國立臺灣科技大學
電子工程系
嵌入式系統設計實習

時鐘與計算器

指導老師：陳　武　田　老師

實習組別：第　十　三　組

實習組員：四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

報告日期：民國九十八年三月二十九日

§～catalog～§

四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

1. 實習目的：

使用 8051 實現一個數字時鐘及計算機整合的裝置，應用 8051 內的「計時／計數器」中斷功能計數時鐘計時功能。

另外使用 4X4 矩陣數字鍵盤透過程式控制，執行計算機功能及時間設定功能，最後輸出至 LCD 液晶顯示器內。

每項功能與功能間均有 10 秒的 Time-out 閒置時間切換，以達到閒置十秒後回到時間顯示。

透過本次實習可充分了解：

1.1 計時／計數器中斷之應用

1.2 按鍵掃描之原理

1.3 LCD 液晶顯示器之控制

1.4 程式流程設計與規劃

2. 實習說明：

以下將本實習所須達成之目標功能，以模式劃分如下：

2.1 Time-out 模式：

2.1.1 顯示當前時間。

2.2 時間設定模式：

2.2.1 Time-out 模式下按 F（SET）鍵可設定時間，時間設定模式可直接按數字鍵設定，閃爍數字爲當前可設定數字，初始設定位置爲小時的十位，每按一次數字設定完後，會自動往右，依此類推。

2.2.2 A（十）鍵、B（一）鍵可以控制欲設定數字的左右調整，A 爲向左，B 爲向右。

四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

2.2.3 在時間設定模式下閒置（未按下任何按鈕）達到十秒自動切換回閒置模式，顯示目前時間。

2.3　計算機模式：

2.3.4 Time-out 模式下按下 SET 以外之任意鍵即可進入計算機模式。

2.3.5 可執行基本的加減乘除運算，並顯示其結果。

2.3.6 在計算機模式下閒置（未按下任何按鈕）達十秒自動切換回閒置模式，顯示目前時間。

3. 實習原理：

3.1　8051 功能與結構：

3.1.1　　8051 的基本功能特性

（1） 8 位元 CPU。

（2） 32 條雙向可獨立定址的 I/O 埠。

（3） 4K 程式記憶體(ROM)，外部可擴充至 64K；

（4） 128byte 資料記憶體(RAM)，外部可擴充至 64K

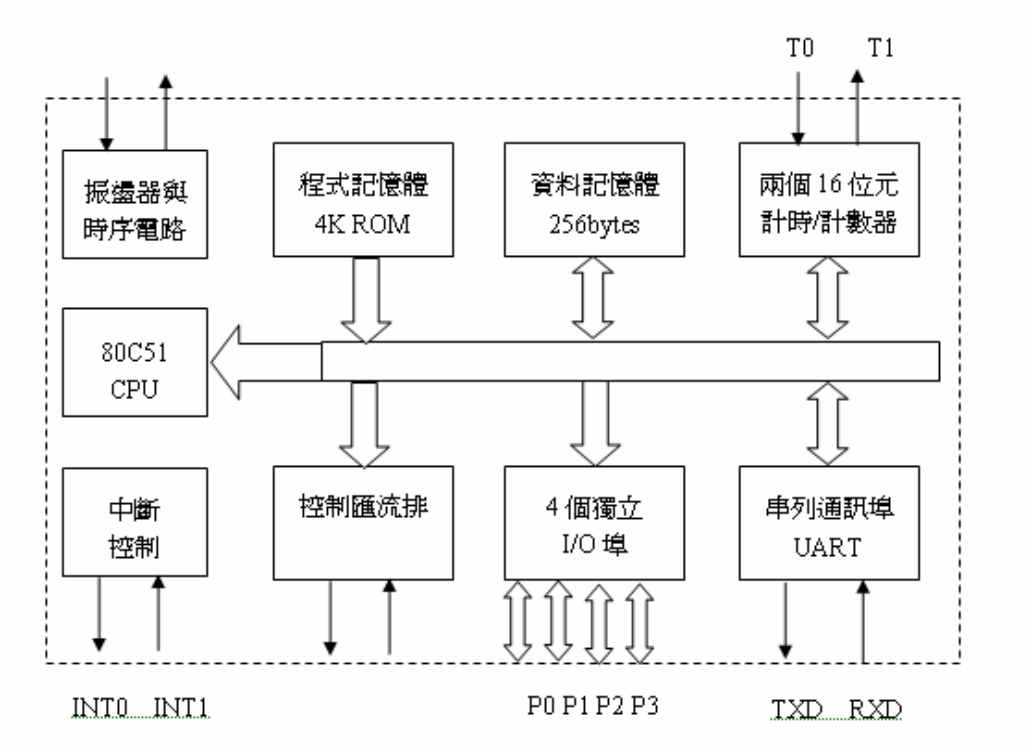
（5） 2 個 16 位元計時/計數器, 5 個中斷源，

（6） 全雙工的串列通訊埠(UART)

