

臺北市立大安高級工業職業學校

電子科

專題報告

16*16 彩色貪吃蛇遊戲機

16*16 COLOR GREEDY SNAKE

GAME CONSOLE

學生 組長：吳晉緯

組員：李汪

組員：周禕

組員：林哲宇

指導老師：黃建中老師

中華民國111年1月

摘要

近年來科技進步，遊戲產業也突飛猛進，古早的遊戲機已經慢慢被手遊給取代，一些小時候經典的遊戲也被大家給遺忘了，所以，我們打算製作 16*16 的彩色貪吃蛇遊戲機來喚起你們對於這隻愛吃蘋果的蛇的記憶！加上為了自我檢視這兩年來的實習成果，並以專題實驗的方式，去探索究竟這些技能是否與我們現今的生活息息相關。並且以小組一起分工合作的方式，訓練人與人之間的溝通、意見分歧時如何擬定解決辦法等等。

關鍵字:貪食蛇，16*16，小遊戲

Abstract

In recent years, science and technology have changed rapidly, and the game industry has also been advancing by leaps and bounds. The ancient game consoles have been gradually replaced by the mobile game market. The classic and extremely interesting games are slowly being forgotten in the minds of the world. Therefore, we intend to Make a 16*16 snake game machine to evoke your memories of this little snake! In addition, in order to self-examine the results of the internship in the past two years, and use thematic experiments to explore whether these skills are closely related to our current lives. And in the way of group work division and cooperation, training people to communicate, how to formulate solutions when disagreements, etc.... It may be easier to do it alone because you don' t have to take care of others, but this experience and memory of working hard with classmates and friends towards the same goal will surely become an irreplaceable treasure.

目錄

摘要.....	1
Abstract.....	2
目錄.....	3
圖目錄.....	5
表目錄.....	6
第 1 章 前言.....	7
1-1 專題製作背景及目的	7
1-2 預期成果	7
1-3 組員貢獻度	8
第 2 章 理論探討.....	9
2-1 電路設計	9
2-1-1 Altium Designer	9
2-1-2 RGB 點矩陣.....	11
2-1-3 金屬氧化物半導體場效電晶體.....	12
2-1-4 74138.....	13
2-2 軟體設計	15
2-2-1 Arduino IDE	15
2-2-2 Arduino Mega2560.....	16
2-2-3 Visual Studio Code.....	17
2-3 外觀設計	18
2-3-1 Laserbox.....	18
2-3-2 金屬按鈕開關(push-buttonswitch).....	19
2-3-3 液晶顯示器(liquid-crystal display , LCD)	20
2-3-4 類比搖桿	21
2-4 電源	22
第 3 章 專題準備.....	23
3-1 系統架構	23
3-2 流程圖	24
3-3 甘特圖	25

第 4 章 專題成果	26
4-1 問題與解決	26
4-2 成果	28
第 5 章 結論與建議	45
5-1 結論	45
5-2 建議	45
參考文獻	46
附錄	47
成員簡歷	49

圖目錄

圖 1 貢獻度圓餅圖.....	8
圖 2 Altium Designer.....	9
圖 3 電路圖.....	10
圖 4 RGB 點矩陣.....	11
圖 5 NMOSFET SS23K7002BFU.....	12
圖 6 74LS138.....	13
圖 7 PMOSFET Si2301DS.....	14
圖 8 Arduino IDE.....	15
圖 9 Arduino Mega2560.....	16
圖 10 Visual Studio Code.....	17
圖 11 Laserbox.....	18
圖 12 金屬按鈕開關.....	19
圖 13 液晶顯示器.....	20
圖 14 類比搖桿.....	21
圖 15 18650 電池.....	22
圖 16 系統架構圖.....	23
圖 17 流程圖.....	24
圖 18 外觀成品.....	43
圖 19 單人模式示意圖.....	43
圖 20 雙人對戰示意圖.....	44

表目錄

表 1 設備清單.....	47
表 2 材料清單.....	48
表 3 成員簡歷.....	49

第 1 章 前言

1-1 專題製作背景及目的

貪吃蛇是以前經典的手機遊戲，是跨時代的經典，而我們現在就有機會可以將它用 16 x 16 RGB LED 點矩陣的硬件表現出來，也是別有一番風味的。

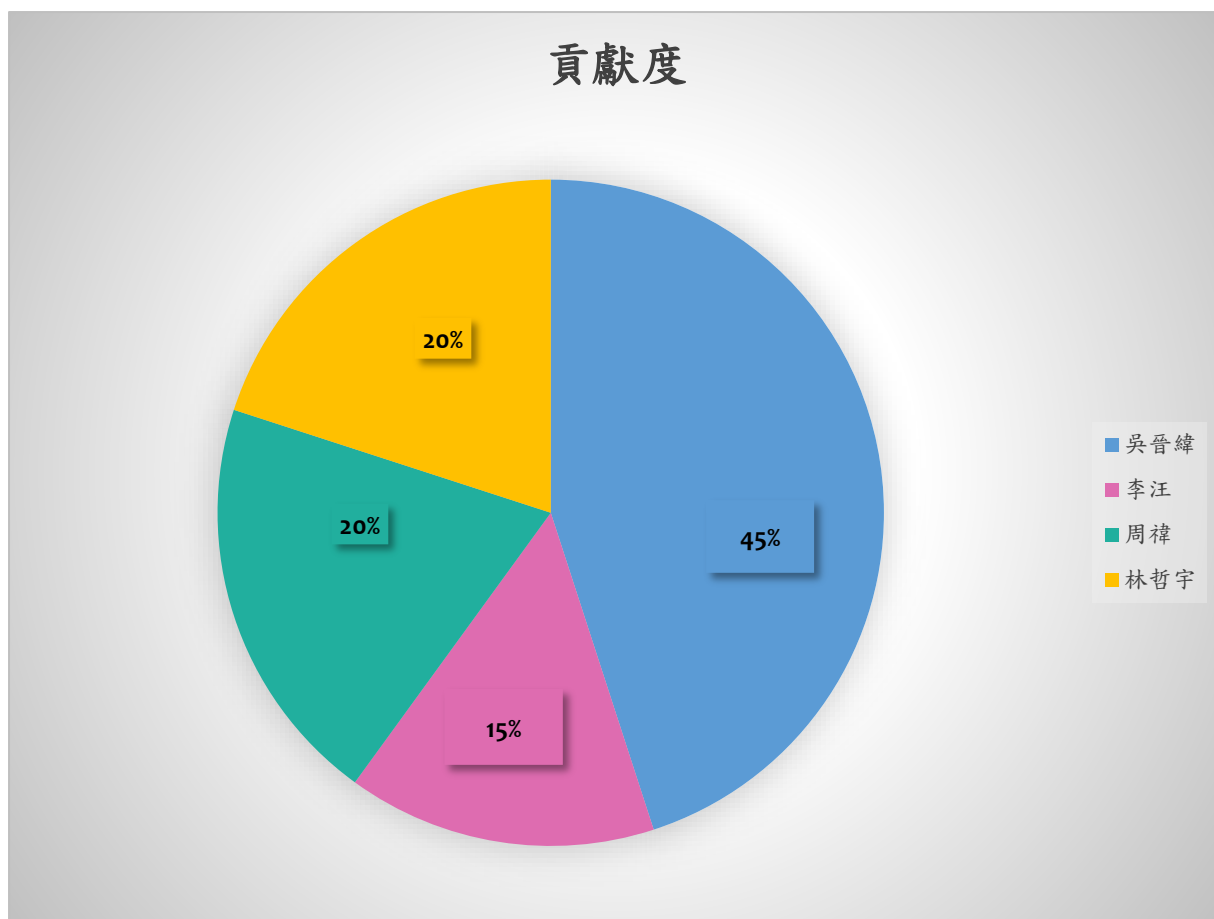
專題的題目我們選擇貪吃蛇，是因為難度適中，非常適合當作專題，且這東西具有基本的娛樂性，可以無聊的時候用來打發時間，可謂是一舉兩得。

