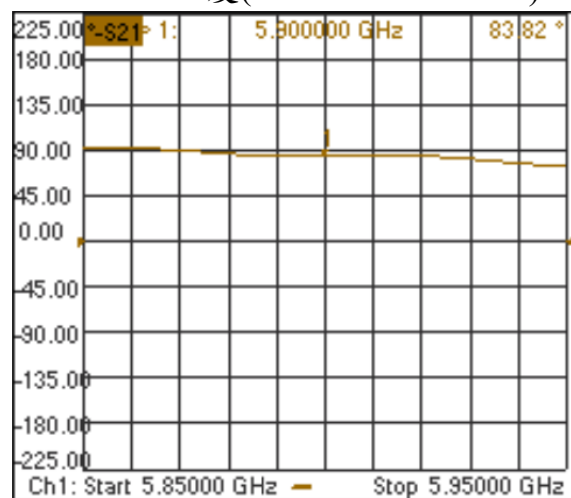
000010→-11.25 度(116.3-128.7=-12.4)



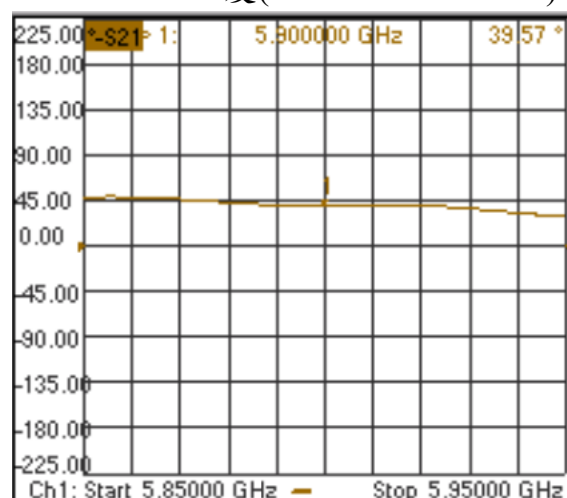
000100→-22.5 度(106.5-128.7=-22.2)



001000→-45 度(106.5-128.7=-44.88)

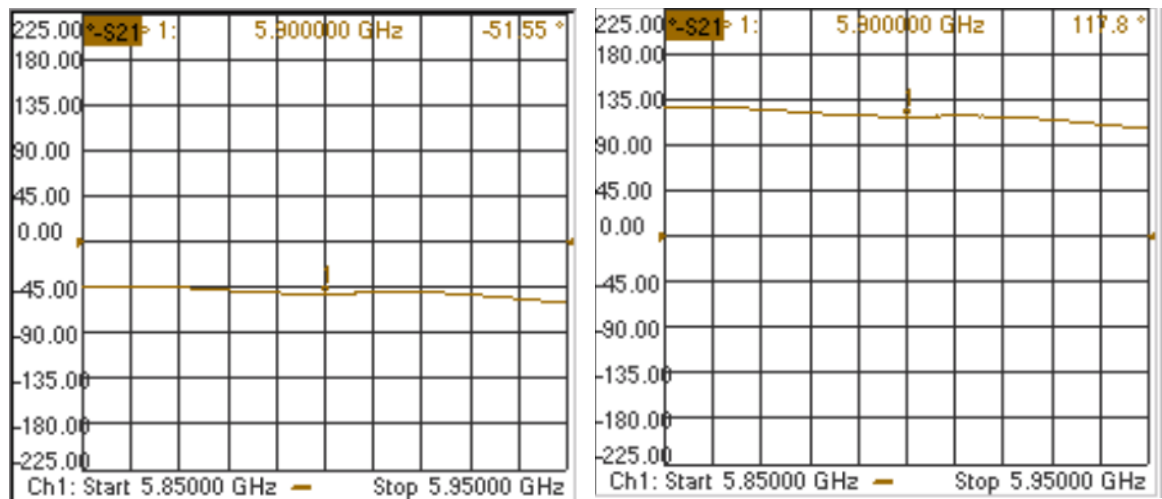


0100000→-90 度(39.57-128.7=-89.13)



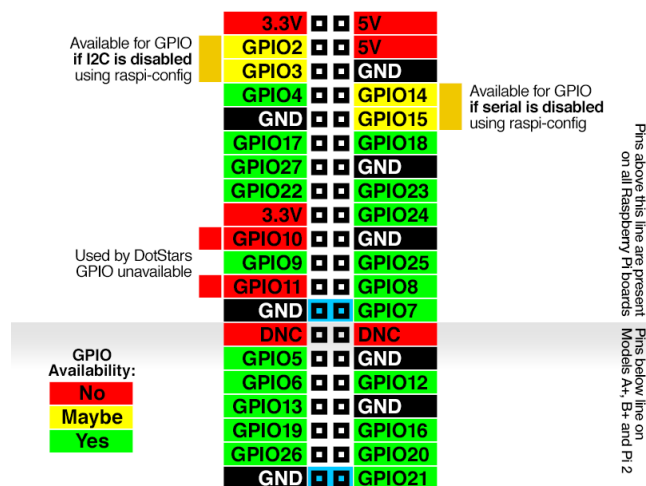
100000→-180(-51.55-128.7=-180.25)

111111→+5.625(128.7-117.8=10.9)



圖十八、PORT 1 到 PORT 2 的相位延遲之量測結果。

展示系統的相位控制必須透過控制器，自動送出相對應相移角度的二位元信號。計畫中的目標是要製作一個 1x4 的相控陣列天線，若以天線 1 為相位參考點，則需要 3 個相移器針對另外三支天線的相位進行調控，來達到主波束偏轉的目的。而每一個相移器都需要 6 個位元來加以控制，3 個相移器共 18 個位元。一般的 Arduino Uno 板並沒有這麼多的 GPIO 腳為可供使用，因此本計畫採用樹莓派單板電腦來達到同時控制 3 個相移器的目的。圖十九所示為樹莓派 GPIO 腳位編號及功能圖，而表一則是 GPIO 腳位與 3 個相移器控制位元間連接的規劃表。



