微算機期末專題企劃書

A專題動機與願景

台灣是我們成長的土地，在這塊滋養我們的土地上奮鬥打拼時，我們卻常常忘記其他在這塊土地的朋友。

2018年年底，社群媒體發狂似的著墨於鬥爭，哪位政治人物提了那些願景，誰與誰又針鋒相對。
他們知道，民眾現在想看甚麼。

然而，有更迫切需要我們關注的議題消失在嗜血的聲浪之中。

2018年1月16日，打著環境保護及休閒觀光的名號，大安溼地公園開工了。

諷刺的是，工程途中，石虎天然生存環境被破壞殆盡。取而代之的是美麗的人工池塘與可愛的石虎雕像。

儘管在相關保育團體的努力下，石虎的生存權在努力被搶救中，但仍舊有許多朋友需要被關注。

動物保育，是一個人變成十個人，最後變成一千個與一萬個的過程。
但於此同時，我們的朋友也可能一萬個變成一千個，甚至剩下一個。

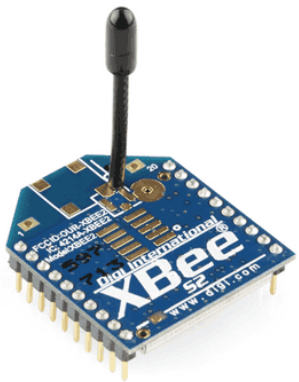
為了讓這些事情被每個人看見，我們決定起身動作，邁出我們的第一步，結合微算機課程所學的。讓象徵我們理念的麋鹿勇往直前。

B系統功能與原理說明

分為**搖桿**跟**麋鹿本體**。其中兩者上面都有**XBee**，所以這部份先特別拿出來講。

XBee

市場上三大無線模組為wifi，藍芽，ZigBee。而我們這次選用的是Digi公司的基於ZigBee出的**XBee S2C**。至於為甚麼我們選用這個呢?原因是想要學習使用比較不常見的模組來**挑戰自己**。XBee雖然傳輸速率較慢，但是可以支援六萬個以上的節點互聯，這在目前正流行的物聯網上是一個很大的優勢。



因為XBee的原理說明部分較多，因此我們放在F及G這兩部分，稍後會做一個詳細的說明

搖桿

上面有 XBee ,Joystick,雷射發射器

- XBee

透過pin20，pin18，pin11收集數據，無線傳輸給麋鹿上的XBee

- 雷射發射器

用以幫麋鹿**充能**

- Joystick

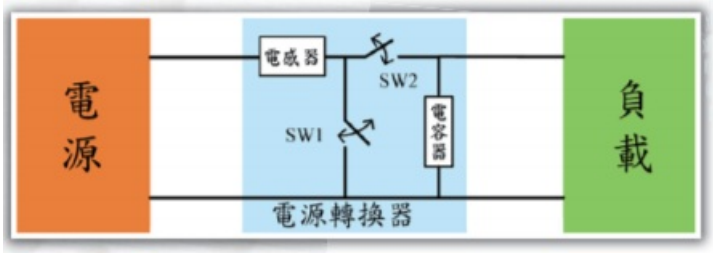
用以控制麋鹿的移動，其中:
x軸控制麋鹿左右移動並接到XBee的pin20(ADC)，
y軸控制麋鹿前後移動並接到pin18(ADC)，
按壓接到pin11(Digital input)

麋鹿

有升壓模組，燈條，雙通道直流電機驅動模組，減速馬達，雷射接收模組，紅外線避障模組，XBee

- 升壓模組

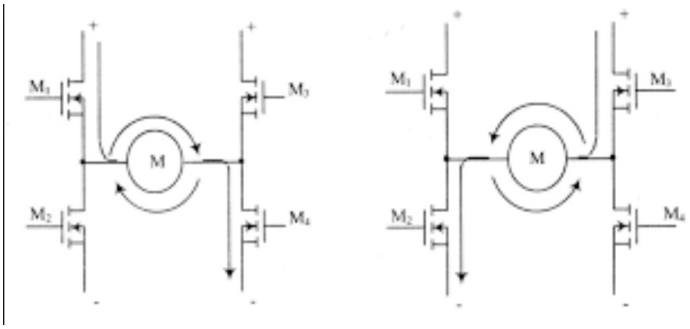
因為燈條的工作電壓需要 12V ，然而 PIC 的電壓只有 5V 便需要藉由升壓模組將電壓調成適當的電壓。



其原理主要藉由電感平穩電流，去**充電**電容，使其電壓上升，藉以達成升壓效果。

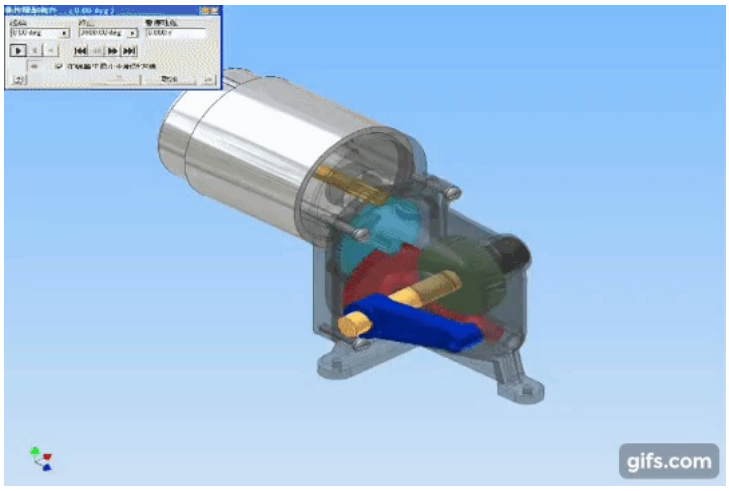
- 燈條
  會亮，很炫，麋鹿的生命象徵。
- L9110S雙通道直流電機驅動模組