1-2 預期成果

1. 可以供兩個同學遊玩貪吃蛇，且可以自己挑選蛇的顏色，可以互相對戰
2. 可以顯示 7 種顏色的像素
3. 液晶顯示器可以顯示分數或其它資訊
4. 外觀精美
5. 用兩顆鋰電池就可以正常使用
6. 蜂鳴器可以發出音效

1-3 組員貢獻度

圖 1 貢獻度圓餅圖



第 2 章 理論探討

2-1 電路設計

這裡是介紹我們運用的材料

2-1-1 Altium Designer

我們的電路圖是用 Altium Designer 如來進行設計。

圖 2 Altium Designer

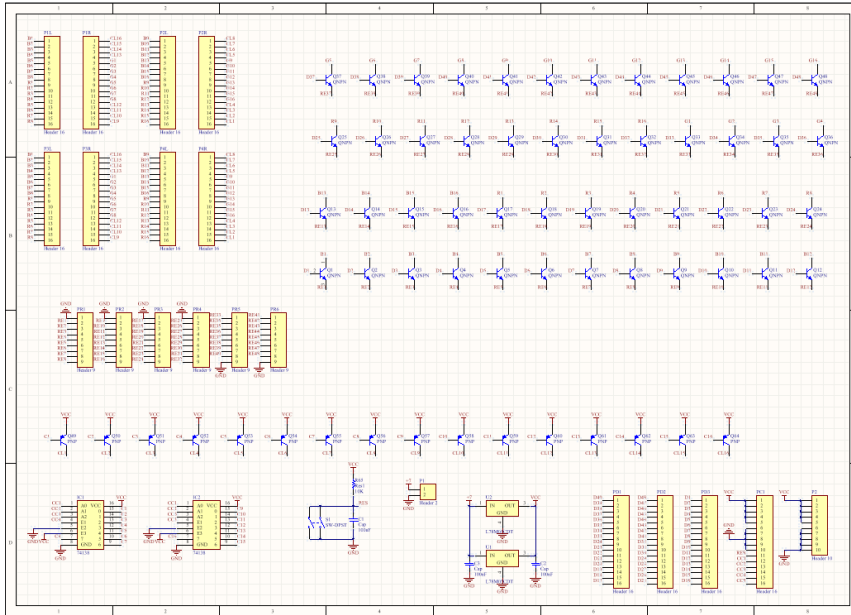
(圖片來源: https://en.wikipedia.org/wiki/Altium_Designer)



Altium designer 是 altium 公司開發的一款電子設計自動化軟體，用於原理圖、PCB、FPGA 設計。結合了板級設計與 FPGA 設計。

我們的主要電路圖如下圖 3:

圖 3 電路圖



2-1-2 RGB 點矩陣

我們做的彩色貪吃蛇遊戲機是運用 RGB 點矩陣來進行顯示。

點陣顯示器 (英語：Dot-matrix display) 是由可亮可暗的許多小單元 (燈或其它結構，只要在色彩上有所區別即可) 排成陣列 (一般為矩形，其它形狀也有但並不常見) 來顯示文字或圖形內容的顯示裝置。它可以用來顯示機器執行狀態、時鐘、鐵路發車指示、公路行車資訊等許多無需高解析度的簡單資訊。

圖 4 RGB 點矩陣

(圖片來源: <https://www.ruten.com.tw/item/show?21607801650129>)




2-1-3 金屬氧化物半導體場效電晶體

(簡稱：金氧半場效電晶體；英語：Metal-Oxide-Semiconductor Field-Effect Transistor，縮寫：MOSFET)，是一種可以廣泛使用在類比電路與數位電路的場效電晶體。金屬氧化物半導體場效電晶體依照其通道極性的不同，可分為電子占多數的N通道型與電洞占多數的P通道型，通常被稱為N型金氧半場效電晶體 (NMOSFET) 與P型金氧半場效電晶體 (PMOSFET)。

我們運用的 NMOSFET 如(圖 5)

圖 5 NMOSFET SS23K7002BFU


(圖片來源: <https://www.ruten.com.tw/item/show?21911190529741>)



www.vishay.com

2N7002K
Vishay Siliconix

N-Channel 60 V (D-S) MOSFET



Top View

Marking code: 7K

PRODUCT SUMMARY	
V _{DS} (V)	60
P _{DS(on)} max. (W) at V _{GS} = 10 V	2
Q _g typ. (nC)	0.4
I _D (mA)	300
Configuration	Single

FEATURES

- Low on-resistance: 2 Ω
- Low threshold: 2 V (typ.)
- Low input capacitance: 25 pF
- Fast switching speed: 25 ns
- Low input and output leakage
- TrenchFET® power MOSFET
- 2000 V ESD protection
- Material categorization: for definitions of compliance please see www.vishay.com/doc?99912

Note

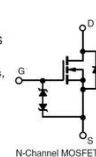
This datasheet provides information about parts that are RoHS-compliant and / or parts that are non RoHS-compliant. For example, parts with lead (Pb) terminations are not RoHS-compliant. Please see the information / tables in this datasheet for details.

BENEFITS

- Low offset voltage
- Low voltage operation
- Easily driven without buffer
- High speed circuits
- Low error voltage

APPLICATIONS

- Direct logic-level interface: TTL/CMOS
- Drivers: relays, solenoids, lamps, hammers, display, memories, transistors, etc.
- Battery operated systems
- Solid state relays



N-Channel MOSFET

2-1-4 74138

74138，是一個 3 到 8 的解碼器，分為 74HC138(cmos)和 74LS138(TTL)兩種，下圖是其邏輯符號及管腳排布，下表中列出了該器件的邏輯功能，從表中可以看出其輸出為低電平有效，使能端 G1 為高電平有效，G'G'為低電平有效，當其中一個為高電平，輸出端全部為 1！在中規模積體電路中解碼器有幾種型號，使用最廣泛！

我們這個電路是運用 74LS138 如(圖 6)

圖 6 74LS138

(圖片來源: <https://html.alldatasheet.com/html-pdf/12624/ONSEMI/74138/363/2/74138.html>)

SN74LS138

1-of-8 Decoder/ Demultiplexer

The LSTTL / MSI SN74LS138 is a high speed 1-of-8 Decoder / Demultiplexer. This device is ideally suited for high speed bipolar memory chip select address decoding. The multiple input enables allow parallel expansion to a 1-of-24 decoder using just three LS138 devices or to a 1-of-32 decoder using four LS138s and one inverter. The LS138 is fabricated with the Schottky barrier diode process for high speed and is completely compatible with all ON Semiconductor TTL families.

- Demultiplexing Capability
- Multiple Input Enable for Easy Expansion
- Typical Power Dissipation of 32 mW
- Active Low Mutually Exclusive Outputs
- Input Clamp Diodes Limit High Speed Termination Effects

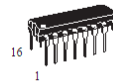
GUARANTEED OPERATING RANGES

Symbol	Parameter	Min	Typ	Max	Unit
V _{CC}	Supply Voltage	4.75	5.0	5.25	V
T _A	Operating Ambient Temperature Range	0	25	70	°C
I _{OH}	Output Current - High			-0.4	mA
I _{OL}	Output Current - Low			8.0	mA



ON Semiconductor
Formerly a Division of Motorola
<http://onsemi.com>

**LOW
POWER
SCHOTTKY**



PLASTIC
N SUFFIX
CASE 648



SOIC
D SUFFIX
CASE 751B

我們運用的 PMOSFET 如(圖 7)

圖 7 PMOSFET Si2301DS

(圖片來源: <http://linhkienviet.vn/si2301ds-p-channel-mosfet-2-3a-20v-sot-23>)

FEATURES


- High dense cell design for extremely low $R_{DS(on)}$
- Rugged and reliable
- Case Material: Molded Plastic.