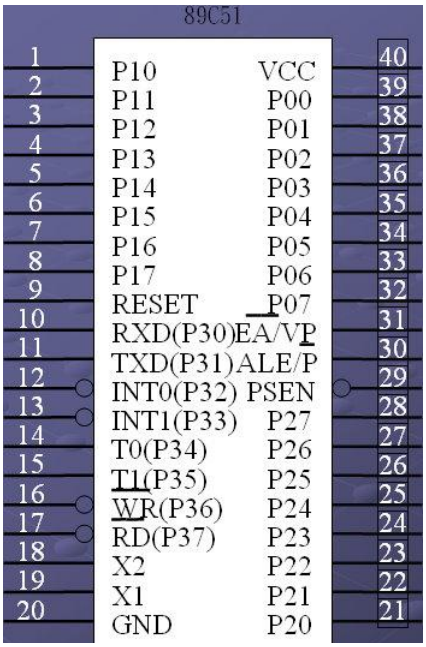
（7） 具有布林運算能力。

四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

3.1.2 8051 內部結構圖：



3.1.3　　8051 接腳電路：



四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

3.1.4　8051 內部組成：

（1） 中央處理單元(CPU)。

（2） 內部程式記憶體(ROM)-4KB。

（3） 內部資料記憶體(RAM)-256Bytes。

（4） 振盪與時序電路(12MHZ)。

（5） I/O 埠(P0,P1,P2,P3)。

（6） 計時/計數器。

（7） 中斷控制電路。

（8） 串列通訊 UART

3.1.5　一般通用暫存器：

（1） ACC：最重要的暫存器，運算與資料轉移都透過 ACC

（2） PC :程式計數器, 記載著程式下一個待執行指令位址。

（3） B 暫存器：用於乘法，除法指令的輔助暫存器。

（4） PSW 程式狀態字組：記錄程式運作時，CPU 各種狀態。

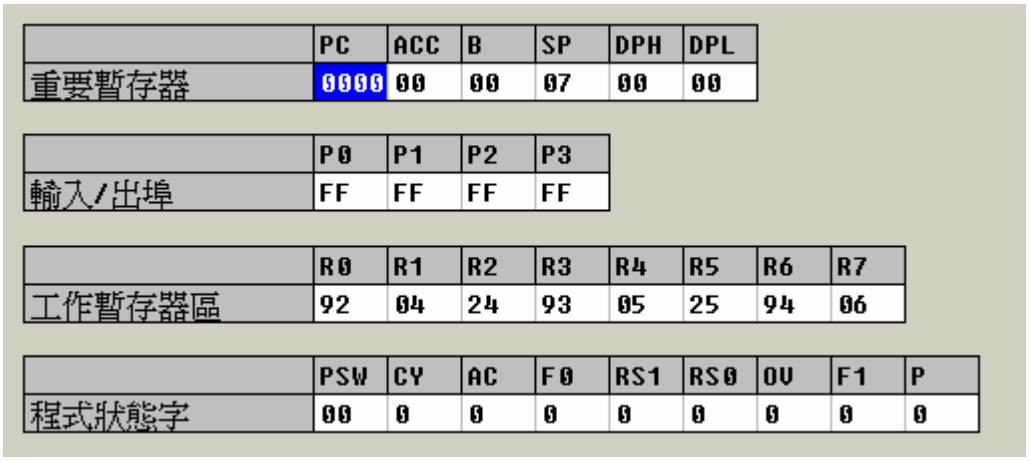
（5） SP 堆疊指標：重置(RESET)時，堆疊指標設爲 07H

（6） DPTR 資料指標暫存器 16 位元暫存器。由 DPH，DPL 兩個 8 位元暫存器組成。

（7） 工作暫存器：共有 RB0、RB1、RB2、RB3 四組工作暫存器庫。每個暫存器庫有 8 個 8 位元暫存器，分別爲 R0、R1、R2、R3、R4、R5、R6、R7。

四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

3.1.6 暫存器結構圖：



	PC	ACC	B	SP	DPH	DPL
重要暫存器	0000	00	00	07	00	00

	P0	P1	P2	P3
輸入/出埠	FF	FF	FF	FF

	R0	R1	R2	R3	R4	R5	R6	R7
工作暫存器區	92	04	24	93	05	25	94	06

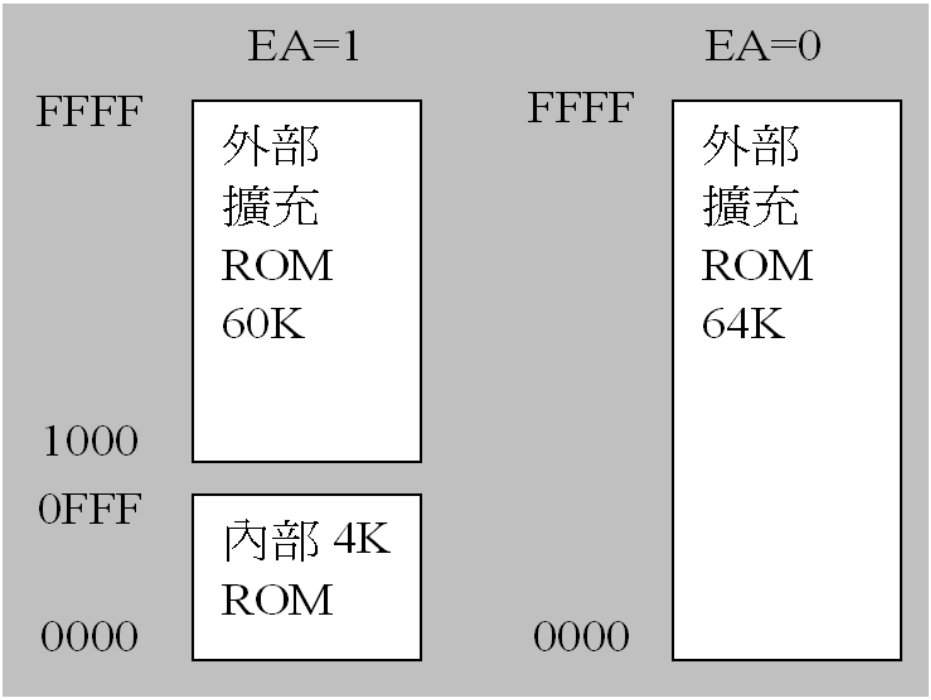
	PSW	CY	AC	F0	RS1	RS0	OV	F1	P
程式狀態字	00	0	0	0	0	0	0	0	0

3.1.7 PSW 程式狀態字：

位元	名稱	功能
D7	CY：進位旗標或借位旗標	(Carry Flag)進位旗標，用來表示算術指令運算後的結果，其資料的 bit 7 是否有進位或借位。 加法運算時(ADD)的結果：有進位 C=1，沒有進位 C=0。 減法運算時(SUB)的結果：有借位 C=1，沒有借位 C=0。
D6	AC：半進位旗標或半借位旗標	(Aux Carry Flag)半進位旗標，用來表示運算後資料的 bit 3 是否有向 bit 4 進位或借位。 加法運算時(ADD)的結果：有進位 AC=1，沒有進位 AC=0。 減法運算時(SUB)的結果：有借位 AC=1，沒有借位 AC=0。
D5	F0：通用位元	可作爲一般的讀/寫位元。
D4	RS1：暫存器庫選擇位元 1	暫存器庫選擇(Regiser Bank Select)位元 1 及位元 0。 RS1 RS0　暫存器庫選擇 0　0　RB0(位址 00h-07h) 0　1　RB1(位址 08h-0Fh) 1　0　RB2(位址 10h-17h) 1　1　RB3(位址 18h-1Fh)
D3	RS0：暫存器庫選擇位元 0	
D2	OV：溢位旗標	(Over)溢位旗標，表示程式經算術或邏輯運算後的結果是否有溢位，若是 OV=1，若不是 OV=0。
D1	-	空位元
D0	P：同位元旗標	(Parity)同位元旗標，表示累積器的內容爲奇數個“1”則 P=0，偶數個“1”則 P=1。

四電子三甲 B9502004 葉俊邑
四電子三甲 B9502022 楊竣宇
四電子三甲 B9502028 陳敬翔

3.1.8　8051 記憶體介紹：



3.1.9　中斷向量位址：

中斷	位址	功能
RESET	0000H	系統重置啓始位址
INT0	0003H	外部中斷 INT0 向量
INT1	0013H	外部中斷 INT1 向量
TIMER0	000BH	計時計數中斷 TIMER0 向量
TIMER1	001BH	計時計數中斷 TIMER1 向量
TIMER2	002BH	計時計數中斷 TIMER2 向量
UART	0023H	串列埠中斷向量

3.1.10資料記憶體結構：

FFH | 80H	8051 特殊功能暫存器(SFR) 或是 8052 的間接定址資料區
7FH | 30H	使用者的一般資料存放區 (亦可透過 SP 設定,存放堆疊資料)
20H~2FH	可位元定址區(20.0~20.7……2F.0.~2F.7)
10H~1FH	暫存器庫 RB3(R0~R7)
10H~17H	暫存器庫 RB2(R0~R7)
08H~0FH	暫存器庫 RB1(R0~R7)
00H~07H	暫存器庫 RB0(R0~R7)

3.1.11　　11、暫存器庫：

RS1	RS0		
		7Fh | 30h	使用者 RAM
		2Fh | 20h	位元定址區
1	1	1Fh | 18h	R7 | 暫存器庫 RB3 R0
1	0	17h | 10h	R7 | 暫存器庫 RB2 R0
0	1	0Fh | 08h	R7 | 暫存器庫 RB1 R0
0	0	07h | 00h	R7 | 暫存器庫 RB0 R0

