逢　甲　大　學

資訊工程學系專題報告

步進馬達系統

學　　生：林 漢 濱（四丁）

指 導 教 授：徐 弘 洋 老 師

中華民國九十三年四月

目　　錄

圖表目錄

摘　　要

此次專題是針對步進馬達之特性，再加上 MCS-51 單晶片之功能，兩者加以互相結合以其設計出一套步進馬達之控制系統。這份報告內容共有五章。以下即是這報告中各章節的內容重點說明：

第一章 導論

說明本專題的設計動機及目的。

第二章 步進馬達之簡介

介紹步進馬達的原理、激磁方式、驅動電路及其特性。

第三章 MCS-51 系統架構分析

介紹 MCS-51 單晶片的功能及其各接腳說明、記憶體結構、CPU 執行時序及其中斷結構。

第四章 系統設計說明

介紹專題之系統發展過程，包括軟體、硬體兩大部分。

第五章 心得感想

本次專題實作的一些心得感想。

第一章 導論

電機機械為工業發展不可欠缺的要素，特別是馬達佔有很重要的地位。但是一般的馬達是連續運轉的，對負載而言，,其動力的傳達是靠離合器(clutch)作ON-OFF的動作，目前以生產工廠自動化、省力化為目標的FA(Factory automation，工廠自動化的簡稱)，以及操作機器人(Robot)所使用的馬達，需考慮到電功率、耐環境性、價格及壽命等因素，同時對於決定位置之精確度、小型化及省能源等方面也必須注重。而且，以事務處理作業合理化及處理能力擴大為目標的OA(Office Automation，辦公室自動化的簡稱)，以及資訊終端機所使用的馬達，更是要求小型化、速應性、定速性、起動時間、分解能力及位置的精確性。步進馬達便是在此一環境下所產生出來的新型馬達。利用脈波信號做數位式的旋轉，是最主要的特性。

另外，由於MCS-51單晶片其功能強大、使用簡易、取得方便等優點。因為如此，我才會興起想運用單晶片之簡單使用之特性，來與步進馬達相結合，以期能設計出一套步進馬達控制系統，來對步進馬達能做到控制之目的，使其能廣泛且靈活地使用於工業用途上。以期能達到工業自動化的目的。而這就是我做此次專題的動機及目的所在。

第二章 步進馬達之簡介

2-1 步進馬達的特點

步進馬達有以下之特點:

1.旋轉的角度和輸入的脈波數成正比，因此用開迴路控制即可達成 高精確角度。
2.啟動、停止、正反轉的應答性良好，控制容易。
3.每一步級的角度誤差小，而且沒有累積誤差。
4.靜止時，步進馬達有很高的保持轉矩(Holding Torque)，可保持在 停止的位置，不需使用煞車迴路就不會自由轉動。
5.可靠性高，整個系統的價格低。

2-2 步進馬達的種類

步進馬達依定子線圈的相數不同可分成二相、三相、四相及五相式，小型步進馬達以四相式較為普遍。圖2-1即為四相步進馬達的內部接線圖。當送入一個脈衝電流至步進馬達，可在相對應處停止轉動，這種走一步即停住而得到的角度稱為基本步進角。步進角會因激磁方式不同而有所不同。

基本步進角的計算公式如下:

基本步進角=360°/(相數×轉子齒數)

例如:四相50尺的基本步進角為360°/(4×50)=1.8°

也就是說，四相50齒的步進馬達走200步正好是一圈。一般 的小型步進馬達齒數為50齒較多。

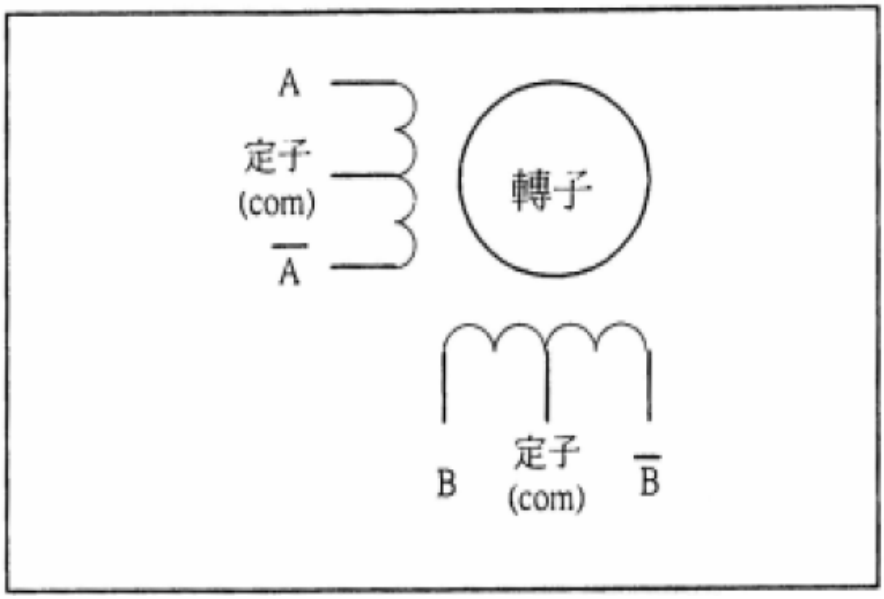


圖 2-1 四相步進馬達內部結構圖

2-3 步進馬達的用途

由於使用步進馬達，無論視作定速轉速、變速轉速、角度控制或位置控制均可免除繁雜的機械結構，使產品達成小型化、響應速度快、定速、解析度高、定位準確等要求。因此步進馬達被大量的使用在自動化方面。

以下即是一些典型的應用：

1. 硬式磁碟機-----磁頭定位。
2. 軟式磁碟機-----磁頭定位。
3. 印表機-----紙張傳送、印字頭驅動、色帶驅動。
4. 傳真機-----紙張傳送。
5. 影印機-----紙張傳送。
6. 紙帶閱讀機-----紙帶傳送。
7. 讀卡機-----卡片傳送。
8. 定長切割機-----定長輸出。
9. xy工作站-----xy軸定位。
10. 血液分析儀-----試紙傳送。

11.機械手臂-----定位控制。

12.放電加工機-----xy軸定位。

2-4 步進馬達的激磁方式

所謂激磁即是令步進馬達的線圈通過電流，以四相步進馬達而言，其定子線圈共有四個相，分別為A、/A、B及/B。而步進馬達的激磁方式有下列三種方式:

1.一相激磁：每次令一個線圈通過電流。步進角等於基本步進角，消耗電力小，角精確度好，但轉矩小，振動較大。其激磁方式及時序如圖3-2及3-4所示。

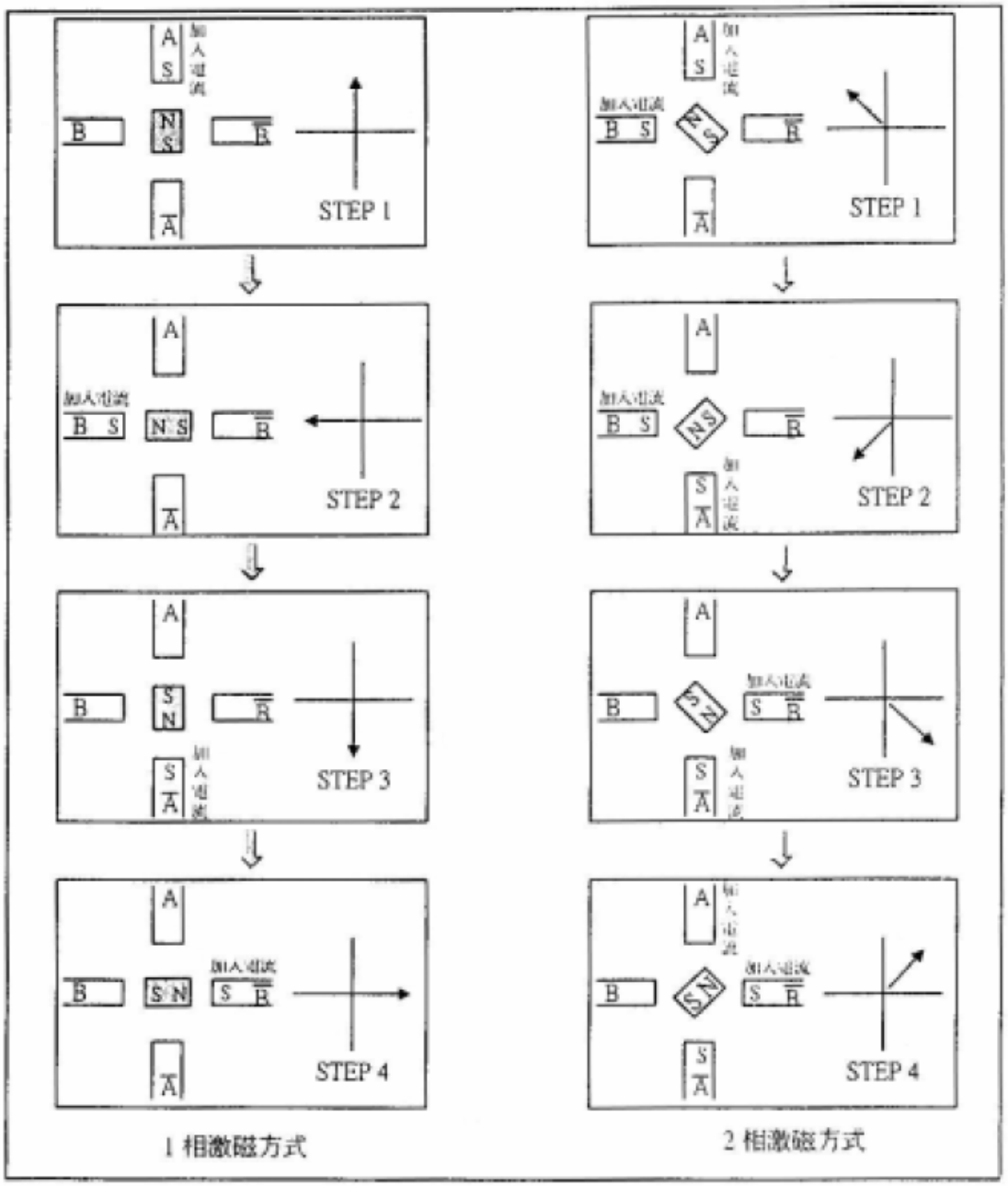
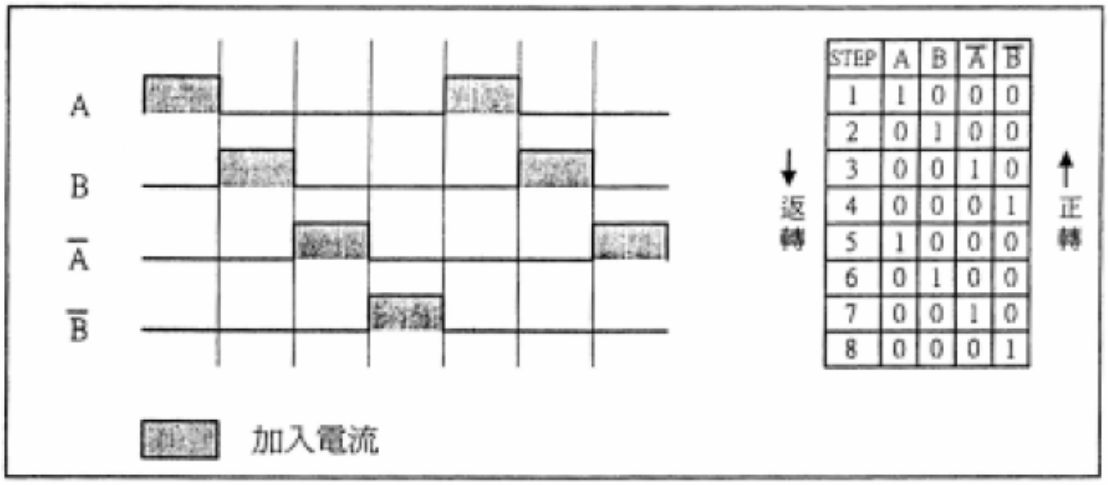


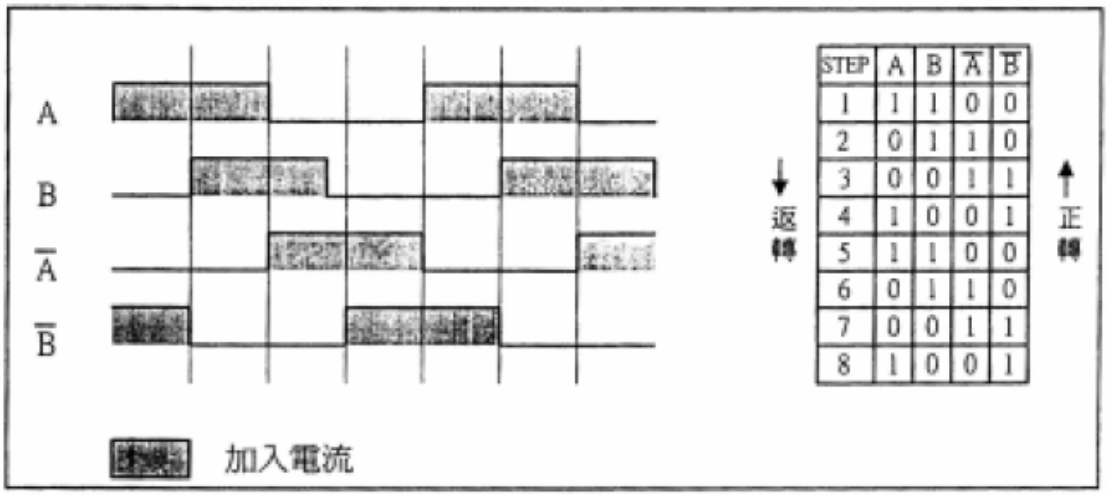
圖 2-2 一相／二相激磁方式



STEP	A	B	$\overline{A}$	$\overline{B}$
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

圖 2-3 一相激磁時序圖

2.二相激磁：每次令兩個線圈通電。步進角等於基本步進角。轉矩大、振動小，是目前較受普通採用的激磁方式。其激磁方式及時序如圖3-2及3-5所示。



STEP	A	B	$\overline{A}$	$\overline{B}$
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1
5	1	1	0	0
6	0	1	1	0
7	0	0	1	1
8	1	0	0	1

