高雄市高英高級工商職業學校

Kao Ying Industrial Commercial Vocational High School

專題製作報告

8x8x8 3D 光立方

科別班級：資 訊 科三年二班

學生姓名：陳 亦 昀（18）
郭 建 順（37）
楊 莉 彤（39）
王 姿 懿（40）

指導老師：簡 琨 祥 老師

中 華 民 國 104 年 05 月

誌　謝

從國中畢業以後進入高英工商學校開始學習技職課程，面對不同的專業領域，除了苦心竭力，更覺得漫長。如今，我們小組順利的來到了三年級，回頭一看這前面兩年的路上的學習歷程，點點滴滴。我想，是我們該將每個階段的學習，留下紀錄的時候了。

感謝我們的指導老師也是我們的導師簡琨祥老師在這三年的學習生涯中，引導我們往最適合的學習方向來一步步邁進，更感謝老師的努力指導，讓我們可以發現自己長久以來的不足與缺點，學習的過程中可以瞭解、信任與師生之間的情感，如此的刻苦銘心，對我們小組而言，更是在這個階段要好好把握剩下的日子。

感謝班上同窗好友及小組們的互相協助，在電路設計、實作與各種電腦工具應用上真的幫了我們相當多的忙，回想剛入學的我們，對諸多情事都一竅不通，幸有你們大家的打氣及協助，使讓我們彼此有勇氣邁向這未知的學習旅程，而没有半途而廢，真的是非常謝謝你們，有這三年同學情誼的陪伴，令我們小組同學內心的感動真是言語所無法形容。

陳亦昀、郭建順、楊莉彤、王姿懿　謹上　2015/05

摘　要

隨著電子資訊越來越發達，晶片控制也一直都是一門重要的課程，但大家在學習 LED 控制通常都僅限於平面的控制，因此我們決定透過電腦程式去模擬及製作動畫，並且輸出 C 語言燒錄進去 8051，接上一個 LED 正方體，讓大家不再只是使用平面上的 LED 點矩陣，也可以對立體概念有更深一步的想法本作品提出利用多個 2D 平面的 LED 矩陣並以並列方式排列而成之“3D 立體面 LED 顯示器”，能產生立體的視覺顯示效果。每個 2D 平面以 SMD LED 元件以矩陣方式排列結合於透明玻璃上。將數個 2D 平面以並列方式排列完成一個立體面的架構。

由於立體面的每個平面皆是以透明玻璃為底板，固使用者可以透過視覺穿透的效果看到 LED 的點亮與關閉狀態。也不會因為觀賞的角度不同而無法看到顯示器的顯示效果，藉此解決 2D 平面顯示器因觀賞角度不同而無法觀賞顯示效果的問題。

關鍵詞：單晶片、組合語言、光立方

目　錄

表目錄

圖目錄

壹、前言

一、製作動機

在科技的逐漸進步之下，LED 相關應用也慢慢被擴大使用，現在已廣泛運用在手機、汽車、交通號誌、戶外看板及照明設備等，皆隨處可見。而現在也有眾多產品或事物與 3D 結合，像是電影、電視、相機、手機等，雖然現在的 3D 技術產品普遍來說還是需要靠一些東西來輔助，不過相信在過一段時間技術就能達到不需再依靠任何的東西就能自由呈現出 3D 的技術。市面上的廣告，無論是公車上、電視上、跑馬燈等等，都只是平面的，若能做成 3D 的，不僅能吸引住路人的目光，更能利用此種技術，做出許多的效果，若其他公司使用的皆是 2D 平面，而我們的是 3D 立體的，是否就能獨創出自己的風格，而我們就是想透過多顆 LED 產生 3D 的效果，因此本組嘗試研究一個 3D LED CUBE 的作品。

二、製作目的

在我們小組討論之下，而最吸引我們這組的專題題目是 3D LED CUBE，於是想能否製作出跟他們一樣，能充滿自己想像的各式各樣的圖形，因此我們著手研究，看過各式各樣的影片有 8x8x8、16x16x16 甚至看到 64x64x64 的超大型 3D LED CUBE。看過這麼多種影片之後，我們對於一段影片感到好奇，它是利用 8x8x8 的 LED 方塊來做成類似打方塊的小遊戲，再加上用遙控器來控制方向，於是我們也想要嘗試做成像影片中的效果，讓 LED 閃爍方塊不僅能顯示 3D 的效果還能做成小遊戲的顯示器，以往市面上相關的 LED CUBE、跑馬燈產品大都是較有經驗的人在設計及開發，我們認為若是每個人都能在任何時間任何地點，都能用自己的意思去創造及改變整個跑馬燈之類商品的變化，這個將是非常與其他市面上產品不同的地方，且更能親近大眾，讓就算是對於電腦不熟悉的民眾也能親易上手。

三、製作架構

(一)專題製作流程

首先我們小組與指導老師討論專題製作過程，定案之後我們小組開始蒐集資料、整理資料並開始購買所需要的材料，反覆確認無誤後，便在電路模擬器(麵包板)上進行模擬，待測試完成即開始進行焊接工作；在整個專題應用過程中，如發現錯誤，即會與相關教師進行討論，想辦法如何去補救，且了解程式是否能夠運用自如。

(二)專題製作流程圖

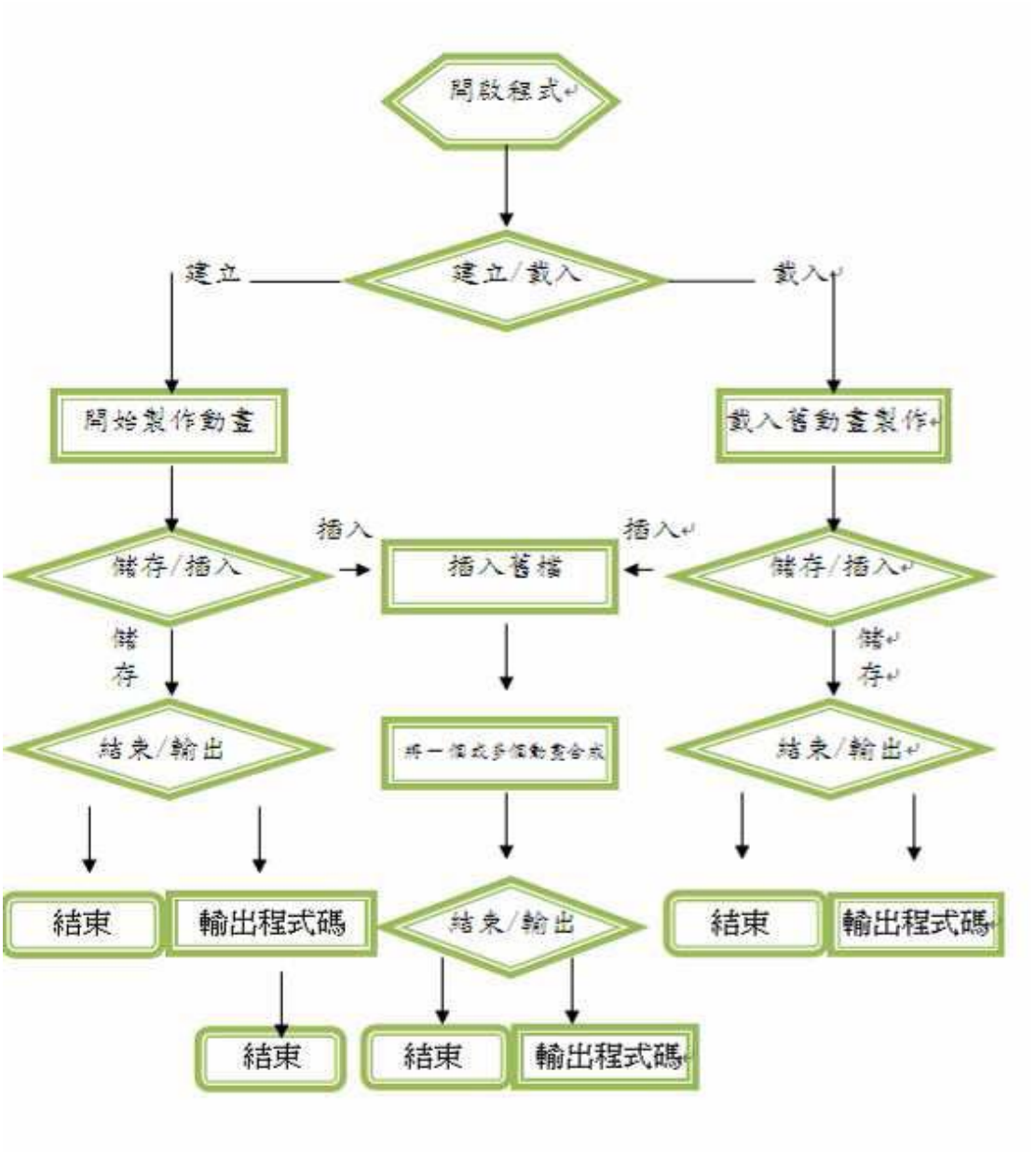


圖 1-3-1 專題製作流程圖

四、製作預期成效

我們小組雖然是第一次製作專題 3D LED CUBE (光立方)，有了指導老師的協助以及同學們的互相協助及辛苦製作的過程，為此，我們小組將專題製作的成效經討論後，定義為：

