十二年國民基本教育

技術型高級中等學校電機與電子群

單晶片微處理機實習

（全一冊）

洪國勝、孫銘宏、蔡懷文
鍾享旭、蔡懷介、陳蘊慈 編著

泉勝出版有限公司
www.goodbooks.com.tw

編輯大意

1. 本書係依據民國 107 年 8 月教育部發布之技術型高級中等學校電機與電子群「單晶片微處理機實習」課程綱要編輯而成。
2. 本書全一冊，供技術型高級中等學校電機與電子群所屬類科等，第二學年第二學期三學分教學之用。
3. 本書所用電子材料都是常見標準零件，可以很平價的方式取得。本書亦提供教學實驗教具，只要使用杜邦線就可以完成電路，非常適合團體教學。
4. 選擇 Arduino 作爲開發工具。Arduino 的特色是它主張開源，它的軟硬體都是開放，官網有非常詳盡的軟硬體資料，是學習單晶片設計最好的入門工具。
5. Arduino 的指令主要來自 C/C++ 語言的精華，但深怕部分學生沒有精熟一年級的「程式設計實習」課程，所以本教材還是先複習一些程式語言基本架構。
6. 所有指令都有範例解說，讓程式不只是程式，而是可以解決日常生活問題的工具。
7. 充分的自我練習。我們有充分的自我練習，這樣學生才能充分練習，達到舉一反三的效果。
8. 跨領域學習。本書的範例涵蓋基本電學、數位邏輯設計、資訊科技與生活科技，完全呼應與配合 108 課綱的跨領域教學與學習。
9. 本書章節前加註 ※ 者，爲進階範例，教師可依實際教學進度酌予刪減。
10. 本書雖力求嚴謹，但疏漏之處在所難免，尚祈先進惠予指正。

本書學習表現如下：

1. 認識單晶片微處理機之相關基本原理，了解單晶片工作原理及設計各種介面硬體電路、軟體技術與發展環境及控制週邊元件，具備符號辨識、查閱專業使用手冊、認識與分析接線圖或電路圖之基礎能力。
2. 具備使用實驗開發工具進行軟硬體開發快速設計之能力，以系統思考、規劃執行及科技資訊運用，以解決專業上的問題。
3. 具備高階程式之除錯能力，以科技資訊運用、問題解決、溝通協調及團隊合作之精神，積極面對與解決職場各種問題。
4. 認識單晶片微處理機工場設施，並了解工業安全及衛生與消防安全相關知識，建立職場倫理及重視職業安全，並展現良好的工作態度與情操。
5. 能思辨勞動法令規章與相關議題，省思自我的社會責任。

教學注意事項：

1. 本科目爲技能領域實習科目，得依據相關規定實施分組教學。
2. 本課程教學內容及實施，須與「微處理機」課程密切配合，由學習內容之主題進行觀察或驗證教學內容，以提高學生學習成效。
3. 在本課程授課中，要使學生能了解使用軟體的智慧財產權問題，培養公民意識與社會責任。

目錄

▶第5章　基礎應用控制

▶第6章　進階應用控制

▶附錄

圖目錄

▶第一章

▶第二章

▶第三章

▶第四章

▶第五章

▶第六章

表目錄

▶第一章

▶第三章

▶第四章

▶第五章

▶第六章

Chapter
01

工場安全及衛生

學習大綱

- 1-1　實習工場設施環境及機具設備的認識
- 1-2　工業安全及衛生、消防安全的認識

學習目標

- 1. 瞭解實習工場的各種設施並且安全的使用及發揮各種設施的最大效能。
- 2. 瞭解工業安全及衛生的相關法規及注意事項以確保生命及財產安全。
- 3. 瞭解消防安全的重要性並培養意外發生時的緊急應變能力。
- 4. 培養學生良好的工作習慣並瞭解程式應用如何帶給人們便利的日常生活。

1-1 實習工場設施環境及機具設備的認識

工場安全教育是要讓從事工業相關之從業人員，透過教育訓練而具有工業安全之知識與技能，讓所有從業人員在安全的工作環境中工作，加上適當的安全防護措施，用科學與安全的方法從事工作，以防止工場意外事故之發生，減少損失。另外，實習課程的教學必須分門別類在各種不同的專門實習工場實施，在一般職業類科學校的實習工場中，主要的設施有：實習儀器設備、場地照明及通風設備、人員機具安全防護設備、及各類標示牌等等，如圖 1-1。

單晶片微處理機實習工場環境及機具說明。

各校實習工場設備如圖 1-2，因地制宜各有特色，通常單晶片微處理機實習工場主要建置設備有下列各項：

▮實習儀器設備

1. 個人電腦	2. 教學廣播系統
3. 作業系統	4. 應用軟體
5. 投影機	6. 印表機
7. 有線網路 / 無線網路	8. 交換機
9. 伺服機	10. 不斷電系統 (UPS)

▮場地照明及通風設備

1. 日光燈	2. 緊急照明燈
3. 通風扇	4. 空調

▮人員機具安全防護設備

1. 滅火器	2. 漏電、過載斷路保護器
3. 煙霧偵測器	4. 急救箱
5. 緊急求救鈴	6. 緊急廣播系統
7. 消防栓	8. 緩降機

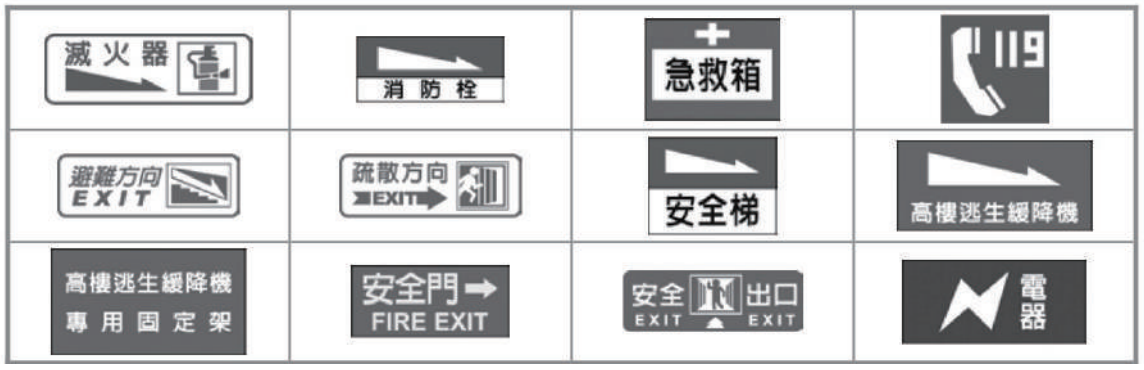


圖 1-1　實習工場各類標示牌

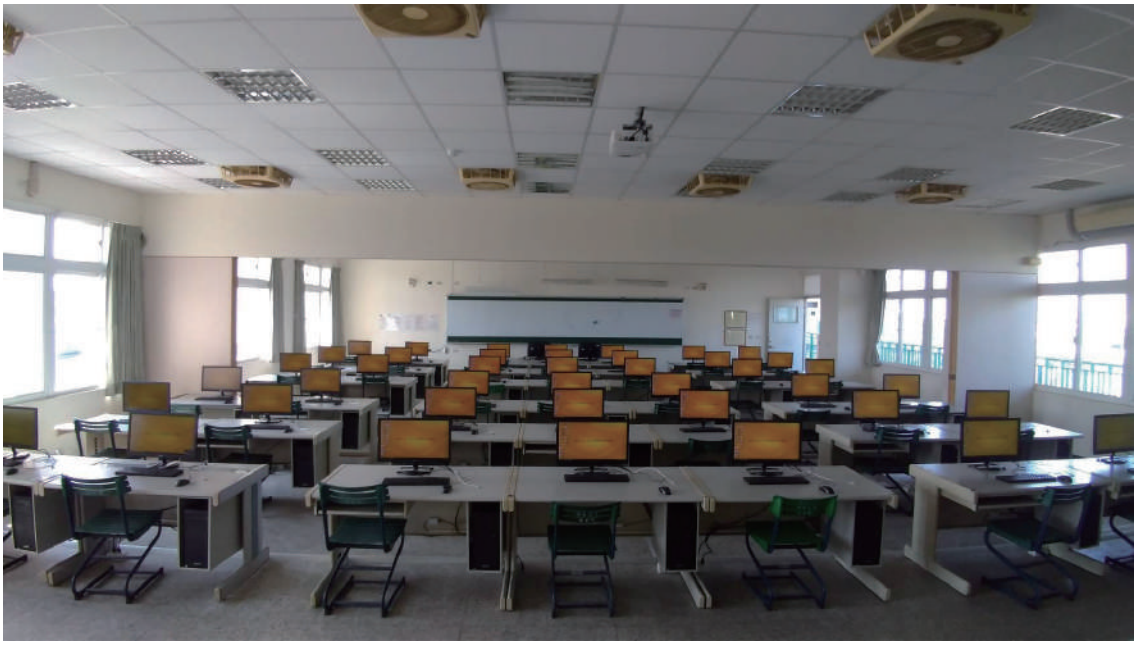


圖 1-2　單晶片微處理機實習工場

1-2 工業安全及衛生、消防安全的認識

工業安全 (職業安全) 與工業衛生 (職業衛生) 以及消防安全的基本目標是一樣的，皆是致力於維護工作人員的安全與健康，避免意外事故的發生，以及災害發生時的緊急應變措施，所以工 (職) 業安全與工 (職) 業衛生以及消防安全是先進國家非常重視的課題。工業安全是指「透過各種安全防護措施，以避免工業災害的發生。」工 (職) 業安全課程包括：意外事故與職業傷害、機具安全與維護、電器安全與維護等。工 (職) 業衛生是指「分析工 (職) 業環境對工作人員健康影響的一切因素，進而利用科學方法去預防和減少工作人員產生疾病和傷害。」工 (職) 業衛生課程包括：危害物的管理與回收處理、急救、工作環境清潔等。我國憲法第 153 條規定：「國家爲改良勞工及農民之生活，增進其生產技能，應制定保護勞工及農民之法律，實施保護勞工及農民政策。」並且明文規定「勞動基準法」爲國家實現此一基本國策所制定之法律。

工業安全與衛生。

▮工業安全及衛生之標示顏色相關規定

1. 紅色表示：危險或禁止標示。(例如：緊急控停止鈕、禁煙標示等。)
2. 橙色表示： 可能發生立即災害或傷害的地方所用之標示。(例如：爆裂物、電擊傷害等。)
3. 黃色表示：注意或警告標示。(例如：易燃物、墜落、跌倒等。)
4. 綠色表示：無危險或衛生有關標示。(例如：逃生門、急救箱。)
5. 藍色表示：提醒注意標示。(例如：控制按鈕、指示燈。)
6. 白色表示：一般標示。(例如：方向指示。)
7. 紫色表示： 放射性危險警告或污染標示。(例如：放射性物質現場警告標示。)
8. 黑色表示：輔助或提醒用標示。(例如：文字標語。)

▮工業安全及衛生之意外發生與急救處理

一般事故的發生，大多數是人爲因素所造成的，只要透過教育訓練，確定每個工作人員具備安全觀念和知識，並且消除不安全之機具設備的因素與環境因素，便可以使意外事故發生減少到最低。爲了防止事故的發生，必先探究事故發生的原因，一般意外事故 發生的主要原因有三：

1. **人員的因素**：意外事故發生中大部分是肇因於工作人員的粗心大意、不熟悉作 業流程或是精神狀態不佳所造成的。
2. **機具設備的因素**：機具設備若缺乏安全防護，或是未落實定期保養也是意外事故發生的主要因素之一。
3. **環境因素**：工場照明不良、通風不良、噪音太大、粉塵、有毒氣體、不適當 的建築結構與隔間或是不整潔的廠房也是意外事故發生的主要因素之一。

另一方面，對於意外事故的發生我們除了須盡力防範之外，也必須了解事故的發生仍然無法絕對避免，因此，急救處理便是給予意外事故發生時受傷者的緊急照料，以維持傷患之生命現象，除去傷害因素，阻止傷勢惡化並給予傷患心理上之鼓舞及信心，直到受傷者獲得醫護人員的救治爲止。簡單來說，急救處理的目的就是救命，當意外事故發生時，請依圖 1-3 學校場所意外通報流程快速通報學校，若有人員受傷，通常會依傷患受傷的類別來進行急救。各受傷類別之急救方法分述如下：

1. **外傷急救**
   (1) 使患者平躺，並將出血部位設法抬高。
   (2) 以清潔的紗布止血，綁緊繃帶，控制出血，但不可過緊以免影響血液循環，使肌肉組織壞死。
   (3) 盡快送外科醫院診治。

2. **骨折急救**
   (1) 閉鎖性骨折：斷骨端未刺穿皮膚。
      a. 使患者平躺，盡可能將傷肢置於自然的位置。
      b. 使用木板固定患部，並用繃帶或布條固定之。

(2) 開放性骨折：斷骨端刺穿皮膚。

a. 用壓力繃帶止血。

b. 使患者平躺，盡可能將傷肢置於自然的位置。

c. 使用木板固定患部，並用繃帶或布條固定之。

3. 灼傷急救

人體的皮膚因受到熱、冷、火、化學藥品等傷害，稱之爲灼傷，灼傷的急救方法應以「沖」、「脫」、「泡」、「蓋」、「送」五大步驟 施行。

(1) 沖：以流動的冷水沖洗灼傷部位 15 ～ 30 分鐘。

(2) 脫：於冷水中小心脫除灼傷部位的衣物。

(3) 泡：於冷水中浸泡 15 ～ 30 分鐘，以降低傷口疼痛。

(4) 蓋：以乾淨的毛巾蓋住傷口，防止細菌感染。

(5) 送：快速送醫治療。

4. 大量出血急救（需注意血液循環的問題）

(1) 直接壓迫傷口止血法。

(2) 壓迫止血點止血法。

(3) 止血帶止血法

5. 昏迷無心跳無呼吸急救

當傷患傷患呼吸停止無心跳時，如沒有即時急救必定死亡，因此必須立即施行心肺復甦術 (Cardio Pulmonary Resuscitation, CPR)(注意：沒有經過完整訓練並取得合格證書者，不可隨意對傷患進行 CPR 急救)，依據 2011 年我國衛生署函文說明的新版 CPR 急救法，分爲六個程序，簡稱「叫叫 CABD」，如圖 1-4 所示：

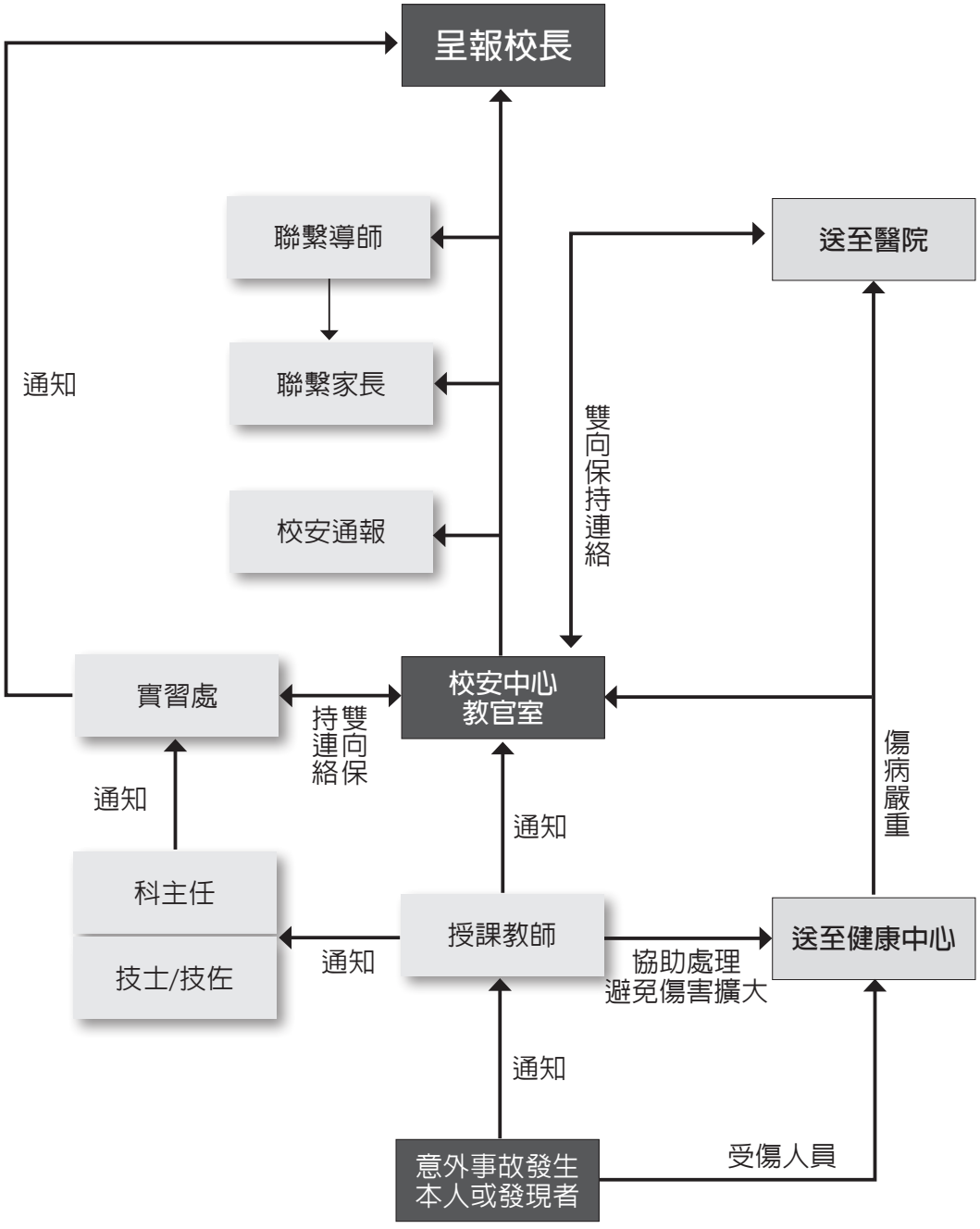


圖 1-3　學校實習場所意外通報流程圖

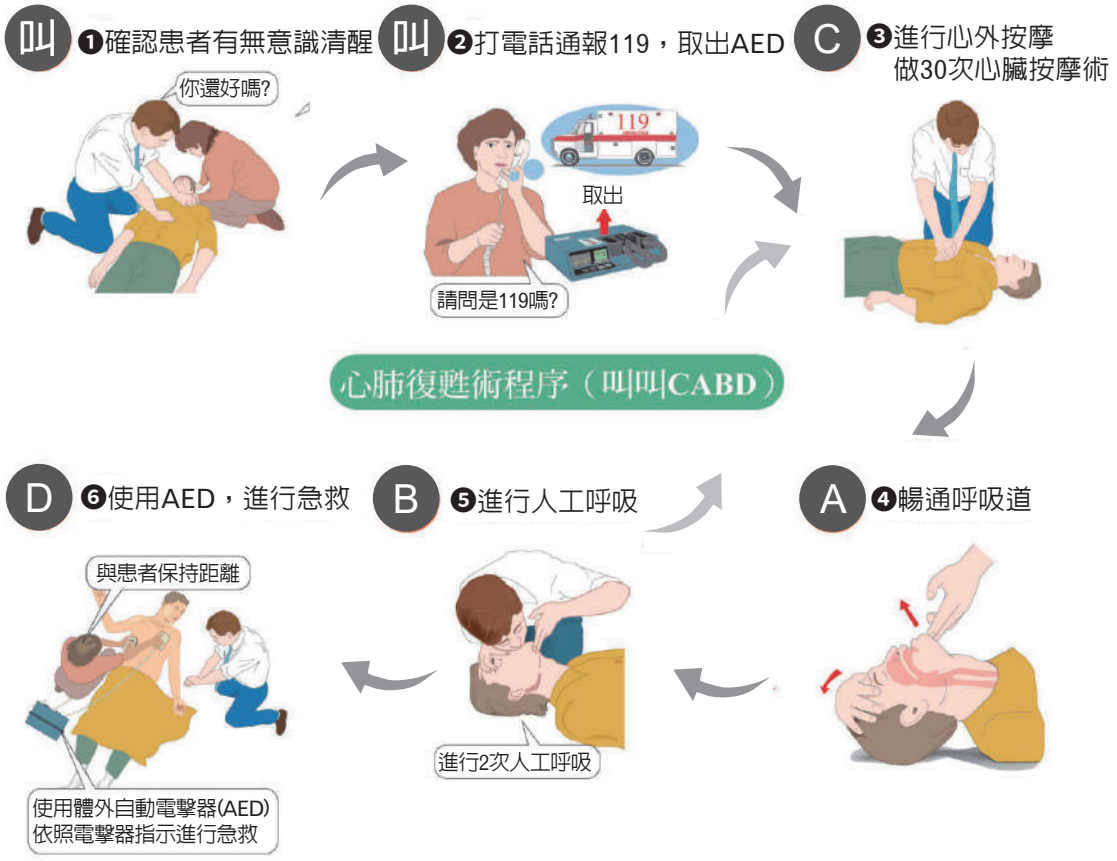


圖 1-4　心肺復甦術程序（摘自苗栗縣政府消防局—救護 e 點靈叫叫 CABD 之急救程序）

消防安全。

火災爲工業安全之意外災害中最嚴重的災害之一，它所造成的災害損失都很大，尤其是人員的傷亡。因此防火、滅火及火災的應變，爲所有人員都必須學習的科目。

▮火災的種類與滅火方法

火災的種類依據火災燃燒物之性質可將火災分爲下列四類，如表 1-1。

▶ 表 1-1 火災分類表

火災類別	火災名稱	燃燒物	適當滅火方法
A 類 (甲類)	一般火災	由一般可燃性固體所引起的火災。如木材、紙張、衣服、塑膠、橡膠等。	使用水或含有大量水分的溶劑滅火最有效。
B 類 (乙類)	油類火災	由可燃性液體或可燃性氣體與油脂類物質所引起的火災，如汽油、機油、燃料油、溶劑、酒精等。	此類火災禁止使用大量的水來灌注滅火，因為油浮於水面上，水的流動會助長火焰的蔓延。正確的滅火方式常用乾化學劑或二氧化碳、泡沫等方式來滅火。
C 類 (丙類)	電器火災	由電力設施、電氣設備所引起的火災，如馬達、開關設備、接線盒、變壓器等。	此類火災滅火必須使用非導電性滅火劑如乾化學劑、二氧化碳等。泡沫及水均不宜使用，因水具有導電性，可能使救火人員觸電，但是如果確定總電源切斷後，可視為甲 (A)、乙 (B) 類火災。
D 類 (丁類)	金屬火災	由易氧化之金屬所引起的火災，如鉀、鈉、鈦、鋁、鎂等。	此類火災滅火需要特種技術並使用特殊金屬化學乾粉的滅火器才能撲滅。
輔助記憶法： 憶猶未盡 (一油電金)(一般火災、油類火災、電器火災燒焦味、金屬火災)			

各類火災因燃燒物不同，可選用的滅火器對照表，如表 1-2。

▶ 表 1-2　各類火災滅火器對照表

適用滅火劑 / 火災類別	水	泡沫	二氧化碳	鹵化烷(乾化學劑)	乾粉		
					ABC類乾粉	BC類乾粉	D類乾粉(特殊金屬化學乾粉)
甲 (A) 類火災	○	○			○		
乙 (B) 類火災		○	○	○	○	○	
丙 (C) 類火災			○	○	○	○	
丁 (D) 類火災							○

▮滅火器的裝設規定

根據我國「各類場所消防安全設備設置標準」第 31 條，滅火器應依下列規定設置：

(1) 設有滅火器之樓層，自樓面居室任一點至滅火器之步行距離在二十公尺以下。
(2) 固定放置於取用方便之明顯處所，並設有以紅底白字標明滅火器字樣之標識，其每字應在二十平方公分以上。但與室內消防栓箱等設備併設於箱體內並於箱面標明滅火器字樣者，其標識顏色不在此限。
(3) 懸掛於牆上或放置滅火器箱中之滅火器，其上端與樓地板面之距離，十八公斤以上者在一公尺以下，未滿十八公斤者在一點五公尺以下。

▮火災的預防

1. 去除可燃物質。
2. 防止高溫及無法散熱的問題。
3. 嚴禁工作人員吸煙。
4. 養成良好工作方法及習慣，慎選慎用高溫設備。
5. 做好電力線路及設備之檢查及維護。
6. 做好防災教育訓練。

▮火災的應變

火災應變能力絕不是與生俱來的，它是經過教育訓練與不斷的演 習才能夠具備的一種能力。當火災發生時才能藉由正確的使用滅火裝 置於第一時間滅火，並能適時的疏散逃生。另外，工場應設置消防組 織，對所有從業人員實施消防訓練，其訓練內容應包括：

1. 消防常識。
2. 各種消防裝置之使用。
3. 逃生與疏散。
4. 依消防任務編組實施演習。
5. 檢討消防裝置是否足夠與汰舊換新。

▮滅火器與消防栓的使用

1. 滅火器的使用

滅火器的使用步驟如圖 1-5 所示：

圖 1-5　滅火器使用步驟圖

2. 消防栓的使用

消防栓的使用步驟如圖 1-6 所示：

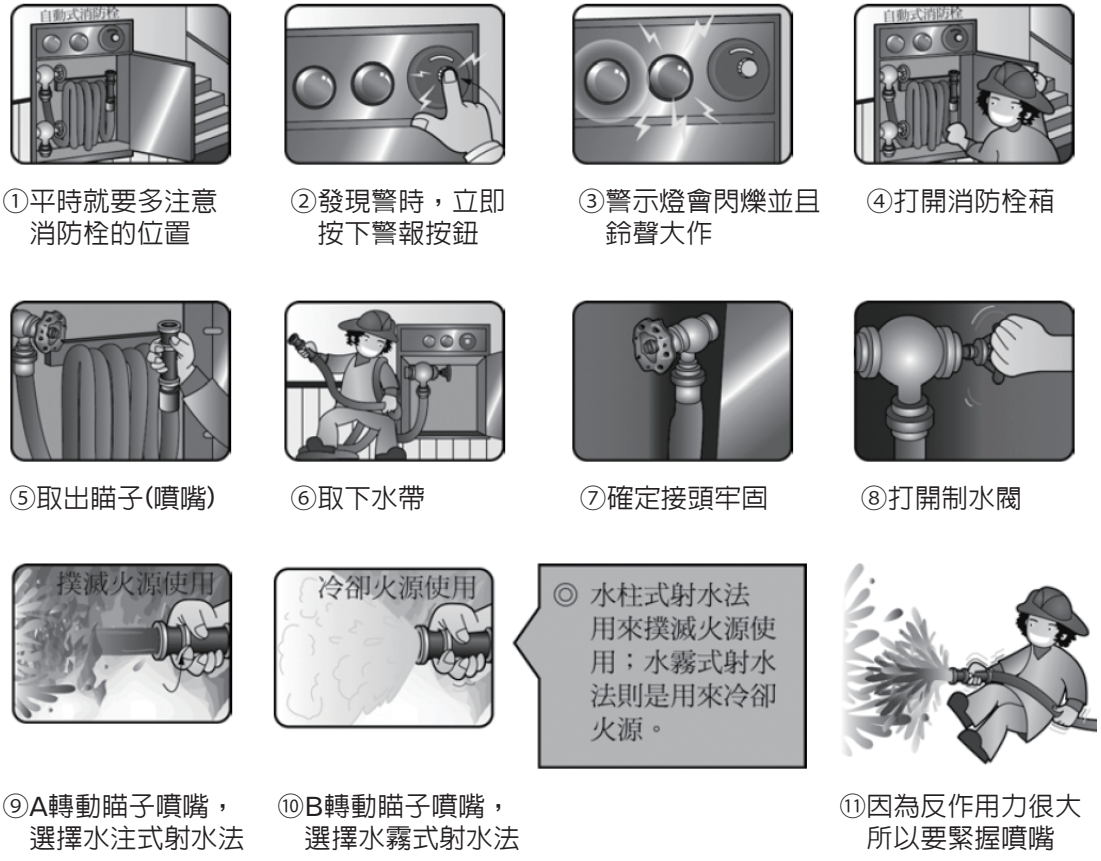


圖 1-6　消防栓使用步驟圖

學後測驗

(　) 1. 工場安全教育其主要且直接的目的是在 (A) 提高生產力 (B) 避免人員與設備受損 (C) 安定勞工生活 (D) 減少投資成本。

(　) 2. 為了防止人員觸電所引起的傷害，下列何者錯誤？ (A) 電器設備均應有接地措施 (B) 手足潮濕，不可碰觸或操作電器設備 (C) 可以用手指測試線路或電源是否有電 (D) 危險的電力設施要有安全標誌，並有適當的限制接近設施。

(　) 3. 下列敘述中何者是正確的？ (A) 在工場中實習可以穿著寬鬆的衣服比較舒服 (B) 工場應保持通風良好，光線充足 (C) 在工場中無需保持整齊、清潔 (D) 學生在工場實習中可不必配戴個人防護器具。

(　) 4. 下列對工業用標示顏色所代表之意義，何者為錯誤？ (A) 紅色表示防火設備、禁止 (B) 黃色表示注意、警告 (C) 綠色表示安全、救護設備 (D) 藍色表示放射性危險。

(　) 5. 下列因素中何者不是由不安全的個人因素所造成的意外事件？ (A) 粗心 (B) 精神狀態不佳 (C) 不使用防護器具 (D) 噪音。

(　) 6. 燙傷時的處理程序為 (A) 叫叫 CABD (B) 沖脫泡蓋送 (C) 牙膏與醬油塗抹 (D) 立即刺破水泡

(　) 7. 下列哪一類火災是電氣火災？ (A) 甲 (A) 類 (B) 乙 (B) 類 (C) 丙 (C) 類 (D) 丁 (D) 類。

(　) 8. 使用滅火器應站在何處？ (A) 上風 (B) 下風 (C) 側風 (D) 逆風。

(　) 9. 下列何種滅火器不適用於撲滅丙 (C) 類電氣火災？ (A)ABC 乾粉滅火器 (B) 泡沫滅火器 (C) 二氧化碳滅火器 (D) 鹵化烷滅火器（俗稱海龍，Halon）。

(　) 10. 一般 18 公斤以下滅火器，放置的高度不得超過多少公尺？（滅 火器上端與樓地板之距離）(A)1 公尺 (B)1.5 公尺 (C)2 公尺 (D)2.5 公尺。

MEMO

Chapter
02

單晶片微處理機實習儀器認識及實作

學習大綱

- 2-1　單晶片微處理機的認識
- 2-2　基本內、外部結構
- 2-3　單晶片微處理機應用
- 2-4　實習儀器

學習目標

- 1. 能認識單晶片微處理機。
- 2. 能瞭解單晶片微處理的內、外結構。
- 3. 能瞭解單晶片微處理機的應用。
- 4. 能瞭解本課程所需儀器。

2-1 單晶片微處理機的認識

微處理機

微處理機是由硬體和軟體所組成，硬體是指有形的電路；軟體則指用來控制微處理機工作的指令和程式。微處理機的硬體結構包括了五個主要單元，分別是輸入單元、輸出單元、記憶單元、算術邏輯單元和控制單元，其中算術邏輯單元和控制單元又可合併成爲一個單元，稱爲中央處理單元 central processing unit），就是所稱的 CPU，這五個單元的關係如圖 2-1 所示：

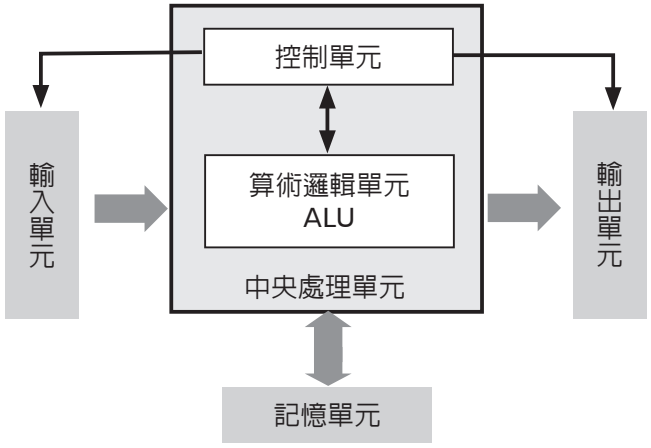


圖 2-1 微處理機架構圖

以上微處理機五大單元，分別說明如下：

1. **輸入單元**（input unit, IU）：輸入單元的工作是將外來的資訊送到 CPU 處理或直接存入記憶單元。 個人電腦上常用的輸入設備有讀卡機、磁碟機、鍵盤、滑鼠和掃描器等。在微處理機實習上常見的輸入元件有指撥開關、按鈕開關、極限開關、鍵盤、光感測器、溫度感測器、濕度感測器、聲音感測器、距離感測等各種物理量感測器。

2. **輸出單元**（output unit, OU）：輸出單元負責將 CPU 處理過的資料輸出或儲存。個人電腦（Personal Computer）上常用的輸出設備有顯示器、列表機和繪圖機、硬碟、光碟等。 在微處理機實習上常見的輸出元件有發光二極體 LED、液晶顯示器 LCD、七段顯示器、步進馬達、直流馬達、喇叭、蜂鳴器等。

3. **記憶單元（memory unit, MU）**：記憶單元是用來存放系統運作所需的程式或資料，一般來說，單晶片微處理機的程式是儲存在唯讀記憶體中，資料儲存在隨機存取記憶體中。
4. **算術邏輯單元（arithmetic logic unit, ALU）**：算術邏輯單元可將送來的資料執行算術運算（如加、減、乘、除等）及邏輯運算（如 AND、OR、NOT、XOR 等），在運算完成後，再由控制單元將資料送回記憶單元存放，或直接送到輸出單元。
5. **控制單元（control unit, CU）**：控制單元是微電腦的指揮中心，負責協調和指揮以上各單元之間的資料傳送及運作，使得微電腦可以按照人們的要求完成工 作。

單晶片微處理機

單晶片微控制機（microcontroller unit，簡稱 MCU）是將上述五大單元、以及一些周邊電路（計時 / 計數器、燒錄介面、類比 / 數位信號轉換）都整合在一塊晶 片上的微型電腦。傳統的工業控制都是由硬體電路所構成，以馬路上紅綠燈爲例，燈號時序的轉換要靠順序邏輯電路的設計；但是單晶片微控制機的控制只要完成輸入與輸出電路就好，所有燈號時序的轉換控制都改爲軟體時序陣列值指派，請看第 5 章範例 5-1a。所以單晶微處理機的控制，具有體積小、耗電低，可節省開發時間與成本，目前已經是所有工業控制的主流。

2-2 基本內、外部結構

Arduino 開發板

目前常聽到的 macro:bit、8051、Arduino 等都是單晶片微處理機開發板，macro:bit 比較著重在圖形開發介面，主要是來讓小學生玩簡單的聲光控制遊戲；8051 與 Arduino 都是使用文字式的開發工具，是眞實的嵌入式工業晶片，可眞實改善許多生活與工業控制，適合中學生以上與社會人士學習嵌入式系統控制。但 8051 已經過時，漸漸被 Arduino 取代，Arduino 官網是 www.arduino.cc。Arduino 之所以能異軍突起，主要是它主張開源，它的軟硬體都是開放，官網有非常詳盡的軟硬體資料，是學習單晶設計最好的入門資料，使用者可以站在巨人的肩膀，繼續接力開發新產品。其次，Arduino 輸出電流變大，可以直接驅動 LED，使得產品的電路也變的簡單。還有，Arduino 腳位也變多，這樣可以直接驅動一些需要重複掃描輸出的元件，例如，本書要介紹的四位數七段顯示器與點矩陣 LED 驅動電路也變簡單了。Arduino 依照使用者不同的需求，開發很多版本的微控板，其中型號 Mega 2560 的全部 I/O 腳位共有 70 隻，因爲腳位多、電流也夠大，可直接驅動本書介紹的四位數七段與點矩陣 LED，這樣電路與軟體最爲省事，本書就選用 Mega 2560，下表是其技術規格（摘自 Arduino 官網）。

OVERVIEW | TECH SPECS | DOCUMENTATION

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm

外部結構

圖 2-2 是其微控板實體照片（摘自 Arduino 官網），中間黑色正方形 IC 就是 ATmega 2560 單晶片，旁邊共有 70 隻 I/O 腳位（數位 54 隻與類比 16 隻），這些接腳都可以使用軟體的設定，分別指派當作數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，這會在本書陸續介紹。

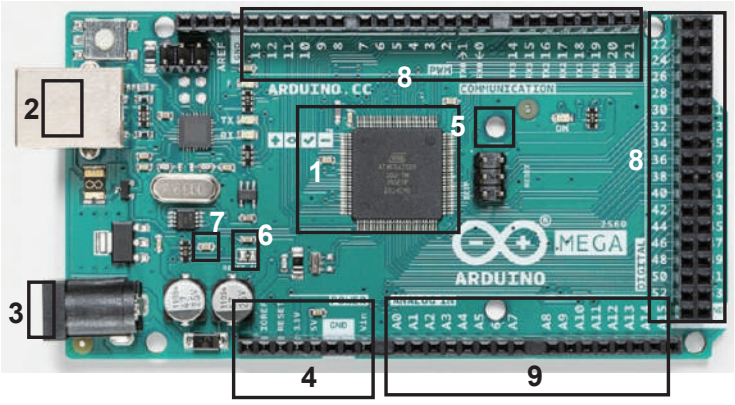


圖 2-2　Arduino Mega 2560 實體圖

1. ATmega 2560 **單晶片**：ATmega 2560 是本微控板的單晶片，其詳細資料如圖 2-3（摘自 Arduino 官網）。

Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V

8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash

DATASHEET

Features

- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-Chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C

圖 2-3　AT mega 2560 特性圖

以上資料是「微處理機」課程闡述的重點，請讀者自行對照課本，並與家用電腦的微處理機規格做比較。

2. **USB 連接器**：此連接器除提供燒錄程式以及監控程式之用外，還被當成開發板之電源連接器，可提供 +5V/0.5A 之電源，而此電源足以驅動整塊開發板。使用者外接電路若短路，或使用電流超過 0.5A，微控版具有自我保護電路，將自動切斷該 USB，不會損害到個人電腦。只要拔開 USB 線，待排除故障後，USB 線再重新插入，即可重啓 USB 之功能，並提供電源。
3. **外接電源與電源穩壓電路**：可使用 +7V ～ +12V 之電源變壓器做爲輸入電壓，而由左下方之連接器引入，經內部電源穩壓電路而得到 +5V 電源。本書的所有實驗電路，USB 供電就綽綽有餘，若是一些較耗電的馬達裝置，才需由此裝置提供額外電源。
4. **電源連接器**：由右而左分別爲 Vin、GND、GND、+5V、+3.3V、RESET、IOREF、與一個空腳，Vin 端可接受外部提供的電源 (+7 ～ +12V)，而這個電源將經過內部穩壓電路而取得 +5V 電源，以供整塊開發板之用。兩個 GND 端提供外部裝置接地之用，+5V 端與 +3.3V 端分別提供外部裝置 +5V 與 +3.3V 電源。RESET 端提供外部重置之用 (低態重置)，IOREF 端提供外部裝置之參考電壓，在此爲 +5V(內部已連接到 +5V 端)。
5. **Reset 按鈕**：此按鈕可重置微處理機。
6. **電源指示 LED**。
7. **內建輸出 LED**：此 LED 連接腳位 13，可讓使用者插入開發板後，即刻進行開發板自我測試，測試與自己電腦是否安裝完成。
8. **數位 I/O 接點**：腳位編號 0 到 53 共 54 個數位 I/O 接點，每個接腳都有印出編號，可用此編號指派這些接點工作。其中編號 23、25、27、29 因爲電路板造型裁切問題，沒有印出。
9. **類比輸入接點**：右下方的 A0 ～ A15 接腳除可做爲通用的數位輸出入埠外（功能同以上 54 支數位接腳），也提供類比信號輸入之用，而這 16 支類比輸入腳具有 10 位元類比 / 數位轉換功能。

自我練習

1. 以上 ATmega 2560 單晶規格是「微處理機」課程的重點，請參考「微處理機」教材內容，將專門名詞中文化，並說明其用途。
2. 請將 ATmega 2560 單晶規格與家用電腦規格作一個比較表。

內部結構

MEGA2560 控制板內部單片片採用 ATmega 2560，其電路架構如圖 2-4：

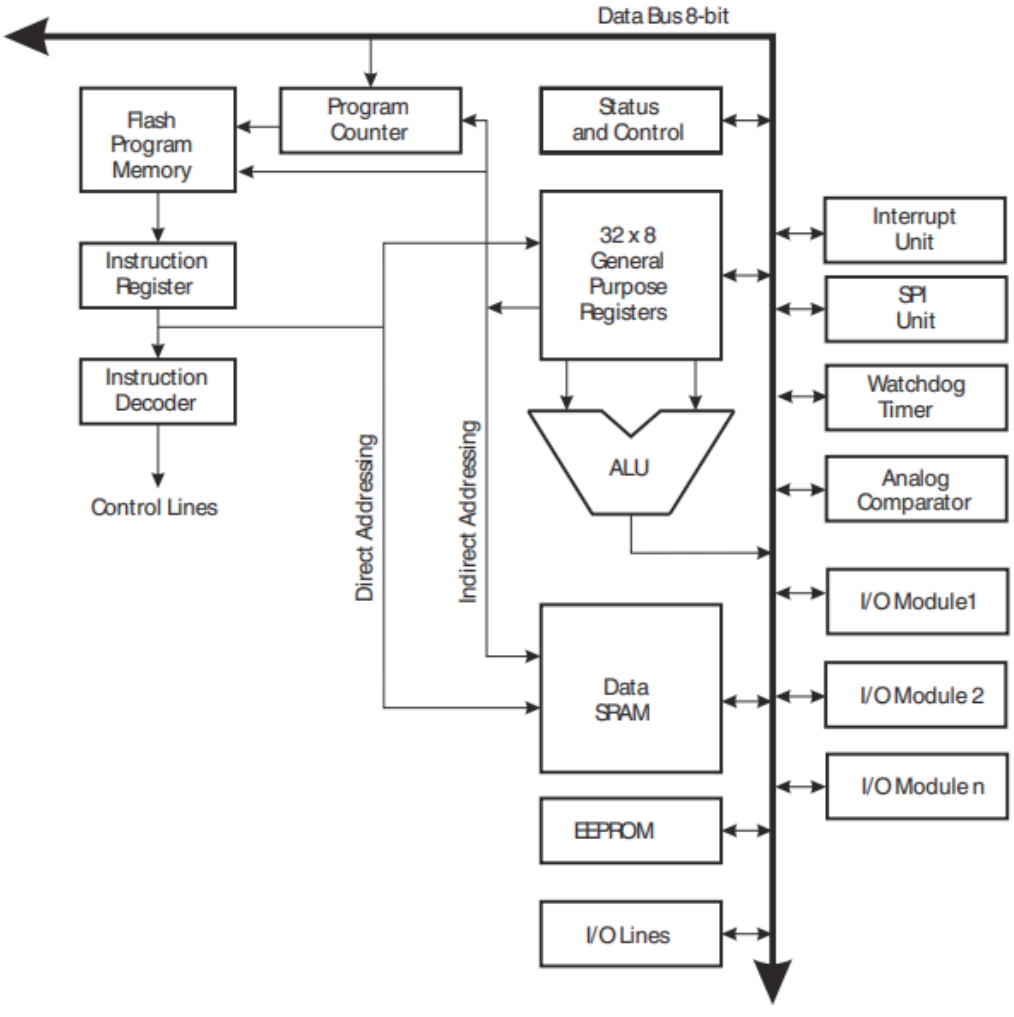


圖 2-4　AT mega 2560 架構圖（摘自 Arduino 官網）

自我練習

1. 圖 2-4 的單元名稱，例如，Flash Program Memory,Instruction,Instruction Decoder,Program Counter ,Status and Control 等是「微處理機」課程重點，請翻閱「微處理機」教材，寫出其中文名稱與功能。

   圖 2-5 則是內外部連結架構圖：

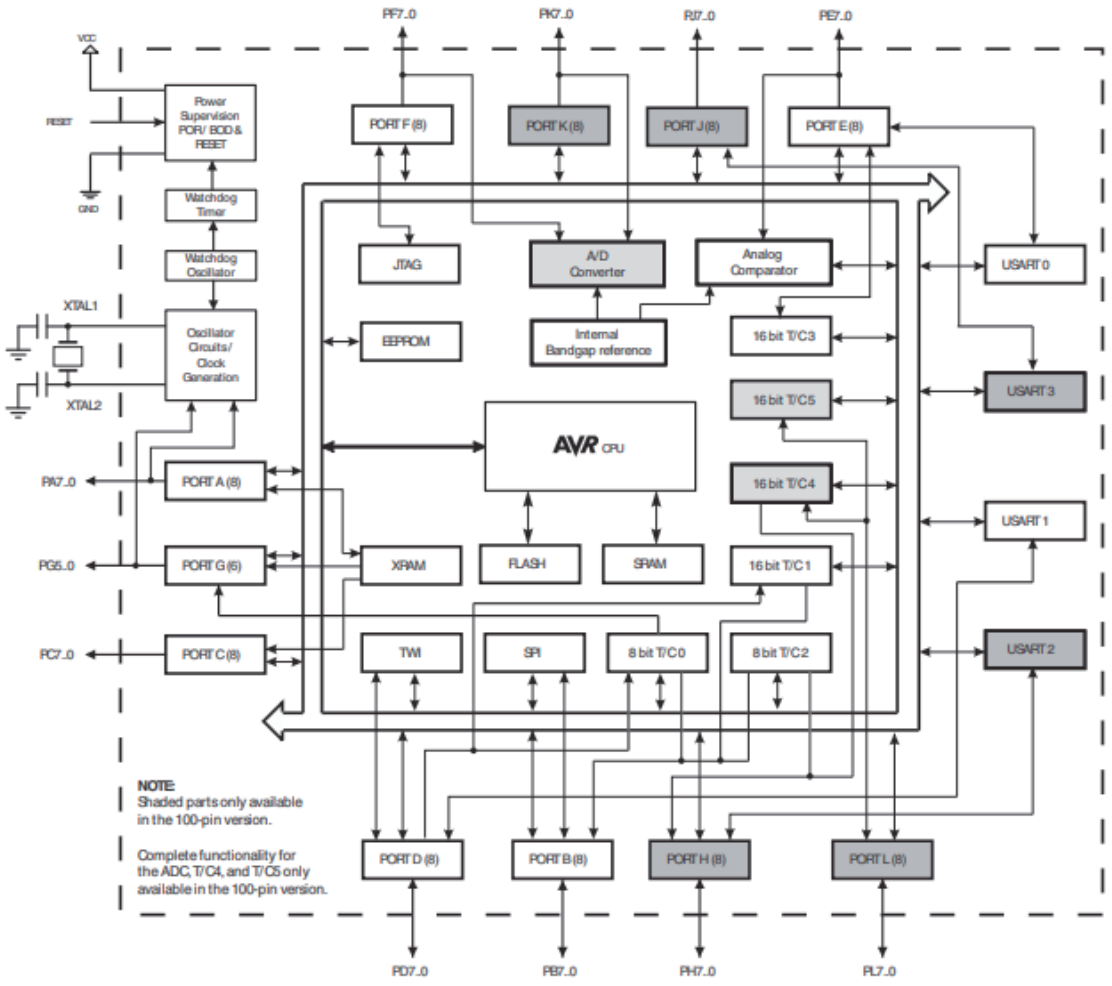


圖 2-5　AT mega 2560 內外部連結架構圖 (摘自 Arduino 官網)

自我練習

1. 圖 2-5 的單元名稱，例如，Power Supervision, Watchdog Timer, Watchdog Oscillator, Oscillator Circuits / Clock Generation 等是「微處理機」課程重點，請翻閱「微處理機」教材，寫出其中文名稱與功能。

2-3 單晶片微處理機應用

現代單晶片微處理機的應用可說與我們日常生活息息相關，生活中的電器產品到處充滿單晶片微處理機的應用，家裡的電視、冰箱、空調、風扇、小至無線耳機、無線滑鼠等、汽車的 ABS 防鎖死剎車系統、TRC 循跡防滑控制系統、EBD 電子剎車力分配系統、倒車雷達、胎壓偵測等等等都是單晶片微處理機應用。以汽車與家用冷氣爲例，以前的冷氣都是吹到一定的溫度，然後送風，等到溫度上升，冷氣又啓動，這樣當然冷冷熱熱不是很舒適，現在的冷氣則都是直流變頻恆溫控制系統，馬達的轉速都是依照所設定的溫度控制轉動，整個過程都是穩定持續運轉，溫度非常穩定，不會忽冷忽熱。又例如，現在的晶片可說越作越小，以小米穿戴式手環爲例，小小的手環功能就包括時間、GPS 定位、全天候心率監測、血氧飽和度監測、睡眠監測、壓力監測、呼吸訓練、女性健康管理等功能。

2-4 實習儀器

單晶片微處理機實習所需儀器如下：

1. 微電腦或筆電。微電腦或筆電可編輯、編譯、燒錄程式到單晶片微控板。
2. 單晶片微控板。本書採用 Arduino Mega 2560 微控板，使用者將 I/O 裝置接在微控板提供的腳位，然後寫程式控制這些腳位電位的變化，即可完成很多工業與家庭控制系統。
3. 三用電表。三用電表用來檢測輸出入電壓與判別電子零件的好壞。
4. I/O 電子零件。本書所使用電子零件都是標準零件，每節開頭都有零件清單，所有零件可單獨購買，再用麵包板組裝。
5. 實驗教具。本書所有電路也開發成實驗教具，如圖 2-6，所有電子零件與 Arduino 都作在一塊實驗板，使用者可用杜邦線讓電子零件與 Arduino 腳位連結而完成本書電路。使用教具的優點是合乎環保議題，所有零件可重複使用，老師與學生上課不用爲張羅零件耗費心力與時間。

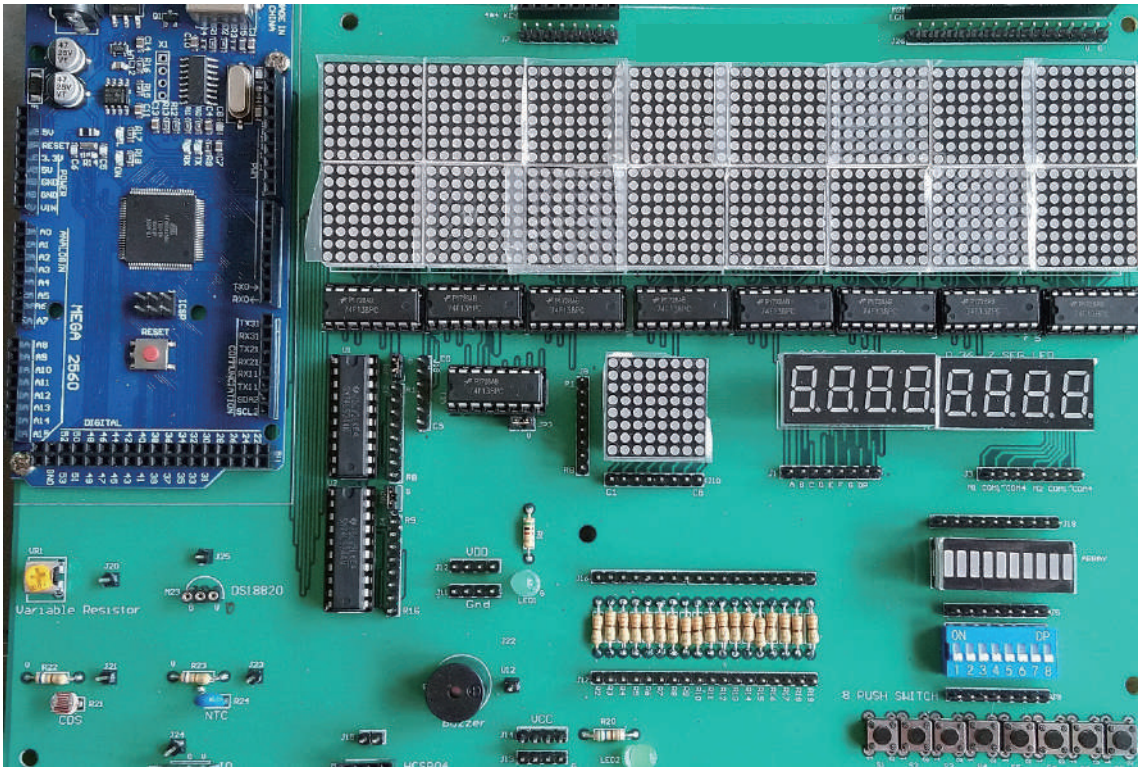


圖 2-6　本書實驗板實體圖

6. Arduino 魔法教具如圖 2-7。圖 2-6 的實驗教具電子零件與 Arduino 獨立放在實驗板，使用者必須自己用杜邦線連接 Arduino 與電子零件，但是初學單晶微處理機的最大的困擾是，當錯誤發生時，根本不知道是硬體錯還是軟體錯。所以本書也將所有電路預接完成，製作 Arduino 魔法教具，如圖 2-7Arduino，特色是所有電子零件與 Arduino 的連接線都已經按照本書的電路接線完成，其電路請看附錄 1，使用者可直接鍵入本書教材的所有程式，就都有執行成果，有如變魔術，所以稱爲 Arduino 魔法教具。優點是硬體接線已完成，可先專注在軟體的學習，當軟體學成後，再練習接線，以上循序漸進的教具，可降低學習的複雜度。

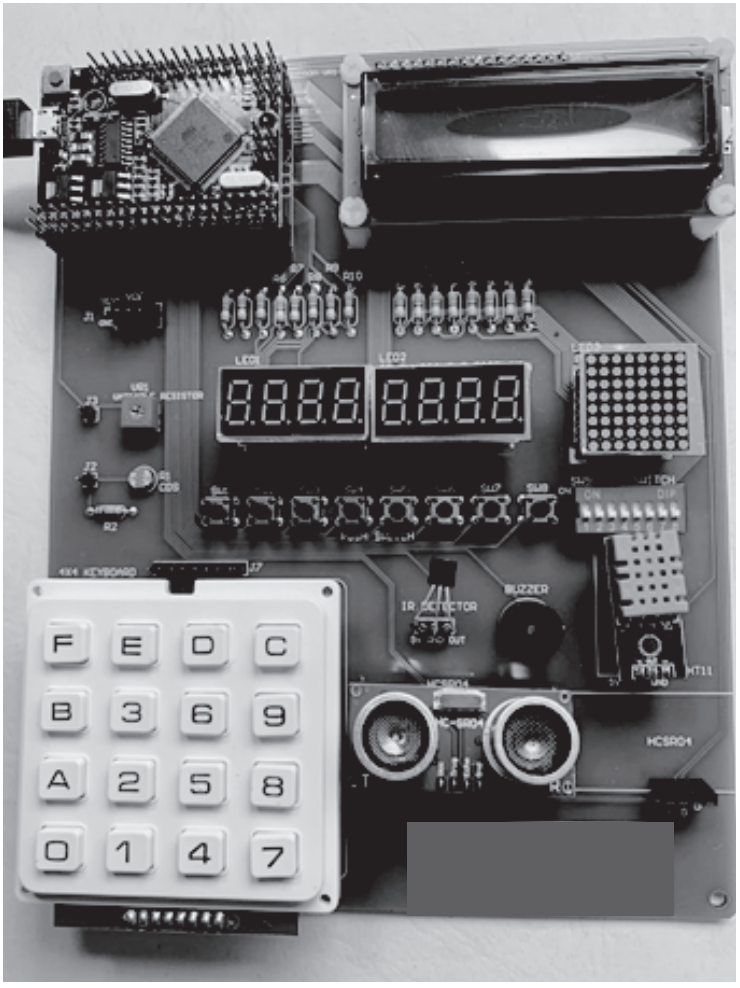


圖 2-7　Arduino 魔法教具實體圖

學後測驗

1. 請寫出微處理機五大單元？_____，_____，_____，_____，_____
2. Arduino 數位接腳共幾支？_____類比接腳共幾支？_____
3. 寫出 Arduino 的接腳可指派為哪三種功能？_____，_____，_____
4. Arduino 接腳高電位時，最大輸出電流為何？_____
5. Arduino 接腳低電壓時，最大可流入電流為何？_____
6. Arduino 接腳輸出高電壓時，電壓為多少？_____

Chapter
03

單晶片微處理機開發流程

學習大綱

- 3-1　高階程式開發流程
- 3-2　程式編輯、編譯及連結
- 3-3　程式的模擬、除錯與燒錄
- 3-4　I/O 腳位探索

學習目標

- 1. 能瞭解高階程式開發流程。
- 2. 能夠完成程式編輯、編譯及連結。
- 3. 能完成程式的除錯與燒錄。
- 4. 能認識微處理機的 I/O 腳位。

3-1 高階程式開發流程

以往開發單晶片微處理機程式僅能使用低階組合語言，要先學習微處理機的暫存器、記憶體與組合語言，現在 Arduino 則使用 C 語言語法搭配 I/O 函式開發產品，所以學習單晶片微處理機開發流程如圖 3-1：