3.1.12　　12、可位元定址區：

位元定址									
		7	6	5	4	3	2	1	0
位元組定址	2Fh	7F	7E	7D	7C	7B	7A	79	78
	2Eh	77	76	75	74	73	72	71	70
	2Dh	6F	6E	6D	6C	6B	6A	69	68
	2Ch	67	66	65	64	63	62	61	60
	2Bh	5F	5E	5D	5C	5B	5A	59	58
	2Ah	57	56	55	54	53	52	51	50
	29h	4F	4E	4D	4C	4B	4A	49	48
	28h	47	46	45	44	43	42	41	40
	27h	3F	3E	3D	3C	3B	3A	39	38
	26h	37	36	35	34	33	32	31	30
	25h	2F	2E	2D	2C	2B	2A	29	28
	24h	27	26	25	24	23	22	21	20
	23h	1F	1E	1D	1C	1B	1A	19	18
	22h	17	16	15	14	13	12	11	10
	21h	0F	0E	0D	0C	0B	0A	09	08
	20h	07	06	05	04	03	02	01	00

3.1.13 中斷向量位址：

中斷	位址	功能
RESET	0000H	系統重置啓始位址
INT0	0003H	外部中斷 INT0 向量
INT1	0013H	外部中斷 INT1 向量
TIMER0	000BH	計時計數中斷 TIMER0 向量
TIMER1	001BH	計時計數中斷 TIMER1 向量
TIMER2	002BH	計時計數中斷 TIMER2 向量
UART	0023H	串列埠中斷向量

3.2　LCD 液晶顯示器：

3.2.1 LCD 液晶顯示器功能：

（1） 14 支接腳的 IC。

（2） LCD 顯示器內部具有字元產生器，因此它可以接收 ASCII 字元碼。

（3） 供許多 LCD 顯示方式的控制指令，例如清除顯示畫面、游標歸位、顯示 On/Off、游標 On/Off、閃爍顯示、游標移動等功能。

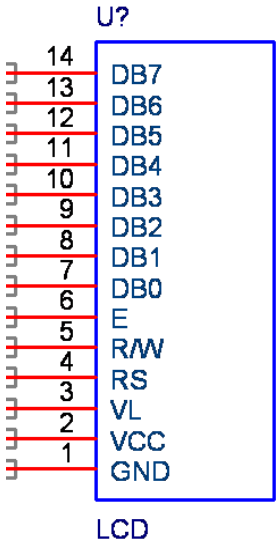
3.2.2 LCD 液晶簡介：

（1） LCD（Liquid Crystal Display）液晶顯示器，顯示方式可分爲：文字型 LCD 與繪圖型 LCD 兩種。

（2） 常見文字型 LCD 有 16 字 X2 列、20 字 X2 列、40 字 X2 列幾種

（3） 介面以 14 支信號接腳最爲常見。

3.2.3 LCD 接腳圖與功能：



接腳編號	信號名稱	名稱及功能
1	VSS	電源接地: 0V
2	Vdd	電源正端: +5V
3	V_0	LCD亮度調整電壓輸入
4	RS	暫存器選擇(Register Select)信號 RS=0:選擇指令暫存器 RS=1:選擇資料暫存器
5	R/W	讀寫信號線 R/W=0:資料寫入(Write)LCD R/W= 1:讀取(Read) LCD資料
6	E	LCD致能(Enable)信號，高電位時，讀取或寫入之資料方爲有效
7~14	DB0~DB7	資料匯流排(Data Bus)， 當使用8位元資料匯流排時，DB0~DB7皆有效， 而使用4位元資料匯流排時，僅DB4~DB7有效

3.2.4 LCD 內部結構與功能：

（1） 文字型 LCD 內部結構包括指令暫存器（IR）、資料暫存器（DR）、顯示記憶體（DD RAM）、與字元產生器 ROM（CG ROM）。

（2） DD RAM：對應 LCD 顯示器上的記憶體。當 ASCII 資料寫入 DD RAM，對應的位址就會顯示該字元。

（3） CG ROM：LCD 內部有預存 192 個 5x7 點陣字型。這些資料只能讀取，不能更改，所以只要將 ASCII 寫入 DD RAM，對應的點陣字型會顯示在指定的位址上。

（4） 區分爲指令暫存器（IR）與資料暫存器（DR），由 RS 來選擇。

（5） IR 主要作用是接受所下達的各項控制指令，諸如清除顯示內容、游標位移、顯示資料 RAM（DD RAM）的位址以及字型產生 ROM（CG ROM）的位址等等指令。

（6） DR 主要作用存取 DD RAM 與 CG ROM 中的資料。當欲將資料寫入 DD RAM 或 CG ROM 時，並非直接寫入，而是透過 DR 作爲緩衝。LCD 執行寫入的程序是先將資料載入 DR，然後再自動轉換至 DD RAM 或 CG ROM。

3.2.5 暫存器之選擇與控制介面信號：

E	RS	R/W	作用
1	0	0	寫入指令暫存器（IR）
1	0	1	讀取忙碌旗標（BF）或位址計數器（AC）
1	1	0	寫入資料暫存器（DR）
1	1	1	讀取資料暫存器（DR）

3.2.6 LCD 晶片輸出入：

項目	指令功能	字組	D7	D6	D5	D4	D3	D2	D1	D0
1	清除螢幕	01H	0	0	0	0	0	0	0	1
2	游標回到原點	02H~03H	0	0	0	0	0	0	1	*
3	進入模態設定	04H~07H	0	0	0	0	0	1	I/D	S
			I/D=1 使用位址遞增；I/D=0 使用位址遞減； S=1 螢幕 ON；S=0 螢幕 OFF；							
4	螢幕/游標顯示開關	08H~0FH	0	0	0	0	1	D	C	B
			D=1 表示螢幕 ON；D=0 表示螢幕 OFF； C=1 表示游標 ON；C=0 表示游標 OFF； B=1 表示閃爍 ON；B=0 表示閃爍 OFF；							
5	螢幕/游標移位控制	10H~1CH	0	0	0	1	S/C	R/L	*	*
			S/C=1 表示螢幕移位；S/C=0 表示游標移位 R/L=1 表示游標右移；R/L =0 表示游標左移；							
6	功能設定	20H~3CH	0	0	1	DL	N	F	*	*
			DL=1 表示規劃為 8 位元 (DB0~DB7); DL=0 表示規劃為 4 位元(DB4~DB7) F=1 表 5×10 點矩陣；F=0 表 5×7 點矩陣 N=1 表 2 行顯示；N=0 表 1 行顯示							
7	指定顯示位置	80H~FFH	1	*	*	*	*	*	*	*

3.2.7 計算 LCD 記憶體位址：

80H	81 H	82 H	…	91 H	92 H	93 H
C0 H	C1 H	C2 H	…	D1 H	D2 H	D3 H

3.2.8 電路製作：

（1） 一組 8 位元的輸出埠（P0）當資料線（DB0～DB7）。

（2） 三條輸出點（P1.0～P1.2）當控制線（E，RW，RS）。

（3） RL 接地，或接可變電壓，調整明亮度。

3.2.9 程式流程：

（1） 設定 LCD 使用模式爲 8 位元，5X10 點矩陣字型，兩行字顯示。

（2） 設定螢幕與游標狀態。

（3） 清除螢幕（CMD=1）。

（4） 設定顯示位址。

（5） 將資料寫入資料暫存器。

3.3 鍵盤掃描控制電路：

4X4 按鍵的控制電路，使用 8051 埠 28 條 I/O 線做 16 個按鍵的鍵盤掃描，由 P2.0～P2.3 送出掃描信號，而由 P2.4～P2.7 讀取按鍵資料返回碼。以程式掃描的方式來偵測湔一按鍵按下，一次掃描一行四個按鍵，掃描的順序如下：