(一)輸入程式可讓 LED CUBE 亮起

(二)所見即所得的 LED CUBE 程式碼產生器

(三)有 3D 圖可即時旋轉檢視

(四)有影格功能可製作動畫

(五)有匯入功能讓多個動畫合而為一

(六)有儲存功能方便隨時開啟編輯

貳、理論探討

本章將綜覽電子實習及單晶片相關的理論與實務研究，共分為二節來進行相關的理論分析及探討。第一節介紹電子相關零組件的理論與原理；第二節說明單晶片的內部架構、特性、理論基礎及功能，以及組合語言程式設計原則。

一、電子相關零組件

(一)硬體架構

「發光二極體(Light Emitting Diode，簡稱 LED)，當外加順向偏壓時，二極體便會發光；當外加逆向偏壓時，二極體便不會發光。」而製造 LED 的材料，通常為化合物半導體，如砷化鎵(GaAs)、磷砷化鎵(GaAsP)等。其原理是：當 LED 外加順向偏壓時，PN 接面的電子電路產生復合作用，此時會以光的形式釋放能量產生大量的光輻射。我們的 LED 立體方塊是由 8x8 的 64 顆 LED 為單一層，再往上疊接八層而成，共需 512 顆 LED。每顆 LED 的 N 極接腳都焊接在一起先組合成單一層，再利用兩條 LED 魔術方塊 2 線來控制每一層所指定座標的 LED。接著將每一層 LED 的 P 極全部連在一起，得到垂直向度的 64 列 LED。因此，只要外加電壓使得這 64 列的其中一隻 P 腳和其中一層的 N 腳構成順向偏壓，就可以輕易控制 512 顆 LED 中的任一個。

1. LED 方塊製作

(1)LED 燈:將 LED 彎成 90 度並把共陽極都接在一起，如圖 2-1-1 所示

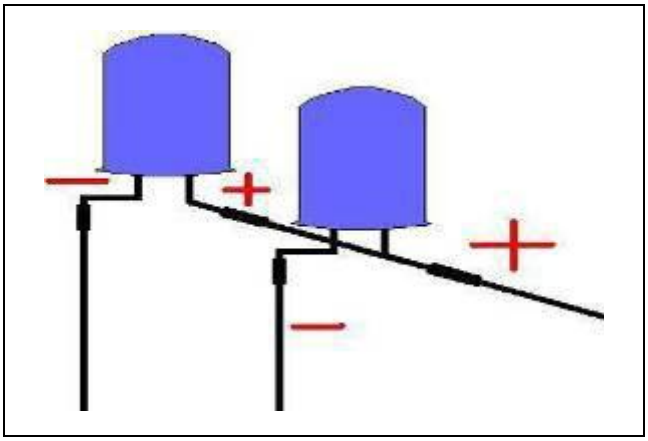


圖 2-1-1 LED 接腳圖

(2)焊接:將 LED 彎成 90 度之後,連接起來變成 8x8 的正方形,如圖 2-1-2 所示。

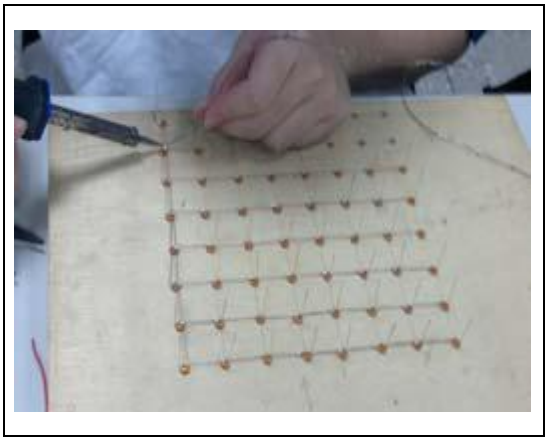


圖 2-1-2 8x8 正方形

(3)焊接:再將一層 64 顆 LED 的正方形，如圖 2-1-3 所示。

圖 2-1-3 每層的 LED

(4)組裝:等到以上的敘述都完成後，就會是一顆 8x8x8 的 LED 方塊了，如圖 2-1-4 所示。

圖 2-1-4 LED 方塊

2.電路測試

在製作電路板之前，不確定我們的想法是否可行，因此先使用麵包板來做實驗，如圖 2-1-5 所示。模擬結果一如預期，於是我們開始著手進行電路板的製作。

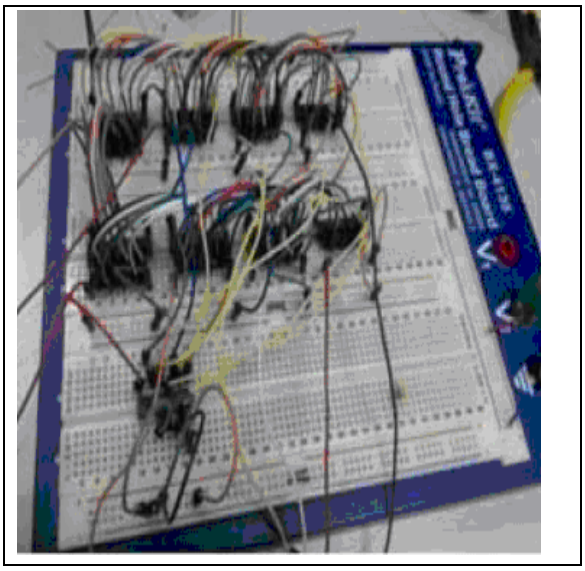


圖 2-1-5 麵包板模擬

(二)軟體架構

在程式方面，我們利用 8051 來撰寫程式，整個程式相當冗長，因為用的是陣列的方式來撰寫每個圖形，然後設定變數讓 LED 方塊一層一層的去做掃描，使每一次掃描就換一次圖形，當掃描的速度很快時，圖形停留的時間就會短很多，會產生類似像視覺暫留的效果。

二、8051 單晶片介紹

(一)8051單晶片的簡介

8051 是 INTEL 公司開發相當成功的單晶片，在教育界中用來當作單片學習的入門首選，由於其使用的普及，因此目前有好幾家設計半導體晶片的公司也有製造與 8051 相容的單晶片，而有些公司所製造出來的單晶片其執行的速度更快，可以高達 40MHz，若使用者想加快單晶片系統的執行速度時可以選用此一類型的 8051 晶片。

8051 單晶片具有以下特點：

1.專為控制應用所設計之八位元 CPU。

2.加強了布林代數(單一位元的邏輯)之運算功能。

3.32 條雙向且可被獨立定址之 I/O。

4.單晶片內部有 128 位元組可供儲存資料的 RAM。

5.內部有兩個 16 位元計時器(8052 有三個)。

6.具全雙工 UART。

7.5 個中斷源，且具有兩層(高/低)優先權順序之中斷結構。

8.單晶片內有時脈(Clock)振盪器線路。

9.單晶片內有 4K(8K/8052)位元組的程式記憶體(ROM)。

10.程式記憶空間可達 64K 位元組。

11.資料記憶體空間可定址到 64K 位元組。

(二)8051 單晶片應用範圍

1.現今的單晶片除了中央處理器(CPU)、記憶體(Memory)、輸入/輸出(I/O)單元等基本架構外，更將計時器/計數器，類比/數位轉換器(A/D)，數位/類比轉換器(D/A)，非同步串列資料傳輸介面(UART)，脈波寬度調變(PWM)信號輸出等，都製作在單晶片中了，所以這麼多種功能的單晶片常被運用於電器產業，例如，冷氣機、液晶顯示器、無人自走車…等。