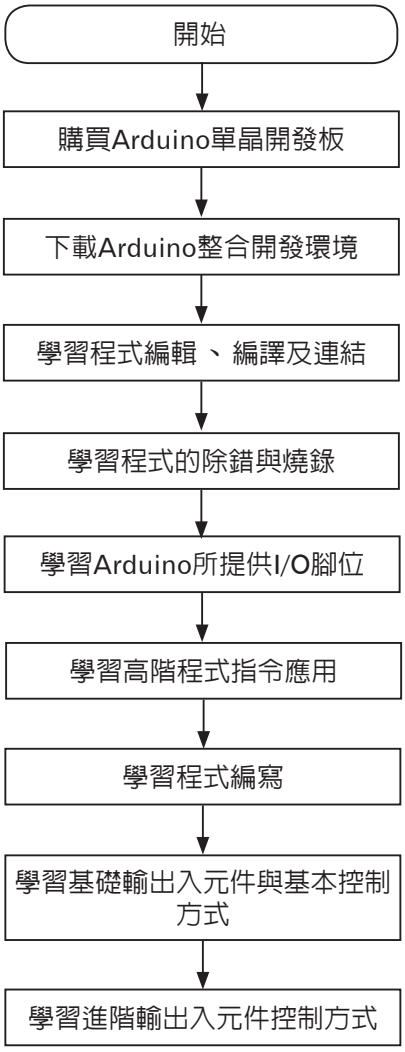


圖 3-1 單晶片微處理機開發流程圖

1. 購買 Arduino 單晶開發板，請看本書第 2 章。
2. 下載 Arduino 整合開發環境，請看本書 3-2 節。
3. 學習程式編輯、編譯及連結，請看本書 3-2 節。
4. 學習程式的除錯與燒錄，請看本書 3-3 節。
5. 學習 Arduino 所提供 I/O 腳位，請看本書 3-4 節。
6. 學習高階程式指令應用，請看本書 4-1 節。
7. 學習程式編寫，請看本書 4-2 節。
8. 學習基礎輸出入元件與基本控制方式，請看本書第 5 章。
9. 學習進階輸出入元件控制方式，請看本書第 6 章。

以上是微處理機高階程式開發流程，微處理機應用實例開發流程將在範例 5-1a 介紹。

3-2 程式編輯、編譯及連結

Arduino軟體下載與安裝

以往所有單晶片的程式開發，都要自備類似記事本的編輯器，編寫程式、存檔、離開，然後使用其所提供的編譯程式，若有錯誤則要繼續開啓記事本修改，再編譯，直到完全正確，然後連結若干目的程式成爲一個完整程式。最後還要購買萬元燒錄器，將程式燒到單晶片。但是 Arduino 就方便了，有免費整合編輯視窗，整合以上編輯、編譯、燒錄（上傳）於單一視窗，此稱爲整合編輯程式（IDE, Integrated Development Environment 的縮寫）。請於 Arduino 官網點選『SOFTWARE/DOWNLOADS』，畫面如圖 3-2，請點選『Windows win10 and newer, 64bits』，即可下載最新安裝執行檔(*.exe)。接著，請開啓檔案總管，至下載區按兩下該執行檔，即可安裝。

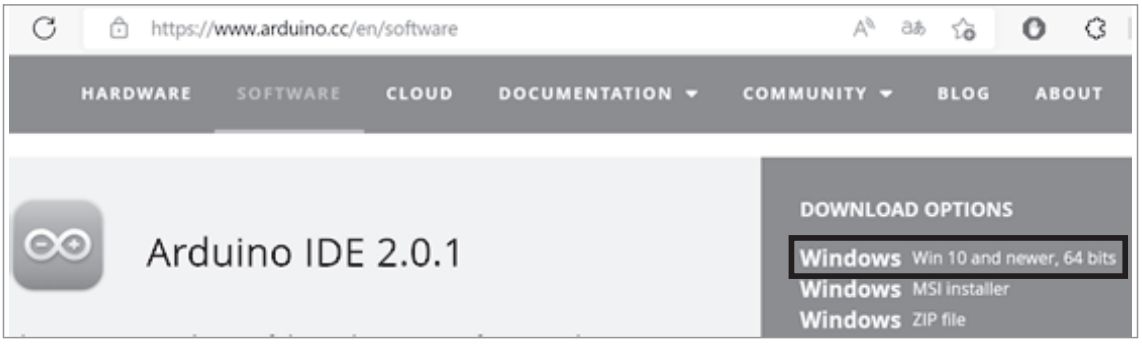


圖 3-2 Arduino 下載點

IDE畫面

圖 3-3 是完成安裝且開啓 Arduino IDE 畫面。

圖 3-3　Arduino IDE 畫面

單晶微控板使用步驟

Arduino 的單晶微控板使用步驟如下：1. 插入微控板、2. 點選開發板型號、3. 點選通訊埠編號、4. 取得開發板資訊，以上動作分別說明如下：

1. **插入微控板**。
   請依照指示，將微控板 USB 插頭插入電腦或筆電 USB 插座。請留意微控板右上角電源指示燈是否亮起。
2. **點選開發板型號**。
   請點選功能表的『工具 / 開發板』即可點選您所使用的微控板型號。請依照您的微控板型號點選，本書使用 Arduino Mega2560，所以是點選『Arduino/Genuino Mega or Mega 2560』，若您是 UNO 板，也是在此點選，因爲不同版本腳位數量不一樣，選錯了就無法正確編譯程式。

3. **點選通訊埠編號**。
   請點選功能表的『工具 / 序列埠』即可點選您所使用的序列埠編號（備註：系統會出現可用編號 com3 或 com4,5,6 等等，讓您點選，有時候會同時出現很多個編號 com3,com4,com5…，請按照順序嘗試點選，直到可上傳 (或稱燒錄) 爲止。其次，有些電腦不會自動抓到通訊埠 (com3、com4…)，請到網路搜尋與下載『CH34x-install-Windows』，並安裝，直到出現通訊埠。）

4. 取得開發板資訊

請點選功能表的『工具 / 取得開發板資訊』即可取得您的微控板資訊。若出現下圖，才表示以上安裝與設定就緒，才能上傳程式。（實驗的中途，若改插入別人的微控板，也要重覆以上步驟，直到看到下圖，才表示有抓到此微控板，才能上傳程式。其次，圖 3-4a 是原廠的微控板，若不是原廠，那可能就像圖 3-4b，顯示『未知的開發板』。）

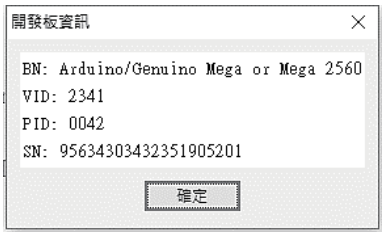


圖 3-4a　原廠開發板資訊

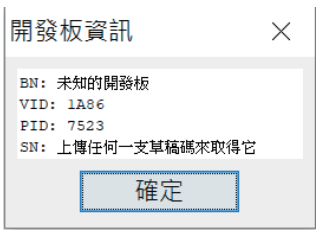


圖 3-4b　非原廠開發板資訊

自我測試

整合開發環境安裝後，內部也含一個自我測試程式，這樣就可以測試此微控板是否已經安裝完成。此自我測試程式使用微控板內植 LED（腳位編號是 13，別名是 LED_BUILTIN）使其明亮 1 秒，熄滅 1 秒。其次，進行單晶片控制都要這樣一步一步來，當問題發生時，才能一步一步除錯，逐漸縮小錯誤範圍，並排除故障。以下說明如何自我測試：

1. 開啓自我測試程式。

圖 3-5 是開啓測試程式 Blink（請點選功能表的『檔案 / 範例 /01.Basics/Blink』），其功能是直接使用微控板預植的 LED（不管什麼板 UNO、MEGA…等），都是腳位 13，以常數『LED_BUILTIN』表示），並令其亮滅各一秒。

```
/*
  Blink
  Turns an LED on for one second, then off for one second, repeatedly.
  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
*/
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

圖 3-5　Arduino 自我測試程式

以上程式，「/*　*/」之間的文字稱爲註解，此註解僅給人看，微處理機不會處理，剩下的程式如下：

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is
                                      the voltage level)
  delay(1000);                     // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by
                                      making the voltage LOW
  delay(1000);                     // wait for a second
}
```

2. **驗證程式**。

按一下工具列的『驗證』按鈕 ，即可編譯程式。

3. 上傳程式

按一下工具列的『上傳』按鈕  ，即可上傳程式到微控板（上傳又稱爲燒錄）。上傳結束將會自動執行程式，請觀察微控板上所預植 LED 是否亮滅（此顆 LED 就在腳位 13 旁）。

補充說明

1. setup() 函式僅在程式一開始時執行一次。所以通常放置執行程式的初始設定、或程式執行後只執行一次的指令。
2. pinMode(LED_BUILTIN, OUTPUT); 是指派編號 13 這隻腳的功能是輸出。
3. loop() 函式則會重複不斷的被執行，所以就放置一些需要反覆執行的指令。
4. digitalWrite(LED_BUILTIN, HIGH); 是指派腳位 13 爲高電位，請用三用電表量其電壓，將會有 5V，這樣就可以讓 LED 發光。
5. delay() 函式是程式延遲程式，單位是毫秒 ms，千分之一秒。因爲指派 LED 亮，總要停留一點時間，使用者才有機會看到。請自行調整時間，並觀察執行結果。
6. digitalWrite(LED_BUILTIN, LOW); 是指派腳位 13 爲低電位，請用三用電表量其電壓，將會有 0V，這樣就可以讓 LED 熄滅。
7. 雙斜線『//』稱爲註解，此爲單列註解，僅給人看，微處理機不予編譯，沒有鍵入也沒關係。其次『/*』與『*/』之間的文字，也都是註解，此稱爲多列註解，這些註解都是給人看的，微處理機不會編譯。

自我練習

1. 請自行修改程式，使得 LED 亮 2 秒，且熄滅 0.5 秒。
2. 請自行修改程式，使得 LED 亮 3 秒、熄滅 1 秒、亮 2 秒、熄滅 0.5 秒、亮 1 秒、熄滅 0.2 秒。

線上使用手冊

點選 Arduino IDE 功能表的『Help/Reference』畫面如圖 3-6（電腦請先連線），此爲 Arduino 所提供的軟體指令參考手冊，不論指令分類、指令解說、範例、函式等解說都很詳細。

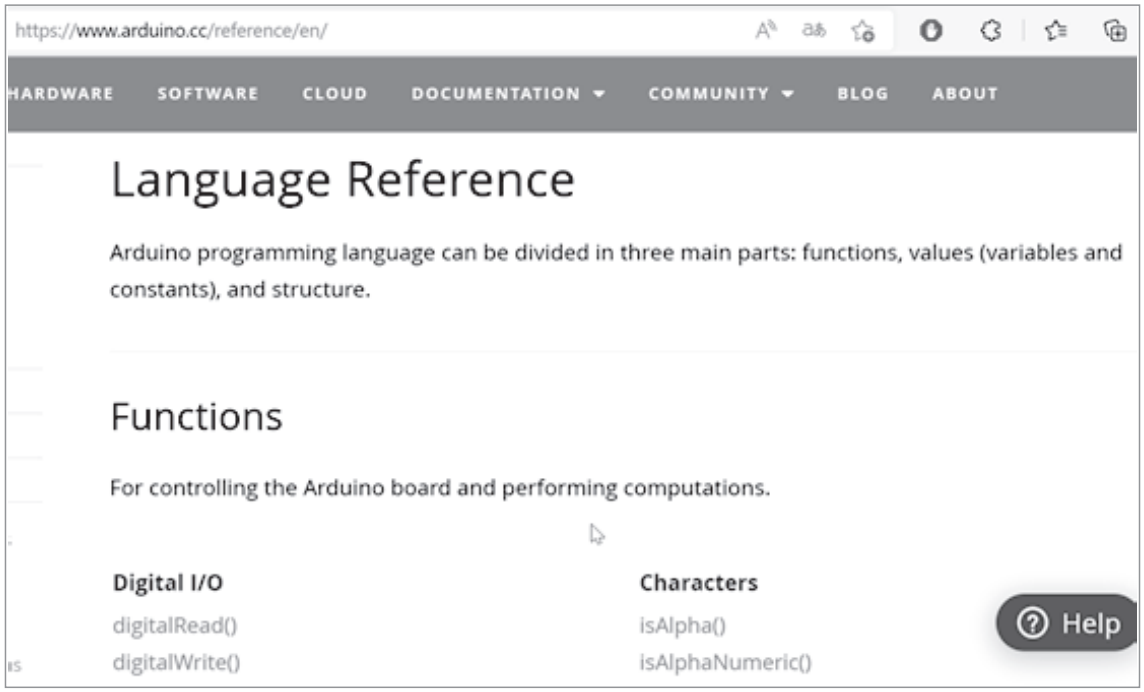


圖 3-6　Arduino 指令、函式參考文件

點選圖 3-6 的 digitalWrite()，畫面出現如圖 3-7。

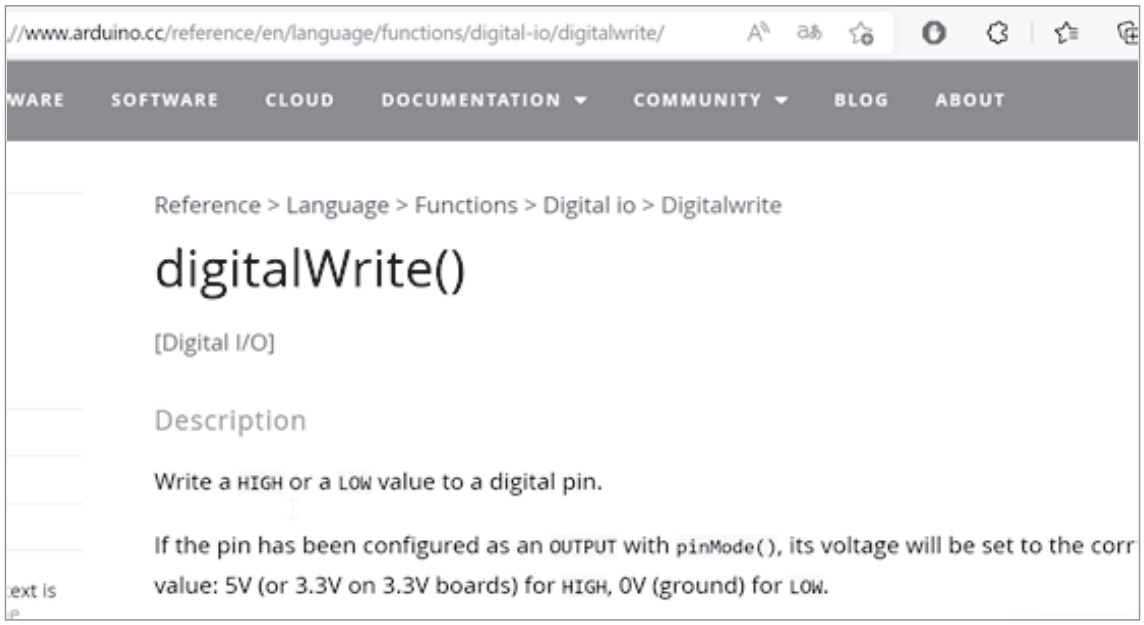


圖 3-7　digitalWrite() 說明畫面

3-3 程式的模擬、除錯與燒錄

程式模擬

以往微處理機使用**可程式唯讀記憶體**（Programmable ROM，PROM）燒錄程式，每個 PROM 僅能燒錄 1 次，所以微處理機廠商都有提供模擬程式，先模擬結果，等到程式都如預期再行燒錄。但是現在 PROM 已經進化爲可重複使用，先是進化使用紫外線抹除的 EPROM，目前則是進化到電子式無限次數抹除的 EEPROM，所以 PROM 成本降低。又因爲 Arduino 提供的序列埠視窗可以先使用電腦的螢幕直接觀察微處理機的執行過程，此一工具不僅可以學習微處理機程式，也可以直接觀察執行結果，比以住的模擬程式完善，所以模擬程式已經沒有存在的必要，Arduino 已經不提供模擬程式。

程式除錯

程式除錯是所有程式設計者的惡夢。程式設計常見的錯誤是指令拚字錯誤、演算法錯誤等，分別說明如下：

▮拚字錯誤

拚字錯誤是初學者最容易發生的錯誤，但是現在很多整合開發環境已經將程式解析，給予提示，能將資料型態、常數、函式、指令解析給予不同的顏色。下圖左是 Arduino 的程式指令解析畫面，共有四種顏色，常數與資料型態解析爲綠色，函式解析爲紅色，指令解析爲灰色，剩下黑色則爲使用者自訂識別字如圖 3-8。所以當鍵入程式並打完資料型態、函式、或指令時，若沒有變色，此時就要馬上更正，直到指令變色爲止。

```
Blink
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HI
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
  delay(1000);                       // wait for a second
}
```

圖 3-8　編譯器指令解析圖

▮演算法錯誤

若程式編譯過程沒有錯誤訊息，結果卻錯誤，此時就要一步一步執行與觀察變數的變化。Arduino 提供序列埠監控視窗用來觀察程式執行過程，是學習程式設計、與除錯程式的最佳工具，請看以下介紹。

序列埠監控視窗

Arduino 的序列埠監控視窗如圖 3-9 所示:(點選『工具』/『序列埠視窗』)

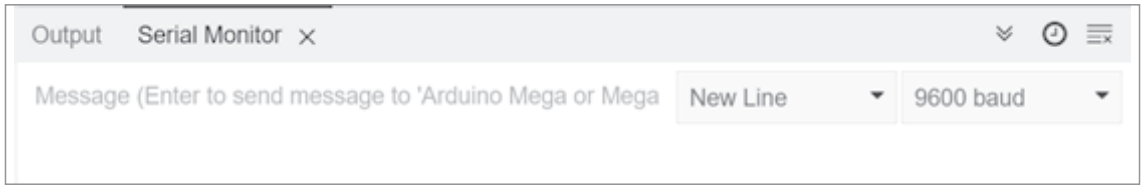


圖 3-9　序列埠視窗

它可以讓我們在程式開發階段，用電腦的鍵盤與螢幕和微處理機雙向溝通。此一功能是以往其他微處理機所沒有的功能，因爲透過此序列埠監控視窗，不僅可以學習 Arduino 語言，也可直接看到程式的執行結果，或在程式執行過程中，將變數內容先行輸出，達到協助除錯的功能，這樣比模擬效果好，可完全取代程式模擬，所以 Arduino 就不提供程式模擬的功能。例如，以下程式可以學習 Arduino 的運算子與數學函式，因爲程式執行後，可在序列埠視窗出現執行結果，如圖 3-10。

```
void setup() {
    Serial.begin(9600);
    Serial.println(4+2);
    Serial.println(4>=2);
    Serial.println(pow(2,3));
} void loop() {}//不可省略
```

```
6
1
8.00
```

圖 3-10　序列埠輸出結果

其次，於開發 Arduino 程式中，從感測器所讀到的值、或運算結果要從輸出元件輸出前，都可將結果先行輸出，先行測試此段程式邏輯與硬體接線是否正確。例如，以下程式，可以將指撥開關的輸入值，先行輸出於電腦螢幕的序列埠視窗，因此可以確認此部分硬體接線是否正確。

```
a=digitalRead(37);//讀取腳位37的電壓值
Serial.println(a);//於序列埠視窗輸出電壓值
```

以下程式，可以先將運算結果先行輸出，這樣可以檢查此軟體運算思維是否正確，然後再輸出於 PORTF。

```
b=a+3//處理
Serial.println(b);//於序列埠視窗輸出電壓值
PORTF=b;
```

Serial物件

電腦或筆電用來編輯微處理機程式，而將程式燒錄微處理機之後，電腦也可以被微處理機控制，電腦的鍵盤、螢幕也可作爲微處理機的 I/O 介面，此功能是以往微處理機所沒有的。要讓電腦的鍵盤與螢幕成爲微處理的輸出入介面，可使用微處理的 Serial 物件。請開啓 Arduino 參考文件 (點選『Help』/『Reference』/『Serial』)。Serial 的 Function 如圖 3-11 所示，這些方法可以讓我們啓用序列埠，然後進行輸出文數字、輸入字元、輸入字串與數值，分別說明如下。(補充說明：以下這些函式，在舊式函式導向程式設計稱爲『函式』，但在目前物件導向的程式設計領域，函式已經改稱爲『方法』)

Functions

If (Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
setTimeout()
write()
serialEvent()

圖 3-11　Serial 物件的方法

啓用序列埠

要使用序列埠，首先要使用 begin() 方法啓用序列埠，先規定雙方傳輸速度，本例規定 9600，程式如下：

```
Serial.begin(9600);
```

然後也於序列埠視窗點選相同的速率，如圖 3-12：

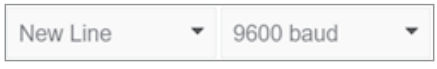


圖 3-12　序列埠傳輸速率

輸出

要在序列埠輸出文數字，要用 print() 或 println() 方法，兩者的差別是 println() 輸出後換行歸位，print() 則沒有。例如，以下程式，可輸出『GwoshengGwosheng』，如下右圖。

`Serial.print("Gwosheng");` `Serial.print("Gwosheng");`	`GwoshengGwosheng`

以下程式可輸出兩列 Gwosheng，如下圖右。

`Serial.println("Gwosheng");` `Serial.println("Gwosheng");`	`Gwosheng` `Gwosheng`

若是 print() 或 println() 方法內接變數，則會轉爲印出此變數的內容。例如，以下程式，可將變數所代表的內容輸出至使用者電腦螢幕。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
String b="Gwosheng";//規定資料的儲存空間，宣告b爲字串變數
Serial.println(a);//3
Serial.println(b);//Gwosheng
```

以下程式可輸出『a=3』。有加雙引號『""』的是字串，沒加雙引號的是變數，遇到字串就直接輸出，遇到變數就往前找此變數所儲存的值，並輸出。

```
int a=3;//規定資料的儲存空間，宣告a爲整數變數
Serial.print("a=");//字串就直接輸出a=
Serial.print(a);//變數，輸出變數的值3
```

自我練習

1. 請鍵入以下程式，並寫出執行結果。

```
1.  void setup() {
2.      Serial.begin(9600);
3.      Serial.print("Gwosheng");
4.      Serial.print("Gwosheng");
5.      Serial.println();//only linefeed
6.      Serial.println("Gwosheng");
7.      Serial.println("Gwosheng");
8.      Serial.println();//only linefeed
9.      int a=3;//規定資料的儲存空間，請看4-1節
10.     String b="Gwosheng";//規定資料的儲存空間，請看4-1節
11.     Serial.println(a);
```

```
12.     Serial.println(b);
13.     Serial.print("a=");
14.     Serial.print(a);
15. }void loop() {}//此函式雖然沒有放程式，但也不能刪除
```

2. 請於序列埠監控視窗輸出自己的座號與名字，本例輸出座號後換行歸位，再輸出姓名。
3. 同上題，請在同一列輸出自己座號與名字。

程式的燒錄

以前的單晶燒錄，是要將單晶 IC 從電路板拆下來，先用紫外線抹除資料，然後放到燒錄器燒錄，最後再重新插入電路板。Arduino 則方便了，在整合編輯環境與 Arduino 微控板裡，此單晶 IC 不用一再插拔，使用者的電路透過微控板與單晶片連接，一個按鍵「Upload」  就可直接燒錄。燒錄結束，USB 連接線不用插拔，USB 連接線繼續扮演提供電源角色，微控板內的單晶片轉爲自己當家，掌控使用者所連接的電路（也含電腦的鍵盤與螢幕），自動執行此程式。

3-4 I/O腳位探索

Arduino Mega 2560 如圖 3-13，有 70 個 I/O 接腳，編號爲 0 到 53 與 A0 ～ A15。

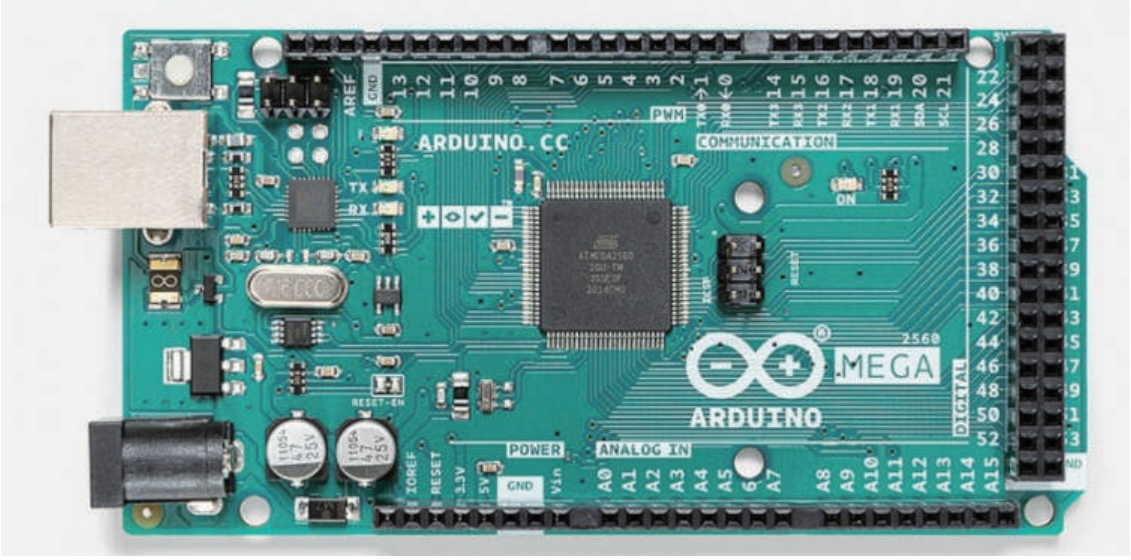


圖 3-13　Arduino Mega 2560 實體圖

這些接腳都可使用腳位編號（例如，0, 1, 2,…A1, A2…）單獨使用，也可使用暫存器名稱指派工作。例如，22, 23, 24, 25, 26, 27, 28, 29 這 8 隻腳暫存器名稱爲 PORTA，亦可以使用暫存器名稱 PORTA，同時使用 22~29 等腳位。Arduino Mega 2560 所有腳位與暫存器名稱，如表 3-1 所示：

▶ 表 3-1　Arduino 暫存器名稱與腳位編號對照表

暫存器\位元	7	6	5	4	3	2	1	0
PORTA	29	28	27	26	25	24	23	22
PORTB	13	12	11	10	50	51	52	53
PORTC	30	31	32	33	34	35	36	37
PORTD	38				18	19	20	21
PORTE			3	2	5		1	0
PORTF	A7	A6	A5	A4	A3	A2	A1	A0
PORTG			4			39	40	41
PORTH		9	8	7	6		16	17

PORTI								
PORTJ						15	14	
PORTK	A15	A14	A13	A12	A11	A10	A9	A8
PORTL	42	43	44	45	46	47	48	49

以上這些接腳都可以使用軟體的設定，指派腳位功能。其次，腳位功能有三種，分別是數位輸出、數位輸入、或有上拉電阻 INPUT_PULL UP 的輸入等 3 種功能，分別說明如下：

指派腳位功能

Arduino 指派腳位功能有兩種方式，分別是單隻腳位的 pinMode 指派與 8 隻腳一起指派的 DDRA 指令（Data Direction of Port A 的縮寫）。例如，以下程式可使用 pinMode 指派腳位 13 作爲數位輸出。

```
pinMode(13, OUTPUT);
```

以下程式可指派腳位 29,28,27,26,25,24,23,22 作爲數位輸出。

```
pinMode(29, OUTPUT);
pinMode(28, OUTPUT);
pinMode(27, OUTPUT);
pinMode(26, OUTPUT);
pinMode(25, OUTPUT);
pinMode(24, OUTPUT);
pinMode(23, OUTPUT);
pinMode(22, OUTPUT);
```

以上腳位 29 ～ 22 剛好是 PORTA 暫存器，所以以上程式亦可簡化爲

```
DDRA=B11111111;// 1是輸出，指派PORTA爲輸出
```

B 表示後續數字代表二進位，1 表示輸出，0 表示輸入，也可寫成

```
DDRA=0xff;//0x開頭代表後續是16進位數字
```

同理，若是

```
DDRA=B00000000;// 0是輸入，指派PORTA爲輸入
```

則指派 PORTA 爲輸入。也就是 pinMode 是配合腳位名稱，一次指派 1 個腳位的功能；DDRA 是配合暫存器名稱，一次指派 8 個腳位的功能，同理 PORTB、PORTC 就用 DDRB、DDRC 等指派其功能。

數位輸出

Arduino 腳位若當作數位輸出，使用手冊的說明如下：

> **Properties of Pins Configured as OUTPUT**
>
> Pins configured as OUTPUT with pinMode() are said to be in a low-impedance state. This means that they can provide a substantial amount of current to other circuits. Atmega pins can source (provide positive current) or sink (provide negative current) up to 40 mA (milliamps) of current to other devices/circuits. This is enough current to brightly light up an LED (don't forget the series resistor), or run many sensors, for example, but not enough current to run most relays, solenoids, or motors.
>
> Short circuits on Arduino pins, or attempting to run high current devices from them, can damage or destroy the output transistors in the pin, or damage the entire Atmega chip. Often this will result in a "dead" pin in the microcontroller but the remaining chip will still function adequately. For this reason it is a good idea to connect OUTPUT pins to other devices with 470Ω or 1k resistors, unless maximum current draw from the pins is required for a particular application.

大意是說，當輸出時，若設定爲 HIGH，則電壓有 5V，且每隻腳位高電位的最大輸出電流是 20mA；其次，設定爲 LOW 時，可承受或稱流入 40mA 的電流。

指派電位

單晶片的優點是您可直接下指令，指派任何接腳爲 HIGH 或 LOW。指派的方式有兩種，分別是單隻腳位的 digitalWrite 指派與八位元的暫存器名稱 (如 PORTA) 整體指派。例如，以下程式，您可指派接腳 22 爲 HIGH。

```
digitalWrite(22, HIGH);
```

以下程式，您可指派其爲 LOW。

```
digitalWrite(22, LOW);
```

所有腳位都有一個暫存器名稱，還可使用暫存器名稱一起指派 8 隻腳的電位，例如，以下程式可快速指派 PORTA 的 8 個位元全爲 HIGH，PORTA 是暫存器名稱。

```
PORTA=B11111111;
```

以下程式可快速指派 PORTF 的 8 個位元輸出全爲 LOW。

```
PORTF=0;//0就0，當然不用再指派任何進位。
```

範例 3-4a

PORTA 腳位探索實習。

1. 請鍵入以下程式，驗證、上傳。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;// 指派PA爲輸出
    PORTA=B11111111;//指派PA爲高電位
}void loop() {}//此loop()函式雖然沒用到，但不能刪除
```

2. 請用三用電表，檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲高電位 5V。
3. 鍵入以下程式，重新驗證、上傳，再檢驗微控板的腳位 22, 23, 24, 25, 26, 27, 28, 29 是否爲低電位 0V。

```
void setup() {
    DDRA=B11111111;// 指派PORTA爲輸出
    PORTA=0;//指派PORTA輸出低電位
}void loop() {}
```

自我練習

1. 請鍵入以下程式，並驗證 PORTA、PORTB、PORTC 對應腳位是否爲高電位。

```
void setup() {
  // put your setup code here, to run once:
    DDRA=B11111111;//指派PA爲輸出
    PORTA=B11111111;//設定PA0～7均爲高電位
    DDRB=B11111111;//指派PB爲輸出
    PORTB=B11111111;//設定PB0～7均爲高電位
    DDRC=B11111111;
    PORTC=B11111111;
} void loop() {}
```

2. 請探索 PORTF、PORTK、PORTL 腳位在哪裡？並寫程式檢驗這些腳位電壓是否能隨著您的程式而變化。

學後測驗

1. 請問 Arduino 安裝完成後，可開啓哪一自我測試程式，此測試程式可以讓內植 LED 連續亮滅？_____
2. 請問 Arduino 程式由哪兩個函式組成？_____，_____。那一個函式僅執行一次，通常當作基本設定？_____，那一個函式無限重複循環執行，通常放主程式？_____
3. 寫出可以讓內植 LED 亮 1 秒的程式？_____
4. 寫出可以讓內植 LED 滅 1 秒的程式？_____
5. 寫出於序列埠視窗輸出「AA」的程式？_____
6. 寫出 PORTA 暫存器由哪些腳位組成？_____
7. 寫出 PORTB 暫存器由哪些腳位組成？_____
8. 寫出指派 PORTC 為輸出的程式？_____
9. 寫出指派 PORTC 為輸入的程式？_____
10. 寫出指派 PORTF 全為高電位的程式？_____
11. 寫出指派 PORTF 全為低電位的程式？_____

MEMO

Chapter
04

程式的撰寫

學習大綱

- 4-1　高階程式指令應用
- 4-2　程式編寫

學習目標

- 1. 能熟練高階語言的資料型態。
- 2. 能熟練高階語言的運算子。
- 3. 能熟練高階語言的決策指令。
- 4. 能熟練高階語言的迴圈指令。
- 5. 能熟練高階語言的陣列型態。
- 6. 能熟練高階語言的自訂函式。
- 7. 能熟練高階語言的程式編寫。

4-1 高階程式指令應用

Arduino 的指令主要來自 C/C++ 語言的精華，但因其主要功能是控制 I/O，有些語法與資料型態有簡化，並增加一些 I/O 專用指令，以下簡介其資料型態、運算子、決策指令、陣列、自訂函式的精華。

4-1-1 資料型態

資料種類

Arduino 所能處理的資料種類分別有數值 (含整數、長整數及浮點數)、字元與字串等。

▮整數(Integers)

Arduino 可以處理的整數有三種進位方式，分別是十進位 (Decimal)、十六進位 (Hexadecimal) 及八進位 (Octal)。其中十進位則以我們平常書寫數字的方式即可。十六進位應以 0X 或 0x 開頭（數字的 0，不是字母的O），且以 0,1,2,3,4,5,7,8,9,a,b,c,d,e,f 代表 0 到 15，例如 0xa、或 0XB 均爲十六進制 (十六進位的 a,b,c,d,e,f 等字元大小寫都可以)，分別代表十進位的 10 與 11；八進位則應以 0 開頭（數字的 0），例如，072 則爲八進位，等於十進位的 58。

▮浮點實數(Floating-point literals)

數字中含有小數點或指數的稱爲浮點數、實數或浮點實數。浮點數可使用標準寫法或科學符號法表示，例如 321.123 即爲標準寫法，1.23e+4 即爲科學符號表示法。以指數爲例，E 或 e 表示 10 的次方，例如 0.0023、2.3E-3 及 2.3e-3 都是表示相同的浮點數;又例如 2.3E+2 則代表 230。但是，Arduino 的主要功能是 I/O 控制，不是數值運算的處理器，實數的表示與運算誤差很大。請自行鍵入以下程式，並觀察執行結果。

```
float a=2.3E-3;
Serial.begin(9600);
Serial.println(a);
```

▮字元(Character literals)

使用單引號『'』圍住的單一字元，稱爲字元，例如 'A' 或 'a' 等。

▮字串

Arduino 與 C 語言相同，字串以雙引號『" "』所圍住，例如 ,"Gwosheng"," 台灣 " 等。

▮布林值

C 語言使用 1 或 true 代表布林的 true，0 或 false 代表布林的 false。

資料型態

電腦爲了有效率的處理資料，就有資料型態（Data Types）的規劃，也就是大的資料用大盒子裝，小的資料用小盒子裝，如此才可節省記憶體，並加快處理效率。反過來說，若不分資料大小，通通用大盒子裝資料，那將會非常浪費記憶體，也拖垮執行效率。例如，所有的東西都用冰箱的盒子裝當然可以，但這樣非常浪費空間，還有，搬運時也很耗時。Arduino 所提供的資料型態、所佔用記憶體、所能代表的數值範圍如表 4-1：

▶ 表 4-1　Arduino 資料型態

資料型態	中文名稱	佔用記憶體的大小（位元）	所能代表的數值的範圍	備註
char	字元	8	-128 ～ 127	
byte	位元組	8	0 ～ 255	
int	整數	16	-32768 ～ 32767	
long	長整數	32	-2147483648 ～ 2147483647	
float	浮點數	32	+/-3.4E+-38	
double	倍精度浮點數	32	+/-3.4E+-38	Arduino 的 double 同 float
unsigned char	正字元	8	0 ～ 255	
unsigned int	正整數	16	0 ～ 65535	
unsigned long	正長整數	32	0 ～ 42949667295	

▶ 表 4-1 Arduino 資料型態（續）

資料型態	中文名稱	佔用記憶體的大小（位元）	所能代表的數值的範圍	備註
bool	布林	8	true or false	boolean 也可，但不鼓勵
String	字串			

自我練習

1. 請線上查詢 Arduino 資料型態。

變數宣告

變數的功能是用來輸入、處理及儲存外界的資料，而變數在使用以前則要事先宣告才可使用。Arduino 語言的變數宣告語法如下：

```
資料型態 變數名稱[=初值];
```

例如：

```
byte a;
```

即是宣告變數 a 爲 byte 型態，佔用 1 個 Byte，此種型態僅能儲存 0 到 255。變數的宣告亦可連同初值一起設定，如以下敘述：

```
float d = 30.2;
```

以下敘述，同時宣告兩個變數，且設定其初值。

```
int f1=3,f2=3;
```

以下敘述可宣告布林型態:（布林型態用來表示，運算結果的『眞』與『僞』）

```
bool g=true;
```

C/C++/Arduino 都可用字元陣列表示字串，以下程式可宣告 a 爲字串型態，請留意字元是單引號，字串是雙引號。

```
char a[]="ABC";//字串用雙引號
```

C++/Arduino 才有字串型態 String，以下程式可宣告變數 h 爲字串型態：

```
String h="123";
```

以下宣告 a 爲一維陣列：

```
byte a[8];
```

這樣就可使用 a[0]~a[7] 等 8 個變數。以下陣列宣告的同時，直接給予初值。

```
byte a[]={0x01,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f};
Serial.println(a[0]);
```

變數經過宣告之後，編譯器即會根據該變數的資料型態配置適當的記憶體儲存此變數，所以若要提高程式的執行效率，則應儘量依照資料性質，選擇佔用記憶體較小的資料型態。

變數的有效範圍

任一變數的宣告，若無特殊聲明，均屬於區域變數，其有效範圍僅止於該變數所在的程式區塊。所以，以下變數 i 的有效範圍僅在 setup()，無法在 loop(){} 函式內存取。

```
void setup() {
  Serial.begin(9600);
  byte i=1;
}
void loop() {
  Serial.println(i);
}
```

其次，請比較圖 4-1a 與圖 4-1b 的差別，圖 4-1a byte i=0; 放在 void loop() {} 裡面，則每次執行 void loop() {} 時，byte i=0 都被執行，所以其值永遠都相同，沒有累加效果。此時就要把此敘述放在外面，成爲全域變數，所以此 i，稱爲計數器，也可以想像成 loop() 被執行的次數，如圖 4-1b。

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  byte i=0;
  i=i+1;
  Serial.println(i);
}
```

圖 4-1a　i 為區域變數

```
void setup() {
  Serial.begin(9600);
}
byte i=0;
void loop() {
  i=i+1;
  Serial.println(i);
}
```

圖 4-1b　i 為全域變數

4-1-2 運算子

所謂運算子（Operator），指的是可以對運算元（Operand）執行特定功能的特殊符號。例如：3＋2的『3』與『2』稱爲運算元，『+』稱爲運算子。Arduino的運算子分爲五大類，分別是：算術（Arithmetic）運算子、比較（Comparison）運算子、布林（Boolean）運算子、位元操作（Bitwise）運算子及複合（Compound）運算子，分別說明如下：

算術運算子

下表是Arduino的算術運算子。（請開啓https://www.arduino.cc/reference/en/）

Arithmetic Operators

% (remainder)

* (multiplication)

+ (addition)

- (subtraction)

/ (division)

= (assignment operator)

圖 4-2　Arduino 算術運算子

以上算術運算子用來執行一般的算術運算，包括指派 (=)、取正負數 (+/-)、加 (+)、減 (-)、乘 (*)、除 (/)、取餘數 (%) 等，表 4-2 是以上算術運算子的功能說明：

▶ 表 4-2　Arduino 算術運算子

運算子	定義	優先順序	結合律
=	指派	15	由右至左
+/-	正負號，一元運算子	2	由右至左
*	乘法運算	4	由左至右
/	除法運算	4	由左至右
%	求餘數（Modulus）	4	由左至右
+/-	加法 / 減法運算	5	由左至右

四則運算

以下是一些簡單四則運算：

```
int a=5,b=4;
Serial.println(a+b);//9
Serial.println(a-b);//1
Serial.println(a*b);//20
Serial.println(a%b);//1 取餘數
```

以上程式，請放在 setup() 裡面，就可觀察執行結果。

```
void setup() {
  Serial.begin(9600);
  //執行一次的放這裡
  int a=5,b=4;
  Serial.println(a+b);//9
  Serial.println(a-b);//1
  Serial.println(a*b);//20
  Serial.println(a%b);//1 取餘數
}
void loop() {}//重複執行的放loop()函式裡面，本例雖然沒用到，也不能省略。
```

整數除法或實數除法

Arduino/C/C++ 的除法運算，只有被除數與除數的型態均爲整數，才是整數除法，商的型態爲整數；否則即爲實數除法，得到實數商。例如：

```
int x=5, y=4,z;
float xf=5,yf=4;
Serial.println(x/y); // 1，被除數與除數的型態均爲整數
Serial.println(xf/y);//1.25
Serial.println(x/yf);//1.25
Serial.println(xf/yf);//1.25
```

比較運算子（Comparison Operators）

比較運算子又稱爲關係（Relational）運算子，用於資料之間的大小比較，比較的結果可得到 bool 型態的 1 (true) 或 0 (false)，以作爲以上決策『運算式』運算依據。表 4-3 是 Arduino 語言中的關係運算子符號，這些都和 C/C++ 相同。

▶ 表 4-3　Arduino 比較運算子

運算子	定義	優先順序	結合律
<	小於	7	由左至右
>	大於	7	由左至右
<=	小於等於	7	由左至右
>=	大於等於	7	由左至右
==	等於	8	由左至右
!=	不等於	8	由左至右

例如：

```
int a=5,b=3;
float c=5;
Serial.println(a>b);//1
Serial.println(a>=b);//1
Serial.println(a==b);//0
Serial.println(a!=b);//1
Serial.println(a=!b);//0 小心不要打錯，且沒有錯誤信息
Serial.println(a==c);//0 變數型態要相同才能比較
Serial.println(c>b);//1 變數型態要相同才能比較，但這次勉強正確
Serial.println(a=b);//3  單個等號是指派，請小心
```

布林運算子（Boolean Operators）

布林運算子又稱邏輯（Logical）運算子。當同一個運算式要同時存在兩個以上的比較運算時，則每兩個比較運算子之間必須使用布林運算子連結。例如，您要找『男生』且『年齡大於 40』，此一決策就同時含有兩個比較運算式，此時就要運用布林運算子連結。Arduino 布林運算子如表 4-4 所示：

▶ 表 4-4　Arduino 布林運算子

運算子	定義	優先順序	結合律
!	布林邏輯 not 運算	2	由右至左
&&	布林邏輯 and 運算	12	由左至右
||	布林邏輯 or 運算	13	由左至右

例如：

```
int a=5,b=3;
Serial.println(!(a>b));  //0  Arduino用0表示false
Serial.println((a>b) && (b>=4));//0 Arduino用0表示false
Serial.println((a>b) || (b>=4)); //1 Arduino用1表示true
```

又例如，要判斷 x 是否滿足 1<x<=6，則敘述如下：

```
int x=5;Serial.println((x>1) &&(x<=6));//1
int x=8;Serial.println((x>1) &&(x<=6));//0
```

自我練習

1. 請寫一程式，任意指派 -10 到 10 的整數 x，若此整數 x 滿足 -3<x<=6，請輸出『1』。
2. 請寫一程式，任意指派 2 到 12 的整數 x，若此整數 x 滿足 x>=8 或 x<4，請輸出『1』。
3. 請寫一程式，可以指派三角形三邊長 a, b, c，若能圍成三角形，則輸出『1』。提示：圍成三角形的條件是，任兩邊之和要大於第三邊，數學語言是：a+b > c and a+c > b and b+c > a，請寫程式完成以上判斷。

位元（Bitwise）運算子

位元（Bitwise）運算子是將一個 8 位元的整數逐位元進行運算，此類運算子還可以分爲兩類：位移（Shift）運算子與布林運算子。位移運算子可以用來將各個位元向左或是向右移；布林運算子則可以逐位元進行布林運算。表 4-5 是 Arduino 語言位元操作運算子的列表：

▶ 表 4-5　Arduino 位元運算子

運算子	定義	優先順序	結合律
~	位元 not 運算	2	由右至左
&	位元 and 運算	9	由左至右
^	位元 xor 運算	10	由左至右
|	位元 or 運算	11	由左至右
<<	逐位元向左位移	6	由左至右
>>	逐位元向右位移	6	由左至右

運算子 ~

『~』是位元 not 運算，not 是將位元 0 變 1，1 變 0，運算結果如表 4-6：

▶ 表 4-6　運算子 not 真值表

a	c=~a
1	0
0	1

例如：

```
byte a=1,b=0;
Serial.println(~a);// -2
```

a=1，分解爲 2 進位是 00000001

```
~a= not 00000001=11111110
```

首位元是 1 表示負數，那到底負多少，再取 2 補數。2 補數的步驟是先取 1 補數再加 1。所謂 1 補數是『逐位元將 1 變 0，0 變 1』，所以上面 11111110 的 1 補數是：

```
00000001
```

將 1 補數再加 1，就是 2 補數，上面 00000001 加 1 就是 00000010，其大小是 2。上面符號既然是『-』，那就代表此數是 -2，此即爲二補數的觀念。

運算子 &

『&』是位元 and 運算，and 運算眞值表如表 4-7，當位元 a 與 b 同時爲 1，c 才得到 1：

▶ 表 4-7　運算子 & 真值表

a	b	c= a & b
0	0	0
0	1	0
1	0	0
1	1	1

例如：

```
byte a=1,b=5;
Serial.println(a&b);//00000001 & 00000101=00000001=1
```

運算子 ^

『^』是位元 xor 運算，xor 運算眞值表如表 4-8，當位元 a 與 b 不相同時，c 才得到 1。

▶ 表 4-8　運算子 ^ 真值表

a	b	c=a^b
0	0	0
0	1	1
1	0	1
1	1	0

例如：

```
byte a=1,b=5;
Serial.println(a^b);//00000001 ^00000101=00000100=4
```

運算子 |

『|』是位元 or 運算，or 運算眞值表如表 4-9，當位元 a 與 b 有一爲 1，c 就得到 1。

▶ 表 4-9　運算子 | 真值表

a	b	c=a | b
0	0	0
0	1	1
1	0	1
1	1	1

例如：

```
byte a=1,b=5;
Serial.println(a|b);//00000001 | 00000101 =00000101=5
```

運算子 <<

『<<』是左移運算子，例如：

```
byte a=1;
Serial.println(a<<1);//2   1向左移爲1位元，結果是2
a=1;
Serial.println(a<<2);//4   1向左移爲2位元，結果是4
```

運算子 >>

『>>』是右移運算子，例如：

```
byte a=4;
a=a>>1;
Serial.println(a);//2    4向右移爲1位元，結果是2
```

運算子的優先順序（Precedence）

同一敘述，若同時含有多個運算子，此時即需定義運算子的優先順序。例如：

```
a=3+4*2;
```

『=』優先順序是12，『+』優先順序是5 ，『*』優先順序是4，所以運算順序是：

```
(a=(3+(4*2)));
```

運算子的結合律（Associativity）

當同一敘述，相鄰的運算子擁有相同優先順序的運算子，此時即需定義運算子是左結合或右結合。例如：

```
x=a-b-c;
```

連續兩個減號『-』，優先順序相同，此時就要靠定義結合律，減法結合律是由左至右，所以以上同義於：

```
x=((a-b)-c);
```

而

```
x=y=z=2;
```

連續三個指派運算子『=』，指派運算子的結合律是由右至左，所以以上式子同義於：

```
(x=(y=(z=2)));
```

所以，以上敘述，x、y、z 的結果都是 2。

各種進位制

我們人類習慣使用 10 進制，逢 10 填 0 進 1，例如，$(242)_{10}$ 是表示 $2*10^2 + 4*10^1 + 2*10^0 = 242$；若是 8 進位，那就是逢 8 填 0 進 1，僅用 0, 1, 2, 3, 4, 5, 6, 7 等 8 個數字，所以 $(11)_8 = 1*8^1 + 1*8^0 = (9)_{10}$；若是 2 進位，那就是逢 2 填 0 進 1，僅用 0, 1 兩個數字，所以 $1011 = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 = 11$；若是 16 進位，那就用 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F 表示 0 到 15，且逢 16 填 0 進 1，所以 $(A2E)_{16} = 10*16^2 + 2*16^1 + 14*16^0 = (302)_{10}$。

Arduino語言可以處理的整數有四種進位方式，分別是十進位（Decimal）、二進位（Binary）、八進位（Octal）及十六進位（Hexadecimal）。其中十進位則以我們平常書寫數字的方式即可，例如12；二進位則以B開頭，例如，B11則代表十進位的3；八進位則應以0開頭，例如011代表十進位的9(1*8+1)；十六進位應以0x開頭，例如，0x11代表十進位的17(1*16+1)。請鍵入以下程式，並觀察執行結果。

```
void setup() {
    Serial.begin(9600);
    int a=242;
    int b=B11;
    int c=011;//數字0，不是字母O
    int d=0x11;//數字0開頭，不是字母O
    Serial.println(a);
    Serial.println(b);
    Serial.println(c);
    Serial.println(d);
}void loop() {}
```

2進位與16進位

前面資料的數位化已經介紹，處理機是以2進位儲存數值，所以若以8位元儲存一個正整數，例如：

```
5
```

就會以

```
00000101
```

表示，我們若以LED觀察結果就是『滅滅滅滅滅亮滅亮』，反過來說，若有8顆LED呈現：

```
滅滅滅滅滅亮滅亮
```

要將此現象數位化，我們也可用2進位表示爲：

```
B00000101
```

其次，以上 2 進位有點長，所以我們習慣以 16 進位表示爲：

```
0x05
```

也就是在自動控制的領域裡，我們會習慣以 2 進位或 16 進位來表示一些燈號或控制結果，請讀者要慢慢習慣這種 2 或 16 進位表示方式。表 4-10 是一些常用數字的 16 進位書寫表示方式。

▶ 表 4-10　10 進位與 16 進位對照表

10進位	2進位	16進位	10進位	2進位	16進位
0	B0	0x0	11	B1011	0xb
1	B1	0x1	12	B1100	0xc
2	B10	0x2	13	B1101	0xd
3	B11	0x3	14	B1110	0xe
4	B100	0x4	15	B1111	0xf
5	B101	0x5	16	B10000	0x10
6	B110	0x6	17	B10001	0x11
7	B111	0x7	18	B10010	0x12
8	B1000	0x8	127	B01111111	0x7f
9	B1001	0x9	254	B11111110	0xfe
10	B1010	0xa	255	B11111111	0xff

自我練習

1. 若有 8 個燈號連續排列，且其燈號是『滅亮亮滅亮亮亮亮』，請問該如何以 16 進制數字回報。

亂數的產生

Arduino 產生亂數是使用 random() 方法，其語法如下：

```
random(min,max)
```

含 min，但不含 max，例如：以下程式產生 1 到 6 的整數亂數。

```
a=random(1,7);
```

min 可省略，若省略表示從 0 開始。例如：以下程式產生 0 到 6 的整數亂數。

```
a=random(7)
```

但使用前，要將 A0 腳位空接，使用以下程式起始亂數種子。（使用 A0 的雜訊得到不同的亂數起點）

```
randomSeed(analogRead(A0));
```

以下程式，可每秒產生 1 個 1 到 6 的整數亂數。

```
void setup() {
  randomSeed(analogRead(A0));
  Serial.begin(9600);
}
void loop() {
  int x=random(1,7);// 產生1個1到6的整數亂數
  Serial.println(x);
  delay(1000);
}
```

4-1-3 決策指令

人類的生活必須不斷面對決策問題，連一個不到三歲的小孩，也常要思考他手裡的十元是要坐電動車還是買棒棒糖。程式語言是協助人類解決問題的工具，當然也有決策流程敘述，Arduino 語言依決策流程點的多寡，分爲以下兩種決策流程敘述，第一是雙向分歧決策流程 if ~else~，第二是多向分歧決策流程的 switch...case，分別說明如下：

if...else

在日常生活領域中，常出現“假如～則～，否則～”時，此種決策流程模式有兩種解決問題的方案，故稱爲雙向分歧決策流程，此時可使用 if...else 敘述。if...else 敘述的語法如下：

```
if（運算式）
{
```

```
    敘述區塊1；
}
[else
{
    敘述區塊2；
}]//語法中的中括號[]，表示裏面內容可省略
```

以上語法流程圖如圖 4-3。

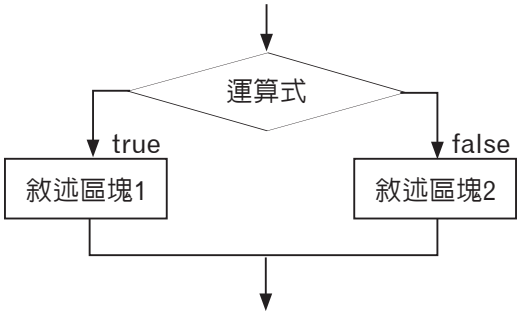


圖 4-3　if~else 流程圖

例如，若有流程圖如圖 4-4：

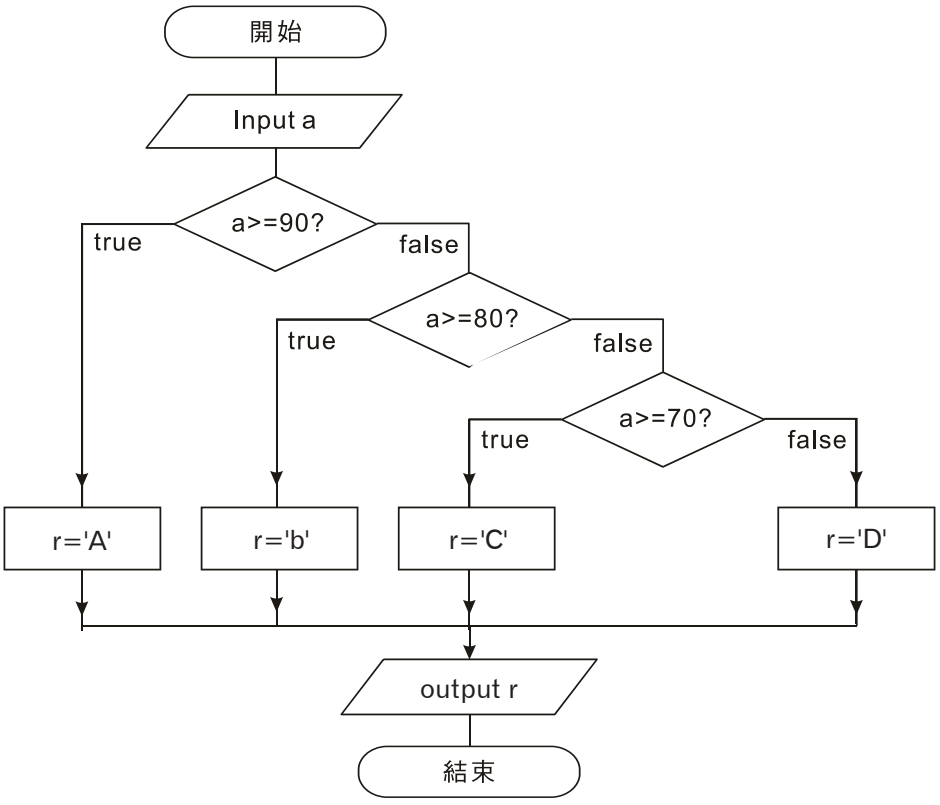


圖 4-4　成績判斷流程圖

請鍵入以下程式，寫出執行結果。