圖十九、樹莓派 GPIO 腳位編號及功能圖

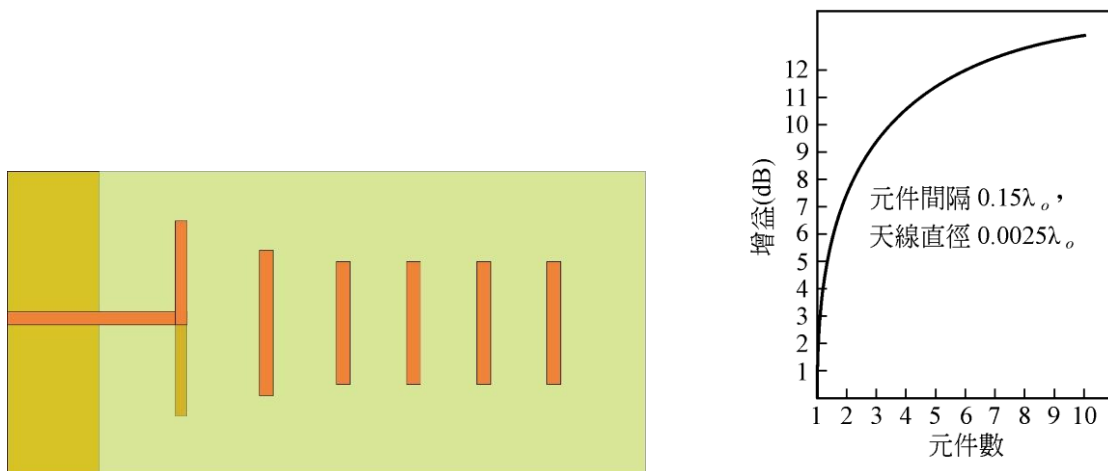
表一、GPIO 腳位與相移器控制位元規劃表

Phase shifter #1			GPIO			Phase shifter #2			GPIO			Phase shifter #3			GPIO		
BIT6	pin #22		9			BIT6	pin #22		25			BIT6	pin #22		18		
BIT5	pin #23		5			BIT5	pin #23		8			BIT5	pin #23		17		
BIT4	pin #24		6			BIT4	pin #24		12			BIT4	pin #24		27		
BIT3	pin #26		13			BIT3	pin #26		16			BIT3	pin #26		22		
BIT2	pin #27		19			BIT2	pin #27		20			BIT2	pin #27		23		
BIT1	pin #28		26			BIT1	pin #28		21			BIT1	pin #28		24		

樹莓派的 Python 程式則在附錄一中呈現。

2.4 天線單元與陣列天線

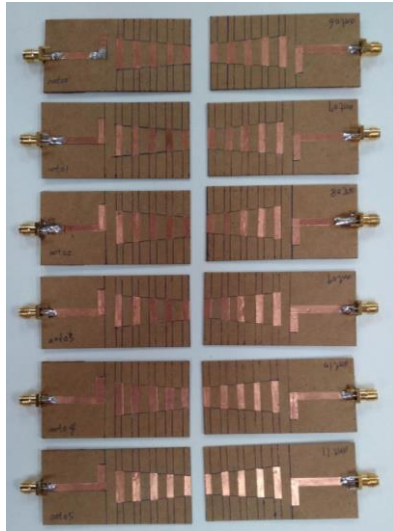
本計畫中採用如圖二十所示之平面式八木天線做為陣列單元。八木天線具有高指向性、低成本、構造簡單的特性。於1926年，由宇田(Uda)先生在日本Tohoku大學完成，1928年由其老師八木(Yagi)先生寫成英文論文發表，因此也有人將之稱為八木宇田天線(Yagi-Uda Antennas)。八木天線最常做為無線電視天線，其結構包括：驅動器(Driver)，反射器(Reflector)，導向器(Director)，而其中只有驅動器有訊號饋入。由驅動器產生電磁場進而激發其餘元件，產生類似陣列的效果，以提高天線增益。此一作法稱為寄生陣列(Parasitic Array)。研究指出，如圖十四所示，各元件最佳距離在 $0.15\sim 0.25$ 波長之間，反射器比驅動器約長5%，而導向器則比驅動器約短5%。研究也發現，增加反射器數目對提高增益並無太大幫助；增加導向器的數目，可以有效提高天線增益。



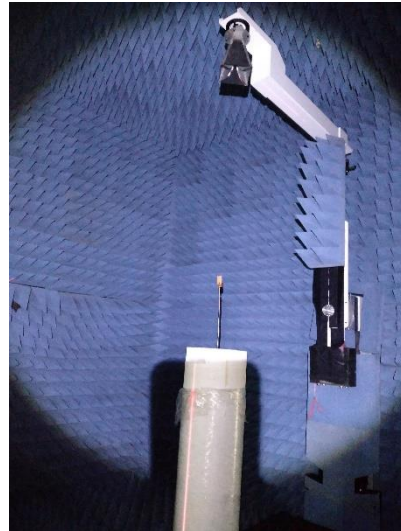
圖二十、平面式八木天線結構圖及增益-元件數之典型關係圖

圖二十一所示為實際完成之5.9 GHz之成品及場型量測圖。共製作12支成品分別編號ant00~ant11，隨機挑選ant00、ant03、ant08、ant11進行量測，其返回損失量測結果如圖二十二所示。其結果顯示，本設計之印刷式八木天線在5.9GHz操作頻率時，其S11都在-20dB以下，匹配非常完美。