2. 8051擅長的功能就是控制，很多家電產品都需要做功能控制，例如微波爐有個操作面板，面板上的按鍵跟液晶顯示還有烹調種類的各種微波時間及微波強度都可以用8051控制。你想搞個家庭自動化也可以用8051來自動控制。

3.例如家用保全主機、時鐘、A/D轉換器、D/A轉換、步進馬達控制、甚至是自走車(電腦鼠)、機械手臂、廣告字幕控制、電壓表、鍵盤控制等。

4.在目前數位控制系統的簡易控制機中，採用單晶片可提高其可靠性及增強功能，降低控制機成本。

(三)8051 內部結構

8051 單晶片內部結構介紹，首先必須要知道怎麼樣接腳，當然在這之前，我們必須知道每隻腳的用途，以下圖 2-2-1 所示。

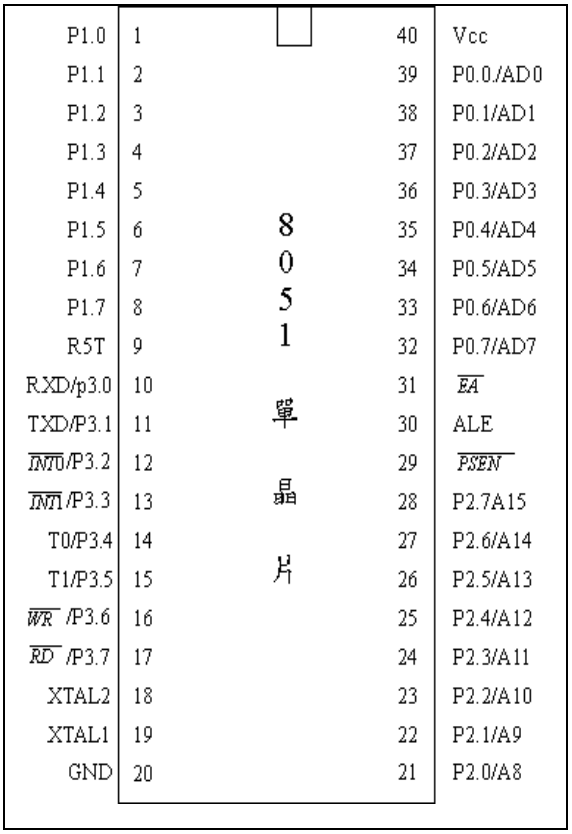


圖 2-2-1 8051 單晶片腳位

1.1~8腳 (P1.0~P1.7)：

這8隻腳是8051的I/O port，稱為PI。第一腳(P1.0)是LSB，第8隻腳(P1.7)是MSB。如果是8052(8032或8752)時，P1.0又可當作Timer2的外部脈波輸入腳，P1.1又當作T2EX，可當作另外一個外部中斷觸發輸入腳。P1上的每隻腳都可推動4個LS TTL。

2.9腳(RESET)：

8051的重置(RESET)輸入腳，當這支腳由外部輸入High(+5V)的信號時，8051就被重置，8051被重置後就從位址0000H開始執行程式。且特殊功能暫存器(SFR)裡的所有暫存器都會被設成已知狀態。

3.10~17腳 (P3.0~P3.7)：

這8隻腳是8051的I/O port，稱為P3。第10隻腳(P3.0)為LSB，第17隻腳(P3.7)為MSB。P3裡的每隻I/O腳除了可以當作單純的輸入/輸出使用外，也當作8051內部的某些週邊與外界溝通個I/O腳。例如

P3.0和P3.1接腳的另外一個名稱為RxD和TxD，當8051內部的UART被軟體啟動後，UART會將串列資料從TxD腳輸出，而UART也接收由外部送進來的串列信號。INT0和INT1是8051的兩個外部中斷輸入部。T0是Timer0的外部脈波輸入腳。T1是Timer1的外部脈波輸入腳。WR，RD，當您再8051的外部擴充資料記憶體(RAM)時，這兩條線是控制寫與讀的信號。P3上的每一隻I/O腳都可以做兩種用途。那8051怎麼知道P3上的某支腳是當I/O或當另一種用途，例如您要使用UART時您將第10腳看成RxD，第11腳看成TxD加以使用就可以了。但是有一點必須特別注意，那就是當作其他功能(不當I/O使用)使用的那支腳的內部栓鎖器的內容必須設為1，其他的功能(如TxD，RxD，RD，ER，…等)才會有作用。P3上的每隻I/O腳都可推動4個LS TTL。

4.18~19腳(XTAL2，XTAL1)：

這兩隻腳是8051內部時脈振盪器的輸入端，您可以在這兩隻腳上跨街一個12MHz的工作頻率，供內部使用。8051會根據這個速度工作。若未特別註明，這個振盪器的工作頻率是在1MHz~12MHz之間的任何一個。如果線路板上已有振盪器，那這個振盪器所產生的脈波(Clock)也可以直接輸入給8051使用。這個外部送給8051使用的脈波是從第18腳(XTAL2)輸入，而19腳(XTAL1)必須接地，以上的接法是CMOS的8051(如8051AH)。

如果您是使用CMOS的8051(80C51，80C31等)，外部的脈波必須從19腳(XTAL1)輸入而18腳空接，這個差別必須特別注意。

5.40，20腳(Vcc，Vss)：

(1)這是8051的電源輸入端，40腳接電源的正端的20腳接地。

(2)電源規格是5V +/- 10% 。

6.21~28腳(P2.0~P2.7)：

這8隻腳是8052的I/O port，稱為P2，P2.0為LSB，P2.7為MSB。P2除了當作I/O使用之外。如果您在8051的外面擴充程式記憶體或資料記憶體時，P2就變成8051的位址匯流排的高位元組(即A8~A15)，此時P2就不能當作I/O使用。P2上的每隻I/O腳可推動4個LS TTL。

7.39~32腳(P0.0~P0.7)：

這8隻腳也是8051的I/O port，稱為P0其中P0.0為LSB，P0.7為MSB。如果將P0當作I/O使用時必須特別注意P0的輸出型態是Open Drain，其他三個I/O port(P1，P2，P3)內部有pull high電路。P0除了當作I/O使用外，如果您在8051的外面擴充程式記憶體或資料記憶體時，P0就當作位址匯流排(A0~A7)和資料匯流排(D0~D7)多工使用。您必須再外部加一個8位元栓鎖器將位址匯流排從PC上分離出來，這個A0~A7與P2所提供的A8~A15合成一個16位元的位址匯流排，因此8051可以在外部定址到64K的記憶體。

8.29腳(PSEN)：

這隻腳是8051用來讀取放在外部程式記憶體的指令時所用的讀去信號，通常這隻腳是接到EPROM的OE腳。8051分別致能放在外部EPROM(程式記憶體)與RAM資料記憶體是兩塊獨立的記憶體，且這兩塊記憶體都可以接到64K，因此我們說8051的定址能力可達128K。

9.30腳(ALE)：

這隻腳的名稱為 ”位址拴住致能”（Address Latch Enable，簡稱ALE)，8051可以使用這隻腳觸發外部的8位元栓鎖器，將P0上的位址匯流排信號(A0~A7)鎖入栓鎖器中。

10.31腳(EA)：

這是一支輸入腳，當EA=0時，8051一律執行外部程式記憶體裡的程式，因此8051內部的4K程式記憶體就沒有用了。因此如果您要使用內部的程式記憶體時，一定要將EA接+5V。因為8031(或8032)內部沒有程式記憶體，它的EA必須接地。

(四)記憶體結構

在MCS-51系列中，除了8031/32沒有內部程式記憶體(ROM)之外，其餘編號均擁有程式記憶體(ProgramMemory)和資料記憶體(DataMemory)，二者是完全分開獨立選址的，其結構如圖2-2-2所示。