其內部晶片有個 H 電橋藉由不同的訊號來源控制給馬達的電流方向。



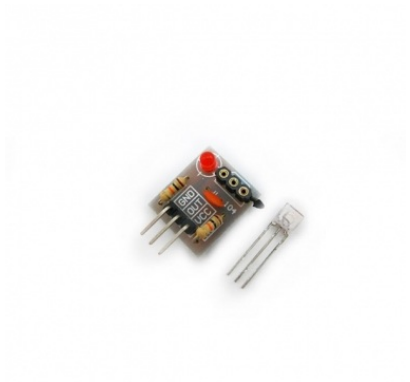
- 減速馬達

因為一般的馬達轉速過快，不太適合直接驅動遙控車的輪子，便須搭配齒輪來降低馬達的速度。



GIF Reference

- 雷射接收模組



當接收器感測到雷射光，便會藉由上面的晶片處理出 0 1 數位訊號給 PIC ， PIC 再開啟升壓模組以開啟燈條。

- 紅外線避障模組

上面有紅外線發射器跟紅外線接收器，當前方有障礙物紅外線便會被反彈回來，紅外線接收器收到紅外線後便會藉由內部晶片處理根據我們所需要的距離回傳數位訊號。

C創意特色描述

可愛的麋鹿，你怎麼能抵抗，你怎麼能不愛。看看他純潔無瑕發光的眼睛，好像看穿你的心思；屹立的鹿角，衝刺時可以幫你突破所有的困境；肥肥短短的尾巴，好像可以掃去你所有的煩惱。

除此，為了提倡生態保育的概念，我們有了雷射光跟燈條這個互動功能，雷射光代表我們對於保護環境的熱情，只要我們願意付出。麋鹿(生態環境)接收到了就有可能繼續在這地球上繼續綻放光芒，反之我

們不願意付出，可愛的麋鹿終究黯淡無光。

而且麋鹿的控制還是使用少見的XBee喔!

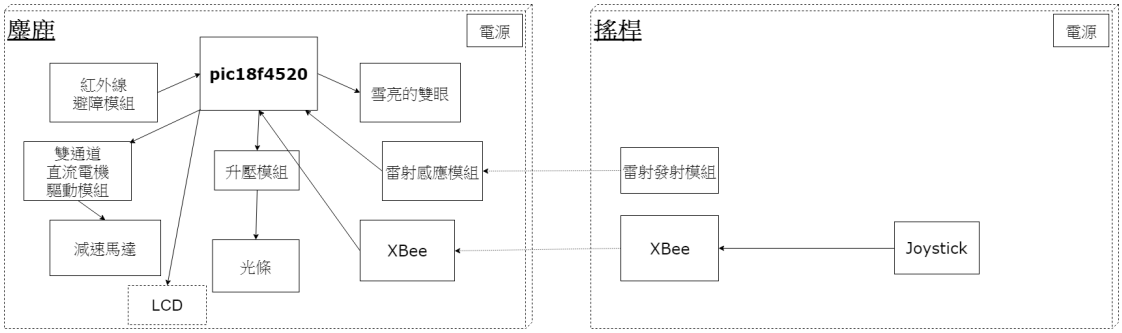
D系統使用環境及對象

狂野的麋鹿，應該可於奔馳在各個地方。由於我們的高底盤設計，在崎嶇的道路上，也不怕上面的零件受到傷害，再加上大的橡膠輪胎，無論何種材質，濕滑的地面，也可以順利奔跑。