圖 2-4 二相激磁時序圖

3.一~二相激磁：一~二相激磁又稱為半步激磁，採用一相及二相輪流激磁；每一步進角等於基本步進角的1/2，因此解析度提高

一倍，且運轉更為平順，和二相激磁方式同樣受到普遍採用。其激磁方式及時序如圖3-3及3-6所示。

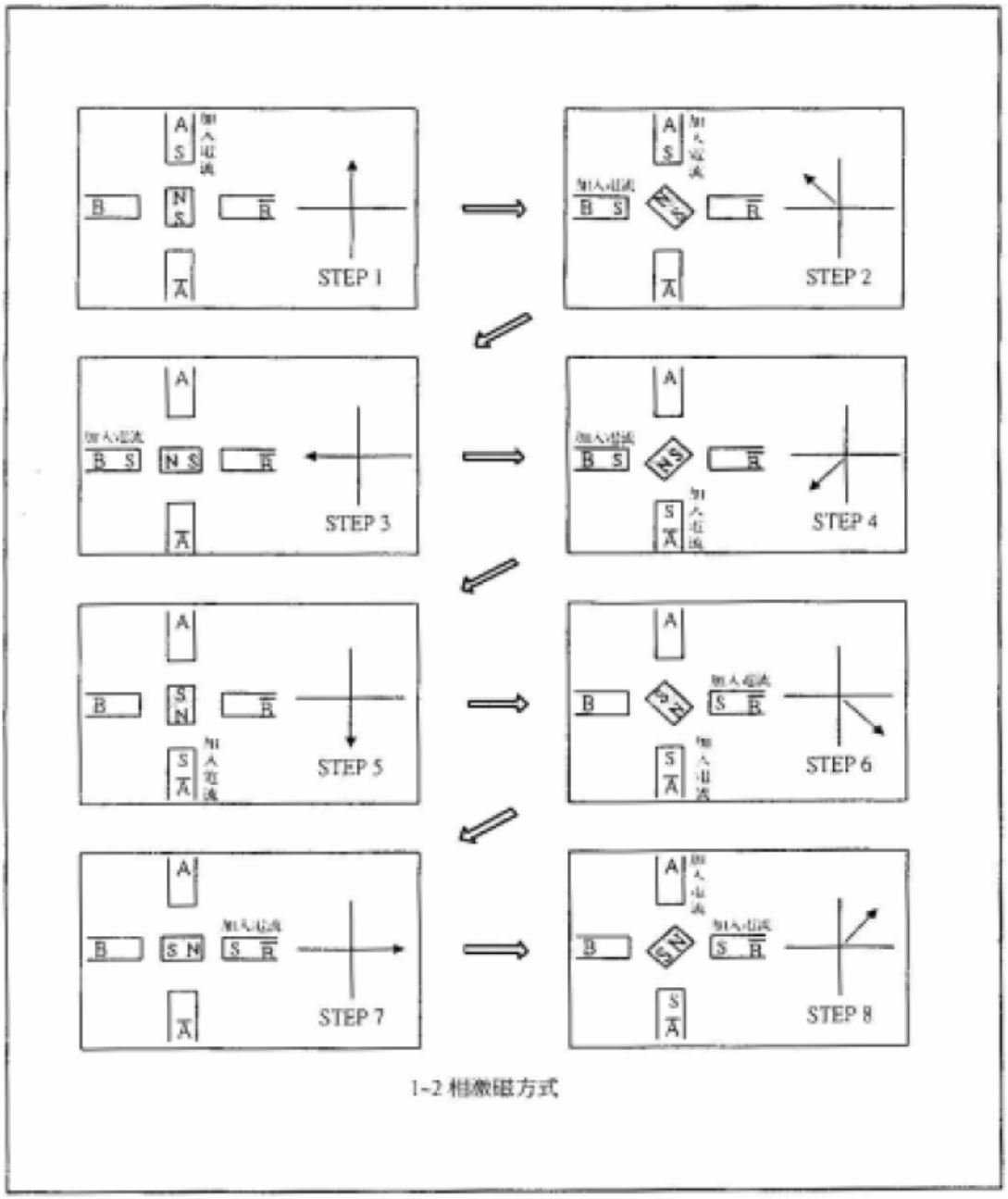
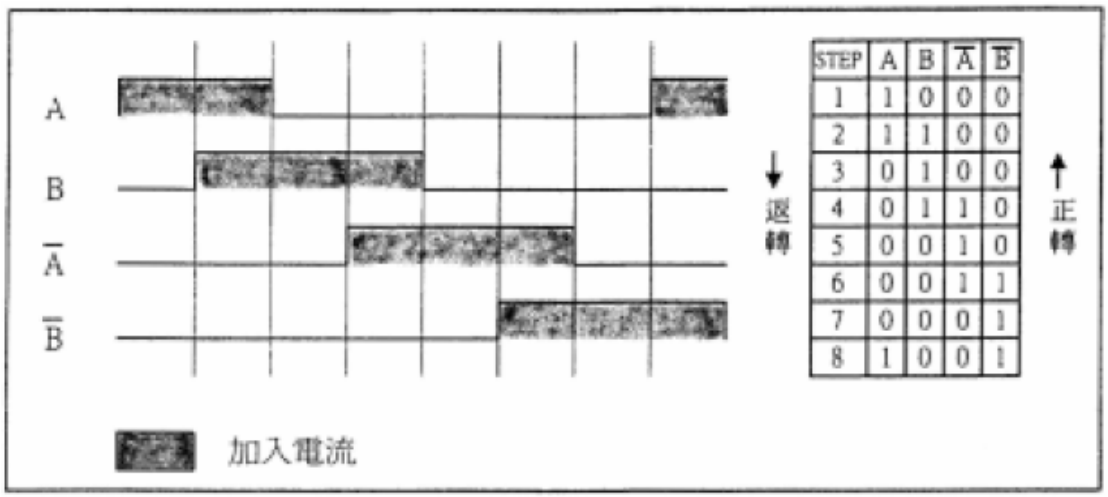


圖 2-5 一~二相激磁方式



STEP	A	B	$\overline{A}$	$\overline{B}$
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

圖 2-6 一~二相激磁時序圖

2-5 步進馬達的驅動電路

一個四相式步進馬達需要使用四個功率達寧頓電晶體來推動，如圖2-7所示。目前市面上也有販售四個達寧頓包裝在一起的高功率達寧頓IC，其型號為FT5754；其內部的四個達寧頓，各有3A的推動能力。如圖2-8即為FT5754內部電路的等效電路以及外部接腳圖。