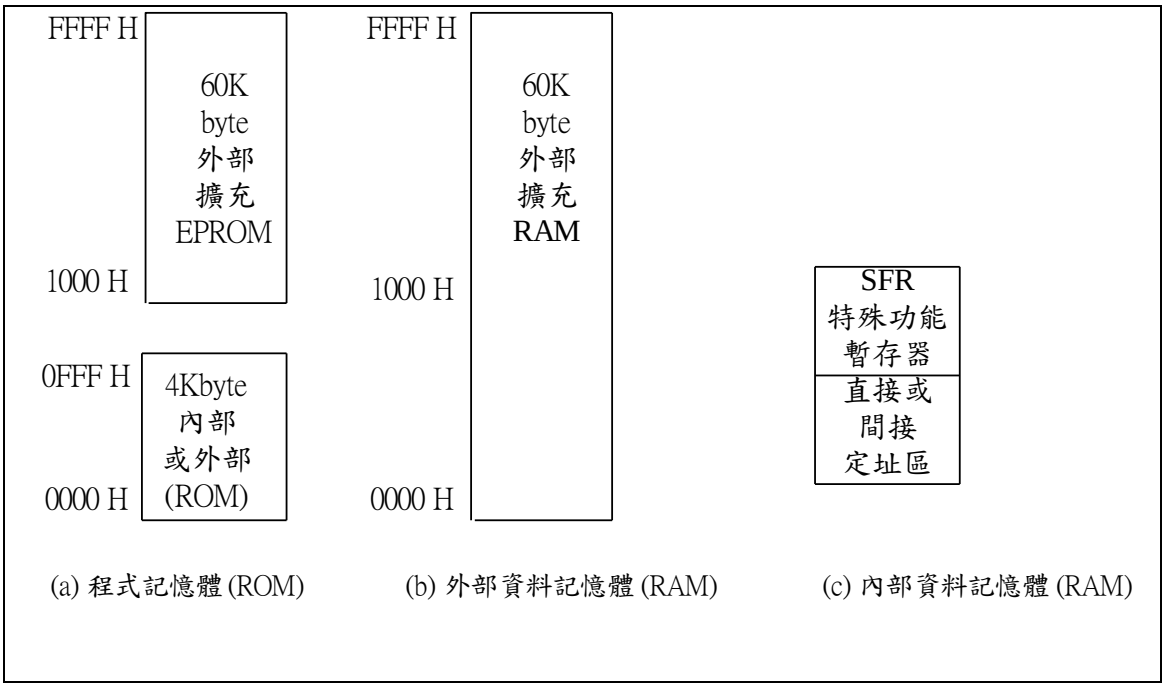


圖2-2-2 記憶體結構

1.程式記憶體

程式記憶體是存放程式碼的地方，CPU會主動到這塊記憶體區域擷取所要執行的指令碼，由於他是採用唯讀記憶體(ROM)的電路結構，因此只能讀取資料而不能執行寫入的動作。圖2-2-3(見下頁)是程式記憶體結構的示意圖，它最大可擴充到64K位元組的記憶體容量。若EA=1時，CPU會到內部程式記憶體中讀取指令碼，如果位址超4K(8051)8K(8052)時，則CPU就會到外部記憶體中提取指令碼。當EA=0時，內部程式記憶體就無效，此時所有的程式指令均從外部程式記憶體中讀取。由於8031/32內部沒有程式記憶體，因此使用必須將EA接腳接地，並從外部擴充程式記憶體。

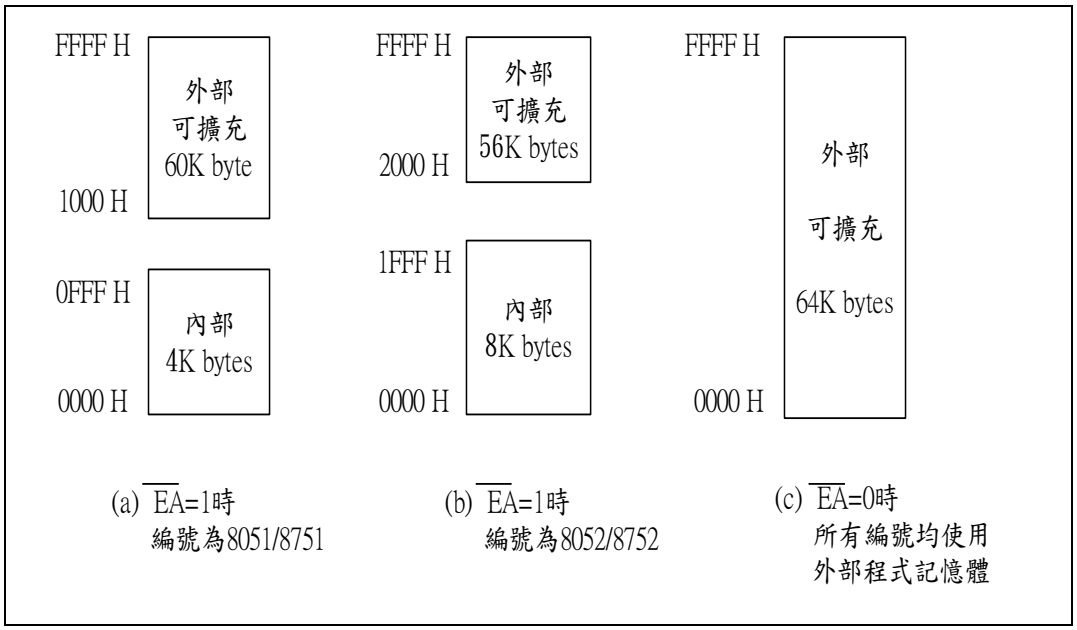


圖2-2-3 程式記憶體

圖2-2-4所示便是MCS-51的中斷向量位址圖，每兩個中斷向量之間隔為8個位元組。假使我們所設計的中斷服務程式之機械碼總長度不超過8個位元組。則需要將其安排在兩個中斷向量間的記憶體空間即可。但若中斷服務程式超過8個位元組，而且下一個中斷向量亦被設為致能的情況下，我們就需要以條件跳躍指令(JMP)作為開頭，以便跳過其下端的中斷向量位址。

中斷源	向量位置	優先權層次
IE 0	0003H	最　高
TF 0	000BH	
IE1	0013H	
TF1	001BH	
R1&T1	0023H	
TF2&EXF2	002BH	最　低

圖2-2-4 MCS-51的中斷向量位址圖

2.資料記憶體

資料記憶體是作為程式運作中暫時存放資料的地方，因其採用隨機存取記憶體(RAM)的電路結構，所以內容會隨電源的關閉消失。可清楚的看到資料記憶體區分為內部與外部，兩者是完全不同的。因此在存取資料時所採用的指令也不相同，存取內部資料記憶體的資料時，必須使用〝MOV〞指令組，然而再存取外部資料記憶體時，則須採用〝MOVX〞指令組。

(1)內部資料記憶體

MCS-51 系列的內部資料記憶體結構如圖 2-2-5 所示，可區分成較低的 128 個位元組(位址 00H~7FH)以及較高的 128 個位元(位址 80H~FFH)和特殊勁能暫存器(SFR，位址 80H~FFH)和特殊勁能暫存器(SFR，位址 80H~FFH)等三個區塊。

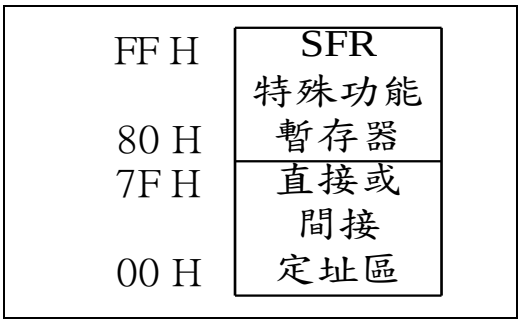


圖 2-2-5 MCS-51 系列內部資料記憶體結構

(五)8051/52 的指令定址模式

所謂定址模式(AddressingMode)是指 CPU 尋找運算元的方法。8051/52 共有六種定址法，分述如下:

1.直接定址法

所謂直接定址法，就是在指令中，直接定運算元所在的位址。僅適用於內部資料記憶體(RAM)及特殊功能暫存器(SFR)。

如下：

```
MOV     A，3FH；把位址 3FH 的內容存入累加器 A
ADD     A，30H；把位址 30H 的內容加到累加器 A
```

2.間接定址法

間接定址法是把運算元的位址存放在一個暫存器，這個暫存器就是運算元位址的指標。

3.暫存器定址法

8051 內部 RAM 的每個暫存器庫均含有 8 個暫存器，稱為 RO-R7，若運算元是使用 RO-R7 的位址都稱為暫存器定址法。

如下：

```
MOV     A，R7；把暫存器 R7 的內容存入累加器 A
MOV     A，R6；把暫存器 R3 的內容加到累加器 A
```

4.立即定址法

立即定址法是把運算元直接放在運算碼的後面。若運算元是常數資料，則必須以“#”號當作立即值的前置符號。

如下：

```
MOV     A ，#30H；把一個常數 30H 存入累加器 A
MOV     R5，#05H；把一個常數 05H 存入 R5 暫存器
```