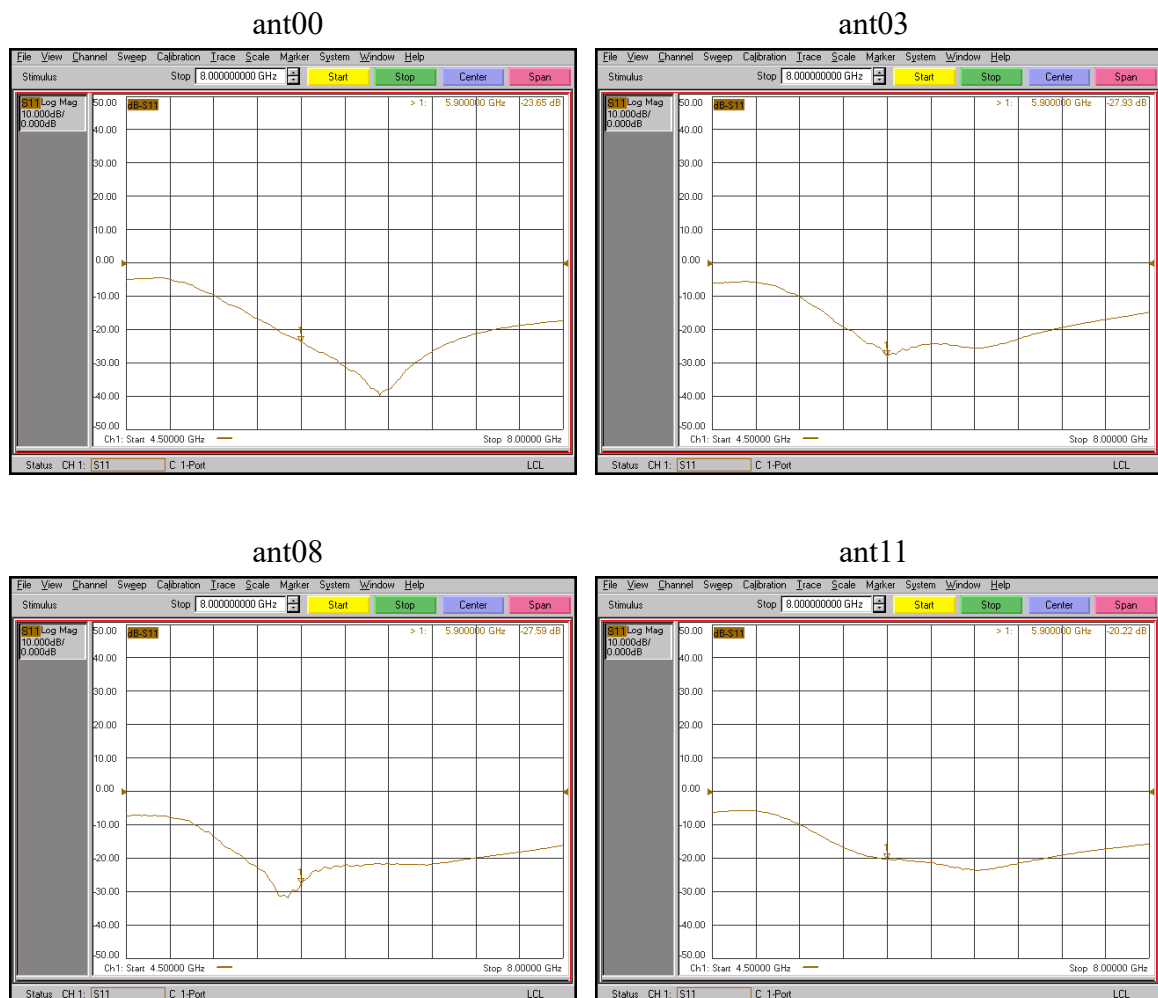
(a)



(b)