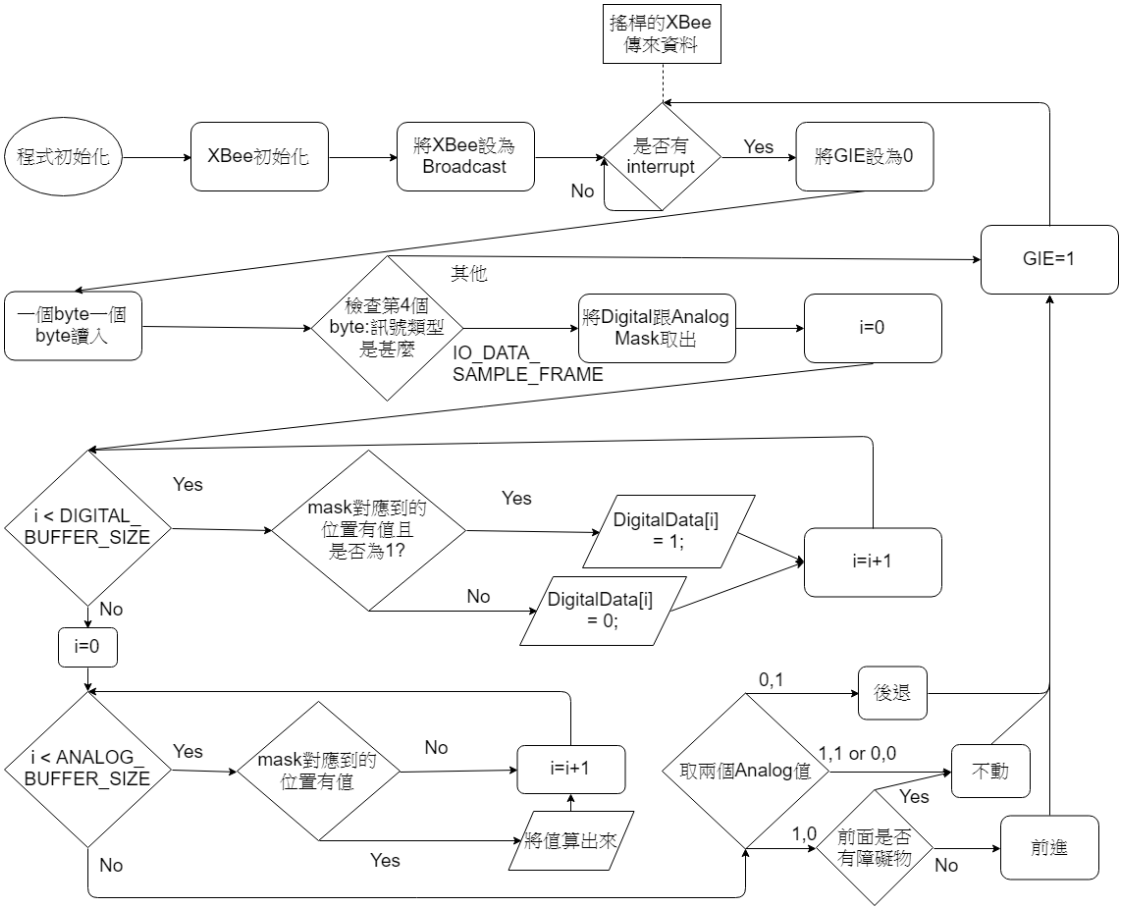
不管你是男是女，小孩還是大朋友，從事何種職業，可愛的麋鹿都將療癒你的身心。

E系統完整架構圖、流程圖、電路圖、設計圖

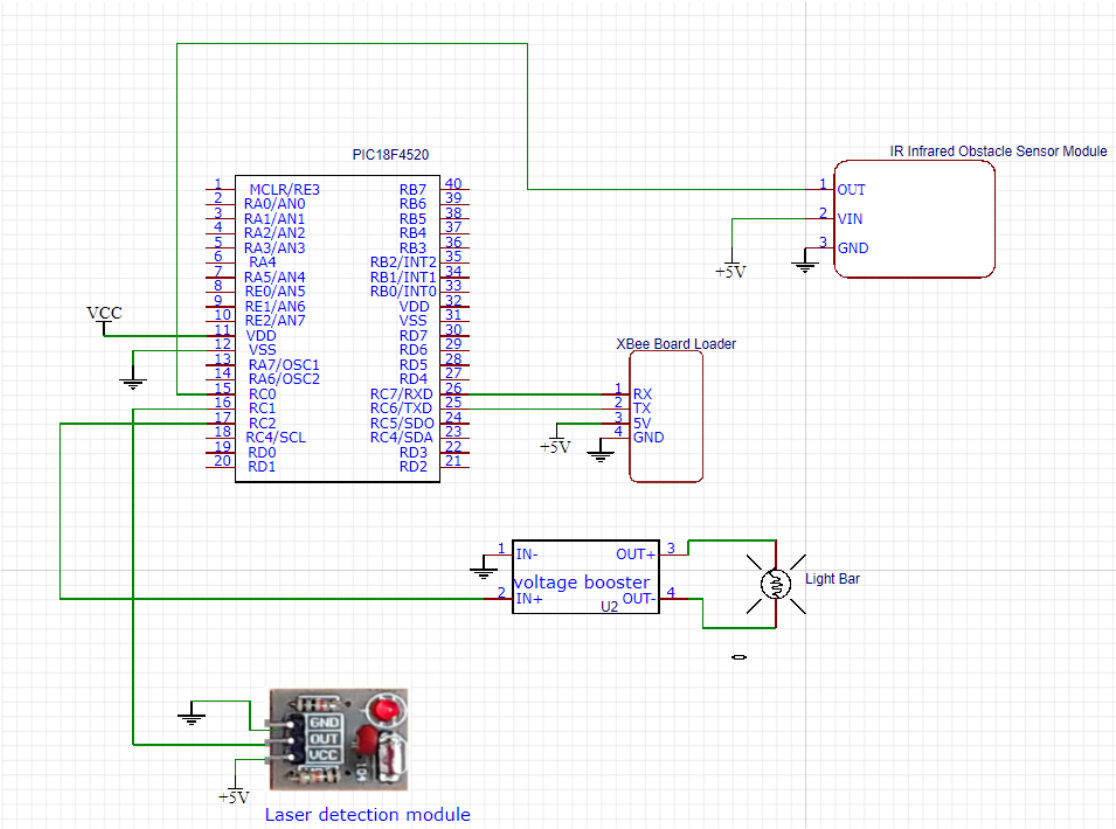
- 架構圖:

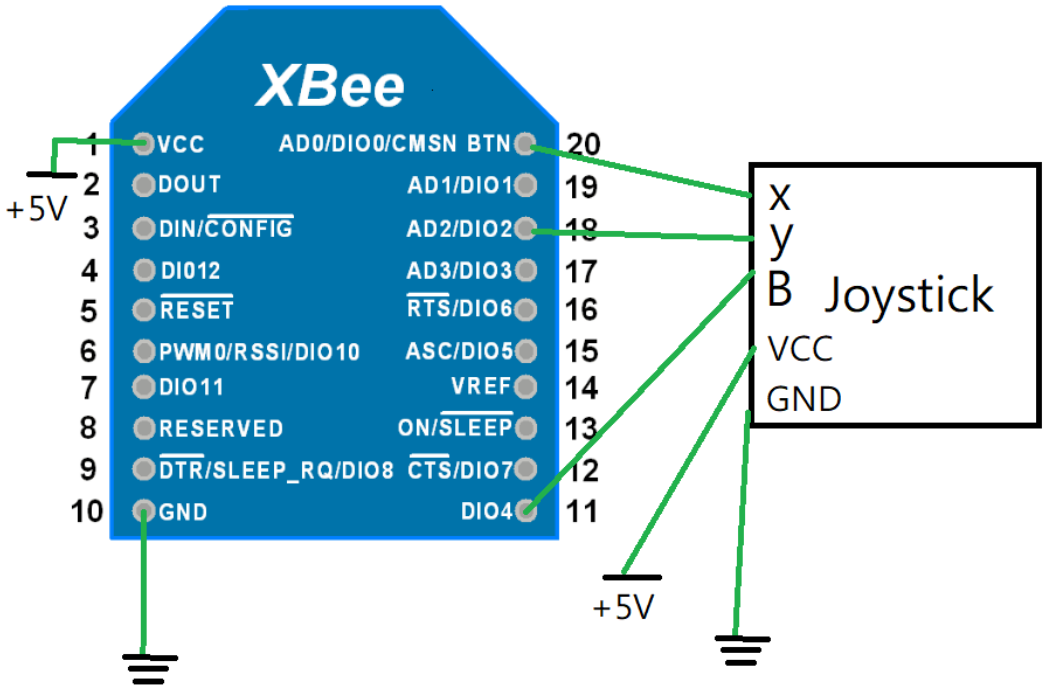


- 流程圖:



- 電路圖:



