第Ⅱ部份

小教授微電腦實驗手冊

小教授微電腦中文操作及實驗手冊

著作人　宏碁股份有限公司編輯委員會
主編　施振榮

台北總公司：台北市民生東路977號
電　　話：(02)7691225(十線)
台中聯絡處：台中市中正路103巷6號
電　　話：(042)237253
高雄聯絡處：高雄市民生二路60號5樓之2
電　　話：(07)2825649・2824470

出版者　全華科技圖書股份有限公司
北市龍江路76巷20-2號
電話 581-1300・541-5342
581-1362・581-1347
郵撥帳號：10083 6
發行人　陳本源
印刷者　欣瑜彩色印刷廠
定　價　新臺幣 200 元
再　版　中華民國72年7月

實習準備　微電腦程式設計過程

程式（Program）是由一連串有秩序、有計劃安排之指令組合而成。CPU 根據程式執行一連串之邏輯動作，以產生吾人所期待之結果。

程式在CPU 執行之前必然以0及1的型態存在記憶組（Memory）中，吾人稱此型態之程式爲機器語言程式（Machine language Program），爲了書寫方便通常以十六進制數字表示之，例如Z80CPU之8－bit 指令10101111B(B代表BINARY)，以十六進制表示則爲OAFH(H代表HEXADECIMAL)，第一個字爲阿拉伯數字表示下面爲數字資料。

以機器語言書寫程式不容易看出其實質上之意義，所以各微處理器製造廠家都將其CPU 指令，按特性加以分類，以具有意義的文字符號來代替每一個機器語言指令，例如Z80CPU指令70若以文字LD A,L（Load data into Reg.A from Reg.L）來表示則一目了然，這種文字寫法可用另一套組合式（Assembler）將其組合成CPU可接受之機器語言，稱之爲組合語言程式（Assembly language）。

以組合語言書寫程式固然方便，但是要輸入記憶組供CPU執行還是必須先將其編譯成機器語言，在有微型電腦發展系統可資運用的場合可以利用其內存組合式（resident Assembler）來編譯，但在MPF－I微型電腦中，使用者必須自行編譯成機器