```
1.  void setup() {
2.      Serial.begin(9600);
3.  }
4.  void loop() {
5.      int a;
6.      char b;
7.      Serial.print("Input a:");
8.      while(Serial.available() ==0) {}//等待使用者由鍵盤輸入資料
9.      a=Serial.parseInt(); //使用鍵盤輸入一個整數
10.     Serial.println(a);
11.     if(a>=90)                   /* 高於90分爲A */
12.         b='A';
13.     else
14.         if(a>=80)               /* 介於 80與90分爲B */
15.             b='B';
16.         else
17.             if(a>=70)           /* 介於 70與80分爲C */
18.                 b='C';
19.             else
20.                 b='D';          /* 不符合上述情況則爲D */
21.     Serial.print("The level is:");
22.     Serial.println(b);
23. }
```

程式說明

所有程式語言當需要使用者輸入資料時，都會原地等待使用者輸入，直到得到資料再往下執行，但是單晶微處理機不一樣，不能原地等待，因爲單晶微處理機要同時偵測與執行很多周邊設備，所以不能不做事而原地等待，若一定要單晶微處理機原地等待，則可使用 available()，也就是若沒有資料備妥，則持續等待。

自我練習

1. 請寫一個程式，每秒產生 1 個 -3 到 3 的亂數，且評判其爲負數、0、或正數。

2. 請寫一個程式，完成以下要求：
   (1) 可由鍵盤輸入一個 0 ～ 25 的整數。
   (2) 當此數大於等於 20 時，輸出五個燈。
   (3) 當此數是 16 ～ 19 時，輸出四個燈。
   (4) 當此數是 11 ～ 15 時，輸出三個燈。
   (5) 當此數是 0 ～ 10 時，輸出兩個燈。
3. 心算練習。請產生與輸出 2 個 1 位數亂數，由使用者輸入相加結果，微處理機判斷是否正確。

switch...case

一個決策點若同時擁有三個或三個以上的解決方案，則稱此爲多向分歧決策。多向分歧決策雖也可使用前面巢狀 if else 解決，但卻增加程式的複雜度及降低程式可讀性，若此一決策點能找到適當的運算式，能使問題同時找到分歧點，則可使用 switch case 敘述。switch case 語法如下：

```
switch（運算式）
  {
    case常數1：
         敘述區塊1；
         break;
    case常數2：
         敘述區塊2；
         break;
    case常數3：
         敘述區塊3；
         break;
     〔default:
         敘述區塊n；]//語法中的中括號[]，表示裏面可省略
  }
```

以上語法說明如下：

1. switch 的運算式值僅能爲整數或字元。
2. case 的常數僅能整數或字元，且其資料型態應與上面的 switch 運算式相同。

3. 處理機將會依 switch 的運算式值，逐一至常數 1、常數 2 尋找合乎條件的 case，並執行相對應的敘述區塊，直到遇到 break 敘述，才能離開 switch。
4. default 可放置特殊情況，也就是沒有適當的 case，則執行 default。若省略 default，且若沒有任何 case 滿足 switch 運算式，則程式會默默離開 switch 敘述。（備註：語法中，兩旁加中括號表示此敘述可省略。）
5. 敘述區塊可放置任何合法的敘述，當然也可放置 switch 或 if。
6. 以下敘述，可將 1、2、3、4 轉爲對應的季節。

```
1.  void setup() {
2.      Serial.begin(9600);
3.      byte a=1;
4.      String b="";
5.      switch(a)
6.      {
7.          case 1:
8.            b="Spring";         /*春*/
9.            break;
10.         case 2:
11.           b="Summer";         /*夏*/
12.           break;
13.         case 3:
14.           b="Fall";           /*秋*/
15.           break;
16.         case 4:
17.           b="Winter";   /*冬*/
18.           break;
19.       default :
20.           b="input error";
21.     }
22.     Serial.println(a);
23.     Serial.println(b);
24. }void loop() {}
```

7. 有些語言可用逗號將兩種 case 放在一起，但在 C/C++、Arduino 語言中每一 case 僅能放置一個常數，所以若兩個或兩個以上 case，有相同的處理方法，則應將兩個 case 分成兩個敘述，請看以下範例。

8. 以下以 switch case 重作以上使用 if else 的成績判斷。

```
1. void setup() {
2.     Serial.begin(9600);
3.     byte a=98;
4.     String b="";
5.     //本例將分數除以10，得到0~10的整數，所以適用多重分歧
6.     switch(a/10) {
7.        case 10:
8.        case 9:
9.           b="A";
10.          break;//do not leave out
11.      case 8:
12.          b="B";
13.          break;
14.      case 7:
15.          b="C";
16.          break;
17.      case 6:
18.      case 5:
19.      case 4:
20.      case 3:
21.      case 2:
22.      case 1:
23.      case 0:
24.          b="D";
25.          break;
26.  }
27.    Serial.println(a);
28.    Serial.println(b);
29.}void loop() {}
```

自我練習

請以 switch case 重做 4-1-3 節的自我練習第 1 與第 2 題。

4-1-4 迴圈指令

Arduino 有 for 與 while 迴圈指令，用於解決重複的工作，其使用時機與差別，請看以下本單元說明。

for 迴圈

for 迴圈是用於程式設計階段就知道重複執行的次數。例如，您想輸出 1 2 3 4 5 6，可以撰寫程式如下：

```
void setup(){
  Serial.begin(9600);
  for(int i=1;i<=6;i=i+1){
    Serial.print(i);Serial.print(",");
  }
}void loop() {}
```

1. 第一個『1』稱爲起始值；第 2 個『i<=6』，稱爲迴圈執行的條件；第 3 個『i=i+1』，是每次遞增或遞減量，i 先加 1，再放回 i，所以每次遞增 1。其次，因爲 i=i+1 這種運算使用非常頻繁，所以也可以寫成 i++，例如，以上程式也可寫成：

```
for(int i=1;i<=6;i++){
    Serial.print(i);Serial.print(",");
}
```

2. 以下程式可以輸出 2 4 6 8 ，每次遞增 2。

```
for(int i=2;i<=8;i=i+2){//每次遞增2
    Serial.print(i);
}
```

3. 以下左邊程式，『i=8』不行 ，因爲「=」是指派運算子，此與比較運算子「==」不同，會產生編譯錯誤；後面「i==8」是迴圈結束條件不同，可以通過編譯，但結果與「i<=8」不同，請鍵入以下程式，就會明瞭。

```
for(int i=2;i=8;i=i+2){
    Serial.print(i);
}
```

```
for(int i=2;i==8;i=i+2){
    Serial.print(i);
}
```

4. 以下程式可以輸出 2 4 6，請留意『<』與『<=』都可以，只是範圍不同。

```
for(int i=2;i<8;i=i+2){
    Serial.print(i);
}
```

5. 以下左邊與右邊都是由大而小遞減輸出 6 5 4 3 2 1，『i--』是『i=i-1』的複合運算子。

```
for(int i=6;i>=1;i=i-1){
    Serial.print(i);
  }
```

```
for(int i=6;i>=1;i--){
    Serial.print(i);
  }
```

6. 以下程式每次遞減 2 輸出，請留意『遞減』時不等式的方向，『>=』表示資料是由大到小。

```
for(int i=8;i>=1;i=i-2){
    Serial.print(i);//8642
}
```

7. 遞增不等式的方向是『<=』，遞減不等式的方向是『>=』。以下程式不等式方向通通錯了，就通通沒有輸出資料。不等式的方向很好記，因爲方向就代表遞增或遞減。

```
for(int i=1;i>=8;i=i+2){
    Serial.print(i);
}//i<=8才有結果
```

```
for(int i=8;i<=1;i=i-2){
    Serial.print(i);
}//i>=1才有結果
```

8. 迴圈的執行範圍是以兩個大括號『{}』表示，如以上程式都有大括號。若省略大括號，那就默認只執行後續一個敘述。例如，下圖左與右效果都相同。

```
for(int i=1;i<=6;i++){
    Serial.print(i);
  }
```

```
for(int i=1;i<=6;i++)
    Serial.print(i);
```

9. for 主要用來實現循序法。例如，人類計算 52*138 是使用直式乘法，但電腦可不用如此麻煩，因爲電腦有超強計算能力，所以就直接一個一個累加，程式如下：

```
void setup() {
  Serial.begin(9600);
  int s=0;
```

```
  int a=52, b=138;
  for (int i=1 ;i<=a;i++)
    s=s+b;
  Serial.println(s);
} void loop() {}
```

範例 4-1-4a

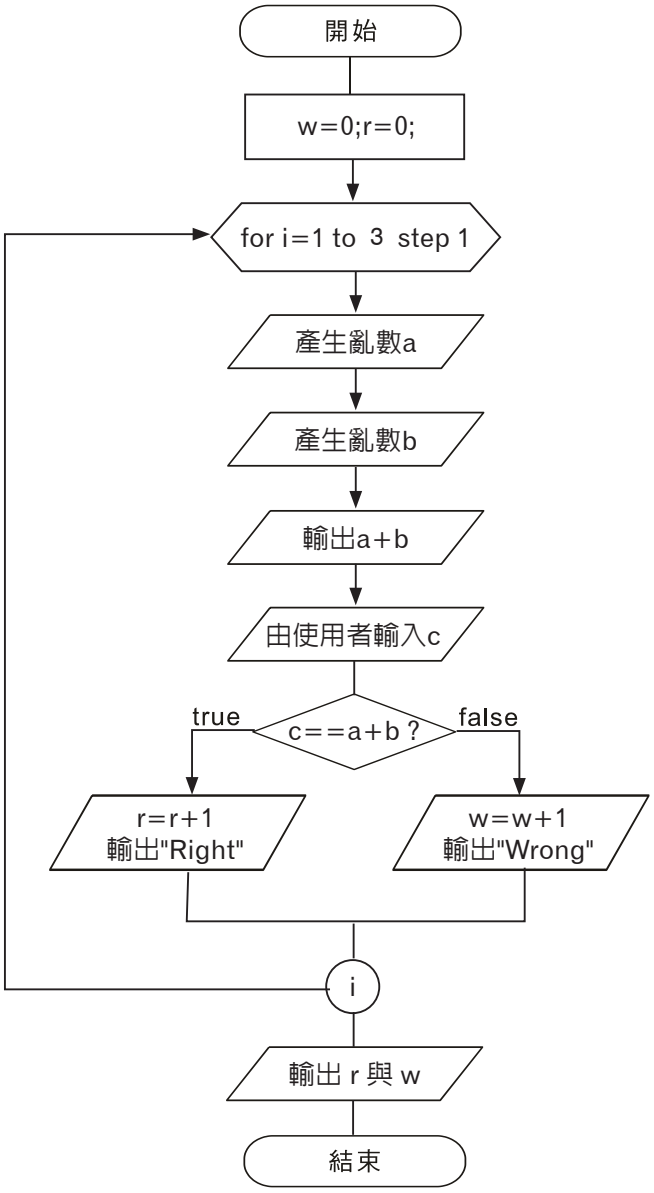
心算練習。請產生與輸出 2 個 1 位數亂數 3 次，由使用者輸入相加結果，微處理機判斷是否正確，且統計使用者答對的題數。

執行結果

```
COM5
1+2=
3
Right
7+3=
0
Wrong
6+5=
11
Right
Right=2
Wrong=1
```

程式列印

1. 以上程式的流程圖如右：



2. 根據以上流程圖，程式撰寫如下：

```
1.  void setup() {
2.    Serial.begin(9600);
3.    randomSeed(analogRead(A0));//播亂數種子，這樣每次才能得到不同亂數
4.    int a,b,c;
5.    int r=0;
6.    int w=0;
7.    for (int i=1;i<=3;i++){
8.        a=random(1,10);//產生1～9整數亂數
9.        b=random(1,10);
10.       //輸出 a+b=
11.       Serial.print(a);Serial.print("+");
                          Serial.print(b);Serial.println("=");
12.       while (Serial.available()==0){}//等待使用者輸入
13.       c=Serial.parseInt();  //取得使用者輸入
14.       Serial.println(c);
15.       if (c==(a+b)){//假如相同
16.         r=r+1;//答對題數累加1
17.         Serial.println("Right");
18.       }
19.       else {
20.         w=w+1;//答錯題數累加1
21.         Serial.println("Wrong");
22.       }
23.   }
24.   Serial.print("Right=");Serial.println(r);
25.   Serial.print("Wrong=");Serial.println(w);
26. }void loop() {}
```

自我練習

1. 請寫一程式，可以輸出“老師我愛您”20 次。
2. 請寫一程式，可以輸出 -4 -8 -12 -16 -20 -24
3. 請寫一程式，可以輸出所有英文小寫字元。
4. 請寫一個程式，可以計算 1+2+3…+100 的和。
5. 請寫一個程式，產生 10 個 -3 到 3 的亂數，並統計負數、0、正數的個數。

6. 認識鍵盤練習。請寫一個程式，電腦自動出現 1 個小寫字元十次，使用者輸入此字元，電腦評判是否正確，最後統計答對與答錯題數。
7. 請寫程式完成以下條件。
   (1) 產生 20 個 1 到 6 的亂數。
   (2) 輸出以上資料。
   (3) 計算並輸出共有多少個 3。
   (4) 統計所有數字出現次數。
8. 請寫程式完成以下條件。
   (1) 產生 50 個 0 到 100 的亂數。
   (2) 輸出以上資料。
   (3) 統計與輸出 0 ～ 9, 10 ～ 19, 20 ～ 29, 30 ～ 39, 40 ～ 49, 50 ～ 59, 60 ～ 69, 70 ～ 79, 80 ～ 89, 90 ～ 100 等區間的人數。

while指令

前面的 for 是用於程式設計階段已知迴圈次數，但有些情況，我們於程式設計階段並不知迴圈的執行次數，此時即可使用 while 指令。其次，有些迴圈可能一次都不執行，所以 while 指令又分爲前測試迴圈與後測試迴圈。while 的前測試迴圈語法如圖 4-6a，後測試迴圈如圖 4-6b。

```
while(運算式) {
    指令區塊;
}
```

圖 4-6a　前測試 while 語法

```
do {
   指令區塊;
}while (運算式);
```

圖 4-6b　後測試 while 語法

以上語法說明如下：

1. 不論是前測試或後測試迴圈，均是運算式值爲 1（true）時，繼續執行迴圈，運算式爲 0（false）時，離開迴圈。
2. 迴圈的範圍是指兩個大括號『{}』之間。
3. 前測試與後測試迴圈的差別爲，前測試迴圈有可能一次均不執行迴圈，但後測試迴圈至少執行一次。
4. 後測試迴圈的 while（運算式）後面要加分號（；），而前測試迴圈的 while 不用加分號。

範例 4-1-4a

探討計算機除法運算子，本例使用加減法，完成除法運算。

演算法則

兩數相乘時，程式設計階段就知道迴圈執行次數，所以使用 for。但除法就是不同了，只能說，當被除數大於除數時，就連續減去除數，能減去的次數，就是商，剩下的就是餘數。以 8/3 爲例，8 可以連續減 3 兩次，所以商就是 2，剩下餘數就是 2。以上說明的演算法如下：

1. a= 被除數。
2. b= 除數。
3. 商 q=0。
4. 所謂的商就是被除數 a 共有幾個除數 b，也就是只要 a 大於等於 b，就要執行以下指令：

   a=a-b；

   q=q+1；
5. 本例請以 8 除以 3，實際演練如下：

第1次	第2次	第3次
a=8;b=3 a>=b 成立 a=a-b=5 q=q+1=1	a=5;b=3 a>=b 成立 a=a-b=2 q=q+1=2	a=2;b=3 a>=b 不成立 q=2(商) r=a=2（餘數）

6. 以上演算法，以流程圖表示如圖 4-7。

程式列印

```
1.  void setup() {
2.      Serial.begin(9600);
3.      int a=8,b=3,q=0;
4.      while(a>=b) {//只要a>=b，則重覆執行迴圈
5.          a-=b;//a=a-b
6.          q++;//q=q+1
7.      }
```

```
8.      Serial.print("quotient = ");Serial.println(q);  /* 商數 */
9.      Serial.print("remainder= ");Serial.println(a);  /* 餘數 */
10. }void loop() {}
```

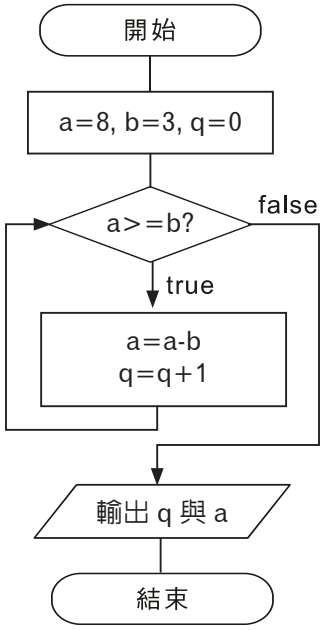


圖 4-7　範例流程圖

1. 本例必須重複很多次，所以適用迴圈，程式才能精簡，迴圈有 for 與 while，本例設計階段不知重複幾次，而是一面減、一面判斷，所以適用 while 迴圈。
2. 本程式僅能使用前測試 while 迴圈，而不能使用後測試迴圈，因爲有可能一開始被除數就小於除數。

※ 範例 4-1-4b

請寫一程式，可以輸入一個整數 (0 到 65535)，然後反向輸出其數字、數字和、幾位數。例如：輸入 2345，則輸出 5,4,3,2,sum=14,4 位數。

執行結果

程式列印

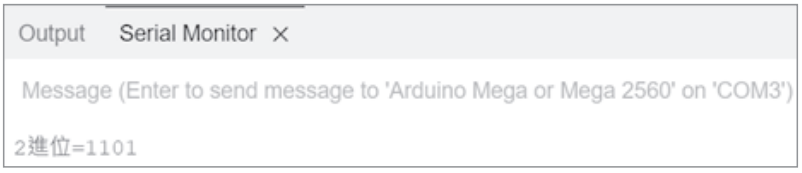
1. 只要每次取除以 10 的餘數，就可以從個位數取到每一個數字。
2. 因爲數字長度不限，設計階段不知迴圈次數，所以使用 while 重複迴圈，程式如下：

```
1.  void setup() {
2.    Serial.begin(9600);
3.    int a=2345;
4.    int sum=0;
5.    int b=0;
6.    int n=0;
7.    while (a>0){
8.      b=a%10;//求餘數
9.      sum=sum+b;//累加餘數和
10.     n=n+1;//計數器，統計迴圈執行次數
11.     Serial.print(b);Serial.print(",");
12.     a=a/10;//除以10
13.   }
14.   Serial.print("sum=");
15.   Serial.print(sum);
16.   Serial.print(",");
17.   Serial.print(n);
18. }void loop() {}
```

※ 範例 4-1-4c

請寫一個程式，可以將 10 進位轉爲 2 進位。

執行結果



程式列印

1. 將 10 進位整數 a 連續除以 2 的餘數串接就是 2 進位，只要 a>0 就要繼續求餘數。例如：

13/2=6..1

6/2=3..0

3/2=1..1

1/2=0..1 // 商已經是 0，迴圈結束

2. 先出爐的餘數放右邊，所以 2 進位是 1101。
3. 以上演算，也是設計階段不知迴圈重複次數，也是要使用 while 迴圈，所以程式如下：

```
1.  void setup() {
2.      Serial.begin(9600);
3.      int a=13;
4.      int b=0;
5.      String s="";
6.      while (a>0){
7.        b=a%2;//求餘數
8.        s=String(b)+s;//將b轉爲字串，向前進行字串串接
9.        a=a/2;//除以2
10.     }
11.     Serial.print("2進位=");
12.     Serial.print(s);
13.   }void loop() {}
```

自我練習

1. 同範例 4-1-4b，但是反相兩個 1 組輸出且輸出其和。例如，指派 12345，則輸出 45,23,1 與和是 69。
2. 心算練習。請連續產生與輸出 2 個 1 位數的亂數，由使用者輸入相加結果，微處理機判斷是否正確，直到使用者答錯爲止，且統計連續正確題數。
3. 請寫一個程式，滿足以下條件：（擲骰子遊戲）
   (1) 可以產生兩個 1 至 6 的亂數。
   (2) 累加以上亂數。
   (3) 輸出此亂數與其和。
   (4) 若亂數和大於 8，則重複 (1) ～ (3)，直到亂數和小於等於 8，則程式結束。

4. 請寫一程式，可以連續產生兩個 1 ～ 6 的亂數，並輸出此兩個亂數，直到後面的亂數大於前面的亂數。

※ 5. 請寫一程式，可以連續產生 3 個 1 ～ 6 的亂數，並輸出此 3 個亂數，直到有其中兩個亂數相等爲止。

※ 6. 請寫一程式，可以連續產生 4 個 1 ～ 6 的亂數，並輸出此 4 個亂數，直到有其中兩個亂數相等爲止，並輸出此不相等的數字與和。例如，產生 6, 4, 5, 1 則繼續產生亂數，若亂數爲 6, 2, 1, 6 則其和爲 3。

4-1-5 陣列

處理少量的資料，可以爲每一筆資料設定一個變數，但若資料數量很多，例如，若有資料如下：

```
3,8,4,7,2,9
```

要求其和、極大、極小呢？是不是指派 6 個變數，當然也可以，但是程式會非常冗長，爲了改善此一問題，就要使用本節的陣列型態了。因爲陣列型態，可以使用『陣列索引』配合 for 與 while 迴圈而簡化程式，此即爲本章重點。例如：以上資料就可宣告陣列如下：

```
int a[7];
```

int 是陣列資料型態，陣列型態可爲 4-1-1 節任一型態，a 是陣列變數名稱，變數名稱也應該遵循 4-1-1 節變數命名規則。然後就可依序指派這些資料到陣列，程式如下：

```
a[1]=3;a[2]=8;a[3]=4;a[4]=7;a[5]=2;a[6]=9;
```

中括號內的 1,2,3 稱爲『陣列索引』，簡稱索引。人類索引編號通常從 1 號開始使用，但電腦卻是從 0 號開始，所以宣告如下：

```
int a[7];
```

其實是使用 0,1,2,3,4,5,6 等 7 個位置。索引 0 號可用可不用，陣列索引從 0 開始，有其獨到用法，若配合迴圈與取餘，可以執行一些無限循環的功能。例如：以下程式，可以讓陣列索引在 0 到 6 之間無限循環。

```
void loop() {
    PORTA=a[i];
    delay(1000);
    i=(i+1) %7;//保障i在0,1,2,3,4,5,6循環
}
```

陣列的存取

陣列的存取都要靠索引，例如：以下程式可重新指派 a 陣列索引 3 的值。

```
a[3]=1;
```

以下程式可將 a 陣列索引 3 的值指派給變數 b，但請留意兩者資料型態要相符。

```
int b=a[3];
```

▮陣列宣告與初值設定

單一變數可宣告變數的同時就指派初值，陣列也可以。例如：前面陣列的宣告與初值指派，程式可以簡化如下：

```
int a[7]={0,3,8,4,7,2,9};
```

本例索引 0 不用，也要給予 1 個 0，3 才會從索引 1 開始放。事實上，陣列宣告時，您可以不用指派陣列的個數，所以以下程式就可以。

```
int a[]={0,3,8,4,7,2,9};
```

但是陣列的大小給的太小也不行。例如，以下程式就不行：

```
int a[3]={0,3,8,4,7,2,9};
```

其次，給大一點當然可以，以下陣列大小給 10，初值僅給 7 個，其索引分別是 0~6，索引 7,8,9 沒給初值，其值由編譯器指派爲 0。

```
int a[10]={0,3,8,4,7,2,9};
```

陣列的資料結構的最大優點是，陣列結構可配合迴圈（前面單一變數不行），例如，以上陣列 a[] 求平均、極大值、與極小值的流程圖如圖 4-8。

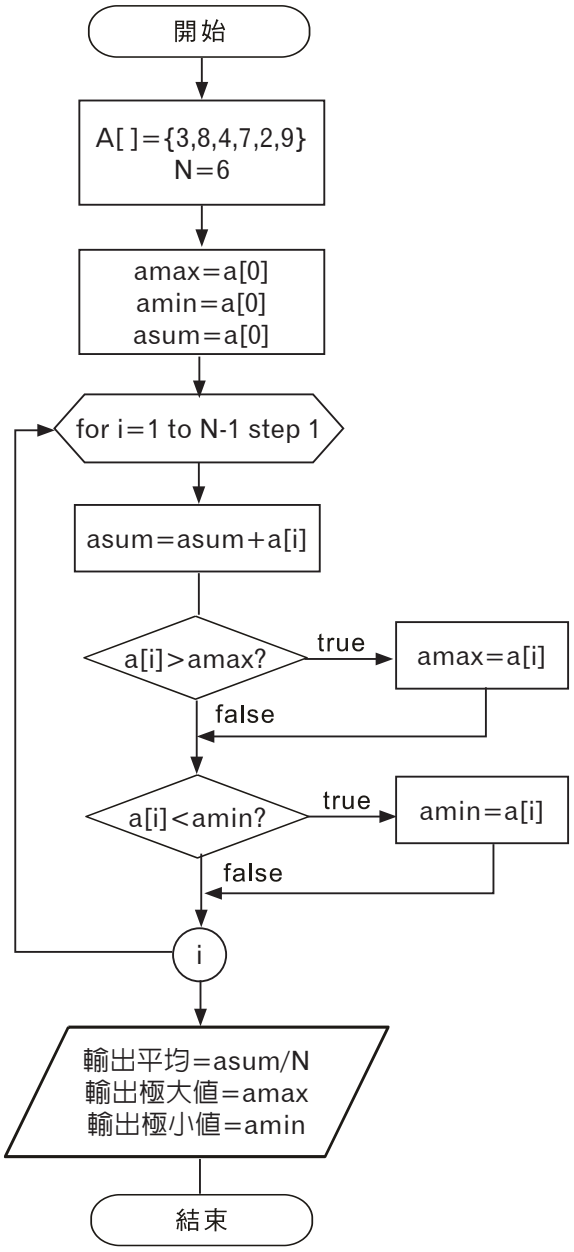


圖 4-8　搜尋極大值流程圖

根據以上流程圖，Arduino 程式如下：

```
1.  void setup() {
2.    Serial.begin(9600);
3.    int a[]={3,8,4,7,2,9};  //指派陣列初值a[0]=3,a[1]=8,a[2]=4…
4.    //有陣列就有索引，有索引就可用迴圈，有了迴圈，程式才能精簡
5.    int  amax, amin, asum;
6.    int N=6;
7.    Serial.println(N);
8.    amax = a[0];  //先指派極大值爲a[0]
9.    amin = a[0];  //先指派極小值爲a[0]
10.   asum = a[0];  //先指派和爲a[0]
11.   for (int i = 1; i <= N-1; i++) {
12.     asum += a[i]; //逐一累加所有陣列值
13.     if (a[i]>amax) //逐一探訪所有陣列值，若大於極大值，
14.       amax = a[i];//極大值被取代
15.     if (a[i]<amin) //逐一探訪所有陣列值，若小於極小值，
16.       amin = a[i]; //極小值被取代
17.   }
18.   Serial.print("average=");//輸出平均
19.   Serial.println( asum / N);
20.   Serial.print("amax=");//輸出極大值
21.   Serial.println(amax );
22.   Serial.print("amin=");//輸出極小值
23.   Serial.println(amin );
24. }
25. void loop() {
```

自我練習

1. 假設有資料如下：

   3,2,1,5,3,2,1,4,5,6

   請統計所有數字出現的次數。

2. 假設有資料如下：

   100,8,9,90,88,4,6,9,11

   請統計其中位數。（若有奇數筆資料，排序後中間那筆資料即爲中位數，若有偶數筆資料，則中間兩筆的平均爲中位數。）

4-1-6 自訂函式

在程式設計時，常會遇到某些程式片段需要在同一個程式或不同程式重複出現多次，如果這些程式片段都分別在每個地方寫一次，那是一件非常浪費時間的事，且會使程式變得冗長而不易閱讀；更糟糕的是，萬一要調整此程式片段的部分功能，還要至不同的地方修改，造成程式維護困難，此時可透過函式解決以上困難。

函式的使用方式是將此常用的程式片段賦予一個名稱，當程式設計者需要使用此程式片段時，只要使用此名稱即可呼叫此程式片段，此程式片段即稱爲『函式』，又稱爲『副程式』（物件導向的程式設計則改稱爲『方法』）。其次，函式完成之後可交由不同的程式呼叫使用，以節省程式設計者的時間。自訂函式的使用步驟如下：

步驟一：實作函式本體。
步驟二：呼叫函式。

實作函式本體

實作函式本體的簡要語法如下：

```
傳回值型態    函式名稱 ( [ 參數 1, 參數 2...] ){
    [ 敘述區塊；]
    [ return (運算式)；]
    [ 敘述區塊；]
}
```

以上語法說明如下：

1. 每個函式可放在 setup() 與 loop() 函式前面或後面均可，且每個函式的地位均相同，所以不可以在函式中再定義其它函式。
2. 函式原則上從左大括號『{』執行到右大括號『}』，但若中途有特殊原因要離開函式，可用 return 提早離開。
3. 參數（Parameter）有些書亦稱爲引數（Argument）。函式呼叫的參數名稱與實作函式本體的參數名稱可不同，但其資料具有傳遞性，也就是

函式呼叫的參數將會傳遞給函式本體，至於函式本體的運算結果是否回傳至主程式，那就得依參數的傳遞方式了，傳遞的方式有傳值、傳址與傳參考，請複習一年級的程式設計實習。

4. 以下程式片段，其函式名稱是『add』，功能是將所傳來的兩個參數相加並傳回。

```
int add (int a, int b){
    int c;
    c=a+b;
    return (c);
}
```

呼叫函式

呼叫函式的簡要語法如下：

```
[ 傳回值 =]    函式名稱 ([ 參數 1, 參數 2, ...]);
```

以上語法說明如下：

1. 函式若無傳回值，那函式會以『void』表示，則呼叫函式名稱前就不用『傳回值』接收。
2. 主程式函式的呼叫參數稱爲實際參數（Actual Parameter），函式本體的參數稱爲形式參數（Formal Parameter），且兩者的名稱可以不同。實際參數會將參數值傳給形式參數，而形式參數是否將值傳回，則要看參數傳遞方式，參數的傳遞方式有傳值、傳址及傳參考，這部分本書不予介紹。
3. 以下程式片段可呼叫 add 函式，傳回值皆爲 8：

```
c=add (6, 2);
```

或

```
m=6; n=2;
c=add (m, n);
```

以下是全部程式列印。

```
1.  int add (int a, int b){
2.    int c=a+b;
3.    return (c);
4.  }
5.  void setup() {
6.    Serial.begin(9600);
7.    int a1=3,b1=4;
8.    int c1;
9.    c1=add(a1,b1);
10.   Serial.println(c1);//7
11. }void loop() {}//雖沒用，但也不可以去掉
```

自我練習

1. 請寫一程式，請先定義一個函式，他可以將所輸入的數取絕對值。例如，函式名稱可以是 myabs()，那 a=myabs(3)，會傳回 3，a=myabs(-2)，會傳回 2。

2. 加、減、乘、除、取餘都有運算子，可否另以函式完成呢。例如，以上範例 add(3,4) 傳回 7，請製作 sub(7,2) 傳回 5，mul(3,8) 傳回 24 等等。

3. 試寫一程式，計算 C_n^m 的值。

   提示：$C_n^m = \frac{m!}{n!(m-n)!}$，m! = 1*2*3…*m。例如，$C_2^5 = \frac{5!}{2!3!} = 10$

4-2 程式編寫

一個 Arduino 程式碼基本上是由 setup() 與 loop() 函式組成，如圖 4-9 所示：

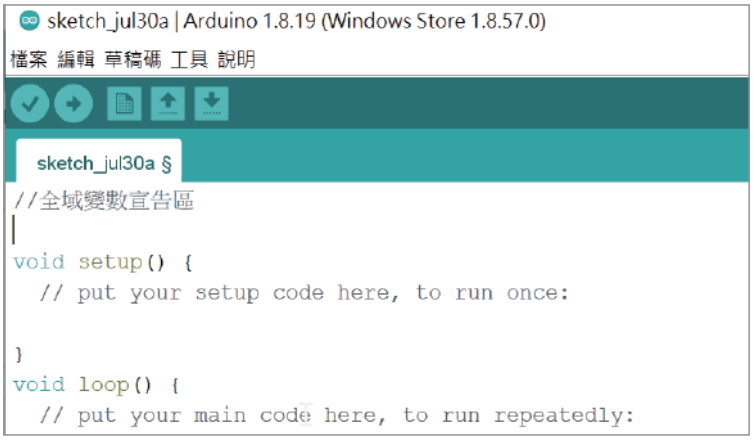


圖 4-9 Arduino 程式架構圖

微處理機啓動後，setup() 函式僅執行 1 次，loop() 函式則重複一直被執行，所以一些基本設定就放在 setup(){} 裡面，需要反覆執行的就放在 loop(){} 函式內。請鍵入以下程式，並比較執行結果。

```
void setup() {
  randomSeed(analogRead(A0));
  Serial.begin(9600);
}
void loop() {
  int x=random(1,7);
  Serial.println(x);
  delay(1000);
}
```

```
void setup() {
  randomSeed(analogRead(A0));
  Serial.begin(9600);
  int x=random(1,7);
  Serial.println(x);
  delay(1000);
}
void loop() {}
```

若是全域變數，也應該放在全域變數區，請鍵入以下程式，並比較其差別。

```
int x=0;
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.println(x);
  x=x+1;
  delay(1000);
}
```

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int x=0;
  Serial.println(x);
  x=x+1;
  delay(1000);
}
```

範例 4-2a

請寫一個程式，可以指派長方形的長與寬，並計算面積。

程式設計解題步驟

1. 資料的數位化
   (1) 本題若使用掌上型計算機，其方法是直接將鍵入「長 * 寬 =」，就可得到答案。例如，鍵入「15*8=」，就可得到答案「120」。
   (2) 但使用 Arduino 語言，必須先將所要計算的值先以一個英文代號儲存，例如 a,b 或 score，以上代號在程式設計的領域，稱爲「變數」。例如：

```
a=15
b=8
```

   設定變數的目的是將資料與程式分開，這樣當資料改變時，還可重複計算。計算過程都是以變數與程式指令描述，此稱爲程式設計。程式設計的優點是有一致性，只要第一次對，往後都對。掌上型計算機就沒有一致性了，每次都要按兩次，才能確認計算結果是否正確，且無法重複使用。
   (3) 選用資料型態。本例需要兩個變數儲存長與寬。請選擇此資料來源是整數還是浮點數，若是整數，還要分是 8 位元的 byte (0 ～ 255)、16 位元的 int (-32768 ～ 32767) 或 32 位元的 long (-2147483648 ～ 2147483647)。本書整數若沒特別註明，就一律折衷，通通取 int，且每一個敘述都要以分號 (;) 結束，所以程式如下：

```
int a=15;
int b=8;
```

2. 寫出演算法。本例是輸入長方形的長與寬來計算面積，所以演算法如下：

```
面積=a*b
```

3. 使用變數與程式指令，完成以上演算法。本例是計算面積，面積也要使用變數儲存，也要選資料型態，本例使用 c，資料型態選 int ，所以是：

```
int c=a*b
```

4. 輸出結果。

```
Serial.print("面積=");
Serial.println(c);
```

5. 以上全部程式如下：

```
1.  void setup() {
2.    Serial.begin(9600);//啓動序列埠
3.    int a=15;
4.    int b=8;
5.    int c=a*b;
6.    Serial.print("面積=");
7.    Serial.println(c);
8.  }
9.  void loop() {}
```

自我練習

1. 指派長方體的長、寬、高，計算其表面積與體積。
2. 請寫一個程式，可以指派一個台斤數，且轉爲公斤數輸出。
3. 請寫一個電子時鐘程式，可以使用序列埠視窗輸出，輸出格式爲『時：分：秒』。
4. 請寫一個倒數計時器，可以使用序列埠視窗輸出，輸出格式爲『分：秒』。

學後測驗

題號	題目	結果

1	`void loop() {`
	`  byte i=0;`
	`  i=i+1;`
	`  Serial.println(i);`
	`}`	

2	`byte i=0;`
	`void loop() {`
	`  i=i+1;`
	`  Serial.println(i);`
	`}`	

3	`byte i=0;`
	`void loop() {`
	`  i=i-1;`
	`  Serial.println(i);`
	`}`	

4	`int i=0;`
	`void setup() {`
	`  Serial.begin(9600);`
	`  i=i-1 ;`
	`  Serial.println(i);`
	`}`	

5	`int a=5,b=4;`
	`float c=4;`
	`Serial.println(a%b);`
	`Serial.println(a/b);`
	`Serial.println(a/c);`	

6	`int a=5,b=3;`
	`Serial.println(!(a>b));`
	`Serial.println((a>b) && (b>=4));`
	`Serial.println((a>b) || (b>=4));`	

7	byte a=3; Serial.println(~a); Serial.println(a^4); Serial.println(a&3); Serial.println(a|5); Serial.println(a<<3); Serial.println(a>>2);	

8	int d=0x23;
	int e=023;
	Serial.println(d);
	Serial.println(e);	

9	oid setup() {
	  Serial.begin(9600);
	  int i=3 ;
	  String b="";
	  if (i%2==0)
	  b="偶數" ;
	  else
	  b="奇數";
	  Serial.println(b);
	}void loop() {}	

10	for(int i=2;i<8;i=i+2){
	  Serial.print(i);
	  }	

11	for(int i=8;i>=1;i=i-2){
	  Serial.print(i);//8642
	  }	

12	int a=52, b=138,s=0;
	for (int i=1 ;i<=a;i++)
	  s=s+b;
	Serial.println(s);	

13	int a=18,b=5,q=0;
	while(a>=b) {
	  a-=b;//a=a-b
	  q++;//q=q+1
	  }
	Serial.print("quotient = ");Serial.
	println(q);
	Serial.print("remainder= ");Serial.
	println(a);	

14	void setup() {
	  Serial.begin(9600);
	  int a[]={13,8,4,37,2,19};
	  int amax, amin, asum; N=6;
	  amax = a[0]; amin=a[0];
	  for (int i = 1; i <= N-1; i++) {
	  if (a[i]>amax)
	  amax = a[i];
	  if (a[i]<amin)
	  amin = a[i];
	  }
	  Serial.println(amax);
	  Serial.println(amin);
	}
	void loop() {	

15	int add (int a, int b){
	  int c=a+b; return (c);
	}
	void setup() {
	  Serial.begin(9600);
	  Serial.println(add(4,6));
	}void loop() {}	

MEMO

Chapter
05

基礎應用控制

學習大綱

- 5-1　發光二極體
- 5-2　一位數七段顯示器
- 5-3　指撥開關
- 5-4　按壓開關
- 5-5　四位數七段顯示器
- 5-6　計數器
- 5-7　計時器
- 5-8　外部中斷控制

學習目標

- 1. 能使用 LED 完成耶誕燈、紅綠燈、公車停靠站等基本電路。
- 2. 能使用一位數七段顯示器顯示計數內容。
- 3. 能使用指撥開關指派所需時間長度。
- 4. 能使用按壓開關當作計數開關。
- 5. 能使用四位數七段顯示計數值。
- 6. 能完成計數器。
- 7. 能完成計時器。
- 8. 能使用輪尋法與中斷法完成計數器。

5-1 發光二極體

▶ 表 5-1　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	限流電阻 220Ω ～ 470Ω	8	

發光二極體

發光二極體（light-emitting diode，簡稱 LED）是一種**半導體**光源，當**電流**通過它時會發光，如圖 5-1 所示（摘自 https://www.ledinside.com.tw/）。其次，因爲 LED 只要 10mA(0.01A) 就很亮。但是 Arduino 輸出爲高電位時，電壓 5V，最大電流爲 20mA，所以直接用高電位驅動 LED 可說綽綽有餘，但是驅動電流若太大，LED 也會燒毀，所以要加上限流電阻如下：

$$R=\frac{5-1.7}{0.01}=330\,\Omega$$

以上 1.7(V) 稱爲 LED 的順向切入電壓，10mA=0.01A，這樣計算時，單位才一致。圖 5-2 是 LED 的符號與驅動電路，LED 長腳爲正端，若腳已經被剪斷，則往 LED 裡面其瞧，裡面有 2 個電極棒，正端的電極棒面積比較小。

圖 5-1　發光二極體實體圖

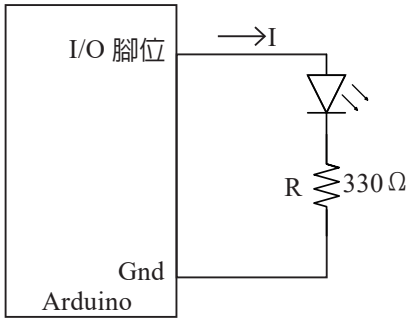


圖 5-2　發光二極體驅動電路驅動電路圖

範例 5-1a

耶誕樹燈。

小明買了一顆耶誕樹，現在要將它裝上 8 顆 LED，且讓其閃爍如表 5-2，請問如何規劃。

▶ 表 5-2　耶誕樹閃爍時序表

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
0	○	○	○	○	○	○	○	●
1	●	○	○	○	○	○	○	○
2	○	○	○	○	○	○	●	●
3	●	●	●	○	○	○	○	○
4	○	○	○	○	●	●	●	●
5	●	●	●	●	●	○	○	○
6	○	○	●	●	●	●	●	●
7	●	●	●	●	●	●	●	○
8	○	●	○	●	○	○	○	○
9	○	○	●	○	●	○	○	○

（實心代表燈亮，空心代表不亮）

單晶片微處理機系統開發流程如下：

1. 需求分析：對需求進行分析，寫出需求分析表。本例要完成一個耶誕樹閃爍燈，共需要 1 顆耶誕樹，8 個 LED 亮滅所需零件如表 5-1。

2. 總體設計：
   (1) 通過分析需求資訊，寫出輸入元件、輸入要求、輸出元件、輸出內容，最終形成概要設計說明文件。本例硬體共需 1 個 Arduino 實驗板，8 個 LED，8 個 330Ω電阻。然後依照使用者需求，以秒爲單位，列出時序圖，完成思考邏輯設計。
   (2) 將資料數位化，以秒爲單位，規劃時序圖。本例規劃如下：1 代表亮，0 代表不亮，時序 0 數位化結果是 0x01，時序 1 是 0x80，餘此類推。

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0	值
0	0	0	0	0	0	0	0	1	0x01
1	1	0	0	0	0	0	0	0	0x80
2	0	0	0	0	0	0	1	1	0x03
3	1	1	1	0	0	0	0	0	0xe0
4	0	0	0	0	1	1	1	1	0x0f
5	1	1	1	1	1	0	0	0	0xf8
6	0	0	1	1	1	1	1	1	0x3f
7	1	1	1	1	1	1	1	0	0xfe
8	0	1	0	1	0	0	0	0	0x50
9	0	0	1	0	1	0	0	0	0x28

3. 詳細設計：依照需求繪出輸入電路、輸出電路、依照時序圖完成程式設計。 本例電路設計如圖 5-3：

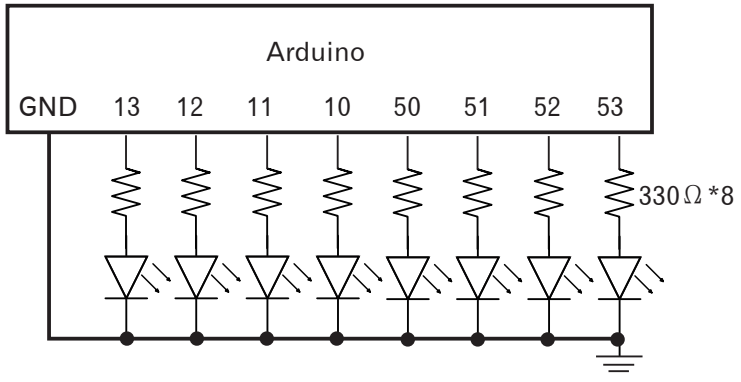


圖 5-3　耶誕樹電路圖

4. 開發程式設計：對系統進行程式碼編寫。
   (1) 將資料以陣列儲存，如以下 a[] 陣列。
   (2) 依照時序圖，使用循序法一一輸出資料。

```
1. //PORTB 13,12,11,10,50,51,52,53 接LED
2. void setup() {
3.    DDRB=B11111111;//指派PORTB爲輸出
4. }
5. byte i=0;
6. int N=10;//共10個時序
```

```
 7. void loop() {
 8.   byte a[]={0x01,0x80,0x03,0xe0,0x0f,0xf8,0x3f,0xfe,0x50,0x28};
 9.   PORTB=a[i];//將資料輸出到PORTB
10.  delay(1000);//單位是ms,延遲1秒
11.  i=(i+1)%N;//i每次遞增1，但%N可保障數字i在0～N-1之間,這樣才不會溢位
12. }
```

5. 測試分析與系統整合：對所有功能模組進行模擬資料測試及其它相關性測試並整合所有模組功能。
6. 現場支援：系統上線試執行進行現場問題記錄、解答。
7. 系統執行支援：系統正式推產後，對系統進行必要的維護和錯誤修改。

補充說明

1. 本例即爲很典型單晶片程式設計，因爲單晶片用來工業控制，使用者只要完成輸入與輸出元件就好，所有的動作變化，都是交由單晶片使用軟體指定與判斷。本例若不使用單晶片，就要學習數位邏輯設計，使用一大堆正反器設計出以上順序邏輯的變化。
2. 若使用麵包板，則實體接線如圖 5-4（電阻 330Ω 顏色是橙橙棕，LED 長腳爲正，若腳被剪斷過，則往 LED 裡面瞧，接點小的爲正端，接點大的爲負端，下圖 LED 長腳請接電阻，短腳接地）。

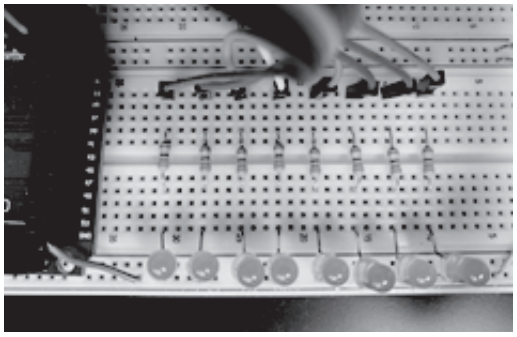


圖 5-4　麵包板接線實體圖

3. 若使用本書以下實驗板，只要將 13, 12, 11, 10, 50, 51, 52, 53 的腳位使用杜邦線連接到限流電阻 (J16)，再從限流電阻另一端 (J17) 連接到 LED (J18) 即可，Gnd 內部已自動連接，如圖 5-5。在電源指示燈正常情況下，即可進行實驗。其次，因爲微控板有過載保護，若微控板電

源指示燈熄了，實驗板指示燈也會熄滅，表示您的電路有問題，有導線短路了，微控板自動切掉電源，此時請迅速拔掉 USB 連接線，檢查電路，直到故障排除，再重新插入 USB 連接線，待指示燈正常為止。（本書實驗板的 LED 是將 10 個 LED 連續排列，稱為 LED Bar，印有型號的為正端）

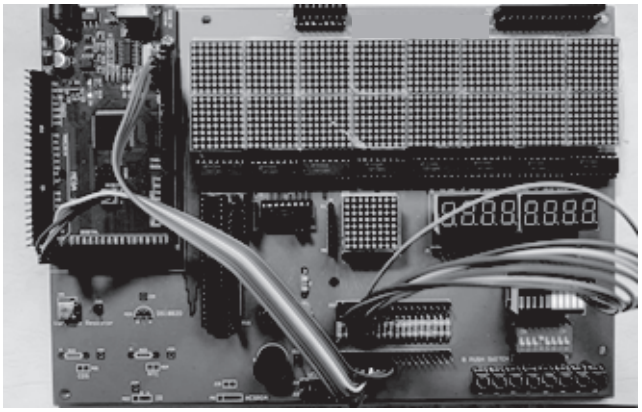


圖 5-5　實驗板接線實體圖

4. 杜邦線的顏色有意義，請依電阻的色碼使用，『棕紅澄黃綠藍紫灰』分別代表 1 到 8，『白黑』請用來連接正負電源，白接正，黑接地。
5. 若使用本書 Arduino 魔法教具實驗板，此實驗板並未準備以上 LED 電路，而是拿 8*8 點陣 LED 來當作 8 位元 LED。此實驗板內部接線如圖 5-6：

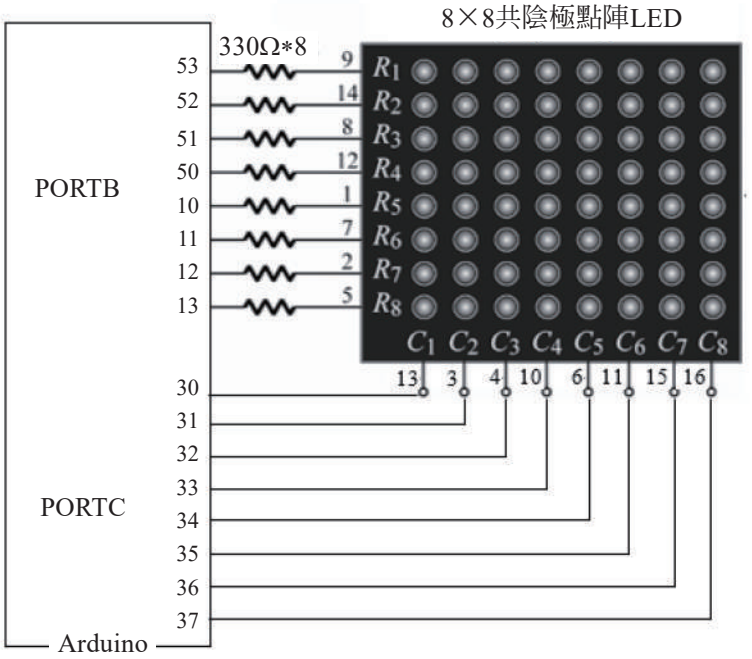


圖 5-6　Arduino 魔法教具實驗板 LED 電路圖

6. 請在以上 p5-4 頁程式加入以下第 4~5 行程式。本書往後使用任何獨立 LED 的程式也都這樣，直接使用軟體設定的方式，設定 C_1 ～ C_7 皆爲高電位，C_8 爲低電位，則此 8*8 點陣 LED，C_8 Column 就可當作 8 個 LED 來使用。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接LED
2.  void setup() {
3.     DDRB=B11111111;//指派PORTB爲輸出
4.     DDRC=0xff;      //指派PORTC爲輸出
5.     PORTC=0xfe; //0xfe is B11111110 將8*8點陣LED當作8個LED使用
6.  }
7.  byte i=0;//計數器
8.  int N=10;//共10個時序
9.  void loop() {
10.   byte a[]={0x01,0x80,0x03,0xe0,0x0f,0xf8,0x3f,0xfe,0x50,0x28};
11.   PORTB=a[i];//將資料輸出到PORTB
12.   delay(1000);//單位是ms,延遲1秒
13.   i=(i+1)%N;//i每次遞增1，但%N可保障數字i在0～N-1之間,這樣才不會溢位
14. }
```

自我練習

1. 若希望耶誕樹霹靂燈的變化如下，請寫程式完成。

時序	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0	值
0	1	0	0	0	0	0	0	1	
1	0	1	0	0	0	0	1	0	
2	0	0	1	0	0	1	0	0	
3	0	0	0	1	1	0	0	0	
4	0	0	0	1	1	0	0	0	
5	0	0	1	1	1	1	0	0	
6	0	1	1	1	1	1	1	0	
7	0	1	1	1	1	1	1	1	
8	1	1	1	1	1	1	1	1	
9	1	1	1	1	1	1	1	0	
10	1	1	1	1	1	1	0	0	
11	1	1	1	1	1	0	0	0	

12	1	1	1	1	0	0	0	0	
13	1	1	1	0	0	0	0	0	
14	1	1	0	0	0	0	0	0	
15	1	0	0	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	

2. 紅綠燈。若有一個單向紅綠燈時序如下，請寫程式完成。

時序	LED2（紅燈）	LED1（黃燈）	LED0(綠燈)	值
0	0	0	1	
1	0	0	1	
2	0	0	1	
3	0	0	1	
4	0	1	0	
5	0	1	0	
6	1	0	0	
7	1	0	0	
8	1	0	0	

3. 請自行觀察路口的雙向紅綠燈，並寫程式完成。
   提示：將所有燈號建立時序圖如下：

時序	LED5紅	LED4黃	LED3綠	LED2紅	LED1黃	LED0綠	值
0	1	0	0	0	0	1	
1	1	0	0	0	0	1	
2	1	0	0	0	0	1	
3	1	0	0	0	0	1	
4	1	0	0	0	1	0	
5	1	0	0	1	0	0	
6	0	0	1	1	0	0	
7	0	0	1	1	0	0	
8	0	0	1	1	0	0	
9	0	0	1	1	0	0	
10	0	0	1	1	0	0	

11	0	0	1	1	0	0	
12	0	0	1	1	0	0	
13	0	1	0	1	0	0	
14	1	0	0	1	0	0	

4. 高鐵停靠站燈號。高鐵的月台上，都有一個告示板，用來顯示即將到站列車的停靠站。假設以下是南港月台的列車停靠告示板，1 代表有停靠（燈亮），0 代表未停靠（燈未亮），請問您如何來設計。

列車編號	南港	台北	板橋	桃園	新竹	苗栗	台中	彰化	嘉義	台南	高雄	PORTK	PORTF
	PORTK			PORTF									
0	1	1	0	0	0	0	1	0	0	1	1	0x6	0x13
1	1	1	1	1	1	1	1	1	1	1	1	0x7	0xff
2	1	1	0	0	1	0	1	0	1	1	1	0x6	0x57
3	1	1	0	0	1	0	1	0	0	0	1	0x6	0x51

5. 捷運到站燈號。搭乘捷運時，車上都有列車停靠站顯示說明。例如，淡水線的站名如下，每一個站名都對應 1 個 LED，且於列車行駛中，即將到站的站名 LED 都會亮起，請問您如何設計。

時序	淡水	紅樹林	竹圍	關渡	忠義	復興崗	北投	奇岩	其哩岸	石牌	明德	芝山	士林	劍潭	圓山	民權西路	雙連	中山	台北車站	台大醫院	中正紀念堂	東門	大安	安和	101	象山
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6. 教學機。本單元的 LED 您可以拿來作爲順序教學的指示燈。例如，霓虹燈、鉤毛線順序、電子琴、跳舞機、CPR 急救順序、葉問拳法練習椿等，也就是您可以用 LED 來提醒使用者要按哪裡、踩哪裡、打哪裡等等等。請自己思考身邊的應用，寫程式完成。以下是筆者完成的電子琴教學機影片。https://youtu.be/CsvtgIQ7Vx0

資料數位化方法

程式設計最常見的資料爲數值、字元、字串，我們必須先將以上資料轉爲二進位的方式，才能儲存於電腦，以上資料的數位化方法，分別說明如下：

▮正整數

所有整數都要先指定資料的長度，才能數位化，本例先假設是 8 位元，其餘 16 位元或 32 位元也是相同原理。其次，還要假設僅儲存正數，還是同時儲存正負整數。首先，若待處理的資料，其範圍通通在 0 ～ 255，所以可以用 1 個 byte 儲存，則其二進位編碼如表 5-3，共可表示 0 到 255 的正整數。

▶ 表 5-3　正數二進位編碼

正數	二進位編碼
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
254	11111110
255	11111111

範例 5-1b

示範正整數的數位化。請鍵入以下程式，並觀察 8 個 LED 的亮滅，請問與上表的正數是否相符。

輸出結果

請留意 8 個 LED 代表 8 位元，燈號『亮』就是 1，燈號『滅』就是 0。

實習步驟

1. 本例電路同範例 5-1a 的圖 5-3。
2. 將 12 以 2 進位正整數編碼，並填寫如下。

3. 鍵入以下程式，並觀察 12 的 8 個燈號輸出結果。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接LED
2.  void setup() {
3.    DDRB=B11111111;  //指派PORTB功能爲輸出
4.    Serial.begin(9600);//啓動序列埠
5.  }
6.  unsigned char i=12; //宣告i爲8位元正整數,這樣i可儲存0~255的整數
7.  void loop() {
8.    PORTB=i;
9.    Serial.println((int)i);//轉爲整數
10. }
```

4. 請鍵入以下程式，並觀察序列埠視窗與 LED 的燈號，驗證 0 到 255 的數位化。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  void setup() {
4.    DDRB=B11111111;
5.    Serial.begin(9600);
6.    DDRC=0xff;PORTC=0xfe;//0xFE is B11111110 將點陣LED當作8個
7.                         LED使用
8.  }
```

```
9.  unsigned char i=0; //宣告i爲8位元正整數,這樣i可儲存0~255的整數
10. void loop() {
11.   PORTB=i; //使用LED顯示記憶體內容
12.   Serial.println((int)i);//轉爲整數，使用序列埠輸出記憶體內容
13.   delay(1000);
14.   i=(i+1)%256;//保障在0到255循環
15. }
```

自我練習

請自行將 112 轉爲二進位，並觀察結果是否相符。

正負整數

若要考慮正負整數，則可用最高位元則用來表示正或負數，最高位元爲 0 時表示正數，所以可表示的範圍是 0 到 127 如表 5-4。

▶ 表 5-4　0～127 的編碼

二進位編碼	數字
00000000	0
00000001	1
00000010	2
01111110	126
01111111	127

最高位元爲 1 時表示負數，所以負數編碼如表 5-5：

▶ 表 5-5　負數的編碼

二進位編碼	數字
10000000	-0
10000001	-1
10000010	-1
11111110	-126
11111111	-127

使用以上方式編碼，會有 0 與 -0 的問題，產生對應重複的問題，此種編碼方式樣不是一對一函式，若要兩邊互轉會比較麻煩。所以計算機前輩就繼續腦力激盪，想出二補數的編碼，且沿用到今天。二補數的編碼如下：正數就直接編碼，共 7 位元，可編出 0~127，此與純正數編碼相同；負數就取其 2 補數，如表 5-6。以 -3 爲例，先將 3 轉爲二進位