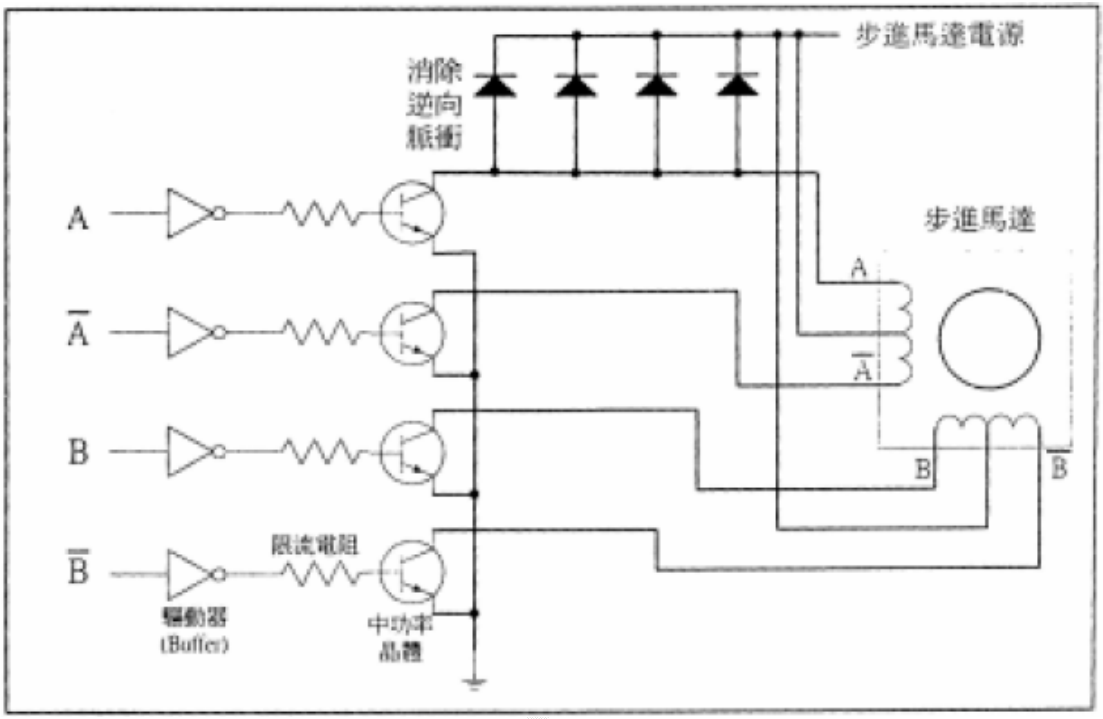


圖 2-7 步進馬達驅動電路圖

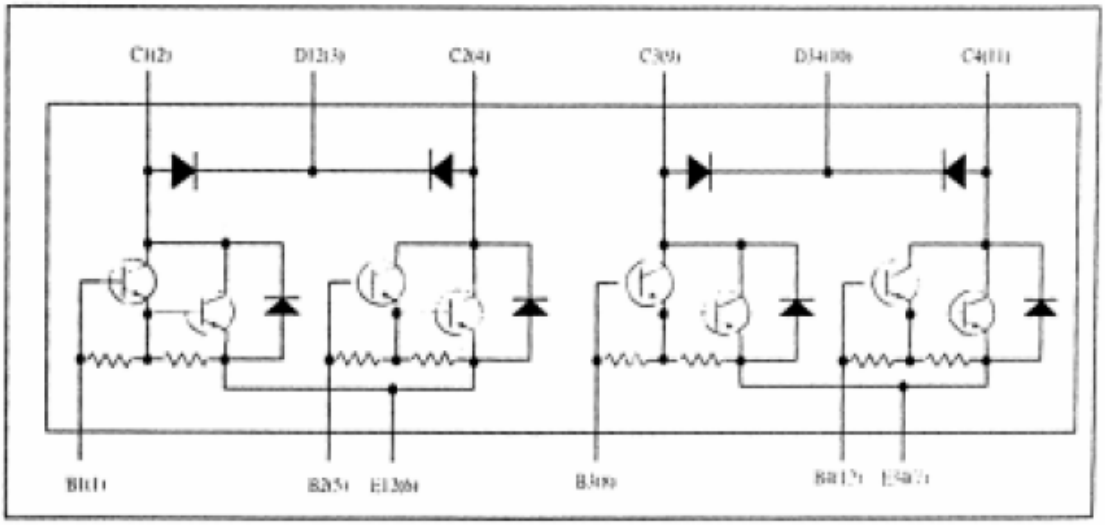


圖 2-8 步進馬達驅動IC—FT5754接腳圖

第三章 MCS-51 系統架構分析

3-1 MCS-51 功能簡介

MCS-51單晶片IC接腳圖如圖3-1所示：

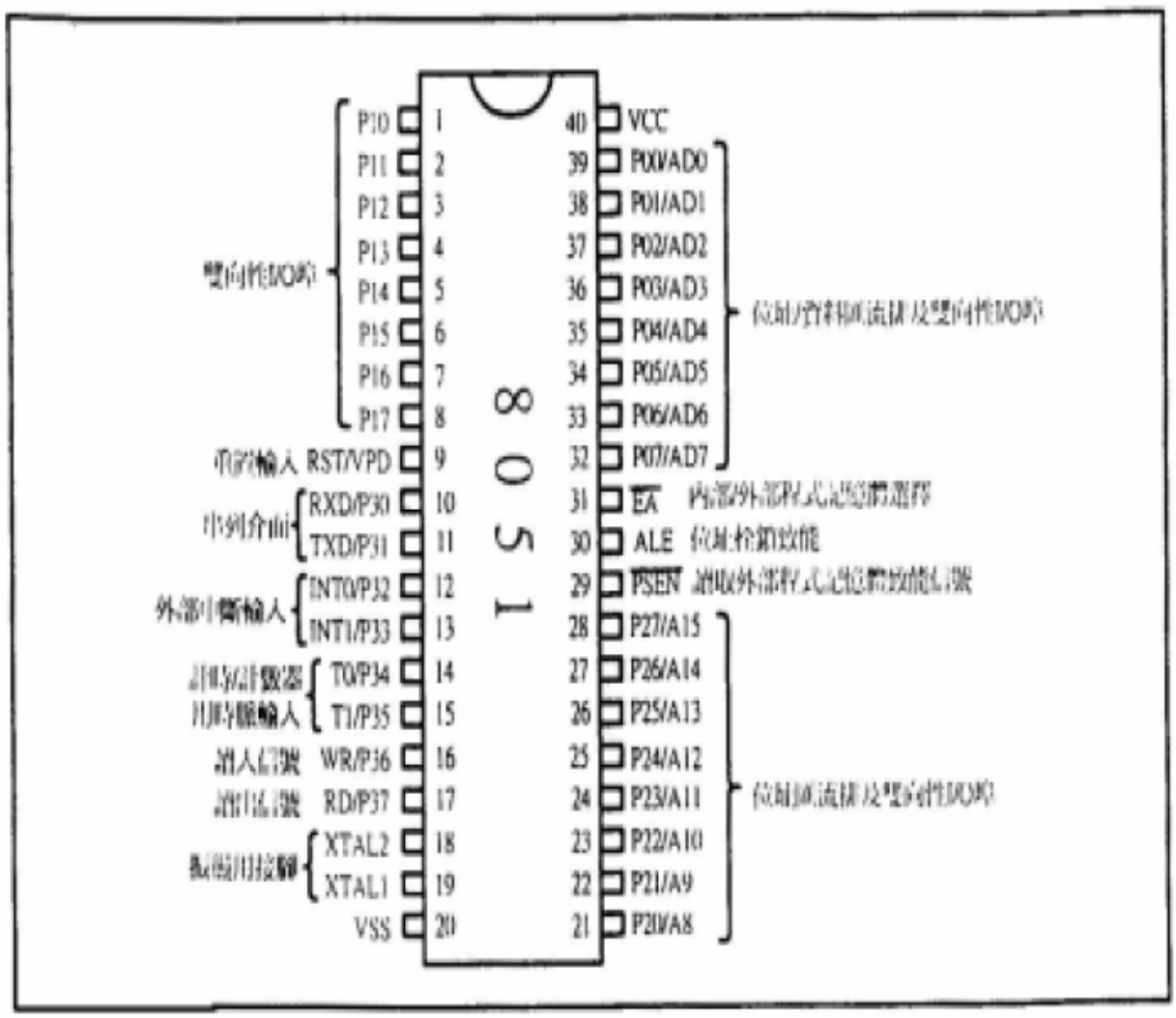


圖 3-1 MCS-51 單晶片接腳圖

MCS-51單晶片具有下列功能：

1. 8位元CPU。
2. 最高工作頻率為12MHZ(最快速度為1us/指令)。
3. 具有布林運算能力(即位元運算)。

4.共有4X8＝32條I/O埠。

5.有一組可規劃之串列通訊I/O埠。

6.有5個中斷源，且可規劃其優先權。

7.內部有兩組16位元計時/計數器。

8.內部程式記憶體有4K Bytes，並且最大可擴充至64K Bytes。

9.內部資料記憶體有128 Bytes，並且最大可擴充至64K Bytes。

3-2 MCS-51 硬體結構

在一顆MCS-51的單晶片中，到底內含有多少的功能呢!若以簡化後之主要方塊圖來描述MCS-51，則如圖3-2所示：

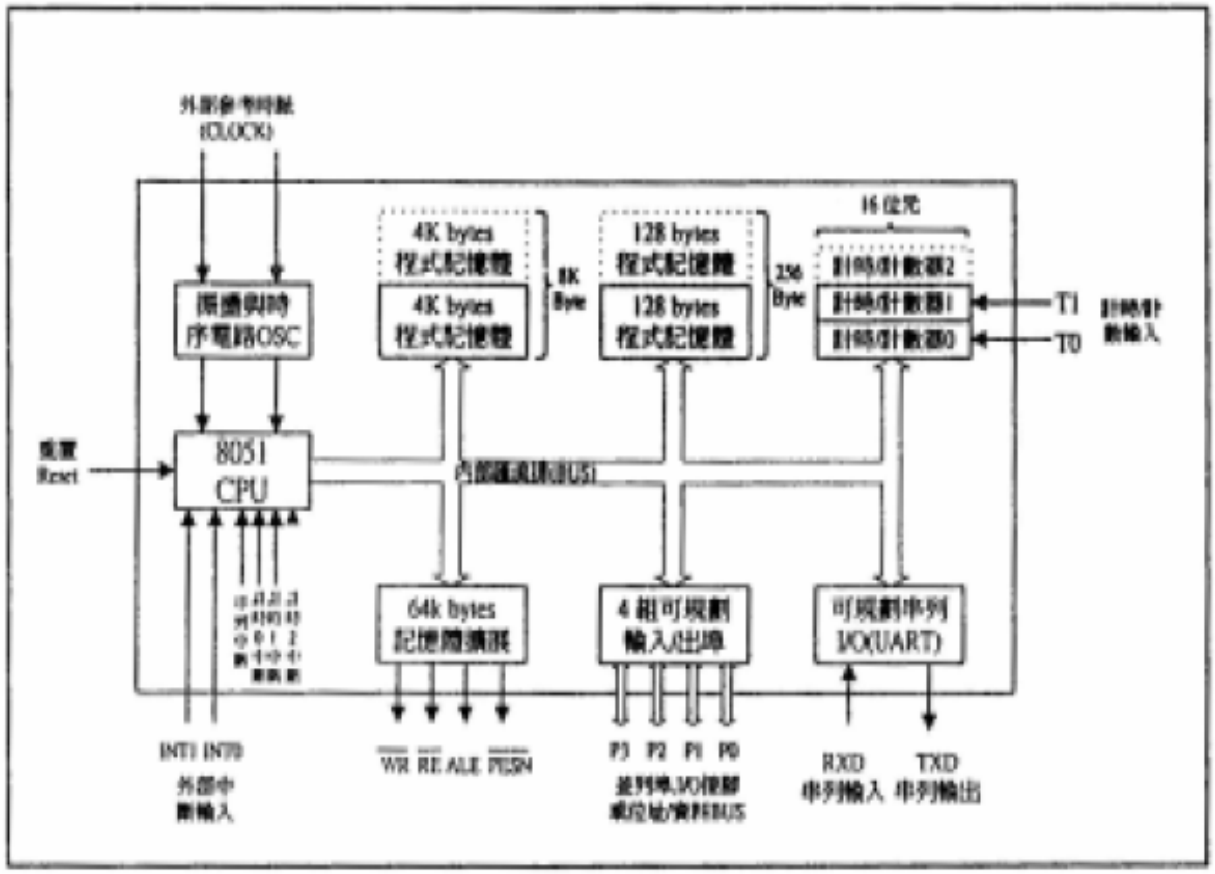


圖 3-2 MCS-51 硬體結構

1.振盪與時序方塊

MCS-51 內部有時脈振盪電路，只要外部加上石英振盪晶體，即可產生頻率非常穩定的脈波信號，所有MCS-51單晶片的時序都是以此為基準。

2. CPU方塊

這是整個單晶片的控制處理中心，PU讀取位於程式記憶體(ROM或EPROM)程式碼，經過計算及處理後，將結果送至各個暫存器或輸出入埠上，並且接受內部和外部的中斷信號，然後執行中斷服務程式。只要電源加入且振盪器開始動作後，CPU就會開始不停地動作。

3. 程式記憶體

MCS-51系列單晶片中，8051/8751提供內部4096 Bytes (4Kbytes)的程式記憶區，專攻儲存程式指令(碼)的地方。CPU所執行的程式指令，即是到這裡來提取的。8052提供內部8192 Bytes(8Kbytes)的程式記憶區，而8031/8031則不提供此方塊。若有內部程式記憶區時，CPU可以選擇執行的程式指令，是由內部的程式區提取或由外部的程式區提取。程式區的內容，只能讀出但不能寫入。

4. 資料記憶體

MCS-51系列中的8051/8031及8751單晶片都提供有128個 Bytes 的可讀/寫的資料記憶區，而8052系列則有256個Bytes資料記億區。這資料區中有16個Bytes共128bits的區域是可直接做單一位元定址的(Bit Address)，同時MCS-51也提供相當好用的位元處理指令。

5. 四組可規劃輸入/輸出埠 p0、p1、p2、p3

這四個埠共提供4X8＝32條I/O線，所有的埠都可以做位元組輸出入埠(Byte I/O)或者做單一位元輸出入埠(bit I/O)，當

MCS-51做外部記憶體擴充時，必須用Port0、Port2當作資料/位址線，配合ALE、/PESN及/WR、/RD等控制線產生必要的控制信號，做讀出(Read)及寫入(Write)信號。

6. 計時/計數方塊

MCS-51系列的單晶片均有2個16位元的計時/計數器，而8052則有3個。每個計時/計數器有多種模式工選擇，詳細內容及其功能將在稍後的章節中加以說明及探討。

7. 可規劃串列I/O方塊

MCS-51單晶片可透過此串列輸出入埠介面，與外界的電腦或儀器設備做資訊的交換，也可透過此介面作I/O的擴充，詳細內容及其功能也將在稍後的章節中加以說明及探討。

3-3 MCS-51 接腳功能說明

接腳編號	接腳名稱	功能說明
1~8	P1(P1.0~P1.7)	埠1(P1)是內部已經具有提升電阻的8位元雙向I/O埠，第1腳(P1.0)是LSB，第8腳(P1.7)是MSB。可位元定址，每隻腳可推動4個LS型TTL負載。
9	RESET	重置信號輸入腳。平常8051工作時這隻腳需保持〝LOW〞狀態。當這隻腳由外部輸入HIGH(+5V)的信號時，8051將被重置，重置後的8051將由位址0000H開始重新執行。
10~17	P3(P3.0~P3.7)	埠3(P3)也是內部已經具有提升電阻的8位元雙向I/O埠，可位元定址，每隻腳可推動4個LS型TTL負載。另外它的每一隻接腳都肩具有第二種功

		能，如下所示： P3.0:RXD(串列埠輸入端子) P3.1:TXD(串列埠輸出端子) P3.2:INT0(外部中斷0輸入端) P3.3:INT1(外部中斷1輸入端) P3.4:T0(計時/計數器0外部輸入端) P3.5:T1(計時/計數器1外部輸入端) P3.6:/WR(外部資料記憶體寫入信號端) P3.7:/RD(外部資料記憶體讀出信號端)
18~19	XTAL1/XTAL2	這兩支接腳是8051外部時脈振盪器的輸入端。
20	VSS	這支腳是8051的接地腳，使用8051時需將此接腳與系統的地線接在一起。
21~28	P2(P2.0~P2.7)	埠2(P2)也是內部已經具有提升電阻的8位元雙向I/O埠，可位元定址，每隻腳可推動4個LS型TTL負載。當存取外部記憶體時，此埠負責送出高位元組位址。
29	/PSEN	程式儲存致能接腳。 1.提取外部程式記憶體時，每一機械週 /PSEN動作2次。 2.存取外部資料記憶體時，/PSEN 保持在高電位，即跳過 2 次未動作，CPU於此時間內，存取外部資料。
30	ALE	位址栓鎖致能接腳。 當CPU對外部記憶體存取資料時，此腳輸出脈波之負緣，可將埠0(P0)送出之低位元組位址拴鎖在拴鎖器中(如74LS373)。在沒有存取外部記憶

		體 體時，此腳會固定輸出一時脈，其頻 率為振盪頻率之1/6。
31	/EA	外部存取致能接腳。 1./EA=1(VCC)，CPU執行內部程式記憶 體(ROM)，8051為4KB，而8052為8KB ，若超過此容量時，CPU將強迫提取外 部程式記憶體。 2./EA=0(GND)，CPU完全提取外部程式記 憶體，最大為64KB。當使用無ROM版 之8031或8032時，即需將此/EA接腳 接地。
32~39	P0(P0.0~P0.7)	埠0(P0)是內部沒有提升電阻的8位元開集極(Open Drain)結構之雙向I/O埠，可位元定址，每隻腳可推動8個LS型TTL負載，當作為I/O使用時，需外接提升電阻。當存取外部記憶體時，此埠為多工接腳，此埠先送出低位元組位址，CPU將其栓鎖在栓鎖器中與埠2之高位元組定址一位址，資料再經此埠進出。
40	VCC	8051的電源輸入端，電源規格為+5V±10%

3-4 系統時序

在MCS-51系列單晶片內，皆有振盪電路設計，如果我們要使用此振盪器，只需在XTAL1及XTAL2接腳之間，串接一石英晶體振盪器或陶質共振器，然後每支接腳再各接一個電容接地即可。內部振盪器將產生CPU執行程式時所需之狀態序列，每一個狀態序列有2個振盪週期，而一個機械週期是由6個狀態序列(S1~S6)，及12個振盪週期所組成，其說明如圖3-3所示：

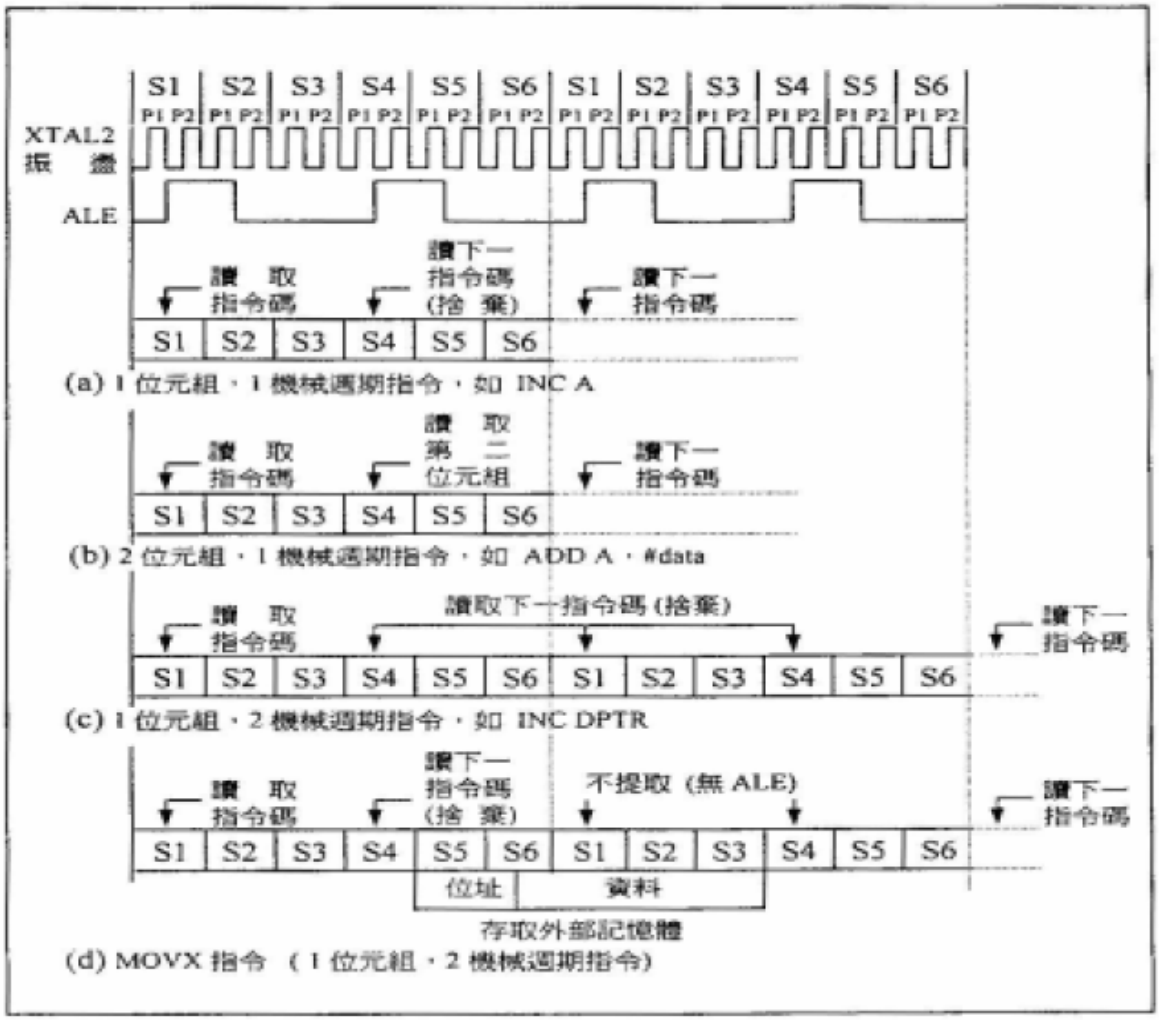


圖 3-3 MCS-51指令執行時序

3-5 MCS-51 記憶體結構

MCS-51的記憶體空間大致可分成底下的四個部分，如圖3-4所示。
1.1個獨立的16位元程式計數器(Program counter)。
2.64K Bytes的程式記憶體空間(Program Memory)。
3.64K Bytes的外部資料記憶體空間(External Data Memory)。
4.256 Bytes的內部資料記憶體空間(Internal Data Memory)。

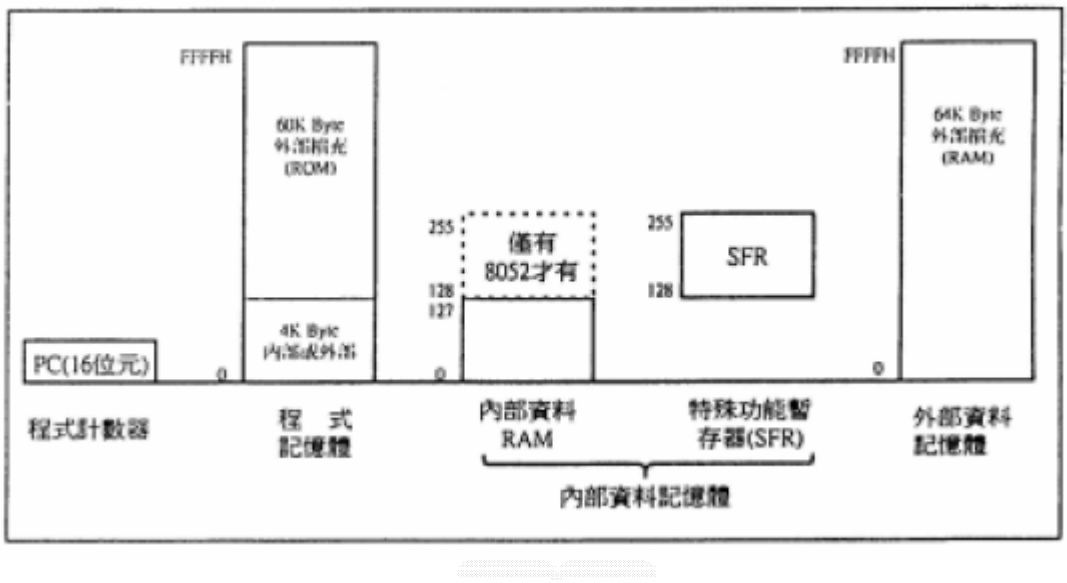


圖 3-4 MCS-51記憶體架構圖

程式計數器（Program Counter）

程式計數器是一個獨立的計數器，CPU執行程式時，必須知道程式執行到哪一個位址，而這就是程式計數器的功能了。MCS-51的程式計數器有16位元，因此其可以定址之最大範圍是2_{16}=64K，又因為MCS-51的記憶體寬度均為8位元，也就是一個位元組(BYTE)，因此也就是說其最大之記憶體空間為64K Bytes。其結構如圖3-5所示：
程式計數器指向CPU所要執行指令的位址，而當系統重置(Reset)時，程式計數器則指向於起始位址0000H。