Absolute Maximum Ratings (TA=25°C, unless otherwise noted)

Parameter	Symbol	Ratings	Units
Drain-Source Voltage	VDS	-20	V
Gate-Source Voltage	VGS	±8	V
Drain Current (Continuous)	ID	-2.8	A
Drain Current (Pulsed) ¹	IDM	-10	A
Total Power Dissipation @TA=25°C	PD	1.25	W
Operating Junction and Storage Temperature Range	Tj, Tstg	-55 to +150	°C
Thermal Resistance Junction to Ambient (PCB mounted) ²	RthJA	100	°C/W

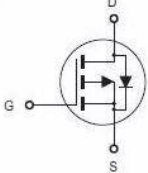
Electrical Characteristics (TA=25°C, unless otherwise noted)

SI2301
P-Channel MOSFET



1.Gate
2.Source
3.Drain

SOT-23



2-2 軟體設計

這裡是介紹我們編寫軟體的程式

2-2-1 Arduino IDE

我們是運用 Arduino IDE 如圖(8)來進行程式的編寫。

圖 8 Arduino IDE

(圖片來源: <https://zh.wikipedia.org/wiki/Arduino>)



Arduino 系列電路板的設計大多使用 Atmel AVR 單片機。這些電路板配有一組數字和類比 I/O 引腳，可以連接各種擴充板或麵包板和其他電路。這些電路板具有串列埠，包括某些型號上的通用串列匯流排（USB），也用於從個人電腦載入程式。

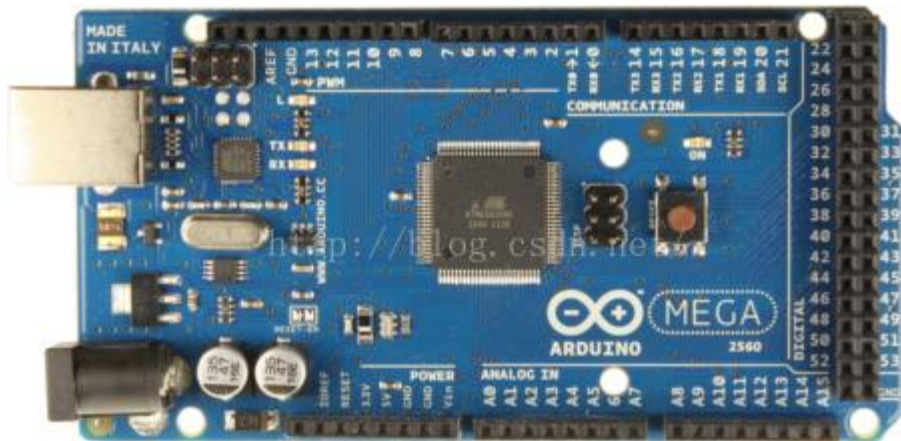
軟體編輯方面，通常使用 C/C++ 程式語言，官方提供了一個 Arduino IDE 用開發。除了使用傳統的編譯工具鏈之外，Arduino 專案還提供了一個基於 Processing 語言專案的整合式開發環境。另外，一些少兒編程教育軟體提供了對 Arduino 的視覺化編程。

2-2-2 Arduino Mega2560

Arduino Mega 2560 如(圖 9)是一款基於 ATmega2560(datasheet)的微控制器板。它有 54 個數位輸入/輸出引腳（其中 15 個可用作 PWM 輸出）、16 個模擬輸入、4 個 UART（硬體序列埠）、1 個 16 MHz 晶體震盪器、1 個 USB 連接、1 個電源插座、1 個 ICSP 頭和 1 個重置按鈕。

圖 9 Arduino Mega2560

(圖片來源: <https://www.itread01.com/content/1549944019.html>)



2-2-3 Visual Studio Code

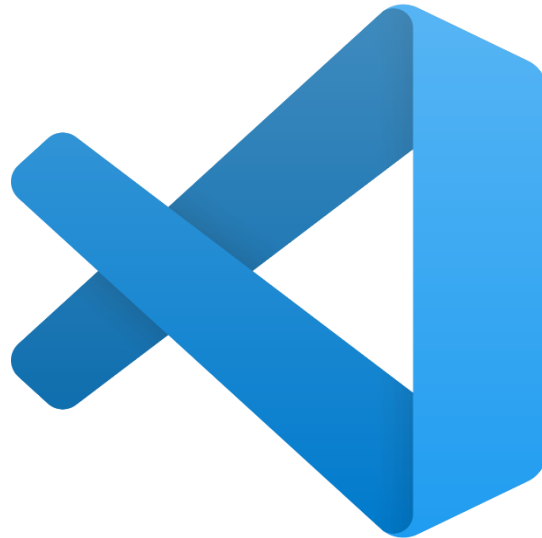
Visual Studio Code 如(圖 10)是我們使用的軟體編輯器

Visual Studio Code (簡稱 VS Code) 是一款由微軟開發且跨平台的免費原始碼編輯器。該軟體支援語法突顯、程式碼自動補全 (又稱 IntelliSense)、程式碼重構功能，並且內建了命令列工具和 Git 版本控制系統。使用者可以更改佈景主題和鍵盤捷徑實現個人化設定，也可以透過內建的擴充元件程式商店安裝擴充元件以加強軟體功能。

VS Code 使用 Monaco Editor 作為其底層的程式碼編輯器。

圖 10 Visual Studio Code

(圖片來源:https://zh.wikipedia.org/wiki/Visual_Studio_Code)



2-3 外觀設計

2-3-1 Laserbox

我們外觀的材料是採用木材,並且運用 Laserbox 如(圖 11)這款軟體來設計模樣,進行雷射切割。

圖 11 Laserbox

(圖片來源: <http://www.kxdw.com/soft/27228.html>)



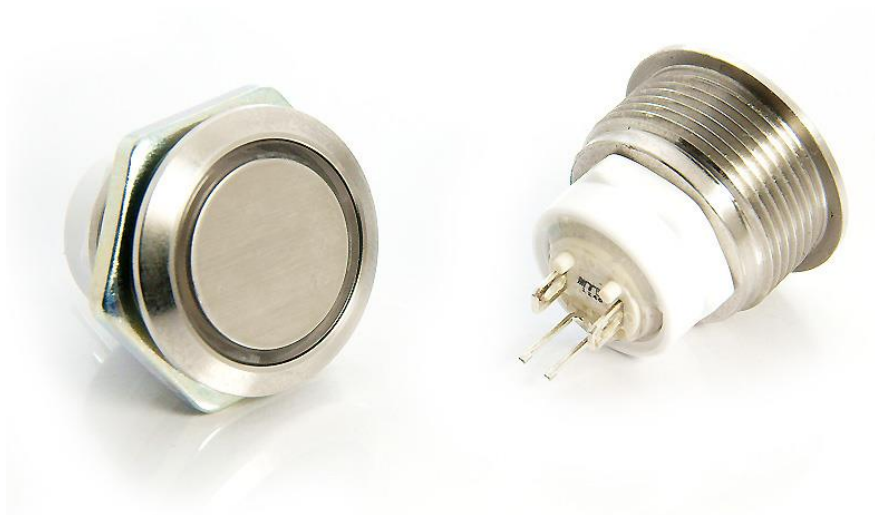
2-3-2 金屬按鈕開關(push-buttonswitch)

我們是運用金屬按鈕開關如(圖 12)來 reset 跟開關機

金屬按鈕開關(push-buttonswitch)是指利用按鈕推動傳動機構，使動觸點與靜觸點按通或斷開並實現電路換接的開關。金屬按鈕開關是一種結構簡單，應用十分廣泛的主令電器。在電氣自動控制電路中，用於手動發出控制信號以控制接觸器、繼電器、電磁起動器等。

圖 12 金屬按鈕開關

(圖片來源: <https://www.itweba.com/zh-TW/catalog/Records-16-Metal.html>)



2-3-3 液晶顯示器(liquid-crystal display , LCD)

我們用液晶顯示器如(圖 13)來顯示分數與功能列表

液晶顯示器為平面薄型的顯示裝置，由一定數量的彩色或黑白畫素組成，放置於光源或者反射環境光源。

圖 13 液晶顯示器

(圖片來源: <https://www.raystar-optronics.com/cht/graphic-lcd-display-module/lcd-192x32.html>)