```
00000011
```

取 1 補數

```
11111100
```

加 1，所以是：

```
11111101
```

也就是 2 補數是 1 補數加 1。等一下我們用燈號證明，您將會看到『亮，亮，亮，亮，亮，亮，滅，亮』。同理，看到燈號爲

```
11111101
```

最高位元爲 1，表示負數，那到底負多少？就取 2 補數，2 補數是先取 1 補數（0 變 1，1 變 0 稱爲 1 補數）再加 1。以上

```
11111101
```

先取 1 補數

```
00000010
```

加 1

```
00000011
```

此即爲 -3。所以 10000000 代表 -128，10000001，代表 -127,…11111111 代表 -1，以上即爲 2 補數的編碼，如下表所示，這樣就沒有正負 0 的問題了。其次，2 補數的編碼竟然也同時解決減法問題，因爲

```
8-3
```

可以看成

```
8+(-3)
```

也就是直接將 3 取 2 補數，再相加就可以。因爲計算機內部組織只有加法器與比較器，請繼續複習一年級的程式設計實習，您會發現乘法也是可以用加法完成，除法也是用加減法完成。

▶ 表 5-6　正負數的編碼

二進位編碼	數字
00000000	0
00000001	1
00000010	2
00000011	3
00000100	4
00000101	5
00000110	6
00000111	7
01111110	126
01111111	127 最大正數
10000000	-128 最小負數
10000001	-127
11111110	-2
11111111	-1

範例 5-1C

示範包含正負整數的數位化。請鍵入以下程式，並觀察 8 個 LED 的亮滅，請問與上表的正負數是否相符？（電路同範例 5-1b）

實習步驟

1. 以 -3 爲例，先將 3 轉爲二進位

```
00000011
```

取 1 補數

```
11111100
```

加 1，所以是

```
11111101
```

2. 看到 11111101，最高位元爲 1，表示此爲負數，到底負多少，也是取 2 補數，先取 1 補數如下：

```
00000010
```

再加 1，結果是

```
00000011
```

所以是 -3。

3. 請鍵入以下程式，並觀察 -3 的燈號是否與以上相同。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  void setup() {
4.    DDRB=B11111111;
5.    DDRC=0xFF;PORTC=0xFE;
6.    //0xFE is B11111110 將8*8點陣LED當作8個LED使用
7.    Serial.begin(9600);
8.  }
9.  char i=-3; //宣告i爲8位元正負整數,這樣i可表示-128~127
10. void loop() {
11.   PORTB=i;
12.   Serial.println((int)i);//轉爲整數
13. }
```

補充說明

本例 Serial.println((int)i);(int) 爲將字元型態轉爲整數，請修改爲 Serial.println(i), 並觀察執行結果。

自我練習

請自行將 -112 轉爲二補數，並觀察結果是否正確。

字元

鍵盤能用的字元有大小寫的 a,b,c、數字 0 ～ 9、還有一些控制字元，例如，跳列、歸位（回到該列最左邊）、跳頁、return 等，這些字元總共沒有超過 127 個，因此使用 1 個 byte 儲存就綽綽有餘。所以這些字元的編碼，在 1960 年就已經編碼完成，稱爲 ASCII 碼 (American Standard Code for Information Interchange)，如表 5-7 所示：也就是我們將所有字元編號，此編號與我們的座號相同，當叫到 65 號，大家都能體認此爲字元『A』，當叫到 33 號，就表示此爲字元『！』。

▶ 表 5-7　ASCII 編碼

DEC Value	Character	DEC Value	Character	DEC Value	Character	DEC Value	Character
0	null	32	space	64	@	96	`
1		33	!	65	A	97	a
2		34	"	66	B	98	b
3		35	#	67	C	99	c
4		36	$	68	D	100	d
5		37	%	69	E	101	e
6		38	&	70	F	102	f
7		39	'	71	G	103	g
8		40	(	72	H	104	h
9	tab	41	)	73	I	105	i
10	line feed	42	*	74	J	106	j
11		43	+	75	K	107	k
12		44	,	76	L	108	l
13	carriage return	45	-	77	M	109	m
14		46	.	78	N	110	n
15		47	/	79	O	111	o
16		48	0	80	P	112	p
17		49	1	81	Q	113	q
18		50	2	82	R	114	r
19		51	3	83	S	115	s
20		52	4	84	T	116	t
21		53	5	85	U	117	u
22		54	6	86	V	118	v

由上圖可知共使用 7 位元，0 到 127，前面 32（0 ～ 31）爲控制字元，48 ～ 57 是 0 ～ 9 的數字，65 ～ 90 是大寫英文字元，97 ～ 122 是小寫英文字元。

範例 5-1d

示範英文小寫字元的表示。請自行鍵入以下程式，並觀察燈號與輸出結果。（電路同範例 5-1b）

輸出結果

1. 請留意字元 'a' 的表示，且內部是以 $(97)_{10}=(61)_{16}$ 表示，燈號的明滅就是其二進位，本例會是『滅，亮，亮，滅，滅，滅，滅，亮』，代表 01100001。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  void setup() {
4.    DDRB=B11111111;
5.    Serial.begin(9600);
6.    DDRC=0xFF;PORTC=0xFE;
7.    //0xFE is B11111110 將8*8點陣LED當作8個LED使用
8.  }
9.  char i ='a';//字元用單引號  ;
10. void loop() {
11.   Serial.print((int)i);Serial.print(":");Serial.println(i);
12.   PORTB=i;//97,0x61
13.   delay(1000);
14. }
```

自我練習

1. 請自行於 Arduino 線上手冊搜尋『ASCII』。
2. 請將 a 改爲大寫 A，並觀察記憶體內容，填入下表：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

3. 請將 char i='a' 改爲 char i=1，並觀察記憶體內容，填入下表。

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

4. 請將 char i='a' 改爲 char i='1'，並觀察記憶體內容，填入下表，請問是否與 ACCII 的符號 1 相同。

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

▮字串

連續字元的集合稱爲字串，例如：

```
"ABC"
```

請留意前面字元用單引號，本例字串則用雙引號『"』。C 語言使用字元陣列表示字串。例如，以下陣列 a 即可儲存以上字串。

```
char a[]="ABC";//字串用雙引號
```

變數經過以上宣告，往後就可以使用索引存取內部字元。請鍵入以下程式，並開啓序列埠視窗，觀察執行結果。

```
void setup() {
  Serial.begin(9600);
  char a[]="ABC";
  Serial.println(a);//ABC
  Serial.println(a[0]); //A
  Serial.println(a[1]); //B
  Serial.println(a[2]);//C
  a[0]='D'; //可修改
  Serial.println(a);//DBC
}void loop() {}
```

C++ 則新增字串型態 string。例如：

```
String a="ABC";//字串用雙引號
```

變數經過以上宣告，往後就可以使用索引存取內部字元。請鍵入以下程式，並開啓序列埠視窗，觀察執行結果。

```
void setup() {
  Serial.begin(9600);
  String a="ABC";
  Serial.println(a);//ABC
  Serial.println(a[0]); //A
  Serial.println(a[1]); //B
  Serial.println(a[2]);//C
  a[0]='D';//可修改
  Serial.println(a);//DBC
}void loop() {}
```

以上兩種字串表示，Arduino 都可以接受，如以上程式。

範例 5-1e

示範字串的表示。請自行鍵入以下程式，並觀察燈號與輸出結果。（電路同範例 5-1b）

輸出結果

1. 請留意 a[0] 是字元 'A'，編號是 65，內部記憶體是 01000001(=0x41)，燈號會是 01000001。
2. a[1] 是字元 'B'，編號是 66(0x42=B01000010)，燈號會是 01000010。
3. a[2] 是字元 'C'，編號是 67(0x43=B01000011)，請觀察其燈號。

```
65:A
66:B
67:C
```

程式列印

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  void setup() {
4.    DDRB=B11111111;
5.    DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將8*8點陣LED當作8個LED使用
6.    Serial.begin(9600);
7.  }
```

```
8.  void loop() {
9.    //char a[]="ABC";//字串用雙引號
10.   String a="ABC";
11.   Serial.print((int)a[0]); //輸出a[0]的ASCII碼到序列埠
12.   Serial.print(":"); Serial.println(a[0]);
                              //輸出a[0] ASCII碼所代表的字元到序列埠
13.   PORTB=a[0]; //輸出a[0]的ASCII碼到LED
14.   delay(1000);
15.   Serial.print((int)a[1]);Serial.print(":");
16.   Serial.println(a[1]);
17.   PORTB=a[1];//66,0x42
18.   delay(1000);
19.   Serial.print((int)a[2]);Serial.print(":");
20.   Serial.println(a[2]);
21.   PORTB=a[2];//67,0x43
22.   delay(12000);
23. }
```

5-2 一位數七段顯示器

▶ 表 5-8　本節零件表

編號	零件名稱	數量	備註
1	一位數共陰極七段顯示器	1	
2	限流電阻 220Ω ～ 470Ω	8	

七段顯示器是由 8 個編號分別是 a , b , c ,d , e , f , g , dp 的 LED 所構成，此 8 個 LED 排列成「日」字，其中 dp 是小數點，如圖 5-8。LED 排列成「日」字，8 個 LED 照理應有 16 隻腳，但爲了簡化腳位 (腳位簡化的優點是省元件的體積與簡化電路)，廠商是將所有 LED 的陽極或陰極共接再拉出，所以分爲共陽極或共陰極七段顯示器，兩者原理都相同。但是，因爲 Arduino 高電位的電流有 20mA，可以直接驅動 LED，所以本書就介紹共陰極如下：

範例 5-2a

共陰極七段顯示器的基本操作。

1. 共陰極七段顯示器的內部電路與腳位圖如圖 5-7（大部分的共陰與共陽七段顯示器腳位排列都相同）。
2. 請自行使用傳統指針型三用電表測量，先撥歐姆檔 ×10，例如黑棒 (+) 接 a，紅棒 (-) 接 com 點，則上面的 a 棒亮。
3. 共陰極七段顯示器的驅動電路如圖 5-8，請完成此電路。

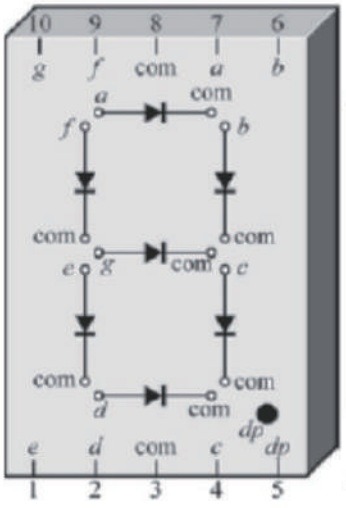


圖 5-7　七段顯示器內部電路圖

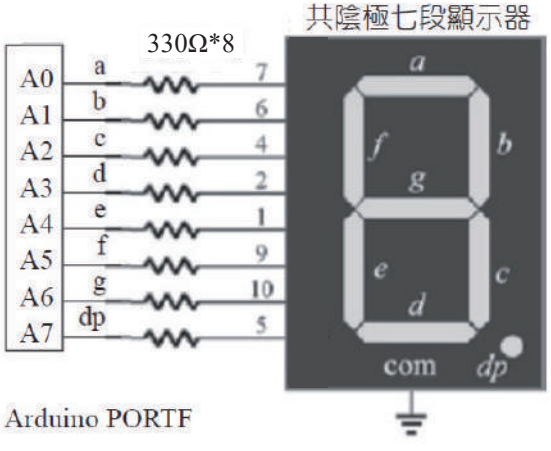


圖 5-8　七段顯示器驅動電路圖

4. 本書教學實驗板並未準備一位數七段顯示器，請將八位數七段顯示器的最右邊的 2com4 接地，就可以當作一位數七段顯示器用，如圖 5-9，其次，a, b, c, d, e, f, g, dp 的腳位本書實驗板已經按順序由左而右排列。

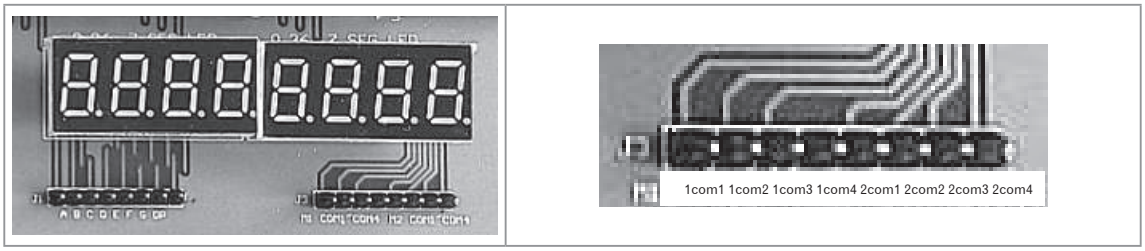


圖 5-9　教學實驗板七段顯示器電路圖

5. 以下程式，可讓 8 個 LED 全亮 1 秒、接著全滅 1 秒，且循環。

```
1.  //PORTF A0~A7接七段顯示器的a,b,c,d,e,f,g,dp
2.  void setup() {
3.    DDRF=0ff;//指派PORTF腳位功能是輸出
4.  }
5.  void loop() {
6.    PORTF=0xff; //指派PORTF腳位電壓爲高電位，電壓5V
7.    delay(1000);
8.    PORTF=0; //指派PORTF腳位電壓爲低電位，電壓0V
9.    delay(1000);
10. }
```

6. 若使用內部接線的魔法教具實驗板，其電路如圖 5-10：

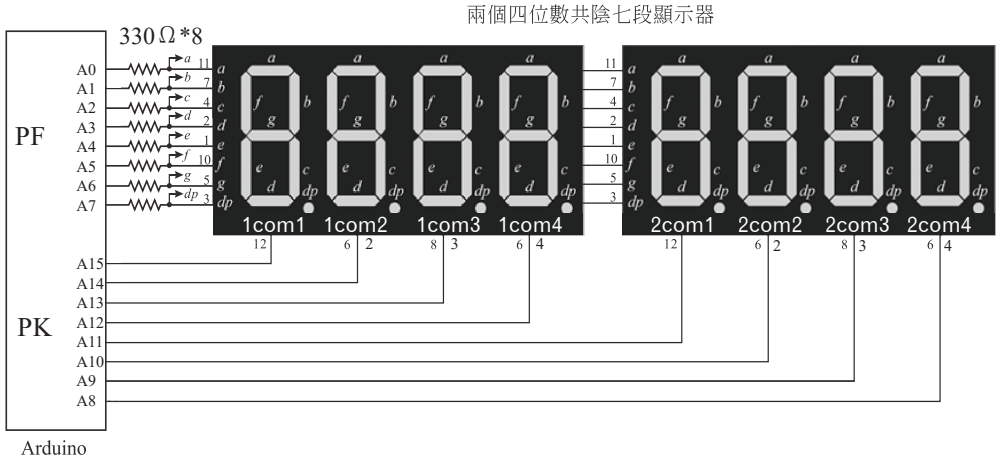


圖 5-10　魔法教具實驗板七段顯示器電路圖

7. 以下我們加入兩行斜體字的程式，以軟體設定的方式，僅讓最右邊的2com4 爲低電位，其餘均爲高電位，所以功能也同上，本書往後程式，也都加上這兩行，將以上 8 位數七段顯示器，拿來當作一位數使用。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
4.    DDRF=0xff;//指派PORTF腳位功能是輸出
5.    DDRK=0xff; //指派PORTK腳位功能是輸出
6.    PORTK=B11111110;  //0xfe，僅最右邊一隻腳com4爲低電位
7.    //DDRK=0xff;PORTK=0xfe;//將四位數七段顯示器當作一位數
8.  }
9.  void loop() {
10.   PORTF=0xff; //指派PORTF腳位電壓爲高電位，電壓5V
11.   delay(1000);
12.   PORTF=0; //指派PORTF腳位電壓爲低電位，電壓0V
13.   delay(1000);
14. }
```

8. 請鍵入以下程式，並觀察此七段顯示器 a,b,c,d,e,f,g,dp 等 8 個 LED 的排列。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
4.    DDRF=B11111111;
5.    DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
6.  }
7.  void loop() {
8.    byte a[]={0x01,0x03,0x07,0x0f,0x1f,0x3f,0x7f,0xFF,0};//8
9.    byte b[]={0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80,0};//
10.   byte c[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f,0};//10
11.   for (byte i=0;i<=8;i++){
12.       PORTF=a[i];
13.       delay(300);
14.   }
15.   for (byte i=0;i<=8;i++){
16.       PORTF=b[i];
17.       delay(500);
```

```
18.   }
19.   for (byte i=0;i<=10;i++){
20.       PORTF=c[i];//0,1,2,3,4,5,6,7,8,9,空白
21.       delay(800);
22.   }
23. }
```

9. 由以上程式執行結果可知，共陰極要顯示 0 到 9，所要輸出的電壓如表 5-9，1 代表亮，0 代表不亮。例如，要顯示 0，則 a, b, c, d, e, f 等腳位要給高電壓（亮），g 與 dp 要給低電位（不亮），所以資料要輸出 0x3f。

▶ 表 5-9　七段顯示器 0~9 編碼

數字	dp	g	f	e	d	c	b	a	十六進制數值
0	0	0	1	1	1	1	1	1	0x3f
1	0	0	0	0	0	1	1	0	0x6
2	0	1	0	1	1	0	1	1	0x5b
3	0	1	0	0	1	1	1	1	0x4f
4	0	1	1	0	0	1	1	0	0x66
5	0	1	1	0	1	1	0	1	0x6d
6	0	1	1	1	1	1	0	1	0x7d
7	0	0	0	0	0	1	1	1	0x7
8	0	1	1	1	1	1	1	1	0x7f
9	0	1	1	0	1	1	1	1	0x6f

10. 例如，以下程式可以重複循環顯示 0, 1, 2 各一秒鐘。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
4.     DDRF=B11111111;
5.     DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
6.  }
7.  void loop() {
8.      PORTF=0x3f;
9.      delay(1000);
10.     PORTF=0x6;
```

```
11.     delay(1000);
12.     PORTF=0x5b;
13.     delay(1000);
14. }
```

11. 查表顯示數字。前面我們要顯示 0，就輸出 0x3f；要顯示 1，就輸出 0x6；要顯示 2 就輸出 0x5b，程式有點冗長，因爲若要顯示 10 個數字，程式就更冗長了，所以通常將以上數字以陣列存放如下：

```
byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
```

現在只要指派陣列索引，就可以得到對應輸出碼，此即爲陣列索引查表功能。例如，要顯示 1，程式如下：

```
PORTF=a[1];
delay(1000);
```

要顯示 5，程式如下：

```
PORTF=a[5];
delay(1000);
```

12. 有了陣列就可使用迴圈簡化程式。例如：以下程式，配合迴圈，可以重複顯示 0 到 9 各 1 秒。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
4.  int i=0;
5.  void setup() {
6.     DDRF=B11111111;
7.     DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
8.  }
9.  void loop() {
10.    PORTF=a[i];  //PORTF
11.    delay(1000);
12.    i=(i+1) %10;  //共10個
13. }
```

自我練習

1. 同範例 5-2a，如何於兩個數字之間，顯示一個空白呢？

範例 5-2b

連續數字的顯示。例如，顯示學號是 825710。

實習步驟

1. 本例假設學號是 825710，則要分別送出 825710 對應的數值 0x7f, 0x5b, 0x6d, 0x7, 0x6, 0x3f，所以將以上資料放在陣列如下：

```
byte b[]={0x7f,0x5b,0x6d,0x7,0x6,0x3f};
```

2. 根據範例 5-2a，將 b 陣列指派給 PORTF 即可，所以程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte b[]={0x7f,0x5b,0x6d,0x7,0x6,0x3f};
4.  byte len=6;
5.  byte i=0;
6.  void setup() {
7.     DDRF=B11111111;
8.     DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
9.  }
10. void loop() {
11.   PORTF= b[i];
12.   delay(1000);
13.   i=(i+1) %len;
14. }
```

3. 前面程式是自己將 825710 查表，轉爲七段顯示器的輸出碼，但其實應該讓微處理機自己查表，程式才會簡單。例如，將 825710 放在陣列，如下：

```
byte b[]={8,2,5,7,1,0};
```

4. 然後程式如下，此稱爲陣列中的陣列。

```
PORTF= a[b[i]];
```

例如，i=0，b[0]=8，則 a[8]=0x7f，也就是送出 0x7f，就顯示『8』。

5. 全部程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
4.  byte b[]={8,2,5,7,1,0};
5.  byte len=6;//b陣列長度
6.  byte i=0;
7.  void setup() {
8.     DDRF=B11111111;
9.   DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
10. }
11. void loop() {
12.   PORTF= a[b[i]];
13.   delay(1000);
14.   i=(i+1) %len;
15. }
```

6. 有時候若是數字相同，例如，88255，每顯示完一個數字，若沒有先熄掉，再輸出下一個數字，看起來會是 825，所以程式改善如下：
先輸出一個空白，將 LED 熄滅，再顯示下一個數字。

```
void loop() {
  PORTF= a[b[i]];
  delay(800);
  PORTF=0;//顯示空白
  delay(200);
  i=(i+1) %len;
}
```

補充說明

1. 程式設計的領域，很多都是重複循環，請多利用陣列索引從 0 開始的特性，且配合取餘（%）運算，才能簡化程式的撰寫。其次，陣列索引從 0 開始，只是給使用者方便，不用 0 開始也沒關係。

自我練習

1. 請將今天的日期，用一個七段顯示器逐一顯示。提示：先將 0 到 9 的輸出碼以陣列存放，日期分隔號「"-"」的輸出碼是「0x40)」放在索引 10，如下：

```
byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f,0x40};
```

   假設今天的日期是『2019-04-26』，也以陣列存放如下：(10 就會對應到 0x40，將顯示 "-"。)

```
byte b[]={2,0,1,9,10,0,4,10,2,6};
```

2. 請將現在的時間，用一個七段顯示器逐一顯示。（提示：資料以單一變數或陣列儲存，請自己思考）

範例 5-2c

如何輸出 1 個四位數整數。例如，int a=1028; 程式如何撰寫？

實習步驟

1. 您要發揮國小數學能力，充分利用整數除法與取餘運算，將 1 個 4 位數，逐一分解成 4 個位數。程式如下：

```
int c=1028;//1028已經超過255，所以要宣告爲int
d=c/1000;//千
c=(c-d*1000);
d=c/100;//百
c=c-d*100;
d=c/10;//十
d=c%10;//個位數
```

2. 再將此四位數一一查表，找出對應輸出碼（陣列的查表功能），再輸出，全部程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
```

```
4.     DDRF=B11111111;
5.     DDRK=0xFF;PORTK=0xFE;
6.     Serial.begin(9600);
7.  }
8.  void loop() {
9.    byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f,0x40};
10.   int c=1028;//1028已經超過255，不能宣告爲byte, 要宣告爲int
11.   byte d=c/1000;//整數除法，得到千位數
12.   PORTF=a[d];//顯示千位數
13.   delay(1000);
14.   c=(c-d*1000);
15.   d=c/100;//百位數
16.   PORTF=a[d];//顯示百位數
17.   delay(1000);
18.   c=c-d*100;
19.   d=c/10;//十位數
20.   PORTF=a[d];//顯示十位數
21.   delay(1000);
22.   d=c%10;//個位數
23.   PORTF=a[d];//顯示個位數
24.   delay(1000);
25. }
```

3. 以上是數字長度固定，題目限定四位數，所以應該善用迴圈，才能簡化程式，因爲 for 迴圈通常用在題目設計階段就已經知道迴圈次數，所以程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
4.  void setup() {
5.     DDRF=B11111111;
6.     DDRK=0xFF;PORTK=0xFE;//將四位數七段當作一位數
7.     Serial.begin(9600);
8.  }
9.  void loop() {
10.   int c=1028;//1028已經超過255，不能宣告爲byte, 要宣告爲int
11.   int d;
12.   int e=1000;
13.   for (int i=3;i>=0;i--){
```

```
14.     d=c/e;
15.     Serial.println(d);
16.     c=c-d*e;
17.     PORTF=a[d];
18.     e=e/10;
19.     delay(500) ;
20.   }
21. }
```

4. 以下程式就不行，因爲 Arduino 的實數運算有誤差，請鍵入以下程式，寫出執行結果。

```
1.  void loop() {
2.    int c=1028;//1028已經超過255，不能宣告爲byte, 要宣告爲int
3.    int d;
4.    int e;
5.    for (int i=3;i>=0;i--){
6.      e=pow(10,i);//10的i次方，但實數運算有誤差，沒有得到整數1000,100,10
7.      Serial.println(e);
8.  }
```

自我練習

1. 同範例 5-2c，數字出現後，請閃爍一下。
2. 同範例 5-2c，可以顯示 3 位數，且含有 1 個小數點。例如，103.2。
3. 於單向紅綠燈的系統中（假設僅有紅黃綠三個燈，且秒數均小於 10），請用七段顯示器顯示紅燈與綠燈倒數剩餘秒數。

範例 5-2d

示範產生 3 個亂數，且同時於序列埠視窗與七段顯示器輸出。

程式列印

1. 程式初步設計如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
```

```
4.  void setup(){
5.    Serial.begin(9600);
6.    randomSeed(analogRead(0));
7.    DDRF=B11111111;
8.    DDRK=0xff;PORTK=0xfe;//將四位數七段當作一位數
9.  }
10. void loop() {
11.   int r;
12.   r = random(1, 7); // produce a random number from 1 to 6
13.   Serial.println(r);// 輸出在序列埠(滿足題目的需求)
14.   PORTF=a[r];       // 輸出在PORTF(滿足題目的需求)
15.   delay(1000);
16.   r = random(1, 7);
17.   Serial.println(r);// 輸出在序列埠(滿足題目的需求)
18.   PORTF=a[r];       // 輸出在PORTF(滿足題目的需求)
19.   delay(1000);
20.   r = random(1, 7);
21.   Serial.println(r);
22.   PORTF=a[r];
23.   delay(1000);
24.   delay (1000);
25. }
```

2. 數字很容易重複，不易觀察，所以應該先熄掉，再顯示下一個數字，程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
4.  void setup(){
5.    Serial.begin(9600);
6.    randomSeed(analogRead(0));
7.    DDRF=B11111111;
8.    DDRK=0xFF;PORTK=0xFE;//將四位數七段當作一位數
9.  }
10. void loop() {
11.   int r;
12.   r = random(1, 7); // produce a random number from 1 to 6
13.   Serial.println(r);
14.   PORTF=a[r];
15.   delay(1000);
16.   PORTF=0;       //熄滅
```

```
17.   delay(200);
18.   r = random(1, 7);
19.   Serial.println(r);
20.   PORTF=a[r];
21.   delay(1000);
22.   PORTF=0;       //熄滅
23.   delay(200);
24.   r = random(1, 7);
25.   Serial.println(r);
26.   PORTF=a[r];
27.   delay(1000);
28.   PORTF=0;       //熄滅
29.   delay(200);
30.   delay (1000);
31. }
```

for迴圈

於範例 5-2d，產生 3 次亂數，這 3 次亂數的程式都相同，處理機有一個指令 for，它可以簡化程式的撰寫，所以我們可以用一個 for 迴圈替代，如以下程式：

```
1.  //setup()同範例5-2d
2.  void loop() {
3.    int r;
4.    for (byte i=1;i<=3;i++){
5.      r = random(1, 7);// print a random number from 1 to 6
6.      Serial.println(r);
7.      PORTF=a[r];
8.      delay(1000);
9.      PORTF=0;     //滅
10.     delay(200);
11.   }
12.   delay (1000);
13. }
```

亮滅

連續明滅稱爲亮滅，以下程式，我們再用一個 j 迴圈讓它連續亮滅 2 次。這樣迴圈裡面有迴圈，就稱爲巢狀迴圈。

```
1.  //setup()同範例5-2d
2.  void loop() {
3.    int r;
4.    for (byte i=1;i<=3;i++){
5.      r = random(1, 7);// print a random number from 1 to 6
6.      Serial.println(r);
7.      PORTF=a[r];
8.      delay(600);
9.      for (byte j=1 ;j<=2;j++){
10.       PORTF=a[r];
11.       delay(100);
12.       PORTF=0;     //滅
13.       delay(100);
14.     }
15.     delay(200);
16.   }
17.   delay (1000);
18. }
```

自我練習

1. 請寫一程式，可以幫您產生一個 1 到 4 的亂數，然後依據亂數對應擲骰子次數。例如，產生亂數 3，那就連續擲 3 次骰子。因爲處理機是連續執行，所以每一回合結束，請停頓 3 秒。
2. 請寫一個程式，可以產生一個 0 到 999 的亂數，然後使用一位數七段輪流顯示。
3. 請寫一個程式，可以產生一個 -9.9 到 9.9 含小數點一位的亂數，然後使用一位數七段輪流顯示。若是負數，請以「-」表示。

5-3 指撥開關

▶ 表 5-10　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	一位數共陰極七段顯示器	1	
3	限流電阻 220Ω ～ 470Ω	16	
4	指撥開關	8	

前面的 LED 與七段顯示器都是輸出設備，本單元起要開始介紹一些輸入設備，例如：指撥開關與按壓開關。

數位輸入

Arduino 當輸入時，有兩種可能，一種是接收其他 IC 的輸出，這時只要設定其爲 INPUT 即可，但若要接收按壓開關或指撥開關等開關裝置，則可設定其具有上拉電阻 (INPUT_PULLUP)，這樣可以簡化外部電路。開關的標準電路如圖 5-11a，此 20kΩ 稱爲上拉電阻（使用 10k~50k 都可以），按鍵沒按壓是高電位，壓下去是低電位。其次，Arduino 爲了讓使用者簡化電路，此上拉電阻可用軟體指派，所以使用者只要接一個開關就好，如下圖 5-11b。（補充說明，沒上拉電阻可以嗎？當然不可以，因爲開關沒按壓時，Arduino 輸入腳位浮接（懸空未接稱爲浮接），此時就無法判定其電壓的高與低了。

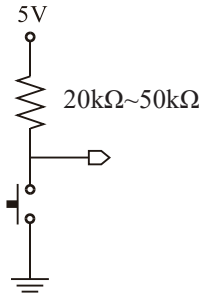


圖 5-11a　開關電路圖

圖 5-11b　開關驅動電路

指撥開關

指撥開關實體如圖 5-12a，內部結構如圖 5-12b。開關可上下滑動，圖 5-12b 是下滑，電路處於斷路；圖 5-12c 往上滑動，則電路接通。往後爲了簡化電路，均以圖 5-12d 表示指撥開關。

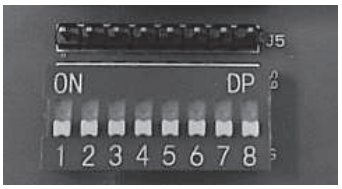


圖 a

圖 b

圖 c

圖 d

圖 5-12　指撥開關實體圖

範例 5-3a

示範使用指撥開關。

實習步驟

1. 完成圖 5-13 電路，PORTB 接 8 個 LED，用來顯示 PORTL 8 位元指撥開關指撥值。
2. 指撥開關的實驗板實體圖如圖 5-12a，杜邦線從 J5 連接到 Arduino 微控板對應腳位即可，Gnd 內部已經連接。

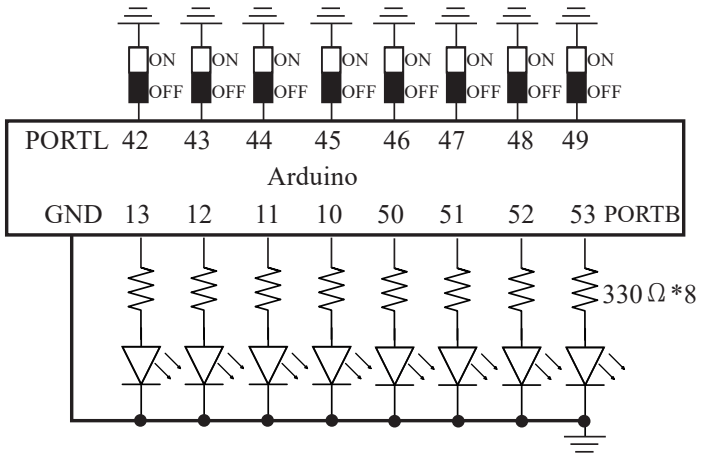


圖 5-13　指撥開關驅動電路

3. digitalRead() 一次僅讀取一個指定腳位電壓，請鍵入以下程式，並觀察執行結果。依據電路圖的連接方式，請留意當指撥開關 ON 時，其輸入值爲低電位（LOW），因此輸出之 LED 爲熄滅。

```
1.  //PORTL 42,43,44,45,46,47,48,49 接8個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  void setup() {
5.      Serial.begin(9600);
6.      pinMode(49,INPUT_PULLUP);//PL0
7.      DDRB=B11111111;PORTB=0;//PORTB接LED
8.      DDRC=0xff;PORTC=0xfe;//將點陣LED當作8個LED使用
9.  }
10. void loop() {
11.     byte b=digitalRead(49);//讀一個位元
12.     Serial.println(b);
13.     digitalWrite(53,b);    //寫入一個位元
14. }
```

4. 本例指撥開關接到 PORTL，也可以用 PINL 讀取 PORTL 暫存器 8 隻腳全部電壓。請鍵入以下程式，並觀察執行結果，請留意開關 ON，其值是 LOW，LED 滅掉。

```
1.  //PORTL 42,43,44,45,46,47,48,49 接8個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  void setup() {
5.      Serial.begin(9600);
6.      pinMode(42,INPUT_PULLUP);//PL7
7.      pinMode(43,INPUT_PULLUP);//PL6
8.      pinMode(44,INPUT_PULLUP);//PL5
9.      pinMode(45,INPUT_PULLUP);//PL4
10.     pinMode(46,INPUT_PULLUP);//PL3
11.     pinMode(47,INPUT_PULLUP);//PL2
12.     pinMode(48,INPUT_PULLUP);//PL1
13.     pinMode(49,INPUT_PULLUP);//PL0
14.     DDRB=B11111111; PORTB=0x00;//PORTB接LED
15.     DDRC=0xFF;PORTC=0xFE;//0xFE is B11111110 將點陣LED當作8個LED使用
16. }
```

```
17. void loop() {
18.     byte b=PINL;//讀8個位元
19.     Serial.println(b);
20.     PORTB=b;
21. }
```

5. 請將以上程式的 8 個 INPUT_PULLUP 通通改爲 INPUT，程式如下，並觀察執行結果。

```
pinMode(37,INPUT);//PC0
```

6. 指撥開關通常可用來作開關，或運作模式的選擇，請看以下範例說明。

範例 5-3b

開關控制。指撥開關可以用來當開關，以下電路使用 3 位元指撥開關，分別控制 3 個 LED 的明滅，且 ON 時才亮。

實習步驟

1. 電路同範例 5-3a。
2. 使用 digitalRead() 函式，每次讀取 1 位元，且逐一控制對應 LED，程式如下：

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  void setup() {
5.      Serial.begin(9600);
6.      pinMode(47,INPUT_PULLUP);//PL2
7.      pinMode(48,INPUT_PULLUP);//PL1
8.      pinMode(49,INPUT_PULLUP);//PL0
9.      DDRB=0xFF;     PORTB=0;
10.     DDRC=0xFF;PORTC=0xFE;//將點陣LED當作8個LED使用
11. }
12. void loop() {
13.     byte b;
14.     b=digitalRead(47);//讀一個位元
15.     Serial.println(b);//輸出看看，確認是否正確
```

```
16.     if (b==0)//==  比較是否相等運算子
17.       digitalWrite(51,HIGH); //用常數符號表示高電位5V
18.     else
19.       digitalWrite(51,LOW);   //表示低電位0V
20.     b=digitalRead(48);//讀一個位元
21.     Serial.println(b);//輸出看看，確認是否正確
22.     if (b==0)//== 是比較是否相等運算子
23.       digitalWrite(52,HIGH); //用常數符號表示高電位5V
24.     else
25.       digitalWrite(52,LOW);   //表示低電位0V
26.     b=digitalRead(49);//讀一個位元
27.     Serial.println(b);//輸出看看，確認是否正確
28.     if (b==0)//== 是比較是否相等運算子
29.       digitalWrite(53,HIGH); //用常數符號表示高電位5V
30.     else
31.       digitalWrite(53,LOW);   //表示低電位0V
32. }
```

3. 以上我們用腳位編號寫程式，但有可能您要改變腳位，此時您就要到處修改，萬一沒有完全修改，會造成程式執行不一致 , 所以我們通常將這些腳位，用常數符號（以下簡稱常數）代替，寫在程式最前面，這樣當您要改變腳位時，只要在最前面修改就行，不用到處修改。現在，將以上腳位定義常數，程式如以下第 5 ～ 8 行：(這是單晶程式的習慣，因爲您有可能要換到不同型號的單晶執行，不同型號的可用腳位不同，或電路變大，不同元件的腳位衝突等，都要改變腳位編號。)

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  const byte s2=47;//switch2
5.  const byte s1=48;
6.  const byte s0=49;
7.  const byte l2=51;//led2
8.  const byte l1=52;//led1
9.  const byte l0=53;//led0
10. void setup() {
11.     Serial.begin(9600);
12.     pinMode(s2,INPUT_PULLUP);//PL2
```

```
13.     pinMode(s1,INPUT_PULLUP);//PL1
14.     pinMode(s0,INPUT_PULLUP);//PL0
15.     DDRB=B11111111;     PORTB=0;
16.     DDRC=0xFF;PORTC=0xFE;//將點陣LED當作8個LED使用
17. }
18. void loop() {
19.     byte b;
20.     b=digitalRead(s2);//讀一個位元
21.     Serial.println(b);//輸出看看，確認是否正確
22.     if (b==0)//== 是比較是否相等運算子
23.       digitalWrite(l2,HIGH); //用常數符號表示高電位5V
24.     else
25.       digitalWrite(l2,LOW);    //表示低電位0V
26.     b=digitalRead(s1);//讀一個位元
27.     Serial.println(b);//輸出看看，確認是否正確
28.     if (b==0)//== 是比較是否相等運算子
29.       digitalWrite(l1,HIGH); //用常數符號表示高電位5V
30.     else
31.       digitalWrite(l1,LOW);    //表示低電位0V
32.     b=digitalRead(s0);//讀一個位元
33.     Serial.println(b);//輸出看看，確認是否正確
34.     if (b==0)//== 是比較是否相等運算子
35.       digitalWrite(l0,HIGH); //用常數符號表示高電位5V
36.     else
37.       digitalWrite(l0,LOW);    //表示低電位0V
38. }
```

4. 使用 PINL 讀取 3 位元。本例一次讀 8 位元，但是我們僅用 3 個位元，沒有用到的 bit7, 6, 5, 4, 3 等 5 位元，請用位元運算子的 and 運算子『&』強制遮罩掉，如程式第 24 行。（這是單晶控制的技巧，將不要的位元，用 and 0 去掉，要的位元用 and 1 留著。）

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  const byte s2=47;//switch2
5.  const byte s1=48;
6.  const byte s0=49;
7.  const byte l2=51;//led2
```

```
8.  const byte l1=52;//led1
9.  const byte l0=53;//led0
10.
11. void setup() {
12.     Serial.begin(9600);
13.     pinMode(s2,INPUT_PULLUP);//PL2
14.     pinMode(s1,INPUT_PULLUP);//PL1
15.     pinMode(s0,INPUT_PULLUP);//PL0
16.     DDRB=B11111111;     PORTB=0;
17.     DDRC=0xFF;PORTC=0xFE;//將點陣LED當作8個LED使用
18. }
19. void loop() {
20.     byte b;
21.     b=PINL;//讀8個位元
22.     Serial.print(b);//輸出看看，確認是否正確
23.     Serial.print(',');
24.     b=b & B00000111;//遮罩沒用到的位元7,6,5,4,3
25.     Serial.println(b);//輸出看看，確認是否正確
26.     switch (b){
27.       case (B000):
28.         digitalWrite(l2,HIGH);
29.         digitalWrite(l1,HIGH);
30.         digitalWrite(l0,HIGH);
31.         break;//語法，不可漏掉
32.       case (B001):
33.         digitalWrite(l2,HIGH);
34.         digitalWrite(l1,HIGH);
35.         digitalWrite(l0,LOW);
36.         break;//不可漏掉
37.       case (B010):
38.         digitalWrite(l2,HIGH);
39.         digitalWrite(l1,LOW);
40.         digitalWrite(l0,HIGH);
41.         break;//不可漏掉
42.       case (B011):
43.         digitalWrite(l2,HIGH);
44.         digitalWrite(l1,LOW);
45.         digitalWrite(l0,LOW);
46.         break;//不可漏掉
47.       case (B100):
```

```
48.         digitalWrite(l2,LOW);
49.         digitalWrite(l1,HIGH);
50.         digitalWrite(l0,HIGH);
51.         break;//不可漏掉
52.       case (B101):
53.         digitalWrite(l2,LOW);
54.         digitalWrite(l1,HIGH);
55.         digitalWrite(l0,LOW);
56.         break;//不可漏掉
57.       case (B110):
58.         digitalWrite(l2,LOW);
59.         digitalWrite(l1,LOW);
60.         digitalWrite(l0,HIGH);
61.         break;//不可漏掉
62.       case (B111):
63.         digitalWrite(l2,LOW);
64.         digitalWrite(l1,LOW);
65.         digitalWrite(l0,LOW);
66.         break;//不可漏掉
67.     }
68. }
```

自我練習

1. 請用 3 位元指撥開關，當作 2 進位的輸入，OFF 代表 0，ON 代表 1，且用七段顯示器輸出其值。例如，000，輸出 0；110，輸出 6。
2. 運作模式的選擇。請寫一個程式，讓您的單向紅綠燈可以用 1 位元指撥開關設定 2 種運作模式，2 種運作模式如下：

模式	綠燈時間	紅燈時間
0	5 秒	5 秒
1	閃黃燈	閃紅燈

3. 運作模式的選擇。請寫一個程式，讓您的單向紅綠燈可以用 2 位元指撥開關設定 4 種運作模式，4 種運作模式如下：

模式	綠燈時間	紅燈時間
0	5 秒	5 秒
1	7 秒	3 秒
2	4 秒	6 秒
3	閃黃燈	閃紅燈

範例 5-3c

利用多個位元指撥開關的切換用來輸入數字。

指撥開關可用來輸入一個數字，本例使用 3 個指撥開關，第 1 個指撥開關代表 1，第 2 個指撥開關代表 2，第 3 個指撥開關代表 4，這樣就可以輸入 0 到 7，且只要改變指撥開關值，七段顯示器亦同步顯示輸入值。

實習步驟

1. 電路同範例 5-3a。
2. 單晶片的程式都要養成一個好習慣，那就是使用的腳位，要取一個名稱。例如，本例使用腳位 49，就將腳位 49 取一個名稱 s0 如下：

```
const byte s0=49;//1
```

這樣，往後的程式都使用 s0 表示腳位 49。例如：

```
pinMode(s0,INPUT_PULLUP);
byte s0val=digitalRead(s0);
```

腳位編號使用常數取代，程式的移植性才會高。因爲別人拿到程式時，有可能在不同型號的晶片執行，因爲不同型號的腳位數目不同，或有些腳已經另有用途，此時只要在最前面改腳位編號就好，不用在程式中到處改變腳位。

3. 以下是完整程式。

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  //指派腳位
```

```
5.  const byte s0=49;//1
6.  const byte s1=48;//2
7.  const byte s2=47;//4
8.  byte b[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f, 0x67,0x0};//資料
9.  void setup() {
10.    DDRF=0xFF;
11.    DDRK=0xFF; PORTK=0xFE;//將四位數七段當作一位數使用
12.    pinMode(s0,INPUT_PULLUP); //指派腳位功能爲具有上拉電阻的輸入
13.   // pinMode(s0,INPUT);//這樣要自己接上拉電阻，不然不行
14.    pinMode(s1,INPUT_PULLUP);
15.    pinMode(s2,INPUT_PULLUP);
16. }
17. int t=0;
18. void loop() {
19.     t=0;
20.     byte s0val=digitalRead(s0);
21.     if (s0val==LOW)
22.         t=t+1;
23.     byte s1val=digitalRead(s1);
24.     if (s1val==LOW)
25.         t=t+2;
26.     byte s2val=digitalRead(s2);
27.     if (s2val==LOW)
28.         t=t+4;
29.     Serial.print(t);//顯示指撥開關所設定的初值
30.     PORTF=b[t];//查表的動作，顯示0就要輸出0x3f，顯示1就要輸出0x6
31.     PORTK=0xFE;
32. }
```

範例 5-3d

同上範例，但是程式一執行，還能倒數計時，結束時停留 3 秒，然後重複以上動作。

程式列印

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  int t=0;//請留意不能用byte,因爲byte 沒有-1
```

```
5.  void loop() {
6.      t=0;
7.      byte s0val=digitalRead(s0);
8.      if (s0val==LOW)
9.          t=t+1;
10.     byte s1val=digitalRead(s1);
11.     if (s1val==LOW)
12.         t=t+2;
13.     byte s2val=digitalRead(s2);
14.     if (s2val==LOW)
15.         t=t+4;
16.     Serial.print(t);//顯示指撥開關所設定的初值
17.     while (t>=0)  {  //只要t>0，就重複
18.       PORTF=b[t];//查表的動作，顯示0就要輸出0x3f，顯示1就要輸出0x6
19.       delay(1000);
20.       t=t-1;
21.     }
22.     delay(3000);
23. }
```

以上是停留 3 秒，若是把

```
delay(3000);
```

換成

```
while(true){}
```

就可等待到使用者按一下單晶片內置的『reset』鍵，待下一單元介紹按壓開關，就可換成自己的『重置』按鈕。以下程式，計時結束，還可持續閃爍。

```
while(true){
       PORTF=b[10];
       delay(500);
       PORTF=b[0];
       delay(500);
}
```

自我練習

1. 博奕程式。請使用 6 個指撥開關，僅撥第 1 個代表 1…, 僅撥第 6 個代表 6。程式一執行，就產生 1 到 6 的亂數，若指撥開關的數字與亂數產生的數字相等，則 8 個 LED 全亮，若數字不相等，則 LED 僅亮其中一個。本例因爲沒有開關，所以每一次請用 3 秒鐘隔開。

範例 5-3e

表決器。假設有一項評審工作，有 3 位評審，當其中兩人或以上同意，則表示過關且燈亮，請設計此電路。

設計步驟

1. 本例先用指撥開關當作評審按鈕，當有 2 個或以上評審 on 時，表示通過，LED 亮，電路設計如圖 5-14：（亦可用按壓開關比較方便，請看下一單元）

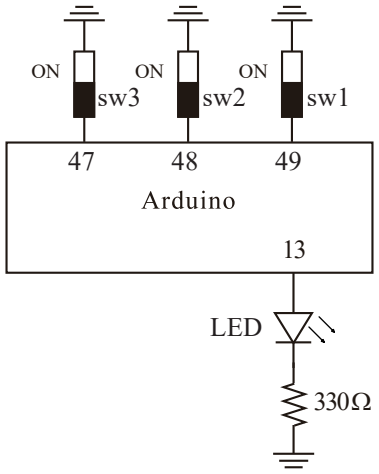


圖 5-14　表決器電路圖

2. 以上想法的程式列印如下：

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  const byte sw1=49;
3.  const byte sw2=48;
4.  const byte sw3=47;
5.  const byte led=13;  //Arduino微控板預植的LED
```

```
byte a;
byte b;
void setup() {
    pinMode(sw1,INPUT_PULLUP);
    pinMode(sw2,INPUT_PULLUP);
    pinMode(sw3,INPUT_PULLUP);
    pinMode(led,OUTPUT);
    digitalWrite(led,LOW);
    DDRF=B111;//24,23,22
    Serial.begin(9600);
}
void loop() {
    a=PINL;
    a=~a;//逐位元反相
    a=a & B00000111;//將不要的位元遮罩
    PORTF=a;
    Serial.print(a);Serial.print(":");
    b=0;
    b=b+bitRead(a,0); //讀取a值2進位位元0
    b=b+bitRead(a,1); //讀取a值2進位位元1
    b=b+bitRead(a,2); //讀取a值2進位位元2

    Serial.println(b);
    if (b>=2)
        digitalWrite(led,HIGH);
    else
        digitalWrite(led,LOW);
    delay(500);
}
```

3. 以上程式第 26~28 行的 bitRead(a,n) 函式可傳回 a 值轉爲 2 進位的 bit n 的值（bit 0 在最右邊）。請鍵入以下程式，寫出執行結果。

```
void setup() {
    Serial.begin(9600);
    byte a=bitRead(9,0);
    Serial.println(a);  //1
    a=bitRead(9,1);
    Serial.println(a);  //0
  }void loop() {}
```

4. 以上作法，程式較簡單，但是評審完，評審還要自己將指撥開關撥回來，有點不方便。待下一單元介紹按壓開關，再用按壓開關重做本範例。

自我練習

1. 同上範例，但每個人增加一個燈號，顯示自己的指撥狀態。
2. 假如有 5 位評審，三位以上通過就通過，請設計電路與程式。

範例 5-3f

選秀表決器。電路同範例 5-3e，但每個人才只有 5 秒鐘可以表演，兩個人（含）以上按鈕，LED 亮，才表示入選。5 秒鐘到了，沒有通過則 LED 持續閃爍，直到使用者按單晶片的「Reset」鍵。（本例 5 秒只是方便測試，實務上，請自行調整表演時間）

程式列印

```
1.  //PORTL 47,48,49 接3個指撥開關
2.  const byte sw1=49;
3.  const byte sw2=48;
4.  const byte sw3=47;
5.  const byte led=13;  //預植的LED
6.  byte a;
7.  byte b;
8.  void setup() {
9.      pinMode(sw1,INPUT_PULLUP);
10.     pinMode(sw2,INPUT_PULLUP);
11.     pinMode(sw3,INPUT_PULLUP);
12.     pinMode(led,OUTPUT);
13.     digitalWrite(led,LOW);
14.     Serial.begin(9600);
15.     DDRB=B11111111;
16.     DDRC=0xFF;PORTC=0xFE;
17.                     //0xFE is B11111110 將8*8點陣LED當作8個LED使用
18. }
19. long t1,t2;
20. void loop() {
21.   t1=millis();//取系統時間，單位是ms
22.   digitalWrite(led,LOW);
```

```
23.   do{ //開始計時
24.     a=PINL;
25.     a=~a;//逐位元反相
26.     a=a & B00000111;//將不要的位元遮罩
27.     PORTF=a;
28.     Serial.print(a);Serial.print(":");
29.     b=0;
30.     b=b+bitRead(a,0);//讀取位元0，且累加此位元
31.     b=b+bitRead(a,1); //讀取位元1，且累加此位元
32.     b=b+bitRead(a,2);//讀取位元2，且累加此位元
33.     Serial.println(b);
34.     if (b>=2)
35.         digitalWrite(led,HIGH);
36.     else
37.         digitalWrite(led,LOW);
38.     t2=millis();
39.   }while(  (t2-t1)<5000);//時間到
40.   if (b>=2){
41.         digitalWrite(led,HIGH);
42.   }
43.   else{
44.      while(true){//無窮迴圈，持續閃爍，直到使用者按單晶片的『reset』
45.           digitalWrite(led,HIGH);
46.           delay(200);
47.           digitalWrite(led,LOW);
48.           delay(200);
49.      }
50.   }
51. }
```

程式說明

以上程式第 23 行的 millis() 可傳回程式執行後所經過的時間，單位是 ms，請鍵入以下程式，寫出執行結果。

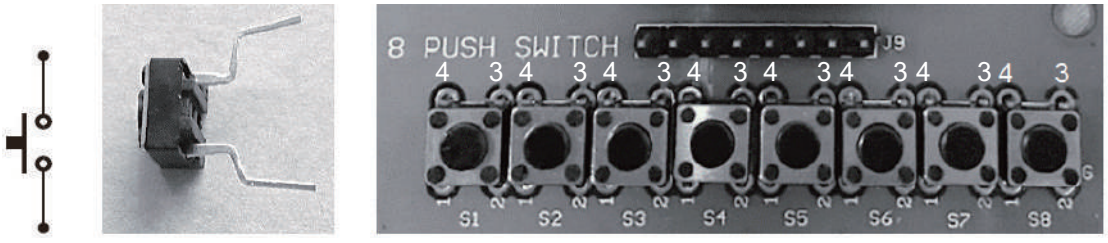
```
void setup() {
    Serial.begin(9600);
    long t=millis();
    Serial.println(t);  //0
    delay(1000);        //單位ms
    t=millis();
    Serial.println(t);  //1000
  }void loop() {}
```

5-4 按壓開關

▶ 表 5-11　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	一位數共陰極七段顯示器	1	
3	限流電阻 220Ω ～ 470Ω	16	
4	指撥開關	8	
5	按壓開關	8	

前面的指撥開關是用手指滑動開關，控制 ON 與 OFF，但是按壓開關則是按下去時 ON，放開時，有彈簧協助，又自動回到 OFF 狀態。按壓開關電路示意圖如圖 5-15a，實體圖如圖 5-15b。還有，按壓開關有兩隻腳，如圖 5-15b，也有四隻腳，如圖 5-15c，兩隻腳的是插麵包板用，請買腳長一點的，這樣可將腳撐開，如圖 5-15b 另一隻腳才可插到麵包板的地線，這樣佈線較簡單；四隻腳的如圖 5-15c，內部 2,3 兩點相通，1,4 也兩點相通，請用三用電表檢驗，以免使用到成雙連接那兩隻，那就永遠是 ON，沒有按壓效果了。通常 1,3 爲 1 組，或 2,4 爲 1 組，沒按不導通，按鍵按下去則導通。四隻腳的沒辦法插麵包板，若買到了，只好剪掉多餘的兩隻腳。圖 5-15c 則是本書實驗板實體圖。



圖a　　圖b　　圖c

圖 5-15　按壓開關實體圖

範例 5-4a

示範使用按壓開關。

實習步驟

1. 完成圖 5-16 電路，其中 LED 同範例 5-1a、按壓開關麵包板接法如圖 5-17。
2. 若使用本書實驗板，請連接 J9 到微控板即可，如圖 5-15c，Gnd 內部已經連接。

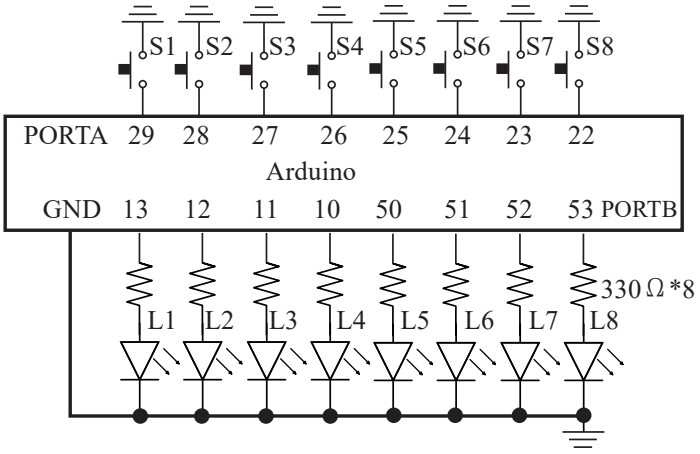
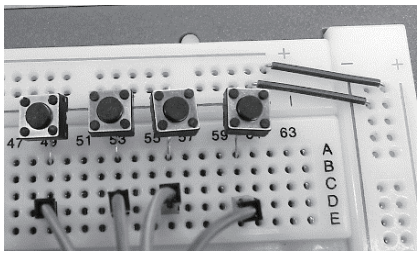


圖 5-16　按壓開關驅動電路圖



5-17　按壓開關麵包板接線圖

3. 鍵入以下程式，並觀察執行結果，請留意開關 ON，其值是 LOW，LED 滅掉。

```
1.  //PORTA 29~22 接8個按壓開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  void setup() {
5.    pinMode(22,INPUT_PULLUP);//PA0
6.    DDRB=B11111111;
7.    DDRC=0xFF;PORTC=0xFE;
8.    Serial.begin(9600);
9.  }
10. void loop() {
11.     byte a=digitalRead(22);
12.     Serial.println(a);
13.     digitalWrite(53,a);
14. }
```

3. 本例 29,28,27,26,25,24,23,22 稱爲 PORTA，也可使用 PINA 一次讀取 8 位元，請鍵入以下程式，並觀察執行結果，請留意開關 ON，其值是 LOW，LED 滅掉。待會介紹蜂鳴器，也是利用此電路製作電子琴。

```
1.  //PORTA 29~22 接8個按壓開關
2.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
3.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
4.  void setup() {
5.    pinMode(29,INPUT_PULLUP);//PA7
6.    pinMode(28,INPUT_PULLUP);//PA6
7.    pinMode(27,INPUT_PULLUP);//PA5
8.    pinMode(26,INPUT_PULLUP);//PA4
9.    pinMode(25,INPUT_PULLUP);//PA3
10.   pinMode(24,INPUT_PULLUP);//PA2
11.   pinMode(23,INPUT_PULLUP);//PA1
12.   pinMode(22,INPUT_PULLUP);//PA0
13.   DDRB=B11111111;//29~22
14.   DDRC=0xFF;PORTC=0xFE;
15.   Serial.begin(9600);
16. }
17. void loop() {
18.     byte a=PINA;          //讀取PORTA所接8個按壓開關的值
19.     Serial.println(a);    //將值輸出在序列埠
20.     PORTB=a;              //將值輸出在PORTB所接8個LED
21. }
```