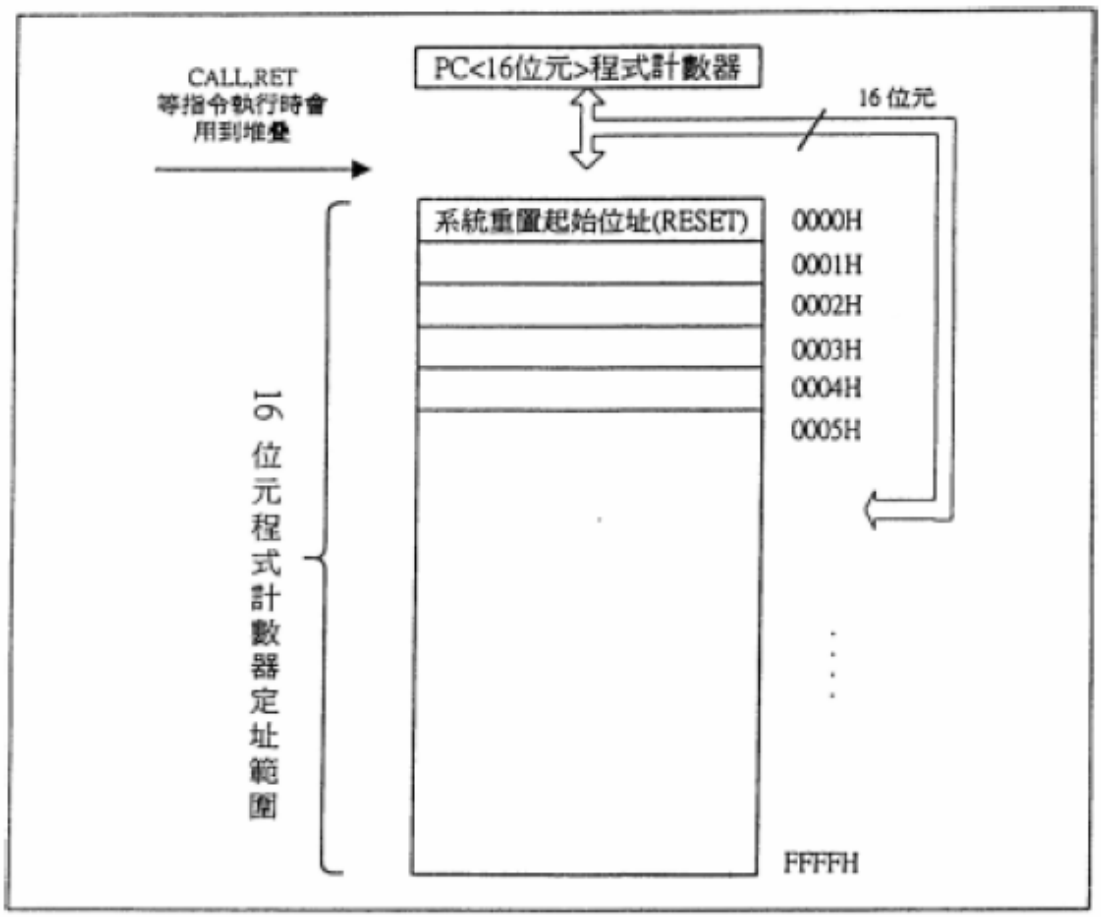


圖 3-5 16位元的程式計數器PC所能定址的空間

程式記憶體（ROM）

程式記憶體之主要用途是儲存程式碼。程式記憶體是存放CPU所要執行程式指令的地方，因此程式記憶體中的資料，只能被CPU讀取，而不能被CPU寫入資料。其結構如圖3-6所示：

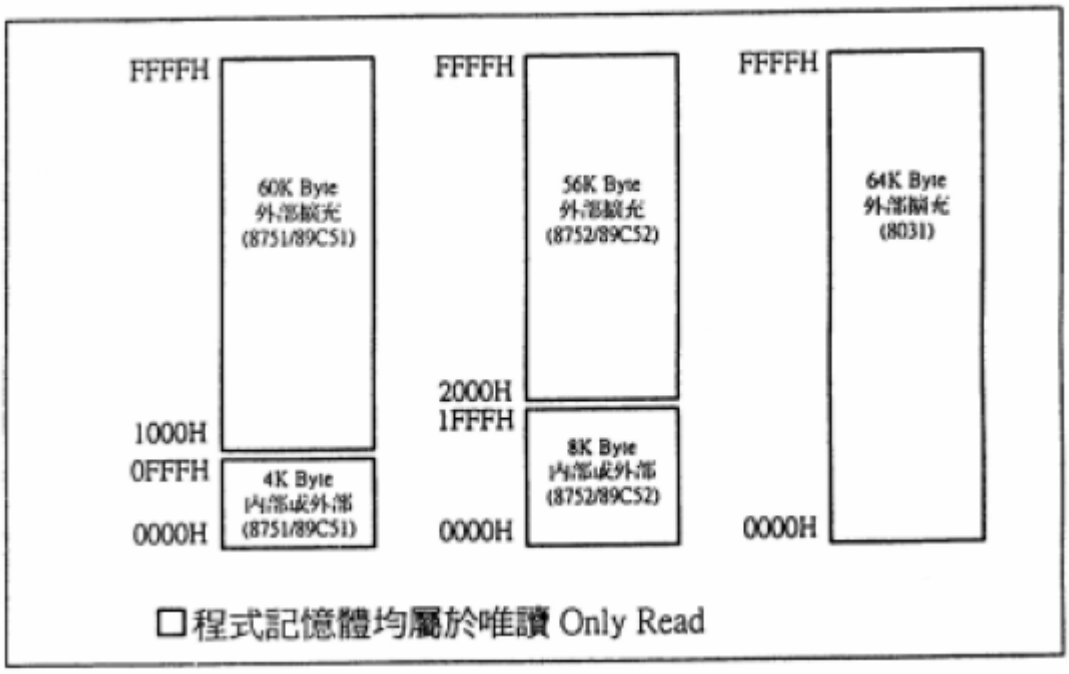


圖 3-6 MCS-51程式記憶體結構圖

資料記憶體（RAM）

MCS-51的資料記憶體，可分成內部資料記憶體與外部資料記憶體，其空間的分配情形如圖3-7所示。在8031、8051、89C51單晶片內部有128個位元組(Byte)的資料記憶體，8032、8052、89C52單晶片內部則有256個位元組(Byte)的資料記憶體。MCS-51允許在外部擴充64K Bytes資料記憶體(RAM)，這64K位址空間裡，除了可以放置RAM外，還可以用來作為一些週邊I/O(例如8255等)的位址空間。

內部資料記憶體使用MOV指令來存取，而外部資料記憶體則使用MOVX指令存取。

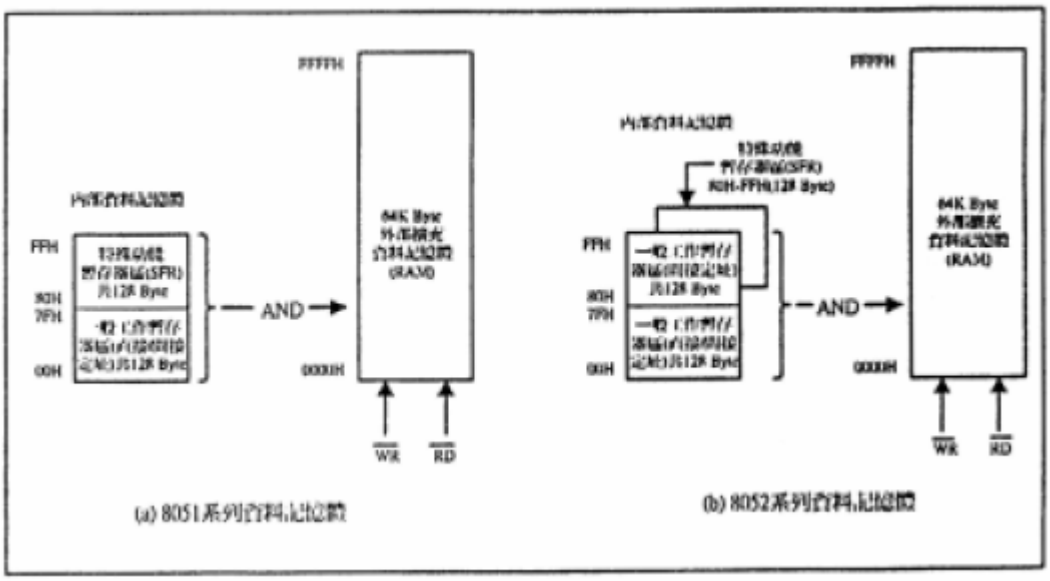


圖 3-7 MCS-51資料記憶體結構圖

內部資料記憶體

MCS-51內部具有256 Bytes的內部資料記憶空間，這塊空間前面128 Bytes(00H~7FH)是當作存放資料暫存區，後面128 Bytes(80H~FFH)，則是作為特殊功能暫存器區(SFR)的地方。其結構如圖3-8所示。

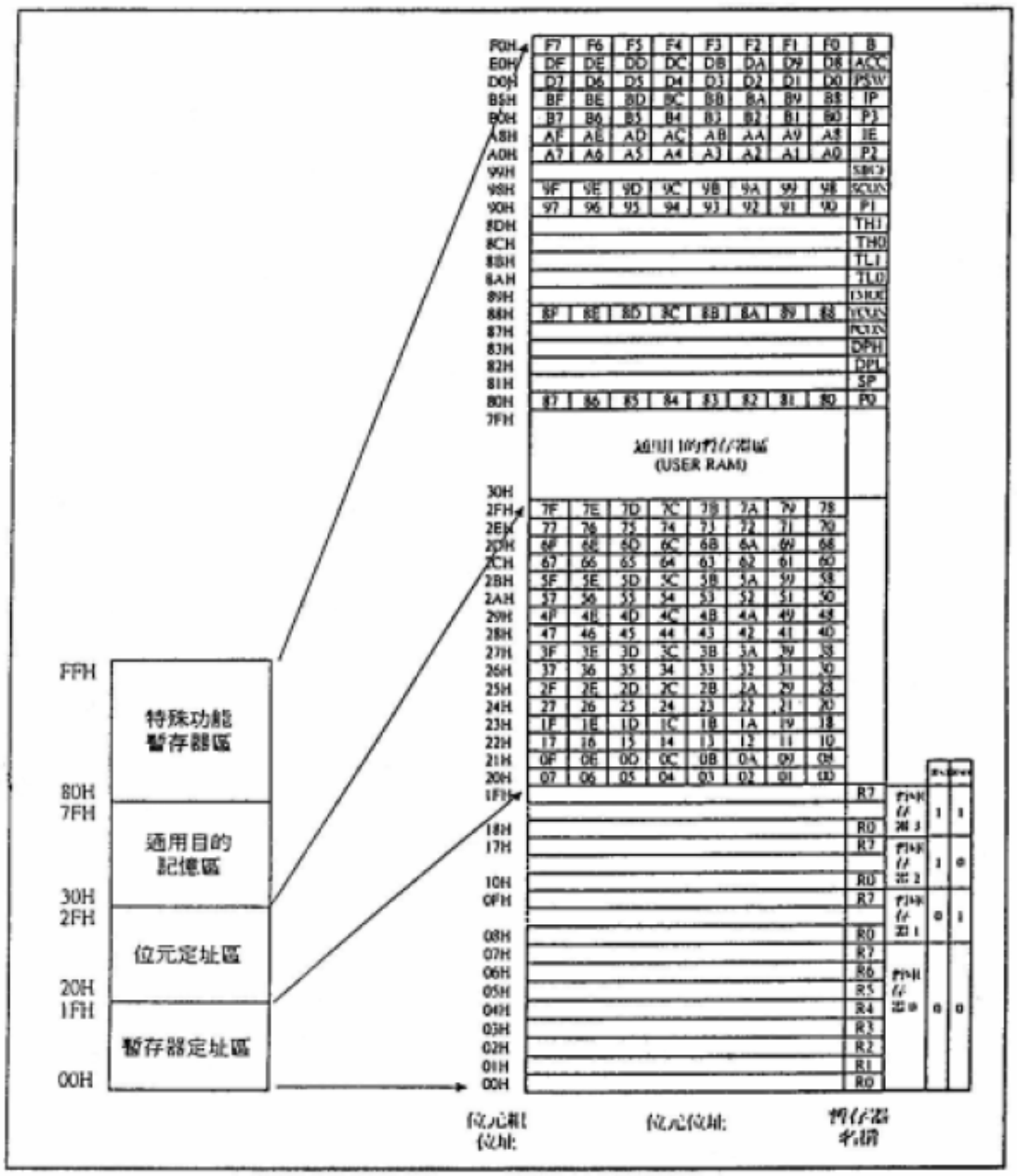


圖 3-8 MCS-51內部資料記憶體RAM結構圖

特殊功能暫存器（SFR）

特殊功能暫存器簡稱為SFR(Special Function Register)其位址範圍是位於資料記憶體位址80H~FFH的地方，共128 Bytes。它是存放8051內部週邊暫存器的地方，例如計時/技術器的Counter等。而下表3-1即是特殊功能暫存器SFR之一覽表。

符　號	名　稱	位 址
●ACC	累積器 (accumulator)	0E0H
●B	暫存器 B(B register)	0F0H
●PSW	程式狀態字組 (program status word)	0D0H
SP	堆疊指標 (stack pointer)	81H
DPTR	資料指標暫存器 (data pointer 2 bytes)	DRL
DRL	DPTR 低位元組 (low byte)	82H
DPH	DPTR 高位元組 (high byte)	83H
●P0	埠 0(port 0)	80H
●P1	埠 1(port 1)	90H
●P2	埠 2(port 2)	0A0H
●P3	埠 3(port 3)	0B0H
●IP	中斷層次控制暫存器 (interrupt priority control)	0B8H
●IE	中斷致能控制暫存器 (interrupt enable control)	0A8H
TMOD	計時/計數模式控制 (timer/counter mode control)	89H
●TCON	計時/計數控制暫存器 (timer/counter control)	88H
TH0	計時/計數器 0 高位元組 (timer/counter 0 high byte)	8CH
TL0	計時/計數器 0 低位元組 (timer/counter 0 low byte)	8AH
TH1	計時/計數器 1 高位元組 (timer/counter 1 high byte)	8DH
TL1	計時/計數器 1 低位元組 (timer/counter 1 low byte)	8BH
●SCON	串列埠控制暫存器 (serial control)	98H
SBUF	串列埠資料緩衝器 (serial data buffer)	99H
PCON	功率控制暫存器 (power control)	87H

註：●＝可位元定址之暫存器。

表 3-1 特殊功能暫存器SFR一覽表

在SFR裡的各位元組都有其各自的暫存器名稱，我們在使用這些暫存器時，可在指令中直接使用它們的名稱即可，而不需要使用位址。另外SFR各暫存器在重置(RESET=1)以後，SFR裡面的暫存器都將會被設定為一個固定值，如下表3-2所示。

暫存器名稱	以二進制表示之值
ACC	000000000
B	000000000
PSW	000000000
SP	000000111
DPTR：	
DPH	000000000
DPL	000000000
P0	111111111
P1	111111111
P2	111111111
P3	111111111
IP	8051 XXX00000/8052 XX000000
IE	8051 0XX00000/8052 0X000000
TMOD	000000000
TCON	000000000
＋T2CON	000000000
TH0	000000000
TL0	000000000
TH1	000000000
TL1	000000000
＋TH2	000000000
＋TL2	000000000
＋RCAP2H	000000000
＋RCAP2L	000000000
SCON	000000000
SBUF	未定
PCON	HMOS 0XXXXXXX/CHMOS 0XXX0000
＋＝只 8052 有	
X＝未定	

表 3-2 SFR各暫存器重置後之初始值

1. ACC(Accumulator)累積器

ACC為一般的8bit累積器，凡是CPU的所有運算動作都得由ACC來運算處理。

2. B工作暫存器

B暫存器是一個8bit的工作暫存器，平常使用與一般工作暫存器

相同，但與累積器(ACC)組成時可做乘除運算。

3. SP(Stack Pointer)堆疊指標
SP是8bit寬的堆疊指標，指示內部RAM上的堆疊起始位址。在RESET 重置後，SP的內容會自動設成07H。

4. DPTR(Data Pointer Register)資料指標暫存器
DPTR是由DPH及DPL(8bit)這兩個8位元暫存器所組成的16位元暫存器，高位元組是DPH，低位元組是DPL。DPH與DPL可以視為 兩個獨立的8位元暫存器使用，也可以合成一個16位元的DPTR16位元暫存器使用。DPTR主要的用途是用來定址外部的資料記憶體（可使用指令MOVX A，@DPTR)，或定址程式記憶體(MOVC A，@A+DPTR）。

5. P0，P1，P2，P3暫存器
P0，P1，P2，P3是MCS-51的四個I/O埠。

6. TH0，TL0，TH1，TL1計時/計數暫存器
TH0，TL0與TH1，TL1這兩隊各8位元的暫存器分別組成二組16 位元的計時/計數暫存器。這二組16位元計時/計數暫存器的名稱分別是Time0，Time1。

3-6 MCS-51 中斷結構

何謂中斷?