3.3.1 送出掃描信號 1110 共掃描第一行的四個按鍵，讀取按鍵資料，判斷該行是否有鍵按下，若有鍵按下，則連接至被按下的該鍵返回線狀態爲 0。

3.3.2 送出掃描信號 1101 以掃描第二行的四個按鍵，讀取按鍵資料，判斷該行是否有鍵按下。

3.3.3 送出掃描信號 1011 以掃描第三行的四個按鍵，讀取按鍵資料，判斷該行是否有鍵按下。

3.3.4 送出掃描信號 0111 以掃描第四行的四個按鍵，讀取按鍵資料，判斷該行是否有鍵按下。

3.3.5 回到步驟 I 繼續做按鍵掃描。

3.3.6 按鍵編號、掃描信號與讀取按鍵資料返回碼：

以上的步驟連續地重覆，若有按鍵被按下，就將該按鍵解碼出來，至於如何解碼，可以使用雙重迴圈做計數編號，當某一按鍵按下時，其按鍵編號便是計數編號，有關按鍵編號、掃描信號及讀取按鍵資料返回碼列表如下：

鍵號	按鍵資料輸入碼				掃描輸出信號				所偵測的按鍵
	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	
0	1	1	1	0	1	1	1	0	K0 鍵
1	1	1	0	1	1	1	1	0	K1 鍵
2	1	0	1	1	1	1	1	0	K2 鍵
3	0	1	1	1	1	1	1	0	K3 鍵
4	1	1	1	0	1	1	0	1	K4 鍵
5	1	1	0	1	1	1	0	1	K5 鍵
6	1	0	1	1	1	1	0	1	K6 鍵
7	0	1	1	1	1	1	0	1	K7 鍵
8	1	1	1	0	1	0	1	1	K8 鍵
9	1	1	0	1	1	0	1	1	K9 鍵
10	1	0	1	1	1	0	1	1	K10 鍵
11	0	1	1	1	1	0	1	1	K11 鍵
12	1	1	1	0	0	1	1	1	K12 鍵
13	1	1	0	1	0	1	1	1	K13 鍵
14	0	0	1	1	0	1	1	1	K14 鍵
15	1	1	1	1	0	1	1	1	K15 鍵

3.4　中斷計時計數與串列通訊：

3.4.1 MCS51 的中斷簡介

（1） 單晶片在處理外部輸出入訊號的方式有兩種：

A.　　PIO（程式 I/O-ProgramI/O）：

透過程式指令讀取 I/O 埠狀態。但這種輸入方式無法在第一時間立即處理某些具有時效性的外部輸入訊號。

B.　IIO（中斷 I/O-InterruptI/O）：

可立即中斷目前正在執行的程式，並做立即的回應與處理，本章將介紹中斷的原理，型式，與應用。

3.4.2 MCS51 的中斷型式：

（1） 外部中斷：

8051 的外部中斷是指來自晶片硬體接腳 INT0、INT1 的中斷。這兩個外部中斷的觸發方式有低準位觸發（lowleveltrigger）與負緣觸發（fallingedgetrigger）兩種，可經由 TCOM 暫存器的 ITX（X=0 或 1）位元設定。

（2） 計時/計數中斷：

當 8051 的計時/計數器（TIMER0、TIMER1）產生溢位時，溢位旗標 TF0（或 TF1）會自動設爲 1，直到 CPU 跳到對應的中斷向量位址，執行中斷服務程式時，才會自動將 TF0（或 TF1）清除爲 1。

（3） 串列埠中斷：

當 8051 的發射中斷旗標 TI 或接收中斷旗標 RI 爲 1 時，會產生串列中斷請求。在中斷服務程式中，必須用指令清除 TI 與 RI，因爲硬體不會自動清除這兩個位元。

3.4.3 MCS-51 的外部中斷：

（1） MCS-51 有 5 個中斷源，2 個外部硬體中斷，2 個計時/計數中斷，1 個串列埠傳輸中斷：

A.　外部中斷：

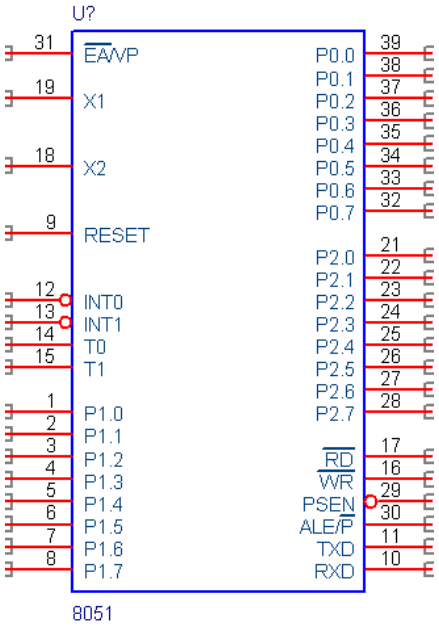
INT0、INT1（P3.2，P3.3）。

B.　計時/計數中斷：

T0，T1（P3.4，P3.5）。

C.　　串列通訊輸出/入：

TXD，RXD（P3.0，P3.1）。



3.4.4 MCS-51 的中斷功能：

（1） 中斷源的致能方法（使用 IE 暫存器）。

（2） 外部中斷觸發方式（TCON）。

（3） 中斷優先順序（使用 IP 暫存器）。

（4） 中斷向量（中斷向量表位址）。

（5） 中斷處理流程。

3.4.5 中斷致能控制（IE 暫存器）：

（1） 中斷源利用中斷致能暫存器（IE 暫存器）來控制其致能與禁能。當中斷源被致能時，中斷才能被 CPU 接受。八位元例如，第 0 個外部中斷與第 1 個計時器中斷，IE 必須令爲#89H=#10001001B，即指令爲＜MOVIE，#89H＞，第 INT0 與第 TIMER1 中斷。

（2） 當 IE 小於或等於 80H（即 IE=#0xxxxxxxB，或 IE=#10000000B），所有中斷皆無效（即 EA=1，或 EX0~EX5 皆爲 0）。

IE.7	IE.6	IE.5	IE.4	IE.3	IE.2	IE.1	IE.0
EA	未用	ET2	ES	ET1	EX1	ET0	EX0

（3） 功能說明：

名稱	位元	說　明
EX0	IE.0	第 0 個外部中斷致能位元。(1:致能；0 禁能)
ET0	IE.1	第 0 個計時中斷致能位元。(1:致能；0 禁能)
EX1	IE.2	第 1 個外部中斷致能位元。(1:致能；0 禁能)
ET1	IE.3	第 1 個計時中斷致能位元。(1:致能；0 禁能)
ES	IE.4	串列通訊中斷致能位元。(1:致能；0 禁能)
ET2	IE.5	第 2 個計時中斷致能位元。(1:致能；0 禁能)
--	IE.6	保留不用
EA	IE.7	全部中斷致能與禁能位元。(1:致能；0 禁能)