5.索引定址法

8051 的索引定址法僅適用於 ROM(程式記憶體)，而且只能讀出，不能寫入。所謂索引定址法就是以一個基底暫存器的內容，再加上一個索引暫存器的內容，所得的值即是運算元所在的位址。採索引定址法時，當基底暫存器的是 DPTR(資料指標暫存器)或 PC(程式記數器)，當索引暫存器的則是累加器 A。

如下：

```
MOV     A ，#30H
MOV     DPTR，#300H
MOVC    A，@A+DPTR
                ；將程式記憶體位址 330H(30H+300H)的內容存入
                  累加器 A
```

6.位元定址法

位元定址法是指對內部資料記憶體(RAM)及特殊功能暫存器(SFR)的某個位元直接設定或清除。就因為 8051 具有位元定址法，所以我們可以輕易的控制功能強大的特殊功能暫存器(SFR)，讓 8051 發揮最大效用，這是 8051 很重要的角色。但是位元定址法，只能使用於可位元址的暫存器。

如下：

SETB C ；設定進位旗標 C 為 1。

SETB P1、0；設定埠 1(P1)的第 0 位元為 1。

MOV C，ACC、2

；把累積器 ACC 的第 2 位元的值存入進位旗標。

(六) ATMEL89C51 主要電氣特性(Vcc=5V 時)，如表 2-2-1 所示。

表 2-2-1 89C51 電器特性

符號	名稱	條件	最小	最大
Vcc	電源電壓	5.0V+-20%	4V	6V
ICC	電源電流	正常模式(工作頻率 12Mz 時)		25mA
		IdIe 模式(工作頻率 12Mz 時)		6.5mA
		PowerDown 模式(Vcc=6v 時)		100uA
		PowerDown 模式(Vcc=3v 時)		40uA
VIL	輸入 0 的電壓	-0.5~0.2*Vcc-0.1	-0.5V	0.9V
VIH	輸入 1 的電壓	0.2*Vcc+0.9~Vcc+0.5	1.9V	5.5V
VOL	輸入 0 的電壓	P1~P3(IOL=0.6mA 時)		0.45V
VOL1	輸入 0 的電壓	P0(IOL=0.6mA 時)		0.45V
VOH	輸出 1 的電壓 (埠 1、3) (Vcc=5v⑥ 10%)	IOH=-60uA	2.4V	
		IOH=-25uA	3.75V	
		IOH=-10uA	4.5V	
VOH1	輸出 1 的電壓 (埠 0) (Vcc=5v⑥ 10%)	IOH=-800uA	2.4V	(須外加提升電阻)
		IOH=-300uA	3.75V	
		IOH=-80uA	4.5V	
IIL	輸入 0 的電流	P1~P3(VIN=0.45V 時)		-50uA
ILI	輸入洩漏電流	P0(0.45<VIN<Vcc 時)		±10uA
ITL	1 到 0 轉換電流	P1~P3(VIN=2V 時)		-650uA
RRST	重置下降電阻	PulldownResistor	50KΩ	300KΩ

參、專題製作

此章共分為三節依序說明本專題所應用到之設備及器材、製作方法與步驟及專題製作等。

一、設備及器材

表 3-1-1 專題製作使用儀器（軟體）設備一覽表

儀器（軟體） 設 備 名 稱	應 用 說 明
個人電腦	專題報告、電路圖製作及進行專題成品電路測試
數位相機	拍攝小組合作過程、專題功能使用及紀錄整個專題製作流程
雷射印表機	列印專題資料、圖片及專題報告成果
三用電錶	測量零件有無損壞及專題電路板各信號之量測
電源供應器	提供專題成品所需之電源
Microsoft Office Word	專題報告、製作過程的撰寫
Microsoft Office Power Point	進行口頭報告、製作及專題成品報告呈現

二、製作方法與步驟

一開始就先用 89S51 單晶片把程式燒入進去，之後由 89S51 同時送 data 到 8 顆 5821 的 IC 裡面，然後由 IRF530 來控制是哪一層 LED，由於我們是以一個圖形來呈現，讓間隔時間很短暫，使圖形看起來是有 3D 立體的感覺。

而礙於時間及金錢等因素我們選擇 8x8x8 來研究這次的專題，希望能依照自己的概念來顯示出所所想要的圖形。完成之後，由於 8x8x8 依個面只有 64 個點導致我們無法顯示中文字，只能以單純的圖形數字及英文字來呈現我們的作品。