自我練習

1. 請用 1 個按壓開關，分別控制 1 個 LED 的明滅，且按下去 LED 才亮。
2. 請用 8 個按壓開關，分別控制 8 個 LED 的明滅，且按下去 LED 才亮。

範例 5-4b

計數器。請寫一個程式，每當按壓開關被按一下，計數值加 1。

操作步驟

1. 本例按壓開關電路同範例 5-4a，但僅用編號 22，名稱命名爲 pb。
2. 本例使用七段顯示器輸出計數值，電路同範例 5-2a。

3. 本例也使用 8 個 LED 顯示計數值，電路同範例 5-1a。
4. 程式設計初步如下：

```
1.  //PORTA 22 接按壓開關
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A11~A8 接七段1com1,1com2,1com3,1com4
4.  const byte pb=22;
5.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
6.  void setup() {
7.      pinMode(pb,INPUT_PULLUP); //按壓開關
8.      Serial.begin(9600);
9.      DDRF=B11111111;
10.     DDRK=0xFF;PORTK=0xFE;            //將四位數七段當作一位數
11. }
12. byte num=0;
13. byte b;
14. void loop() {
15.     b=digitalRead(pb);          //讀取按壓開關
16.     if (b==LOW)
17.         num++;
18.     Serial.println(num);
19.     PORTF=a[num];                    //將計數值輸出在七段顯示器
20. }
```

5. 每按一下按鍵，計數值卻遞增很多，那是因爲單晶速度太快了，當您按下按鈕的短短時間內，單晶已經重複偵測很多次，而且開關接觸瞬間，開關接點還會有彈跳現象（on 與 off 之間連續快速反覆變化），如圖 5-18 所示。

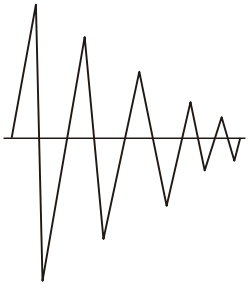


圖 5-18　開關彈跳波形圖

在此彈跳期間單晶也會重複偵測到此按壓開關很多次的 on 與 off，有些負載如馬達等會吃不消，很容易壞掉。本例使用 while 迴圈延遲一段時間，跳過這些彈跳電壓，程式如以下第 17 ～ 18 行。

```
1.  //PORTA 22 接按壓開關
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A11~A8 接七段1com1,1com2,1com3,1com4
4.  const byte pb=22;
5.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
6.  void setup() {
7.      pinMode(pb,INPUT_PULLUP);
8.      Serial.begin(9600);
9.      DDRF=B11111111;
10.     DDRK=0xFF;PORTK=0xFE;//將四位數七段當作一位數
11. }
12. byte num=0;
13. byte b;
14. void loop() {
15.     b=digitalRead(pb);
16.     if (b==LOW){
17.         while(digitalRead(pb)==LOW)
18.             delay(10);
19.         num++;
20.     }
21.     Serial.println(num);
22.     PORTF=a[num];
23. }
```

6. 也就是當按壓開關未被放開時，使用 while 迴圈繼續等待，直到按鍵被釋放，程式再繼續執行，這樣計數值才不會一直被灌水。本例的 while 迴圈如下：

```
while(digitalRead(pb)==LOW)
       delay(10);
```

和以下程式效果相同，只是 while 迴圈僅執行一個敘述，所以大括號省略。

```
while(digitalRead(pb)==LOW){
    delay(10);
}
```

7. 本例使用軟體克服電路彈跳問題。但是，非單晶的傳統數位電路，則要用電容與電阻作一個微分電路，以便排除此一彈跳電壓。
8. 以下程式，我同時用 LED 與七段顯示計數值。（請在 PORTF 連接七段顯示器，在 PORTB 連接 8 個 LED）

```
1.  //PORTA 22 接按壓開關
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
4.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
5.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
6.  const byte pb=22;//按壓開關
7.  byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
8.  void setup() {
9.      pinMode(pb,INPUT_PULLUP);
10.     Serial.begin(9600);
11.     DDRF=B11111111;
12.     DDRK=0xFF;PORTK=0xFE;//將四位數七段當作一位數
13.     DDRB=B11111111;
14.     DDRC=0xFF;PORTC=0xFE;//將8*8點陣LED當作8個LED使用
15. }
16. byte num=0;
17. byte b;
18. void loop() {
19.     b=digitalRead(pb);//讀按鍵時否被按
20.     if (b==LOW){
21.         while(digitalRead(pb)==LOW)
22.             delay(10);
23.         num++;
24.     }
25.     Serial.println(num);//輸出在序列埠視窗
26.     PORTF=a[num];//輸出在七段顯示器
27.     PORTB=num;//輸出在LED
28. }
```

自我練習

1. 請安排 2 個按壓開關，其功能分別是遞增一、遞減一。（本題使用序列埠視窗輸出）

2. 同上題，但輸出請用八個 LED 顯示 a 的值，計數值 1 就亮 1 個 LED，2 就亮 2 個 LED，依此類推。
3. 同上題，但使用一個七段顯示器顯示其值，此即爲叫號器了。沒有印表機也沒關係，請用一疊紙先寫好數字，讓消費者抽。待 5-5 節，再繼續介紹 4 位數叫號器。

範例 5-4c

請使用一個按壓開關控制 LED 的明亮，按一下時亮，且自保持，再按一下時滅，且自保持。

思考步驟

1. 本例爲了能記憶目前的狀態且自保持，我們需要一個變數 state，記錄且保留目前的狀態。

```
bool state=false;
```

2. 當按壓開關被按時，狀態改變。程式如下：

```
state=!state;//此爲布林的反相，!true =false，請看下一單元
```

程式列印

```
1.  //22接pb按壓開關
2.  //13爲內接LED
3.  const byte pb=22;//按壓開關
4.  const byte led=13;//微控板預植LED
5.  bool state=false;
6.  byte b;
7.  void setup() {
8.      pinMode(pb,INPUT_PULLUP);
9.      pinMode(led,OUTPUT);
10.     digitalWrite(led,state);
11. }
12. void loop() {
13.     b=digitalRead(pb);
14.     if (b==LOW)       {
```

```
15.         while(digitalRead(pb)==LOW)
16.             delay(10);
17.         state=!state;
18.     }
19.     digitalWrite(led,state);
20. }
```

補充說明

1. 請鍵入以下程式，並觀察執行結果。

```
void loop() {
    b=digitalRead(pb);
    if (b==LOW)       {
        state=!state;
    }
    digitalWrite(led,state);
}
```

2. 本例與範例 5-4b 相同，因爲單晶速度太快，而且有接點彈跳問題，都要延遲一段時間。

自我練習

1. 請用 2 個按壓開關控制一個 LED 的明滅，其中一個按下去亮，且自保持；另一個按下去滅掉，且自保持。
2. 請用 3 個按壓開關控制 3 個 LED，通通是按一下亮，且自保持，再按一下滅，且自保持。
3. 請用 2 個按壓開關控制 8 個 LED 的左旋或右旋。
4. 請用 1 個按壓開關控制 8 個 LED 的左旋或右旋，每按一下，改變狀態。

※ 5. 請寫一程式，使用 1 個按壓開關，控制三個 LED，讓使用者在一個短暫的時間內，按一下時亮一個燈；按兩下時亮兩個燈；按三下時亮三個燈；按四下時全熄滅。

※ 6. 請寫一程式，使用 1 個按壓開關，控制三個 LED，讓使用者在一個短暫的時間內，按 1 秒時亮一個燈；按 2 秒時亮兩個燈；按 3 秒時亮三個燈；按 4 秒時全熄滅。

7. 假設單向紅綠燈有兩種運轉模式，一種是紅燈、綠燈各 5 秒，另一種是一邊閃紅、一邊閃黃，請寫一個程式，使用一個按壓開關，控制單向紅綠燈的模式，每按一下按壓開關，就轉爲另一模式。

※ 8. 打地鼠遊戲。請利用 8 個 LED 對應 8 個按壓開關，可以寫一個打地鼠遊戲，LED 可依亂數點亮，按鍵按這些亮的 LED 可得分，得分由電腦螢幕輸出。

※ 9. 同上題，得分用一個七段顯示器輸出。

※ 範例 5-4d

電子搶答器。輸入設備：雙方各有一個按鈕可搶答，先按的人，燈才會亮。主持人有一按鈕，可重置所有燈號，且亮一個燈號，表示可開始搶答。輸出設備：雙方各有一個 LED，用來表示取得答題權。

電路設計

電路設計如圖 5-19。

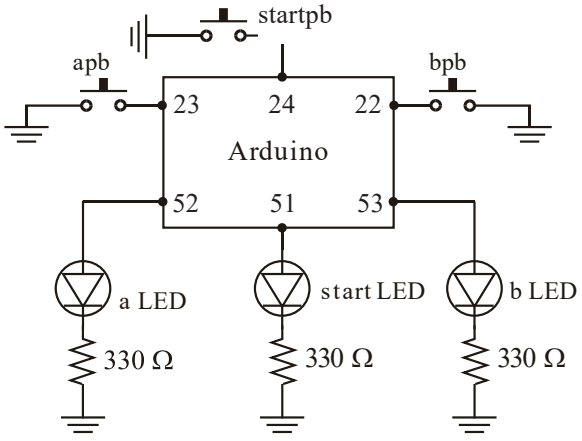


圖 5-19　電子搶答器電路圖

程式列印

```
1.  //23,24,22 接apb,startpb,bpb等按壓開關
2.  //52,51,53 接aLED,startLED,bLED
3.  const byte apb=23;//a搶答按鈕
4.  const byte aled=52;//a搶答燈號
```

```
5.  const byte bpb=22; //b搶答按鈕
6.  const byte bled=53; //b搶答燈號
7.  const byte startpb=24;//主持人
8.  const byte startled=51;//開始搶答燈號
9.  byte a;
10. bool ae=true;//a enable
11. byte b;
12. bool be=true;//b enable
13. void setup() {
14.   pinMode(apb,INPUT_PULLUP);
15.   pinMode(aled,OUTPUT);
16.
17.   pinMode(bpb,INPUT_PULLUP);
18.   pinMode(bled,OUTPUT);
19.
20.   pinMode(startpb,INPUT_PULLUP);//host
21.
22.   pinMode(startled,OUTPUT);
23.
24.   digitalWrite(aled,LOW);
25.   digitalWrite(bled,LOW);
26.   digitalWrite(startled,LOW);
27.   Serial.begin(9600);
28.   DDRB=B11111111;
29.   DDRC=0xFF;PORTC=0xFE;//將8*8點陣LED當作8個LED使用
30. }
31. void loop() {
32.     //進行歸位重置動作
33.     ae=true;
34.     digitalWrite(aled,LOW);
35.     be=true;
36.     digitalWrite(bled,LOW);
37.     digitalWrite(startled,LOW);
38.     //只要主持人未按鈕，就等待
39.     while(digitalRead(startpb)==HIGH){}
40.     if(digitalRead(startpb==LOW)){//單晶太快，通通要延遲
41.       while(digitalRead(startpb)==LOW){
42.         delay(10);
43.         }
44.     }
```

```
45.     digitalWrite(startled,HIGH);
46.     //可以搶答了
47.     while(digitalRead(startpb)==HIGH){
48.        a=digitalRead(apb);
49.        b=digitalRead(bpb);
50.        //Serial.print(a);
51.        //Serial.println(b);
52.        //當我可以按，我按了
53.        if ( ae && (a==LOW)) {
54.            digitalWrite(aled,HIGH);//我燈亮
55.            digitalWrite(startled,LOW);
56.            be=false;//設定對方無法按
57.        }
58.        //當我可以按，我按了
59.        if ( be && (b==LOW)) {
60.           digitalWrite(bled,HIGH);
61.           digitalWrite(startled,LOW);
62.           ae=false;//設定對方無法按
63.        }
64.     }
65.     //按了start
66.     Serial.println("The host press the start");
67.     Serial.println("進行歸位重置動作");
68. }
```

自我練習

1. 同範例 5-4d，如果是 3 或 4 個人同時搶答呢？請設計電路與程式。
2. 選秀表決器。有一次我在漢神巨蛋廣場看到某公司在選秀，每一個表演者只有至多 3 分鐘可以表演，有 3 個評審依據表演評分，每個評審認爲通過就按鈕且該評審的燈亮，2 個或超過 2 個評審同意，就表示通過，綠燈亮。所以若唱得很好，一下子就入選，唱的不好，時間到了，若沒有 2 個人給燈，那就亮紅燈，請設計此電路。（本題目在 5-3f 已經使用指撥開關完成，請重新使用按壓開關）

5-5 四位數七段顯示器

▶ 表 5-12　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	四位數共陰極七段顯示器	1	
3	限流電阻 220Ω ～ 470Ω	16	
4	指撥開關	8	
5	按壓開關	8	

共陰極四位數七段顯示器

前面我們已經介紹一位數七段顯示器，此七段顯示器共用了 9 隻腳，那四位數是不是拿四個七段來用呢，這樣一共要 36 隻腳，答案竟然是否定的，因爲單晶控制接腳很昂貴，不可能爲了一個四位數七段輸出就用掉 36 隻腳，而且就算 Arduino MEGA 有 36 隻腳可用，但這樣接線與程式反而不好寫。

四位數七段顯示器也和一位數七段相同，分爲共陽極與共陰極。因爲 Arduino 電流夠大，可直接使用高電位驅動 LED，所以本書使用共陰極。共陰極四位數七段顯示器內部發光二極體接腳如下圖，四個位數內部的 a,b,c,d,e,f,g,dp 都接在一起，再共用一個接點拉出，每個位數的 8 個 LED 共用陰極且分別輸出，千位數命名爲 com1、百位數命名爲 com2、十位數名爲 com3、個位數命名爲 com4，如圖 5-20：

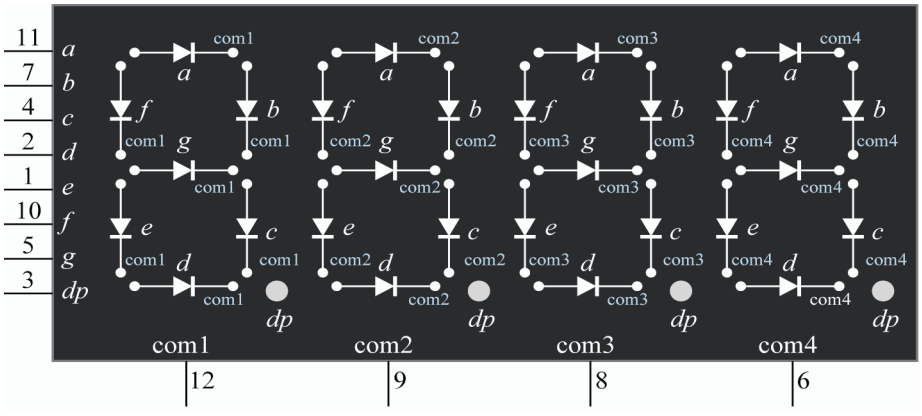


圖 5-20　四位數共陰七段顯示器內部結構圖

圖 5-21 是正面俯視（文字朝上，接腳朝下）的腳位圖，且不論是共陰極或共陽極的腳位圖都相同（本書以左下角 e 腳位編號爲 1，逆時針繼續編號）。圖 5-22 是本書實驗板的八位數七段顯示器，腳位 a , b ,c , d , e , f , g , dp ,1com1,1com2, 1com3,1com4,2com1,2com2,2com3,2com4 已經按照順序排列。

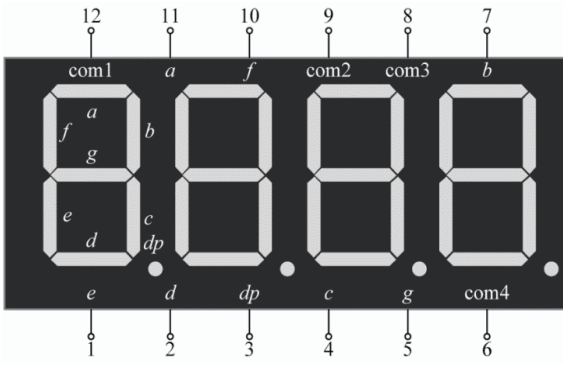


圖 5-21　四位數七段顯示器接腳圖

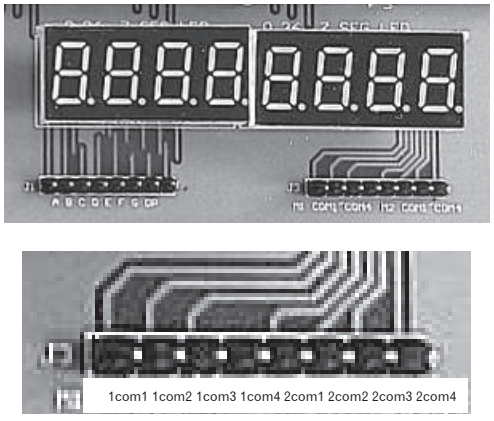


圖 5-22　四位數七段實體圖

以共陰極爲例，請拿出傳統指針型三用電表驗證（小型數位電表不行），請先撥到歐姆檔 *10，例如，正極（黑棒）接 a 點，負極（紅棒）接 com4，則個位數的 a 將亮起，又例如正極接 a，負極接 com3，則十位數的 a 將亮起，如表 5-13 所示（補充說明：三用電表電池正極是用黑棒拉出）：

▶ 表 5-13　四位數七段 LED 位置圖

正極	負極	功能
a	com4	個位數 a
a	com3	十位數 a
c	com2	百位數 c
d	com1	千位數 d

驅動電路

四位數七段顯示器的驅動電路如圖 5-23。

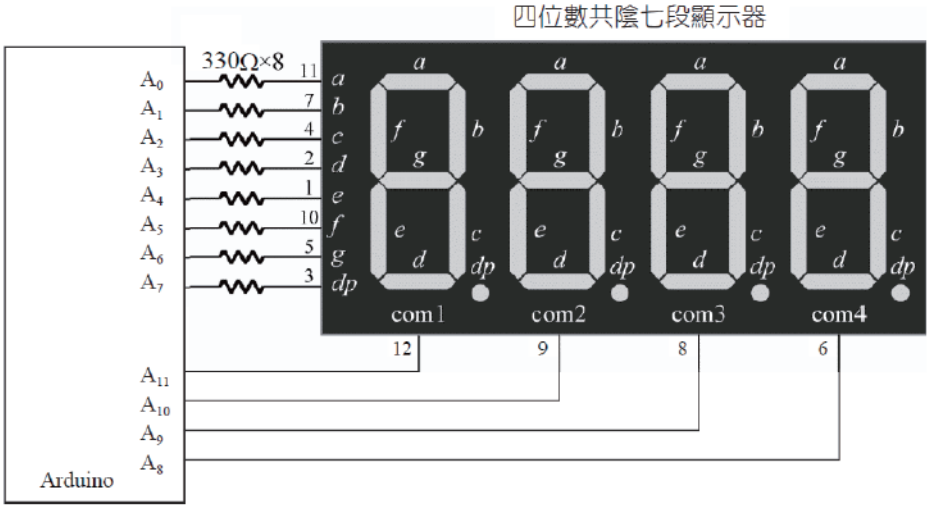


圖 5-23　四位數七段驅動電路圖

硬體測試

請鍵入以下程式，觀察所有 LED 是否全亮，可藉此檢查硬體接線是否正常。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
4.    DDRF=0xFF;//指派PORTF腳位功能是輸出
5.    DDRK=0xFF; //指派PORTK腳位功能是輸出
6.  }
7.
8.  void loop() {
9.    PORTF=0xFF;//指派腳位電壓爲高電位，電壓5V
10.   PORTK=0;// //指派腳位電壓爲低電位，電壓0V
11. }
```

四位數七段顯示器

請鍵入以下程式，觀察四個七段有沒有一個一個全亮，再將 delay(1000); 改爲 delay(1)，觀察有沒有全亮。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  void setup() {
5.    DDRF=0xFF;//指派PORTF爲輸出
6.    DDRK=0xFF; //指派PORTK爲輸出
7.  }
8.  void loop() {
9.    for (int i=0;i<=3;i++){
10.     PORTK=a[i];//位址
11.     PORTF=0xff;
12.     delay(1000);//分別亮
13.     //delay(1);//4個一起亮
14.   }
15. }
```

由以上程式可知，位址線 PORTK 輸出 0x7,0xb,0xd,0xe 時 com1,com2,com3,com4 的電位如表 5-14：千 (com1)、百 (com2)、十 (com3) 與個位數 (com3)，每次僅能 1 個位數爲低電位，資料將會輪流傳送到指定位置。只要速度夠快，因爲人類眼睛有視覺暫留現象，所以看起來就會一起亮。

▶ 表 5-14　四位數七段位址控制表

值	千 com1(bit3)	百 com2(bit2)	拾 com3(bit1)	個 com4(bit0)	顯示燈號
0x7	0	1	1	1	千位數
0xb	1	0	1	1	百位數
0xd	1	1	0	1	十位數
0xe	1	1	1	0	個位數

由上表可知，由於共用資料線，以下程式，可將資料送到千位數，將千位數 8 個 LED 全點亮，請藉此檢查硬體接線是否正確。

```
PORTK=0x7;//位址
PORTF=0xff;//資料
```

以下程式，可將資料送到百位數，將百位數全點亮，請藉此檢查硬體接線是否正確。

```
PORTK=0xb;//位址
PORTF=0xff;//資料
```

以下程式，可將資料送到十位數，將十位數全點亮，請藉此檢查硬體接線是否正確。

```
PORTK=0xd;//位址
PORTF=0xff;//資料;
```

以下程式，可將資料送到個位數，將個位數全點亮，請藉此檢查硬體接線是否正確。

```
PORTK=0xe;//位址
PORTF=0xff;//資料
```

爲了簡化程式的撰寫，位址也應該用 a 陣列儲存如下：

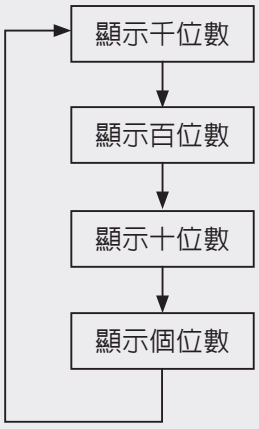
```
const byte a[]={0x7,0xb,0xd,0xe};//位址
```

若將 delay(1000); 改爲 delay(1)；因爲人類眼睛有視覺暫留現象，那就會同時亮。

四位數數字的顯示

前面已經介紹資料與位址的關係，那要如何分別顯示四個數字呢？根據以上特性，我們應該將所要對應顯示的數字，快速依序輸出就好，例如要同時顯示『0123』，則程式如下，也就是 0 送到千位數，1 送到百位數，2 送到十位數，3 送到個位數。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  void setup() {
4.    DDRF=0xFF;
5.    DDRK=0xFF;
6.  }
7.  void loop() {
8.    PORTK=0x7;//千位數
9.    PORTF=0x3F;//0
10.   delay(1);
11.   PORTK=0xb;//百位數
12.   PORTF=0x6;//1
13.   delay(1);
14.   PORTK=0xd;//十位數
15.   PORTF=0x5b;//2
16.   delay(1);
17.   PORTK=0xe;//個位數
18.   PORTF=0x4f;//3
19.   delay(1);
20. }
```



以上程式雖然是輪流給，但因人類眼睛有視覺暫留現象，所以看起來是一起亮。但是，以上程式有點冗長，若要簡化程式撰寫，就要善用迴圈與陣列，也就是將資料使用陣列先放好，就可使用迴圈輸出，以下程式的功能同上，也是顯示『0123』：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,
5.                    0x7f, 0x67,0x0};//資料
6.  void setup() {
7.    DDRF=0xFF;
8.    DDRK=0xFF;
9.  }
10. void loop() {
11.   int i;
12.   for (i=0;i<=3;i++){
13.     PORTK=a[i];//位址
14.     PORTF=b[i];//資料,b[0] is 0x3f,b[1] is 0x6,b[2] is 0x5b,
```

```
15.                         b[3] is 0x4F
16.     delay(1);
17.   }
18. }
```

顯示指定的數字

前面是顯示固定的四位數，那如何顯示任意一個四位數呢？例如，要顯示數字 1234，就將 1234 先分解爲 c[0]=1,c[1]=2,c[2]=3,c[3]=4，然後將此陣列再放到 b[] 陣列查表，如以下程式。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,
5.                    0x7f, 0x67,0x0};//資料
6.  byte c[4];
7.  int k=1234;//初值
8.  void setup() {
9.    DDRF=0xFF;
10.   DDRK=0xFF;
11. }
12. void loop() {
13.   //將k分解，放到c[0],c[1],c[2],c[3]
14.   c[0]=k/1000;//千位數
15.   c[1]=(k-c[0]*1000)/100;//百位數
16.   c[2]=(k-c[0]*1000-c[1]*100)/10;//十位數
17.   c[3]=k%10;//個位數
18.   //快速掃描
19.   for (int j=0;j<=3;j++){
20.       PORTK=a[j];
21.       PORTF=b[c[j]];
22.       delay(1);
23.   }
24. }
```

自我練習

1. 請寫一個程式，可以顯示 1 個浮點數。例如，如何顯示 30.2 呢。

5-6 計數器

▶ 表 5-15　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	四位數共陰極七段顯示器	1	
3	限流電阻 220Ω ～ 470Ω	16	
4	指撥開關	8	
5	按壓開關	8	

生活上很多多需要計數器，例如：景區入口、醫院門診叫號等，此即爲本節的重點，製作一個計數器。

範例 5-6a

請使用一個按壓開關，每當按一下，計數值加 1，使用四位數七段顯示器顯示其值。

操作步驟

1. 電路設計如圖 5-24。

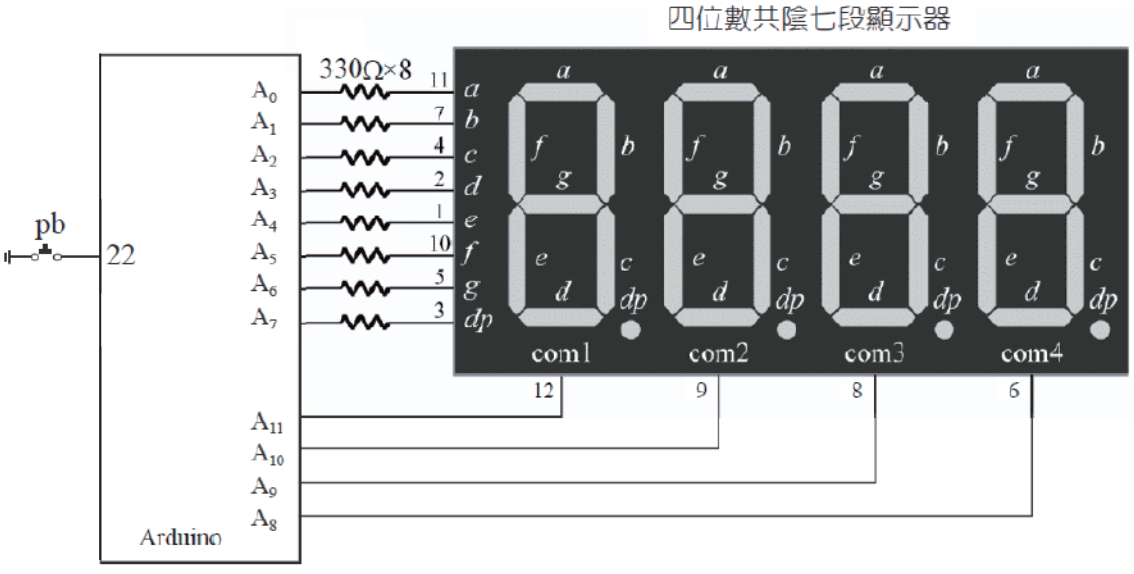


圖 5-24　計數器電路圖

2. 按壓開關、四位數七段前面都已經介紹，只要有按鍵，計數值累加 1，將計數值 k 分解爲 4 位數，輸出在四個七段顯示器，流程圖如圖 5-25。
3. 全部程式如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,
5.                    0x67,0x0};//資料
6.  byte c[4];
7.  int k=1234;//初值
8.  const byte pb=22;// 按鈕接腳
9.  void setup() {
10.   DDRF=0xFF;
11.   DDRK=0xFF;
12.   pinMode(pb,INPUT_PULLUP);
13.   Serial.begin(9600); //
14. }
15. void loop() {
16.   byte pbval=digitalRead(pb);
17.   if (pbval==LOW){
18.     while (digitalRead(pb)==LOW)
19.       delay(10);
20.     k=k+1; //計數值
21.   }
22.   Serial.println(k);//輸出到序列埠視窗
23.   //將k值分解爲c陣列
24.   c[0]=k/1000;//千位數
25.   c[1]=(k-c[0]*1000)/100;//百位數
26.   c[2]=(k-c[0]*1000-c[1]*100)/10;//十位數
27.   c[3]=k%10;//個位數
28.   //掃描輸出
29.   for (int j=0;j<=3;j++){
30.       PORTK=a[j];
31.       PORTF=b[c[j]];
32.       delay(1);
33.   }
34. }
```

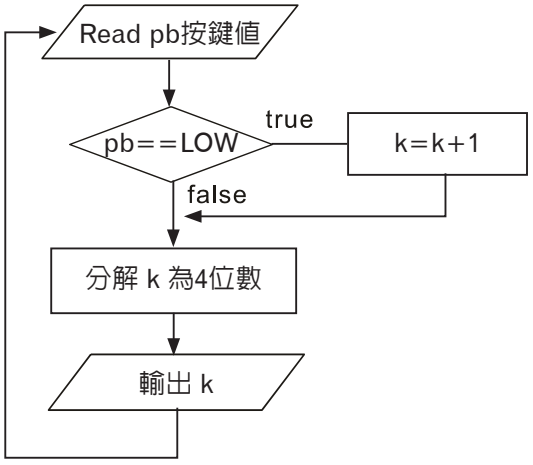


圖 5-25　計數器程式流程圖

自我練習

1. 同範例 5-6a，如何先閃爍新值兩次，再固定下來。
2. 同範例 5-6a，但增加 3 個按鈕，分別可遞減 1，遞增 10，遞減 10。
3. 同範例 5-6a，但設計屬於自己的球類比賽計分板，例如，羽球、桌球、籃球等計分板。

5-7 計時器

▶ 表 5-16　本節零件表

編號	零件名稱	數量	備註
1	LED	8	
2	四位數共陰極七段顯示器	1	
3	限流電阻 220Ω ～ 470Ω	16	
4	指撥開關	8	
5	按壓開關	8	

很多場合都需要計時器，例如：演講比賽、政見發表、研討會心得發表。本單元的重點即是製作此計時器。

範例 5-7a

計時器的製作。

操作步驟

1. 本範例電路圖同圖 5-23。
2. 前面的計數器是每按一個按鈕，計數值改變，計時器則是每 1 秒自動遞增 1，所以程式初步如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  char a[]={0x7,0xb,0xd,0xe};//位址
4.  char b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,0x67,
5.              0x0};//資料
6.  byte c[4];
7.  int k=1234;
8.  void setup() {
9.      DDRF=0xFF;
10.     DDRK=0xFF;
11. }
12. void loop() {
13.   c[0]=k/1000;//千位數
14.   c[1]=(k-c[0]*1000)/100;//百位數
```

```
15.   c[2]=(k-c[0]*1000-c[1]*100)/10;//十位數
16.   c[3]=k%10;//個位數
17.   //掃描輸出
18.   for (int j=0;j<=3;j++){
19.       PORTK=a[j];
20.       PORTF=b[c[j]];
21.       delay(1);
22.   }
23.     k=k+1;
24.     delay(1000);
25. }
```

3. 但是以上程式，七段顯示器幾乎沒有看到數字，那是因爲以上程式，微處理機的工作都停留在 delay(1000)，四個七段顯示器只要無法在 1/20 秒內重複點亮，當然看起來就熄滅了。所以，掃描輸出才是主角，唯有一面掃描輸出一面計時，四位數七段顯示器才能一直亮著，所以程式修改如下：

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,
5.                    0x67,0x0};//資料
6.  byte c[4];
7.  int k=1234;
8.  unsigned long t1,t2;
9.  void setup() {
10.   DDRF=0xFF;
11.   DDRK=0xFF;
12. }
13. void loop() {
14.   t1=millis();  //取到系統時間，單位是ms
15.   c[0]=k/1000;//千
16.   c[1]=(k-c[0]*1000)/100;//百
17.   c[2]=(k-c[0]*1000-c[1]*100)/10;//十
18.   c[3]=k%10;  //個位數
19.   do{
20.     for (int j=0;j<=3;j++){
21.       PORTK=a[j];
22.       PORTF=b[c[j]];
```

```
23.       delay(1);
24.       t2= millis();
25.     }
26.   }while ((t2-t1)<1000);//一面計時，一面掃瞄輸出
27.   k=k+1;
28. }
```

4. 以下程式，可將秒數分解爲分與秒。例如，k=66，初值顯示 0106，1 分 06 秒。

```
1.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
2.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
3.  const byte a[]={0x7,0xb,0xd,0xe};//位址
4.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,
5.                    0x67,0x0};//資料
6.  byte c[4];
7.  unsigned long k=66;//初值顯示0106，1分06秒
8.  unsigned long t1,t2;
9.  void setup() {
10.   DDRF=0xFF;
11.   DDRK=0xFF;
12. }
13. void loop() {
14.   int m,s;
15.   t1=millis();
16.   m=k/60;
17.   s=k %60;
18.   c[0]=m/10;//千,分鐘的十位數
19.   c[1]=m%10;//百,分鐘的個位數
20.   c[2]=s/10;//十,秒的十位數
21.   c[3]=s%10;  //個位數，秒的十位數
22.   do{
23.     for (int j=0;j<=3;j++){
24.       PORTK=a[j];
25.       PORTF=b[c[j]];
26.       delay(1);
27.       t2= millis();
28.     }
29.   }while ((t2-t1)<1000);
30.   k=k+1;
31. }
```

自我練習

1. 同範例 5-7a，但數字先滅掉，再顯示下一個數字。
2. 同範例 5-7a，但微處理機每秒產生 1 個 0 到 9999 的亂數，並於四位數七段顯示。
3. 同範例 5-7a，但可用變數設定一個指定的時間，例如 ,1 分鐘 ,2 分鐘，按鍵後開始倒數計時，且顯示剩餘時間，時間到時一直閃爍 LED。
4. 假設有 6 組數字，分別是 1234, 5566, 3388, 1688, 2688, 3099，請問如何每秒顯示一組數字。提示：請用一個陣列事先儲存 6 組四位數，例如，d[]={1234, 5566, 3388, 1688, 2688, 3099}。

範例 5-7b

倒數計時器。很多場合都需要一個能顯示倒數秒數的計時器。例如，演講比賽、政見發表會等都要一個可以指派時間的倒數計時器。

操作步驟

1. 本例我使用 3 個指撥開關，第 1 個指撥開關代表 30 秒，第 2 個指撥開關代表 1 分鐘，第 3 個指撥開關也代表 1 分鐘，累加這些時間當作起始時間（只要有動指撥開關，就有顯示計時的時間）。然後提供一個『start』按鈕，此按鈕可開始倒數計時，且一直顯示所剩時間。時間到時，一直閃爍顯示 0，直到使用者按一下『reset』按鈕，表示計時結束，才繼續顯示計時初值。以上功能的電路如圖 5-26：

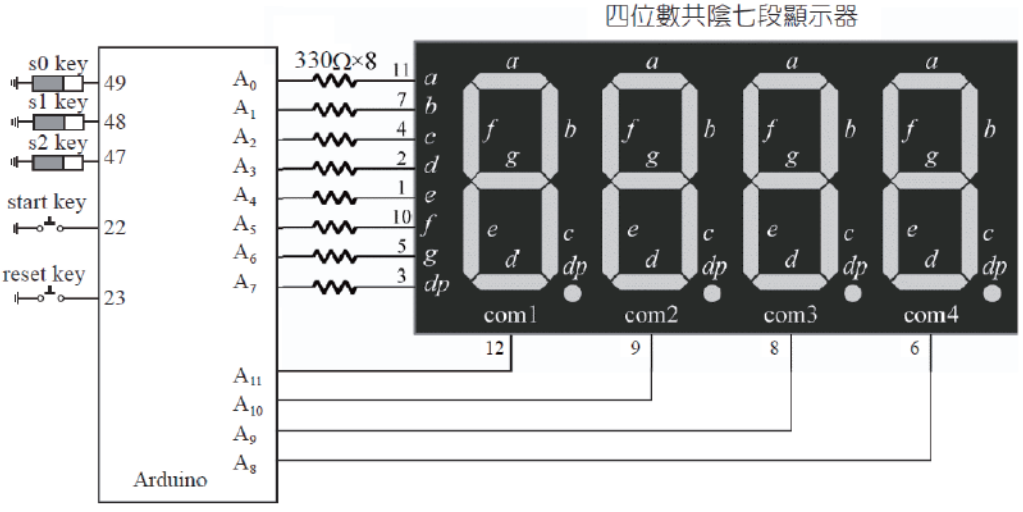


圖 5-26　倒數計時器電路圖

2. 程式如下：

```
1.  //49接s0key，48接s1key，47接s2key
2.  //22接 startkey，23接 resetkey
3.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
4.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
5.  const byte a[]={0x7,0xb,0xd,0xe};//位址
6.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,
                      0x67,0x0};//資料
7.  byte c[4];
8.  int k=0;//初值顯示0
9.  const byte s0key=49;//30秒
10. const byte s1key=48;//1分
11. const byte s2key=47;//1分
12. const byte startkey=22;//開始鍵
13. const byte resetkey=23;//reset
14. //boolean work=false;
15. void setup() {
16.   DDRF=0xFF;
17.   DDRK=0xFF;
18.   pinMode(startkey,INPUT_PULLUP);//具有上拉電阻的開關輸入
19.   pinMode(resetkey,INPUT_PULLUP);
20.   pinMode(s0key,INPUT_PULLUP);
21.   pinMode(s1key,INPUT_PULLUP);
22.   pinMode(s2key,INPUT_PULLUP);
23.   Serial.begin(9600);
24. }
25. void loop() {
26.   while  (digitalRead(startkey)==HIGH ) {
27.               //還沒開始計時，一直讀取指撥開關，顯示指撥開關所設定的初值
28.     k=0;
29.     byte s0val=digitalRead(s0key);
30.     if (s0val==LOW)
31.       k=k+30;      //本指撥代表30秒
32.     byte s1val=digitalRead(s1key);
33.     if (s1val==LOW)
34.       k=k+60;      //本指撥代表60秒
35.     byte s2val=digitalRead(s2key);
36.     if (s2val==LOW)
37.       k=k+60;      //本指撥代表60秒
```

```
38.     Serial.print(k);
39.     scan();
40.   }
41.   //開始計時
42.   while(k>=0){
43.     k=k-1;
44.     scan();
45.   }
46.   //時間到
47.   while  (digitalRead(resetkey)==HIGH ) {
48.     k=0;
49.     scan();
50.   }
51. }
52. void scan(){ //此段程式是掃瞄輸出
53.   long t1,t2;
54.   int j;
55.   int m,s;
56.   m=k/60;         //將k分解爲m分s秒
57.   s=k %60;
58.   c[0]=m/10;//千,分鐘的十位數
59.   c[1]=m%10;//百,分鐘的個位數
60.   c[2]=s/10;//十,秒的十位數
61.   c[3]=s%10;  //個位數，秒的十位數
62.   t1=millis();
63.   do{
64.     for (j=0;j<=3;j++){ //將四個位數快速輪流輸出
65.       PORTK=a[j];
66.       PORTF=b[c[j]];
67.       delay(1);
68.       t2= millis();
69.     }
70.   }while ((t2-t1)<100);//測試完，再改回1000
71. }
```

程式說明

以上在調整時間、計時，與結束顯示 0 等 3 個階段，八個七段顯示器都要不斷重複輸出資料（以上重複輸出資料，我們稱爲掃描），所以將掃描寫成 scan() 函式，如以上程式 52 ~71 行，然後於調整時間、計時，與結束顯示 0 等 3 個階段，都要執行 scan() 函式，這樣才符合結構化程式設計。

自我練習

1. 同範例 5-7b，但計時結束，請改爲持續閃爍 0，且蜂鳴器持續發出聲音。

※8位數七段顯示器

前面已經介紹一位數與四位數七段顯示器，但一個電子時鐘至少要八位數才能顯示，那八位數如何完成呢？答案是，可以將多個四位數七段一直串接，圖 5-27 就是串接兩個七段，構成一個八位數七段顯示器。

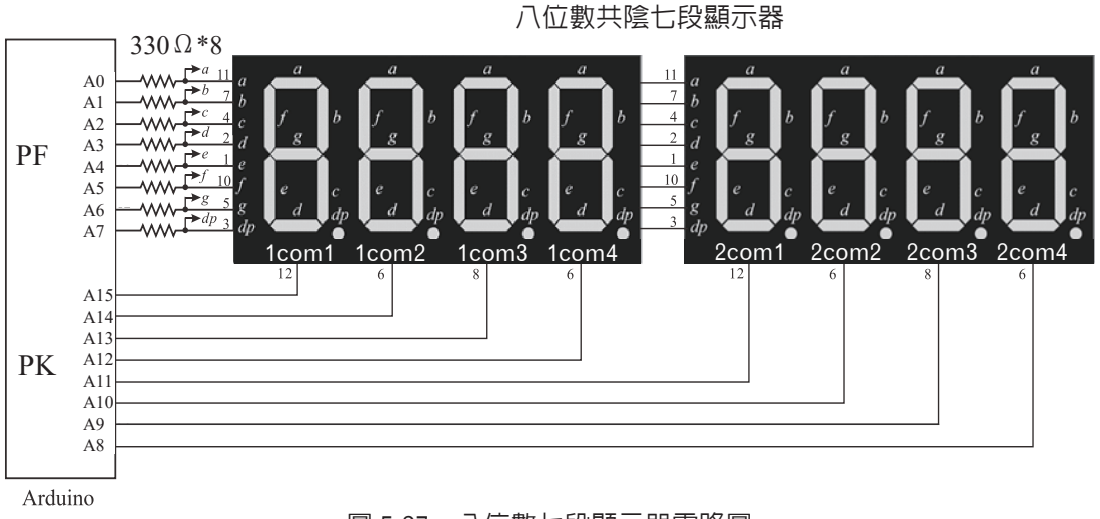


圖 5-27　八位數七段顯示器電路圖

1. 以上電路，要讓 8 個七段顯示器輪流亮的位址控制如表 5-17：

▶ 表 5-17　八位數七段顯示器位址控制表

顯示燈號	1com1	1com2	1com3	1com4	2com1	2com2	2com3	2com4	值
左 1	0	1	1	1	1	1	1	1	0x7f
左 2	1	0	1	1	1	1	1	1	0xbf
左 3	1	1	0	1	1	1	1	1	0xdf
左 4	1	1	1	0	1	1	1	1	0xef
左 5	1	1	1	1	0	1	1	1	0xf7
左 6	1	1	1	1	1	0	1	1	0xfb
左 7	1	1	1	1	1	1	0	1	0xfd
左 8	1	1	1	1	1	1	1	0	0xfe

2. 上圖的 PORTK 是位址線，也就是讓 PORTK 輪流依照以下 e[] 陣列輸出，就可讓資料輪流送到每一個對應的七段顯示器，只要速度夠快，因爲人類視覺有暫留現象，看起來是同時顯示。

```
char a[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
```

3. 程式設計如下，這樣就是電子時鐘了。

```
1.  //電子時鐘
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A15~A8 接七段的1com1,1com2,1com3,1com4,2com1,2com2,2com3,2com4
4.  //位址
5.  char a[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
6.  //資料,0x40 is -
7.  char b[] = {0x3f, 0x6, 0x5b, 0x4f,0x66,0x6d,0x7c,0x7,0x7f,0x67,0x0,0x40};
8.  char c[8];//輸出掃描用
9.  long k=22*3600L+20*60+30;//初值顯示22-20-30
10. unsigned long t1,t2;
11. byte h,m,s;
12. void setup() {
13.   DDRF=0xFF;//資料
14.   DDRK=0xFF; //位址
15. }
16. void loop() {
17.   t1=millis();
18.   h=k/3600L;
19.   m=(k-h*3600L)/60;
20.   s=k %60;
21.   c[0]=h/10;//時針的十位數
22.   c[1]=h%10;//時針的個位數
23.   c[2]=11;//-
24.   c[3]=m/10;  //分的十位數
25.   c[4]=m%10;//分的十位數
26.   c[5]=11;//-
27.   c[6]=s/10;//秒的十位數
28.   c[7]=s%10;  //秒的個位數
29.   do{
30.     for (byte j=0;j<=7;j++){
31.       PORTK=a[j];//位址
```

```
32.       PORTF=b[c[j]]; //資料
33.       delay(1);
34.       t2= millis();
35.     }
36.   }while ((t2-t1)<1000);
37.   k=(k+1)%86400L;
38. }
```

※可調時間的電子時鐘

前面已經完成電子時鐘，現在我們再用 2 個按鈕，名稱分別是 sm,sh，分別接到 Arduino 的腳位 22 與 23，程式如下，這樣就可以調整時間。

```
1.  //22,23接sm,sh按壓開關，分別調整分與時
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A15~A8 接七段的1com1,1com2,1com3,1com4,2com1,2com2,2com3,2com4
4.  //位址
5.  char a[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
6.  //資料,0x40 is -
7.  char b[] = {0x3f, 0x6, 0x5b, 0x4f,0x66,0x6d,0x7c,0x7,0x7f,0x67,0x0,0x40};
8.  char c[8];//輸出掃描用
9.  long k=22*3600L+20*60+30;//初值顯示22-20-30
10. unsigned long t1,t2;
11. const byte sm=22;//調整分針
12. const byte sh=23;//調整時針
13. byte h,m,s;
14. void setup() {
15.   DDRF=0xFF;//資料
16.   DDRK=0xFF; //位址
17.   pinMode(sm,INPUT_PULLUP);
18.   pinMode(sh,INPUT_PULLUP);
19.   //pinMode(sd,INPUT_PULLUP);
20. }
21. void loop() {
22.   t1=millis();
23.   byte smval=digitalRead(sm);
24.   if (smval==LOW){        //調整分
25.     while (digitalRead(sm)==LOW)
26.       delay(1);
27.     k=k+60;//每按一下，增加60秒，等於調整1分鐘
```

```
28.   }
29.   byte shval=digitalRead(sh); //調整時
30.   if (shval==LOW){
31.     while (digitalRead(shval)==LOW)
32.       delay(1);
33.     k=k+3600;//每按一下，增加3600秒，等於調整1小時
34.   }
35.   h=k/3600L;
36.   m=(k-h*3600L)/60;
37.   s=k %60;
38.   c[0]=h/10;//時針的十位數
39.   c[1]=h%10;//時針的個位數
40.   c[2]=11;//-
41.   c[3]=m/10;  //分的十位數
42.   c[4]=m%10;//分的十位數
43.   c[5]=11;//-
44.   c[6]=s/10;//秒的十位數
45.   c[7]=s%10;  //秒的個位數
46.   do{
47.     for (byte j=0;j<=7;j++){
48.       PORTK=a[j];
49.       PORTF=b[c[j]];
50.       delay(1);
51.       t2= millis();
52.     }
53.   }while ((t2-t1)<1000);
54.   k=(k+1)%86400L;
55. }
```

自我練習

※ 1. 萬年曆與時鐘。請再增加顯示日期，且新增 3 個按鈕，sweek、sday、smon，腳位分別是 24,25,26，這樣就可以調整星期、日與月。其次，讓時間與日期各顯示 30 秒鐘，也就是前面 30 秒顯示時間、後面 30 秒顯示日期。

5-8 外部中斷控制

前面使用 digitalRead(skey) 詢問按鍵是否被按，稱爲輪循法輸入，程式有執行到資料輸入點，輸入才有效。但眞實的生活中，就不能這樣了，因爲眞實的生活裡，有些情況很緊急，例如，電話來了、警鈴響了、東西突然撞過來、溫度突然上升等，這些都不能等到程式偵測到此點時，才讀取這些狀態，所以就有所謂外部中斷法控制輸入。微處理機的外部中斷控制就很多，例如、按鍵的中斷、時間的中斷等，這樣也可以簡化程式，且隨時都可以偵測到這些按鍵的變化。本書以下介紹使用按鍵的外部中斷，重做以上計數器。使用外部中斷法的步驟如下：

1. 指派按鍵腳位。MAGA 2560 僅能使用 2,3,18,19,20,21 等腳位。
2. 指派腳位功能。本例指派按鍵腳位爲具有上拉電阻的輸入。
3. 使用 attachInterrupt(digitalPinToInterrupt(pb),aa,FALLING); 指派使用中斷方式，aa() 是按鍵被按所要執行的函式，FALLING 就是當按鍵被按，電位下降時，就觸發中斷，就執行 aa() 函式。（函式名稱 aa 可自己任意變化）
4. 撰寫中斷函式內容。就是按鍵被按，所要執行的函式內容，本例配合以上第 3 點的宣告，也就是撰寫 aa() 函式。
5. 使用 volatile 宣告中斷副程式傳遞的變數，本例是變數 k，此變數僅能 byte 型態。（此規定請線上查詢 volatile）

以上步驟的流程圖如圖 5-28。

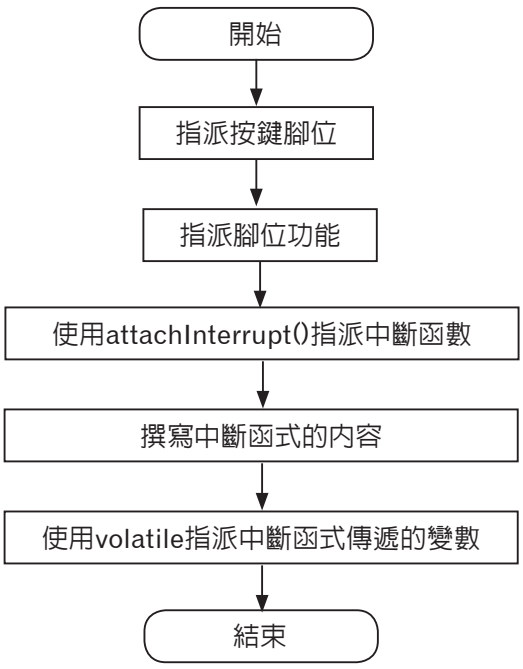


圖 5-28　外部中斷法流程圖

範例 5-8a

示範以上外部中斷控制。

操作步驟

1. 根據以上規定，本例連接一個按壓開關至 Arduino 腳位 2。
2. 程式撰寫如下：

```
1.  //2 接pb 按壓開關，本例每按一次，計數值加1
2.  //PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
3.  //PORTK A11~A8 接七段的1com1,1com2,1com3,1com4
4.  const byte a[]={0x7,0xb,0xd,0xe};//位址
5.  const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f,
6.                    0x67,0x0};//資料
7.  byte c[4];
8.  const byte pb=2;
9.          //1 指派按鍵腳位，MAGA 2560 僅能使用2,3,18,19,20,21等腳位
10. volatile byte k=0;
11.         //5.使用volatile宣告中斷副程式傳遞的變數，此變數僅能byte型態
12. void setup() {
13.   DDRF=0xFF;
14.   DDRK=0xFF;
15.   pinMode(pb,INPUT_PULLUP);
16.                  //2.指派腳位功能，本例按鍵腳位爲具有上拉電阻的輸入
17.   attachInterrupt(digitalPinToInterrupt(pb),aa,FALLING);
18.                                                  //3指派使用中斷方式
19. //aa是按鍵被按所要執行的函式
20. //FALLING就是當按鍵被按，電位下降時，就觸發中斷，就執行aa()函式
21. }
22. void loop() {
23.   //將k值分解爲c陣列
24.   c[0]=k/1000;//千位數
25.   c[1]=(k-c[0]*1000)/100;//百位數
26.   c[2]=(k-c[0]*1000-c[1]*100)/10;//十位數
27.   c[3]=k%10;//個位數
28.   for (int j=0;j<=3;j++){
29.       PORTK=a[j];
30.       PORTF=b[c[j]];
31.       delay(1);
```

```
32.   }
33. }
34. //4.指派中斷副程式，也就是按鍵被按，所要執行的函式內容
35. void aa(){//強制計數值加1
36.     k++;
37. }
```

補充說明

1. 請特別留意，指派中斷副程式傳遞的變數，由線上使用手冊可知（查詢 volatile），此變數僅能 byte 型態，也就是 k 值僅能 0 到 255。請您改爲 integer，並觀察執行結果，眞的沒有錯誤訊息，但計數值就是亂跳。寄望以後 Arduino 改版時，編譯器會主動偵錯，告知此 volatile 的型態僅能 byte，那就可以減少挫敗感。
2. 同範例 5-8a，但增加 3 個按鈕，分別可遞減 1，遞增 10，遞減 10。
3. 同範例 5-8a，但設計屬於自己的球類比賽計分板，例如，羽球、桌球、籃球等計分板。

學後測驗

題號	題目	輸出結果

1	void setup() {
	  DDRB=B11111111;
	}
	char i=-5,a='a'
	void loop() {
	  PORTB=i; //寫出8個LED的亮滅
	  PORTB=a; //寫出8個LED的亮滅
	  Serial.println((int)i);//轉爲整數
	}	

2	//PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
	byte a[]={0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7d,0x7,0x7f,0x6f};
	byte b[]={8,2,5,7,1,0};
	void setup() {
	  DDRF=B11111111;
	  PORTF=0x6; //寫出顯示數字
	  PORTF=a[3]; //寫出顯示數字
	  PORTF=a[b[3]]; //寫出顯示數字
	}void loop() { }	

3	//PORTF A0~A7接七段的a,b,c,d,e,f,g,dp
	//PORTK A11~A8 接七段的
	1com1,1com2,1com3,1com4
	const byte a[]={0x7,0xb,0xd,0xe};//位址
	const byte b[] = {0x3f,0x6,0x5b,0x4f,0x66,0x6d,0x7c,0x7,0x7f, 0x67,0x0};//資料
	byte c[4];
	int k=8234;//初值
	void setup() {
	  DDRF=0xFF;
	  DDRK=0xFF;
	}
	void loop() {
	  c[0]=k/1000;//千位數
	  c[1]=(k-c[0]*1000)/100;//百位數
	  c[2]=(k-c[0]*1000-c[1]*100)/10;//十位數	

	c[3]=k%10;//個位數
	//快速掃描
	for (int j=0;j<=3;j++){
	PORTK=a[j];
	PORTF=b[c[j]];
	delay(1);
	}
	}	

4	int b=5; Serial.println(bitRead(b,2))	
5	假設 8 個指撥開關連接在 PORTL，請寫出如何讀取 8 個指撥狀態？	
6	假設 1 個按壓開關連接在 22，請寫出如何讀取腳位 22 的電壓值？	
7	假設 1 個按壓開關連接在 22，請寫出讓使用者按一下，計數值增加 1，且計數值顯示在序列埠。	
8	請寫出一個程式，可以使用中斷方式，每按一下按鈕，計數值加 1，且計數值顯示在序列埠。（請也說明按壓開關使用那一個腳位）	

Chapter
06

進階應用控制

學習大綱

- 6-1　點矩陣發光二極體控制
- 6-2　液晶顯示器控制
- 6-3　聲音控制
- 6-4　鍵盤控制
- 6-5　密碼鎖
- 6-6　步進馬達

學習目標

- 1. 能使用 8*8 點陣 LED 顯示簡單中文、英文字母與數字。
- 2. 能使用液晶顯示器輸出文數字。
- 3. 能使用蜂鳴器製作電子琴、播出音樂。
- 4. 能使用 4*4 按鍵開關數入數字。
- 5. 能製作密碼鎖。
- 6. 能使用步進馬達製作 x-y 軸平台。

6-1 點矩陣發光二極體控制

▶ 表 6-1　本節零件表

編號	零件名稱	數量	備註
1	8*8 共陰點矩陣 LED	1	
2	電阻 330Ω	8	
3	按壓開關	8	

8*8點矩陣LED

8*8 點矩陣 LED 分爲共陰極與共陽極，因爲 Arduino 高電位輸出電流有 20mA，已經可以直接驅動 LED，所以本書採用共陰極，電路才會簡單。共陰極並不是所有 LED 的陰極都共用，而是每一行 (C_1,C_2..C_8) 的陰極共用輸出腳位，如圖 6-1 的 C_1,C_2..C_8，這樣等一會資料才能同時送到每一行。(本書使用行優先，因爲有些國家以列優先，所以到材料行買零件，請由圖 6-1 確認共陰或共陽)

