**基本程式撰寫**

入門的程式撰寫通常都由計算機內「加減乘除」的指令，若在需要寫更大的程式，也必定需要更多的小程式來撰寫，故在此單元中，將會介紹一些基本的程式，供後續撰寫程式之用。

* 1. **讀取表格的數字**

為了要讀取表格內的數字，須將表格先放在DPTR暫存器裡面，接著再將暫存器裡表格內的數字按照順序讀取，如下表程式中，將表格的第一個數字放於50H中，放完一個數字時，使位置加一，即可讓下一個數字放於下一個位置上。

|  |
| --- |
| **ORG 00H** ;從00H開始執行指令  **JMP START** ;跳到程式START  **START:**  **MOV R2,#16** ;表格有16個數字  **MOV R0,#50H** ;將表格的數字放在50H位置  **MOV R1,#00** ;初值(表格內的數字需要一個一個掃，故需要一個指向掃到哪一個表格數字的值)  **MOV DPTR,#SEG7\_TABLE** ;將表格SEG7\_TABLE放在DPTR暫存器裡  **LOOP\_1:**  **MOV A,R1** ;將R1的值放於A  **MOVC A,@A+DPTR** ;將表格內第A(R1)個的數字放於A  **MOV @R0,A** ;將找到的值放回R0  **INC R1** ;R1+1  **INC R0** ;R0+1  **DJNZ R2,LOOP\_1** ;將R2-1放入R2，若R2不等於0則返回LOOP\_1去  **JMP $** ;Jump here  ; SEG7\_TABLE表格  **SEG7\_TABLE:**  **DB 0C0H,0F9H,0A4H,0B0H,099H,092H,082H,0F8H**  **DB 080H,090H,088H,083H,0C6H,0A1H,086H,08EH**  **END** ;程式結束 |

* 1. **四則運算**

在程式中很多地方都用到的加減乘除，故將他們寫成模組程式，供其他程式呼叫，使其他程式較乾淨簡潔，如下面所示。

|  |
| --- |
| **EXTRN CODE(YYY) (呼叫YYY外部模組)**  **PUBLIC XXX(宣告XXX為可供外部呼叫之模組)**  **ROM SEGMENT CODE (共享 ROM空間)**  **RSEG ROM**  **XXX: (模組名稱標記)**  **:**  **CALL YYY**  **:**  **RET (模組程式結束用 RET)**  **END (宣告結束)** |

以下將介紹16進制的加減乘除。

**16進制加法**

|  |
| --- |
| **;----------------------------------------------- --------------------------------**  **;(1) 方程式: (7FH)(7EH)+(7DH)(7CH)=(7BH)(7AH)**  **;(2) 輸入 : (7FH),(7EH),(7DH),(7CH)**  **;(3) 輸出 : (7BH),(7AH),C**  **;(4) 用到的暫存器 : A**  **;------------------------------------------------ --------------------------------**  **PUBLIC ADD\_16BIT**  ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **ADD\_16BIT:** ;模組名稱  **;(1) 低位元組相加**  **CLR C** ;清除進位旗標C  **MOV A,7CH** ;將7CH位置的值放在A  **ADD A,7EH** ;將A位置的值與7EH位置的值相加，加總後放在A  **MOV 7AH,A** ;將A的值放在7AH位置  **MOV A,7DH** ;再將7DH的值放在A  **;(2) 高位元組相加**  **ADDC A,7FH** ;將A的值與7FH位置的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7FH(數值)+C)  **MOV 7BH,A** ;將A的值放在7BH位置上  **RET** ;從模組返回  **END** ;程式結束 |