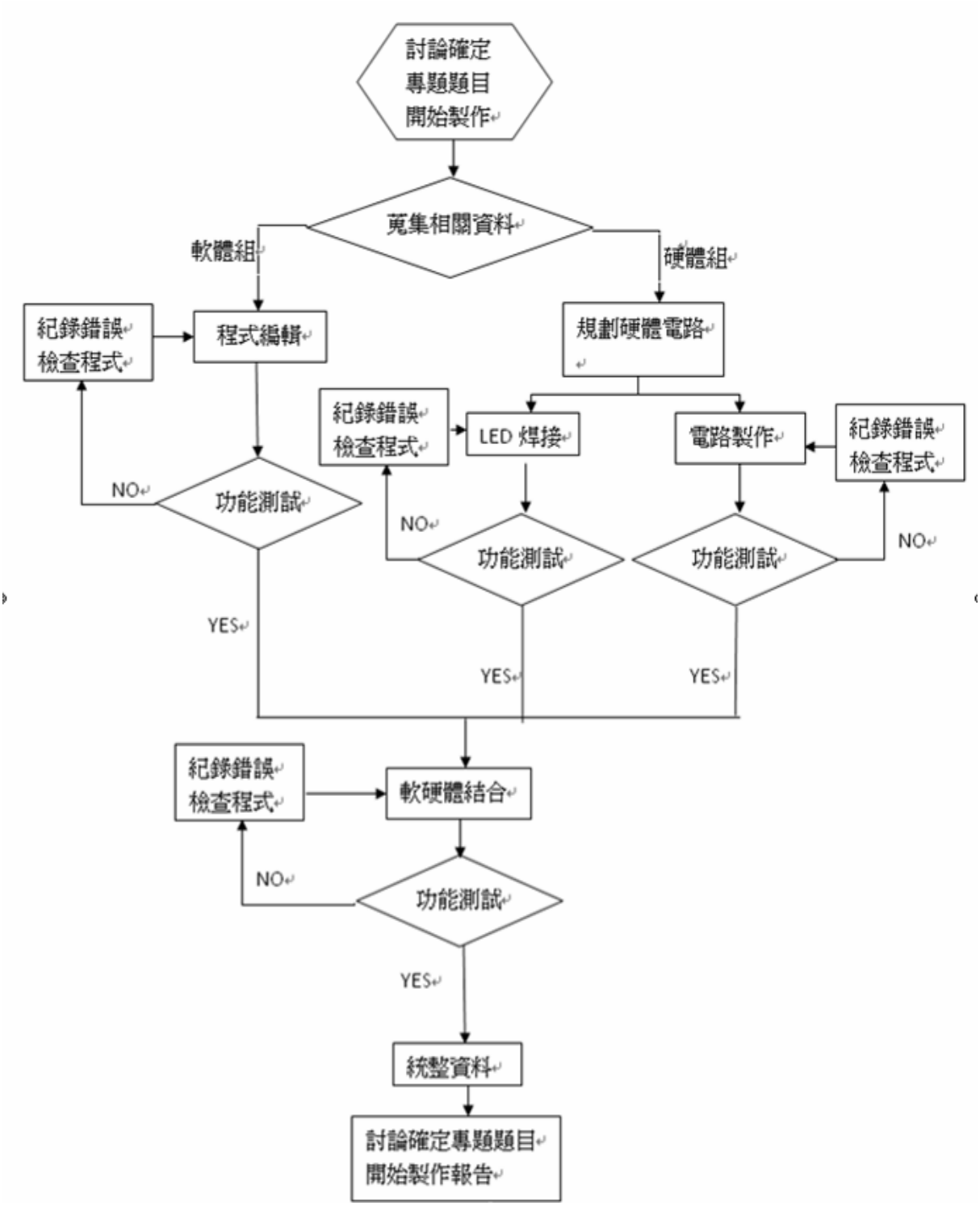


圖 3-2-1　製作方法與步驟

三、專題製作

表 3-3-1 專題製作計畫書

<table>
<tr><td colspan="2">專題型別</td><td colspan="2">☐ 個人型專題　　☑ 團隊型專題</td></tr>
<tr><td colspan="2">專題性質</td><td colspan="2">利用單晶片 89C51 製作自動化控制電路</td></tr>
<tr><td colspan="2">科別／年級</td><td colspan="2">資訊 科 三 年級</td></tr>
<tr><td rowspan="2">專題名稱</td><td>中文名稱</td><td colspan="2">8x8x8 3D 光立方</td></tr>
<tr><td>英文名稱</td><td colspan="2">8x8x8 led cube</td></tr>
<tr><td colspan="2" rowspan="8">專題內容簡述</td><td colspan="2">為了節能省電 LED 有逐漸取代傳統燈泡的趨勢，但是目前</td></tr>
<tr><td colspan="2">市面上的 LED 廣告燈大多是以平面的方式呈現，我們希望</td></tr>
<tr><td colspan="2">能夠組合出一個 LED 方塊，可以讓平面變成立體。在數位</td></tr>
<tr><td colspan="2">邏輯實習課程中,我們曾經學習到如何使用 TTL 邏輯閘來控</td></tr>
<tr><td colspan="2">制 LED 的方法，於是聯想到若將大量的 LED 組成一個大</td></tr>
<tr><td colspan="2">的立體方塊，又可讓我們隨意控制每顆 LED 的明滅時，會</td></tr>
<tr><td colspan="2">使得立體方塊呈現許多不同圖案或文字的組合變化，應該會</td></tr>
<tr><td colspan="2">很有趣。</td></tr>
<tr><td colspan="2">指導老師姓名</td><td colspan="2">簡琨祥 老師</td></tr>
<tr><td colspan="2" rowspan="2">參與同學姓名</td><td>陳亦昀(資訊 3-2)</td><td>郭建順(資訊 3-2)</td></tr>
<tr><td>楊莉彤(資訊 3-2)</td><td>王姿懿(資訊 3-2)</td></tr>
<tr><td colspan="2">專題執行日期</td><td colspan="2">103 年 09 月　日至 104 年 5 月　　日</td></tr>
</table>

	
圖 3-3-2 查詢專題資料	圖 3-3-3 小組同學進行專題討論

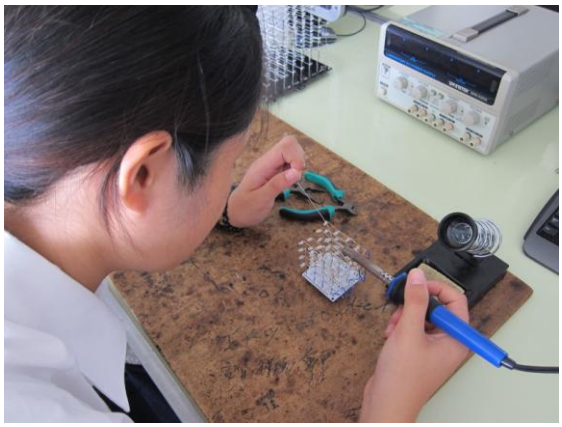
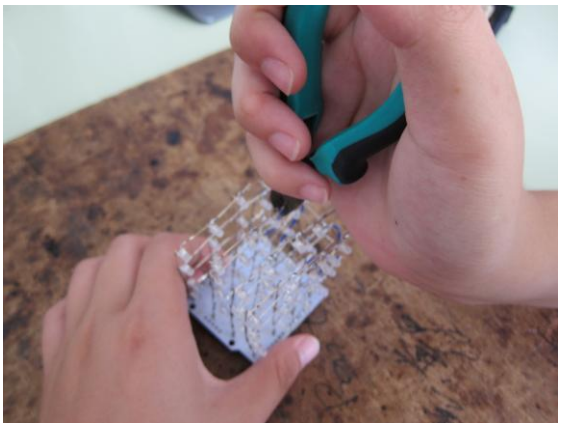
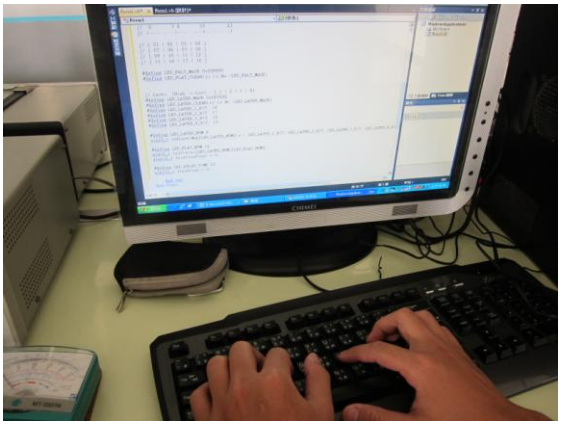

	

圖 3-3-4 小型模型製作(一)	圖 3-3-5 小型模擬製作(二)
	
圖 3-3-6 撰寫程式與燒程式(一)	圖 3-3-7 撰寫程式與燒程式(二)