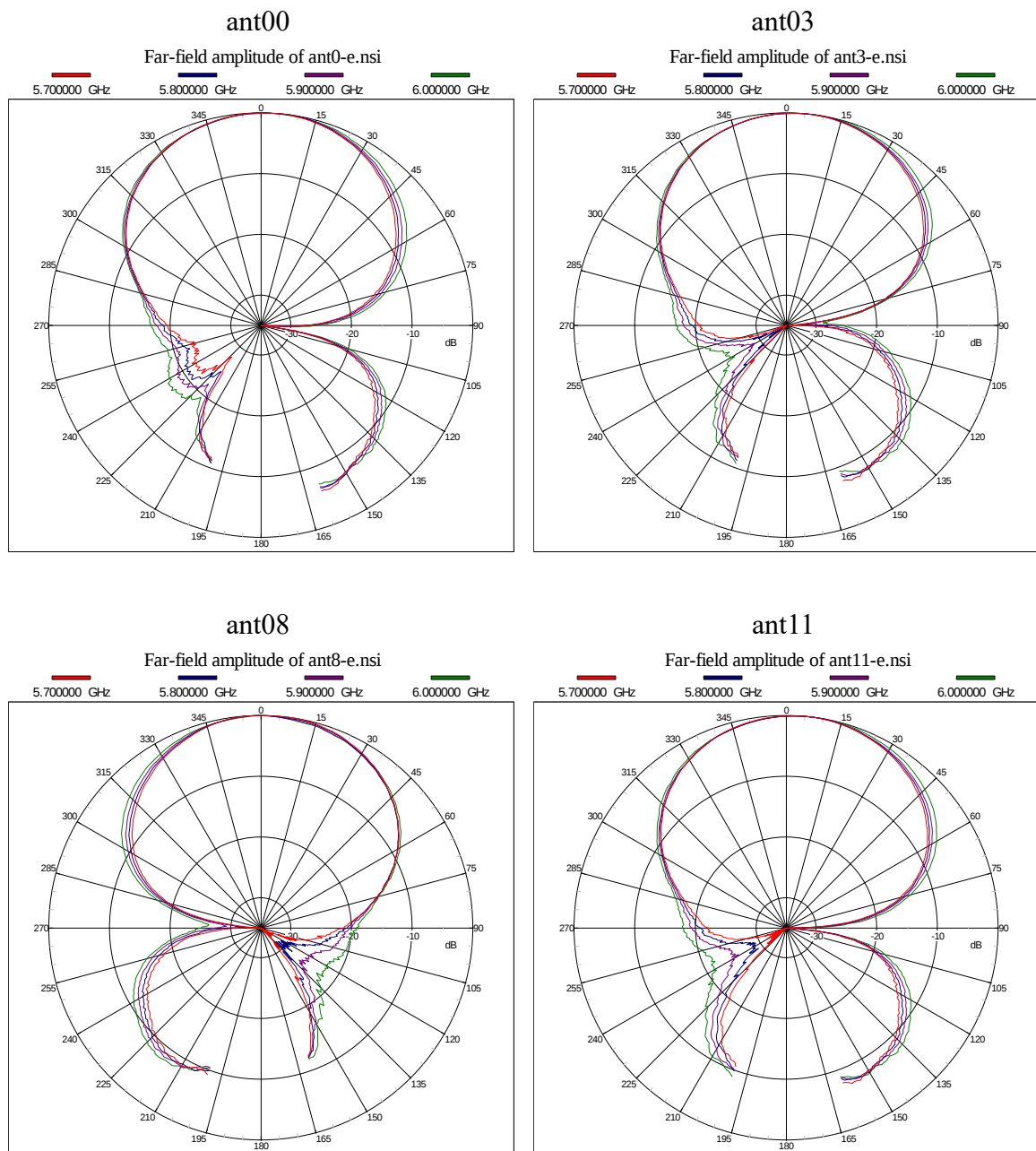


圖二十一、5.9 GHz 印刷式八木天線之成品及場型量測圖

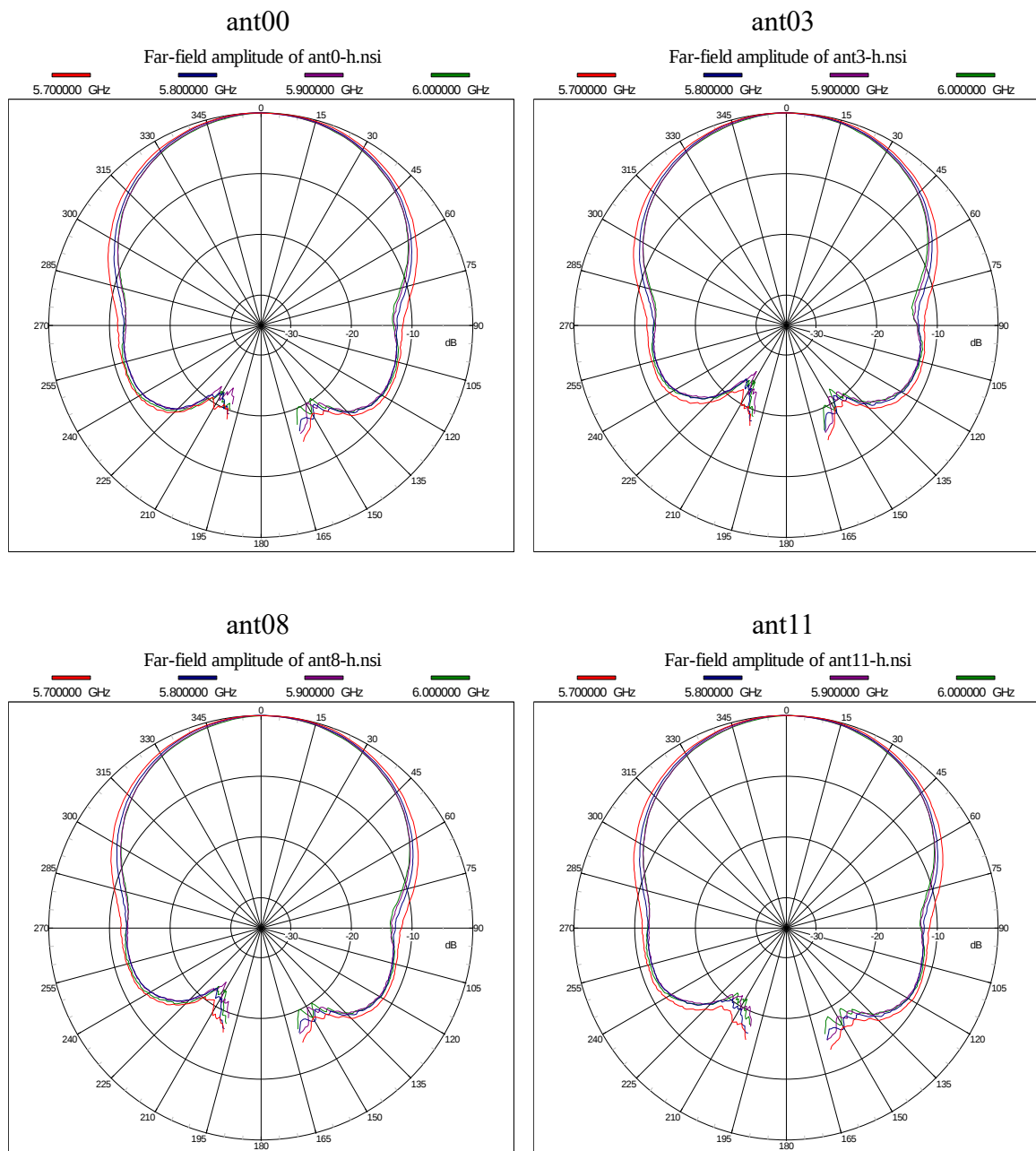


圖二十二、隨機挑選 4 支天線之返回損失量測結果

圖二十三所示為 ant00、ant03、ant08、ant11 電場平面(E-plane)輻射場型量測結果。圖二十四則為磁場平面(H-plane)輻射場型量測結果。由結果可以看出來，本設計之印刷式八木天線在 5.7~6.0 GHz 操作頻率下，E-plane 跟 H-plane 場型都是寬邊輻射(Broadside)，其 3dB 束寬(半功率束寬 half-power beam width)約為 75 度，適合本項教學系統的需求。



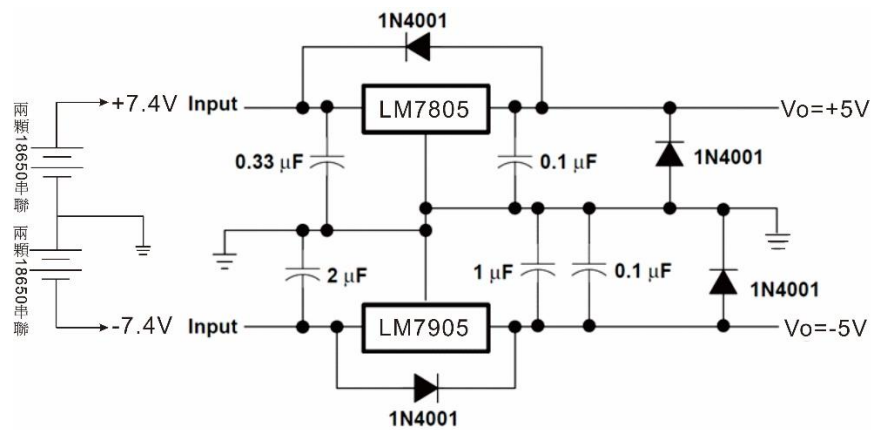
圖二十三、ant00、ant03、ant08、ant11電場平面(E-plane)輻射場型量測結果