**16進制減法**

|  |
| --- |
| **;-----------------------------------------------**  **;SUB\_16BIT**  **;(1) FUNCTION: (7FH)(7EH)-(7DH)(7CH)=(7BH)(7AH)**  **;(2) INPUT : (7FH),(7EH),(7DH),(7CH)**  **;(3) OUTPUT : (7BH),(7AH),C**  **;(4) USE REG.: A**  **;------------------------------------------------**  **PUBLIC SUB\_16BIT** ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **SUB\_16BIT:** ;模組名稱  **;(1) 低位元組相減**  **CLR C** ;清除進位旗標C  **MOV A,7EH** ;將7EH位置的值放在A  **SUBB A,7CH** ;將A的值與7FH位置的值相加，加總後放在A(加的時候含進位旗標的數值)  **MOV 7AH,A** ;將A位置的值放在7AH  **;(2) 高位元組相減**  **MOV A,7FH** ;將7FH位置的值放在A  **SUBB A,7DH** ;將A的值與7FH位置的值相加，加總後放在A(加的時候含進位旗標的數值)  **MOV 7BH,A** ;將A位置的值放在7BH  **JNC SUB\_2** ;若進位旗標C為0時，則進入SUB\_2  **;(3) 相減結果為負數, 進行2'S 補數運算**  **SUB\_1:**  **MOV A,7AH**;將7AH位置的值放在A  **CPL A** ;將累加器A的值反向，如Not(A)  **ADDC A,#00** ;將A的值與00H相加，加總後放在A(加的時候含進位旗標的數值，即為A+00H+C)  **MOV 7AH,A**;將A的值放在7AH位置  **MOV A,7BH**;將7BH位置的值放在A  **CPL A**;將累加器A的值反向，如Not(A)  **ADDC A,#00**;將A的值與00H相加，加總後放在A(加的時候含進位旗標的數值，即為A+00H+C)  **MOV 7BH,A**;將A的值放在7BH位置  **SETB C** ;設定C的值為1  **;(4) 相減結果為正數, 運算完成**  **SUB\_2:**  **RET** ;從模組返回  **END** ;程式結束 |

**16進制乘法**

|  |
| --- |
| **;-----------------------------------------------**  **;MUL\_16BIT**  **;(1) FUNCTION: (7FH)(7EH)\*(7DH)(7CH)=(7BH)(7AH)(79H)(78H)**  **;(2) INPUT : (7FH),(7EH),(7DH),(7CH)**  **;(3) OUTPUT : (7BH),(7AH),(79H),(78H)**  **;(4) USE REG.: A,B**  **;------------------------------------------------**  **PUBLIC MUL\_16BIT** ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **MUL\_16BIT:** ;模組名稱  **;(1) 清除輸出緩衝區**  **MOV 7BH,#00H** ;將00H放進7BH  **MOV 7AH,#00H** ;將00H放進7AH  **MOV 79H,#00H** ;將00H放進79H  **MOV 78H,#00H** ;將00H放進78H  **;(2) (7EH)\*(7CH)=(79H)(78H)**  **MOV A,7EH** ;將7EH的資料放進A  **MOV B,7CH** ;將7CH的資料放進B  **MUL AB** ;將A\*B的值放進A  **MOV 78H,A** ;將A的資料放進78H  **MOV 79H,B** ;將B的資料放進79H  **;(3) (7FH)\*(7CH)=(7AH)(79H)**  **MOV A,7FH** ;將7FH的資料放進A  **MOV B,7CH** ;將7CH的資料放進B  **MUL AB** ;將A\*B的值放進A  **ADD A,79H**  ;將A+(79H的資料)的值放在A  **MOV 79H,A** ;將A的資料放進79H  **MOV A,B** ;將B的資料放進A  **ADDC A,7AH** ;將A的值與7A的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7AH的值+C)  **MOV 7AH,A** ;將A的資料放進7AH  **JNC MUL\_1**  ;若C=0時 則跳入MUL\_1  **INC 7BH** ;7BH的值+1  **;(4) (7EH)\*(7DH)=(7AH)(79H)**  **MUL\_1:**  **MOV A,7EH** ;將7FH的資料放進A  **MOV B,7DH** ;將7DH的資料放進B  **MUL AB** ;將A\*B的值放進A  **ADD A,79H** ;將A+(79H的資料)的值放在A  **MOV 79H,A** ;將A的資料放進7FH  **MOV A,B** ;將B的資料放進A  **ADDC A,7AH** ;將A的值與7A的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7AH的值+C)  **MOV 7AH,A** ;將A的資料放進7AH  **JNC MUL\_2** ;若C=0時 則跳入MUL\_2  **INC 7BH** ;7BH的值+1  **;(5) (7FH)\*(7DH)=(7BH)(7AH)**  **MUL\_2:**  **MOV A,7FH** ;將7FH的資料放進A  **MOV B,7DH** ;將7DH的資料放進B  **MUL AB** ;將A\*B的值放進A  **ADD A,7AH** ;將A+(7AH的資料)的值放在A  **MOV 7AH,A** ;將A的資料放進7AH  **MOV A,B** ;將B的資料放進A  **ADDC A,7BH** ;將A的值與7B的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7BH的值+C)  **MOV 7BH,A** ;將A的資料放進7BH  **RET** ;從模組返回  **END** ;程式結束 |