表 9-1:中斷致能暫存器(IE 暫存器)

3.4.6 外部中斷觸發方式（TCON 暫存器）：

（1） INT0、INT1 外部中斷的觸發方式有兩種：

A. 低準位觸發（lowleveltrigger）：

I. 第 0 外部中斷（INT0），使用 CLRIT0 指令或 IT0=0;

II. 第 1 外部中斷 INT1，使用 CLRIT1 指令或 IT0=0;

B.　負緣觸發（fallingedgetrigger）：

I.　第 0 外部中斷（INT0），使用 SETBIT0 指令或 IT0=1;

II.　第 1 外部中斷 INT1，使用 SETBIT1 指令或 IT1=1;

TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0
TF	TR1	TF0	TR0	IE1	IT1	IE0	IT0

（2） 功能說明：

位元	名稱	功　　能
D3	IE1	外部中斷INT1顯示旗標，INT1中斷成立時，IE1=1。執行RETI時，IE1=0。
D2	IT1	外部中斷INT1中斷信號選擇，IT1=1爲負緣觸發輸入，IT1=0爲低準位輸入。
D1	IE0	外部中斷INT0顯示旗標，INT0中斷成立時，IE0=1。中斷執行完畢時，IE0=0。
D0	IT0	外部中斷INT0中斷信號選擇，IT0=1爲負緣觸發輸入，IT0=0爲低準位輸入。

3.4.7 中斷優先順序控制（IP 暫存器）：

（1） MCS-51 所有中斷源都是使用中斷優先權暫存器（IP 暫存器）來控制中斷優先順序。

例如，IP=00000100B 時，PX1=1，PX0=0，所以第 1 個外部中斷優先權高於第 0 個外部中斷；即指令爲:

```
MOV     IP,#04H      ; PX1=1，PX0=0，INT1 優先權高於 INT0
```

IP.7	IP.6	IP.5	IP.4	IP.3	IP.2	IP.1	IP.0
未用	未用	PT2	PS	PT1	PX1	PT0	PX0

（2） 功能說明：

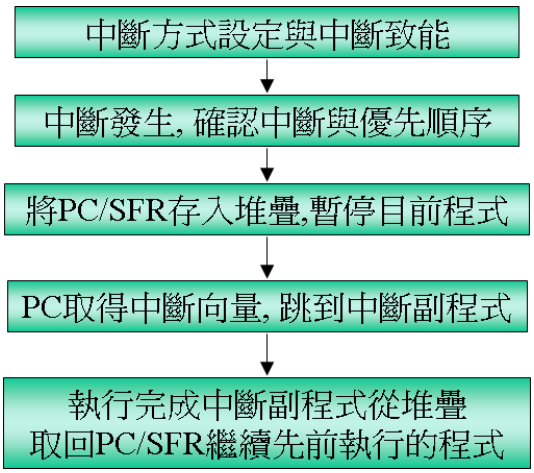
名稱	位元	說　明
PX0	IP.0	定義外部/INT0 之優先權
PT0	IP.1	定義外部 TIMER0 之優先權
PX1	IP.2	定義外部/INT1 之優先權
PT1	IP.3	定義外部 TIMER1 之優先權
PS	IP.4	定義串列埠之優先權
PT2	IP.5	定義外部 TIMER2 之優先權
- -	IP.6	保留不用
- - -	IP.7	保留不用

表 9-3: 中斷優先權暫存器(IP 暫存器)

3.4.8 中斷向量位址：

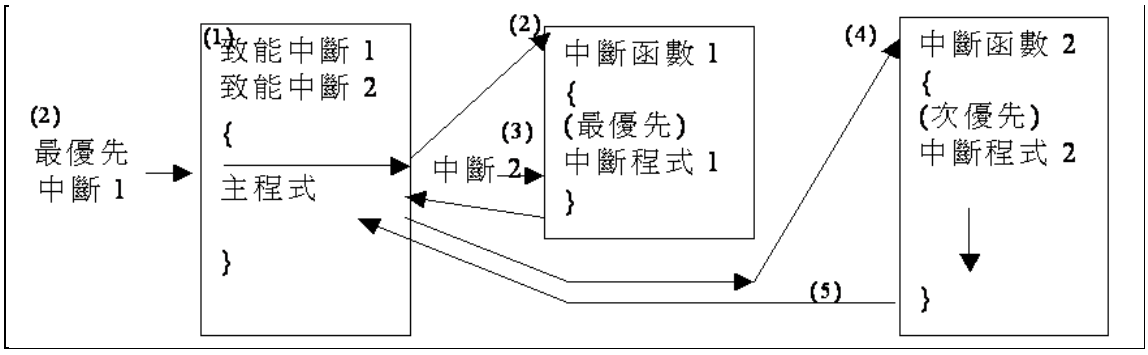
中斷	位址	功能
RESET	0000H	系統重置啓始位址
INT0	0003H	外部中斷 INT0 向量
INT1	0013H	外部中斷 INT1 向量
TIMER0	000BH	計時計數中斷 TIMER0 向量
TIMER1	001BH	計時計數中斷 TIMER1 向量
TIMER2	002BH	計時計數中斷 TIMER2 向量
UART	0023H	串列埠中斷向量