(一) 硬體電路圖:3D 光立方

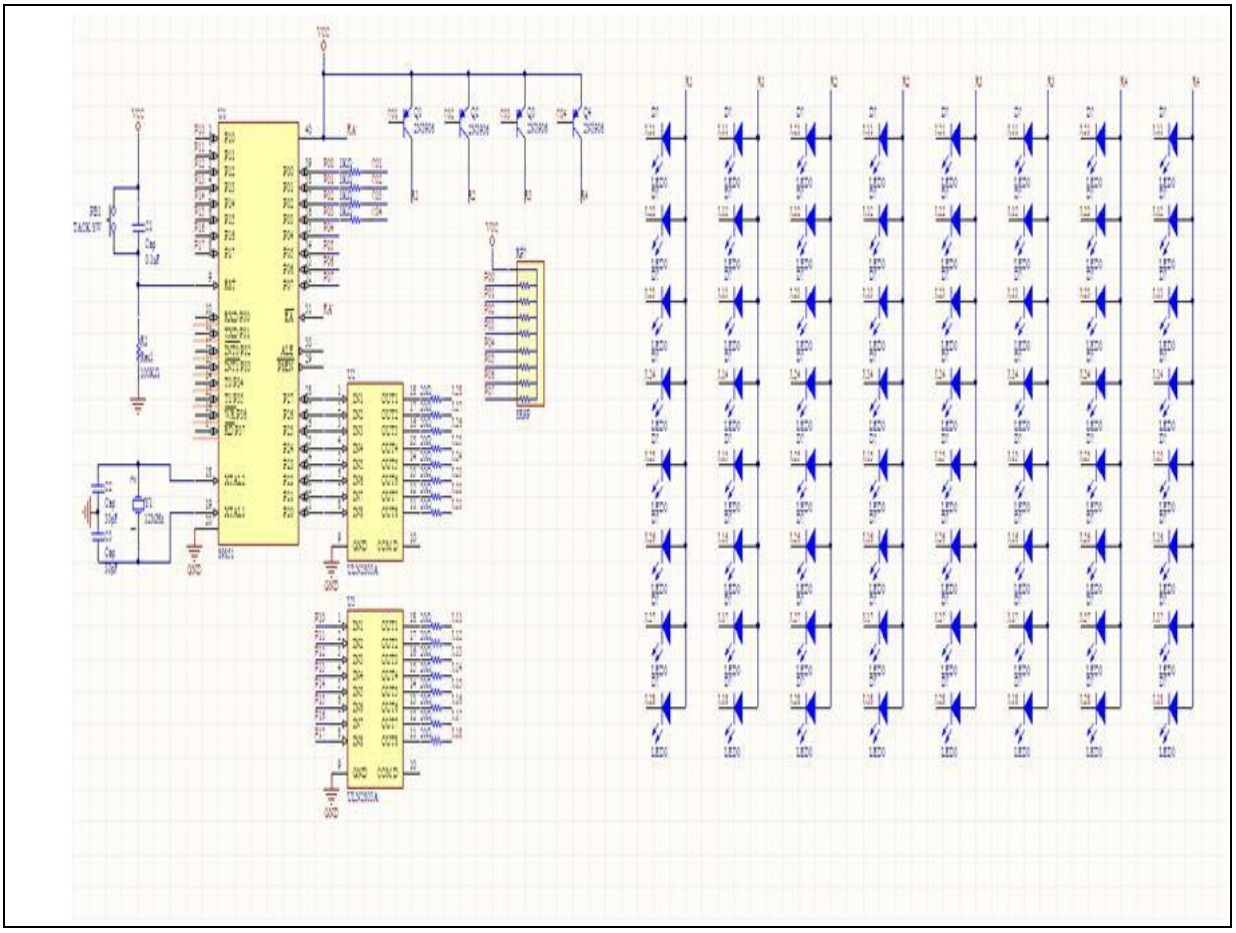


圖 3-3-8 3D 光立方之完整電路圖

(二) 8x8x8 3D 光立方之電路板 Layout 圖及材料表：

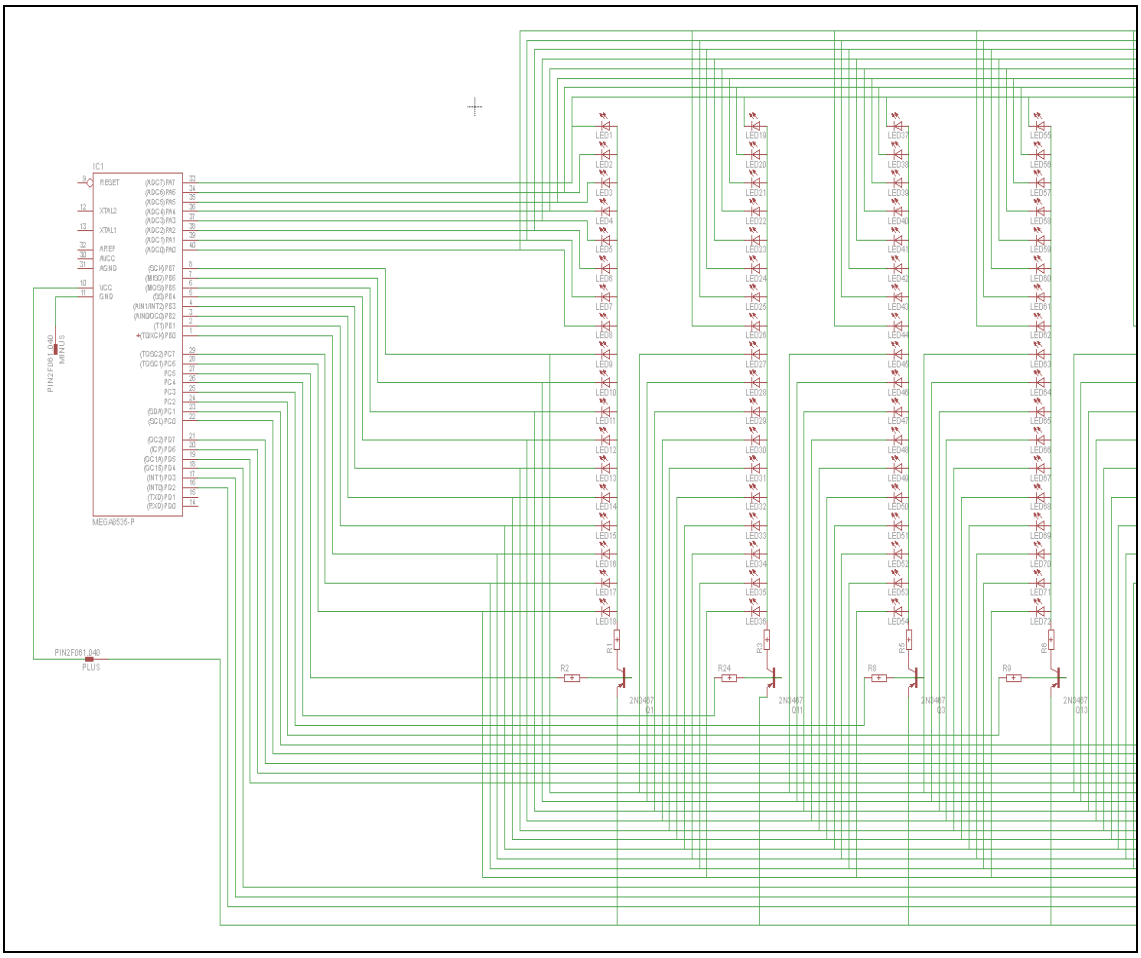


圖 3-3-9 8x8x8 3D 光立方的 Layout 圖

表 3-3-2 8x8x8 光立方之材料表

材料名稱	規格	單位	數量	備註
LED	3mm	顆	512	
微控制器	ATmega32	顆	1	
單晶片	74HC574	顆	8	
單晶片	74HC138	顆	1	
晶體管	PN2222	顆	16	
電阻	1k	顆	16	
電晶體	14.7456	顆	1	
陶瓷電容器	22PF	顆	2	
陶瓷電容器	0.1uF	顆	16	
電容	10uF	顆	3	
電容	100uF	顆	1	
IC 座	40pin	顆	1	
IC 座	20pin	顆	1	
IC 座	16pin	顆	2	
按鈕		顆	4	

(四)小組分工的配置：

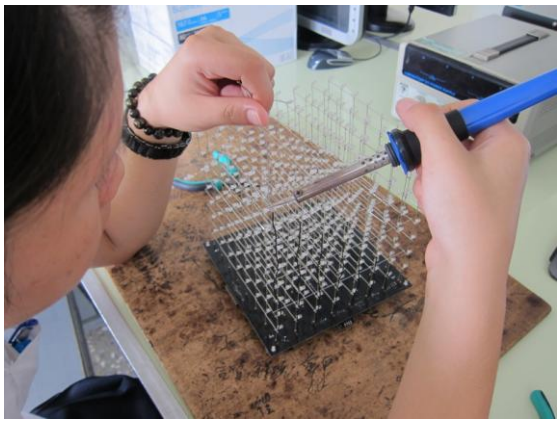
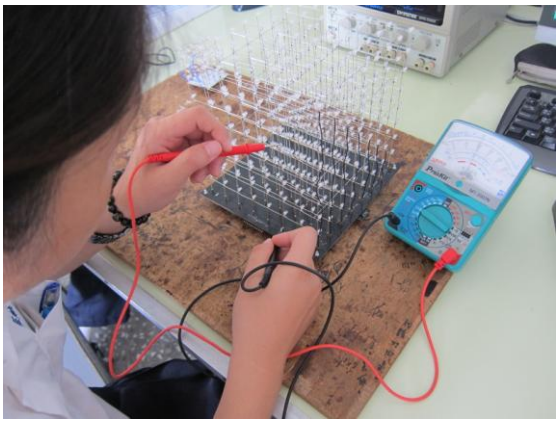
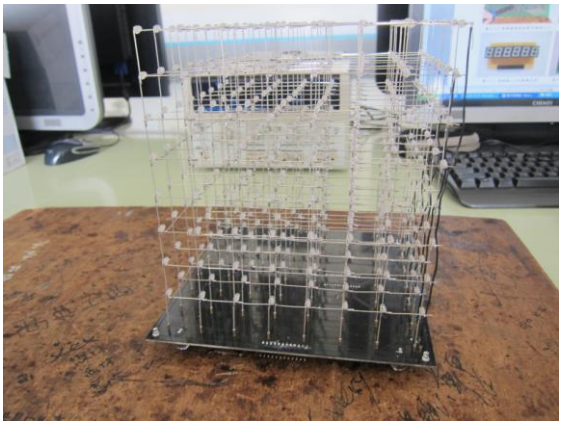
姿懿負責小組的資料，及整合簡報內容，過程中會有購買相關書籍當成參考資料，選擇要如何去製作專題，讓亦昀和莉彤知道要如何做專題，然後再經過小組討論、商量，有問題時，會再去徵詢老師的意見與想法。

亦昀要知道如何做出 8x8x8 3D 光立方的電路，負責焊接及畫出電路圖與焊接背面的電路圖；在製作過程中，如發現錯誤時，會再和小組想辦法如何補救，順便了解程式是否能夠有作用。

莉彤和建順負責去買電路中所需之零件，我們決定要多買一組當備用零件，假如一次就可成功就算是多買的，假如不成功，就需要去用到備份零件；且我們可少買一些可以重複使用的零件，如：IC、單晶片等，藉此便可知道要花多少錢，大家再去平分所需的金額。