**16進制除法**

|  |
| --- |
| **;-----------------------------------------------**  **;DIV\_16BIT**  **;(1) FUNCTION: (7FH)(7EH)/(7DH)(7CH)=(7BH)(7AH)...(79H)(78H)**  **;(2) INPUT : (7FH),(7EH),(7DH),(7CH)**  **;(3) OUTPUT : (7BH),(7AH),(79H),(78H)**  **;(4) USE REG.: A,R5,R6,R7**  **;------------------------------------------------**  **PUBLIC DIV\_16BIT** ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **DIV\_16BIT:**  **;(1) 清除緩衝區 (79H)(78H)**  **MOV R7,#16** ;將16放入暫存器R7  **MOV 79H,#00**  ;將00放入79H位址  **MOV 78H,#00** ;將00放入78H位址  **DIV\_1:**  **;(2) C <- (7FH)(7EH)**  **MOV A,7EH** ;將7E的資料放入A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7EH,A** ;將A的資料放進7AH  **MOV A,7FH** ;將7FH的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7FH,A** ;將A的資料放進7FH  **;(3) (79H)(78H) <- C**  **MOV A,78H** ;將78H的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 78H,A** ;將A的資料放進78H  **MOV A,79H** ;將79H的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 79H,A** ;將A的資料放進79H  **;(4) R6 <- (79H), R5 <- (78H)**  **MOV R6,79H** ;將79H的資料放進R6  **MOV R5,78H** ;將78H的資料放進R5  **;(5) TEST (79H)(78H) >= (7DH)(7CH) ?**  **MOV A,78H** ;將78H的資料放進A  **SUBB A,7CH**  ;將A-7CH的資料-C(進位旗標)的值放在A  **MOV 78H,A** ;將A的資料放進78H  **MOV A,79H** ;將79H的資料放進A  **SUBB A,7DH**  ;將A-7DH的資料-C(進位旗標)的值放在A  **MOV 79H,A** ;將A的資料放進79H  **JNC DIV\_2** ;若C=0時 則跳入DIV\_2  **;(6) IF (79H)(78H) < (7DH)(7CH), THEN**  **; (79H) <- R6, (78H) <- R5**  **MOV 78H,R5** ;將R5的資料放進78H  **MOV 79H,R6** ;將R6的資料放進79H  **DIV\_2:**  **;(7) 商數的處理,**  **; (7BH)(7AH) <- (CPL C)**  **CPL C** ;將進位旗標C反向  **MOV A,7AH** ;將7AH的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7AH,A** ;將A的資料放進7AH  **MOV A,7BH** ;將7BH的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7BH,A** ;將A的資料放進7BH  **;(8) 共計作了 16 次了嗎?**  **DJNZ R7,DIV\_1** ;若R7-1不等於0則跳入DIV\_1  **RET** ;從模組返回  **END** ;程式結束 |

若需要10進制的運算，就需要靠以下的程式將16進制轉10進制，也可使用此程式配合上述的加減乘除，做出10進制的算法。

**16進制轉10進制**

|  |
| --- |
| **;------------------------------------------**  **;(1) FUNCTION: 16BIT HEX TO BCD CODE**  **;(2) INPUT : (7FH)(7EH) -> HEX**  **;(3) OUTPUT : (7DH)(7CH)(7BH) -> BCD**  **;(4) USE REG.: A,R7**  **;------------------------------------------**  **PUBLIC HEX\_BCD16**  ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **HEX\_BCD16:**  **;(1) 清除 BCD 碼之緩衝區(7DH)(7CH)(7BH)**  **MOV 7BH,#00** ;將00放入7BH位址  **MOV 7CH,#00** ;將00放入7CH位址  **MOV 7DH,#00** ;將00放入7DH位址  **MOV R7,#16** ;將16放入暫存器R7  **;(2) 將待轉 HEX 碼 (7FH)(7EH)左旋一個位元至 CY**  **HEX\_1:**  **MOV A,7EH** ;將7EH的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7EH,A** ;將A的資料放進7EH  **MOV A,7FH** ;將7FH的資料放進A  **RLC A** ;將累加器與進位旗標C一起左旋一個位元  **MOV 7FH,A** ;將A的資料放進7AF  **;(3) (7DH)(7CH)(7BH) = (7DH)(7CH)(7BH) \* 2 + CY**  **; 其結果作十進位調整(DA)**  **; 重覆執行 16 次**  **MOV A,7BH** ;將7BH的資料放進A  **ADDC A,7BH** ;將A的值與7B的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7BH的值+C)  **DA A** ;將累加器做十進制調整  **MOV 7BH,A** ;將A的資料放進7BF  **MOV A,7CH** ;將7CH的資料放進A  **ADDC A,7CH** ;將A的值與7C的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7CH的值+C)  **DA A** ;將累加器做十進制調整  **MOV 7CH,A** ;將A的資料放進7CF  **MOV A,7DH** ;將7DH的資料放進A  **ADDC A,7DH** ;將A的值與7D的值相加，加總後放在A(加的時候含進位旗標的數值，即為A+7DH的值+C)  **DA A** ;將累加器做十進制調整  **MOV 7DH,A** ;將A的資料放進7DF  **DJNZ R7,HEX\_1** ;若R7-1不等於0則跳入HEX\_1  **RET** ;從模組返回  **END** ;程式結束 |