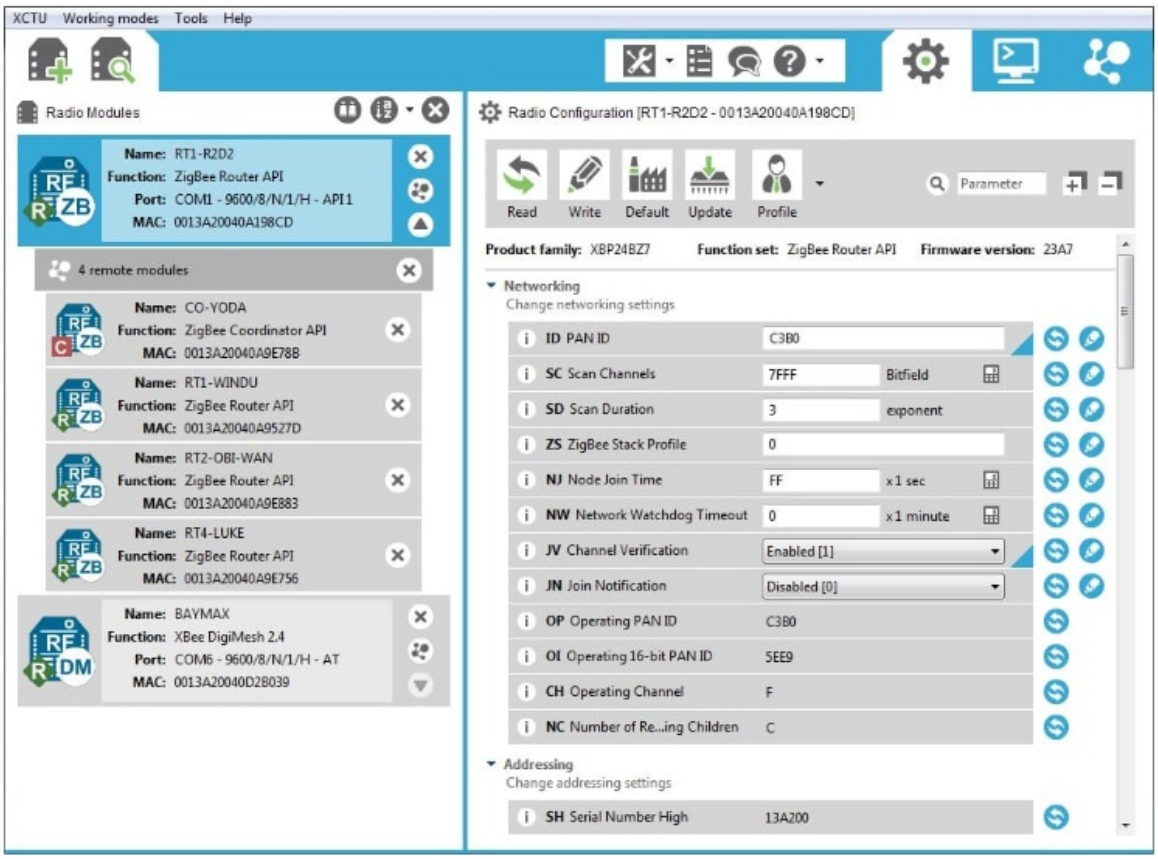


F系統開發工具、材料及技術

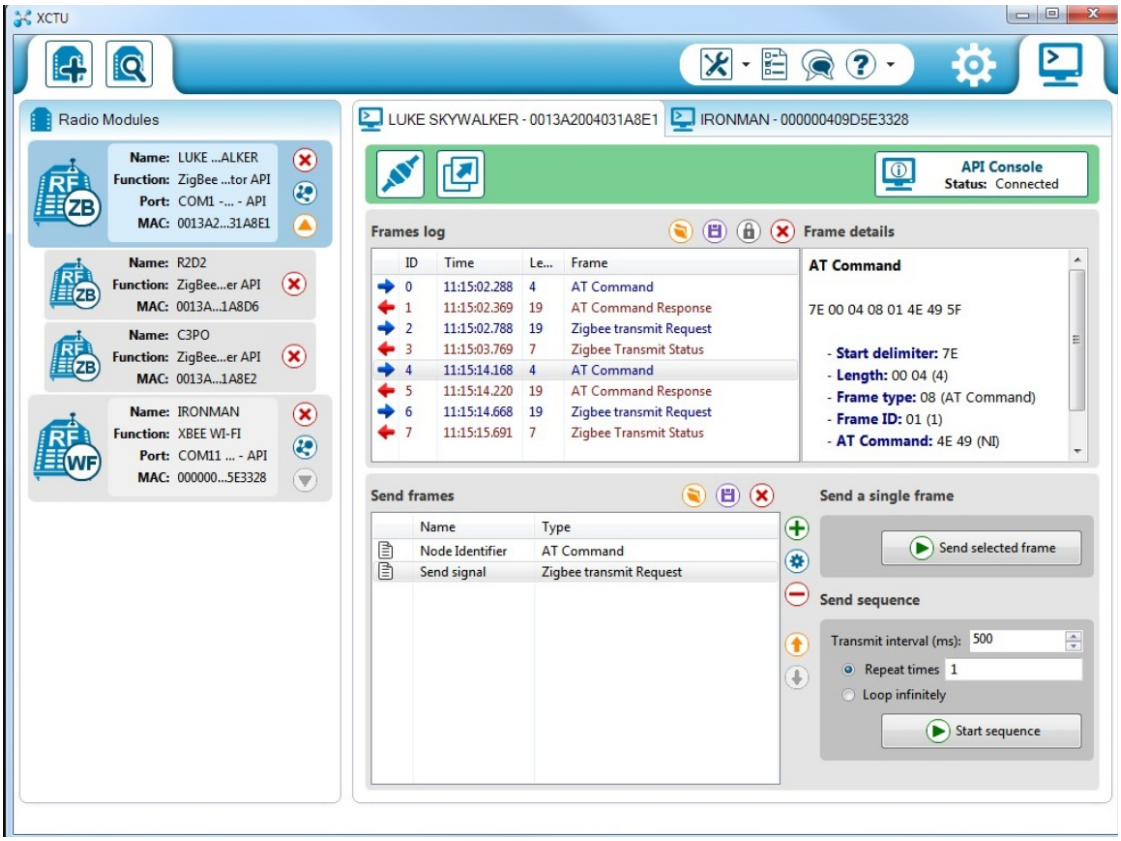
XBee

開發方面我們使用了:
(1)**兩個XBee**(這部分在G項會做詳細說明)
(2)Digi公司設計的**XCTU**軟體的協助來完成工作

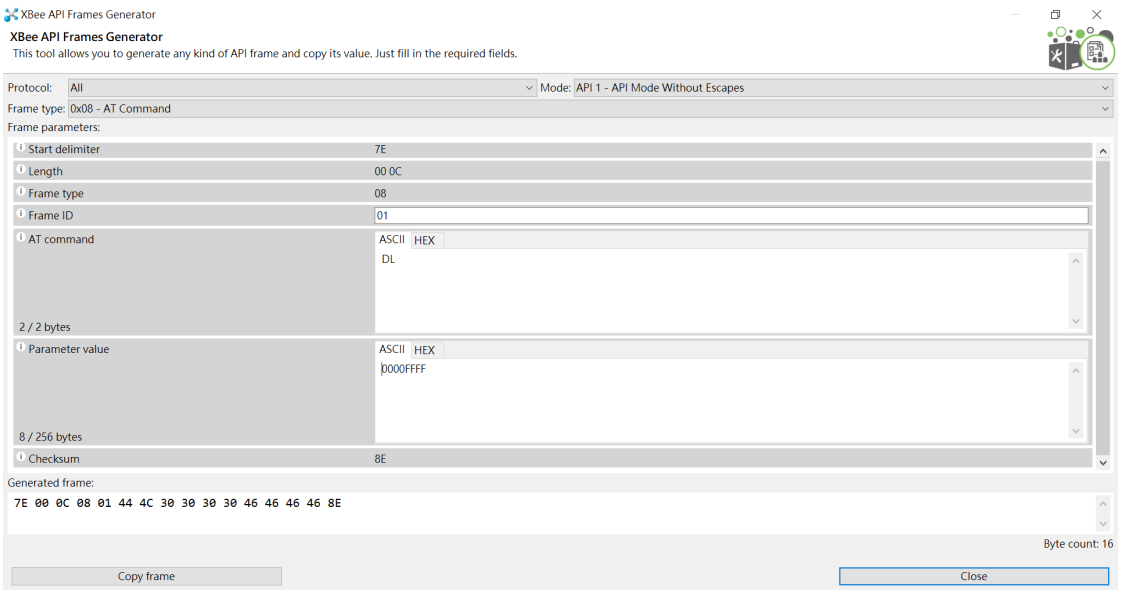
XCTU:(這部分的圖為上網抓取)