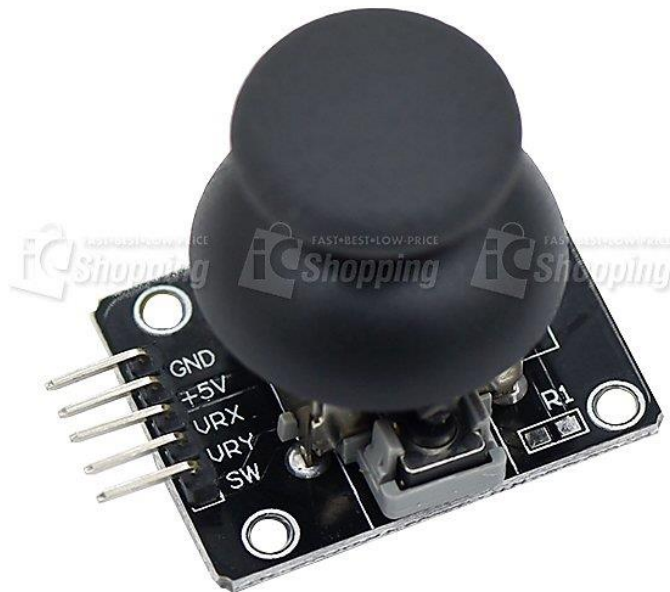
2-3-4 類比搖桿

我們運用類比搖桿來控制貪吃蛇跟操控面板

類比搖桿如(圖 14)，又稱模擬搖桿，是遊戲手把上用於控制方向的零件，通常用於控制遊戲中的人物角色移動方向，其作用等同於控制桿。它突出於遊戲手把之上，通過與固定中心點的偏離位置確定輸入方向。

圖 14 類比搖桿

(圖片來源: <https://www.ruten.com.tw/item/show?21909981625934>)



2-4 電源

我們的電源供應方式是由 18650 鋰電池如(圖 15)兩顆串聯組成。
18650 鋰電池，標準電壓都是 3.6V 或者 3.7V，充滿電的時候是 4.2V。
18650 的意思是，直徑 18 毫米，長 65 毫米

圖 15 18650 電池

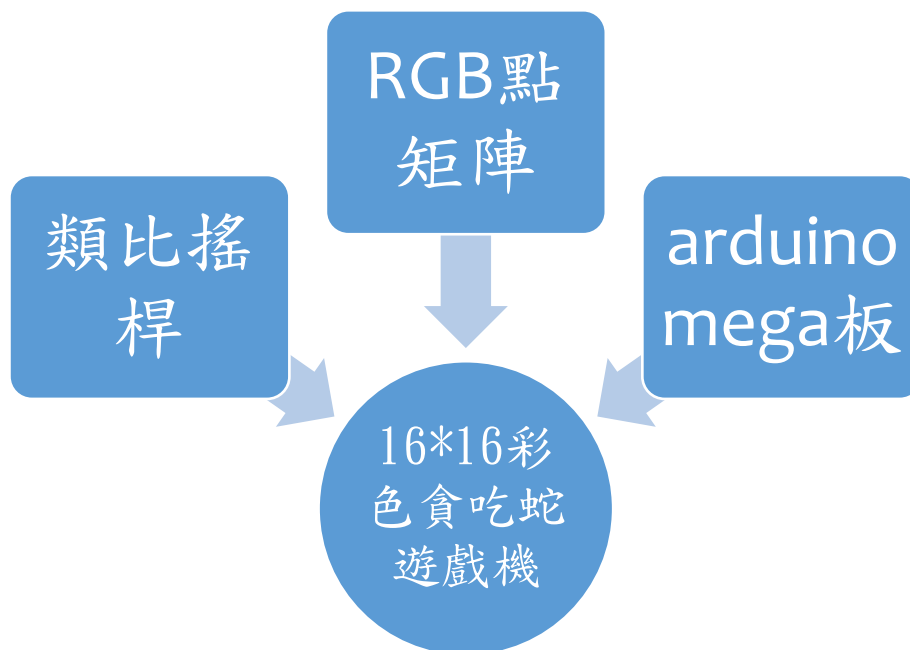
(圖片來源: <https://kknews.cc/digital/av6gko6.html>)