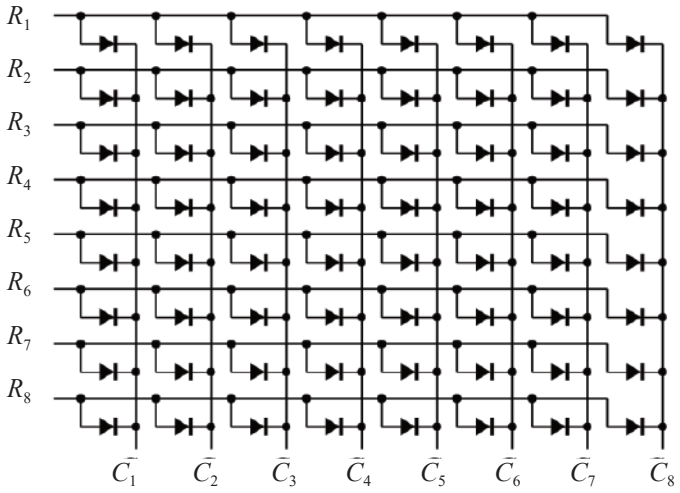


圖 6-1　共陰極點矩陣 LED 電路圖

不論是共陽極或共陰極其腳位排列都相同，如圖 6-2（LED 朝上），廠商通常將型號印在下面，所以請將有型號的那一面朝下，本書以 R_5 爲腳位 1，逆時針繼續編號 (此與 IC 腳位編號相同)，例如腳位 9 是 R_1。因爲點矩陣 LED 腳位很亂，初學者很容易接錯，所以本書已經重新排列，將 R_1~R_8,C_1~C_8 按照順序排列，如圖 6-3。

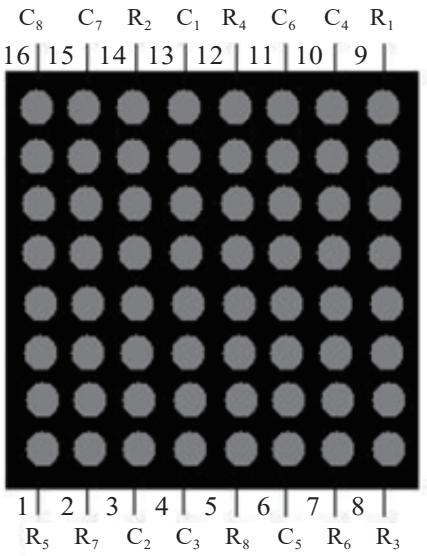


圖 6-2　點矩陣 LED 腳位圖

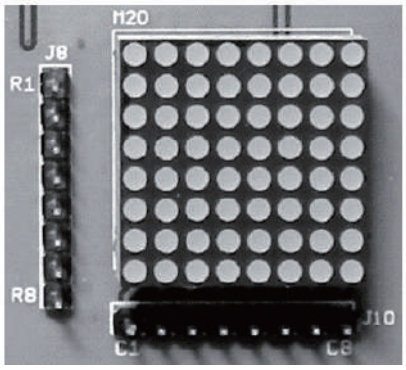


圖 6-3　點矩陣 LED 實體圖

請自行拿傳統型指針式三用電表驗證（請撥 R*10，小型數位電錶不行）。例如，共陰極的測量如下：正極（黑棒）接 R_1，負極（紅棒）接 C_1，則第 1 列第 1 行將亮；正極接 R_1，負極接 C_2，則第 1 列第 2 行將亮（補充說明：三用電表電池正極是用黑棒拉出。）。

點矩陣LED驅動電路

現在請將點矩陣 LED 接線如圖 6-4：

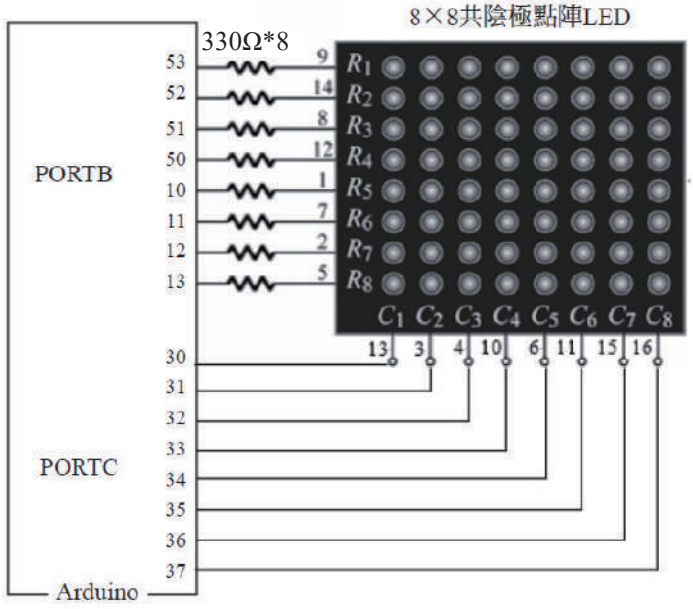


圖 6-4　點矩陣 LED 驅動電路圖

硬體測試

請鍵入以下程式，觀察 LED 有沒有一排一排輪流全亮。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
4.  void setup() {
5.       DDRB=0xFF;DDRC=0xFF;
6.  }
7.  void loop() {
8.       for (int i=0;i<=7;i++){
9.       PORTC=c[i];//位址
10.      PORTB=0xff;//資料
11.      delay(500); //請修改爲1，並比較其效果
12.      }
13. }
```

請將 delay(500); ，修改爲 delay(1);，並觀察所有 LED 有沒有全亮。以上程式的 c[] 陣列如下：

```
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
```

以上數值 0x7f,0xbf,0xdf…可以讓 C_1,C_2,C_3…每次一個得到低電位，所以，C_1..C_8 的電壓時序圖如表 6-2：

▶ 表 6-2　點陣 LED 位址控制時序圖

值	bit7 C_1	bit6 C_2	bit5 C_3	bit4 C_4	bit3 C_5	bit2 C_6	bit1 C_7	bit0 C_8	顯示位址
0x7f	0	1	1	1	1	1	1	1	C1
0xbf	1	0	1	1	1	1	1	1	C2
0xdf	1	1	0	1	1	1	1	1	C3
0xef	1	1	1	0	1	1	1	1	C4
0xf7	1	1	1	1	0	1	1	1	C5
0xfb	1	1	1	1	1	0	1	1	C6
0xfd	1	1	1	1	1	1	0	1	C7
0xfe	1	1	1	1	1	1	1	0	C8

以上 c[] 陣列稱爲位址，主要是讓 $C_1..C_8$ 輪流接地，由於共用資料線，資料就可以輪流傳送到 $C_1..C_8$。例如，以下程式，可讓左邊 C_1 第一行 8 個燈全亮。

```
PORTC=0x7f; //位址
PORTB=0xff; //資料
```

以下程式，將會使點陣 LED 左邊 C_1 最上面 1 個 LED 亮。

```
PORTC=0x7f; //位址
PORTB=0x01; //資料
```

以下程式，將會使點陣 LED 左邊 C_2 的上面兩個 LED 全亮。

```
PORTC =0xbf;
PORTB =0x3;
```

範例 6-1a

文字數位化

運算思維

若要將文數字顯示在此 8*8 的點陣 LED，也是要將此文字數位化，將此文數字資料數位化的步驟如下：將此文數字寫在以下方格紙，並計算每行 (Column) 的值。表 6-3 是將『洪』寫在方格紙上，則 C_1 值二進位是 B10010001（高位元在下面），以 16 進位表示是 0x91、C_2 值是 0x4a，依此類推。

▶ 表 6-3　點陣 LED 文字數位化

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1	1	0	0	0	1	0	1	0
R_2	0	1	0	1	1	1	1	1
R_3	0	0	0	0	1	0	1	0
R_4	0	1	0	0	1	0	1	0
R_5	1	0	0	1	1	1	1	1
R_6	0	0	0	0	1	1	0	0
R_7	0	1	0	1	0	0	1	0
R_8	1	0	1	0	0	0	0	1
值	0x91	0x4a	0x80	0x52	0x3f	0x32	0x5f	0x92

將以上每一行的值以陣列儲存如下：（影像變識要將文字數位化，也是此相同的原理）

```
byte d[]={0x91,0x4a,0x80,0x52,0x3f,0x32,0x5f,0x92};
```

然後快速依序傳送到對應的行，例如，0x91 送到 C_1、0x4a 送到 C_2…我們稱此爲掃描輸出，程式如下：

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
4.  byte d[]={0x91,0x4a,0x80,0x52,0x3f,0x32,0x5f,0x92};
5.  void setup() {
6.      DDRB=0xFF;DDRC=0xFF;
7.  }
8.  void loop() {
9.      for (int i=0 ;i<=7;i++){
10.     PORTC=c[i];//位址
11.     PORTB=d[i];//資料
12.     delay(500);//delay(1)
13.     }
14. }
```

請觀察以上執行結果，點矩陣 LED 有沒有一行行輪流亮。只要掃描速度夠快，因爲人類眼睛有視覺暫留現象，看起來就會一起亮，此一快速掃描輸出原理就可以顯示任何文字與影像，請將 delay(500) 改爲 delay(1)，觀察有沒有顯示「洪」。

自我練習

1. 請自行找一個筆畫較少的中文字，以能填入表 6-4 8*8 方格爲原則，計算其陣列值，並顯示在此 8*8 LED 上。（填文字時，請留意線條之間要留空白，才表示可辨識）

▶ 表 6-4　點陣 LED 文字數位化方格紙

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

範例 6-1b

跑馬燈

運算思維

範例 6-1a，我們先顯示一個字，若要顯示的文字長度超過字幕機的寬度，可以使用跑馬燈的原理，請鍵入以下程式，並觀察執行結果。

```
1.  //PORTB 13,12,11,10,50,51,52,53接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
4.  byte d[]={0,0,0,0,0,0,0,0,0,0x91,0x4a,0x80,0x52,0x3f,0x32,0x5f,
              0x92,0,0,0x47,0x45,0x45,0x39,0,0,0,0x7c,0x12,0x11,0x12,
              0x7c,0,0,0,0,0,0,0,0,0,0};
5.  int slen= 41;//此數字爲以上d[]的長度
6.  byte a[8];
7.  void setup() {
8.    DDRB=0xFF;
9.    DDRC=0xFF;
10. }
11. void loop() {
12.   //以跑馬燈左旋顯示文字
13.     for (int k=0 ;k<=slen-8;k++){
14.         for (int i=0;i<=7;i++){//依序每次抓8個
15.           a[i]=d[i+k] ;
16.         }
```

```
17.          for (int i=0;i<=50;i++){//停留時間
18.            for (int j=0 ;j<=7;j++){
19.                    PORTC=c[j];//位址
20.                    PORTB=a[j];//資料
21.                    delay(1);
22.                }
23.          }
24.    }
25. }
```

本例以跑馬燈顯示『洪 5A』，先將『洪 5A』數位化，計算字型如表 6-5：

▶ 表 6-5　跑馬燈文字數位化方格紙

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
R1	1	0	0	0	1	0	1	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0
R2	0	1	0	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0
R3	0	0	0	0	1	0	1	0	0	0	1	1	1	0	0	0	0	1	0	0	0	1	0	0
R4	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
R5	1	0	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	0
R6	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
R7	0	1	0	1	0	0	1	0	0	0	1	1	1	0	0	0	0	1	0	0	0	1	0	0
R8	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	91	4a	80	52	3f	32	5f	92	0	0	47	45	45	39	0	0	0	7c	12	11	12	7c	0	0

將以上資料以陣列儲存如下，文字前後都要加上空白 0，使用者才來得及看，slen 是輸出文字的行數，也是 d 陣列的長度。

```
byte d[]={0,0,0,0,0,0,0,0,0x91,0x4a,0x80,0x52,0x3f,0x32,
          0x5f,0x92,0,0,0x47,0x45,0x45,0x39,0,0,0,0x7c,0x12,
          0x11,0x12,0x7c,0,0,0,0,0,0,0,0,0,0};
int slen=40;
```

現在逐一將 d[] 陣列，每次複製 8 筆資料到 a[] 陣列，

```
for (int i=0;i<=7;i++){//依序每次抓8個
          a[i]=d[i+k] ;
        }
```

第 1 次抓到 0,0,0,0,0,0,0,0; 第 2 次抓到 0,0,0,0,0,0,0,0x91; 第 3 次抓到 0,0,0,0,0,0x91,0x4a…依此類推。再將 a[] 陣列，同範例 6-1a 掃瞄輸出。但微處理機眞的太快了，每一次的掃描還要重複一點時間，如以上程式的 i 迴圈。請自行調整 i 迴圈的值，即可調整跑馬燈輪轉速度。請留意每次抓 8 個，所以最後 1 次是抓全部長度減 8（如以上程式的 k 迴圈的 slen-8），不然會造成陣列溢位。

自我練習

1. 請於以下填入適當的文字，並依序每次 8 筆資料輸出文字。

	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0
1																															
2																															
3																															
4																															
5																															
6																															
7																															
8																															

範例 6-1c

告白板

運算思維

範例 6-1b 是跑馬燈的輸出，輸出效果比較平淡，適用於一般公告，若讓文字逐一閃爍顯示，那會比較霸氣，這樣可用在告白板、升官、競賽、考試榜單等的顯示。例如，以下程式逐一顯示 8,2,5,7,1,0，每 1 秒自動變換 1 個數字。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte i=0;
4.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
5.  byte d[8];
6.  byte e[6][8]={ { 102, 153, 137, 150, 96, 0, 0, 0},       //8
7.                 { 130, 193, 161, 145, 206, 0, 0, 0},      //2
8.                 { 132, 135, 137, 113, 0, 0, 0, 0},        //5
9.                 { 2, 1, 225, 25, 7, 0, 0, 0},             //7
10.                { 0, 130, 255, 128, 0, 0, 0, 0},          //1
11.                { 126, 129, 129, 129, 126, 0, 0, 0}};     //0
12. long t,t1,t2;
13. void setup() {
14.     DDRB=0xff;
15.     DDRC=0xff;
16. }
17. void loop() {
18.     //放字型
19.     for (int j=0 ;j<=7;j++)
20.         d[j]=e[i][j];
21.     //掃描輸出
22.     t1= millis();
23.     do{
24.         for (int j=0 ;j<=7;j++){
25.             PORTC=c[j];//位址
26.             PORTB=d[j];//資料
27.             delay(1);
28.         }
29.         t2=millis();
30.         t=t2-t1;
31.     }while( (t<1000)  );//1秒，根據需求調整
32.     i=(i+1) % 6;//本例6個字
33. }
```

以上程式如何思考呢？方法一：

1. 依序將以上數字 8,2,5,7,1,0 數位化。同範例 6-1a，在 8*8 的方格紙填入文字，每個數字 8 個 bytes。

2. 可用一維陣列儲存，則此一維陣列長度爲 48，如下：

```
byte b[]= { 102, 153, 137, 150, 96, 0, 0, 0,            //8
            130, 193, 161, 145, 206, 0, 0, 0,           //2
            132, 135, 137, 113, 0, 0, 0, 0,             //5
            2, 1, 225, 25, 7, 0, 0, 0,                  //7
            0, 130, 255, 128, 0, 0, 0, 0,               //1
            126, 129, 129, 129, 126, 0, 0, 0};          //0
```

3. 也可用二維陣列儲存此 6 組輸出碼如下：

```
byte e[6][8]={ { 102, 153, 137, 150, 96, 0, 0, 0},      //8
               { 130, 193, 161, 145, 206, 0, 0, 0},     //2
               { 132, 135, 137, 113, 0, 0, 0, 0},       //5
               { 2, 1, 225, 25, 7, 0, 0, 0},            //7
               { 0, 130, 255, 128, 0, 0, 0, 0},         //1
               { 126, 129, 129, 129, 126, 0, 0, 0}};    //0
```

4. 可用時間、也可用按鈕變換要輸出的數字。
5. 不論是一維陣列或二維陣列，每次都要取 8 個 byte 進行掃描輸出，二維陣列的寫法，如以上程式。
6. 方法二：將 0 到 9 通通依序計算其輸出碼，通通放到陣列，然後將要顯示的文字以字串儲存，每次抓一個字元，且到陣列找對應的輸出碼，並輸出此輸出碼，程式如下：

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte i=0;
4.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
5.  byte d[8];
6.  byte e[10][8]={{ 126, 129, 129, 129, 126, 0, 0, 0},      //0
7.                 { 0, 130, 255, 128, 0, 0, 0, 0},         //1
8.                 { 130, 193, 161, 145, 206, 0, 0, 0},     //2
9.                 { 130, 137, 137, 118, 0, 0, 0, 0},       //3
10.                { 48, 40, 38, 255, 32, 0, 0, 0},         //4
11.                { 132, 135, 137, 113, 0, 0, 0, 0},       //5
12.                { 124, 146, 137, 137, 112, 0, 0, 0},     //6
13.                { 2, 1, 225, 25, 7, 0, 0, 0},            //7
14.                { 102, 153, 137, 150, 96, 0, 0, 0},      //8
15.                { 14, 145, 145, 81, 62, 0, 0, 0}};       //9
```

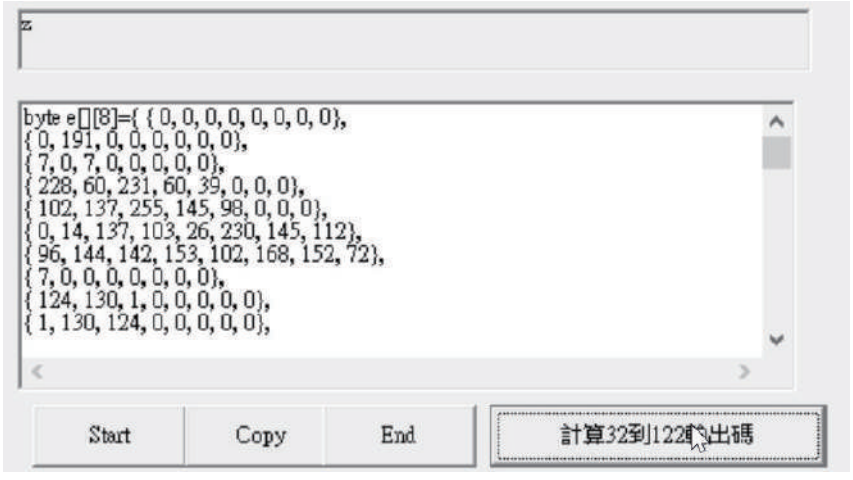
```
long t,t1,t2;
String f="825710";
void setup() {
    DDRB=0xFF;
    DDRC=0xFF;
    Serial.begin(9600);
}
void loop() {
    //放字型
    byte g=byte(f[i]) ; //取到的字元，是其ASCII碼，所以要減48
    Serial.println(g);
    g=g-48;
    Serial.println(g);
    for (int j=0 ;j<=7;j++)
        d[j]=e[g][j];//到e陣列取到對應的輸出碼
    //掃描輸出
    t1= millis();
    do{
        for (int j=0 ;j<=7;j++){
            PORTC=c[j];//位址
            PORTB=d[j];//資料
            delay(1);
        }
        t2=millis();
        t=t2-t1;
    }while( (t<1000)  );//1秒，根據需求調整
    i=(i+1) % 6;//本例6個字
}
```

7. 方法三，將 ASCII（American Standard Code for Information Interchange）從 32 到 122 一一計算其輸出碼，這樣包含了所有空白、數字、大小寫英文字元。此一計算有點多了，可以用人工算，也可以用電腦算，用電腦算的 Visual Basic 程式如下：（更多的 Visual Basic 程式，或想要瞭解此 Visual Basic 程式的撰寫，請參考『Ardunio 字幕機自造與程式設計』）

```
Private Sub Command1_Click()
    lf = Chr(13) + Chr(10) '跳列
    '請先安排一個Picturebox,修改Name爲 pic
    pic.ScaleMode = 3 '以像素爲座標單位
    pic.FontSize = 8
```

```
6.      pic.CurrentX = 0: pic.CurrentY = 0 '輸出位置
7.
8.      's = "byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};" + lf
9.      s = s + "byte e[][8]={ "
10.     '當場計算e[][]
11.     For k = 32 To 122
12.         s = s + "{"
13.         pic.Cls
14.         pic.Print Chr(k)
15.         For i = 0 To 7
16.             p = 0
17.             For j = 1 To 8
18.                 If pic.Point(i, j) = 0 Then '黑點
19.                     p = p + 2 ^ (j - 1)
20.                 End If
21.             Next
22.             If i = 7 Then
23.                 s = s + Str(p) + "}," + lf
24.             Else
25.                 s = s + Str(p) + ","
26.             End If
27.         Next
28.
29.     Next
30.     s = s + "};"
31.     Text2.Text = s
32. End Sub
```

8. 以上程式執行結果如下圖：



9. 配合以下 Arduino 程式，此程式與方法二相同，就可顯示所有數字與大小寫字元。

```
byte i=0;
byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
byte d[8];
byte e[][8]={ { 0, 0, 0, 0, 0, 0, 0, 0},//32 is space
{ 0, 191, 0, 0, 0, 0, 0, 0},
{ 7, 0, 7, 0, 0, 0, 0, 0},
{ 228, 60, 231, 60, 39, 0, 0, 0},
{ 102, 137, 255, 145, 98, 0, 0, 0},
{ 0, 14, 137, 103, 26, 230, 145, 112},
{ 96, 144, 142, 153, 102, 168, 152, 72},
{ 7, 0, 0, 0, 0, 0, 0, 0},
{ 124, 130, 1, 0, 0, 0, 0, 0},
{ 1, 130, 124, 0, 0, 0, 0, 0},
{ 9, 6, 31, 6, 9, 0, 0, 0},
{ 16, 16, 254, 16, 16, 0, 0, 0},
{ 128, 128, 0, 0, 0, 0, 0, 0},
{ 32, 32, 32, 0, 0, 0, 0, 0},
{ 0, 128, 0, 0, 0, 0, 0, 0},
{ 224, 24, 7, 0, 0, 0, 0, 0},
{ 126, 129, 129, 129, 126, 0, 0, 0},
{ 0, 130, 255, 128, 0, 0, 0, 0},
{ 130, 193, 161, 145, 206, 0, 0, 0},
{ 130, 137, 137, 118, 0, 0, 0, 0},
{ 48, 40, 38, 255, 32, 0, 0, 0},
{ 132, 135, 137, 113, 0, 0, 0, 0},
{ 124, 146, 137, 137, 112, 0, 0, 0},
{ 2, 1, 225, 25, 7, 0, 0, 0},
{ 102, 153, 137, 150, 96, 0, 0, 0},
{ 14, 145, 145, 81, 62, 0, 0, 0},
{ 0, 136, 0, 0, 0, 0, 0, 0},
{ 0, 136, 136, 0, 0, 0, 0, 0},
{ 16, 40, 40, 68, 68, 0, 0, 0},
{ 72, 72, 72, 72, 72, 0, 0, 0},
{ 68, 68, 40, 40, 16, 0, 0, 0},
{ 2, 1, 177, 14, 0, 0, 0, 0},
{ 0, 60, 66, 153, 165, 157, 162, 92},
{ 128, 224, 156, 19, 156, 224, 128, 0},
```

```
38. { 129, 255, 137, 137, 137, 118, 0, 0},
39. { 60, 66, 129, 129, 130, 71, 0, 0},
40. { 129, 255, 129, 129, 129, 66, 60, 0},
41. { 129, 255, 137, 137, 157, 195, 0, 0},
42. { 129, 255, 137, 9, 29, 3, 0, 0},
43. { 60, 66, 129, 129, 146, 247, 16, 0},
44. { 129, 255, 137, 8, 137, 255, 129, 0},
45. { 129, 255, 129, 0, 0, 0, 0, 0},
46. { 192, 129, 127, 1, 0, 0, 0, 0},
47. { 129, 255, 137, 20, 163, 193, 128, 0},
48. { 129, 255, 129, 128, 128, 192, 0, 0},
49. { 129, 255, 134, 56, 192, 56, 134, 255},
50. { 129, 255, 134, 24, 97, 255, 1, 0},
51. { 60, 66, 129, 129, 129, 66, 60, 0},
52. { 129, 255, 137, 9, 9, 6, 0, 0},
53. { 60, 66, 129, 129, 129, 66, 60, 0},
54. { 129, 255, 137, 25, 41, 198, 128, 0},
55. { 230, 73, 145, 146, 103, 0, 0, 0},
56. { 0, 3, 129, 255, 129, 3, 0, 0},
57. { 1, 127, 129, 128, 129, 127, 1, 0},
58. { 1, 7, 57, 192, 57, 7, 1, 0},
59. { 1, 15, 56, 225, 31, 49, 192, 57},
60. { 129, 195, 165, 24, 165, 195, 129, 0},
61. { 1, 3, 141, 240, 141, 3, 1, 0},
62. { 195, 161, 145, 137, 133, 195, 0, 0},
63. { 0, 255, 1, 0, 0, 0, 0, 0},
64. { 3, 28, 224, 0, 0, 0, 0, 0},
65. { 1, 1, 255, 0, 0, 0, 0, 0},
66. { 8, 6, 1, 6, 8, 0, 0, 0},
67. { 0, 0, 0, 0, 0, 0, 0, 0},
68. { 1, 2, 0, 0, 0, 0, 0, 0},
69. { 80, 168, 168, 112, 128, 0, 0, 0},
70. { 1, 127, 136, 136, 112, 0, 0, 0},
71. { 112, 136, 136, 80, 0, 0, 0, 0},
72. { 112, 136, 137, 127, 128, 0, 0, 0},
73. { 112, 168, 168, 176, 0, 0, 0, 0},
74. { 132, 254, 133, 1, 0, 0, 0, 0},
75. { 144, 232, 168, 152, 8, 0, 0, 0},
76. { 129, 255, 8, 240, 128, 0, 0, 0},
77. { 136, 249, 128, 0, 0, 0, 0, 0},
```

```
78. { 8, 249, 0, 0, 0, 0, 0, 0},
79. { 129, 255, 32, 216, 136, 0, 0, 0},
80. { 129, 255, 128, 0, 0, 0, 0, 0},
81. { 8, 240, 8, 8, 240, 8, 8, 240},
82. { 136, 248, 8, 240, 128, 0, 0, 0},
83. { 112, 136, 136, 136, 112, 0, 0, 0},
84. { 8, 248, 136, 136, 112, 0, 0, 0},
85. { 112, 136, 136, 248, 8, 0, 0, 0},
86. { 136, 240, 136, 0, 0, 0, 0, 0},
87. { 144, 168, 168, 72, 0, 0, 0, 0},
88. { 8, 126, 136, 0, 0, 0, 0, 0},
89. { 8, 120, 128, 248, 128, 0, 0, 0},
90. { 8, 56, 192, 56, 8, 0, 0, 0},
91. { 8, 56, 192, 56, 192, 56, 8, 0},
92. { 136, 216, 32, 216, 136, 0, 0, 0},
93. { 8, 56, 192, 56, 8, 0, 0, 0},
94. { 152, 200, 168, 152, 200, 0, 0, 0},
95. };
96. long t,t1,t2;
97. String f="This is a book 825710";
98. void setup() {
99.     DDRB=0xFF;
100.     DDRC=0xFF;
101.     Serial.begin(9600);
102. }
103. void loop() {
104.     //放字型
105.     byte g=byte(f[i]) ;
106.             //取到的字元，是其ASCII碼，本例從32開始計算，所以要減32
107.     Serial.print(i);Serial.print(",");
108.     Serial.println(g);
109.     g=g-32;
110.     Serial.println(g);
111.     for (int j=0 ;j<=7;j++)
112.         d[j]=e[g][j];
113.     //掃描輸出
114.     t1= millis();
115.     do{
116.         for (int j=0 ;j<=7;j++){
117.             PORTC=c[j];//位址
```

```
118.              PORTB=d[j];//資料
119.              delay(1);
120.          }
121.          t2=millis();
122.          t=t2-t1;
123.      }while( (t<1000)  );//1秒，根據需求調整
124.      i=(i+1) % f.length();//長度
125. }
```

10. 亮滅。因爲有時候連續兩個字元若相同，例如顯示『88255』，若沒有加上顯示空白碼，會變成『825』。要控制點陣 LED 亮滅，一樣要掃瞄空白碼 (0x0) 一個短暫的時間，顯示空白碼的測試程式如下，請加在以上程式『掃描輸出』的後面就可以。

```
1.      t1= millis();
2.      do{
3.          for (int j=0 ;j<=7;j++){
4.              PORTC=c[j];//位址
5.              PORTB=0;//資料滅掉
6.              delay(1);
7.          }
8.          t2=millis();
9.          t=t2-t1;
10.     }while( (t<200)  );//滅掉0.2秒，根據需求調整
```

自我練習（以下程式，電路請自行設計，且繪出電路圖）

1. 同範例 6-1c，請增加二個指撥開關調整文字輸出速度？
2. 同範例 6-1c，請增加一個按鈕，使用此按鈕變換數字？例如，每按一下按鈕，變換下一個文字。
3. 請寫一程式，可以每按一下按鈕，就產生 1 個 1 到 6 的亂數，且由點陣 LED 輸出且停住一直顯示。
4. 請寫一程式，使用 8 個按鈕，編號是 1 到 8，每按一個按鈕，分別顯示對應的 1 到 8。
5. 請寫一程式，可以用 8 個指撥開關指派輸出 0 到 8 的數字。
6. 請寫一程式，可以用 3 個指撥開關指派輸出 0 到 7 的數字。

※ 7. 以上 8*8 點陣 LED 僅能顯示數字、英文字母與筆畫很少的中文字，請有樂趣的同學，自行使用以上原理與演算法，自製 16*16 與 16*64 字幕機，這樣就可用跑馬燈或告白板顯示任何中文字。

範例 6-1d

動畫。

相信大家都看過紅綠燈下的小綠人動畫，動畫製作步驟如下：

1. 先設計共幾個畫面。本例假設 4 個畫面，如表 6-6。
2. 逐一填入每個畫面的亮點。
3. 逐一計算每個畫面 8 個 col 的值。

表6-6a

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								1
R_5								
R_6								
R_7								
R_8								
值	0	0	0	0	0	0	0	08

表6-6b

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								1
R_4							1	1
R_5								1
R_6								
R_7								
R_8								
值	0	0	0	0	0	0	08	1c

表6-6c

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								1
R_3							1	1
R_4						1	1	1
R_5							1	1
R_6								1
R_7								
R_8								
值	0	0	0	0	0	08	1c	3e

表6-6d

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								1
R_2							1	1
R_3						1	1	1
R_4					1	1	1	1
R_5						1	1	1
R_6							1	1
R_7								1
R_8								
值	0	0	0	0	08	1c	3e	7f

4. 將以上四個畫面的 col 值，以陣列儲存。如以下 e[4][8] 陣列。
5. 逐一輸出以上 e[4][8] 陣列，其方法爲每次一個畫面，將此畫面放入 d[] 陣列。

```
1.  //PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
2.  //PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
3.  byte i=0;
4.  byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};//位址
5.  byte d[8];
6.  byte e[4][8]={ { 0, 0, 0, 0, 0, 0, 0, 0x08},
7.                 { 0, 0, 0, 0, 0, 0,  0x08,0x1c},
8.                 { 0, 0, 0, 0, 0, 0x08,0x1c,0x3e},
9.                 { 0, 0, 0, 0, 0x08,0x1c,0x3e,0x7f}};
10. long t,t1,t2;
11. void setup() {
12.     DDRB=0xff;
13.     DDRC=0xff;
14. }
15. void loop() {
16.   //放字型
17.   //逐一輸出以上e[4][8]陣列，其方法爲每次一個畫面，將此畫面放入d[]陣列。
18.     for (int j=0 ;j<=7;j++)
19.         d[j]=e[i][j];
20.     //掃描輸出d[]陣列
21.     t1= millis();
22.     do{
23.         for (int j=0 ;j<=7;j++){
24.             PORTC=c[j];//位址
25.             PORTB=d[j];//資料
26.             delay(1);
27.         }
28.         t2=millis();
29.         t=t2-t1;
30.     }while( (t<400)  );//1秒，根據需求調整
31.     i=(i+1) % 4;//本例共4個畫面
32. }
```

自我練習

1. 請發揮自己的美感，製作一個動物跑、或飛、或跳的畫面。

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
R_1								
R_2								
R_3								
R_4								
R_5								
R_6								
R_7								
R_8								
值								

16*16點陣LED

前面是 8*8LED，僅能顯示英文字母與數字，無法顯示中文字，要顯示中文字，至少要 16*16，大家可以自己使用 4 個 8*8 拚一個 16*16，也可以直接購買 16*16 點陣 LED。圖 6-5 是筆者拿到的兩種廠牌 16*16 點陣 LED 的腳位圖。請把廠牌商標放在下面，各位可以發現，前面 8*8 點陣 LED 腳位沒有順序，但 16*16 的腳位竟然都按照順序排列了，上面 16 支腳由左到右分別 R16,R15,R14…R1；下面 16 支腳，由左到右分別是 C1,C2,C3….C16，這樣使用起來比較方便。

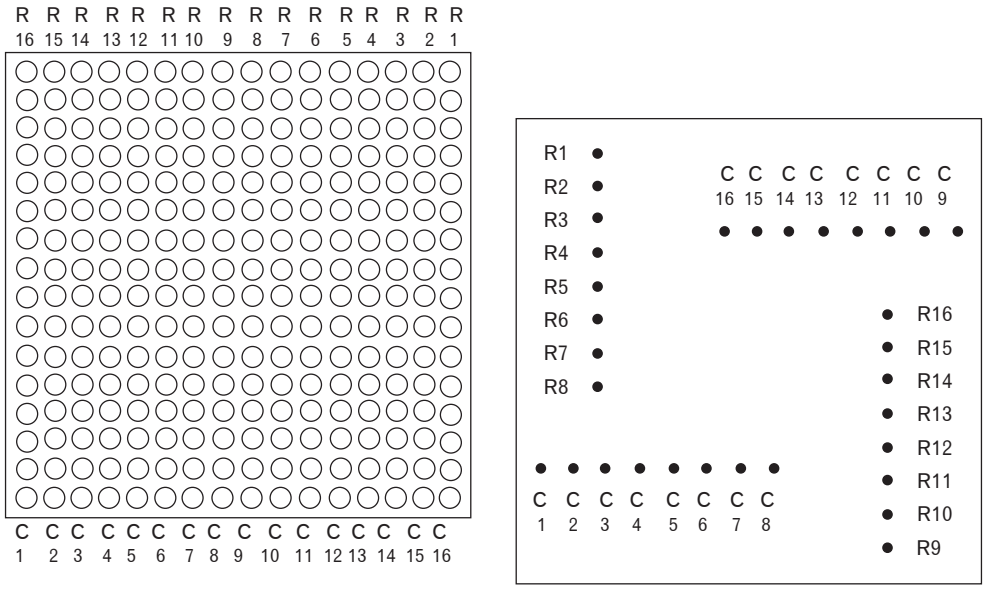


圖 6-5　16*16 點矩陣 LED 腳位圖

請將點陣 LED 插在麵包板，但因爲尺寸與麵包板不合，只好將麵包板拆一排底座，這樣才能插下去，然後用三用電表檢驗，三用電表撥 R*10 檔位，正極 (黑棒) 接 R1，負極 (紅棒) 接 C1，請留意左上角第 1 個 LED 是否亮起，餘此類推。

▮簡易測試

因爲 Arduino2560 腳位超多，所以請在麵包板上接線如圖 6-6，這樣就可進行簡易測試。

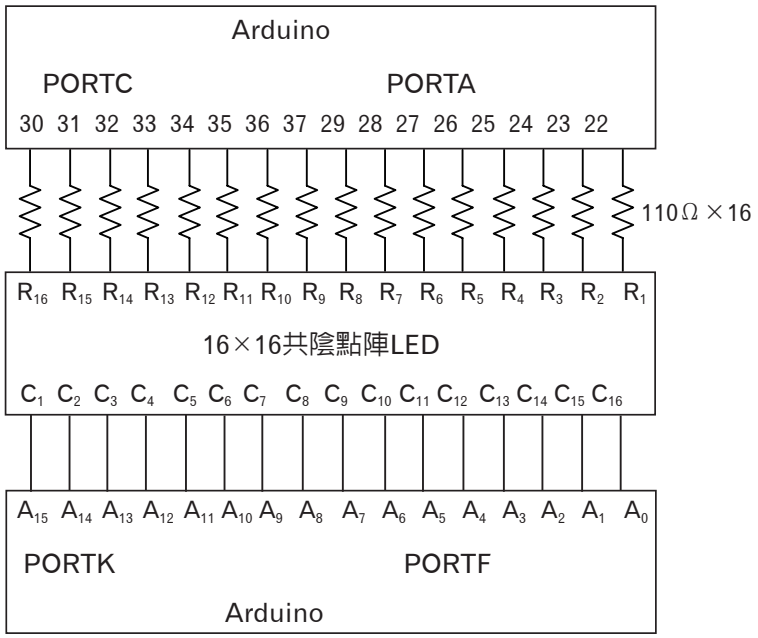


圖 6-6　16*16 點矩陣 LED 簡易驅動電路圖

上圖 R_1 ～ R_8 連接到 PORTA，R_9 ～ R_{16} 連接到 PORTC，C_1~C_8 連接到 PORTK ，C_9~C_{16} 連接到 PORTF。

▮硬體測試

請輸入以下程式，觀察 LED 有沒有全亮。

```
1.  //PORTA 22~29 接R1~R8
2.  //PORTC 37~30 接R9~R16
3.  //PORTF A0~A7 接C1~C8
4.  //PORTK A8~A15 接C9~C16
5.  void setup() {
6.    DDRA=0xFF;  DDRC=0xFF;  DDRF=0xFF;  DDRK=0xFF;
7.  }
8.  long a[]={0x7fff,0xbfff,0xdfff,0xefff,0xf7ff,0xfbff,0xfdff,
              0xfeff,0xff7f,0xffbf,0xffdf,0xffef,0xfff7,0xfffb,
              0xfffd,0xfffe};
9.  int i=0;
10. void loop() {
```

```
11.   PORTA=0xFF;
12.   PORTC=0xFF;
13.   //從C1,C2…C16,每次僅一隻腳低電位
14.   PORTK=a[i]/256;//高位元
15.   PORTF=a[i]%256;//低位元
16.   delay(1);//delay(1)全亮，delay(500);輪流亮
17.   i=(i+1)%16;
18. }
```

補充說明

1. 請將以上程式的 delay(1) 改爲 delay(500)，並觀察每一行 LED 是否照順序輪流亮。
2. 以上 a 陣列共 16 個，每次都只有 1 個位元是低電位，這樣就能將資料輪流送到每一行，只要速度夠快，看起來就會同時亮，完成掃描輸出的要求。
3. Arduino 是 8 位元單晶片，所以 16 位元就要分兩次輸出，作法是將 a[] 陣列的長整數除以 256，整數商就是高位元，餘數就是低位元。

▮實用電路

請摸一下以上電路 Arduino 的 CPU，這一 CPU 有點溫度，這樣僅能短時間實驗，長時間運轉則不行，實用的產品都要緩衝，不能讓 CPU 提供訊號進行控制，又要提供功率，所以實用電路如圖 6-7：也就是 CPU 僅提供訊號，高電位出去的電流由 74244 提供，低電位回去的電流由 74138 吸收，這樣就可減輕 Arduino 的功率負擔，請繼續看以下 IC 的說明。

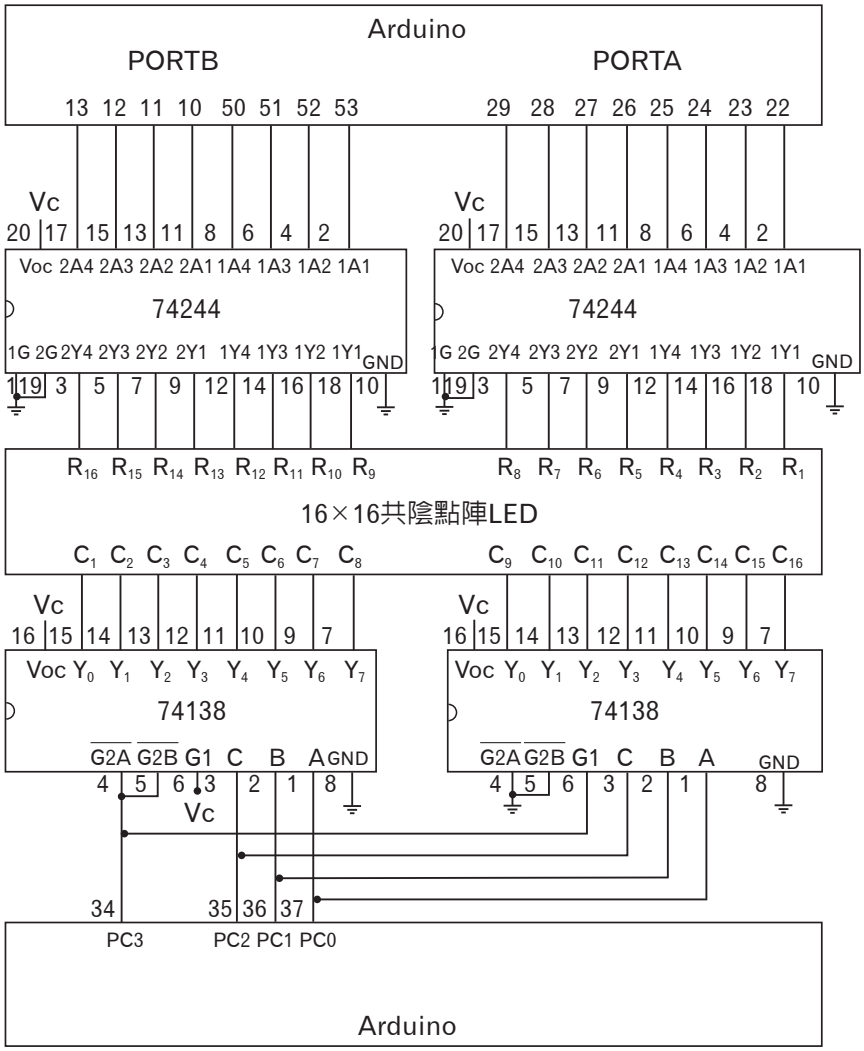


圖 6-7　16*16 點矩陣 LED 實用驅動電路圖

Arduino 的 PORTA 與 PORTB，本例稱爲資料線，資料線都連接到 74244 的資料輸入端，74244 的輸出端再連線到點陣 LED 的 R1..R16;其次，Arduino 的 PORTC（PC3,PC2,PC1,PC0，由左而右分別是 34,35,36,37)，本例稱爲控制線，將會用來控制 C1..C16，讓每次只有 1 條線是低電位，這樣就可以將資料傳到指定的 Column。

74244

74244 是一個電流緩衝器，此 IC 的腳位圖如圖 6-8（於網際網路輸入 74244 就可以得到很多相關資料），由 244 供電給 LED，這樣可以保護 Arduino 的控制晶片 (ATmega2560)。(IC 腳位編號說明:請將 IC 缺口朝左，缺角下面編號是 1，然後逆時針編號）

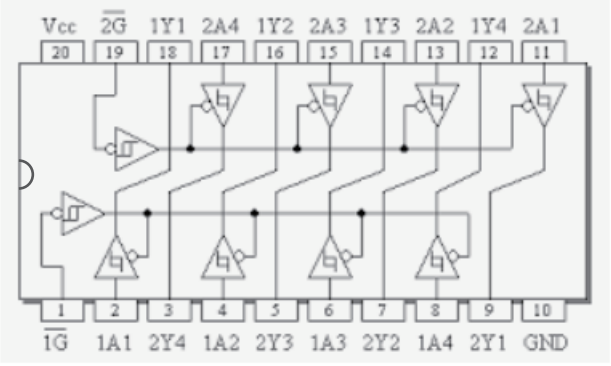


圖 6-8　74244 內部電路圖

74138

74138 是 3 對 8 的解碼器，其腳位圖如圖 6-9：

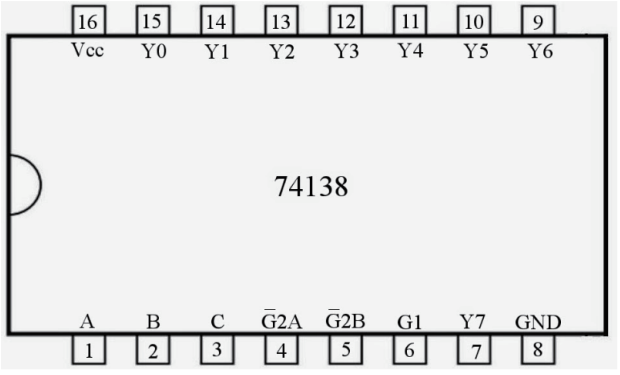


圖 6-9　74138 內部腳位圖

74138 的眞值表，如表 6-7a。

▶ 表 6-7a　74138 真值表

致能輸入 En			輸入			輸出							
G1	$\overline{G2A}$	$\overline{G2B}$	C	B	A	$\overline{Y}_0$	$\overline{Y}_1$	$\overline{Y}_2$	$\overline{Y}_3$	$\overline{Y}_4$	$\overline{Y}_5$	$\overline{Y}_6$	$\overline{Y}_7$
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	1	X	X	X	1	1	1	1	1	1	1	1

也就是 C,B,A 分別給 000,001,010,…111 就可以讓 Y_0，Y_1 ～ Y_7 分別得到低電位，這樣就可完成每次一條線低電位的掃描功能。其次，爲了可以擴充，74138 還有『致能 / 除能腳』，由以上眞值表可知，要讓 74138 致能的接法是

$En=G_1*(/G_{2A}+/G_{2B})$

所以電路設計如圖 6-7，這樣當 PC_3 是 0 時，第 1 個 138 致能 (從左邊算起)，第 2 個 138 除能；其次，當 PC_3 是 1 時，第 1 個 138 除能，右邊第 2 個 138 致能，以上致能與除能請複習數位邏輯設計。綜合以上 C,B,A 與 G_1,/G_{2A},/G_{2B}，所以此電路部分眞值表如表 6-7b(PC_3、PC_2、PC_1、PC_0 是輸入，C_1..C_{16} 是輸出)：以下僅列出時序 0,1,2,7,8,9,15，其餘時序 3,4,5…依此類推。

▶ 表 6-7b　16*16 點陣 LED 電路真值表

PORTC	PC_3	PC_2	PC_1	PC_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

這樣等一會寫程式時，只要 PORTC 輪流給 0 到 15，C_1,C_2…C_{16} 就可以每次僅一條線輪流得到低電位，達到掃描輸出的需求。

軟體測試

以下程式可讓 LED 全亮與全暗。

```
1.  //PORTA 22~29 接低八位元1A1,1A2,1A3,1A4,2A1,2A2,2A3,2A4
2.  //PORTB 53,52,51,50,10,11,12,13 接高八位元
       1A1,1A2,1A3,1A4,2A1,2A2,2A3,2A4
3.  //PORTc 37,36,35 接A,B,C
4.  void setup() {
5.    DDRF=0xFF; DDRK=0xFF; DDRC=0xFF;
6.  }
7.  void loop() {
8.    int i,j;
9.    //全亮
10.   for (i=0;i<=100;i++)//控制時間
11.     for (j=0 ;j<=15;j++){
12.       PORTC=j;//位址
13.       PORTA=0xff;//資料
14.       PORTB=0xff;//資料
15.       delay(1);
16.     }
17.   //全暗
18.   for (i=0;i<=10;i++)//控制時間
```

```
19.     for (j=0 ;j<=15;j++){
20.       PORTC=j;//位址
21.       PORTA=0;//資料
22.       PORTB=0;//資料
23.       delay(1);
24.     }
25. }
```

以上外迴圈 i 用來控制全亮與全滅的時間，請自行調整，並觀察結果。

自我練習

1. 請用以上 16*16 點矩陣 LED，完成前面 8*8 點矩陣 LED 的所有範例與自我練習。

6-2 液晶顯示器控制

▶ 表 6-8　本節材料表

編號	零件名稱	數量	備註
1	16*2 液晶顯示器	1 ～ 2	
2	電阻 330Ω	1 ～ 2	
3	10kΩ 可變電阻	1 ～ 2	
4	按壓開關	8	

液晶顯示器（Liquid Crystal Display，以下以縮寫 LCD 表示），如圖 6-10 所示，爲目前使用最爲廣泛的顯示裝置之一，諸如計算機、電子儀器、事務機器、電器產品等都可以見到其蹤影。它能夠顯示英文、數字、各種特殊符號等各種字型，而且本身不發光，當然非常省電，必須藉助外界光源才可以讀取其內容（此與 LED 不同，LED 本身自動發光，所以不需外界光源）。下圖是本書實驗板所使用 LCD 實體圖，每一隻接腳還有腳位名稱，請用雙長排針焊接，杜邦母接線也可以從上面直接連接，這樣比較不會插錯。

圖 6-10　LCD 實體圖

圖 6-11 是 LCD 腳位編號與腳位名稱：

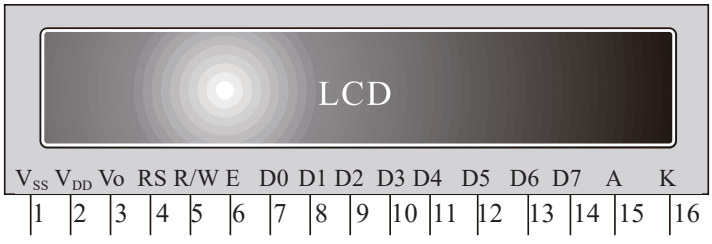


圖 6-11　LCD 腳位圖

以上腳位說明如下：

1. V_{SS}：電源接地腳，請連接 Gnd。
2. V_{DD}：電源正極腳，請連接 5V。
3. V_0：亮度對比調整腳，通常是接 Gnd 最亮，接 5V 最暗，請連接至 1 ～ 10k 可變電阻的中間輸出腳，調整適當的對比，有些是先接 470~2k 再接地，有些 LCD 直接接地也行。
4. RS：暫存器選擇線（Register Select）。低電位代表選擇指令暫存器，高電位代表選擇資料暫存器。
5. R/W：讀寫選擇線。高電位代表讀取 LCD 的資料，低電位代表寫入 LCD。本書範例僅拿 LCD 當輸出，所以直接將此腳位接地；若使用含有觸控輸入的 LCD，那此腳位就不能強制接地，而必要由軟體依照所需執行指令的功能，設爲輸出或輸入。
6. E：資料致能線。採用負緣觸發方式，也就是此腳位由 1 變 0 的負緣時段，模組開始讀取或寫入資料線資料。
7. d7~d0：LCD 模組的資料線。資料線有 8 條，就要耗掉單晶 8 隻腳，但是 LCD 廠商爲了讓使用者節省腳位，也允許您僅用 d4,d5,d6,d7 傳輸資料，請看以下範例。
8. A：背光 LED 的電源正極腳，請先連接 100 ～ 330Ω 再連接到電源 5V。（有些是直接連接 5V，有些 LCD 不接也行）
9. K：背光 LED 的接地腳，請連接到 Gnd。

控制方式

在 Arduino 以前，您必須參考硬體技術手冊，自己寫程式控制 LCD，這就留待讀者自行研究。但是 Arduino 已經將這些程式包裝成 LiquidCrystal 類別，您只要建構此類別，就可以使用此物件。LiquidCrystal 類別的建構子如下：

```
LiquidCrystal(rs,e,d4,d5,d6,d7)
LiquidCrystal(rs,rw,e,d4,d5,d6,d7)
LiquidCrystal(rs,e,d0,d1,d2,d3,d4,d5,d6,d7)
LiquidCrystal(rs,rw,e,d0,d1,d2,d3,d4,d5,d6,d7)
```

共有四種多型表現，本例使用第一種建構子，以 lcd 爲物件名稱，建構 LiquidCrystal 類別，程式如下：

```
LiquidCrystal lcd(17,16,15,14,9,8);
```

物件建構以後，往後就可以使用物件名稱，本例爲 lcd，初始化與起動此 LCD。例如：以下程式，初始化此 LCD 爲 2 列 16 行。

```
lcd.begin(16,2)
```

此一建構子，因爲 R/W 腳位未用，您必須使用硬體接線的方式強迫接地，表示所有指令都是寫入。所以電路如圖 6-12，可變電阻與 330Ω 本書實驗板都有。Vo 的接法：請將圖 2-6 實驗板可變電阻輸出 (J20) 接到 Vo；腳位 A 的接法：從 5V 接線到 330Ω (J16)，再從 J17 連接到 LCD 腳位 15；腳位 1 與 2 內部已經連接，以上 J20,J16,J17 請對照圖 2-6 的實驗板。

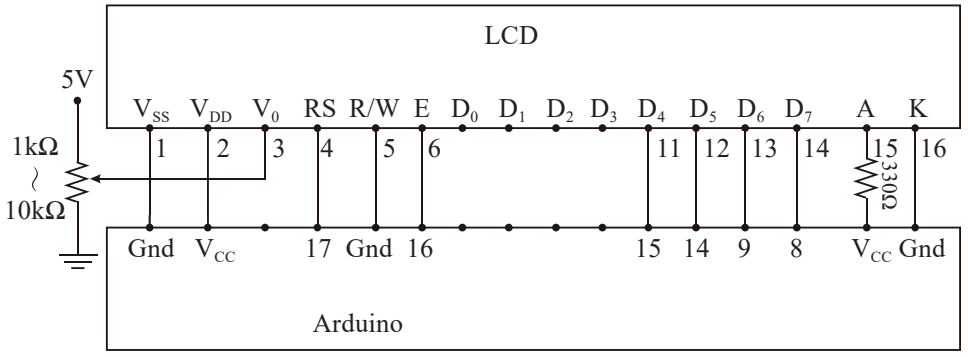


圖 6-12　LCD 驅動電路圖

範例 6-2a

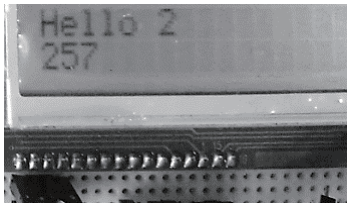
LCD 硬體測試。先檢查接線與硬體是否正確，觀察是否有輸出結果。

操作步驟

1. 完成圖 6-12 LCD 電路圖。
2. 鍵入以下程式，請觀察是否出現輸出結果。

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd(17,16,15,14,9,8);//以lcd建構LiquidCrystal類別
3.  void setup() {
4.    lcd.begin(16,2);  //初始化LCD，游標預設(0,0)
5.    lcd.write("Hello  2 ");//only write char
6.  }
7.  void loop() {
8.    lcd.setCursor(0,1);//(col,row)左上角是(0,0),本例設定輸出在(0,1)
9.    lcd.print(millis()/1000);//輸出程式執行時間
10. }
```

3. 請觀察執行結果是否如下，若沒有輸出結果，請重新檢查接線是否全部正確。
4. 若電路沒錯，但沒有輸出結果，請調整可變電阻，調整輸出亮度。



▮ begin(columns,rows)

begin() 函式提供 LCD 初始化與啓動 LCD 功能，columns 是行數，rows 是列數。例如：以下程式，初始化此 LCD 爲 2 列 16 行。

```
lcd.begin(16,2)
```

以上 lcd 是建構類別的物件名稱。

▮ setCursor

設定游標位置。其語法如下：

```
setCursor(col,row)
```

▮ write

輸出函式，語法如下：

```
write(string data)
```

▮ print

輸出函式，語法如下：

```
print( data,進位方式)
```

前面的 write 僅限於輸出 string 型態的資料，但是 print 則可輸出任何型態的資料，若是數值，還可指定進位方式，例如，BIN（二進位）、OCT（八進位）、DEC（十進位，預設）、HEX（十六進位）。

▮ cursor()/noCursor()

設定游標顯示與否。

▮ blink()/noBlink()

設定游標閃爍與否。

▮ display()/noDisplay()

設定螢幕的文字顯示或關閉。

▮ leftToRight()

設定文字的顯示由左到右。

▮ rightToLeft()

設定文字的顯示由右到左。

▮ createChar()

自建字元。一個 LCD 可以自建 8 個 5*7 的字元，請看範例 6-2e。

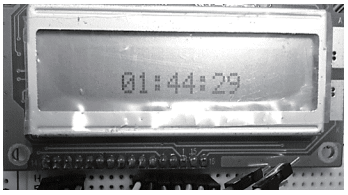
自我練習

1. 請寫一個程式，有 8 個按壓開關，當按第 1 個開關顯示 1，第 2 個開關顯示 2，依此類推。
2. 請寫一個程式，可以每秒出現 1 個 0 到 999 的整數亂數，且用 LCD 顯示其值。
3. 請寫一個程式，有 8 個指撥開關，可用 LCD 顯示其代表的值。例如，只有最後 3 個指撥開關 ON，則其值輸出爲 7。

範例 6-2b

本例將於 LCD 顯示「時：分：秒」的電子時鐘。

輸出結果



程式列印

本例以時間 t 當作計時器，每秒累加 1，因爲一直累加，t 值當然會溢位，爲了避免溢位，所以將累加值除以 24*60*60 取餘，這樣就可以將 t 控制在一天的 0~24*60*60-1 秒內。

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd(17,16,15,14,9,8);
3.  void setup() {
4.    lcd.begin(16,2);
5.  }
6.  unsigned long t=(unsigned long) 1*60*60+44*60+29;//時間初值01:44:29
7.  unsigned long t1=(unsigned long) 24*60*60;
8.  void loop() {
9.      int h,m,s;
10.     s=t%60;
11.     m=(t/60)%60;
12.     h=t/3600;
13.     lcd.setCursor(4,1);
14.     if (h<10)
15.       lcd.print("0");
16.     lcd.print(h);
17.     lcd.print(":");
18.     if (m<10)
19.       lcd.print("0");
20.     lcd.print(m);
21.     lcd.print(":");
22.     if (s<10)
```

```
23.       lcd.print("0");
24.     lcd.print(s);
25.     t=(t+1) % (t1);
26.     delay(1000);
27. }
```