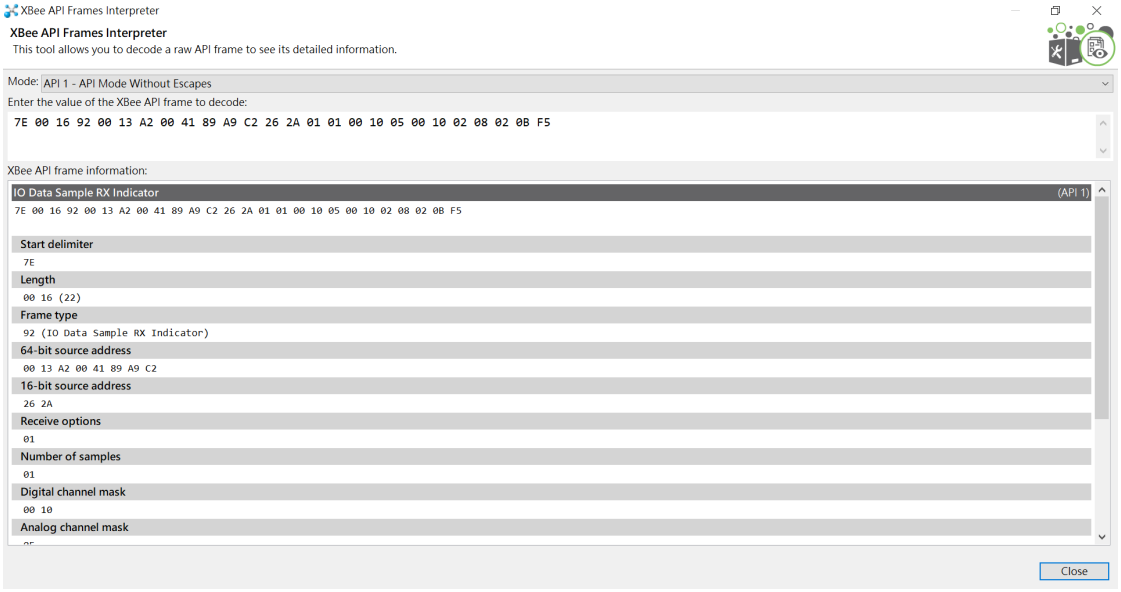
- 藉由這工具，我們可以對XBee做初始化設定(在G項我們會詳細說明)
- 可以透過這個XCTU的Console mode來查看封包是否正確傳遞，以及傳遞內容