第 3 章 專題準備

3-1 系統架構

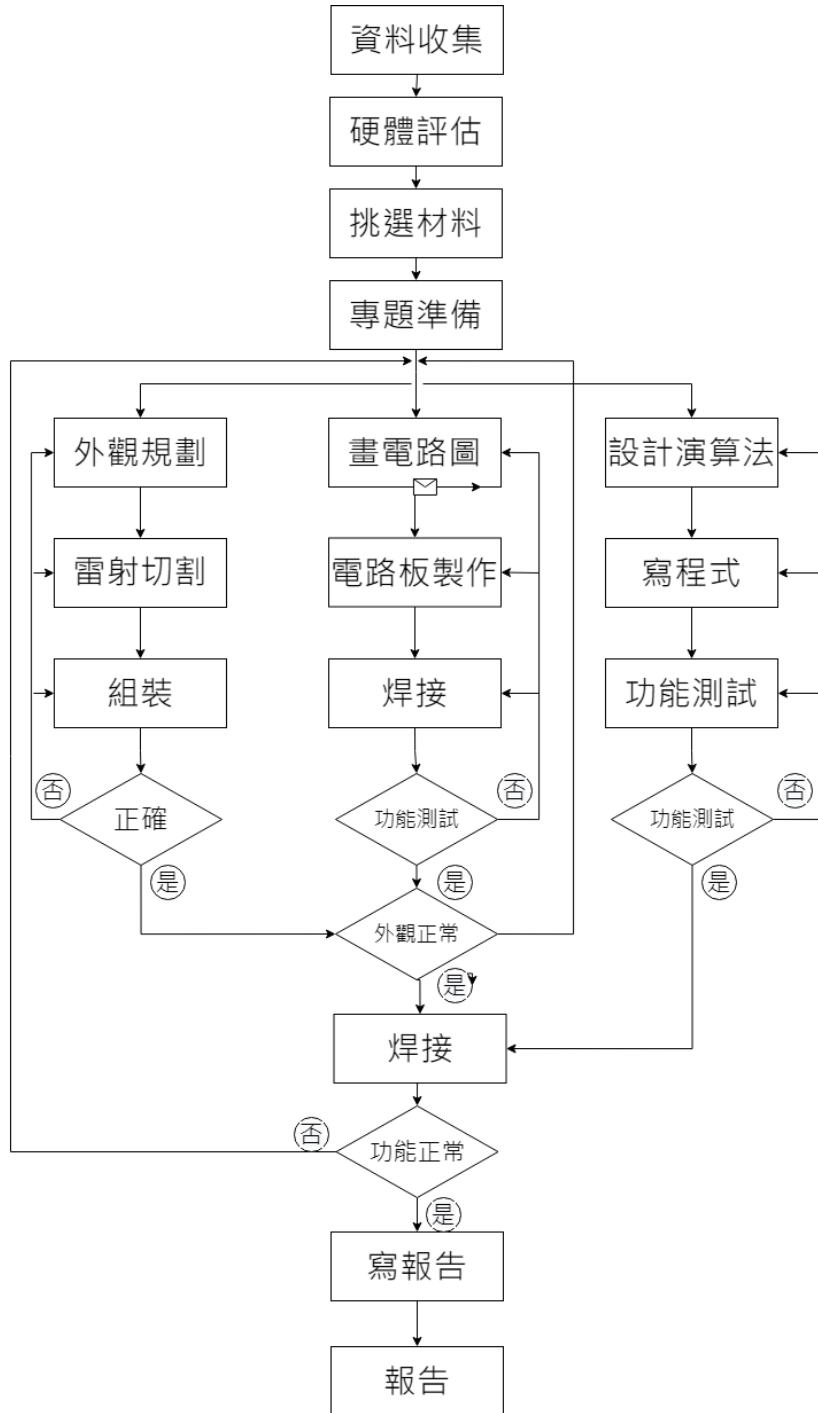
圖 16 系統架構圖



3-2 流程圖

(一) 專題製作步驟

圖 17 流程圖



3-3 甘特圖

工作項目	週次 (日期)																		負責成員
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
資料收集	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	全員
硬體評估	■																		吳晉緯
挑選材料				■															李汪、林哲宇
專題準備	■	■	■	■															全員
畫電路圖					■	■	■	■	■	■	■	■	■	■	■				林哲宇
設計演算法					■	■	■	■	■	■	■	■	■	■	■				吳晉緯
電路板製作									■	■	■	■	■	■	■				林哲宇
外觀規劃									■	■	■	■	■	■	■				周禕
雷射切割										■	■	■	■	■	■				周禕
寫程式								■	■	■	■	■	■	■	■				吳晉緯
焊接										■	■	■	■	■	■				李汪
組裝											■	■	■	■	■				周禕
功能測試(硬體、外觀)												■	■	■	■				吳晉緯、李汪
整合測試													■	■	■				全員
寫報告																■	■		全員
報告																		■	全員
預定進度	5	5	5	10	10	30	30	30	50	55	55	70	80	80	80	90	90	100	累積百分比%

第 4 章 專題成果

4-1 問題與解決

遭遇的問題

- ▶ 1.如果沒有 SMD 的元件，電路板會有很多鑽孔，不易佈線
- ▶ 2.altium designer 內建的零件庫沒有我們要用的原件
- ▶ 3.零件庫中元件的腳位和實際的不同
- ▶ 4.元件擺放的位置
- ▶ 5.電阻太多太佔空間
- ▶ 6.用來當電源的行動電源電壓高於電路設計的 5V
- ▶ 7.電路最大電流有可能會超過穩壓 IC 的最大電流
- ▶ 8.電路板顯像不大成功
- ▶ 9.點矩陣亮度不夠
- ▶ 10.掃描速度不夠
- ▶ 11.點矩陣掃描有餘光

解決方式

- ▶ 1.MOSFET 全採用 SMD 元件
- ▶ 2.找內建零件庫中較相似的零件
- ▶ 3.在零件屬性中手動調正腳位
- ▶ 4.一個一個慢慢試，找到最好的排法
- ▶ 5.把上拉和下拉電阻拆掉只留限流電阻
- ▶ 6.電路多加穩壓 IC
- ▶ 7.把兩個穩壓 IC 並聯
- ▶ 8.換另外一台正常的曝光機

- ▶ 9.更改程式寫法
- ▶ 10.更改程式寫法
- ▶ 11.目前無法解決

4-2 成果

遊戲機的程式碼

矩陣掃描

```
ISR(TIMER1_OVF_vect)
{
    TCNT1 = -5; //5 個 16K 後產生中斷
    static byte scan_bcd = 0; //設定掃描計數
    scan_bcd = (scan_bcd + 1) % 16; //掃描計數+1(數值為 1~16)
    Binary(scan_bcd); //轉二進位並存值

    clear_led(); //掃描及資料線腳位暫存器紀錄歸零

    byte S_LED = 0; //暫存像素資料

    for (byte ii_data = 0; ii_data < 16; ii_data++) //掃描 16 顆 LED(每顆
    包括紅綠藍)
    {
        if (scan_bcd < 8) //如果目前掃描的是前 8 條(掃描上面兩個點矩陣)
        {
            if (ii_data < 8) //前 8 顆 LED 資料的取得方法
                S_LED = Matrix[7 - ii_data][scan_bcd];
            else //後 8 顆 LED 資料的取得方法
                S_LED = Matrix[15 - ii_data][scan_bcd + 8];
        }
        else //如果目前掃描的是後 8 條(掃描下面兩個點矩陣)
        {
            if (ii_data < 8) //前 8 顆 LED 資料的取得方法
                S_LED = Matrix[15 - ii_data][scan_bcd - 8];
            else //後 8 顆 LED 資料的取得方法
                S_LED = Matrix[23 - ii_data][scan_bcd];
        }

        if (S_LED % 2 == 1) //LSB = 1
            RED_DigitalWrite(ii_data); //儲存紅色
        S_LED /= 2;
        if (S_LED % 2 == 1)
```

```

        GREEN_DigitalWrite(ii_data); //儲存綠色
    S_LED /= 2;
    if (S_LED % 2 == 1) //MSB = 1
        BLUE_DigitalWrite(ii_data); //儲存藍色
}

PORTJ |= 0xFF; //掃描線關閉

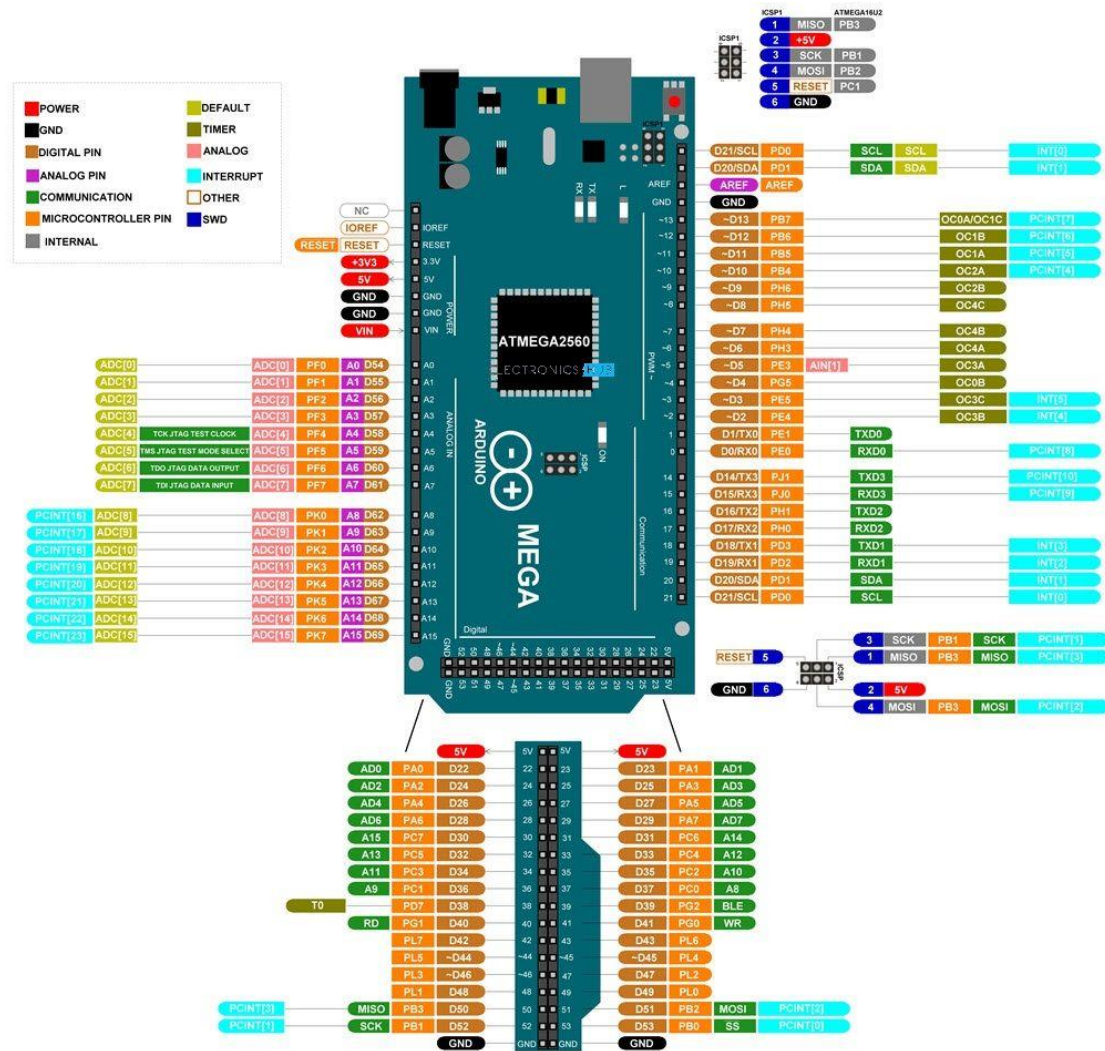
equal(); //資料線送出

for (byte cc = 0; cc < 3; cc++) //送出掃描線(CBA)
{
    if (scan_binary_array[cc + 1] == 1) //如果內容是 1 就送出
        PORTE |= _BV(scan_pin[cc]);
}

if (scan_binary_array[0] == 0) //送出掃描線(選擇 7138 解碼器，分別控制
上下兩顆矩陣)
    PORTJ &= ~0x02; //選擇上面的矩陣
else
    PORTJ &= ~0x01; //選擇下面的矩陣
}

void RED_DigitalWrite(byte pin) //紅色資料線的紀錄
{
    if (pin < 8) //前八顆紅色紀錄
    {
        PORT_A |= _BV(pin);
    }
    else //後八顆紅色紀錄
    {
        PORT_C |= _BV(pin - 8);
    }
}
}