3.4.9 中斷處理流程：

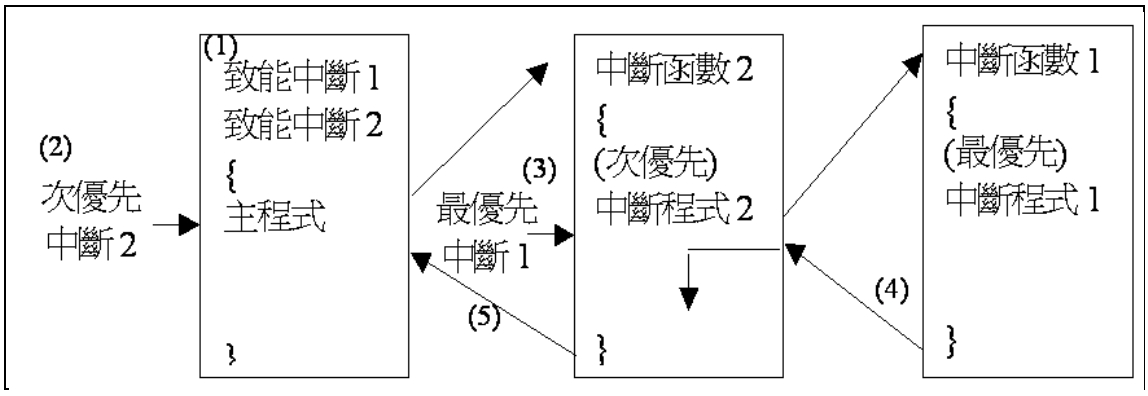


3.4.10中斷程式的工作方式：

（1） 最優先中斷先輸入：



（2） 次優先中斷先輸入：



4. 設計內容：

4.1 程式碼：

```
//------------------------------

//加入 AT89X51.H 檔。
#include <AT89X51.H>

//定義 RS 爲 LCD 之 RS 接腳。
#define RS      P2_7

//定義 RW 爲 LCD 之 RW 接腳。
#define RW     P2_6

//定義 Enable 爲 LCD 之致能接腳。
#define Enable       P2_5

//模式設定。
//mode = 0 -> clock mode
//mode = 1 -> set clock mode
//mode = 2 -> calculator
int mode=0;

//鍵盤暫時儲存用。
int KeyTemp=0xff;

//計算鍵盤的值。
int key=0;

//用來儲存鍵盤的值。
int KeyData=0xff;

//儲存鍵盤的行值。
char col=0;

//儲存鍵盤的列值。
char row;
```

```
//防鍵盤的彈跳。
char one=0;

//防鍵盤的彈跳。
char zero=0;

//鍵盤掃描用。
char ScanLine=0x08;
//above is for KeyScan function used

//計算時間十秒回復時鐘。
int timeoutcnt=0; //count 10 second then go to mode 0

//顯示時間數字陣列。
int timenum[6]={0};//stored clock number

//計數時間用（配合中斷）。
int secondcnt = 0;//use to count every second

//是否跳出的指標。
char flag_timeout;

//------------------------------

//宣告延遲函數。
void delay(void);

//定義檢查 LCD 是否做完一道指令函數。
void CheckBusy(void);

//定義鍵盤掃描函數。
void KeyScan(void);

//定義寫入 LCD 指令函數。
void WriteIns(char instruction);
```

```
//定義初始 LCD 函數。
void InitialLCD(void);

//定義寫入 LCD 資料函數。
void WriteData(char i);

//定義時鐘計數函數。
void Clock(void);

//定義顯示時間的函數。
void ShowTime(void);
//timer 0

//-----------------------------

//宣告中斷函數。
void T0_int(void) interrupt 1
{
	TH0=(65536-50000)/256;
	TL0=(65536-50000)%256;
	secondcnt++;
	if(secondcnt == 20)
	{
		if(flag_timeout==1)
		{
			timeoutcnt++;
			if(timeoutcnt==10)
			{
				timeoutcnt=0;
				mode=0;
				WriteIns(0x01);
				WriteIns(0x82);
			}
		}
		Clock();
		secondcnt=0;
	}
}
```

```
//-----------------------------

//定義時鐘計數函數。
void Clock(void)
{
    timenum[0]++;
    if(timenum[0]>9)
    {
        timenum[1]++;
        timenum[0]=0;
        if(timenum[1]>5)
        {
            timenum[2]++;
            timenum[1]=0;
            if(timenum[2]>9)
            {
                timenum[3]++;
                timenum[2]=0;
                if(timenum[3]>5)
                {
                    timenum[4]++;
                    timenum[3]=0;
                    //hour
                    if( (timenum[4]>9)&(timenum[5]<2) )
                    {
                        timenum[5]++;
                        timenum[4]=0;
                    }
                    if( (timenum[4]>3)&(timenum[5]==2) )
                    {
                        timenum[5]=0;
                        timenum[4]=0;
                    }
                }
            }
        }
    }
```

```
}

//------------------------------

//定義顯示時間的函數。
void ShowTime(void)
{
        WriteIns(0xc4);
        WriteData(0x30+timenum[5]);
        WriteData(0x30+timenum[4]);
        WriteData(0x3A);
        WriteData(0x30+timenum[3]);
        WriteData(0x30+timenum[2]);
        WriteData(0x3A);
        WriteData(0x30+timenum[1]);
        WriteData(0x30+timenum[0]);
}

//------------------------------

//主程式
//code by group 13
main()
{
    int resultnum[4]={0};
    int inputnum[4]={0};
    int result=0;
    char inputcnt=0;
    char calcmd=0;
    char flag_mode2=0;
    char flag_mode1=0;
    char flag_next=0;
    char setcursor=5;
    float refloat=0;
    float ftemp1,ftemp2;
    unsigned char shinecnt=0;
    int resultmod = 0;
    int tempf1=0,tempf2=0;
```

```
P0=0x00;
P2=0x00;
P3=0x0F;
TMOD=0x11;
IE=0x8a;
TH0=(65536-50000)/256;
TL0=(65536-50000)%256;
TR0=1;
InitialLCD();
WriteIns(0x01);
WriteIns(0x82);
WriteIns(0x0e);
WriteIns(0x0c);
while(1)
{
    KeyScan();//detect key
    if( KeyData != 0xff )//if any have key
    {
        if(KeyData==0x0f)//if key is 15 then set mode=1
        {
            mode = 1;
        }
        else
        {
            if(flag_mode1 != 1)
                mode = 2;
        }
        flag_timeout = 1 ;
        timeoutcnt=0;
    }

    if( mode == 0 )
    {
        if(secondcnt<3)
            ShowTime();
        flag_mode1 = 0;
        flag_mode2 = 0;
    }
```

```
if( mode == 1 )
{
    flag_mode2 = 0;
    if(flag_mode1 == 0)
    {
        flag_mode1 = 1;
        TR0=0;
        setcursor=5;
    }
    else
    {
        switch(KeyData)//set clock number
        {
            case 0:
            case 1:
            case 2:
            case 3:
            case 4:
            case 5:
            case 6:
            case 7:
            case 8:
            case 9:
                switch(setcursor)
                {
                    case 0:
                    case 2:
                        timenum[setcursor] = KeyData;
                    break;
                    case 1:
                    case 3:
                        if(KeyData <6)
                            timenum[setcursor] = KeyData;
                        else
                            timenum[setcursor] = 5;
                    break;
                    case 4:
```

```
                    if( (timenum[5]==2)&(KeyData>3) )
                        timenum[4] = 3;
                    else
                        timenum[4] = KeyData;
                break;
                case 5:
                    if(KeyData<3)
                    {
                        if( (timenum[4]>3)&(KeyData==2) )
                        {
                            timenum[5] = 1;
                        }
                        else
                            timenum[5] = KeyData;
                    }
                    else
                    {
                        if(timenum[4]<4)
                            timenum[5] = 2;
                        else
                            timenum[5] = 1;
                    }
                break;
                default:break;
            }
            setcursor--;
            if(setcursor<0)
                setcursor=5;
        break;
        case 11://set cursor
            setcursor--;
            if(setcursor<0)
                setcursor=5;
        break;
        case 10:
            setcursor++;
            if(setcursor>5)
                setcursor=0;
```

```
                break;
                case 12:break;
                case 13:break;
                case 14:break;
                case 15:
                        mode = 0;
                        TR0=1;
                break;
                default:break;
        }
        shinecnt++;
        if(shinecnt==150)
                shinecnt=0;
        if(shinecnt>=75)//shine which will be setted
        {
                switch(setcursor)
                {
                        case 0:
                                WriteIns(0xc4+0x07);
                        break;
                        case 1:
                                WriteIns(0xc4+0x06);
                        break;
                        case 2:
                                WriteIns(0xc4+0x04);
                        break;
                        case 3:
                                WriteIns(0xc4+0x03);
                        break;
                        case 4:
                                WriteIns(0xc4+0x01);
                        break;
                        case 5:
                                WriteIns(0xc4);
                        break;
                }
                WriteData(' ');
        }
```

```
                else
                {
                    ShowTime();
                }

            }
        }
        if( (mode == 2)&(KeyData!=0xFF) )//calculator
        {
            flag_mode1 = 0;
            if(flag_mode2 == 0)//initail
            {
                flag_mode2 = 1;
                inputcnt=0;
                WriteIns(0x01);
                calcmd = 0;
                flag_next=0;
                inputnum[0]=0;
                inputnum[1]=0;
                inputnum[2]=0;
                inputnum[3]=0;
            }
            else
            {
                if( (KeyData>9)&(KeyData<14) )//set command + - * /
                {
                    switch(KeyData)
                    {
                        case 10:
                            calcmd = 1;
                            WriteIns(0xc4+0x02);
                            WriteData(0x2B);
                        break;
                        case 11:
                            calcmd = 2;
                            WriteIns(0xc4+0x02);
                            WriteData(0x2D);
                        break;
```

```
                case 12:
                    calcmd = 3;
                    WriteIns(0xc4+0x02);
                    WriteData(0x2A);