- 使用Frame Generator去看封包的產生是否正確



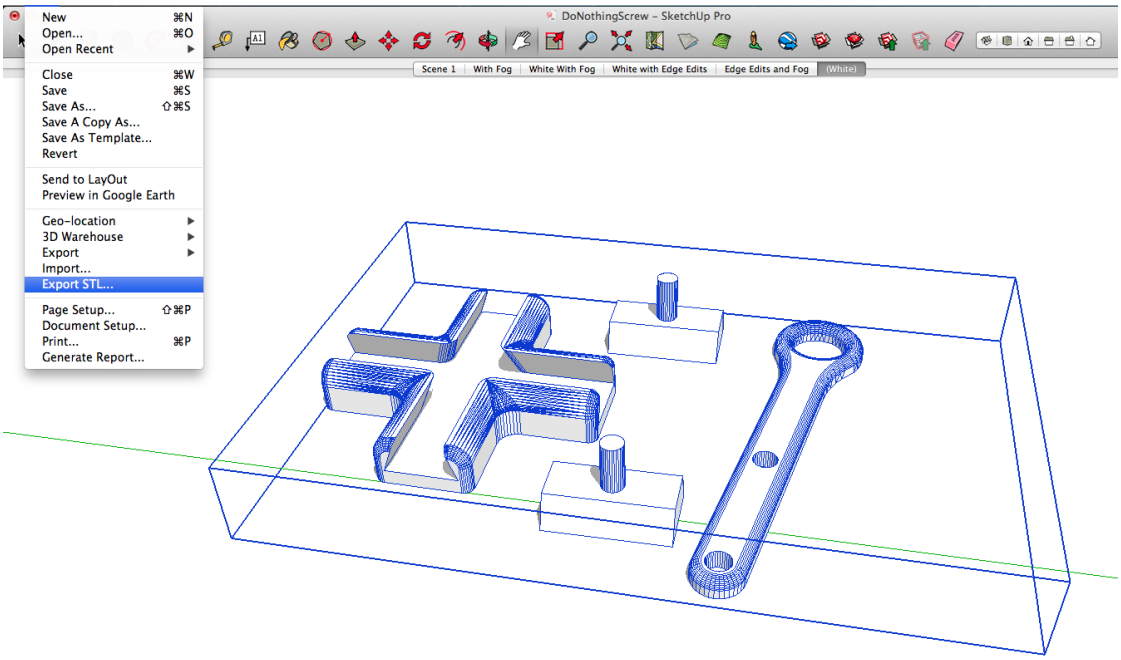
- 使用Frame Interpreter去看封包的解析是否正確



車體

整個車體大致上由木板以及3D列印出來的零件組成

- SketchUp
  我們使用SketchUp來做3D列印的建模
  而其中要特別裝SketchUp STL的插件以匯出3D列印用的STL檔



- 木板
  木板主要是從文具店購得雕刻板再用鋸成我們想要的大小

G周邊接口或 Library 及 API 使用說明

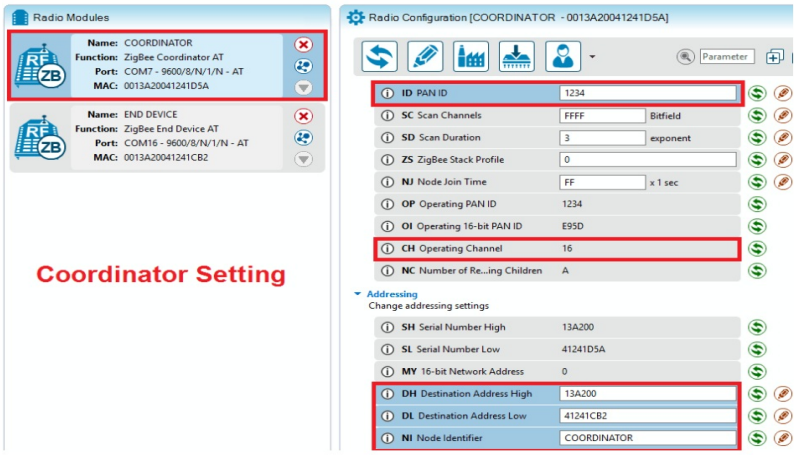
這次使用到比較特殊的硬體有**XBee**跟**LCD**，而這當中的技術比較深入，因此在底下做一個比較詳細的解釋

XBee

以下將分為**事前設定**，**封包處理**，**資料傳送與接收**，**資料使用**，四個部份來講解，並透過這四部分來完成遙控功能:

(1)事前設定: XCTU

為了要兩個XBee互聯，需要先將對方設為傳輸目標，而這部分需要先透過XCTU這個軟體去做設定

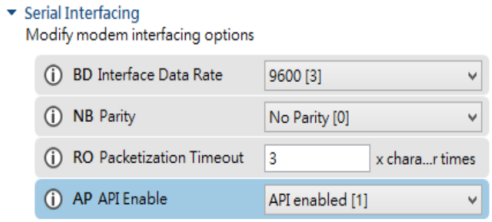


先透過TTL線將XBee連上電腦，然後透過XCTU手動設定

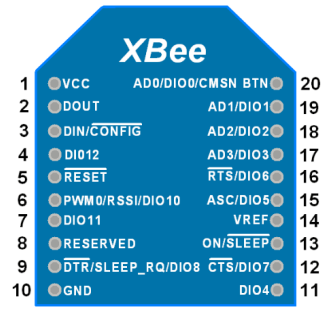
1. **DH**與**DL**欄位(destination)，DH與DL要設定為對方的SH與SL(source)，也就是互相設為對方。
   或是設為Broadcoast:
   DH (Destination High Address) = 0x00000000
   DL (Destination Low Address) = 0x0000FFFF)

2. 其中一個設定為**Coordinator**、另一個設為**Router**

3. 接著是很重要的一步驟，要將**API Enable**打開，這樣才能透過PIC18F4520對封包做處理
    - 當初因為不知道要做這設定，所以卡住好久



4. 將搖桿部分(收集資料)的**IR**設為100ms，也就是資料收集速率一秒10次
5. pin腳設定: 將pin20及pin18指定為ADC，pin11指定為Digital input



(2) 封包處理: 這部分為需要徹底了解，後面才有可能寫得出程式碼

首先要了解XBee封包(Frame)的形式

1. **Start Delimeter**固定為0x7E
2. **Frame Length**我們使用的是2 bytes(長度計算方式為從Frame Type到Checksum總共的byte數)

```
if (action == Write)
{
    /*Length=basic 4 bytes + x bytes*/
    /*Frame Length = Length of (Frame Type + Frame ID + AT Command) in Bytes*/
    if(Command_Value > 0x00FFFFFF) Length = 8;/* Define parameter value depend frame length */
    else if(Command_Value > 0x00FFFF) Length = 7;
    else if(Command_Value > 0x00FF) Length = 6;
    else Length = 5;
}
else
    Length = 4;
```

3. **Frame Type**跟**Frame ID**這部分跟傳輸的功能有關(比如設定XBee用的封包的Frame type為0x08，Frame ID為0x01)
    - 這部分我們把從data sheet查來的資料define在程式最前面，方便之後使用!

```
#define START_DELIMITER                 0x7E
#define AT_COMMAND_FRAME                0x08
#define TRANSMIT_REQUEST_FRAME          0x10
#define REMOTE_AT_COMMAND_FRAME         0x17
#define IO_DATA_SAMPLE_FRAME            0x92
#define AT_COMMAND_RESPONSE_FRAME       0x88
#define REMOTE_AT_COMMAND_RESPONSE_FRAME 0x97
#define RECEIVE_PACKET_FRAME            0x90
#define TRANSMIT_STATUS_FRAME           0x8B

#define FRAME_ID                        0x01
#define REMOTE_AT_COMMAND_OPT           0x02
#define TRANSMIT_REQUEST_OPT            0x00
#define TRANSMIT_REQUEST_BROADCAST_RADIUS 0x00
```

4. 接下來是**Data**的部分，這取決於你要送甚麼資料

5. 最後是**Checksum**，計算方式有點複雜，為從Frame type開始，一個一個byte相加(進位捨去)，然後最後用0xff減掉，結果就是checksum，程式碼裡長這樣:

```
/*reference http://www.electronicwings.com/sensors-modules/xbee-module*/
/*Checksunm=0x08+0x01+data(add every byte)*/    /*see line 17*/
/*FRAME_ID: set to non-zero. You can disable the response by setting the frame ID to 0 in the request.*/
Checksum = AT_COMMAND_FRAME + FRAME_ID + ATCommand[0] + ATCommand[1];
if (action == Write)
Checksum = Checksum + (Command_Value >> 24) + (Command_Value >> 16) + (Command_Value >> 8) + Command_Value ;
Checksum = 0xFF - Checksum;
```

(3)資料傳送與接收: 這部分為XBee程式撰寫最難的部分

資料傳送: 透過USART傳輸

將封包每個byte設定好後，透過USART傳出去: (程式碼片段)

```
USART_TxChar(START_DELIMITER);
USART_TxChar(Length >> 8);  /*le
USART_TxChar(Length);
USART_TxChar(AT_COMMAND_FRAME);
USART_TxChar(FRAME_ID);
USART_SendString(ATCommand);
```

資料接收: 先透過interrupt收到第一個byte後開始處理

範例說明:
7E 00 16 92 00 13 A2 00 41 89 A9 C2 26 2A 01 01 00 10 05 00 10 02 08 02 0B F5

Start delimiter	Digital channel mask
7E	00 10
Length	Analog channel mask
00 16 (22)	05
Frame type	DIO4/AD4 digital value
92 (IO Data Sample RX Indicator)	High
64-bit source address	DIO0/AD0 analog value
00 13 A2 00 41 89 A9 C2	02 08 (520)
16-bit source address	DIO2/AD2 analog value
26 2A	02 0B (523)
Receive options	
01	
Number of samples	
01	