```



```
initMatrix_Red(22, 23, 24, 25, 26, 27, 28, 29, 37, 36, 35, 34, 33, 32,
31, 30); //設定紅色 pinout 腳位
```

貪食蛇物件

```
class snake //宣告貪食蛇類別
{
private:
    byte xy = 0, dir = 1, LinkedList_dir = 1; //上 1, 搖桿現在狀態, 上一次狀態, LinkedList 上一次狀態
    long setX = 0, setY = 0; //搖桿未移動準位
    byte analog_pin[3] = {0, 0, 0}; //貪食蛇類比控制腳(按鈕、X 值、Y 值)
    byte color = 1; //貪食蛇顏色
public:
```

```

int speed = 300;
    //貪食蛇移動速度
LinkedList<byte> snakeList = LinkedList<byte>();
    //https://github.com/ivanseidel/LinkedList
snake(byte a, byte b, byte c, byte d, byte e, byte f, byte g, byte
h, byte i); //建立貪食蛇物件和設定前三點的初始位置及搖桿控制腳位
~snake();
    //遊戲結束時刪除貪食蛇物件
void Start_Matrix();
    //初始蛇的身體
void joystick();
    //搖桿每次紀錄
void linkedlist_add();
    //依照搖桿紀錄移動
void snake_main(byte first);
    //貪食蛇每次移動主程式
byte kill_snake(snake *all, byte first);
    //判斷蛇是否死掉並回傳數值
void snake::lcdPrint_and_endset(byte first, byte kill_number, snake
*Snake); //遊戲結束時 LCD 提示字元及設定結束時要做的事
void apple(snake *all);
    //生成蘋果程式
void cheak_hit_apple();
    //碰到蘋果程式
void set_joystick();
    //初始設定搖桿準位
void setcolor();
    //設定蛇的顏色
void againcolor(byte last_color);
    //設定蛇的顏色為上一次遊玩時的顏色
byte outputcolor();
    //輸出現在蛇顏色
void reset();
    //設定蛇的初始方向及遊戲開始設定
};

```

```

snake::snake(byte a, byte b, byte c, byte d, byte e, byte f, byte g, by
te h, byte i) //建構子設定蛇的初始位置及類比輸入腳位

```



```

{
    snakeList.add(a); //貪食蛇第一個像素(X)
    snakeList.add(b); //貪食蛇第一個像素(Y)
    snakeList.add(c); //貪食蛇第二個像素(X)
    snakeList.add(d); //貪食蛇第二個像素(Y)
    snakeList.add(e); //貪食蛇第三個像素(X)
    snakeList.add(f); //貪食蛇第三個像素(Y)
    analog_pin[0] = g; //搖桿按鈕腳
    analog_pin[1] = h; //搖桿類比輸入 X
    analog_pin[2] = i; //搖桿類比輸入 Y
}

```

設定貪食蛇顏色

```

void snake::setcolor() //設定貪食蛇顏色
{
    clear_Matrix(); //清除矩陣資料
    lcd.setCursor(0, 0); //設定游標
    lcd.print("Select color"); //顯示 LCD
    lcd.setCursor(0, 1); //設定游標
    lcd.print("Up = + Down = -"); //顯示 LCD
    while (true)
    {
        if (digitalRead(analog_pin[0]) == 1) //先將按鈕放開
        {
            color = 1; //設定預設顏色
            bool start = 0; //是否選擇為否
            while (true)
            {
                if (analogRead(analog_pin[2]) > 700) //搖桿往下
                {
                    start = 1; //是否選擇
                    lcd.clear(); //清除 LCD
                    if (color > 1) //color 大
                    {
                        color = color - 1; //color 減
                    }
                    Matrix_display_square(color); //顯示預覽
                }
            }
        }
    }
}

```



```

if (play_people == 2)
  //遊玩人數為 2
  {
    snake Snake[2] = {snake(8, 8, 8, 9, 8, 10, 67, 68, 69), snake(4
, 8, 4, 9, 4, 10, 64, 65, 66)}; //建立兩個 snake 的物件，
https://crmne0707.pixnet.net/blog/post/317240238-c%2B%2B-%E9%A1%9E%E5%88%A5-class-%E9%A1%9E%E5%88%A5%E7%89%A9%E4%BB%B6%E9%99%A3%E5%88%97
    access = Snake;
                                //設定物件指標
    if (again == 0)
                                //如果要重新開始
    {
      Snake->setcolor();        //重新選擇蛇的顏色
      (Snake + 1)->setcolor(); //重新選擇蛇的顏色
    }
    else //否則
    {
      Snake->againcolor>Last_snake0_color); //設定貪吃蛇顏色
為上一次遊玩的顏色
      (Snake + 1)->againcolor>Last_snake1_color); //設定貪吃蛇顏色
為上一次遊玩的顏色
    }
    three_two_one();           //倒數 321
    Snake->apple(access);       //創造蘋果
    Snake->reset();             //貪吃蛇數據重置
    (Snake + 1)->reset();      //貪吃蛇數據重置
    Snake->Start_Matrix();      //初始蛇的身體
    (Snake + 1)->Start_Matrix(); //初始蛇的身體
    while (!end)               //如果遊戲還沒結束
    {
      if (millis() - move_J > 1) //1ms 紀錄一次搖桿數據
      {
        move_J = millis(); //設定時間戳記
        Snake[0].joystick(); //紀錄右邊搖桿
        Snake[1].joystick(); //紀錄左邊搖桿
      }
    }
  }

```

```

        if (millis() - delay_time_snake > Snake[0].speed) //貪食蛇 1
移動
        {
            delay_time_snake = millis(); //設定時間戳記
            Snake[0].snake_main(0);      //右邊貪食蛇移動
        }

        if (millis() - delay_time_snake1 > Snake[1].speed) //貪食蛇 2
移動
        {
            delay_time_snake1 = millis(); //設定時間戳記
            Snake[1].snake_main(1);      //左邊貪食蛇移動
        }
    }
}
else if (play_people == 1) //遊玩人數為 1
{
    snake Snake(8, 8, 8, 9, 8, 10, CTRL[0], CTRL[1], CTRL[2]); //建
立一個 snake 物件
    access = &Snake; //設
定物件指標
    if (again == 0) //如
果要重新開始
        Snake.setcolor(); //重
新選擇蛇的顏色
    else
        Snake.againcolor(Last_snake0_color); //設定貪吃
蛇顏色為上一次遊玩的顏色
    three_two_one(); //倒數 321
    lcd.setCursor(0, 1); //設定游標
為第二列
    lcd.print("Record = " + String(Read_EEPROM(), DEC)); //顯示歷史
最高分數
    Snake.apple(access); //創造蘋果
    Snake.reset(); //貪吃蛇數
據重置
    Snake.Start_Matrix(); //初始蛇的
身體