所謂中斷是指：當為處理機正在執行一個程式(例如主程式)時，突然有信號來打斷目前的工作，這時CPU必須暫停原來執行的程式，而先跳去指定的另一個程式(例如中斷服務程式)執行，等到執行完指定的程式之後，再回去繼續執行原來的主程式。

一個完整的微處理機需要有很多的週邊裝置。以8051為例，

如內部的兩個計時/計數器、串列埠UART，以及兩個外部中斷要求輸入腳，這些都是屬於8051的週邊裝置。CPU若要完成各項週邊或I/O的工作，最普遍的方式是採用輪詢(Polling)方法，及CPU依序地去詢問每一週邊裝置或I/O是否需要服務。因此採用輪詢方式的主程式，必須是一個能夠持續詢問各裝置的迴圈。如此一來，不管週邊裝置是否有提出要求，CPU都必須花時間(浪費時間)去詢問每個週邊裝置，將使得程式執行效率很低。

採用中斷的方式，則能有效地管理眾多的週邊裝置。CPU平時執行主程式，當某個週邊裝置需要服務的時候，才會發出中斷要求信號，告訴CPU停止目前的主程式的工作，去坐週邊裝置的工作，或者說去執行該週邊裝置的中斷副程式。執行完中斷副程式後，在跳回主程式繼續執行為完成的工作。

對於8051的週邊裝置而言，Timer0、Time1及UART是屬於內部週邊裝置，其中斷要求信號已在內部街道8051的中斷系統。而8051外部也提供了兩條中斷要求線(INT0和INT1)，以提供外部擴充的週邊裝置對8051提出中斷要求使用。

8051中斷介紹

8051提供了5個中斷源，8052則提供了6個中斷源。
圖3-9所示的是MCS-51的中斷邏輯控制圖，在MCS-51中任何的中斷產生，都必須經過下列三個步驟方才有效：

1. 設定中斷致能暫存器IE中的EA位元為1。
2. 設定中斷致能暫存器IE中，相關的個別中斷位元為1。(例如：INT0 的中斷致能位元是EX0)
3. 中斷要求旗標提出中斷要求，也就是設定中斷要求旗標為1。

下表3-3是MCS-51的各個中斷源所屬的中斷要求旗標及中斷旗標所屬暫存器圖。

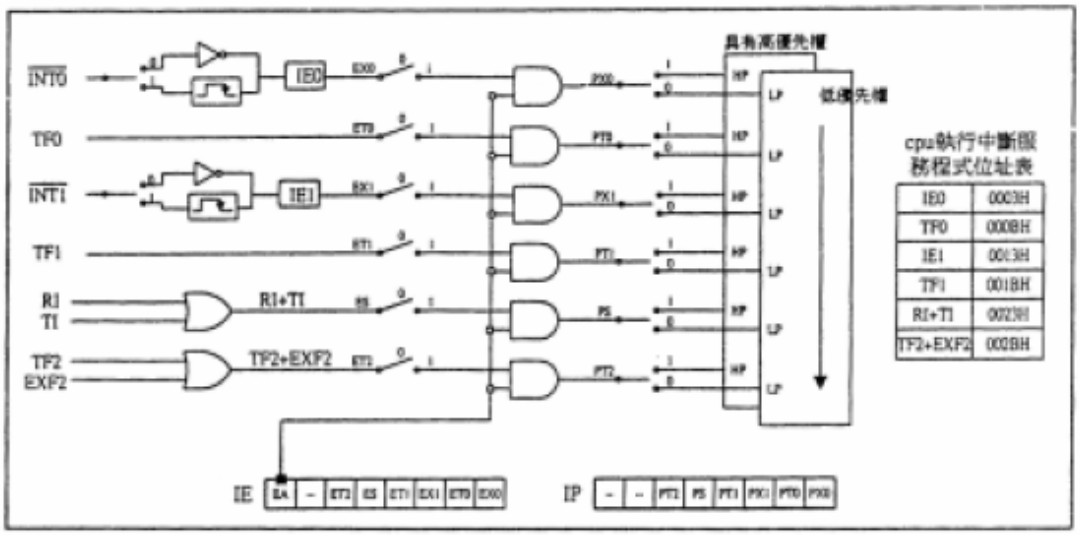


圖 3-9 MCS-51中斷結構邏輯控制圖

中斷來源	中斷要求旗號	所屬暫存器	中斷向量位址
外部中斷 $\overline{INT0}$	IE0	TCON: TF1 TR1 TF0 TR0 IE1 IE1 IT1 IE0• IT0	0003H
Timer0	TF0	TCON: TF1 TR1 TF0• TR0 IE1 IE1 IT1 IE0 IT0	000BH
外部中斷 $\overline{INT1}$	IE1	TCON: TF1 TR1 TF0 TR0 IE1• IE1 IT1 IE0 IT0	0013H
Timer1	TF1	TCON: TF1• TR1 TF0 TR0 IE1 IE1 IT1 IE0 IT0	001BH
UART (TXD)	T1	SCON: SM0 SM1 SM2 REN TB8 RB8 T1• RI	0023H
UART (RXD)	R1	SCON: SM0 SM1 SM2 REN TB8 RB8 T1 RI•	
Timer2	TF2	T2CON: TF2• EXF2 RCLK TCLK EXEN2 TR2 C/T2 CP/RL2	002BH
T1EX	EXF2	T2CON: TF2 EXF2• RCLK TCLK EXEN2 TR2 C/T2 CP/RL2	

表 3-3 MCS-51中斷要求旗號一覽表

計時/計數器1的中斷

Timer0和Timer1的中斷是由TF0和TF1旗號所產生的，這兩個旗號是在當它們各自的計時/計數器產生溢位時，會由內部自動設定為1，並提出中斷要求(Timer1在模式3時例外)。此時若計時/計數氣得中斷以啟用(致能)，則CPU會跳往自己本身的中斷向量位址處執行(Timer0是000BH，Timer1是001BH)，並且將旗號(TF0或TF1)清除為0。

外部中斷

外部中斷INT0和INT1，我們可以將其規劃成準位觸發動作(偵測輸入中斷信號是否為LOW)或是規劃成負緣觸發動作，它是由TCON暫存器中的IT0及IT1位元來選擇用那一種觸發方式。另外在TCON暫存器中的IE0及IE1兩位元，才是真正提出中斷要求的兩個旗標。當外部中斷發生時，IE0或IE1會被設定為1，以便向CPU提出中斷要求。

串列埠UART的中斷

串列埠之中斷發生在RI和TI兩位元上，只要這兩個旗標中有任何一個位元為1時，即可配合IE暫存器的ES位元來產生中斷致能，則CPU將會自動跳往串列埠中斷向量位址(0023H)處執行。要注意的是在執行中斷服務程式後，並不會自動清除RI或TI為0，我們必須自己在執行串列埠中斷副程式中，用指令將此位元清除為0。另外還要注意的是，一旦發生中斷跳至串列埠中斷服務程式執行後，由於發送及接收共用一個向量位址(0023H)，因此再中斷副程式的開頭，就必須先用指令去判斷究竟是何者提出中斷要求的(判斷RI或TI)。

計時/計數器2的中斷

在8051中，當TF2或EXF2二者中任一為1時，都將發生計時/計數器2的中斷要求動作。中斷發生後會跳往本身的中斷向量(002BH)位址處去執行，執行後並不會自動將TF2或EXF2兩旗標清除為0。因此程式設計者必須用指令，在中斷副程式中，將該位元清除為0。且由於兩者共用同一個中斷向量位址(002BH)，因此在中斷副程式中，我們必須使用指令去判斷究竟是何者要求中斷(判斷是TF2或EXF2)。

中斷優先權

MCS-51的中斷優先權層次共有兩層，即高優先權(High Priority)及低優先權(Low Priority)。每一個中斷源都可經由IP暫存器中的位元，規劃為高優先權或低優先權，以設定整個程式的中斷優先權順序。若相對應的位元設定為1時，則在高優先權層次；反之若設定為0，則在較低優先權層次。中斷優先權暫存器的說明如下：

IP：中斷優先權暫存器(可位元定址)
位址：B8H

---	---	PT2	PS	PT1	PX1	PT0	PX0

--- IP.7 未使用；保留給將來使用
--- IP.6 未使用；保留給將來使用
PT2 IP.5 定義Timer2之優先權層次(8052)
PS IP.4 定義串列埠之優先權層次
PT1 IP.3 定義Timer1之優先權層次
PX1 IP.2 定義INT1之優先權層次
PT0 IP.1定義Timer0之優先權層次

PX0 IP.0定義INT0之優先權層次

一個低層次的中斷源可以被高層次的中斷源中斷，但是不會被同層次的中斷源所中斷，而一個高層次的中斷源不會被任何其他的中斷源所中斷。如圖3-10所示，如果有兩個不同層次的中斷源同時要求中斷，此時高層次優先權之中斷源優先被執行。如果兩個相同優先權層次的中斷源，同時提出中斷要求時，則CPU將依據內定之輪詢順序安排，已決定何者中斷源被接受執行。如表3-4所示，如果程式設定並沒有設定各個中斷源的優先順序，CPU即自動以表3-4的內定中斷優先權來排定中斷優先順序。

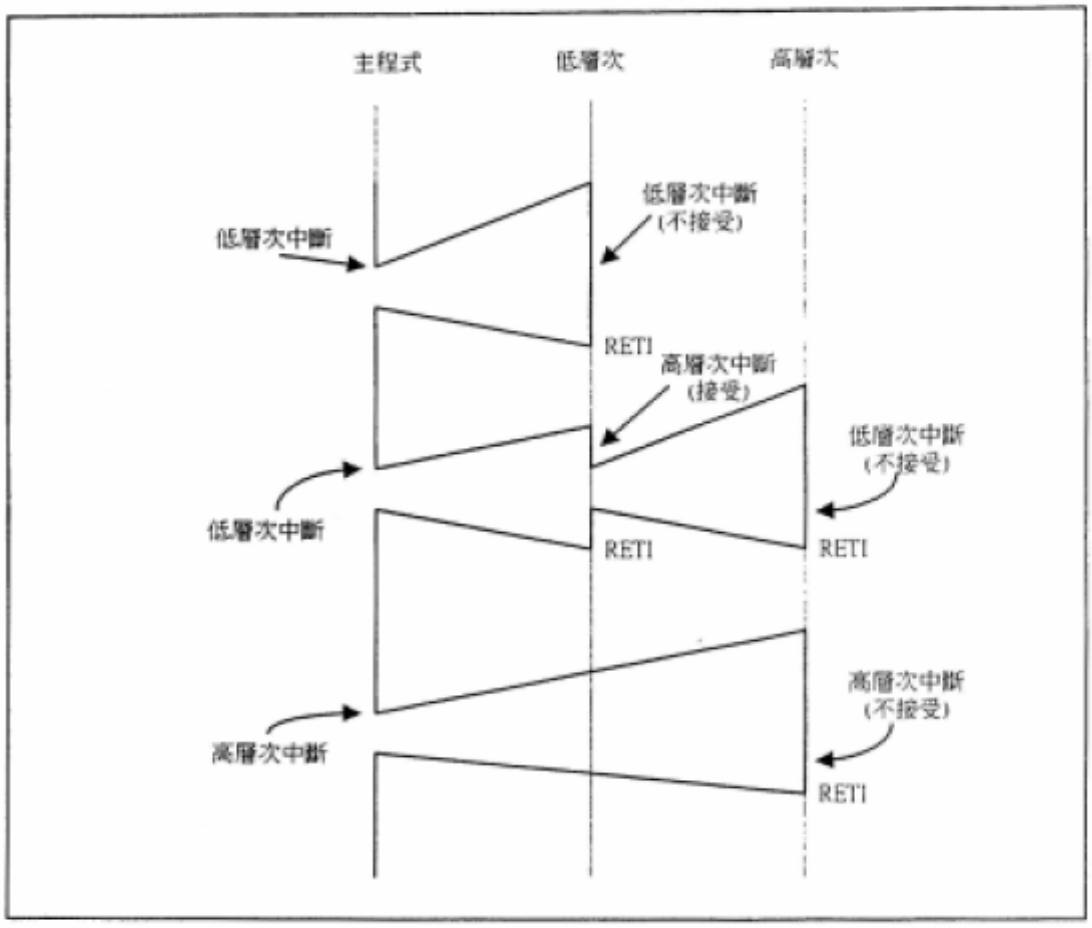


圖 3-10 中斷優先層次執行圖

中斷源	內定優先權層次
1. IE0 2. TF0 3. IE1 4. TF1 5. R1+T1 6. TF2+F2	最 高 最 低

表 3-4 中斷優先權內定優先權表

CPU如何執行中斷

中斷要求旗標在每一個機械週期的S5P2時間會被取樣，如圖3-11所示。

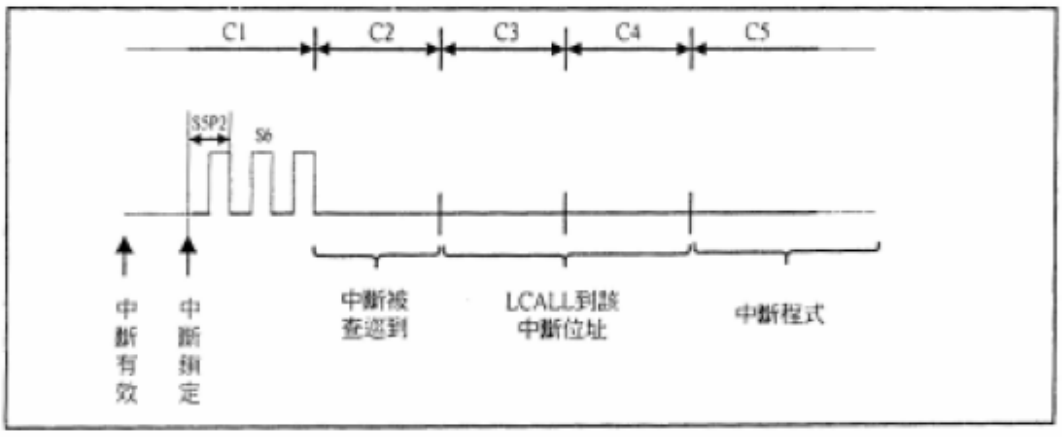


圖 3-11 中斷發生時，CPU反應時序表

被取樣的中斷要求旗標，將在下一個週期時，被輪詢加以檢查，如圖3-11所示之時序C2時。假如此時這些被取樣的中斷要求旗標有一個為1時，則會在這個機械週期裡被認可，然後中斷系統產生一長程呼叫(LCALL)之動作，使目前正在執行的主程式被中