本例 24*60*60 超過整數 int 所能表示的範圍 0~65535，所以要宣告爲 unsigned long ，且 24*60*60 已經超過 65535，也要先用 (unsigned long) 轉型，請先複習一年級程式設計。

自我練習

1. 同範例 6-2b，但增加 2 個按鈕，可調整時與分。
2. 請以 LCD 爲輸出，重做範例 5-6a 計數器。共有 4 個按鈕，可加 1、減 1、加 10、減 10。
3. 請以 LCD 爲輸出，重做範例 5-7b 節的倒數計時器。

範例 6-2c

跑馬燈。與四位七段顯示器相同的問題，一個 LCD 僅能顯示 16 個字元，若超過 16 字元時，如何處理呢？

程式列印

前面 8*8 點矩陣 LED 的跑馬燈效果是自己寫程式，LCD 的跑馬燈效果則利用現成 scrollDisplayLeft() 函式，程式設計者只要一直丟資料就好，超過螢幕寬度時，文字會自己移動，所以參考程式如下：

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd(17,16,15,14,9,8);
3.  void setup() {
4.    lcd.begin(16,1);
5.    lcd.leftToRight();
6.    lcd.setCursor(16,0);
7.  }
8.  char a[]= "  Hello.There are four people in my family.";
9.  void loop() {
10.   int i=0;
```

```
11.   do{
12.       lcd.write(a[i]);
13.       lcd.scrollDisplayLeft();//設定LCD以跑馬燈方式輸出
14.       delay(500);
15.       i++;
16.   } while(a[i]!='\0');//字串結束字元
17. }
```

自我練習

1. 請由微處理機每秒產生 1 個 1 到 1000 的亂數，且由 LCD 以跑馬燈的方式一直連續輸出，例如：『1：22　2：325…』數字與數字之間請流 1 個空白。

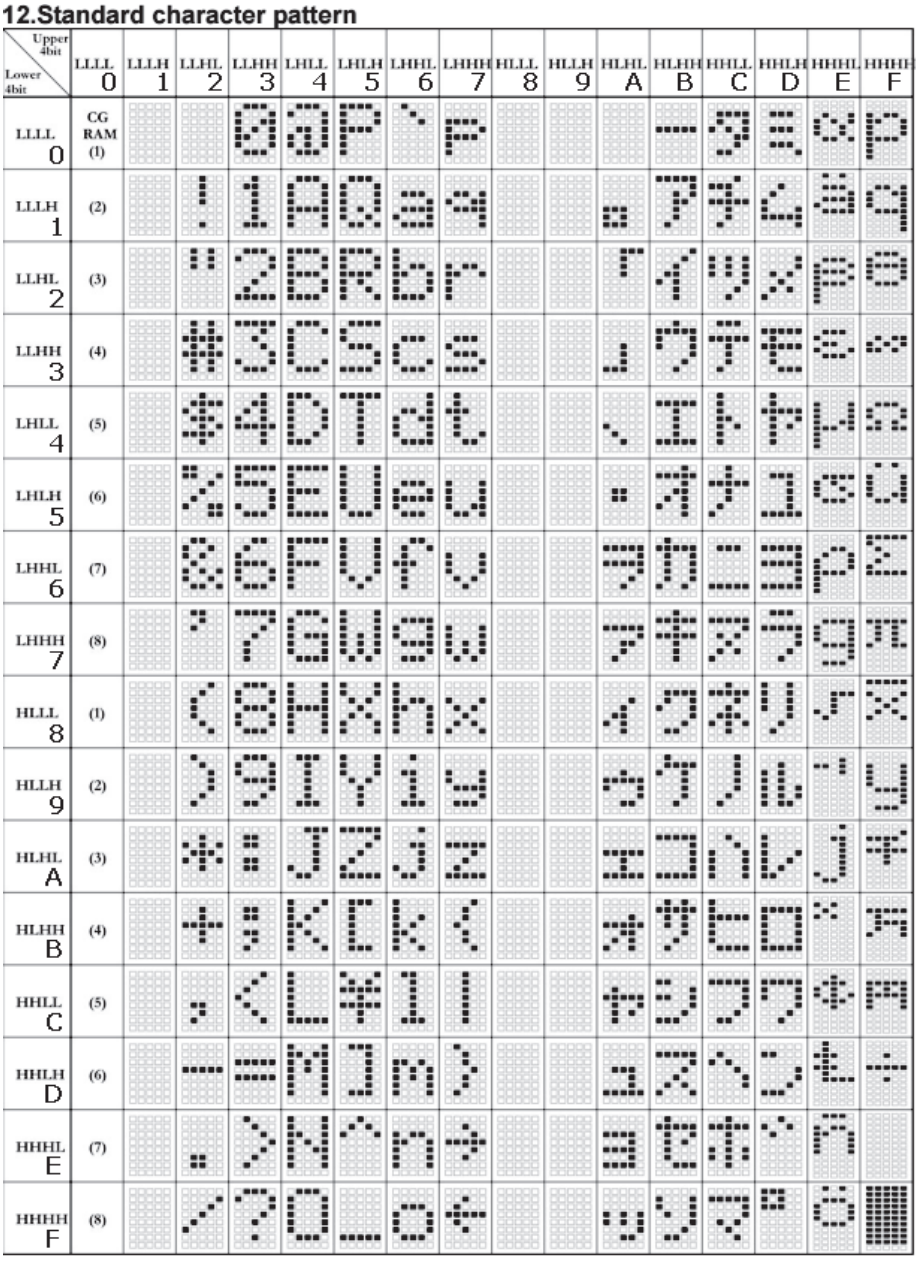
預設字元

LCD 預先儲存的字元如表 6-9，一共存放著 160 個 5*7 點陣字型，如表 6-9 所示。例如，溫度符號由上表可知其編碼是「B11011111」，所以，以下敘述可印出溫度符號上標的小圈圈。

```
lcd.write(B11011111); //僅能使用write()函式，強迫將輸出當作字元，
//轉爲字元輸出
```

▶ 表 6-9　LCD 預儲字元表

12.Standard character pattern



MODULE NO : ADM1602K-NSW-FBS/3.3V

範例 6-2d

示範印出溫度符號上標的小圈圈。

程式列印

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd(17,16,15,14,9,8);
3.  void setup() {
4.    lcd.begin(16,2);
5.    lcd.setCursor(4,0);
6.    int a=24;
7.    lcd.print(a);
8.    lcd.write(B11011111); //僅能使用write()函式，強迫將輸出當作字元，
9.    //轉爲字元輸出
10.   lcd.print("C");
11. }
```

範例 6-2e

本例示範自建『年』、『月』、『日』等三個字。

執行結果

實習步驟

1. 一個 LCD 可自造 8 個 5*7 字元，以下示範自造『年』、『月』、『日』三個字。
2. 每個字是 5*7，但字與字中間要空一列，所以年、月、日的輸出碼如下表：

1					0x10
1	1	1	1	1	0x1f
			1		0x02
	1	1	1	1	0x0f
	1		1		0x0a
1	1	1	1	1	0x1f
			1		0x02
					0x00

	1	1	1	1	0x0f
	1			1	0x09
	1	1	1	1	0x0f
	1			1	0x09
	1	1	1	1	0x0f
	1			1	0x09
1			1	1	0x13
					0x00

	1	1	1	1	0x0f
	1			1	0x09
	1			1	0x09
	1	1	1	1	0x0f
	1			1	0x09
	1			1	0x09
	1	1	1	1	0x0f
					0x00

3. 以陣列儲存如下：

```
const byte year[]={0x10,0x1f,0x02,0x0f,0x0a,0x1f,0x02,0x00};
const byte month[]={0x0f,0x09,0x0f,0x09,0x0f,0x09,0x13,0x00};
const byte day[]={0x0f,0x09,0x09,0x0f,0x09,0x09,0x0f,0x00};
```

4. 以 createChar() 函式放到記憶體。

```
lcd.createChar(0,year);  //放入「年」
lcd.createChar(1,month); //放入「月」
lcd.createChar(2,day);   //放入「日」，可放0~7等7個字元
```

程式列印

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd(17,16,15,14,9,8);
3.  void setup() {
4.    lcd.begin(16,2);
5.    const byte year[]={0x10,0x1f,0x02,0x0f,0x0a,0x1f,0x02,0x00};
6.    const byte month[]={0x0f,0x09,0x0f,0x09,0x0f,0x09,0x13,0x00};
7.    const byte day[]={0x0f,0x09,0x09,0x0f,0x09,0x09,0x0f,0x00};
8.    lcd.createChar(0,year);
9.    lcd.createChar(1,month);
10.   lcd.createChar(2,day);
11.   int y=107,m=6,d=20;
12.   lcd.setCursor(4,0);
13.   lcd.print(y);
14.   lcd.write(byte(0)); //輸出「年」
15.   lcd.print(m);
16.   lcd.write(byte(1)); //輸出「月」
17.   lcd.print(d);
18.   lcd.write(byte(2)); //輸出「日」
19. }
```

自我練習

1. 同範例 6-2d，還可同時顯示時間與日期。
2. 請自行自造「日一二三四五六」等 8 個字。

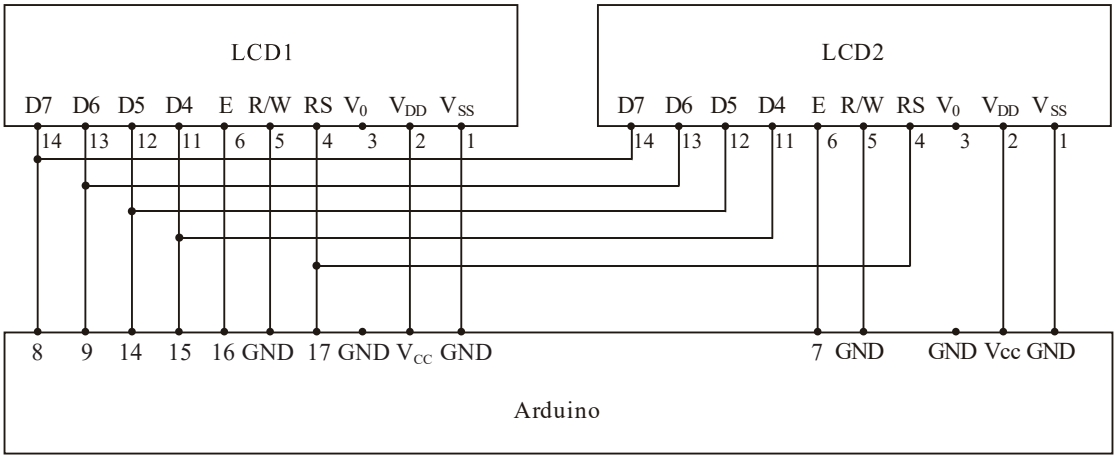
同時使用多個LCD

前面僅使用一個 LCD 就用掉 5 隻腳，若要同時連接多個 LCD，例如，銀行的叫號器，每個行員都要至少一個 LCD，要如何辦到呢？答案就是要使用 Enable 接腳，以下範例示範連接兩個 LCD 的電路圖與程式。

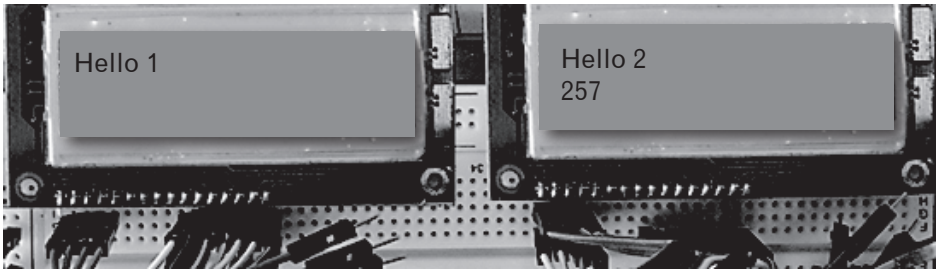
範例 6-2f

示範同時使用兩個 LCD。

電路圖（V_0 的接法同範例 6-2a）



執行結果



程式列印

```
1.  #include <LiquidCrystal.h>
2.  LiquidCrystal lcd1(17,16,15,14,9,8);
3.  LiquidCrystal lcd2(17,7,15,14,9,7);
4.  void setup() {
5.    lcd1.begin(16,2);
6.    lcd1.print("Hello 1");
7.    lcd2.begin(16,2);
8.    lcd2.print("Hello 2");
9.  }
10. void loop() {
11.   lcd2.setCursor(0,1);
12.   lcd2.print(millis()/1000);
13. }
```

自我練習

1. 請用兩個 LCD，讓兩個人可玩猜拳遊戲。
2. 叫號器。假設某一櫃臺有 2 個辦事員，那就有 2 個按鈕，且要有 2 個 LCD 顯示燈號。來賓也要一個按鈕，按一下得到 1 個號碼，本例來賓沒有印表機，可用第 3 個 LCD 顯示其號碼。

6-3 聲音控制

▶ 表 6-10　本節材料表

編號	零件名稱	數量	備註
1	蜂鳴器	1	
2	按壓開關	8	

蜂鳴器

只要震動周圍的空氣就可以產生聲音，所以蜜蜂或蚊子飛來飛去震動空氣就有聲音；喇叭或蜂鳴器如圖 6-13 就是應用此一原理，所完成的一種電氣裝置，只要反覆給 0 與 1 的電壓，就可以讓其薄膜震動，進而可以發出聲音，喇叭材質較好，聲音品質較好，但是較貴，本書就拿價錢便宜的蜂鳴器作實驗就好，如右圖所示（U12 就是其輸入，Gnd 已內部連接）。但是購買時，請特別留意，因爲蜂鳴器有兩種，一種是自激式，另一種是它激式。自激式是只要給電壓，那就會發出內建的頻率聲響；它激式則是要給震動的頻率，才會發出不同頻率的聲音，本節實驗請購買它激式蜂鳴器。

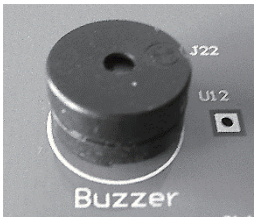


圖 6-13　蜂鳴器實體圖

頻率表

震盪頻率的高低就是音高，表 6-11 是樂譜各個音階的頻率表。

▶ 表 6-11　電子琴鍵盤音階頻率表

	n	1	2	3	4	5	6	7	8	9	10	11	12
音階	音符	(Do)	(Do#)	(Re)	(Re#)	(Mi)	(Fa)	(Fa#)	(So)	(So#)	(La)	(La#)	(Si)
低音	頻率 (Hz)	262	277	294	311	330	349	370	392	415	440	466	494
中音	頻率 (Hz)	523	554	587	622	659	698	740	784	831	880	932	988
高音	頻率 (Hz)	1046	1109	1175	1245	1318	1397	1480	1568	1661	1760	1865	1976

發聲

由上表可知，您只要能讓薄膜震盪 523Hz 的頻率，其發出聲音的音高就是中音的 Do，Arduino 有一函式 Tone()，就可震盪出所要的頻率，其語法如下：

```
tone(pin, frequency)
```

例如，以下敘述可發出『Do』的音，且持續 0.5 秒。

```
tone(pin,523);
delay(500);//要給時間讓蜂鳴器震盪此頻率
noTone();//關閉震盪
```

範例 6-3a

示範發出 Do、Re、Mi、Fa、So、La、Si、Do…等八個音。

操作步驟

將蜂鳴器標有正號的腳接到 Arduino 的任一數位接腳，本例接到腳位 4，另一隻腳接到 Gnd。

程式列印

```
1.  //4接到蜂鳴器
2.  const byte sp=4;
3.  void setup() {
4.      pinMode(sp,OUTPUT);
5.  }
6.  void loop() {
7.        tone(sp,523);//中音Do ,直到noTone或下一個tone()
8.        delay(500);
9.        noTone(sp);
10.       tone(sp,587);//Re
11.       delay(500);
12.       tone(sp,659);//Mi
13.       delay(500);
14.       tone(sp,698);//Fa
15.       delay(500);
```

```
16.       tone(sp,784);//So
17.       delay(500);
18.       tone(sp,880);//La
19.       delay(500);
20.       tone(sp,988);//Si
21.       delay(500);
22.       tone(sp,1046);//高音Do
23.       delay(500);
24.       noTone(sp);
25.       delay(1000);
26. }
```

自我練習

1. 爲讓大家感受快速震動薄膜就能發音，請鍵入以下程式，感受是不是Do的音。提示：Hz就是每秒震動的次數，所以週期T=1000000us/523=1912us, 半週期就是956us。

```
//4接蜂鳴器
const byte sp=4;
void setup() {
    pinMode(sp,OUTPUT);
}
void loop() {
    digitalWrite(sp,HIGH);
    delayMicroseconds(956);
    digitalWrite(sp,LOW);
    delayMicroseconds(956);
}
```

2. 請將光敏電阻放在門口，當有人進來時，蜂鳴器即發出聲音。

演奏音樂

要讓微處理機演奏音樂，請先下載一個簡譜，如圖6-14（下圖摘自http://blog.roodo.com/midiland/archives/1679070.html，在此致謝）。

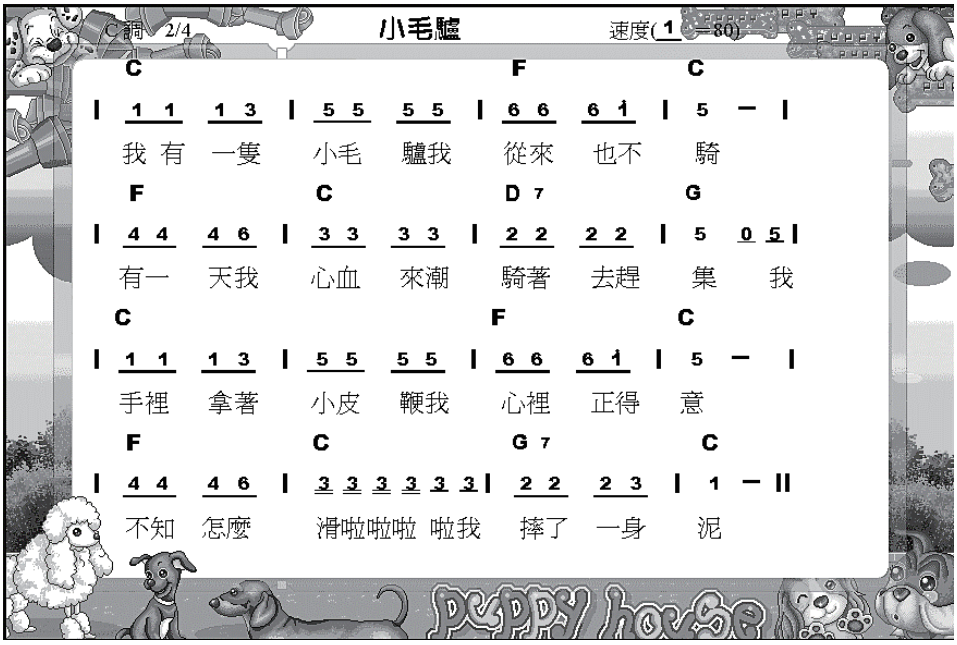


圖 6-14　小毛驢簡譜

本例一分鐘 80 拍，所以 1 拍的時間是 60000ms/80，以一個四分音符爲 1 拍，所以上圖一個八分音符所佔的時間是，60000ms/(2*80)，每小節是 2 拍。其次，上圖，所有音符的最小速度是八分音符，所以我就取八分音符的時間爲 1 個單位，且以陣列兩兩一組儲存每一音符的音高與時間如下，以下僅作 4 小節，『我有一隻小毛驢我從來也不騎』)：第 1 音『1 我』就數位化爲 1,1；最後 1 個『5- 騎』是 2 拍，就數位化爲 5,4。

```
const byte a[]={1,1,1,1,1,1,3,1,5,1,5,1,5,1,5,1,6,1,6,1,6,1,
8,1,5,4};//{音高,時間, 音高,時間,…}
```

然後每一個音高就剩查表了，如以下陣列。

```
const int b[]={0,523,587,659,698,784,880,988,1046};
```

拍數就是所要延遲的時間，如以下程式的 delay()。

範例 6-3b

本例演奏以上小毛驢音樂。

程式列印

```
1.  //4接蜂鳴器
2.  //小毛驢
3.  const byte sp=4;
4.  const int b[]={0,523,587,659,698,784,880,988,1046};
5.  const byte s=80;
6.  const byte a[]={1,1,1,1,1,1,3,1,5,1,5,1,5,1,5,1,6,1,6,1,6,1,8,1,
                    5,4};
7.  const byte l=26;
8.  //蝴蝶
9.  //const byte a[]={1,2,1,2,2,1,3,2,3,2,2,1,1,1,2,1,3,1,1,2,3,2,3,
                     1,4,1,5,2,5,2,4,1,3,1,4,1,5,1,3,2};
10. //const byte l=40;
11. void setup() {
12.     pinMode(sp,OUTPUT);
13. }
14. void loop() {
15.     for (int i=0;i<l/2;i=i+1){
16.       tone(sp,b[a[2*i]]); //音高
17.       delay(a[2*i+1]*60000/(2*s));//拍數，也就是延遲時間
18.       noTone(sp);
19.     }
20.     noTone(sp);
21.     delay(1000);
22. }
```

自我練習

1. 範例 6-3b 僅完成四分之一，請繼續完成未完成的演奏。
2. 請自行找一張簡譜，並製作一個音樂檔。
3. 同範例 6-3b，但還可使用 LED 顯示每一小節的音符。
4. 請輸入 8 個簡譜，並可用指撥開關指定演奏哪一曲。
5. 同範例 6-3b，可用一個按壓開關，選擇演奏哪一曲。

警車聲音

警車的警報聲是連續發出頻率 265Hz 與 350Hz 所形成，以下範例示範發出此聲音。

範例 6-3c

示範發出警車警報聲。警車鳴笛聲是反覆發出頻率 f=265Hz 時間 t=0.3 秒，然後發出頻率 f=350Hz 時間 t=0.7 秒所形成，請寫程式模擬此聲音。

程式列印

```
1.  //4接蜂鳴器
2.  const byte sp=4;
3.  void setup() {
4.      pinMode(sp,OUTPUT);
5.  }
6.  void loop() {
7.      for (int i=0;i<10;i++){
8.          tone(sp,265);
9.          delay(300);
10.         tone(sp,350);
11.         delay(700);
12.         noTone(sp);
13.         delay(50);
14.
15.     }
16. }
```

自我練習

1. 市內電話來電大都採用 1000Hz 與 500Hz 交錯形成的樂音，請寫程式完成。
2. 同範例 6-3c，練習讓鈴聲可越來越急。
3. 音感練習。請觀察您家微波爐、洗衣機、汽車安全帶未繫、倒車警告、手機的各種聲響，並寫程式完成。

電子琴

使用 8 個按壓開關當作電子琴的鍵盤，隨時監控這八個鍵，每當按下任一鍵時，就震盪該鍵的頻率，請看以下範例。

範例 6-3d

示範電子琴的製作。

操作步驟

完成圖 6-15 接線。

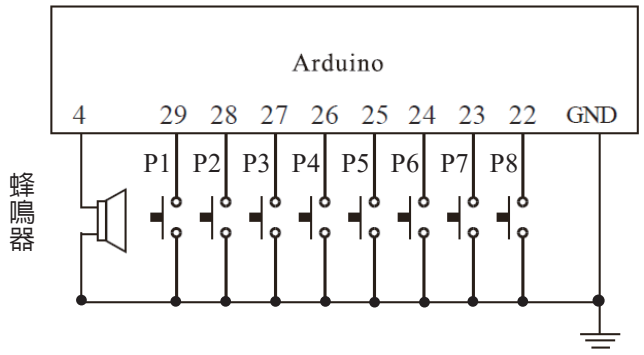


圖 6-15　電子琴電路圖

程式列印

1. 使用 switch，程式如下：

```
1.  //4接蜂鳴器
2.  //29~22接按壓開關
3.  const byte sp=4;
4.  const byte b[]={29,28,27,26,25,24,23,22};//PA
5.  void setup() {
6.      pinMode(sp,OUTPUT);
7.      for (int i=0;i<8;i++)
8.          pinMode(b[i],INPUT_PULLUP);
9.  }
10. void loop() {
11.     byte a=PINA;//一次讀取8位元
```

```
12.     switch (a){
13.         case B01111111:
14.             tone(sp,523); delay(300); noTone(sp); break;
15.         case B10111111:
16.             tone(sp,587); delay(300); noTone(sp); break;
17.         case B11011111:
18.             tone(sp,659); delay(300); noTone(sp); break;
19.         case B11101111:
20.             tone(sp,698); delay(300); noTone(sp); break;
21.         case B11110111:
22.             tone(sp,784); delay(300);noTone(sp); break;
23.         case B11111011:
24.             tone(sp,880); delay(300); noTone(sp); break;
25.         case B11111101:
26.             tone(sp,988); delay(300); noTone(sp); break;
27.          case B11111110:
28.             tone(sp,1046);delay(300); noTone(sp); break;
29.     }
30. }
```

自我練習

1. 範例 6-3d 使用 switch case ，請使用陣列查表改寫本程式。
2. 請用 8 顆 LED，可以顯示所按到的對應鍵。
3. 請用 13 個按壓開關，完成有升降鍵的中音鍵盤。

※ 範例 6-3e

電子琴教學機的製作

卡拉 OK 伴唱機的發明，讓每個人都能高歌歡唱，這學期老師教我們用 Arduino 製作電子琴與音樂的播放，也教我們製作霹靂燈，老師說電子琴教學機的作法其實和霹靂燈相近，就是要彈哪裡，就亮哪裡，所以本範例研究如何製作一個電子琴教學機。

硬體電路

1. 電路接法比照範例 5-1a 的霹靂燈，先將 Arduino 的 A0、A1…A15 接腳接限流電阻，再接到 LED，此 16 個 LED 分別放在低音 Mi, 低音 Fa, 低音 So, 低音 La, 低音 Si , 中音 Do , Re, Mi, Fa, So, La, Si, 高音 Do, 高音 Mi, 高音 Fa, 高音 So 等琴鍵的上方，如圖 6-16：

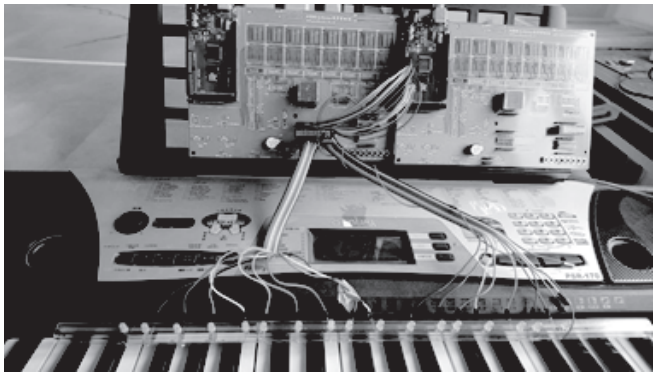


圖 6-16　電子琴教學機電路實體圖

資料的數位化

1. 每首音樂都由音高與拍數組成。
2. 音高的數位化。本例將 Do, Re, Mi, Fa, So, La, Si，分別以 1, 2, 3, 4, 5, 6, 7 表示，低音 Mi 則是 -4, 也就是 PORTF=0x01; PORTK=0x00; ，低音 Fa 是 -3，低音 So 是 -2，如表 6-12：其餘依此類推：（因爲陣列是從 0 開始，所以先將這些值加 4。）

▶ 表 6-12　電子琴燈號與位置對照表

	Mi	Fa	So	La	Si	Do	La	Mi	Fa	So	La	Si	Do	La	Mi	Fa
值	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11
配合陣列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PORTF(0x)	01	02	04	08	10	20	40	80	0	0	0	0	0	0	0	0
PORTK(0x)	0	0	0	0	0	0	0	0	01	02	04	08	10	20	40	80

3. 例如：按 A0 鍵，表示要亮最左邊低音 Mi，所以是 PORTK=B0;PORTF=B00000001。

4. 拍數。本例將八分音符（半拍，簡譜下面兩條線，如圖 6-17 的『6』）定義爲 1，四分音符（5，1 拍，簡譜下面 1 條線）就定義爲 2，二分音符（5-）2 拍，就定義爲 4，全音音符（5- - -）2 拍就定義爲 8。

圖 6-17　兩隻老虎簡譜

5. 圖 6-16 是兩隻老虎的簡譜。本例將音高、拍數，每兩個 1 組，例如「1,4」,「2,4」爲 1 組，以 a[] 陣列儲存如下：

```
const int
a[]={1,4,2,4,3,4,1,4,1,4,2,4,3,4,1,4,3,4,4,4,5,8,3,4,4,4,5,
8,5,2,6,1,5,2,4,1,3,4,1,4,5,2,6,1,5,2,4,1,3,4,1,4,1,4,-
2,4,1,8,1,4,-2,4,1,8};
```

6. 依照霹靂燈的原理，只是將以上音符以時序圖的方式輸出，所以 Arduino 程式如下：

```
1.  //小毛驢
2.  //const byte a[]={1,2,1,2,2,1,3,2,3,2,2,1,1,1,2,1,3,1,1,2,3,
                      2,3,1,4,1,5,2,5,2,4,1,3,1,4,1,5,1,3,2};
3.  //const byte l=40;
4.  //蝴蝶
5.  a[]={1,4,2,4,3,4,1,4,1,4,2,4,3,4,1,4,3,4,4,4,5,8,3,4,4,4,5,
         8,5,4,6,2,5,4,4,2,3,4,1,8,5,4,6,2,5,4,4,2,3,4,1,8,1,4,-
         2,4,1,8,1,4,-2,4,1,8};
6.  //兩之老虎
7.  const int
8.  a[]={1,4,2,4,3,4,1,4,1,4,2,4,3,4,1,4,3,4,4,4,5,8,3,4,4,4,5,
         8,5,2,6,1,5,2,4,1,3,4,1,4,5,2,6,1,5,2,4,1,3,4,1,4,1,4,-
         2,4,1,8,1,4,-2,4,1,8};
9.  const byte l=64;//a陣列長度
10. const byte s=80;
```

```
11. void setup() {
12.   // put your setup code here, to run once:
13.   DDRF=0xFF;  DDRK=0xFF;
14. }
15. void loop() {
16.     PORTF=0xFF;
17.     PORTK=0xFF;
18.     delay (2000);
19.      for (int i=0;i<l/2;i=i+1){
20.       //tone(sp,b[a[2*i]]);
21.       PORTF=f[a[2*i]+4];
22.       PORTK=k[a[2*i]+4];
23.       delay(a[2*i+1]*60000/(s*2));
24.       PORTF=0;
25.       PORTK=0;
26.       delay(40);
27.       //noTone(sp);
28.     }
29. }
```

專題成果

1. 如以下影片，這樣眞的很好用，可作爲電子琴、各種樂器的教學，反正亮哪裡，就按哪裡（延遲時間還很精準），成本也很低，可作爲長者、幼兒、特殊教育教學用。https://youtu.be/CsvtgIQ7Vx0
2. 以上音高、延遲時間，本例用一維陣列儲存，請讀者用二維陣列儲存，改寫程式，並比較兩者的便利性。

自我練習

※1. 範例 6-3e 電子琴是現成的，僅拿單晶的 LED 當作按鍵的指示。請擴充此範例，自行用 8 個按鈕製作一個電子琴，再用 8 個 LED 當作按鍵指示，完成範例 6-3e 的效果。

※2. 貴婦電子琴。現在汽車已經可以自駕，請製作一個電子琴，預先儲存一首歌，使用者只要按鍵就好，電子琴都可以發正確的音。也就是使用者的按鍵只是節奏，電腦都發正確的音。這樣就可讓貴婦或街頭藝人盡情表演，反正電腦都可以發出正確的音。

6-4 鍵盤控制

▶ 表 6-13　本節材料表

編號	零件名稱	數量	備註
1	4 * 4 按壓鍵盤	1	

前面的指撥開關或按壓開關，都是每條資料線對應一個開關，那如果是一個 100 鍵的鍵盤，是否需要 100 條資料線，答案當然是否定的，因爲一個 Arduino Maga 2560 單晶片也至多 70 個 I/O 接點。若要提高外部開關的數量，通常可以使用矩陣排列的鍵盤輸入，然後再運用快速掃描原理，達到模擬多個按鍵的效果。例如，圖 6-18a 是本書實驗板的標準 4*4 鍵盤輸入模組，只要使用 4 條資料線，4 條狀態線，即可排列成 16 個按鍵，如下圖中。其次，前面已經一再介紹掃瞄的原理，矩陣鍵盤亦要使用掃瞄的觀念，逐一輪流輸出低電壓於 R_1、R_2、R_3、R_4，再逐一檢查 C_1、C_2、C_3、C_4 等是否得到低電位，即可判斷那一個按鍵被『按』。圖 6-18b 是其內部接線圖。

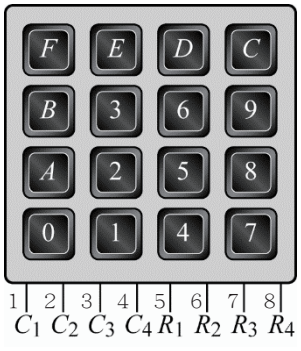


圖 6-18a　4*4 鍵盤實體圖

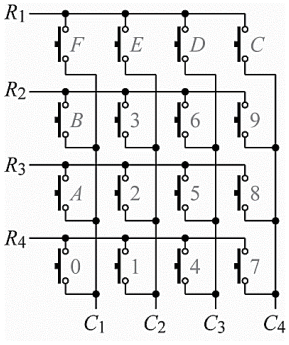


圖 6-18b　4*4 鍵盤內部電路圖

例如，使用三用電表的歐姆檔，測試棒接 R_1,C_1，只有在『F』按鈕被按時，此兩點相通；又例如測試棒接 R_1,C_4，只有在『C』按鈕被按時，此兩點相通。其次，4*4 鍵盤也有一種薄膜按鍵模組，如圖 6-19，但腳位與上面左圖不同，由左到右分別是 R_1,R_2,R_3,R_4,C_1,C_2,C_3,C_4。

圖 6-19　薄膜按鍵實體圖

四位元按壓開關

若只需要一個四位元按壓開關，亦可將以上電路的 R_1 直接接地，然後判斷 C_1~C_4 何者爲低電位，即可偵測 F、E、D、C 的哪一鍵被按。例如，若 C_1 爲低電位，表示 F 鍵被按，若 C_2 爲低電位，表示 E 被按。這就留給讀者自行練習。

4＊4按壓開關的掃瞄

前面已經一再介紹掃瞄的原理，矩陣鍵盤亦要使用掃瞄的觀念，逐一輸出低電壓於 R_1、R_2、R_3、R_4，再逐一檢查 C_1、C_2、C_3、C_4 等是否得到低電位，即可判斷那一個按鍵被按。例如：R_1~R_4 送出 B0111 時（表示 R_1 爲低電壓，如表 6-14），逐一偵測 C_1~C_4，當偵測 C_1 得到低電位時 (B0111)，表示按鍵 F 被按；當偵測 C_2 得到低電位時 (B1011)，表示按鍵 E 被按；當偵測 C_3 得到低電位時 (B1101)，表示按鍵 D 被按；當偵測 C_4 得到低電位時 (B1110)，表示按鍵 C 被按。

▶ 表 6-14　4*4 傳回值與按鍵位置對照表一

PORTA / PORTC值	C_1 0111	C_2 1011	C_3 1101	C_4 1110
R1 低電壓 B0111	F	E	D	C

接著，R_1~R_4 送出 (B1110)，表示按鍵 9 被按，如表 6-15。也是逐一偵測 C_1~C_4，當傳回值爲 (B0111)，表示按鍵 B 被按；當傳回值爲 (B1011)，表示按鍵 3 被按；當傳回值爲 (B1101)，表示按鍵 6 被按；當傳回值爲 (B1110)，表示按鍵 9 被按，如表 6-15。

▶ 表 6-15　4*4 傳回值與按鍵位置對照表二

PORTA / PORTC值	C_1 0111	C_2 1011	C_3 1101	C_4 1110
R2 低電壓 B1011	B	3	6	9

其餘 R_1~R_4 送出 B1101 與 B1110，也是相同的道理。

範例 6-4a

示範以上 4*4 鍵盤。以下我們將以上演算，寫程式示範以上演算，且於序列埠視窗輸出鍵盤按鍵值。

電路圖

請將以上鍵盤模組接線如圖 6-20：

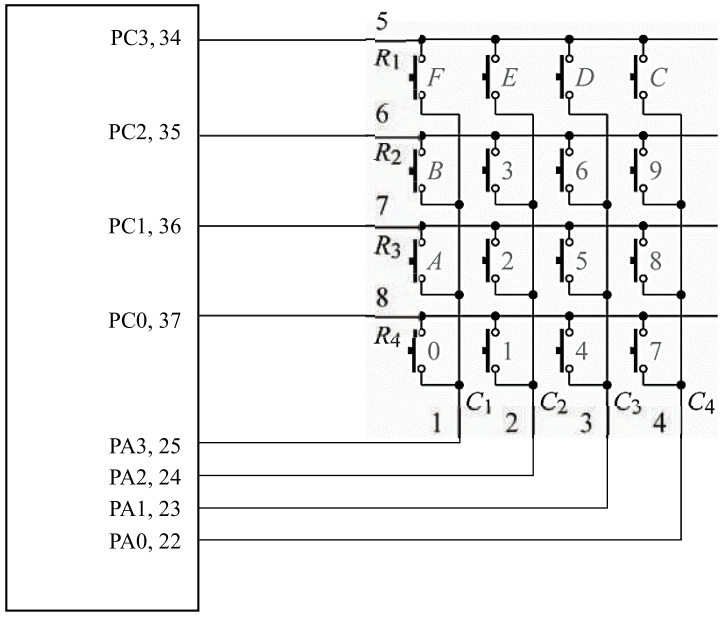
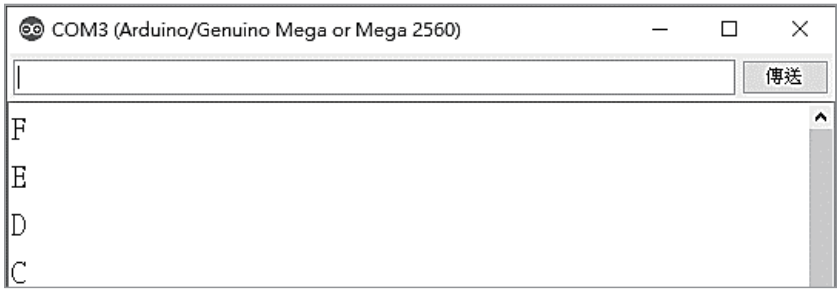


圖 6-20　4*4 鍵盤驅動電路圖

執行結果



程式列印

1 .以上有點複雜，本例先不要用迴圈，而是採用基本運算，一步步寫出程式。其次，爲了不浪費版面，以下僅讓 R1~R2 輪流得到低電壓，R3~R4 請自行擴充。

```
1.  //37~36接R4~R1
2.  //22~25 接C4~C1
3.  const byte rowp[]={37,36,35,34};//R4,R3,R2,R1
4.  const byte colp[]={22,23,24,25};//C4,C3,C2,C1
5.  void setup() {
6.      Serial.begin(9600);
7.      for(int i=0 ;i<=3;i++){
8.          pinMode(rowp[i],OUTPUT);
9.          pinMode(colp[i],INPUT_PULLUP);
10.   }
11. }
12. void loop() {
13.     byte b;
14.     PORTC=B0111;//表示R1低電位
15.     delay(50);
16.     b=PINA & 0x0f;//僅使用低四位元，將高四位元去掉
17.     if (b!=0xf){//判斷是否有按鍵
18.         while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
19.               delay(1);
20.         switch (b){
21.             case 0x7://0111 偵測行1
22.                 Serial.println("F");
23.                 break;
24.             case 0xb://1011偵測行2
25.                 Serial.println("E");
26.                 break;
27.             case 0xd://1101偵測行3
28.                 Serial.println("D");
29.                 break;
30.             case 0xe://1110偵測行4
31.                 Serial.println("C");
32.                 break;
33.           }
34.           delay(100);
35.       }
```

```
36.     PORTC=B1011; //表示R2低電位
37.     delay(50);
38.     b=PINA & 0x0f;//僅使用低四位元，將高四位元去掉
39.     if (b!=0xf){//判斷是否有按鍵
40.         while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
41.               delay(1);
42.         switch (b){
43.             case 0x7://0111 偵測行1
44.                 Serial.println("B");
45.                 break;
46.             case 0xb://1011偵測行2
47.                 Serial.println("3");
48.                 break;
49.             case 0xd://1101偵測行3
50.                 Serial.println("6");
51.                 break;
52.             case 0xe://1110偵測行4
53.                 Serial.println("9");
54.                 break;
55.           }
56.           delay(100);
57.       }
58. }
```

2. 由以上程式，找出規律性，將 4 個輸出以 row[] 陣列儲存，16 個按鍵值以 a[4][4] 字元陣列儲存，a[0][0]= 'F', a[0][1]= 'E',…這樣就可以使用迴圈簡化程式，程式如下：

```
1.  //37~36接R4~R1
2.  //22~25 接C4~C1
3.  const byte rowp[]={37,36,35,34};//R4,R3,R2,R1
4.  const byte colp[]={22,23,24,25};//C4,C3,C2,C1
5.  const byte row[]={B0111,B1011,B1101,B1110};
                                        //R1,R2,R3,R4輪流得到低電位
6.  const char a[4][4]={'F','E','D','C',
7.                 'B','3','6','9',
8.                 'A','2','5','8',
9.                 '0','1','4','7'};
                            //字元陣列a[0][0]= 'F', a[0][1]= 'E',…
10. void setup() {
11.     Serial.begin(9600);
```

```
12.     for(int i=0 ;i<=3;i++){
13.         pinMode(rowp[i],OUTPUT);
14.         pinMode(colp[i],INPUT_PULLUP);
15.   }
16. }
17. void loop() {
18.     byte b;
19.     for (int i=0 ;i<=3;i++)     {
20.         PORTC=row[i];
21.         delay(50);
22.         b=PINA & 0x0f;//僅使用低四位元，將高四位元去掉
23.         if (b!=0xf){//判斷是否有按鍵
24.             while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
25.                 delay(1);
26.             switch (b){
27.                 case 0x7://0111 偵測行1
28.                     Serial.println(a[i][0]);
29.                     break;
30.                 case 0xb://1011偵測行2
31.                     Serial.println(a[i][1]);
32.                     break;
33.                 case 0xd://1101偵測行3
34.                     Serial.println(a[i][2]);
35.                     break;
36.                 case 0xe://1110偵測行4
37.                     Serial.println(a[i][3]);
38.                   break;
39.           }
40.           delay(100);
41.       }
42.   }
43. }
```

自我練習

1. 寫一程式，可將 4*4 按鍵當作 4 個按壓開關使用，功能如下：

按鍵	功能
F	數值加一
E	數值減一
D	數值加十
C	數值減十

輸出請分別使用四位數七段顯示器與 LCD 輸出。

2. 寫一程式，可使用 4*4 按鍵輸入 0 到 9，輸出請分別於一位數七段顯示器與 LCD 顯示其值。

3. 寫一程式，可使用 4*4 按鍵連續輸入 0 到 9，且最多 4 個數字，並分別於四位數七段顯示器，由左到右顯示其值。（本例限制至多可輸入 4 個數字）

4. 同上題，可使用另一按鍵，當按此鍵時，開始倒數計時。

5. 寫一程式，可使用 4*4 按鍵輸入 0 到 9，並於四位數七段顯示器，由左到右顯示其值，當超過 4 個位數時，請左移一個數字。

6. 寫一程式，滿足以下功能。

   (1) 使用 16 個按鍵控制 8 個 LED。
   (2) 每一個 LED 由兩個按鍵控制其明或滅。

7. 寫一程式，滿足以下功能。

   (1) 使用 8 個按鍵控制 8 個 LED。
   (2) 每一個 LED 由一個按鍵控制其亮或滅，第一次按鍵使其『亮』，第二次按鍵使其『滅』。

範例 6-4b

將以上範例的鍵盤輸入部分改爲函式，按「1」亮第 1 個 LED，按「2」亮第 2 個 LED，按「3」亮第 3 個 LED。

程式列印

以下程式將鍵盤輸入部分獨立出來，寫成 input() 函式，此函式傳回按鍵值，這樣才能簡化主程式。

```
1.  //37~36接R4~R1
2.  //22~25 接C4~C1
3.  const byte rowp[]={37,36,35,34};
4.  const byte colp[]={22,23,24,25};
5.  const byte row[]={B0111,B1011,B1101,B1110};
6.  const char a[4][4]={'F','E','D','C',
7.                      'B','3','6','9',
8.                      'A','2','5','8',
9.                      '0','1','4','7'};
10. void setup() {
11.     Serial.begin(9600);
12.     for(int i=0 ;i<=3;i++){
13.         pinMode(rowp[i],OUTPUT);
14.         pinMode(colp[i],INPUT_PULLUP);
15.     }
16.     DDRF=0xff;
17. }
18. void loop() {
19.    char f=input();//偵測鍵盤輸入
20.    Serial.println(f);//於序列埠視窗輸出按鍵值
21.    if (f=='1'){
22.       digitalWrite(A0,HIGH);
23.       delay(1000);
24.       digitalWrite(A0,LOW);
25.    }
26.    else if(f=='2'){
27.       digitalWrite(A1,HIGH);
28.       delay(1000);
29.       digitalWrite(A1,LOW);
30.    }
31.    else if(f=='3'){
```

```
32.       digitalWrite(A2,HIGH);
33.       delay(1000);
34.       digitalWrite(A2,LOW);
35.    }
36. }
37. //鍵盤輸入函式
38. char input(){
39.     byte b;
40.     char d=' ';
41.     for (int i=0 ;i<=3;i++)    {
42.         PORTC=row[i];
43.         delay(50);
44.         b=PINA & 0x0f;
45.         if (b!=0xf){
46.             while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
47.               delay(1);
48.             switch (b){
49.                 case 0x7://0111
50.                     d=a[i][0];
51.                     break;
52.                 case 0xb:
53.                     d=a[i][1];
54.                     break;
55.                 case 0xd:
56.                     d=a[i][2];
57.                     break;
58.                 case 0xe:
59.                     d=a[i][3];
60.                   break;
61.           }
62.           delay(100);
63.       }
64.   }
65.   return(d);
66. }
```

範例 6-4c

寫一程式，可使用 4*4 按鍵，完成簡單計算機功能，按鍵定義如下：只有加、減、乘、除、求餘數等運算，輸出則使用序列埠監控視窗。本例按鍵定義如下表：

+	-	*	/
%	3	6	9
=	2	5	8
0	1	4	7

例如，輸入：(數字長度不限)

```
1000+2=
```

則於序列埠監控視窗輸出

```
1000+2=1002
```

程式列印

```
1.  //37~36接R4~R1
2.  //22~25 接C4~C1
3.  const byte rowp[]={37,36,35,34};
4.  const byte colp[]={22,23,24,25};
5.  const byte row[]={B0111,B1011,B1101,B1110};//依序輸出值
6.  const char a[4][4]={'+','-','*','/',
7.                      '%','3','6','9',
8.                      '=','2','5','8',
9.                      '0','1','4','7'};//字元陣列
10. void setup() {
11.   Serial.begin(9600);
12.   for(int i=0 ;i<=3;i++){
13.     pinMode(rowp[i],OUTPUT);//R1~R4爲輸出
14.     pinMode(colp[i],INPUT_PULLUP);//C1~C4爲上拉電阻輸入
15.   }
16. }
17. int readop(char d[20]){
18.     byte b;
```

```
19.     char d1;
20.     int j=0;
21.     do {
22.         for (int i=0 ;i<=3;i++)    {
23.             PORTC=row[i];
24.             delay(50);
25.             b=PINA & 0x0f;
26.             if (b!=0xf){//不等於空白，表示有按鍵
27.                 while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
28.                   delay(1);
29.                 switch (b){
30.                     case 0x7:
31.                         d1=a[i][0];//傳回按鍵值
32.                         Serial.print(d1);
33.                         break;
34.                     case 0xb:
35.                         d1=a[i][1];
36.                         Serial.print(d1);
37.                         break;
38.                     case 0xd:
39.                         d1=a[i][2];
40.                         Serial.print(a[i][2]);
41.                         break;
42.                     case 0xe:
43.                         d1=a[i][3];
44.                         Serial.print(a[i][3]);
45.                       break;
46.                 }
47.                 delay(100);
48.                 d[j]=d1;
49.                 j++;
50.             }
51.       }
52.     }while (d1!='=');//直到「=」符號時才結束
53.     return (j); //length
54. }
55. void loop() {
56.     char d[20];//String
57.     int len;//length
58.     char d1;
```

```
59.     int op1,op2,r;//op1 是第1個運算元，op2 是第2個運算元
60.     char op;//運算子
61.     int j;
62.     len=readop(d);
63.     if (len!=0){
64.       Serial.println();
65.       for (int i=0;i<=len;i++)
66.           Serial.print(d[i]);//輸出按鍵內容
67.       Serial.println();
68.       op1=byte(d[0]-48); //因爲取到ASCII，所以要減48，才能得到0,1,2
69.       j=1;
70.       while( isDigit(d[j])){//若繼續是數值,op1=前面位數*10+目前位數
71.         op1=op1*10+byte(d[j]-48);
72.         j++;
73.       }
74.       op=d[j];//運算子
75.       j++;
76.       op2=byte(d[j]-48);//因爲取到ASCII，所以要減48，才能得到0,1,2
77.       j++;
78.       while (isDigit(d[j])){ //若繼續是數值,op2=前面位數*10+目前位數
79.         op2=op2*10+byte(d[j]-48);
80.         j++;
81.       }
82.       switch(op){//依運算子種類，執行相對運算
83.         case ('+'):
84.             r=op1+op2;
85.             break;
86.         case ('-'):
87.             r=op1-op2;
88.             break;
89.         case ('*'):
90.             r=op1*op2;
91.             break;
92.         case ('/'):
93.             r=op1/op2;
94.             break;
95.         case ('%'):
96.             r=op1%op2;
97.             break;
98.       }
```

```
99.       Serial.print(op1);
100.       Serial.print(op);
101.       Serial.print(op2);
102.       Serial.print('=');
103.       Serial.println(r);
104.       Serial.println();
105.     }
106. }
```

自我練習

1. 請練習由 LCD 輸出結果，那就是一台計算器了。
2. 請套用上一範例，將鍵盤輸入改爲函式。

※ 3. 連續運算。同上範例但可完成連續運算。例如，輸入 120+20/2= 可輸出 70，運算一律僅考慮由左而右，遇到等號才結束。

※ 4. 若要能先乘除後加減，請參考資料結構的「四則運算」。

6-5 密碼鎖

▶ 表 6-16　本節材料表

編號	零件名稱	數量	備註
1	4*4 按壓開關	1	
2	LED	8	
3	限流電阻 220Ω ～ 470Ω	8	
4	繼電器模組	1	若無，可用 LED 模擬
5	小型低壓電磁閥	1	若無，可用 LED 模擬

常見的密碼鎖如圖 6-21：（摘自飛利浦智能鎖 https://www.ljtwm.com/）

圖 6-21　密碼鎖實體圖

密碼鎖硬體主要含鍵盤輸入、LED 指示燈、電磁閥。當密碼正確，可關閉電磁閥，門就可以打開，以下我們逐一由小程式研究密碼鎖的製作。

範例 6-5a

假設密碼是「0147」，當使用者數入此密碼，腳位編號 A3 的 LED 亮。

1. 電路同範例 6-4a 且於 A3 腳位接一個 LED。
2. 程式設計
   (1) 鍵盤輸入程式同範例 6-4b，但新增不管按鍵按多久，僅能算 1 次。
   (2) 當傳回不是空字元，表示鍵盤有被按，才累加此字元。

(3) 新增密碼比對程式如下：

```
1.  //A3接LED
2.  //37~36接R4~R1
3.  //22~25 接C4~C1
4.  const byte rowp[]={37,36,35,34};
5.  const byte colp[]={22,23,24,25};
6.  const byte row[]={B0111,B1011,B1101,B1110};
7.  const char a[4][4]={'F','E','D','C',
8.                 'B','3','6','9',
9.                 'A','2','5','8',
10.                '0','1','4','7'};
11. void setup() {
12.     Serial.begin(9600);
13.     for(int i=0 ;i<=3;i++){
14.         pinMode(rowp[i],OUTPUT);
15.         pinMode(colp[i],INPUT_PULLUP);
16.     }
17.     pinMode(A3,OUTPUT);
18. }
19. String pa="";
20. void loop() {
21.    char in=input();
22.    Serial.println(in);
23.    if (in!=' '){ //傳回不是空字元，表示鍵盤有被按
24.       pa=pa+in; //累加此字元
25.       Serial.println(pa);
26.       if (pa=="0147"){ //密碼比對
27.         digitalWrite(A3,HIGH);
28.       }
29.    }
30. }
31. char input(){
32.     byte b;
33.     char d=' ';
34.     for (int i=0 ;i<=3;i++)     {
35.         PORTC=row[i];
36.         delay(50);
37.         b=PINA & 0x0f;
38.         if (b!=0xf){
```

```
39.             while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
40.               delay(1);
41.             switch (b){
42.                 case 0x7://0111
43.                     d=a[i][0];
44.                     break;
45.                 case 0xb:
46.                     d=a[i][1];
47.                     break;
48.                 case 0xd:
49.                     d=a[i][2];
50.                     break;
51.                 case 0xe:
52.                     d=a[i][3];
53.                   break;
54.           }
55.           delay(100);
56.       }
57.   }
58.   return(d);
59. }
```

範例 6-5b

續範例 6-5a，但增加 2 個按鈕功能，按「E」鍵表示上鎖，LED 熄滅，按「F」鍵可連續輸入 4 個字元的密碼，當密碼是「0147」，LED 亮起。

操作步驟

1. 電路同範例 6-5a。
2. setup() 與 input() 函式同範例 6-5a。

```
1.  //A3接LED
2.  //37~36接R4~R1
3.  //22~25 接C4~C1
4.  String pass="0147";
5.  String pa="";
6.  int num=0;
7.  void loop() {
```