圖二十四、ant00、ant03、ant08、ant11磁場平面(H-plane)輻射場型量測結果。

2.5 電源供應電路

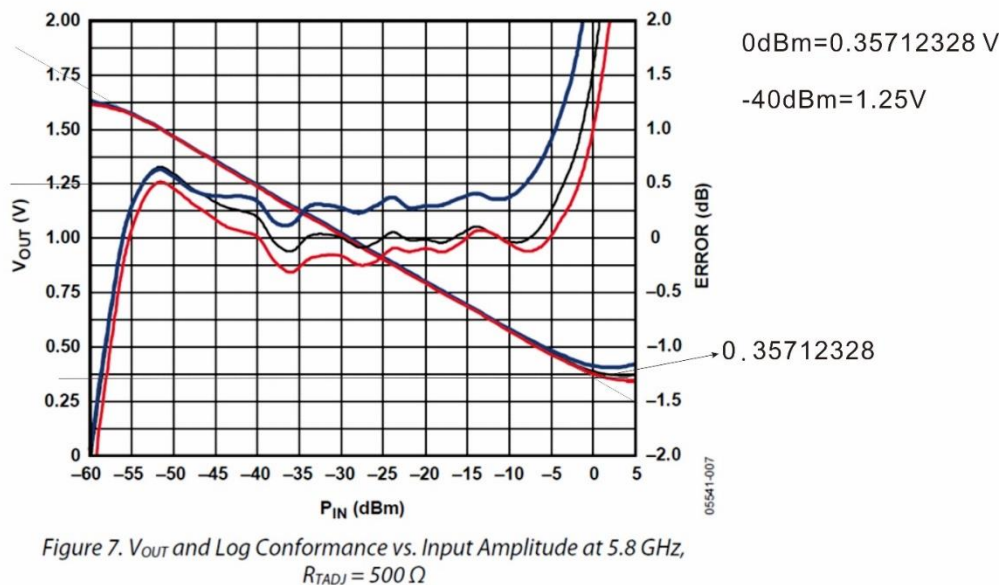
綜觀主功能板各部偏壓設計，直流電源需求為+5V、-5V 等兩個電壓。因此計畫中採取四個 18650 鋰電池串聯，再以中間抽頭當作電位參考點的方式來獲得+7.4V 及-7.4V。此一 $\pm 7.4\text{V}$ 的電源，輸入到由 LM7805 及 LM7905 所構成的穩壓電路，即可獲得穩定的 $\pm 5\text{V}$ 的電源供應。相關電路設計如圖二十五所示。



圖二十五、由 LM7805 及 LM7905 所構成的穩壓電路圖

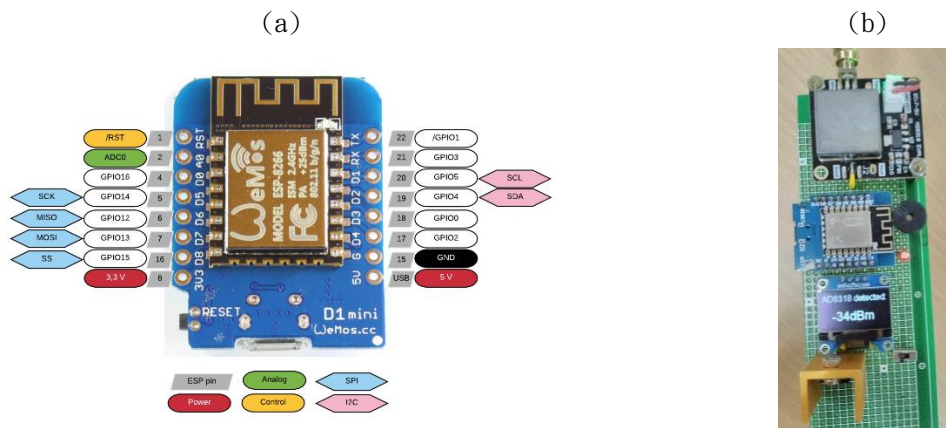
2.6 接收指示器

接收指示器是一個簡單的微波檢波器，當他接收到微波功率時，將點亮LED燈、蜂鳴器發出“嗶”聲、並在OLED螢幕上顯示接收到的訊號強度。接收指示器的檢波功能主要由亞德諾半導體生產之AD8318晶片來完成。由外接天線將接收到的電磁波功率輸入給AD8318，AD8318將輸入的功率轉換成電壓輸出。當操作在5.8 GHz時，AD8318的功率-電壓轉移曲線如圖二十六所示：輸入功率0 dBm時，輸出電壓0.36V；輸入功率-40 dBm時，輸出電壓1.25V。再將AD8318的輸出電壓饋入到WeMos D1 Mini的開發板進行運算，即可判斷所接收到的電磁波功率是否高於臨界值。一旦高於臨界值，則可啟動點亮LED燈、蜂鳴器發出“嗶”聲、並在OLED螢幕上顯示接收到的訊號強度等動作。