1. 先把收到的第四個byte(Frame Type)取出來，看是哪種資料，然後決定要怎麼parse封包。

```
switch (Frame_Type)
{
    case (IO_DATA_SAMPLE_FRAME):     /* Parse received I/O data sample frame */
```

2. 第四個byte我們發現為IO_DATA_SAMPLE_FRAME，接著對這種type做特殊處理:
   16，17 bytes(00 10)為Digital channel mask，我們用Is_Digital跟Is_Analog去紀錄mask的值，稍後作比對需要用到
   ```
   Is_Digital = ((int)ReceiveBuffer[16]<<8) + ReceiveBuffer[17];
   Is_Analog = ReceiveBuffer[18];
   ```
   - Digital channel mask表示在等等的19，20byte的00 10位置的值表示digital為0或是1
   - 比如19，20byte
     如果是0000 **1**000則為high，
     如果是0000 **0**000則為low
     因為16，17byte的mask為0000 **1**000，所以要去對mask的位置做比對
     ```
     for (uint8_t i = 0; i < DIGITAL_BUFFER_SIZE; i++)
     {
         if(((Is_Digital >> i) & 0x01) == 1 && ((Digital_Value>>i) & 0x01) != 0){
             DigitalData[i] = 1;
         }
         else if(((Is_Digital >> i) & 0x01) == 1 && ((Digital_Value>>i) & 0x01) == 0){
             DigitalData[i] = 0;
         }
         else
             DigitalData[i] = -1;
     }
     ```

   18 byte(05)為Analog channel mask

   - 05=1+4(=2^**0**+ 2^**2**)表示接下來(21開始的byte)會收到AD**0**跟AD**2**的傳來的資料
     ```
     for (uint8_t i = 0, j = 0; i < ANALOG_BUFFER_SIZE; i++)
     {
         if(((Is_Analog >> i) & 0x01) == 1)
         {
             if(Is_Digital != 0) /*if the input has analog and digital, the analog will be 21~22+j*2*/
                 AnalogData[i] = 256 * ReceiveBuffer[21+(j*2)] + ReceiveBuffer[22+(j*2)];    /*y is in [0], x is in [2]*/
             else    /*if there is only analog, it will be 19 and 20*/
                 AnalogData[i] = 256 * ReceiveBuffer[19+(j*2)] + ReceiveBuffer[20+(j*2)];
             j++;    /*how many analog*/
         }
         else
         {
             AnalogData[i] = -1;
         }
     }
     ```

3. 把傳來的資料解析並收集存起來後，接下來進入第4部分，資料處理

(4)資料處理

取得了傳來的Analog及Digital的資料後，開始使用這些資料來達成我們要完成的功能。
而在我們的程式中就是把收集來的Analog來做為麋鹿前後移動的依據。

```
if(AnalogData[0] <=1523 && AnalogData[0]>=900 && AnalogData[2] <=1523 && AnalogData[2]>=900){/*x,y not move*/
    LATDbits.LATD1=0;
    LATCbits.LATC3=0;
}
else if(AnalogData[0] <=650 && AnalogData[0]>=350){/*y down*/
    LATDbits.LATD1=1;
    LATCbits.LATC3=0;
}
else if(AnalogData[0] <=200 && AnalogData[0]>=0 && RC0!=0){/*y up*/
    LATCbits.LATC3=1;
    LATDbits.LATD1=0;
}
```

小結

原本我們做好了digital跟analog的處理，但是最後礙於時間的問題，只有實作analog的部分。

這部分真的很複雜，花了我們很多時間去瞭解，去實作，並且在過程中遇到了各種障礙，還好最後努力將它克服了。
但也因為程式碼部分難以在這裡解釋，所以還請老師及助教直接去看我們的程式碼，配合旁邊的註解使用。

詳細的資料來源: Data Sheet

LCD

INTERFACE PIN ASSIGNMENT

PIN NO.	PIN OUT	DESCRIPTION
1	VSS	Ground
2	VDD	Logic Circuit Power Supply
3	Vo	Power Supply For LCD Panel
4	RS	Data/ Instruction Register Select
5	R/W	Read/ Write Select
6	E	Enable Signal
7	DB0	3-State I/O Data Bus
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	NC	Power Supply for Backlight. 100V/400Hz AC for EL, 4.2V or 120~180mA DC for LED backlight Don't care if no backlight
16	NC	

Commands

Instruction	Instruction Code										Description	Execution time(f_{osc} is 270kHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM. and set DDRAM address to "00H" from AC	1.53mS
Return Home	0	0	0	0	0	0	0	0	1	*	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53mS
Entry Mode	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and make shift of entire display enable.	39 μS
Display ON/OFF	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	39 μS
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Set cursor moving and display shift control bit, and the direction, without changing DDRAM data.	39 μS
Function Set	0	0	0	0	1	DL	N	F	*	*	Set interface data length (DL : 4-bit/8-bit), numbers of display line (N : 1-line/2-line), display font type(F : 5 X 8 dots/ 5 X 11 dots)	39 μS
Set CG RAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39 μS
Set DD RAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39 μS
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0 μS
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43 μS
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43 μS

* means don't care

LCD的程式碼參考了LCD的pin腳功能及Commands的使用方法，還有找出每一行DDRAM的address是從哪裡到哪裡。

參考資料

1. XBee Module
2. XBee interface with PIC18F4550
3. XBee模組通訊實驗(一)~(五)
4. Data Sheet
5. XBee radio communication between PICs
6. ZIGBEE XBEE S2C–How to configure as Coordinator , Router / End Device
7. XBee X-CTU tutorial

H實際組員之分工項目

黃柏愷:馬達運作、燈條、雷射接收發射、紅外線避障、JoyStick、車體製作
蕭佑永:LCD運作
邵鏡軒:XBee

I遇到的困難及如何解決

車體製作

為了不讓車身太低，我們想要直流減速馬達立著，但立著因為必須用接觸面積小的部分連結車體，而且那面也沒有可以鑽螺絲的部分，便很難連結車的本體，為此我們特別用**3D列印**做出一塊可以包覆馬達，同時完美接和車體的零件。

直流馬達

一開始以為只要像接 LED 接馬達正負在 PIC 上就可以直接運作。但儘管馬達上有電流通過，不知為何就是無法運作。

途中也買了工作電壓比較小的馬達，想驗證看看是不是買的直流減速馬達需要更大的電壓，但依舊沒有用。查了網路發現大家使用直流馬達都會搭配直流馬達的模組

便買到了 L9110S 這個雙通道直流電機驅動模組，把訊號輸入處理一下後馬達也順利動了。幸運的是，L9110S 有兩個訊號輸入，，不用透過額外的類比電路就可以調控電流的方向，一併解決了前進後退的問題。