```
8.      char in=input();
9.      //Serial.println(in);
10.     if (in=='E'){//上鎖
11.          digitalWrite(A3,HIGH);
12.          pa="";
13.     }
14.     if (in=='F'){//開門
15.         digitalWrite(A0,HIGH);
16.         pa="";
17.         num=1;//進入輸入密碼
18.         while (num<=4){ //連續輸入4個字元
19.           in=input();
20.           if (in!=' '){
21.             pa=pa+in; //累加輸入的字元
22.             num=num+1;
23.             //Serial.println(pa);
24.           }
25.         }
26.         if (pa==pass){//密碼正確，開門
27.           digitalWrite(A3,LOW);
28.           pa="";
29.         }
30.         digitalWrite(A0,LOW);
31.     }
32. }
```

範例 6-5c

續範例 6-5b 加入更換密碼功能。按「D」鍵表示進入密碼更換程序，此時 A0 LED 亮起，請先輸入原密碼，若密碼正確，A1 LED 亮起，接著輸入新密碼兩次，輸入第 1 次密碼（4 個字元）A2 LED 亮起，接著輸入第 2 次新密碼，若兩次新密碼相同，代表密碼更換成功，A2 LED 亮滅 3 次。

操作步驟

1. 電路圖如圖 6-22。
2. 本例密碼鎖操作流程，如圖 6-23。

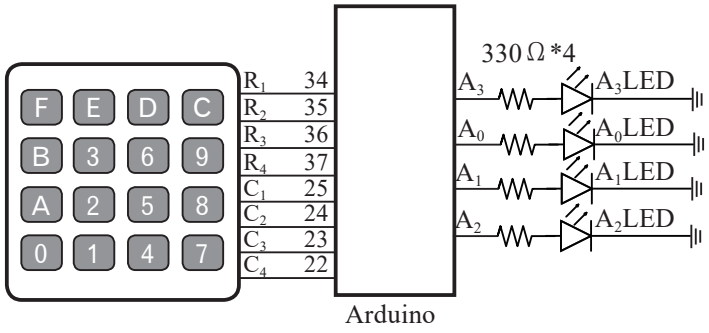


圖 6-22　密碼鎖電路圖一

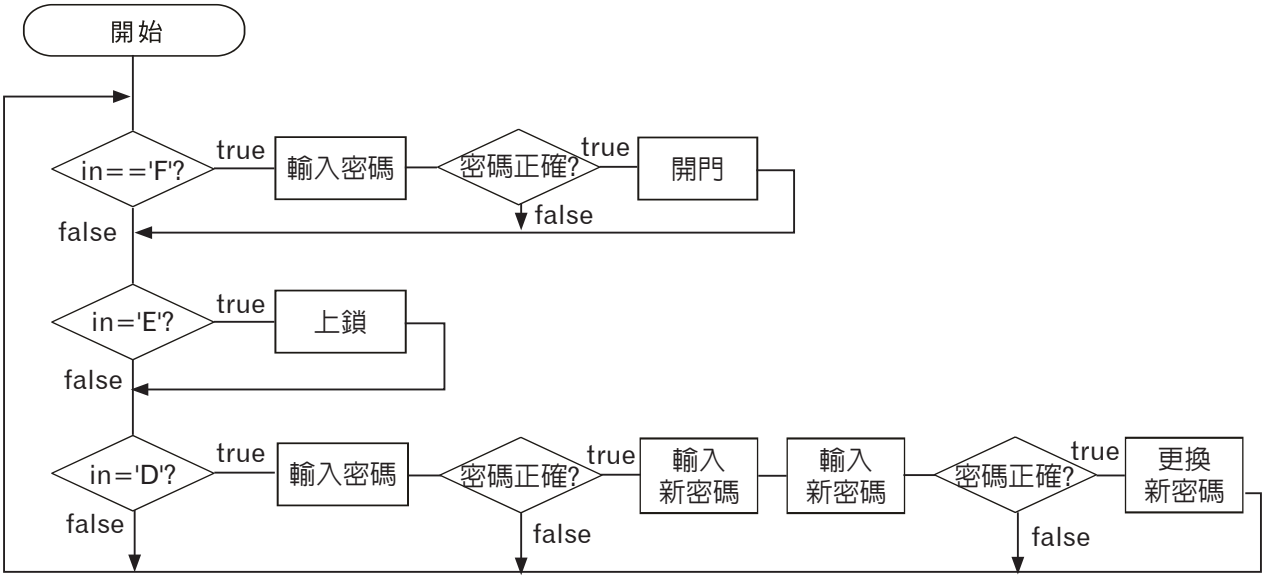


圖 6-23　密碼鎖流程圖

3. 全部程式如下：

```
1.  //A0,A1,A2,A3接LED
2.  //37~36接R4~R1
3.  //22~25 接C4~C1
4.  const byte rowp[]={37,36,35,34};
5.  const byte colp[]={22,23,24,25};
6.  const byte row[]={B0111,B1011,B1101,B1110};//4*4鍵盤輸出
7.  const char a[4][4]={'F','E','D','C',
8.                      'B','3','6','9',
9.                      'A','2','5','8',
10.                     '0','1','4','7'};//字元陣列，4＊4鍵盤所代表的符號
11. char input(){
12.     byte b;
```

```
13.     char d=' ';
14.     for (int i=0 ;i<=3;i++)     {
15.         PORTC=row[i];//R1~R4輪流輸出低電位
16.         delay(50);
17.         b=PINA & 0x0f;//讀取22,23,24,25 PORTA,僅要低四位元
18.         if (b!=0xf){
19.             while((PINA & 0x0f)!=0xf)//不管按多久，僅能算1次
20.               delay(1);
21.             switch (b){
22.                 case 0x7://0111
23.                     d=a[i][0];
24.                     break;
25.                 case 0xb://1011
26.                     d=a[i][1];
27.                     break;
28.                 case 0xd://1101
29.                     d=a[i][2];
30.                     break;
31.                 case 0xe://1110
32.                     d=a[i][3];
33.                   break;
34.           }
35.           delay(100);
36.       }
37.   }
38.   return(d);
39. }
40.
41. void setup() {
42.     Serial.begin(9600);
43.     pinMode(13,OUTPUT);
44.     pinMode(A0,OUTPUT);
45.     pinMode(A1,OUTPUT);
46.     pinMode(A2,OUTPUT);
47.     digitalWrite(13,LOW);
48.     for(int i=0 ;i<=3;i++){
49.         pinMode(rowp[i],OUTPUT);
50.         pinMode(colp[i],INPUT_PULLUP);
51.   }
52. }
53. String pass="0147";
54. String pa="";
55.
56. String patemp1="";
```

```
String patemp2="";
int num=0;
void loop() {
    char in=input();
    //Serial.println(in);
    if (in=='F'){//進入輸入密碼
        digitalWrite(A0,HIGH);
        pa="";
        num=1;
        while (num<=4){//輸入原密碼,四位數
          in=input();
          if (in!=' '){
            pa=pa+in;//字串串接
            num=num+1;//僅四位數
            //Serial.println(pa);
          }
        }
        if (pa==pass){//密碼正確，開門
          digitalWrite(13,LOW);
          pa="";
        }
        digitalWrite(A0,LOW);
    }
    if (in=='E'){//上鎖
         digitalWrite(13,HIGH);
         pa="";
    }

    if (in=='D'){//進入更改密碼
          digitalWrite(A0,HIGH);
          pa="";
          num=1;
          while (num<=4){ //先輸入原密碼
            in=input();
            if (in!=' '){
              pa=pa+in;
              num=num+1;
              //Serial.println(pa);
            }
          }
          if (pa==pass){//密碼正確，進入更改密碼
              digitalWrite(A0,LOW);
              digitalWrite(A1,HIGH);
               num=1;
```

```
101.                patemp1="";
102.                while (num<=4){ //輸入新密碼
103.                  in=input();
104.                  if (in!=' '){
105.                    patemp1=patemp1+in;
106.                    num=num+1;
107.                  }
108.                }
109.                digitalWrite(A1,LOW);
110.                digitalWrite(A2,HIGH);
111.                num=1;
112.                patemp2="";
113.                while (num<=4){ //再一次輸入新密碼
114.                  in=input();
115.                  if (in!=' '){
116.                    patemp2=patemp2+in;
117.                    num=num+1;
118.                  }
119.                }
120.                digitalWrite(A2,LOW);
121.                if(patemp1==patemp2){//若兩次新密碼相同
122.                  pass=patemp1;//更換密碼
123.                  Serial.print("pass=");
124.                  Serial.println(pass);
125.                  digitalWrite(A3,HIGH);
126.                  delay(1000);
127.                  digitalWrite(A3,LOW);
128.                  delay(1000);
129.                  digitalWrite(A3,HIGH);
130.                  delay(1000);
131.                  digitalWrite(A3,LOW);
132.                  delay(1000);
133.                  digitalWrite(A3,HIGH);
134.                  delay(1000);
135.                  digitalWrite(A3,LOW);
136.                }
137.            }
138.      }
139. }
```

範例 6-5d

同範例 6-5c，但加入繼電器與電磁閥。

1. 小型電磁閥門鎖如圖 6-24：（摘自露天生活 https://www.ruten.com.tw/）

海外：迷你電動栓鎖 dc24v/小型櫥櫃鎖/電磁閥電門鎖

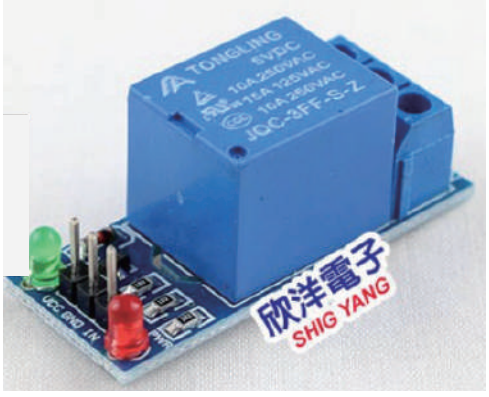
海外：迷你電動栓鎖 dc12v/小型櫥櫃鎖/電磁閥電門鎖

海外：迷你電動栓鎖 dc12v/小型櫥櫃鎖/電磁閥電門鎖

圖 6-24　電磁閥實體圖

以上電磁閥電器規格爲 12V，1.5~3A，Arduino 高電位輸出規格是 5V,20mA，所以要透過繼電器推動。

2. 繼電器模組如圖 6-25：（摘自蝦皮商城 https://shopee.tw/）



產品技術參數

控制信號：TTL電平
額定負載：10A/250VAC，10A/125VAC
　　10A/300DC，10A28VDC
額定通過電流：10A(NO)5A(NC)
最大開關電壓：250VAC，30VDC
VCC：系統電源正極
GND：系統電源負極
IN1~IN8：繼電器控制端口

圖 6-25　繼電器模組實體圖

3. 繼電器模組內部電路示意圖如圖 6-26：是一個含有電流放大與繼電器的模組，所以除了與繼電器相同，有訊號源接腳 IN，也要接 5V 電源。繼電器可看成是一個電磁閥控制的搖頭開關，此搖頭開關的控制來自電磁閥，只要輸入端有 5V,10mA 以上訊號即可推動搖頭開關，而 Arduino 高電位輸出爲 5V,20mA，所以可推動此繼電器模組。平常搖頭開關處於 NC 的位置，當訊號端 IN 爲高電位時，繼電器電磁閥激磁，搖頭開關移到 NO 位置。

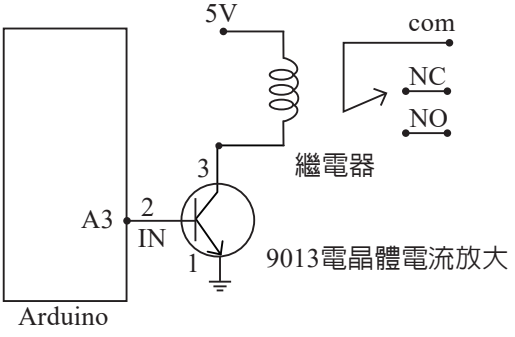


圖 6-26　繼電器模組驅動電路

4. 完成接線，如圖 6-27，即可測試以上關門與開門功能。（下圖 4 個 LED，分別接在 A3,A0,A1,A2 腳位，其名稱分別定義爲 A_3LED，A_0LED,A_1LED,A_2LED）

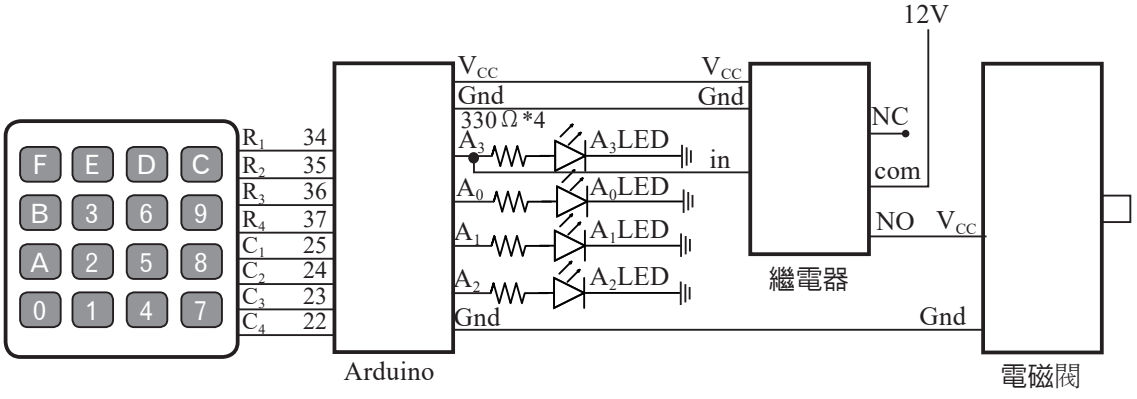


圖 6-27　密碼鎖電路圖二

6-6 步進馬達

▶ 表 6-17　本節材料表

編號	零件名稱	數量	備註
1	步進馬達模組	1~3	
2	步進馬達電流放大模組	1~3	
3	按壓開關	8	
4	極限開關	8	

步進馬達如圖 6-28，本例爲 28BYJ-48（摘自露天商城 https://www.pchomeus.com/item）：

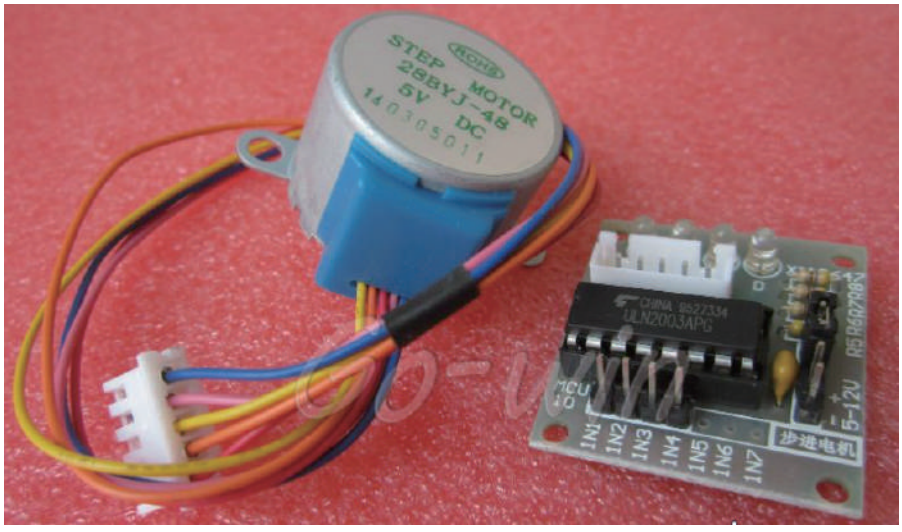


圖 6-28　步進馬達實體圖

步進馬達內部結構示意圖如圖 6-29：中間稱爲轉子，旁邊四個爲定子，兩者以齒輪狀連結，透過定子依序激磁，每激磁 1 次，可以讓轉子轉動齒輪 1 小格，因爲每次都是精準 1 小格，所以不用使用迴授電路檢查位置是否正確，可作爲開迴路精準位移控制器。因爲開迴路就可精準控制位移量，所以電路簡單，成本經濟，用途非常廣泛，例如、磁碟機、印表機、繪圖機、掃描器、雷射雕刻機、洗床 x-y 控制平台等。

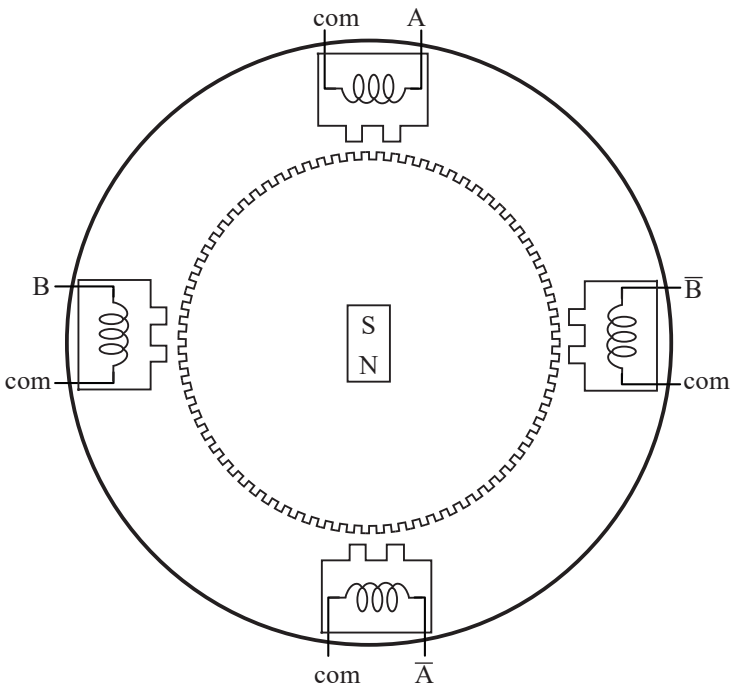


圖 6-29　步進馬達內部結構圖

上圖四個 com 表示彼此互相連通，共用接腳輸出，圖 6-30 是四個轉子內部接線圖：

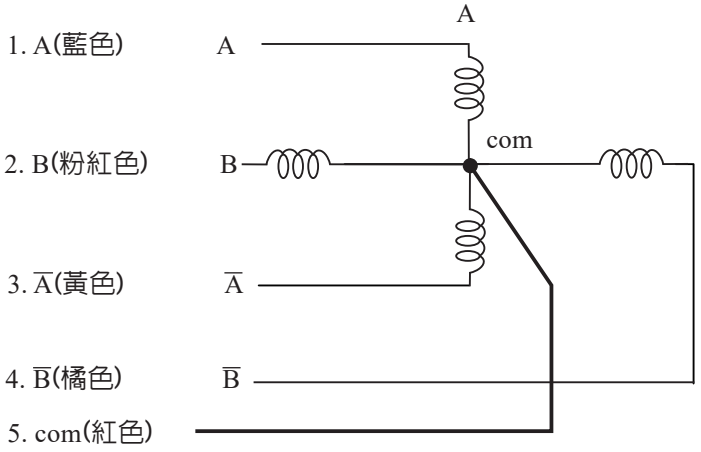


圖 6-30　步進馬達內部電路圖

因爲 Arduino 高電位最大電流僅 20mA，而 28BYJ-48 步進馬達至少需要 55mA，所以必須使用達靈頓電流放大電路，本例使用 ULN2003 模組，如圖 6-31：（摘自蝦皮商城）

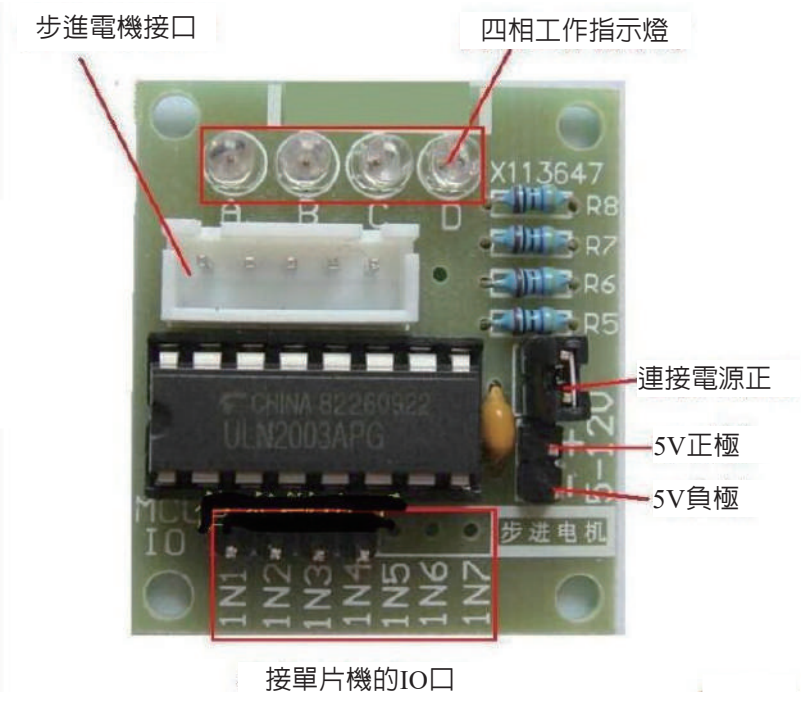


圖 6-31　步進馬達驅動器

Arduino、電流放大模組 ULN2003 與 28BYJ-48 步進馬達接線方式如圖 6-32：

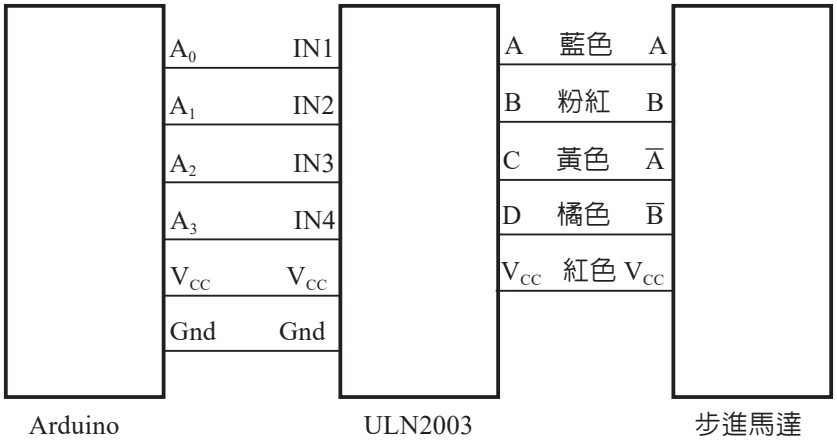


圖 6-32　步進馬達驅動電路

激磁方式

常見激磁有 1 相激磁、2 相激磁、1-2 相激磁，分別說明如下：

1相激磁

任何時間於線圈 A,B,$\overline{A}$,$\overline{B}$ 只有 1 個線圈激磁，產生 1 個磁力，方向如圖 6-33 箭號所示，此磁力的大小剛好牽引定子轉動一個齒輪的距離，激磁順序是 1000,0100,0010,0001，如圖 6-33 所示：

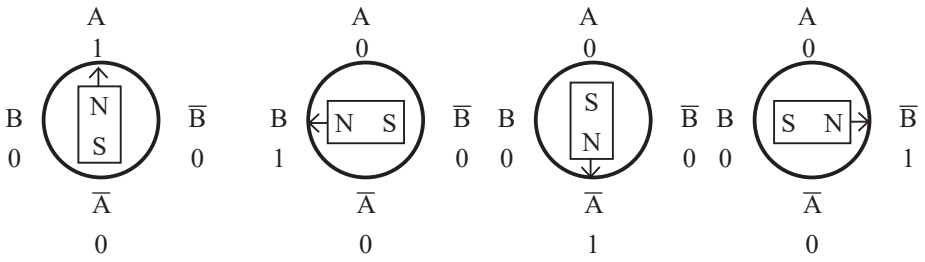


圖 6-33　1 相激磁示意圖

請鍵入以下程式，觀察步進馬達的轉動。

```
1.  //A0～A3 接UNL2003的IN1～IN4
2.  byte a[]={0x8,0x4,0x2,0x1};
3.  byte n=4;
4.  byte m=10;
5.  void setup() {
6.    DDRF=0xFF;
7.  }
8.  int i=0;
9.  void loop() {
10.     PORTF=a[i];
11.     delay(m);//調整延遲時間可調整轉速
12.     i=(i+1)%n;
13. }
```

以上 m 是延期時間，調整 m 可以調整轉速，但也不能太小，太小步進馬達扭力跟不上，就停止了，請自行實驗延遲時間的最小值。

自我練習

1. 請實驗延遲時間 m 的最小值。
2. 請計算步進馬達繞轉一圈，所需要激磁的次數。

2相激磁

任何時間於線圈 A,B,$\overline{A}$,$\overline{B}$ 只有 2 個線圈激磁，共同產生 1 個磁力，牽引定子轉動 1 個齒輪的距離，激磁順序是 1001,1100,0110,0011，如圖 6-34 所示：因爲每次 2 個線圈通電，產生的合力矩（方向如圖 6-34 的箭號）比前面 1 相激磁 1 個線圈通電還大，所以扭力比較大但是比較耗電。

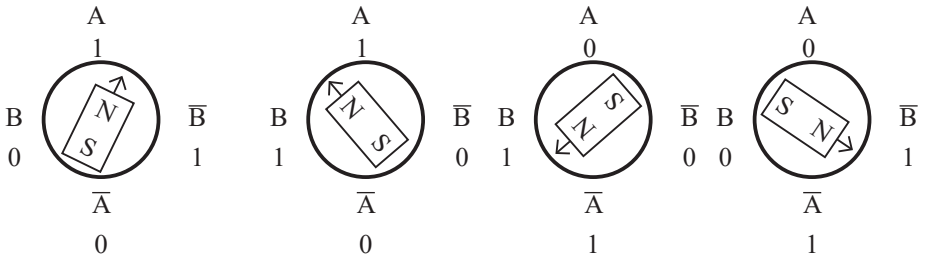


圖 6-34　2 相激磁示意圖

請鍵入以下程式，觀察步進馬達的轉動。

```
1.  //A0~A3 接ULN2003的IN1~IN4
2.  byte a[]={0x9,0xc,0x6,0x3};
3.  byte n=4;
4.  byte m=10;
5.  void setup() {
6.    DDRF=0xFF;
7.  }
8.  int i=0;
9.  void loop() {
10.     PORTF=a[i];
11.     delay(m);//調整延遲時間可調整轉速
12.     i=(i+1)%n;
13. }
```

自我練習

1. 請實驗延遲時間 m 的最小值。
2. 請計算步進馬達繞轉一圈，所需要激磁的次數。
3. 請使用 8 個指撥開關，調整轉速。
4. 請使用 1 個可變電阻，調整轉速。

1-2相激磁

交叉使用前面 1 相與 2 相激磁，線圈 A,B,$\overline{A}$,$\overline{B}$，激磁順序是 1001,1000,1100,0100,0110,0010,0011,0001，以上 8 個時序的合力矩方向請同時對照以上 1 相激磁與 2 相激磁，例如，1001 合力矩是右上，1000 合力矩是向上，1100 合力矩是左上，0100 合力矩是向左，所以每次移動半格，精密度比較高，由於是輪流使用 1 相與 2 相激磁，所以扭力也比 1 相激磁大。請鍵入以下程式，觀察步進馬達的轉動。

```
1.  //A0~A3 接ULN2003的IN1~IN4
2.  byte c[]={0x9,0x8,0xc,0x4,0x6,0x2,0x3,0x1};
3.  byte n=8;
4.  byte m=10;
5.  void setup() {
6.    DDRF=0xFF;
7.  }
8.  int i=0;
9.  void loop() {
10.     PORTF=c[i];
11.     delay(m);//調整延遲時間可調整轉速
12.     i=(i+1)%n;
13. }
```

自我練習

1. 請實驗延遲時間 m 的最小值。
2. 請計算步進馬達繞轉一圈，所需要激磁的次數。

反轉

只要調整激磁順序，就可改變方向，請鍵入以下程式，觀察步進馬達轉動方向。

```
1.  //A0~A3 接ULN2003的IN1~IN4
2.  byte a[]={0x9,0xc,0x6,0x3};//正轉
3.  byte b[]={0x3,0x6,0xc,0x9};//反轉
4.  byte n=4;
5.  byte m=10;
```

```
6.  void setup() {
7.    DDRF=0xFF;
8.
9.  }
10. int i=0;
11. void loop() {
12.     PORTF=b[i];
13.     delay(m);//調整延遲時間可調整轉速
14.     i=(i+1)%n;
15. }
```

按壓開關

請連接兩個按壓開關，腳位編號分別是 22 與 23，電路圖如圖 6-35：

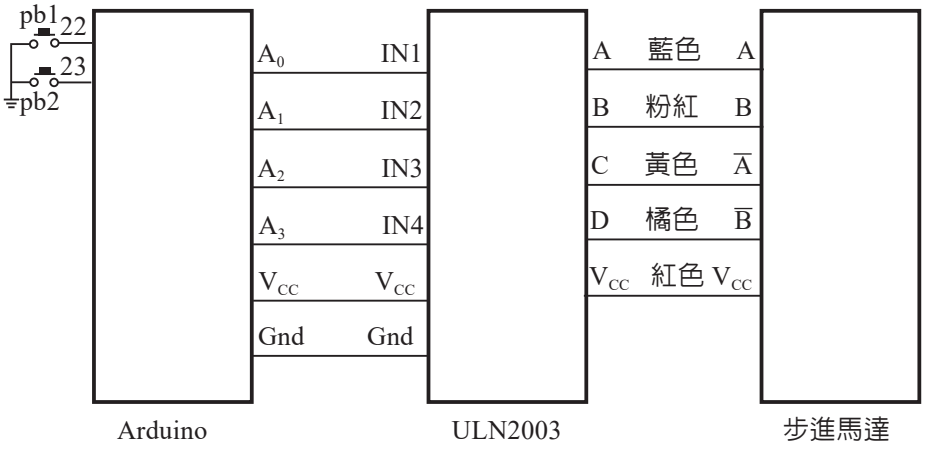


圖 6-35　步進馬達按壓開關控制圖

請鍵入以下，請使用按壓開關，控制馬達正反轉。

```
1.  //22,23 接pb1,pb2
2.  //A0~A3 接ULN2003的IN1~IN4
3.  byte a[]={0x9,0xc,0x6,0x3};//正轉
4.  byte b[]={0x3,0x6,0xc,0x9};//反轉
5.  byte n=4;
6.  byte m=10;
7.  void setup() {
8.    DDRF=0xff;
9.    for (byte i=22;i<=23;i++){
```

```
10.     pinMode(i,INPUT_PULLUP);//指派爲具有上拉電阻的輸入點
11.   }
12. }
13. int i=0;
14. void loop() {
15.     byte c=PINA;
16.     if(bitRead(c,0)==0){//偵測bit0的電位,腳位22
17.       forro();//正轉
18.     }
19.     if(bitRead(c,1)==0){ //偵測bit1的電位,腳位23
20.       revro();//反轉
21.     }
22. }
23. void forro(){//正轉
24.     PORTF=a[i];
25.     delay(m);//調整延遲時間可調整轉速
26.     i=(i+1)%n;
27. }
28. void revro(){//反轉
29.     PORTF=b[i];
30.     delay(m);//調整延遲時間可調整轉速
31.     i=(i+1)%n;
32. }
```

自我練習

1. 請到 Arduino 官網，線上查詢 bitRead(c,0) 的用法。
2. 請用 digitalRead(22)、digitalRead(23) 改寫本程式。

極限開關

極限開關如圖 6-36。（摘自露天生活 https://www.ruten.com.tw/）

圖 6-36　極限開關實體圖

若將步進馬達應用於位移控制，平台總有移動的左右極限，通常我們會安裝極限開關於左右極限位置，以便偵測平台是否接觸極限位置。圖 6-37 新增兩個極限開關，安裝於移動平台的左右極

限位置，此極限開關與指撥開關電路相同，平常處於高電位，當平台移動接觸極限開關時，極限開關傳回低電位。

於圖 6-37 電路圖中，按鍵 pb1 是右移按鍵，按鍵 S1 是右極限開關，當使用者連續一直按 pb1，就會碰到 S1 極限開關；按鍵 pb2 是左移按鈕，按鍵 S2 是左極限開關，當使用者一直按 pb2，平台就會碰到 S2 左極限開關。以下程式將要示範碰到極限開關就不能繼續移動，以免機台壞掉。

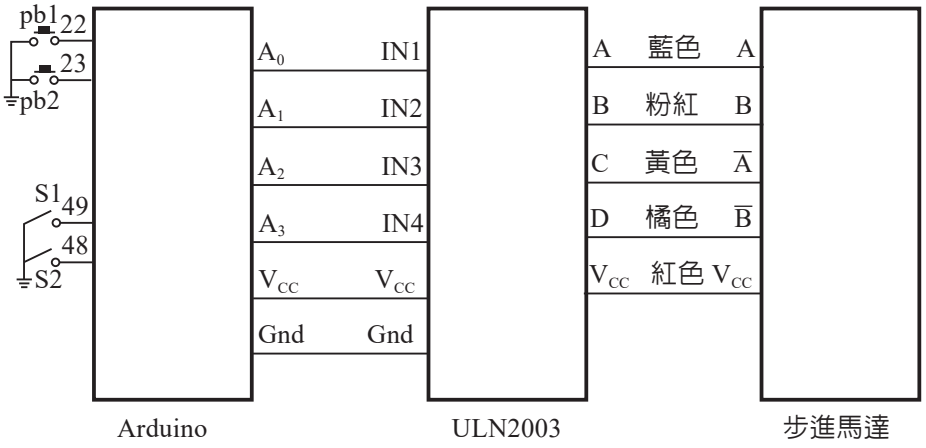


圖 6-37　極限開關控制圖

請鍵入以下程式，測試極限開關的動作（若無極限開關，亦可用指撥開關模擬）。

```
1.  //22,23 接pb1,pb2
2.  //49,48 接s1,s2
3.  //A0~A3 接ULN2003的IN1~IN4
4.  byte a[]={0x9,0xc,0x6,0x3};
5.  byte b[]={0x3,0x6,0xc,0x9};
6.  byte n=4;
7.  byte m=10;
8.  void setup() {
9.    DDRF=0xff;
10.   for (byte i=22;i<=23;i++){//按壓開關
11.     pinMode(i,INPUT_PULLUP);
12.   }
13.   for (byte i=49;i>=48;i--){//極限開關
14.     pinMode(i,INPUT_PULLUP);
```

```
15.   }
16. }
17. int i=0;
18. void loop() {
19.     byte c=PINA;//按壓開關
20.     byte d=PINL;//極限開關
21.     bool c1=bitRead(c,0);
22.     bool d1=bitRead(d,0);
23.     //本例按pb1時，平台將會碰到右極限開關，所以必須先判斷是否沒碰到右極
          限開關，才能繼續向右移動
24.     if(c1==0 && d1==1){
25.       forro();
26.     }
27.     bool c2=bitRead(c,1);
28.     bool d2=bitRead(d,1);
29.     //本例按pb2時，平台將會碰到左極限開關，所以必須先判斷是否沒碰到左極
          限開關，才能繼續向左移動
30.     if((c2==0) && (d2==1)){
31.       revro();
32.     }
33. }
34. void forro(){
35.     PORTF=a[i];
36.     delay(m);//調整延遲時間可調整轉速
37.     i=(i+1)%n;
38. }
39. void revro(){
40.     PORTF=b[i];
41.     delay(m);//調整延遲時間可調整轉速
42.     i=(i+1)%n;
43. }
```

自我練習

1. 請計算兩個極限開關可以移動的距離與所需呼叫 forro() 函式的個數個數。
2. 請計算移動 1cm，所需呼叫 forro() 函式的個數。

x-y平台移動

繪圖機、雕刻機、雷射雕刻機、洗床等如圖 6-38 所示：（摘自露天拍賣 https://www.ruten.com.tw/）都需要 3 個步進馬達移動平台，

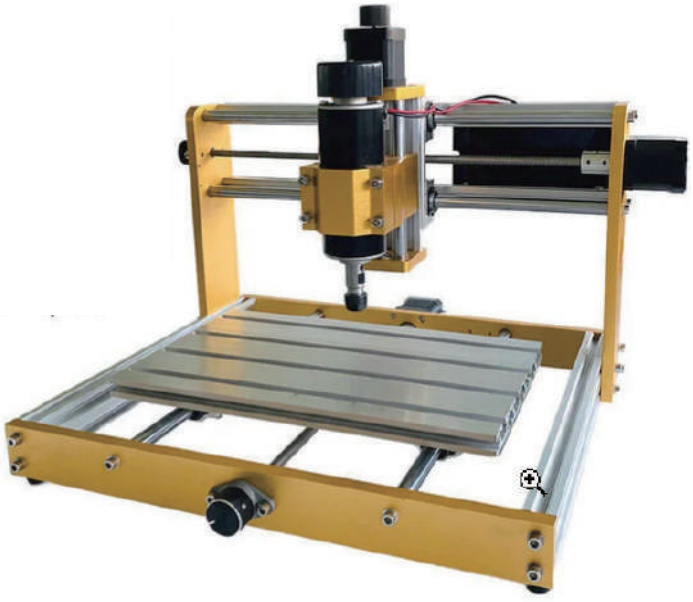


圖 6-38　x-y 平台

圖 6-39 左分別使用 PORTF、PORTK、PORTC 連接三個步進馬達，分別提供 x,y,z 軸的移動，此種平台通常有手動按鈕移動，也可程式控制移動，本例先製作 8 個按鈕（腳位 49~42），可以 8 個方向（如圖 6-40）手動移動 x-y 平台；按鈕 17,16 可以手動上昇與下降 z 軸。

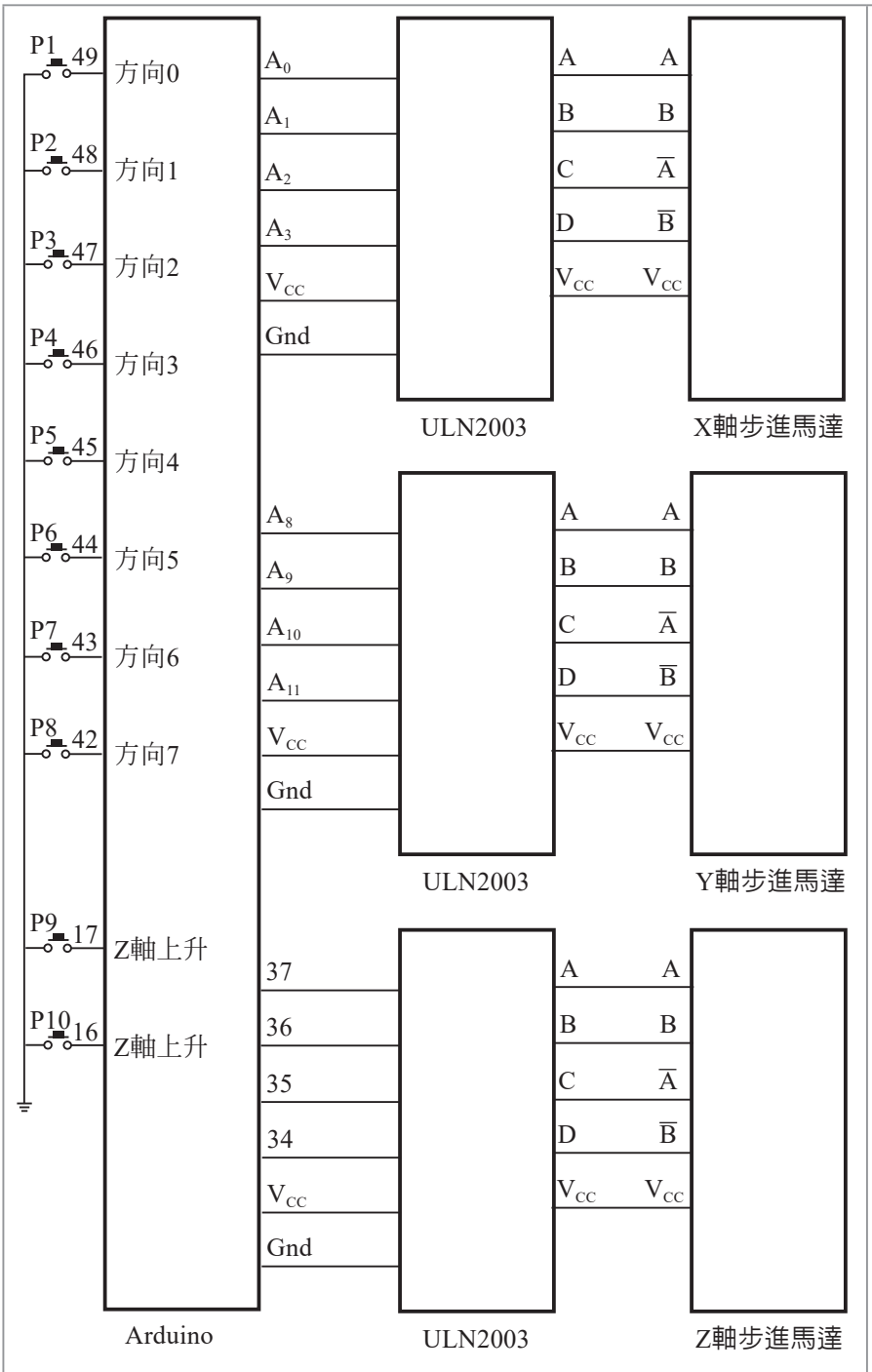


圖 6-39　x-y 平台電路圖

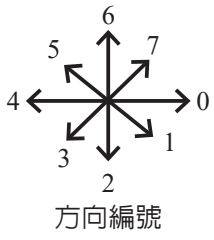


圖 6-40

請鍵入以下程式，按壓 8 個方向的按壓開關，觀察 x-y 軸步進馬達的移動；按壓開關 17,16 觀察 z 軸步進馬達的移動。

```
1.  //49~42 接p1~p8
2.  //17~18 接p9~p10
3.  byte a[]={0x9,0xc,0x6,0x3};//前進
4.  byte b[]={0x3,0x6,0xc,0x9};//後退
5.  byte m=10;//延遲時間
6.  void setup() {
7.    Serial.begin(9600);
8.    DDRF=0xff;//步進馬達x軸
9.    DDRK=0xff;//步進馬達y軸
10.   DDRC=0xff;//步進馬達z軸
11.   //按壓開關x,y軸
12.   for (byte i=22;i<=29;i++){//PORTA
13.     pinMode(i,INPUT_PULLUP);
14.   }
15.   //按壓開關z軸
16.   for (byte i=16;i<=17;i++){//PORTE
17.     pinMode(i,INPUT_PULLUP);
18.   }
19. }
20. int i=0;
21. void loop() {
22.     byte c=PINA;//偵測8個按壓開關
23.     ro(c);//x,y軸的移動
24.     //偵測按鍵16
25.     byte e1=digitalRead(16);
26.     if (e1==LOW)
27.       upz();//z軸上升
28.     //偵測按鍵17
29.     byte e2=digitalRead(17);
30.     if (e2==LOW)
31.       downz(); //z軸下降
32. }
33. void ro(byte x){ //x,y軸
34.     switch(x){
35.       case B11111110://0
36.           PORTF=a[i];//x軸右
37.           delay(m);
38.           break;
39.       case B11111101://1
40.           PORTF=a[i];//x軸右
```

```
41.           PORTK=b[i];//y軸下
42.           delay(m);
43.           break;
44.       case B11111011:  //2
45.           PORTK=b[i];//y軸下
46.           delay(m);
47.           break;
48.       case B11110111://3
49.           PORTF=b[i];//x軸左
50.           PORTK=b[i];//y軸下
51.           delay(m);
52.           break;
53.       case B11101111://4
54.           PORTF=b[i];//x軸左
55.           delay(m);
56.           break;
57.       case B11011111://5
58.           PORTF=b[i];//x軸左
59.           PORTK=a[i];//y軸上
60.           delay(m);
61.           break;
62.       case B10111111: //6
63.           PORTK=a[i];//x軸上
64.           delay(m);
65.           break;
66.       case B01111111://7
67.           PORTF=a[i];//x軸右
68.           PORTK=a[i];//y軸上
69.           delay(m);
70.           break;
71.     }
72.     i=(i+1)%4;
73. }
74. byte j=0;
75. void upz(){
76.     PORTC=a[j];//z軸上升
77.     delay(m);
78.     j=(j+1)%4;
79. }
80. void downz(){
```

```
81.     PORTC=b[j];//z軸上升
82.     delay(m);
83.     j=(j+1)%4;
84. }
```

自我練習

1. 以上範例使用 PINA 讀取按壓開關，請改用 digitalRead() 讀取，並改寫 ro() 函式。
2. 以上 3 個步進馬達，就需要 6 個極限開關，請自行加入極限開關控制。
3. 程式動作。以上是使用按鈕移動步進馬達控制，請讀者自行練習使用程式移動步進馬達。例如：
   (1) 分別計算 x 軸，y 軸，z 軸兩個極限開關之間，所需執行的脈衝個數。
   (2) x,y,z 軸能自動歸零。
   (3) x,y,z 軸移動 1cm，所需執行的脈衝個數。
   (4) x,y,z 軸能依照使用者所輸入單位移動。
   (5) x,y 軸能移動到指定座標，例如移到標 (3,4)。
   (6) z 軸能於 x,y 軸達定位前後，上下移動。

影像處理

前面 6-1 節已經介紹使用 Visual Basic 的 point(x,y) 函式處理字幕機的亮點，洗床、雕刻機、鑽孔機也是同樣原理，也是要用 point() 追蹤 x-y 軸移動的方向。追蹤黑點的方式，常見的有兩種，分別是循序法與迷宮法，說明如下：

循序法

循序法就如同印表機的噴墨原理，y 軸每次前進 1 格，x 軸依序左右掃描，若遇到黑點就給雷射光（雕刻機是遇到黑點，z 軸就下降）。以下圖爲例，雷射燒酌順序是 (R1,C1),(R1,C5),(R1,C7)，然後 y 軸下移一單位，繼續燒酌 (R2,C8),(R2C7),(R2,C6),(R2,C5),(R2,C4),(R2,C2)…

x軸 →
y軸 ↓

	C1	C2	C3	C4	C5	C6	C7	C8
R1	1	0	0	0	1	0	1	0
R2	0	1	0	1	1	1	1	1
R3	0	0	0	0	1	0	1	0
R4	0	1	0	0	1	0	1	0
R5	1	0	0	1	1	1	1	1
R6	0	0	0	0	1	1	0	0
R7	0	1	0	1	0	0	1	0
R8	1	0	1	0	0	0	0	1

▮迷宮法

迷宮法追蹤黑點的原理與繪圖機的繪圖相同。例如，連續兩個黑點如下：

●●

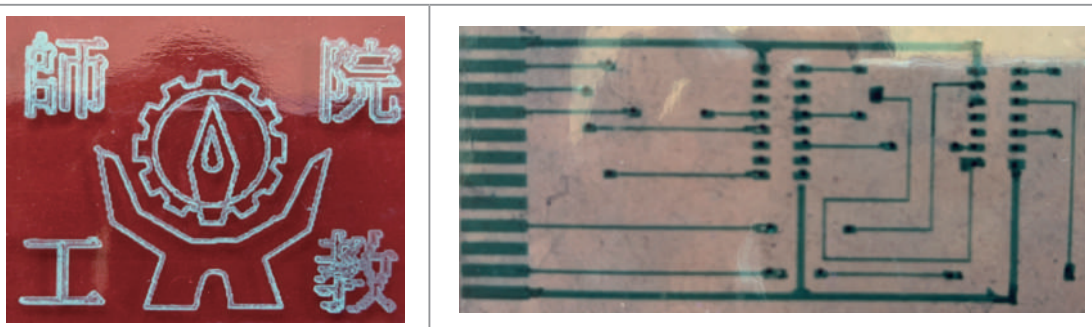
遇到第 1 黑點，表示 z 軸要能移來此座標且給雷射光（雕刻機是 z 軸向下），第 2 個黑點表示 x 軸向右移動，也就是連續呼叫以上 ro(0) ，至於呼叫的次數，就要依照自己的座標單位。又例如連續兩個黑點如下：

●
　●

第 1 黑點，同樣表示 z 軸要能移來此座標且給雷射光（雕刻機是 z 軸向下），第 2 個黑點表示 x 軸向「右下」移動，也就是連續呼叫以上 ro(1) 。以上每個黑點都要 8 個方向依序偵測旁邊是否繼續有黑點，以作爲 x-y 平台移動的方向。同樣以下圖爲例，當找到 (R1,C1)，表示 z 軸移來此，然後於 (R1,C1) 旁邊 8 個方向，偵測是否有黑點，接著在右下方找到黑點，將此方向紀錄下來，本例可記錄方向 1，等一會 x-y 步進馬達就依此方向移動。

	C1	C2	C3	C4	C5	C6	C7	C8
R1	1	0	0	0	1	0	1	0
R2	0	1	0	1	1	1	1	1
R3	0	0	0	0	1	0	1	0
R4	0	1	0	0	1	0	1	0
R5	1	0	0	1	1	1	1	1
R6	0	0	0	0	1	1	0	0
R7	0	1	0	1	0	0	1	0
R8	1	0	1	0	0	0	0	1

此一偵測黑點的演算法可參考資料結構裡的「老鼠走迷宮」程式，讀者可依照以上演算法撰寫洗床、雕刻機、鑽孔機等應用程式。例如，圖 6-40a 是筆者民國 76 年完成的作品，可雕刻文字與圖形，圖 6-40b 是掃描電路圖的黑點，洗床自動完成鑽孔動作。



(a)　　(b)

圖 6-40　x-y 平台作品圖

自我練習

1. 請到刻印章店鋪，觀察其雕刻機移動順序。
2. 請觀察印表機、繪圖機、3D 列印機的列印順序。

學後測驗

一、填充題

題號	題目	輸出結果

1	//PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
	//PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
	//請在8*8方格紙畫出哪幾個LED亮
	byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
	void setup() {
	DDRB=0xFF;DDRC=0xFF;
	}
	void loop() {
	PORTC=c[i];//請寫出8*8點陣LED哪幾個LED亮
	PORTB=0x32;
	}	

2	//PORTB 13,12,11,10,50,51,52,53 接8*8LED R1~R8
	//PORTC 30,31,32,33,34,35,36,37接8*8LED C1~C8
	//請在8*8方格紙畫出哪幾個LED亮
	byte c[]={0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd,0xfe};
	byte d[]={0xa1,0x4a,0x84,0x52,0x38,0x32,0x5f,0x92};
	void setup() {
	DDRB=0xFF;DDRC=0xFF;
	}
	void loop() {
	for (int i=0;i<=7;i++){
	PORTC=c[i];//位址
	PORTB=0xff;//資料
	delay(1); //
	}
	}	

二、實作題

1. 請用一個 8*8 點陣 LED 與一個按壓開關，每按一下按壓開關，計數值加 1。
2. 請用一個 LCD 與一個按壓開關，每按一下按壓開關，計數值加 1。
3. 請用一個蜂鳴器製作一個 8 個按鍵的電子琴。
4. 請用一個 4*4 的鍵盤，製作一個可以執行「+」,「-」,「*」,「/」的計算機。
5. 請用一個 4*4 的鍵盤與一個 LED，製作一個密碼門，內含 10 組員工編號，每個員工有一組 3 位數編號，若按對編號，LED 就亮，代表可通行。
6. 請用 2 個按鈕控制一個步進馬達，可以讓馬達正反轉。

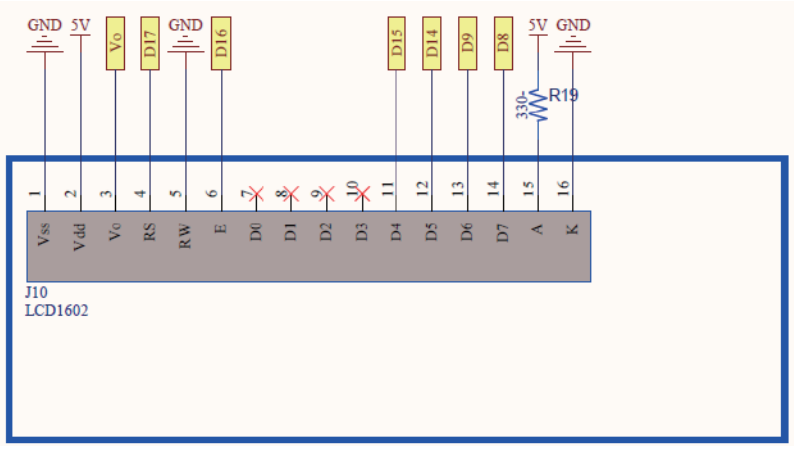
附錄

附錄1　本書教具電路圖

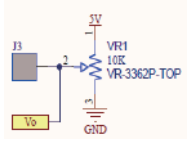
Arduino 魔法教具電路圖

更多程式說明請看 www.goodbooks.com.tw

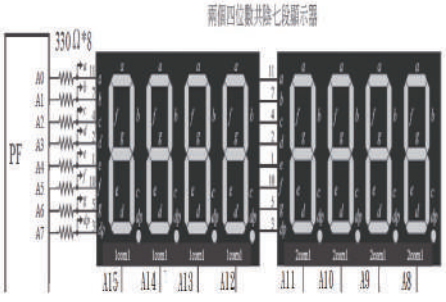
Arduino 2560




```
#include <LiquidCrystal.h> //lcd
LiquidCrystal lcd(17,16,15,14,9,8);//以lcd建構LiquidCrystal類別
```

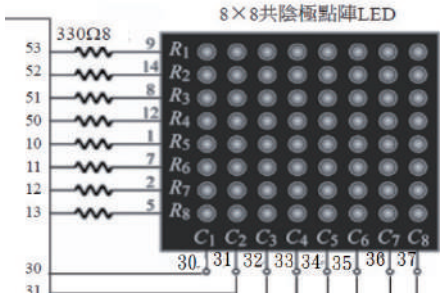


LCD亮度在此調整

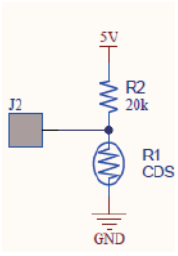


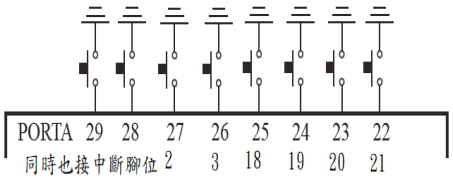

```
DDRF=0xFF;//資料線
DDRK=0xFF; //位址線
```

此圖依照教具電路板實際位置縮小排列，略微模糊，於次頁有清晰圖片。

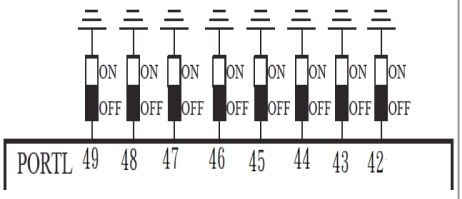



```
DDRB=0xFF;//資料線
DDRC=0xFF;//位址線
```

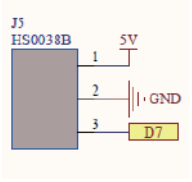


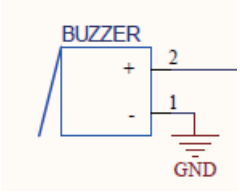


低位元在右邊

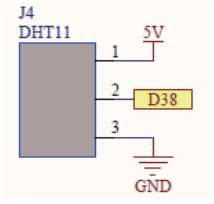


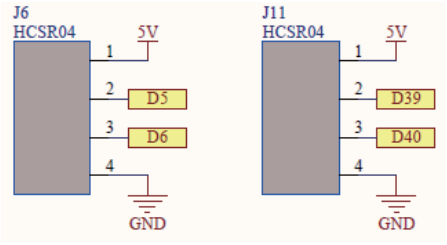
抱歉，低位元在左邊



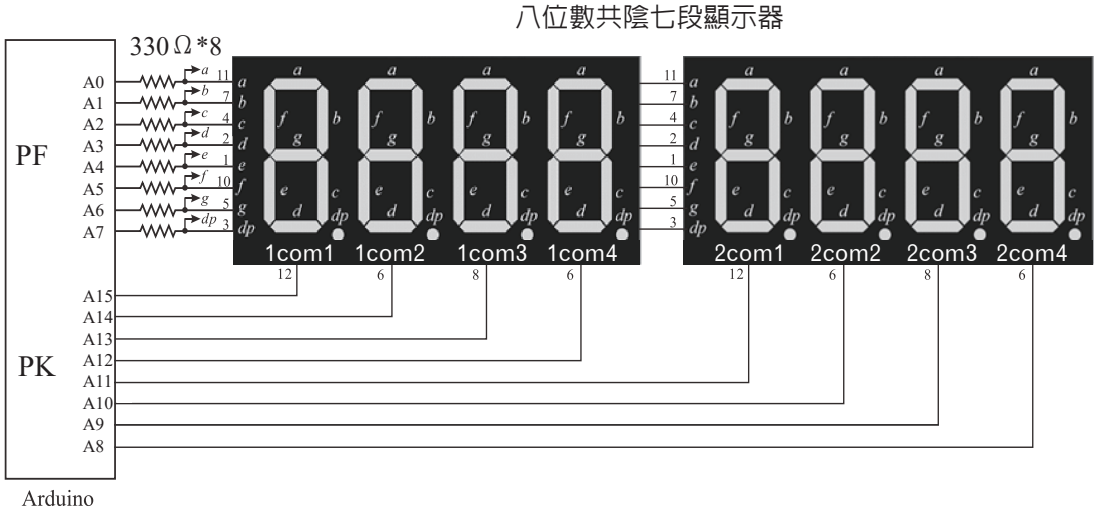


使用D4接腳





八位數共陰七段顯示器



國家圖書館出版品預行編目 (CIP) 資料

單晶片微處理機實習 / 洪國勝 , 孫銘宏 , 蔡懷文 , 鍾享旭 , 蔡懷介 , 陳蘊慈編著 . -- 初版 . -- 高雄市 : 泉勝出版有限公司 , 民 112.04
面 ; 公分
技術型高級中等學校電機與電子群

ISBN 978-986-99632-8-2(平裝)

1.CST: 微處理機 2.CST: 電腦程式設計 3.CST: 技職教育
528.831 112004022

十二年國民基本教育
技術型高級中等學校電機與電子群
單晶片微處理機實習（全一冊）
審定字號：技審字第 112012 號
書號：SE002
編著者：洪國勝、孫銘宏、蔡懷文、鍾享旭、蔡懷介、陳蘊慈
責任編輯：洪國勝
排版封面設計：陳彥慈（芒現）
發行所：泉勝出版有限公司
地址：高雄市左營區華夏路 708 號 17 樓
電話：0900757159 LINE ID：5abook
網址：www.goodbooks.com.tw
Email:aa163677@yahoo.com.tw
訂購方式：電話：0900757159 LINE ID：5abook Email:aa163677@yahoo.com.tw
初版日期：112 年 4 月
版次：初版第一刷
定價：400
ISBN：ISBN 978-986-99632-8-2 （平裝） NT$: 400