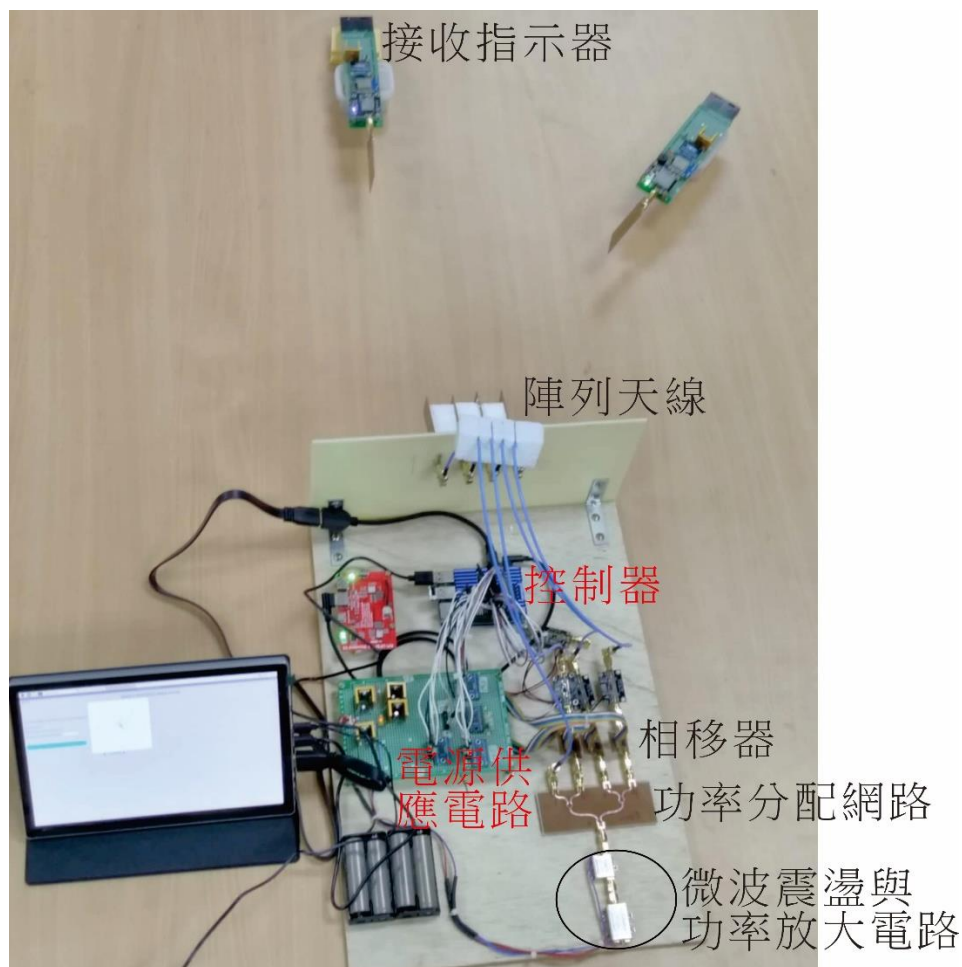
圖二十六、AD8318的功率-電壓轉移曲線

WeMos D1 Mini開發板的接腳圖如圖二十七(a)所示，AD8318感測器 OUT 連接 A0、LED 連接 D3、蜂鳴器正極 (+) 連接 D5。WeMos D1 Mini架構同ESP-8266，因此可用Arduino IDE來開發其應用程式。將上述動作要求寫成應用程式，編譯之後上載到D1 Mini晶片內即可。完成的接收指示器如圖二十七(b)所示，程式碼則於附錄二中呈現。



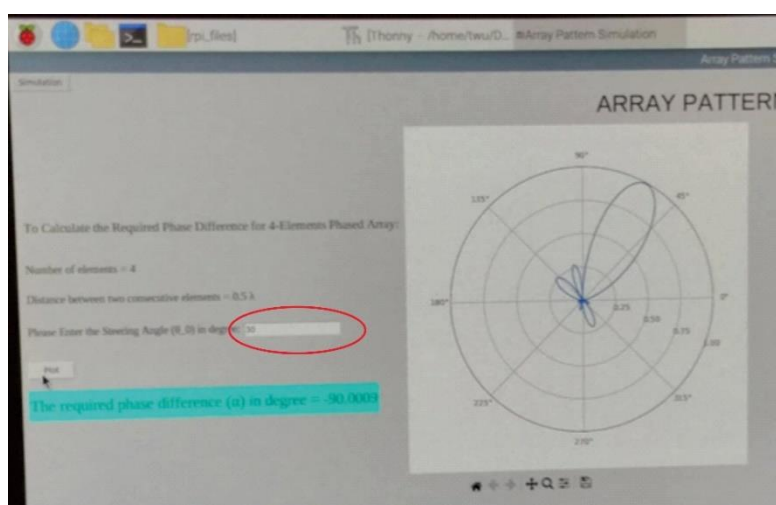
圖二十七、(a)WeMos D1 Mini 開發板接腳圖
(b)接收指示器完成圖

將上述2.1~2.6節完成的各部，依序組合成一個完整的「相控陣列天線教學展示系統」，如圖二十八所示。



圖二十八、「相控陣列天線教學展示系統」完成圖

簡單說明此一「相控陣列天線教學展示系統」的功能如下：在樹莓派上執行附錄一的Python程式，其畫面如圖二十九所示。在紅圈處輸入希望主波束偏轉的角度(在此以30度為例)，畫面右側為模擬場型圖。而輸入的偏轉角度30度經過程式計算，得到每一個相移器的相移角度之後，將之轉換為6位元的二進位控制信號。6位元的控制信號輸入至相移器的控相腳位，以得到各分量該有的相移量。經過相位移之後的各分量饋入至1×4的天線陣列輻射出去。而陣列的輻射場型，則由接收指示器來檢驗。



圖二十九、樹莓派上執行附錄一Python程式的畫面

如圖二十八所示，在與主功能板固定距離的扇形0度與30度位置上各擺放一個接收指示器，當在樹莓派的程式裡設定要求主功能板上的「相控陣列天線」主波束偏轉30度時，擺放在30度位置上的接收指示器，參見圖三十(a)，會接收到足夠強的功率，進而驅動包括：點亮LED燈、蜂鳴器發出“嗶”聲、並在OLED螢幕上顯示接收到的訊號強度等動作。而擺放在0度位置上的接收指示器，參見圖三十(b)，則因為接收不到足夠的功率強度，僅在OLED螢幕上顯示接收到的訊號強度，其他LED燈與蜂鳴器並不會有任何反應。如此我們相信透過本套「相控陣列天線教學展示系統」可立即得知目前主波束所偏轉的角度，可以給學生對「相控陣列天線」建立具體的觀念，能有效提昇教學成效。

(a)



(b)



圖三十、(a)擺放在30度位置上的接收指示器的反應
(b)擺放在0度位置上的接收指示器的反應

第 三 章

結 論

本計畫主要在於精進本校（陸軍軍官學校）電機系「電波專題」課程。本計畫執行成效卓著，透過計畫之執行，修習本課程之學生皆能嫻熟天線設計與量測技術、微波電路設計與量測技術、雷達系統、相控陣列天線之原理與操作技術。此外，計畫內完成的「相控陣列天線教學展示系統」，可以幫助學生建立具體的「相控陣列天線」觀念，能有效提昇教學成效。經由系統化之整合與教學，達到將專業知識應用於軍事實務之教學目標。計畫執行成果包含設計原理與製作流程，國軍各教學單位若有興趣製作，計畫主持人非常樂意協助。

參考文獻

- [1] G.W. Stimson, H. Griffiths, C. Baker, and D. Adamy, “Introduction to Airborne Radar, 3rd Edition,” SciTech Publication, 2014.
- [2] M.A. Richards, J.A. Scheer, W.A. Holm, “Principles of Modern Radar: Basic Principles,” SciTech Publication, 2013.
- [3] B.R. Mahafza, “Radar Signal Analysis and Processing Using MATLAB,” CRC Press, 2009.
- [4] G.L. Charvat, “The MIT IAP radar course: Build a small radar system capable of sensing range, Doppler, and synthetic aperture (SAR) imaging.” IEEE Radar Conference (RADAR), 7-11 May, 2012.
- [5] G.L. Charvat, “Doppler Radar Explanation and Demo using the coffee can radar [YouTube video],” Available:
<https://www.youtube.com/watch?v=FOWopYv-JTM>, Jan. 6, 2012.
- [6] G.L. Charvat, *Small and Short-Range Radar System*. NW, USA: CRC Press, 2014.
- [7] C.A. Fowler, “Old Radar Types Never Die; They Just Phased Array, or 55 Years of Trying to Avoid Mechanical Scan,” IEEE Aerosp. Electron. Syst. Mag. 13 (9), 1998, pp. 24A–24L.
- [8] G.N. Tsandoulas, “Unidimensionally Scanned Phased Arrays,” IEEE Trans. Antennas Propag. 28 (1), 1980, pp. 86–99.
- [9] G.N. Tsandoulas and G.H. Knittel, “The Analysis and Design of Dual-Polarization Square-Waveguide Phased Arrays,” IEEE Trans. Antennas Propag. 21 (6), 1973, pp. 796–808.