                break;
                case 13:
                    calcmd = 4;
                    WriteIns(0xc4+0x02);
                    WriteData(0x2F);
                break;
                default:break;
            }
            if(flag_next==0)
            {
                flag_next=1;
                inputcnt=2;
            }
            else
            {
                flag_next=0;
                inputcnt=0;
            }
        }
        if( (KeyData<10) ) // set operand number
        {
            if(flag_next==0)
            {
                if(inputcnt==0)
                {
                    inputnum[1] = KeyData;
                }
                if(inputcnt==1)
                {
                    inputnum[0] = inputnum[1];
                    WriteIns(0xc4+inputcnt-0x01);
                    WriteData(0x30 + inputnum[0]);
                    inputnum[1] = KeyData;
                }
```

```
                    WriteIns(0xc4+inputcnt);
                    WriteData(0x30 + KeyData);
                    inputcnt++;
                    if(inputcnt>=2)
                        inputcnt=0;
                }
                else
                {
                    if(inputcnt==2)
                    {
                        inputnum[3] = KeyData;
                    }
                    if(inputcnt==3)
                    {
                        inputnum[2] = inputnum[3];
                        WriteIns(0xc4+inputcnt);
                        WriteData(0x30 + inputnum[2]);
                        inputnum[3] = KeyData;
                    }
                    WriteIns(0xc4+inputcnt+0x01);
                    WriteData(0x30 + KeyData);
                    inputcnt++;
                    if(inputcnt>=4)
                        inputcnt=2;
                }
            }
            if(KeyData==14)
            {
                flag_mode2 = 0;
                switch(calcmd) //if it doesn't set the command and press the answer then
display Error
                {
                    case 0:
                        WriteIns(0x01);
                        WriteIns(0x82);
                        WriteIns(0xc4+0x01);
                        WriteData('E');
                        WriteData('r');
```

```
                    WriteData('r');
                    WriteData('o');
                    WriteData('r');
                break;
                case 1:    //add
                    result = (inputnum[0]*10 + inputnum[1])+(inputnum[2]*10 +
inputnum[3]);
                    resultnum[0] = result%10;
                    resultnum[1] = (result/10)%10;
                    resultnum[2] = result/100;
                    WriteIns(0x01);
                    WriteIns(0x82);
                    WriteIns(0xc4+0x01);
                    if(resultnum[2]>0)
                        WriteData(0x30 + resultnum[2]);
                    if( (resultnum[1]>0)|(resultnum[2]!=0) )
                        WriteData(0x30 + resultnum[1]);
                    WriteData(0x30 + resultnum[0]);
                break;
                case 2: //minus
                    result = (inputnum[0]*10 + inputnum[1])-(inputnum[2]*10 +
inputnum[3]);
                    WriteIns(0x01);
                    WriteIns(0x82);
                    WriteIns(0xc4);
                    if(result<0)
                    {
                        result = (~result)+1;
                        WriteData(0x2D);
                    }
                    resultnum[0] = result%10;
                    resultnum[1] = (result/10)%10;
                    if(resultnum[1]>0)
                        WriteData(0x30 + resultnum[1]);
                    WriteData(0x30 + resultnum[0]);
                break;
                case 3: //product
                    result = (inputnum[0]*10 + inputnum[1])*(inputnum[2]*10 +
```

```
inputnum[3]);
                    resultnum[0] = result%10;
                    resultnum[1] = (result/10)%10;
                    resultnum[2] = (result/100)%10;
                    resultnum[3] = result/1000;
                    WriteIns(0x01);
                    WriteIns(0x82);
                    WriteIns(0xc4+0x01);
                    if(resultnum[3]>0)
                        WriteData(0x30 + resultnum[3]);
                    if( (resultnum[2]>0)|(resultnum[3]!=0) )
                        WriteData(0x30 + resultnum[2]);
                    if( (resultnum[1]>0)|(resultnum[2]!=0)|(resultnum[3]!=0) )
                        WriteData(0x30 + resultnum[1]);
                    WriteData(0x30 + resultnum[0]);
                break;
                case 4: //divid
                    WriteIns(0x01);
                    WriteIns(0x82);
                    WriteIns(0xc4+0x01);
                    ftemp1 = inputnum[0]*10 + inputnum[1];
                    ftemp2 = inputnum[2]*10 + inputnum[3];
                    tempf1 = inputnum[0]*10 + inputnum[1];
                    tempf2 = inputnum[2]*10 + inputnum[3];
                    if( ftemp2!=0 ) //if ftemp2 = 0 then show Error
                    {
                        refloat = ftemp1/ftemp2;
                        result = refloat;
                        resultmod = tempf1%tempf2;


                        resultnum[0] = resultmod%10;
                        resultnum[1] = resultmod/10;

                        resultnum[2] = result%10;
                        resultnum[3] = result/10;

                        if(resultnum[3]>0)
```

```
                                        WriteData(0x30 + resultnum[3]);
                                    WriteData(0x30 + resultnum[2]);
                                    WriteData('.');
                                    WriteData('.');
                                    WriteData('.');
                                    if(resultnum[1]>0)
                                        WriteData(0x30 + resultnum[1]);
                                    WriteData(0x30 + resultnum[0]);
                                }
                                else
                                {
                                    WriteData('E');
                                    WriteData('r');
                                    WriteData('r');
                                    WriteData('o');
                                    WriteData('r');
                                }
                            break;
                            default:break;
                        }

                    }

                }
            }
            KeyData = 0xff;
        }
}

//------------------------------

//定義初始 LCD 函數。
void InitialLCD(void)
{
   WriteIns(0x38);
   WriteIns(0x38);
   WriteIns(0x38);
   WriteIns(0x38);
```

```
   //WriteIns(0x08);/* off display */
   WriteIns(0x01);  /* clear buffer */
   WriteIns(0x0c);  /* on display */
   WriteIns(0x06);  /* set input mode */
}

//------------------------------

//定義寫入 LCD 指令函數。
void WriteIns(char instruction)
{
   RS=0;
   RW=0;
   Enable=1;
   P0=instruction;
   Enable=0;
   CheckBusy();
}

//------------------------------

//定義寫入 LCD 資料函數。
void WriteData(char i)
{
   RS=1;
   RW=0;
   P0=i;
   Enable=1;
   P0=i;
   Enable=0;
   CheckBusy();
}

//------------------------------

//定義檢查 LCD 是否做完一道指令函數。
void CheckBusy(void)
{
```

```
   char i=0x80;
   while(i&0x80)
      {
         RS=0;
         RW=1;
         Enable=1;
         i=P0;
         Enable=0;
         delay();
      }
}

//------------------------------

//宣告延遲函數。
void delay(void)
{
   int i;
   for(i=0;i<100;i++)
      ;
}

//------------------------------

//定義鍵盤掃描函數。
void KeyScan(void)
{
   int KeyStatus;


   P3=~ScanLine;
   KeyStatus=~P3;
   KeyStatus&=0xf0;
   for(row=0;row<4;row++)
      {
          if(KeyStatus==0x80)
              {
            one=0;
```

```
                if(KeyTemp!=key)
                {
            KeyTemp=key;
            one=0;
            zero=1;
                    }
                else
                {
                   zero++;
                   if(zero==5)
                      KeyData=KeyTemp;
                }
          }
           key+=1;
           KeyStatus<<=1;
      } /* row */
   ScanLine>>=1;
   if(ScanLine==0)
       ScanLine=0x08;
   col++;
   if(col==4)
      {
         col=0;
         key=0;
         one++;
         if(one==5)
            {
               zero=0;
               KeyTemp=0xff;
               KeyData=0xff;
            }
      }
}
```