* 1. **順向查表法**

在建立表格之後，我們若需要尋找表格的某一個數值，即需要查詢表格的方法，以下即是順向查表法。

**順向查表 : index -> code**

**輸入 : index(R1),table\_name(DPTR)**

**輸出 : code(R2)**

**使用 : R1,R2**

**查表在R1,結果放R2**

|  |
| --- |
| **;----------------------------------------------- --------------------------------**  **;順向查表 : index -> code**  **;輸入 : index(R1),table\_name(DPTR)**  **;輸出 : code(R2)**  **;使用 : R1,R2**  **;查表在R1,結果放R2**  **;----------------------------------------------- --------------------------------**  **PUBLIC FLOOK\_TABLE** ;模組名稱  **ROM SEGMENT CODE** ;共享ROM空間  **RSEG ROM** ;選取可重置的 ROM空間  **FLOOK\_TABLE:** ;模組名稱  **MOV A,R1** ;將R1位置的值放在A  **MOVC A,@A+DPTR** ;將表格內第A(R1)個的數字放於A  **MOV R2,A** ;將A的值放在R2  **RET** ;從模組返回  **END** ;程式結束 |

* 1. **逆向查表法**

若是需要將按鍵對應到表格上的值，就需要另一種” 逆向查表法”，以下為逆向查表法的程式。

|  |
| --- |
| **PUBLIC BLOOK\_TABLE**;模組名稱  **ROM SEGMENT CODE**;共享ROM空間  **RSEG ROM**;選取可重置的 ROM空間  **BLOOK\_TABLE:** ;模組名稱  **MOV R1,#00H**;將00放入R1位址  **BL1:**  **MOV A,R1**;將R1位置的值放在A  **MOVC A,@A+DPTR**;將表格內第A(R1)個的數字放於A  **XRL A,R2**  **JNZ BL2**  **JMP BL3**  **BL2:**  **INC R1**;R1+1  **DJNZ R3,BL1**  **MOV R1,#0FFH**;將FFH放入R1位址  **BL3:**  **RET**;從模組返回  **END**;程式結束 |

* 1. **延遲01ms**

若需要將LCD閃爍，此時就是需要延遲程式的幫助，故以下即為延遲的程式。

* **延遲10ms**
* **功能 : 延遲R4\*0.1ms時間**
* **輸入 : R4**
* **輸出 : None?**
* **使用 : R3,R4**

|  |
| --- |
| **PUBLIC DELAY\_01MS**;模組名稱  **ROM SEGMENT CODE**;共享ROM空間  **RSEG ROM**;選取可重置的 ROM空間  **DELAY\_01MS:** ;模組名稱  **MOV R3,#48**;將48放入R3位址  **NOP** ;無動作  **DJNZ R3,$** ; 將R3-1，若R3不等於0則返回到目前指令，"再" 執行一次目前的指令  **DJNZ R4,DELAY\_01MS** ;將R4-1，若R2不等於0則回到DELAY\_01MS  **RET**;從模組返回  **END**;程式結束 |

* 1. **延遲10ms**
* **延遲10ms**
* **功能 : 延遲 R5 \*10ms時間**
* **輸入 : R5**
* **輸出 : 無 ?**
* **使用 : R3,R4,R5**

|  |
| --- |
| **PUBLIC DELAY\_10MS**;模組名稱  **ROM SEGMENT CODE**;共享ROM空間  **RSEG ROM**;選取可重置的 ROM空間  **DELAY\_10MS:** ;模組名稱  **L\_DELAY:** ;迴圈名稱  **MOV R4,#100**;將100放入R4位址  **S\_DELAY:** ;迴圈名稱  **MOV R3,#48**;將48放入R3位址  **DJNZ R3,$** 將R3-1，若R3不等於0則返回到目前指令，"再" 執行一次目前的指令  **DJNZ R4,S\_DELAY**將R4-1，若R4不等於0則跳到S\_DELAY  **DJNZ R5,L\_DELAY** 將R5-1，若R5不等於0則跳到L\_DELAY  **RET**;從模組返回  **END**;程式結束 |