燈條

為了讓我們的麋鹿變得更炫，所以買了一條燈條想裝在牠身上。買回來才發現燈條的工作電壓需要 12V，然而 PIC 的 5V 遠遠不夠。就去買了升壓模組來使用。

Vin 接麵包板上的正負， Vout 接正負到燈條上。就可以得到升壓過的電壓了。

比較小一點的技巧應該是，要把升壓模組當作 PIC 控制的物件，藉由開關升壓模組，間接達到燈條的開關。

燈條控制

一開始也拿了紅外線接發收器想作為麋鹿燈條**充能**的開關。但我們用紅外線接收器ADC處理後無法收到得結果不如預期。

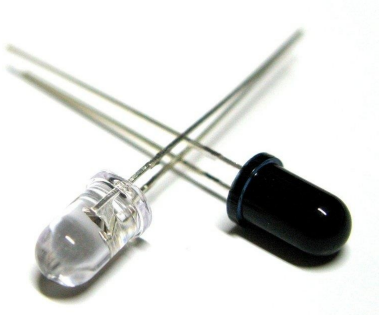
後來有看到另外的雷射接發收，覺得雷射相對於看不到的紅外線，比較潮。而且雷射接收模組會直接處理出 01 數位訊號給我們，不用多做 ADC 處理，便改用雷射接發收模組了。

紅外線避障模組

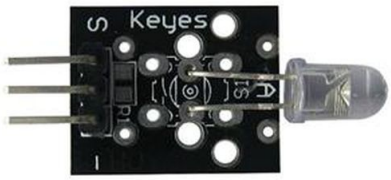
算是比較小一點的問題，當初紅外線避障模組的偵測距離不夠遠，當他收到訊號，使輪子停下來。其慣性還是會讓麋鹿稍微撞到障礙物，藉由慢慢微調紅外線避障模組偵測的距離，讓我們的智慧駕駛能好好運作。

無線操控

- 紅外線操控
  起初有看到網路上有人以紅外線達成無線控制的部分，便去買了紅外線 LED 及接收器。



其運作原理是利用PWM讓LED燈以不同的 duty cycle 發射訊號。而接收器
再判斷不同的訊號 decode 出不同的結果。
但發現紅外線接收器一直收不到訊號，後來才發現紅外線的載波頻率有分
38KHz，32KHz，36KHz，40KHz，56KHz等
不知道自己的接發收訊號是否相符合，保險起見就又去買了一套的模組來試但依舊還是沒反應，便想先試其他方法了



- XBee
  原本以為XBee的使用上會跟藍芽一樣簡單，結果實際操作上才發現有夠困難。
  1. 網路上雖然有不少資源介紹XBee，但是大多數是配合Arduino使用，關於PIC18F4520的介紹少之又少，因此只好自己慢慢摸索。
  2. XBee分為AT mode跟API mode，為了傳遞資料因此選擇使用API mode，因此要先去了解封包(Frame)的形式是甚麼，這部分就花了好多時間去學習。學會之後，透過PIC18F4520去做處理及傳遞，但是不論怎麼用，都無法成功，搞了好幾個小時後才發現原來是API mode沒有把它設為**enable**，難怪傳不出去。
  3. 一開始的IR設定封包傳遞也一直遇到問題:**傳不過去或是接收不到**。這部分我上網查了好多好多資料，結果最後發現是XBee本身好像不太穩定，傳好幾次才有一次可以傳成功。因此後來IR設定我就放棄使用PIC18F4520去做控制設定，而是改用直接使用XCTU去做設定。
  4. 配合Joystick使用時，不知道是Joystick還是XBee的問題，**傳輸資料也是有時正常有時不正常**。比如Joystick往下跟往左推的時候沒有反應，換了一個Joystick以後也是一樣(這部分用XCTU直接去看傳遞的值為多少)，所以後來把Joystick往前推設為前進，往右推設為後退。

PIC18F4520

我們把程式燒錄到PIC18F4520之後反應常常慢半拍，比如我們要麋鹿往前進，有時隔了一兩秒才有反應，或甚至沒有反應，這部份我們無法解決，因為明明接電腦使用Debug模式都還好好的。
(因此當時實體Demo及影片拍攝反應會有點慢)

LCD


是使用RG1602A的16x2LCD，當初遇到的困難是顯示的code已經寫好了，但是不管怎麼試就是無法顯示，code也有跟網路上的比對過確定是正確的。因為沒有在製作期間讓它顯示，所以沒有放進最後的完成品裡面。不過最後試出來的時候是在多按了Make and Program Device Main Project這個動作之後就正常顯示，不曉得是不是因為之前都直接按Debug Main Project所以沒有好好的燒進去的原因?(因為之前寫的都是可以直接按Debug Main Project就好，所以沒有如期想到這塊)

其他

原本想用PWM去控制伺服馬達當作尾巴的使用，但是這部分遇到了些問題，為了把其他功能做好，因此捨棄了這部分(相關code還留著程式碼裡，之後有機會可以再做修改)

J預期效益與結語

能獲得多少的效益實際上是難以量化的，他不會是麋鹿們所銷售的數量以及營收。

要賺大錢，只要用精美的外觀包裝它，不用甚麼昂貴的零件或者艱澀的技術。再搭配上些許的行銷手法，獲得一定的金錢收益並不是甚麼難事。

但打從一開始我們的討論結果就決定**注重技術**，以及其所能帶來的**影響力**，而不是華而不實的外觀或收益。

那預期效益會是他能帶來多少的社群聲量或引起多大的旋風嗎?

好像也不夠準確。

在台灣，有些特別的現象。

厚奶茶、胖老爹之前造成多可怕的喪屍人潮，現在卻唾手可得。

選舉前，公投議案在社群軟體上吵得沸沸揚揚，如今卻消聲匿跡

如果麋鹿變成下一個胖老爹或公投議案，那牠也只成功了一半而已。

我們的鹿不是幾疊鈔票，不是幾百次鍵盤的點擊，而是我們的祈願奔馳的道路，我們要登陸桃花源。一個人與生物和諧生存的仙境。

牠不一定要病毒式的散播於我們的生活中，只要牠的靈魂能照亮人們的心靈，鼓起某種群眾意識，讓他們記得這塊土地上，還有其他的同居人要給予尊重。那就足夠了。

況且，牠真的很**可愛**。