- [10] A.J. Fenn, "Theoretical and Experimental Study of Monopole Phased Array Antennas," IEEE Trans. Antennas Propag. 34(10), 1985, pp. 1118–1126.
- [11] G.H. Knittel, "Relation of Radar Range Resolution and Signal-to-Noise Ratio to Phased-Array Bandwidth," IEEE Trans. Antennas Propag. 22 (3), 1974, pp. 418–426.
- [12] E. Stern and D.H. Temme, "Magnetostriction Effects in Remanence Phase Shifters," IEEE Trans. Microw. Theory Tech. 13(6), 1965, pp. 873–874.
- [13] H.M. Aumann, A.J. Fenn, and F.G. Willwerth, "Phased Array Antenna Calibration and Pattern Prediction Using Mutual Coupling Measurements," IEEE Trans. Antennas Propag. 37(7), 1989, pp. 844–850.
- [14] E. J. Wilkinson, "An N-Way Hybrid Power Divider," IRE Trans. Microwave Theory and Techniques, vol. 8, no. 1, pp.116-118, January 1960.

附 錄 一

樹 莓 派 控 制 器 之 P y t h o n 程 式

```
import tkinter as tk
from tkinter import Tk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import (FigureCanvasTkAgg, NavigationToolbar2Tk)
from matplotlib.backend_bases import key_press_handler
from matplotlib.figure import Figure
import numpy as np
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
#####phase shifter #1  bit1--> 26, bit2-->19, bit3-->13, bit4-->6, bit5-->5, bit6-->9 #####
ps1b1 = 26
ps1b2 = 19
ps1b3 = 13
ps1b4 = 6
ps1b5 = 5
ps1b6 = 9
GPIO.setup(ps1b1, GPIO.OUT)
GPIO.setup(ps1b2, GPIO.OUT)
GPIO.setup(ps1b3, GPIO.OUT)
GPIO.setup(ps1b4, GPIO.OUT)
GPIO.setup(ps1b5, GPIO.OUT)
GPIO.setup(ps1b6, GPIO.OUT)
#####phase shifter #2  bit1--> 21, bit2-->20, bit3-->16, bit4-->12, bit5-->8, bit6-->25 #####
ps2b1 = 21
ps2b2 = 20
ps2b3 = 16
ps2b4 = 12
ps2b5 = 8
ps2b6 = 25
GPIO.setup(ps2b1, GPIO.OUT)
GPIO.setup(ps2b2, GPIO.OUT)
GPIO.setup(ps2b3, GPIO.OUT)
GPIO.setup(ps2b4, GPIO.OUT)
GPIO.setup(ps2b5, GPIO.OUT)
GPIO.setup(ps2b6, GPIO.OUT)
#####phase shifter #3  bit1--> 24, bit2-->23, bit3-->22, bit4-->27, bit5-->17, bit6-->18 #####
ps3b1 = 24
ps3b2 = 23
ps3b3 = 22
ps3b4 = 27
ps3b5 = 17
ps3b6 = 18
GPIO.setup(ps3b1, GPIO.OUT)
GPIO.setup(ps3b2, GPIO.OUT)
GPIO.setup(ps3b3, GPIO.OUT)
GPIO.setup(ps3b4, GPIO.OUT)
GPIO.setup(ps3b5, GPIO.OUT)
GPIO.setup(ps3b6, GPIO.OUT)
```

```

class Page(tk.Frame):
    def __init__(self, *args, **kwargs):
        tk.Frame.__init__(self, *args, **kwargs)
    def show(self):
        self.lift()

class Page4(Page):
    def __init__(self, *args, **kwargs):
        Page.__init__(self, *args, **kwargs)
        lbl = tk.Label(self, text="ARRAY PATTERN SIMULATION", font=("Arial", 30))
        lbl.pack(side=tk.TOP)

        Enter = tk.Label(self, text="To Calculate the Required Phase Difference for 4-Elements Phased Array:", font=("Times
New Roman", 16))
        Enter.place(height=50, x=10, y=200)

        Enter = tk.Label(self, text="Number of elements = 4", font=("Times New Roman", 14))
        Enter.place(height=50, x=10, y=270)

        Enter = tk.Label(self, text="Distance between two consecutive elements = 0.5 \u03BB ", font=("Times New Roman",
14))
        Enter.place(height=50, x=10, y=320)

##### modified by TWU, enter the steering angle to calculated the phase difference

        Theta0 = tk.Label(self, text="Please Enter the Steering Angle (\u03B8_0) in degree:", font=("Times New Roman", 14))
        Theta0.place(height=50, x=10, y=375)

        Theta0B = tk.Entry(self, width=20)
        Theta0B.place(height=25, x=370, y=390)

        pi = np.pi
        efactor= np.loadtxt('ef.txt')

        def get_bit_val(byte, index):
            if byte & (1<< index):
                return 1
            else:
                return 0

        def clicked():
            n = 4
            d = 0.5
            Theta0 = float(Theta0B.get())*np.pi/180
            alpha=-2*pi*d*np.sin(Theta0)
            alphad=alpha*180/np.pi

            Alpha = tk.Label(self, bg='cyan', fg='black', text="The required phase difference (\u03B1) in degree =
{:.3f}".format(alphad) , font=("Times New Roman", 20))
            Alpha.place(height=50, x=10, y=500)

#####Phase Shifter #1 Parameters Setting#####
            alpha_d = abs(alphad) % 360
            print(alphad)
            print(alpha_d)