斷，並且將目前的PC推入堆疊區存放，然後程式跳往相關的中斷服務程式處去執行。但是如果有下列情形發生時就不會產生中斷動作：

1. 有相同層次或更高優先權層次的中斷正在執行中。
2. 目前的週期(即中斷系統輪詢及處理的週期C2)並非正在執這個指令的最後一個週期。
3. 正在執行指令RETI；或者正在執行任何寫入IE或IP指令的動作時。

上述的三種情況，都將會阻止CPU產生一長程呼叫LCALL至中斷服務程式。上述之情況2，是要確保目前正在執行的指令會在控制權跳至中斷服務程式前輩完整的執行完。而情況三則是在確保執行完RETI或對IE、IP的指令做過存取的動作後，需再執行一個指令才可以再接受中斷。

有一點要注意的是，如果有一個中斷要求動作了，但卻因上述所說的三個情況的關係而沒有被8051所接受中斷時如果中斷的動作沒有繼續維持下去，則當上述的三種情況解除過後，CPU還是無法接受到中斷要求。由上面分析得知，中斷控制系統的輪詢結果是不會被保留的，每一次的輪詢都是新的。

一旦中斷被確認後，硬體會產生一長程呼叫LCALL的動作，CPU也會把目前程式計數器PC的值，推入到堆疊區存放，然後載入中斷源之中斷向量位址，並跳至此位址執行中斷副程式。一直要等到執行RETI指令後，才表示中斷結束，然後從堆疊區取回(POP)兩個Byte的返回位址到程式計數器PC中，以回到原來主程式被中斷的位址繼續執行。

第四章 系統設計說明

4-1 軟、硬體設計

在這次的專題當中，主要是以控制步進馬達的旋轉動方向、加減速和角度定位。就硬體方面，先焊出一個可以讓步進馬達做基本運轉的簡單電路，後面要求的功能可以陸續加在硬體上。就軟體方面，以 Keil 為主。利用組合語言編輯成的程式皆可以在視窗下完成 compile 及 link 的動作。

硬體方面主要是先將步進馬達的驅動電路完成，如圖 4-1 所示為硬體的府視圖。

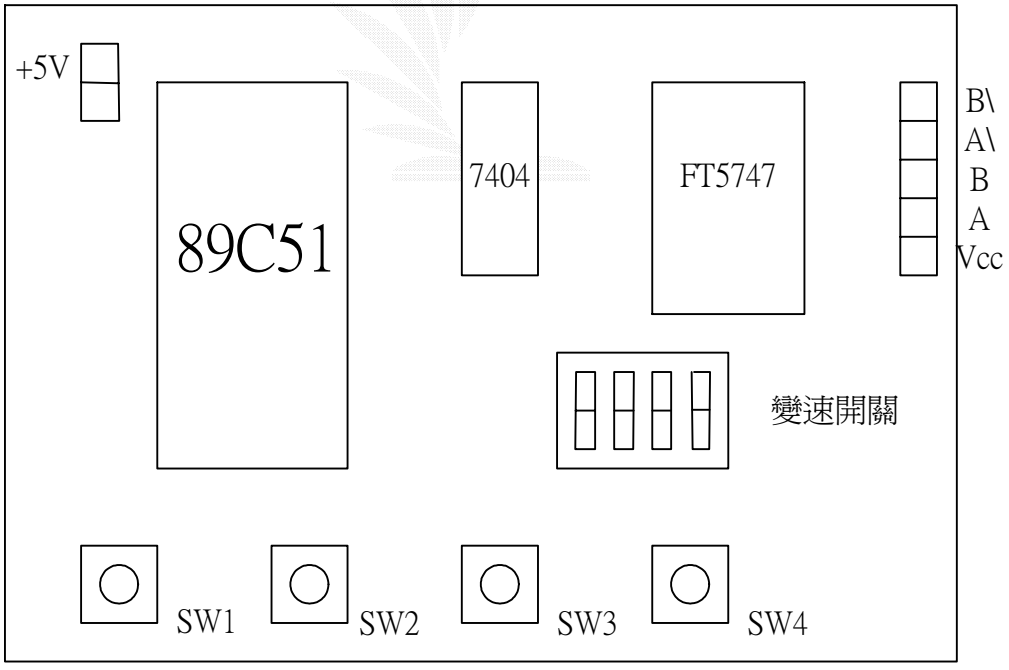


圖 4-1 硬體的府視圖

在圖 4-1 的功能說明將如下表說明：

＋5V	輸入電源 5V。
變數開關	變換步進馬達運轉的速度。
/B, /A, B, A, Vcc	步進馬達線圈的相序順序。
SW1	功能 1/正轉（FORWARD）。
SW2	功能 2/逆轉（REVERSE）。
SW3	功能 3/增加運轉的角度。
SW4	回主功能選單

下圖 4-2 所示即為軟體 Keil 的操作視窗。

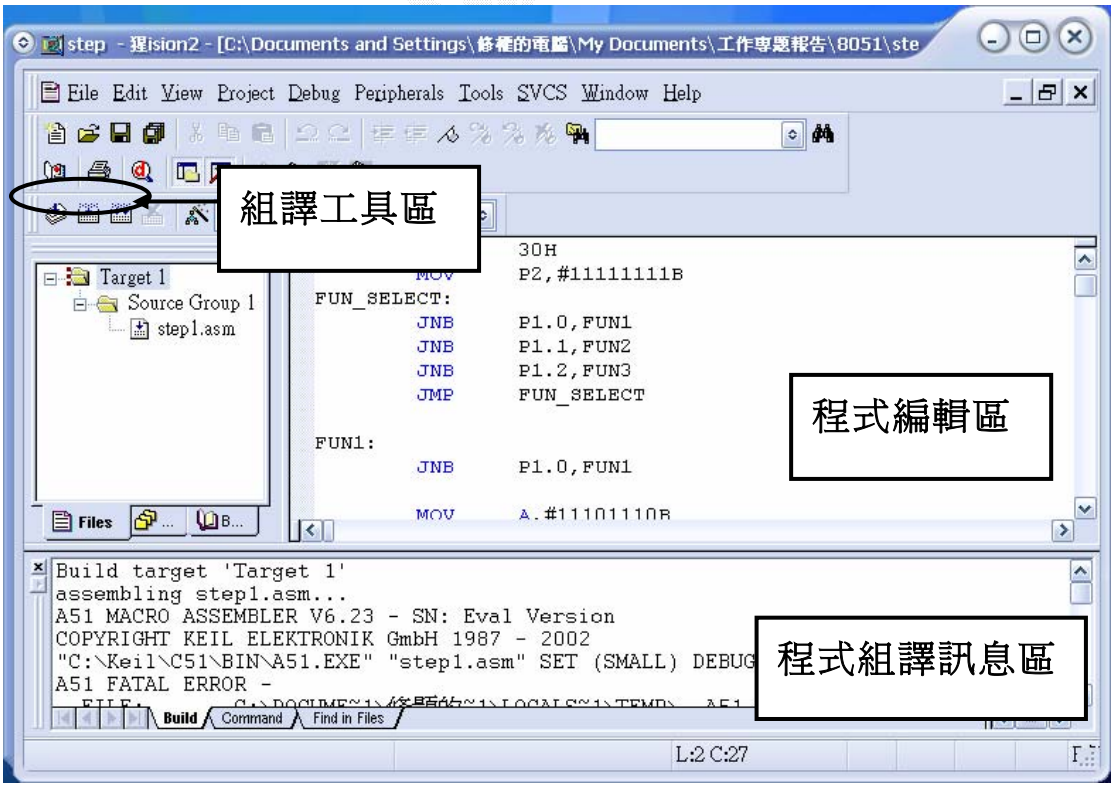


圖 4-2 軟體操作視窗

4-2 硬體電路及程式說明

本節內容主要是電路圖及寫一個簡單的程式測試硬體是否可以完成基本功能。如圖 4-3 所示，為本次實作的電路圖。如圖 4-4 所示，為一個步進馬達基本的驅動電路。

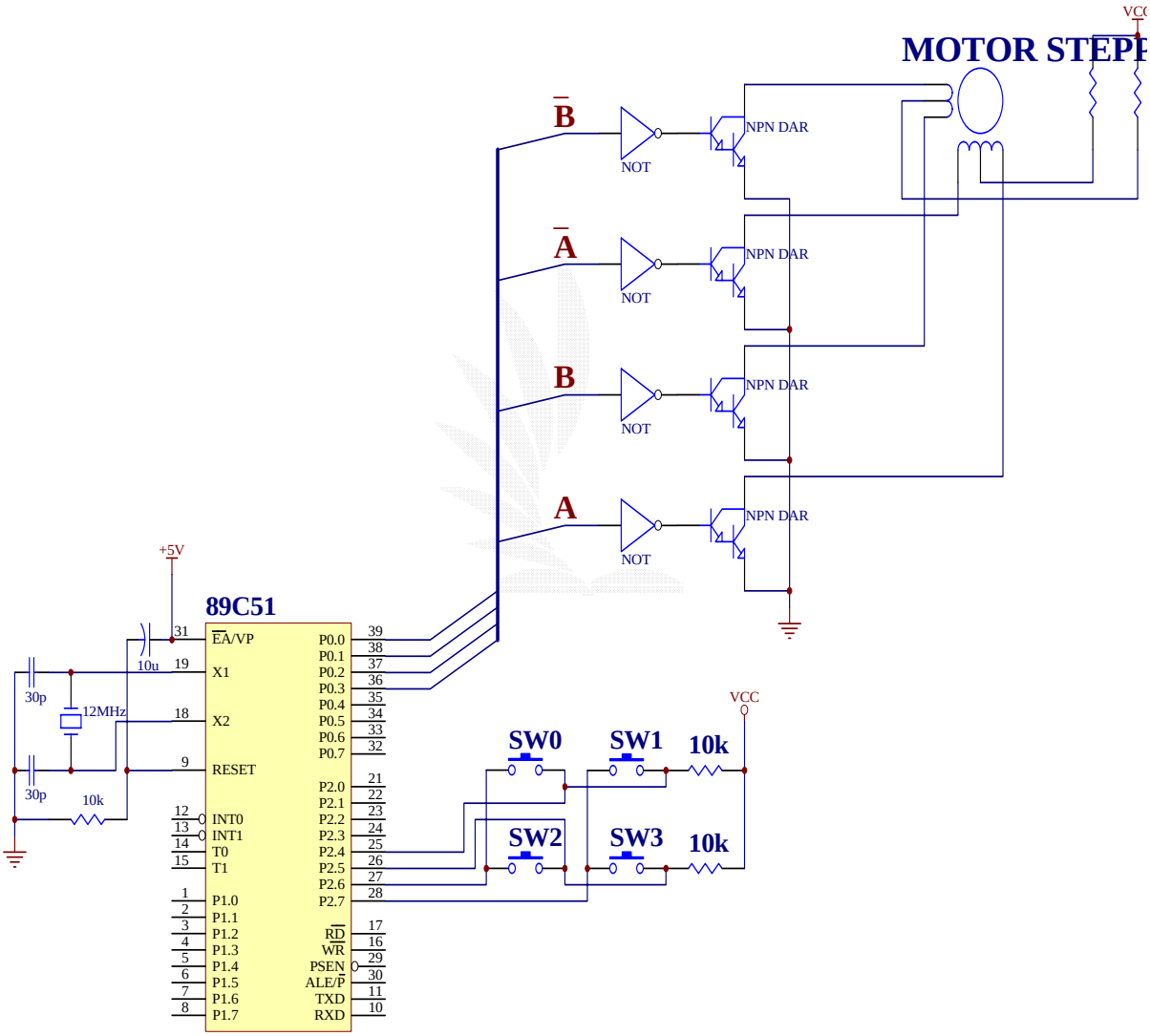


圖 4-3 電路圖

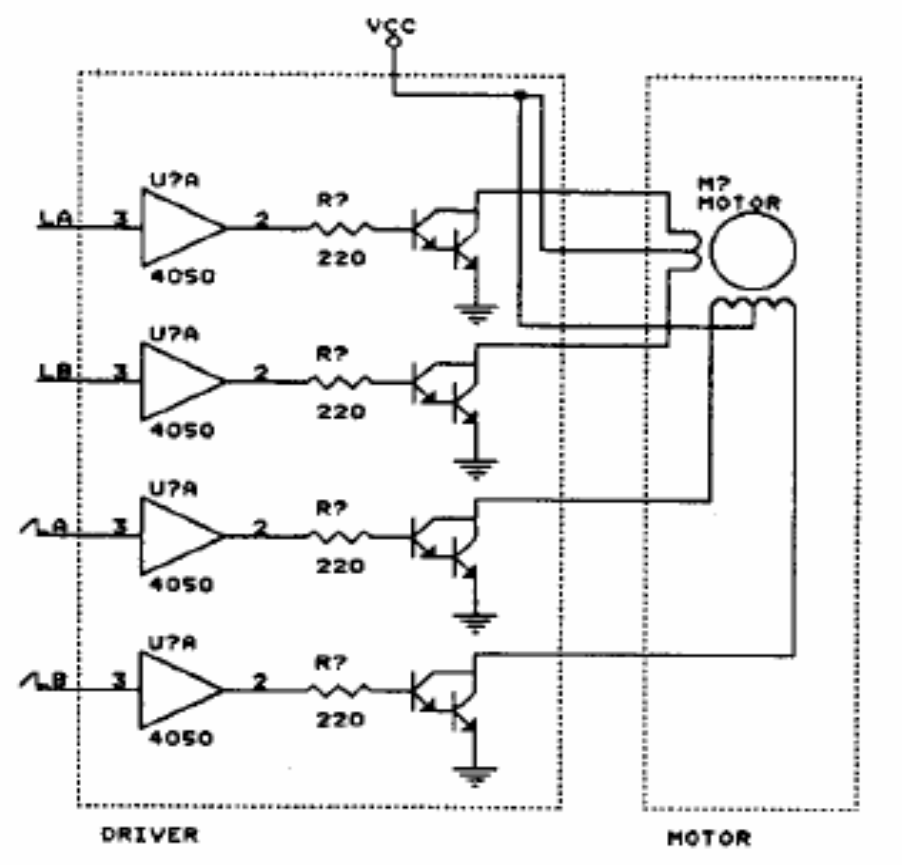


圖 4-4 步進馬達驅動電路圖

就 89C51 的 I/O 配置如下所示：

```
B\      BIT    P0.0          ;指定 P0.0 為 B\
A\      BIT    P0.1          ;指定 P0.1 為 A\
B       BIT    P0.2          ;指定 P0.2 為 B
A       BIT    P0.3          ;指定 P0.3 為 A
```