5. 設備及材料：

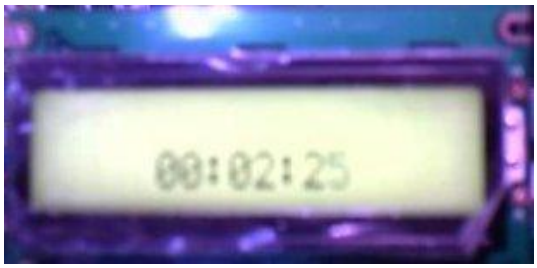
5.1　德源科技 8051 及 AVR 綜合實驗板

6. 實做及查錯

我們這一組在做此次實習時一開始就規劃是否能在老師規定的規格上做延伸，加入了在設定時間時，可以使用十或一鍵來調整欲控制的數字，以達到更符合人性的控制，最後順利完成。

7. 實習結果

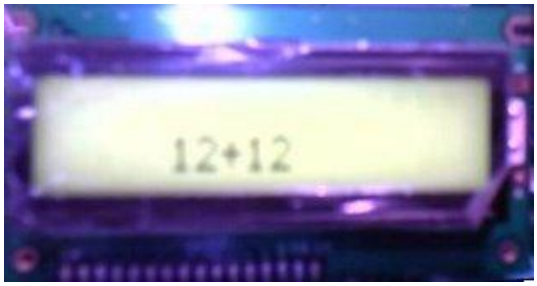
7.1　Time-out 模式（時鐘模式）：

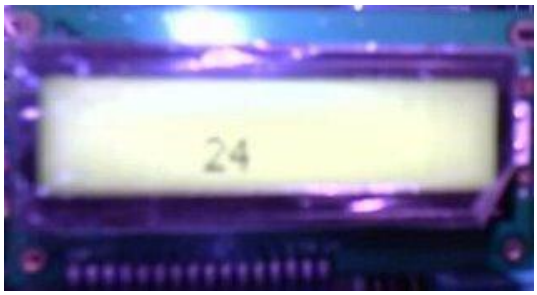


7.2　時間設定模式：

7.3　計算機模式

7.3.1 加法：





7.3.2 減法：

7.3.3 乘法：

7.3.4 除法：

8. 實習討論

本實習成仍有一些小細節仍需加強與改善，條列如下：

8.1 實習板上之數字矩陣鍵盤趨於老舊而導致難以操作。

8.2 程式需使用的變數過多，因爲要在僅有兩個中斷的限制下完成許多功能，所以在變數很多的情形下，導致除錯困難。

9. 實習心得

這次實驗，老實來說不是很難，運用到了以前數位邏輯系統跟設計的實習，雖然大部分都做過，但也沒這麼容易，因爲光是第一次的題目，就是以前我們好不容易完成作品的總和，意思就是要把以前學會的都整合在一起，真正學會的人，是不會被這個整合的地方卡住的，但往往一般人都會卡在這裡，因爲大家都還沒練到靈活運用，憑著這樣的技術是無法再未來社會上立足的，雖然我們算是前幾組完成的，但我們也坦承，曾經被不明的問題給卡住過，以上種種似乎都意味著我們還需加強實力，而我們也都知道這次的實驗題目只是個開胃菜而已，期待到學期末的最後一個題目，是如此的有挑戰性，也希望能在最後找到自我的成就感。

10. 參考資料

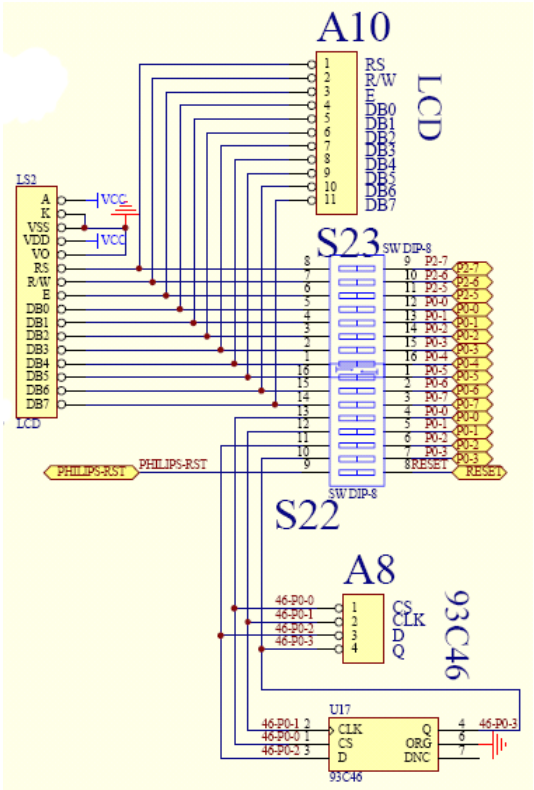
10.1 楊明豐，碁峰資訊股份有限公司，「8051 單晶片 C 語言設計實務：使用 Keil C」

10.2 實習板功能介紹光碟，各單元電路圖／完整電路圖.pdf

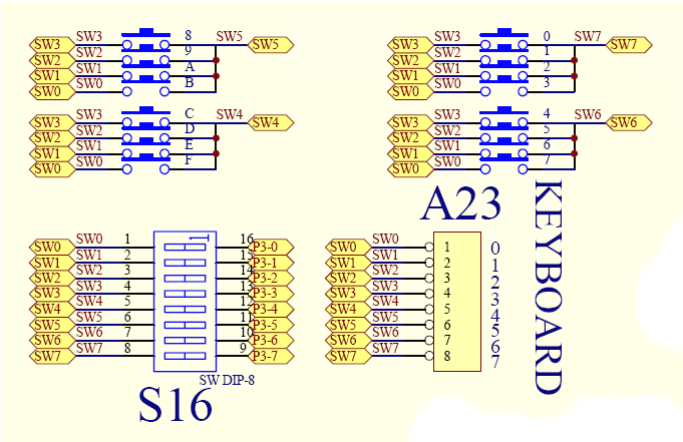
11. 附件

11.1 完整電路圖：

11.1.1 LCD：



11.1.2 KEY BOARD：



國立臺灣科技大學
電子工程系
嵌入式系統設計實習

報告結束

感謝老師指導與詳閱