```

```

if alphad < 0:
    multi = round(alpha_d/5.625)
    ps_bits="{0:b}".format(round(multi))
    print(-int(ps_bits, 2)*5.625)

else:
    multi = round((360-alpha_d)/5.625)
    ps_bits="{0:b}".format(multi)
    print(-int(ps_bits, 2)*5.625)

GPIO.output(ps1b6, get_bit_val(multi, 5))
GPIO.output(ps1b5, get_bit_val(multi, 4))
GPIO.output(ps1b4, get_bit_val(multi, 3))
GPIO.output(ps1b3, get_bit_val(multi, 2))
GPIO.output(ps1b2, get_bit_val(multi, 1))
GPIO.output(ps1b1, get_bit_val(multi, 0))

#####Phase Shifter #2 Parameters Setting#####
alpha_d = abs(2*alphad) % 360
print(alpha_d)
if alphad < 0:
    multi = round(alpha_d/5.625)
    ps_bits="{0:b}".format(round(multi))
    print(-int(ps_bits, 2)*5.625)

else:
    multi = round((360-alpha_d)/5.625)
    ps_bits="{0:b}".format(multi)
    print(-int(ps_bits, 2)*5.625)

GPIO.output(ps2b6, get_bit_val(multi, 5))
GPIO.output(ps2b5, get_bit_val(multi, 4))
GPIO.output(ps2b4, get_bit_val(multi, 3))
GPIO.output(ps2b3, get_bit_val(multi, 2))
GPIO.output(ps2b2, get_bit_val(multi, 1))
GPIO.output(ps2b1, get_bit_val(multi, 0))

#####Phase Shifter #3 Parameters Setting#####
alpha_d = abs(3*alphad) % 360
print(alpha_d)
if alphad < 0:
    multi = round(alpha_d/5.625)
    ps_bits="{0:b}".format(round(multi))
    print(-int(ps_bits, 2)*5.625)

else:
    multi = round((360-alpha_d)/5.625)
    ps_bits="{0:b}".format(multi)
    print(-int(ps_bits, 2)*5.625)

GPIO.output(ps3b6, get_bit_val(multi, 5))
GPIO.output(ps3b5, get_bit_val(multi, 4))
GPIO.output(ps3b4, get_bit_val(multi, 3))
GPIO.output(ps3b3, get_bit_val(multi, 2))
GPIO.output(ps3b2, get_bit_val(multi, 1))
GPIO.output(ps3b1, get_bit_val(multi, 0))
print("*****GOOD JOB !!!*****")

```

```

phi = np.arange(0, 2 * np.pi, .01)
cos_phi = np.cos(np.array(phi))
psi = (2 * pi * d * np.array(cos_phi)) + alpha
E = (1 / n) * (np.sin(n * np.array(psi) / 2)) / np.sin(np.array(psi) / 2)*efactor
fig = Figure(figsize=(6, 6), dpi=100)
a = fig.add_subplot(111, projection='polar')
a.plot(phi,abs(E))
a.set_rmax(1)
a.set_rticks([0.25, 0.5, 0.75, 1]) # less radial ticks
a.set_rlabel_position(-22.5) # get radial labels away from plotted line
a.grid(True)

canvas = FigureCanvasTkAgg(fig, master=self) # A tk.DrawingArea.
canvas.draw()
canvas.get_tk_widget().place(x=650, y=60)
toolbar = NavigationToolbar2Tk(canvas, self)
toolbar.update()
toolbar.place(x=750, y=663)

def on_key_press(event):
    print("you pressed {}".format(event.key))
    key_press_handler(event, canvas, toolbar)

canvas.mpl_connect("key_press_event", on_key_press)

Plot = tk.Button(self, text=" Plot ", command=clicked)
Plot.place(height=30, x=10, y=450)

class MainView(tk.Frame):
    def __init__(self, *args, **kwargs):
        tk.Frame.__init__(self, *args, **kwargs)
        p4 = Page4(self)

        buttonframe = tk.Frame(self)
        container = tk.Frame(self)
        buttonframe.pack(side="top", fill="x", expand=False)
        container.pack(side="top", fill="both", expand=True)

        p4.place(in_=container, x=0, y=0, relwidth=1, relheight=1)

        b4 = tk.Button(buttonframe, text="Simulation", command=p4.lift)

        b4.pack(side="left")

        p4.show()

if __name__ == "__main__":
    root = Tk()
    main = MainView(root)
    main.pack(side="top", fill="both", expand=True)
    root.wm_title("Array Pattern Simulation")
    root.wm_geometry('1300x700')
    root.mainloop()

```

附 錄 二

接 收 指 示 器 之 A r d u i n o 程 式

```
/*
The AD8318 is a demodulating logarithmic amplifier, capable of
accurately converting an RF input signal to a corresponding
decibel-scaled output voltage.
*/
#include < pitches.h>
#include < Wire.h>
#include "SSD1306Wire.h"

#define AD8318Sensor A0          // AD8318 感測器 OUT 連接 A0
#define ledPin D3                // LED 連接 D3
#define buzzer D5                // 蜂鳴器正極 (+) 連接 D5

int vmin_i = 1023;               // AD8318 感測器訊號變數初始值
int nloop = 1000;
int vdet;
long vdet2dbm;
SSD1306Wire display(0x3c, D2, D1);

void setup() {
  pinMode(AD8318Sensor, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(115200);
  Serial.println();
  Serial.println();

  // Initialising the UI will init the display too.
  display.init();

  display.flipScreenVertically();
}

void detect(){
  vmin_i = 1023;
  //
  for (int iloop = 1; iloop <= nloop; iloop++) {
    vdet = analogRead(AD8318Sensor);
    if (vdet < vmin_i) {
      vmin_i=vdet;
    }
    //delayMicroseconds(14);
  }
}

void loop(){
  detect();          //讀取 AD8318 感測器訊號
  vdet2dbm=(vmin_i-204)*(-0.12);
  display.clear();
```

```

Serial.println(vmin_i);           //將訊號值印在序列埠螢幕上
display.setFont(ArialMT_Plain_16);
display.drawString(0, 0, "AD8318 detected:");
display.setFont(ArialMT_Plain_24);
display.drawString(20, 30, String(vdet2dbm) + "dBm");
delayMicroseconds(1000);
display.display();

                                //判斷 Vmin 的狀態
if (vmin_i <= 411) {           //如果 Vmin 低於臨界值則啟動動作+
    digitalWrite(ledPin, HIGH);
    tone(buzzer, NOTE_C5);
    delay(200);
}
vmin_i = 1023;
digitalWrite(ledPin, LOW);
tone(buzzer, 0);
}

```