如圖 4-5 所示，為 89C51 讀取外部按鈕時的電路設計。

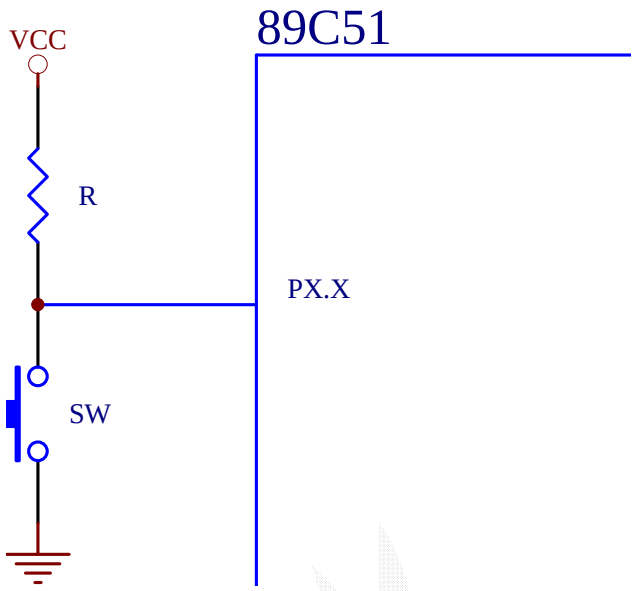


圖 4-5 外部按鈕電路設計圖

電路原理

當 SW 開關若未按下，89C51 的 I/O 將會判讀為 1；若是 SW 開關按下後 89C51 的 I\O 將會判讀為 0。

程式技巧

1. IO 的配置

```
SW      BITPX.X
```

2. SW 開關動作產生訊號的讀取

```
JNBSW,ADDR
或
JB      SW,ADDR
```

如圖 4-6 所示，為步進馬達變速運轉的電路設計。

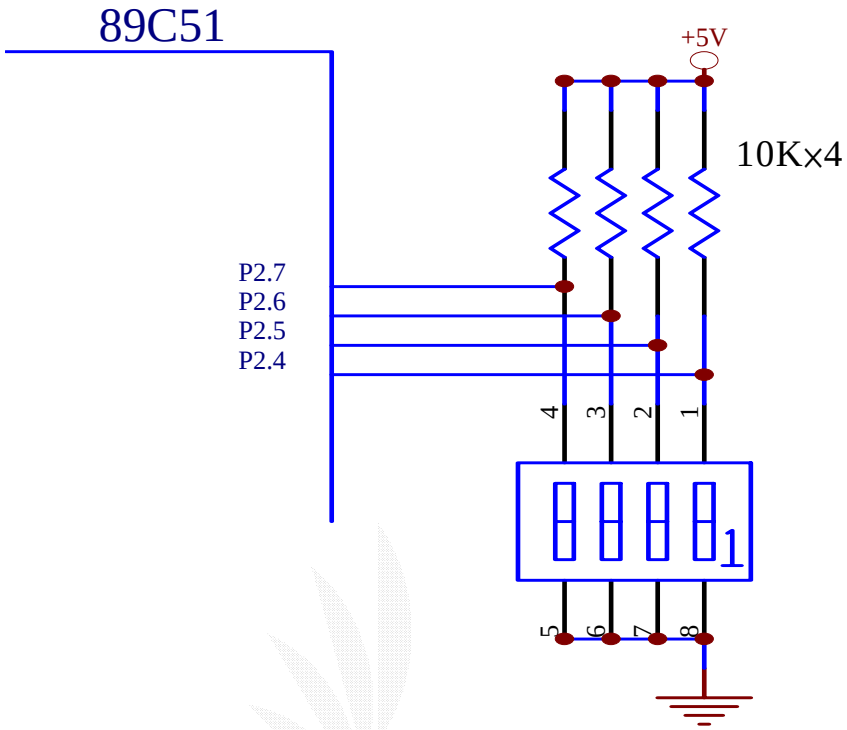


圖 4-6 步進馬達變速運轉的電路設計圖

電路原理

由 P2 來讀取外部指撥開關的值，主要是為了藉由指撥開關的值來改變 P0 輸出訊號的時間，也就是改變步進馬達的運轉速度。

程式技巧

1. 利用間接定址法

```
MOV R6,P2
```

可以將 P2 讀取指撥開關的值存了暫存器 R6。

2. 呼叫副程式就可以改變 P0 輸出的時間。

4-3 程式流程圖

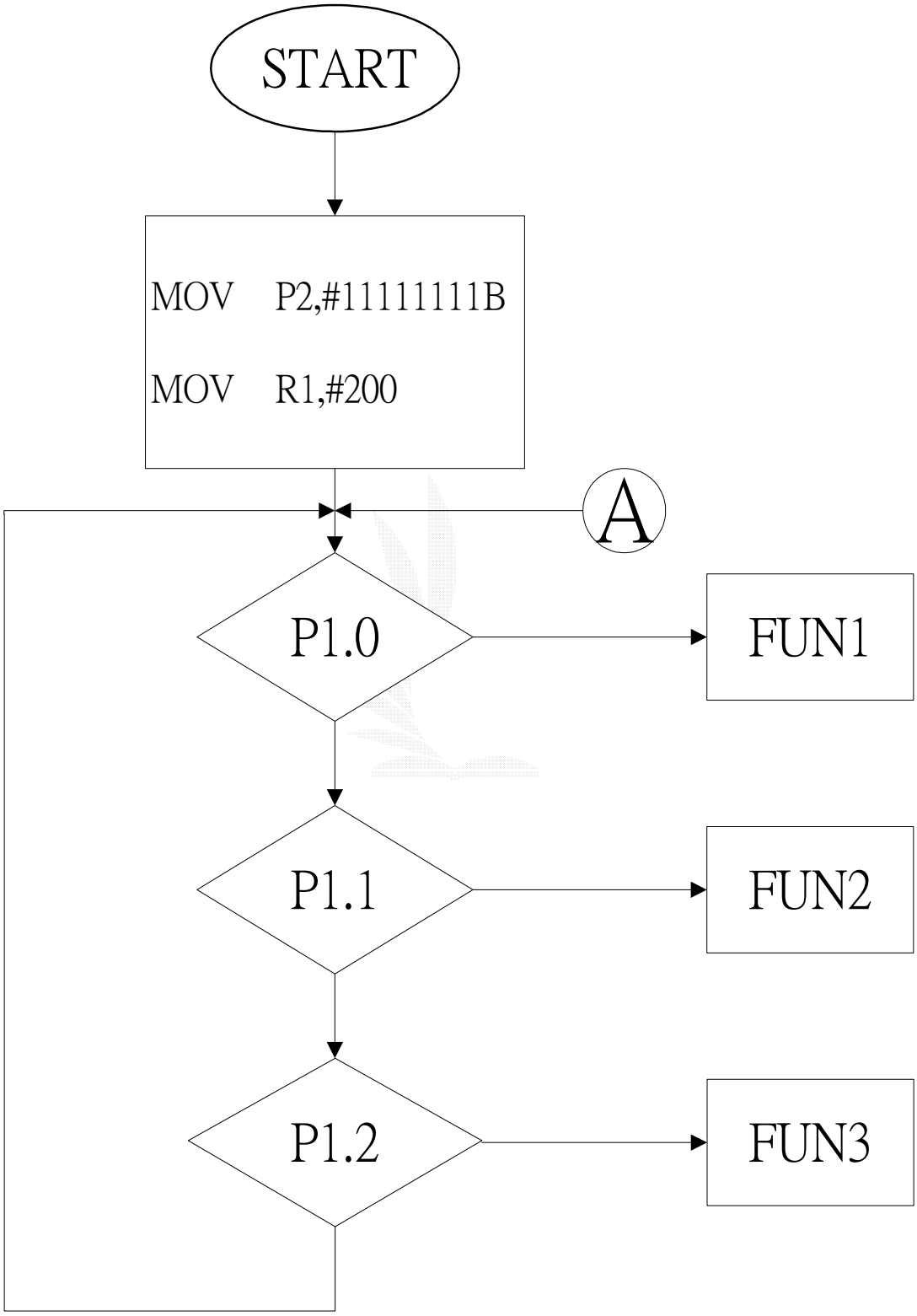


圖 4-7 程式流程圖

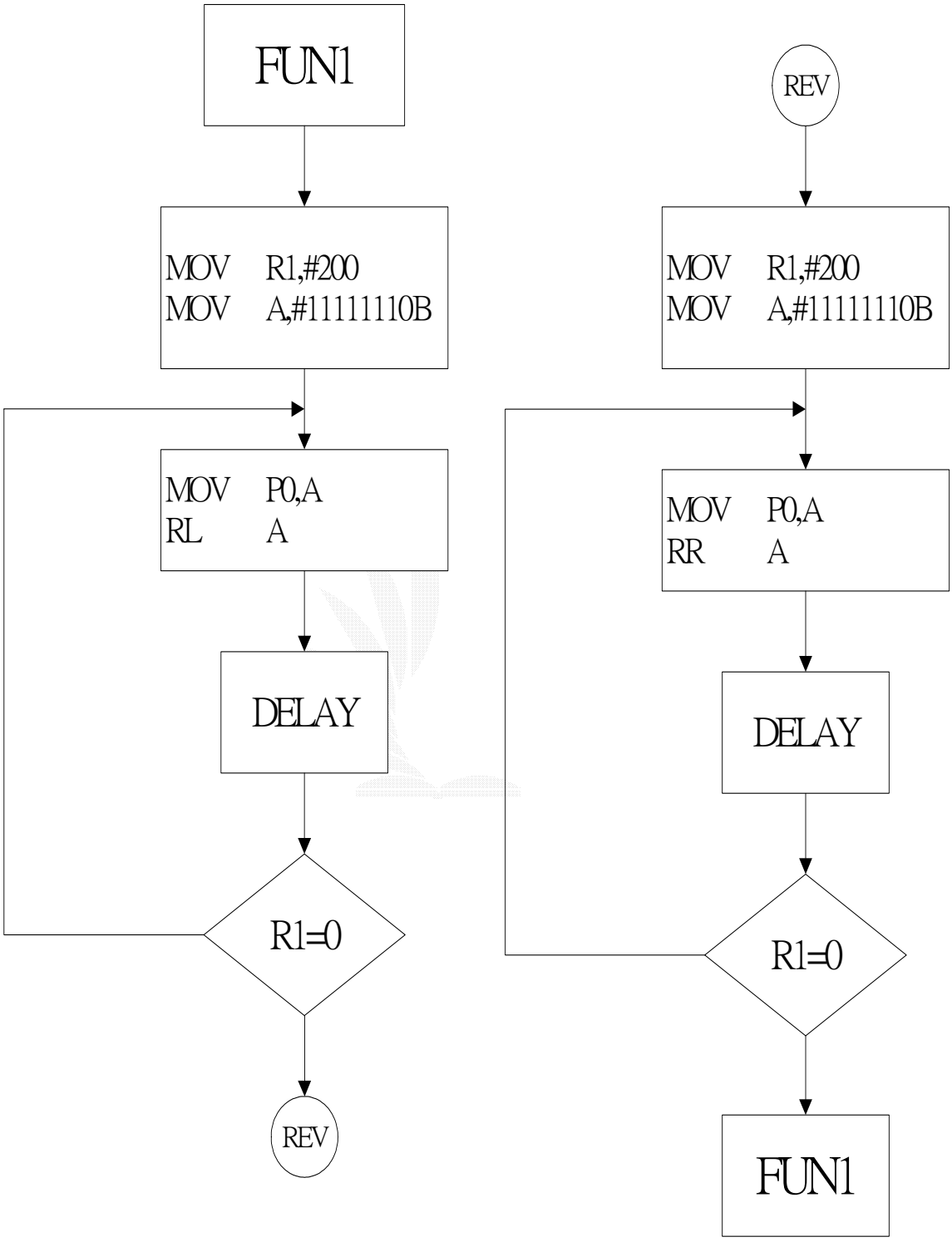

FUN1
MOV R1,#200
MOV A,#11111110B
MOV P0,A
RL A
DELAY
R1=0
REV
REV
MOV R1,#200
MOV A,#11111110B
MOV P0,A
RR A
DELAY
R1=0
FUN1