```

```

        while (!end) //如果遊戲
還沒結束
        {
            if (millis() - move_J > 1) //1ms 紀錄一次搖桿數據
            {
                move_J = millis(); //設定時間戳記
                Snake.joystick();
            }
            if (millis() - delay_time_snake > Snake.speed) //貪食蛇移動
            {
                delay_time_snake = millis(); //設定時間戳記
                Snake.snake_main(0);
            }
        }
    }
    end_game(access); //遊戲結束
}

```

貪食蛇的主程式

```

void snake::snake_main(byte first) //蛇每次動作的主程式
{
    linkedlist_add(); //蛇依照搖桿方向移動

    byte kill_number = kill_snake(access, first); //檢查遊戲是否結束
    if (kill_number == 5) //如果回傳值是 5(蛇沒有
撞到任何東西)
    {
        Matrix[snakeList.get(1)][snakeList.get(0)] = color; //讀出來依照
想要顯示的顏色顯示每個像素//
        cheak_hit_apple(); //是否碰到蘋
果
    }
    else
        lcdPrint_and_endset(first, kill_number, access); //如果回傳值是其他
的就停止遊戲
}

```

搖桿紀錄及貪食蛇動作

```

void snake::joystick() //搖桿紀錄
{

```

```

if (digitalRead(analog_pin[0]) == 0) //按下按鈕時加快速度
    speed = 50;
else
    speed = 300; //放開按鈕時速度變慢

if (analogRead(analog_pin[2]) - setY < -300) //搖桿往上時
    dir = xy = 1;
else if (analogRead(analog_pin[2]) - setY > 300) //搖桿往下時
    dir = xy = 3;
else if (analogRead(analog_pin[1]) - setX < -300) //搖桿往右時
    dir = xy = 2;
else if (analogRead(analog_pin[1]) - setX > 300) //搖桿往左時
    dir = xy = 4;
else //搖桿不動
    xy = dir; //設定方向為先前搖桿的移動方向
}

void snake::linkedlist_add()//依照搖桿紀錄的方向移動貪食蛇
{
    switch (xy)
    {
        case 1: //如果搖桿移動方向為上
            if (LinkedList_dir != 3) //如果蛇的移動方向不是下
                LinkedList_dir = xy = 1; //向上移動
            else
                LinkedList_dir = xy = 3; //向下移動
            break;
        case 2: //如果搖桿移動方向為右
            if (LinkedList_dir != 4) //如果蛇的移動方向不是左
                LinkedList_dir = xy = 2; //向右移動
            else
                LinkedList_dir = xy = 4; //向左移動
            break;
        case 3: //如果搖桿移動方向為下
            if (LinkedList_dir != 1) //如果蛇的移動方向不是上
                LinkedList_dir = xy = 3; //向下移動
            else
                LinkedList_dir = xy = 1; //向上移動
    }
}

```

```

        break;
    case 4:                //如果搖桿移動方向為左
        if (LinkedList_dir != 2)    //如果蛇的移動方向不是又
            LinkedList_dir = xy = 4; //向左移動
        else
            LinkedList_dir = xy = 2; //向右移動
        break;
    default:
        break;
}
switch (xy)
{
    case 1:                //向上移動
        snakeList.add(0, snakeList.get(1) - 1); //加入 Y(值為前一元素的 Y
減一)
        snakeList.add(0, snakeList.get(1));    //加入 X(值為前一元素的 X)
        break;
    case 2:                //向右移動
        snakeList.add(0, snakeList.get(1));    //加入 Y(值為前一元素的 Y)
        snakeList.add(0, snakeList.get(1) + 1); //加入 X(值為前一元素的 X
加一)
        break;
    case 3:                //向下移動
        snakeList.add(0, snakeList.get(1) + 1); //加入 Y(值為前一元素的 Y
加一)
        snakeList.add(0, snakeList.get(1));    //加入 X(值為前一元素的 X)
        break;
    case 4:                //向左移動
        snakeList.add(0, snakeList.get(1));    //加入 Y(值為前一元素的 Y)
        snakeList.add(0, snakeList.get(1) - 1); //加入 X(值為前一元素的 X
減一)
        break;
    default:
        break;
}
}
}

```

遊戲結束的判斷條件

```
byte snake::kill_snake(snake *all, byte first) //判斷是否結束遊戲
```

```

{
    for (int ii = 2; ii < snakeList.size(); ii += 2) //判斷是否撞到自己
    (將除了第一個像素點的像素都讀出來)
    {
        if (snakeList.get(0) == snakeList.get(ii)) //判斷X值是否一樣
        {
            if (snakeList.get(1) == snakeList.get(ii + 1)) //判斷Y值是否一樣
            {
                return 2; //如果都一樣就回傳 2
            }
        }
    }
    if (!(snakeList.get(0) >= 0 && snakeList.get(0) < 16 && snakeList.get(1) >= 0 && snakeList.get(1) < 16)) //如果出界
        return 0;
        //就回傳 0
    if (play_people == 2)
        //如果人數為 2
    {
        first = !first;
        //將索引換成別人
        if ((snakeList.get(0) == (all + first) -
        >snakeList.get(0)) && (snakeList.get(1) == (all + first) -
        >snakeList.get(1))) //如果兩個蛇的頭互相碰撞
        {
            return 3; //回傳 3
        }
        for (int ii = 2; ii < (all + first) -
        >snakeList.size(); ii += 2) //將所有別人的像素點讀出來
        {
            if (snakeList.get(0) == (all + first) -
            >snakeList.get(ii)) //如果X值一樣
            {
                if (snakeList.get(1) == (all + first) -
                >snakeList.get(ii + 1)) //如果Y值一樣
                {
                    return 1; //回傳 1
                }
            }
        }
    }
}

```



```

    }
    }
}
return 5;
}

```

依照顏色設定顯示想要顯示的內容

```

void Matrix_display(byte (*matrix)[16]) //將要顯示的內容送到矩陣顯示
{
    if (Matrix_display_color < 8) //如果顏色是符合要求的(8 設定為快速變換顯
    色)
    {
        for (byte ii = 0; ii < 16; ii++) //取出所有像素
        {
            for (byte jj = 0; jj < 16; jj++)
            {
                if (*(matrix + ii) + jj != 0) //如
                果像素裡有資料
                    *(matrix + ii) + jj = Matrix_display_color; //就
                改變顏色
                Matrix[ii][jj] = *(matrix + ii) + jj; //並
                顯示出來
            }
        }
    }
}

```

創造蘋果

```

void snake::apple(snake *all) //創造蘋果
{
    bool repect; //是否和貪食蛇的位置重複到
    if (play_people == 1) //如果遊玩人數是 1
    {
        score += 10; //進行加分
        lcd.setCursor(0, 0); //設定游標位置為(0,0)
        lcd.print("Score = " + String(score, DEC)); //LCD 印出分數
    }
    do
    {

```

```

    repect = 0; //是否位置重複
設定為 0
    apple_x = random(16); //隨機產生
0~15 的數當作 X 值
    apple_y = random(16); //隨機產生
0~15 的數當作 Y 值
    for (byte times = 0; times < play_people; times++) //查看玩遊戲的
人數
    {
        for (int ii = 0; ii < (all + times)-
>snakeList.size(); ii += 2) //將貪食蛇的所有像素的 XY 值讀出來
        {
            if ((all + times)-
>snakeList.get(ii) == apple_x && (all + times)-
>snakeList.get(ii + 1) == apple_y) //如果一樣
            {
                repect = 1; //是否位置重複設定為 1
                break; //直接離開迴圈
            }
        }
    }
} while (repect); //如果重複就重複執行，沒重複就離開迴圈
Matrix[apple_y][apple_x] = 1; //將符合資格的蘋果點亮
}