肆、製作成果

我們小組由決定題目，製作模擬電路、繪製設計電路圖，進而完成焊接製作整個電路；這整個流程，我們小組都用數位相機及相關電腦設備將之紀錄下來，經將這些資料整理過後，我們將之呈現在我們的專題報告之中，如下所示：

<table>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 4-1-1 光立方製作</td><td>圖 4-1-2 光立方成品測量</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 4-1-3 小組研究討論</td><td>圖 4-1-4 問題探討</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 4-1-5 與指導老師解決問題</td><td>圖 4-1-6 成品圖</td></tr>
</table>

伍、結論與建議

本單元我們將針對我們小組對專題製作的整個學習過程，做最後完整的彙總及記錄，以期作為未來學弟妹們日後學習之參考。

一、結論

(一) 透過此次專題製作學習的方式能幫助我們提升對課程的學習與興趣。

(二) 透過此次專題製作學習的方式更能幫助我們獲得知識的建構及整合，且亦可以幫助我們提升其創造思考的能力。

(三) 加入更多的學習互動設計與具吸引力的輔助學習機制

(四) 專題製作學習鼓勵小組成員分工和合作學習的精神。

(五) 小組同學認同資訊科技的知識在專題製作學習過程中，扮演著重要的角色，因其對電路製作、資料呈現及成果報告製作是很有幫助的。

(六) 專題製作學習可以培養我們學習者具備問題解決、研究、反省、團體合作及應用資訊科技等多項能力。

(七) 小組同學認為專題製作學習的階段中，會遇到不同的困難及問題，但看到自己的成品時，會很有成就感。

(八) 經過這次製作及研究 3D LED CUBE 中，許多東西都是我們未接觸過的事物，我們從無到一知半解再到學會製作、控制整個實體電路與 LED CUBE 並可以做出許多變化的動作，

二、建議

我們在進行專題製作學習的過程後，提出以下幾點建議：

(一)經過這次製作及研究 3D LED CUBE 中，許多東西都是我們未接觸過的事物，我們從無到一知半解再到學會製作、控制整個實體電路與 LED CUBE 並可以做出許多變化的動作，中間遇到許多的難題不管是在電路方面還是程式方面，每當遇到問題時，老師都會不厭其煩的重複為我們講解指導，令我們能理解各項問題的所在。

(二)我們學會了如何使用 Protel99SE 繪製設計電路配線圖，和單晶片軟體來撰寫程式，還有第一次使用製作將 LED 每顆相連接排列成一層一層，最後再一層一層的架起來變成 8x8x8 的 LED CUBE。經過這次的專題我們接觸了許多第一次嘗試的工具，還有接收新的知識讓我們收穫甚多。

(三)在學習過程中給予回饋，同學建議，在專題製作學習研究過程中，老師能否可以在學習的進行過程中，給予立即性的回饋，讓學生可以及早發現其缺失，盡早進行改善。

參考文獻

1.黃嘉輝，2013，8051 單晶片原理與應用，台北縣：台科大圖書公司。

2.長高企業，1998，U3-TARGET 單晶微電腦實驗裝置，台中市：長高企業公司。

3.郭庭吉，2008，8051 單晶片專題製作，台北縣：台科大圖書公司。

4.蔡朝洋，2007，單晶片電腦 8051/8951 原理與應用，台北縣：全華圖書公司。

5.鄧明發，陳茂璋，2000，專題製作應用電路，台北市：知行文化公司。

6.鍾明政，1999，單晶片 8051 原理與實作，台中市：長高企業公司。

附錄一　8x8x8 3D 光立方程式碼

```
/*
   Project :
      Cube LED
   Module :
      LED module
   Description :
      This module implement how to display information by led.

Version 0.1 :
   2013/10/29 -[Alen Chen] - Initial

*/

#define LED_DATA_PIN 2
#define LED_LATCH_PIN 3
#define LED_CLOCK_PIN 4

#define LED_ENABLE 7
#define LED_ALL_CLEAR 8

#define LED_DISPLAY_ON digitalWrite(LED_ENABLE, LOW);
#define LED_DISPLAY_OFF digitalWrite(LED_ENABLE, HIGH);


#define LED_MAX_NUM 24
#define LED_SHIFT_WIDTH 3    // 2^x
#define LED_SET_ONE_BIT(data,index) (data |= ((uint32_t)1 << index))
```

```
//    0             7 8             15              23
// |---- ----|---- ----|---- ----|

// [ 01 | 02 | 03 | 04 ]
// [ 05 | 06 | 07 | 08 ]
// [ 09 | 10 | 11 | 12 ]
// [ 13 | 14 | 15 | 16 ]

#define LED_FALT_MASE 0x00FFFF
#define LED_FLAT_CLEAR(x) (x &= ~LED_FALT_MASE)



// Layer    [High -> Low] : [ 1 | 2 | 3 | 4]
#define LED_LAYER_MASK 0x0F0000
#define LED_LAYER_CLEAR(x) (x &= ~LED_LAYER_MASK)
#define LED_LAYER_1_BIT    16
#define LED_LAYER_2_BIT    17
#define LED_LAYER_3_BIT    18
#define LED_LAYER_4_BIT    19

#define LED_LAYER_NUM 4
uint8_t       ledLayerMap[LED_LAYER_NUM]       =       {       LED_LAYER_1_BIT,
LED_LAYER_2_BIT, LED_LAYER_3_BIT, LED_LAYER_4_BIT };

#define LED_FLAT_NUM 16
uint8_t ledStatus[LED_LAYER_NUM][LED_FLAT_NUM];
uint32_t durationTimer = 0;

#define LED_DELAY_TIME 20
uint32_t flushTime = 0;
```

```
uint32_t systemTime = 0;

void LED_send_data(uint32_t data)
{
  digitalWrite(LED_LATCH_PIN, LOW);
  for(int index = 0; index < (LED_MAX_NUM >> LED_SHIFT_WIDTH); index++)
  {
    shiftOut(LED_DATA_PIN, LED_CLOCK_PIN, MSBFIRST, (data >>
(((LED_MAX_NUM >> LED_SHIFT_WIDTH) - index - 1) <<
LED_SHIFT_WIDTH)));
  }
  digitalWrite(LED_LATCH_PIN, HIGH);
}

void setup()
{
  pinMode(LED_DATA_PIN, OUTPUT);
  pinMode(LED_LATCH_PIN, OUTPUT);
  pinMode(LED_CLOCK_PIN, OUTPUT);
  pinMode(LED_ENABLE, OUTPUT);
  pinMode(LED_ALL_CLEAR, OUTPUT);

  for(int layer_index = 0; layer_index < LED_LAYER_NUM; layer_index++)
    for(int flat_index = 0; flat_index < LED_FLAT_NUM; flat_index++)
      ledStatus[layer_index][flat_index] = 0;

  LED_reset();
}

void LED_reset()
```

```
{
  digitalWrite(LED_ALL_CLEAR, LOW);
  delay(10);
  digitalWrite(LED_ALL_CLEAR, HIGH);
}

void LED_setOne(uint8_t layerIndex, uint8_t flatIndex, uint8_t value)
{
  ledStatus[layerIndex][flatIndex] = value;
}

void LED_setFlat(uint8_t layerIndex, uint8_t * value)
{
  for(int index = 0; index < LED_FLAT_NUM; index++)
    LED_setOne(layerIndex, index, value[index]);
}

void LED_setCube(uint8_t ** value)
{
  for(int layerIndex; layerIndex < LED_LAYER_NUM; layerIndex++)
    for(int flatIndex = 0; flatIndex < LED_FLAT_NUM; flatIndex++)
      LED_setOne(layerIndex, flatIndex, value[layerIndex][flatIndex]);
}

void LED_flushFlat(uint8_t layerIndex)
{
  uint32_t data = 0;

  if( ((flushTime & 0xC0) >> 6) == layerIndex)
    data = (uint32_t)1 << ((flushTime & 0x3C) >> 2);
```

```
  LED_SET_ONE_BIT(data, ledLayerMap[layerIndex]);
  LED_send_data(data);
}

void loop()
{
      systemTime++;
      if((systemTime % LED_DELAY_TIME) == 0)
      {
  flushTime++;
         if(flushTime > 0xFF) flushTime = 0;
      }

      for(int index = 0; index < LED_LAYER_NUM; index++)
  LED_flushFlat(index);
}
```