圖 4-8 FUN1 流程圖

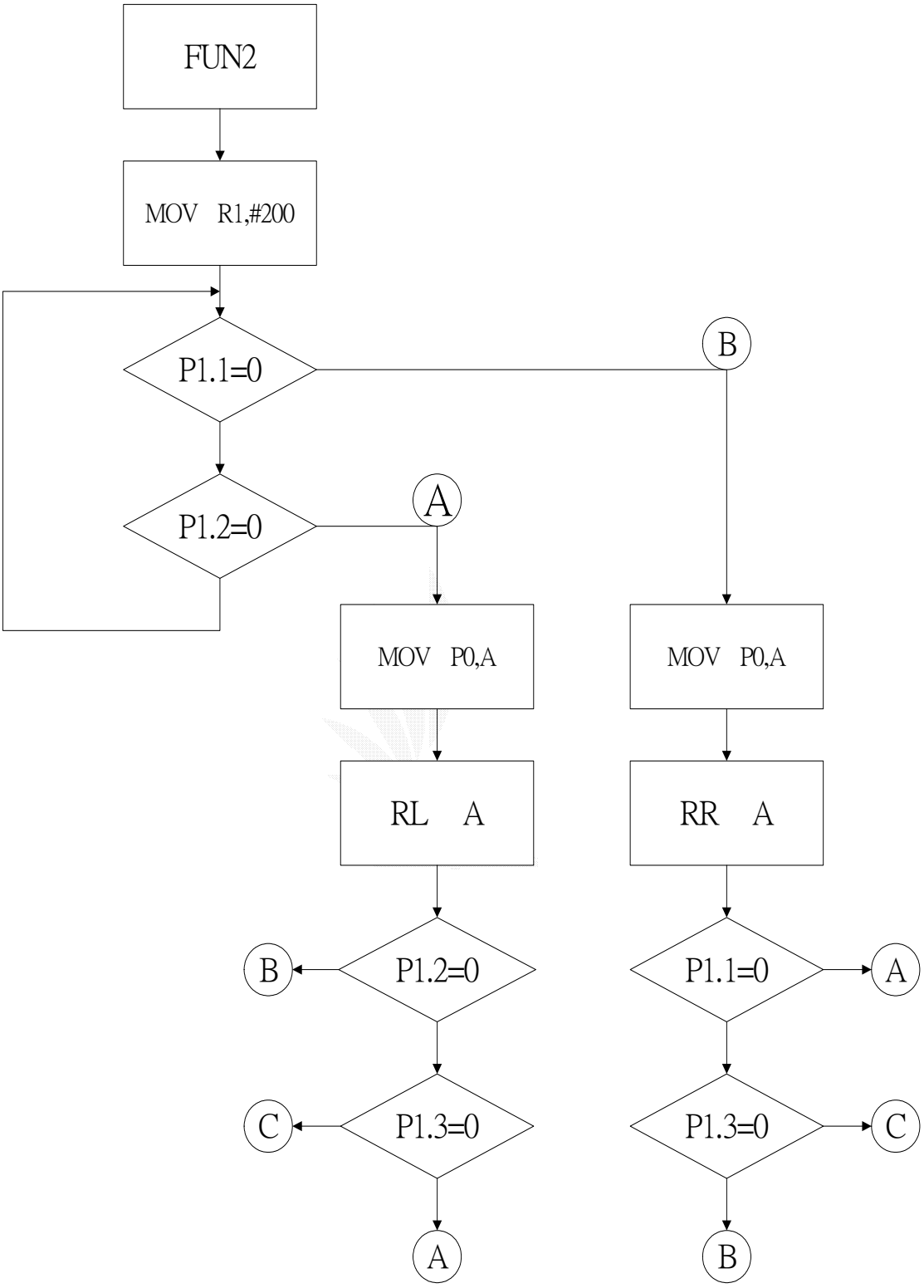

FUN2
MOV R1,#200
P1.1=0
B
P1.2=0
A
MOV P0,A
MOV P0,A
RL A
RR A
B
P1.2=0
P1.1=0
A
C
P1.3=0
P1.3=0
C
A
B

圖 4-9 FUN2 流程圖

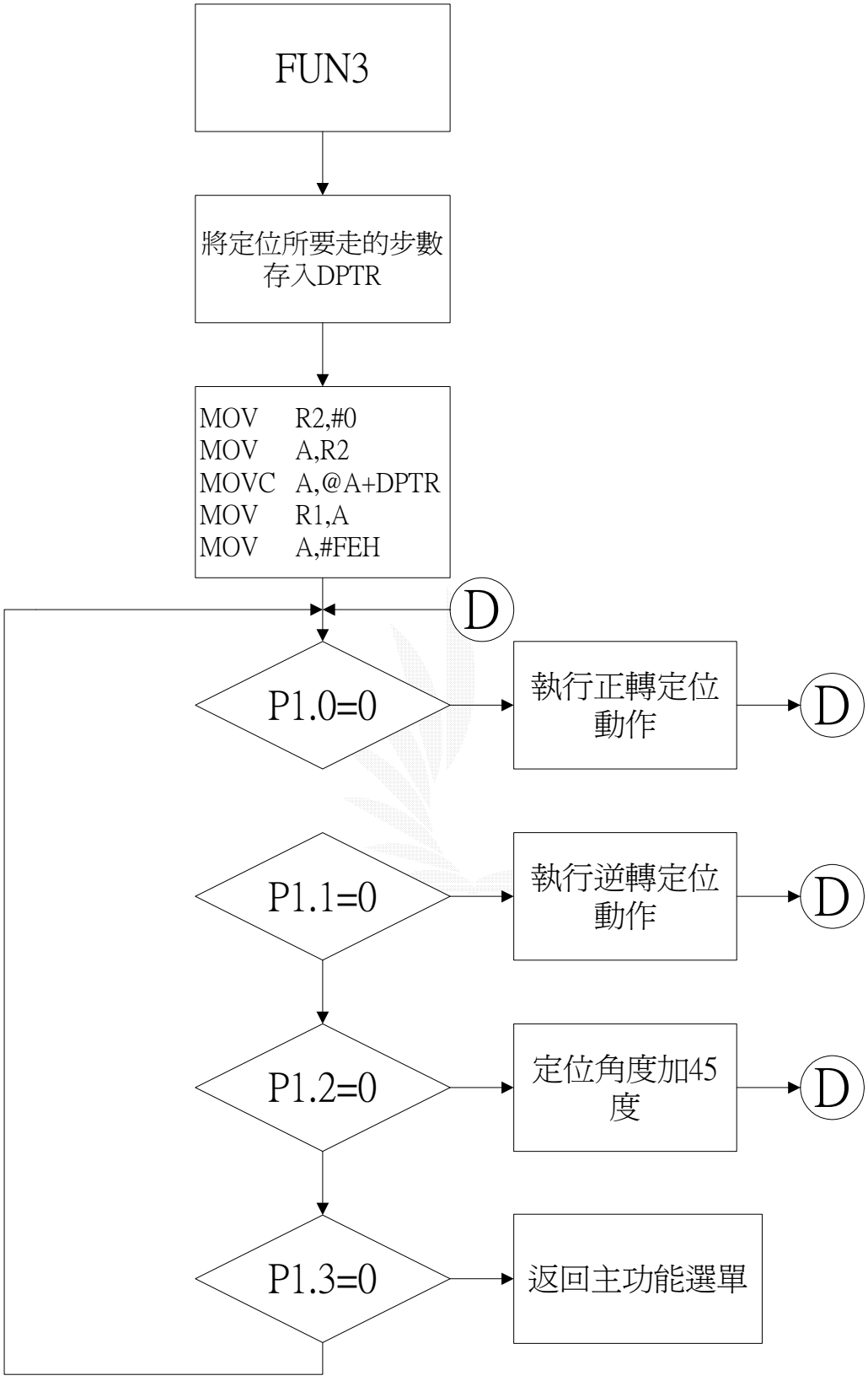


圖 4-10 FUN3 流程圖

4-4 程式碼

```
;================================================
;程式開始
;================================================
        ORG     00H
        JMP     FUN_SELECT
        ORG     30H
        MOV     P2,#11111111B
;================================================
;FUN1：正轉一圈，逆轉一圈。
;FUN2：控制正、逆轉方向。
;FUN3：角度定位控制。
;================================================
FUN_SELECT:
        JNB     P1.0,FUN1           ;選擇功能一
        JNB     P1.1,FUN2           ;選擇功能二
        JNB     P1.2,FUN3           ;選擇功能三
        JMP     FUN_SELECT
;================================================
;功能一：正轉一圈，逆轉一圈。
;================================================
FUN1:
        JNB     P1.0,FUN1           ;檢查是否仍按住
        MOV     A,#11101110B
        MOV     R1,#200             ;設定一圈的步數
FOR:
        MOV     P0,A
        RL      A
        JNB     P1.3,FUN_SELECT     ;是否跳回主選單
        CALL    DELAY
        DJNZ    R1,FOR              ;判別是否正轉一圈
```

```
        CALL    HOLD            ;正逆轉切換延遲
        MOV     R1,#200         ;重新設定步數
REV:
        MOV     P0,A
        RR      A
        JNB     P1.3,FUN_SELECT ;是否跳回主選單
        CALL    DELAY
        DJNZ    R1,REV          ;判別是否逆轉一圈
        CALL    HOLD            ;正逆轉切換延遲
        MOV     R1,#200         ;重新設定步數
        JMP     FOR             ;回到正轉部分
;================================================
;功能二：控制正、逆轉方向。
;================================================
FUN2:
        JNB     P1.1,FUN2       ;檢查是否仍按住
FUN2_LOOP:
        MOV     A,#11101110B
        MOV     R1,#200
CONTROL:
        JNB     P1.0,FOR        ;執行正轉
        JNB     P1.1,REV        ;執行逆轉
        JNB     P1.3,FUN_SELECT ;返回主選單
        JMP     CONTROL
FOR1:
        MOV     P0,A
        RL      A
        JNB     P1.2,CONTROL    ;馬達停止運轉
        CALL    DELAY
        DJNZ    R1,FOR1
        MOV     R1,#200
```

```
        JMP     FOR1
REV1:
        MOV     P0,A
        RR      A
        JNB     P1.2,CONTROL      ;馬達停止運轉
        CALL    DELAY
        DJNZ    R1,REV1
        MOV     R1,#200
        JMP     REV1
;================================================
;功能三：角度定位控制。
;================================================
FUN3:
        JNB     P1.2,FUN3
FOR2:
        CALL    HOLD
FFOR:
        MOV     P0,A
        RL      A
        JNB     P1.2,REV2
        CALL    DELAY
        DJNZ    R1,FFOR
        MOV     R1,#200
        JMP     FOR2
REV2:
        CALL    HOLD
RREV:
        MOV     P0,A
        RR      A
        JNB     P1.1,FOR2
        CALL    DELAY
```

```
        DJNZ    R1,RREV
        MOV     R1,#200
        JMP     REV2
;================================================
;設定可改變的延遲時間副程式。
;================================================
DELAY:
        MOV     R6,P2                   ;讀取外部資料
DL1:
        MOV     R7,#150
        DJNZ    R7,$
        DJNZ    R6,DL1
        RET
;================================================
;正、逆轉切換的延遲時間副程式。
;================================================
HOLD:
        MOV     R6,#100
DL2:
        MOV     R7,#100
        DJNZ    R7,$
        DJNZ    R6,DL2
        RET
;================================================
;程式結束
;================================================
        END
```

第五章 心得感想

在本次專題實作過程中，讓我有所領悟的就是人際關係。從一開始的選定題目、找尋參考書目、選購材料，都和幾位同學一起行動，而在這個專業領域裡面其實本身的能力有限，所以也請教了幾位同學，幸虧他們的鼎力相助我才能完成。這項作業特別要感謝的是我的高中同學，從硬體的製作到軟體撰寫的理解他都幫了我很大的忙。

其實在做任何事情都會遇上困難，但是只要勇敢的面對努力找尋解決之道方能突破。雖然本次實作習得的專業知識並不多，但我學習到做事的道理，也希望在未來的日子裡我也可以像這次做專題的精神---勇往向前。

參考資料

[1].陳熹編譯，步進馬達應用技術，全華圖書，1987

[2].陸志誠著，8051單晶片微電腦原理與實習，碁峰資訊，1995

[3].鍾自立、張正賢著，8051 實作與燒錄器製作，宏友書局，2000

[4].陳龍三著，8051 入門與介面控制，松崗書局，1999

[5].李齊雄、游國幹著，8051 單晶片微電腦原理與實作，儒林書局，1995

[6].鄧錦城編著，8051 單晶片入門與實作，宏友圖書公司出版，1998

附錄 A

MCS-51 指令集

算數運算指令

指令	說明	位元組	工作週期（時脈數）
ADD A，Rn	暫存器累加至累加器	1	12
ADD A，direct	直接位元組加至累加器	2	12
ADD A，@Ri	間接位元組加至累加器	1	12
ADD A，#data	常數值加至累加器	2	12
ADDC A，Rn	與C一起將暫存器加至累加器	1	12
ADDC A，direct	與C一起將直接位元組加至累加器	2	12
ADDC A，@Ri	與C一起將間接位元組加至累加器	1	12
ADDC A，#data	與C一起將常數值加至累加器	2	12
SUBB A，Rn	累加器減暫存器再減C	1	12
SUBB A，direct	累加器減直接位元組再減C	2	12
SUBB A，@Ri	累加器減間接位元組再減C	1	12
SUBB A，#data	累加器減常數值再減C	2	12
INC A	累加器加一	1	12
INC Rn	暫存器加一	1	12
INC direct	直接位元組加一	2	12
INC @Ri	間接位元組加一	1	12
DEC A	累加器減一	1	12
DEC Rn	站存器減一	1	12
DEC direct	直接位元組減一	2	12
DEC @Ri	間接位元組減一	1	12
INC DPTR	資料指標加一	1	24
MUL AB	A 乘以B	1	48
DIV AB	A 除以B	1	48
DA A	累加器作BCD 調整	1	12

邏輯運算指令

指令	說明	位元組	工作週期（時脈數）
ANL A，Rn	暫存器AND 至累加器	1	12
ANL A，direct	直接位元組AND 至累加器	2	12
ANL A，@Ri	間接位元組AND 至累加器	1	12
ANL A，#data	常數值AND 累加器	2	12
ANL direct，A	累加器AND 至直接位元組	2	12
ANL direct，#data	常數AND 至直接位元組	3	24
ORL A，Rn	暫存器 OR 至累加器	1	12
ORL A，direct	直接位元組OR 至累加器	2	12
ORL A，@Ri	間接位元組OR 至累加器	1	12
ORL A，#data	常數值加OR 累加器	2	12
ORL direct，A	累加器OR 至直接位元組	2	12
ORL direct，#data	常數OR 至直接位元組	3	24
XRL A，Rn	暫存器XRL 至累加器	1	12
XRL A，direct	直接位元組XRL 至累加器	2	12
XRL A，@Ri	間接位元組XRL 至累加器	1	12
XRL A，#data	常數值加XRL 累加器	2	12
XRL direct，A	累加器XRL 至直接位元組	2	12
XRL direct，#data	常數XRL 至直接位元組	3	24
CLR A	清除累加器	1	12
CPL A	累加器反相	1	12
RL A	累加器向左旋轉	1	12
RLC A	累加器與C 一起向左旋轉	1	12
RR A	累加器向右旋轉	1	12
RRC A	累加器與C 一起向右旋轉	1	12
SWAP A	累加器的高低四位元交換	1	12

資料轉移指令

指令	說明	位元組	工作週期（時脈數）
MOV A，Rn	暫存器內容移至累加器	1	12
MOV A，direct	直接位元組內容移至累加器	2	12
MOV A，@Ri	間接位元組內容移至累加器	1	12
MOV A，#data	常數值移至累加器	2	12
MOV Rn，A	累加器內容移至暫存器	1	12
MOV Rn，direct	直接位元組內容移至暫存器	2	24
MOV Rn，#data	常數值移至暫存器	2	12
MOV direct，A	累加器內容移至直接位元組	2	12
MOV direct，Rn	暫存器內容移至直接位元組	2	24
MOV direct,direct	直接位元組內容移至直接位元組	3	24
MOV direct，@Ri	間接位元組內容移至直接位元組	2	24
MOV direct，#data	常數移至直接位元組	3	24
MOV @Ri，A	累加器內容移至間接位元組	1	12
MOV @Ri，direct	直接位元組內容移至間接位元組	2	24
MOV @Ri，#data	常數移至間接位元組	2	12
MOV DPTR，#data 16	16 位元常數移至資料指標	3	24
MOVC A，@A+DPTR	程式記憶體的資料移入累加器	1	24
MOVC A，@A+PC	程式記憶體的資料移入累加器	1	24
MOVX A，@Ri	外部RAM的資料移入累加器(8位元定址)	1	24
MOVX A，@DPTR	外部RAM 的資料移入累加器(16位元定址)	1	24
MOVX @Ri，A	累加器內容寫到外部RAM(8位元位址)	1	24
MOVX @DPTR，A	累加器內容寫到外部RAM(16位元位址)	1	24

PUSH direct	直接位元組內容放至堆疊區	2	24
POP direct	從堆疊區拿回資料至直接位元組	2	24
XCH A，Rn	累加器與暫存器的內容互換	1	12
XCH A，direct	累加器與直接位元組的內容互換	2	12
XCH A，@Ri	累加器與間接位元組的內容互換	1	12
XCHD A，@Ri	累加器與間接位元組的低四位元互換	1	12

布林變數指令

指令	說明	位元組	工作週期（時脈數）
CLR C	清除進位旗標	1	12
CLR bit	清除bit	2	12
SETB C	設定進位旗標	1	12
SETB bit	設定bit=1	2	12
CPL C	進位旗標反相	1	12
CPL bit	bit 反相	2	12
ANL C，bit	bit AND 至進位旗標	2	24
ANL C，/bit	bit 反相後再AND 至進位旗標	2	24
ORL C，bit	bit OR 至進位旗標	2	24
ORL C，/bit	bit 反相後OR 至進位旗標	2	24
MOV C，bit	bit 之狀態移至進位旗標	2	12
MOV bit，C	進位旗標之狀態移至bit	2	24
JC rel	若C=1 就跳躍	2	24
JNC rel	若C=0 就跳躍	2	24
JB bit，rel	若bit=1 就跳躍	3	24
JNB bit，rel	若bit=0 就跳躍	3	24
JBC bit，rel	若bit=1 就跳躍，且清除此位元	3	24

程式分控指令

指令	說明	位元組	工作週期（時脈數）
ACALL addr11	絕對式副程式呼叫	2	24
LCALL addr16	遠程副程式呼叫	3	24
RET	從副程式返回	1	24
RETI	從中斷副程式返回	1	24
AJMP addr11	絕對式跳躍	2	24
LJMP addr11	遠程跳躍	3	24
SJMP rel	短程跳躍	2	24
JMP @A+DPTR	間接跳躍	1	24
JZ rel	若A=0 就跳躍	2	24
JNZ rel	若A≠0 就跳躍	2	24
CJNE A，direct，rel	若累加器與直接位元組內容不相等就跳躍	3	24
CJNE A，#data，rel	若累加器內容不等於data 就跳躍	3	24
CJNE Rn，#data，rel	若暫存器內容不等於data 就跳躍	3	24
CJNE @Ri，#data，rel	若間接位元組內容不等於data 就跳躍	3	24
DJNZ Rn，rel	暫存器內容減一，若不等於零就跳躍	2	24
DJNZ direct，rel	直接位址內容減一，若不等於零就跳躍	3	24
NOP	沒動作	1	12