```

寫入資料到 EEPROM

```

void Write_EEPROM_Score() //將分數寫入 EEPROM 裡
{
    String Score_string = ""; //存放分數字串
    int divide = 999; //判斷位數
    for (byte i = 0; i < 4; i++, divide /= 10) //不足位進行補 0
    {
        if (score > divide) //判斷位數
        {
            for (byte j = 0; j < i; j++) //依照位數在數字前面補零
                Score_string += '0';
            Score_string += String(score, DEC); //最後將分數一起存入
        }
    }
}

```

```

        break; //離開迴圈
    }
}
for (byte i = 0; i < 4; i++) //讀取字串的 4 個字
{
    EEPROM.write(i, Score_string[i]); //分別寫入字
}
}

int Read_EEPROM()
{
    int history = 0; //從 EEPROM 讀出來要儲存的地方
    int num = 1000; //讀出的值要乘上的權位
    for (byte i = 0; i < 4; i++) //讀出 4 個位元
    {
        history += (num * (EEPROM.read(i) - 48)); //將讀出的值做轉換再進
行儲存
        num /= 10; //每做完一次權位除 10
    }
    return history; //回傳儲存的值
}

```

做好的外觀成品如(圖 18)

圖 18 外觀成品



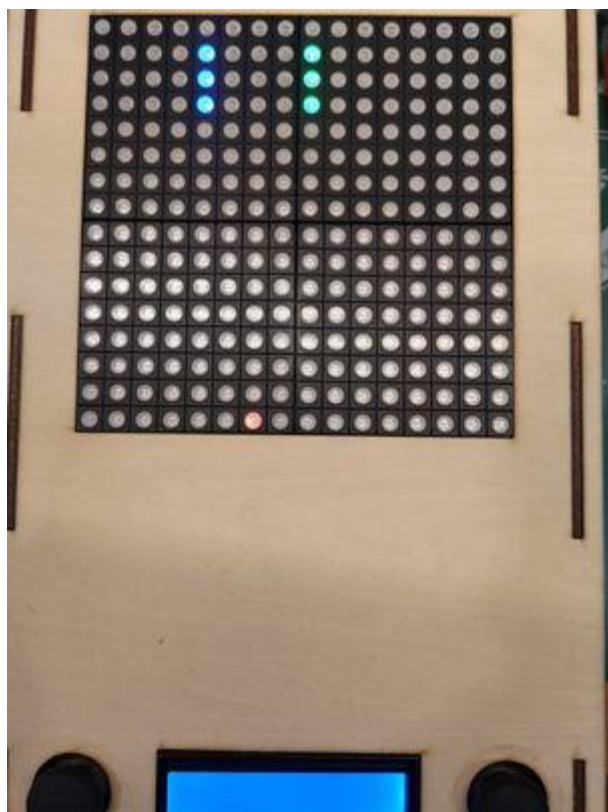
遊戲機單人模式遊玩模樣如(圖 19)

圖 19 單人模式示意圖



遊戲機雙人對戰遊玩模樣如(圖 20)

圖 20 雙人對戰示意圖



第 5 章 結論與建議

5-1 結論

我們的專題是做彩色貪吃蛇遊戲機。預期成果能讓兩個人進行對戰。經過努力後我們確實達成預期的成果了。這個專題的最大難處是程式設計，花費了最多時間，不過讓大家都學到了很多東西，相信這次的實作經驗能讓往後類似的作業如魚得水。

5-2 建議

經過這個專題實習課程，我們要給想做遊戲機當專題的學弟妹一些建議，在設計電路圖時，能用 SMD 儘量用 SMD，比較好焊也比較方便。在洗電路板時曝光機的使用要很小心，不然電路板會容易洗壞。外觀的製作最好使用堅固的材料，因為我們是使用木材，而且是用卡榫來固定，所以在交接處常常斷掉，就又再用一個新的。最後建議大家好好分工合作，不要為了小事起爭執，容易讓進度落。

參考文獻

林家德 109 單晶片微處理機 臺科大圖書股份有限公司

張義和、程兆龍 108 Altium Designer 全華圖書股份有限公司

Ivan Seidel Arduino LinkedList 9/27 <https://github.com/ivanseidel/LinkedList>

<https://crmne0707.pixnet.net/blog/post/317240238-c%2B%2B->

[%E9%A1%9E%E5%88%A5-class-](#)

[%E9%A1%9E%E5%88%A5%E7%89%A9%E4%BB%B6%E9%99%A3%E5%88%97](#)

<https://github.com/ivanseidel/LinkedList>

<https://cms.35g.tw/coding/arduino-%E9%80%B2%E9%9A%8E-digitalwrite->

[%E9%80%9F%E5%BA%A6%E6%9C%89%E9%BB%9E%E6%85%A2%EF%BC%9F/](#)

附錄

表 1 設備清單

類別	設備、軟體名稱	應用說明
硬體	三用電表	電路板檢測
硬體	曝光機	曝光電路板
硬體	蝕刻機	蝕刻電路板
硬體	鑽孔機	電路板鑽孔
硬體	電烙鐵	焊接電路板
硬體	雷射機	雕刻外觀
軟體	Arduino IDE	撰寫 arduino 程式
軟體	Altium Designer	電路板設計
軟體	Powerpoint	製作簡報

表 2 材料清單

類別名稱	材 料 名 稱	單位	數量	應 用 說 明	備註
硬體	Arduino mega2560	個	1	控制遊戲機	
硬體	8X8 RGB 點矩陣	個	4	顯示器	
硬體	搖桿	個	2	遊戲輸入裝置	
硬體	LCD	個	1	顯示分數	
硬體	傳輸線	條	1	連接電腦和 mega2560	
硬體	椴木板	片	3	外觀	
硬體	感光電路板	片	3	遊戲機主機板	
硬體	顯影劑	包	1	洗電路板	
硬體	電阻 220/1K	個	48/16	電路元件	
硬體	開關	個	2	RESET/電源	
硬體	MOSFET P 通道/N 通道	個	16/48	電路元件	
硬體	74138	個	2	掃描線解碼	
硬體	圓孔座	條	4	固定元件	
硬體	穩壓 IC 5V	個	1	穩壓 5V	
硬體	18650	個	2	電源	
硬體	電池盒	個	1	裝電池	

成員簡歷

表 3 成員簡歷

姓 名	吳晉緯	班 級	電子三甲	
曾修習 專業科目	基本電學與實習 電子學與實習 數位邏輯設計與可程式邏輯設計實習 微處理機與實習			
參與專題 工作項目	資料收集 硬體評估 專題準備 設計演算法 寫程式 功能測試 整合測試 寫報告 報告			
經歷簡介	擁有工業電子丙級證照 擔任大安高工電腦研究社 22 屆教學長加硬體教學 二年級下學期參加過校歌比賽 擔任 109 學年度學藝幹事 擔任 110 學年度副班長			
姓 名	李汪	班 級	電子三甲	
曾修習 專業科目	基本電學與實習 電子學與實習 數位邏輯設計與可程式邏輯設計實習 微處理機與實習			

參與專題工作項目	資料收集 專題準備 挑選材料 焊接 整合測試 寫報告 報告		
經歷簡介	當過班級幹部學藝和環保 擁有工業電子丙級證照 二年級下學期參加過校歌比賽		
姓名	周禕	班級	電子三甲
曾修習專業科目	基本電學與實習 電子學與實習 數位邏輯設計與可程式邏輯設計實習 微處理機與實習 整合測試 報告		
參與專題工作項目	資料收集 專題準備 外觀規劃 雷射切割 組裝 寫報告 報告		
經歷簡介	擁有工業電子丙級證照 二年級下學期參加過校歌比賽		

姓名	林哲宇	班 級	電子三甲	
曾修習專業科目	基本電學與實習 電子學與實習 數位邏輯設計與可程式邏輯設計實習 微處理機與實習 整合測試			
參與專題工作項目	資料收集 專題準備 挑選材料 畫電路圖 電路板製作 寫報告			
經歷簡介	擁有工業電子丙級證照 一年級參加過校慶短跑 二年級下學期參加過校歌比賽			