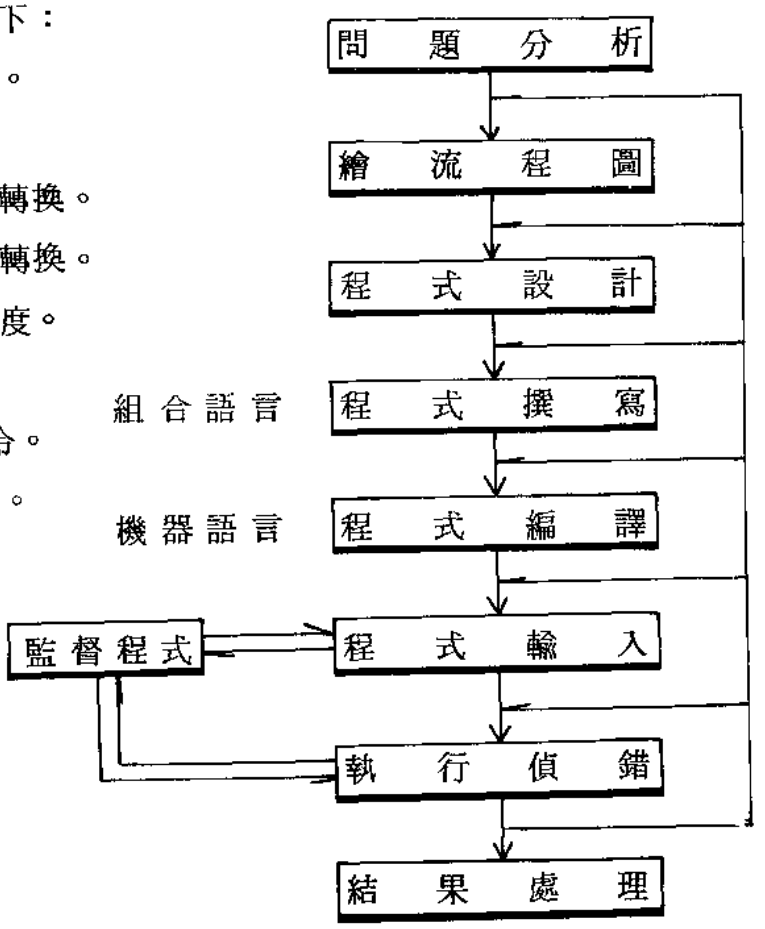
語言，再輸入電腦中執行。

不論是多繁複的程式，都是由許多小程式組織而成，或者說是由主程式配合許多副程式組合而成。指令的數目雖然有限，但是其組合却幾近於無限。所以軟體的設計是千變萬化的。同一問題有無數種解法。由於要求的條件與限制條件的不同，程式設計者應配合擁有的條件做出較適切的設計，圖2-A-1表示一般微電腦程式之設計過程。從問題分析開始至完成程式設計也因爲問題與要求的不同而繁簡不一，但設計者應多充實相關知識，磨練程式設計能力則是完成較佳設計必要的條件。

1. 問題分析

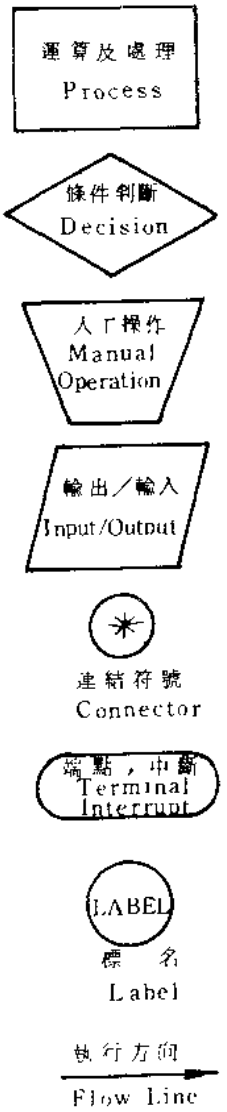
簡單的問題只要簡單的將題目之輸入與輸出及要求功能下定義清楚即可着手程式設計，或者只要將現有程式稍微修改，或者套用與組合，但是較複雜的程式譬如監督程式、編譯程式、或控制與特殊應用的程式則需要週詳的分析後方能着手程式設計，一些必要分析的項目如下：

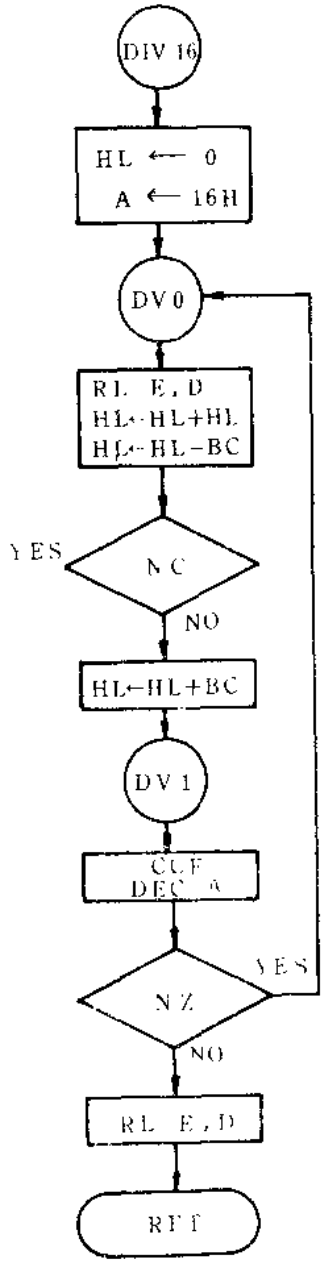
(1) 問題的要求與性質。
(2) 已知的條件。
(3) 資料來源的型式與轉換。
(4) 資料輸出的型式與轉換。
(5) 資料的種類與精確度。
(6) 執行時間的要求。
(7) CPU的性能與指令。
(8) 記憶組容量的大小。
(9) 解答的可能性。
(10) 解答的方案。
(11) 價值分析。
(12) 結果的處理等。



2. 繪製流程圖

流程圖能夠將一個繁複的程式以圖解的方式將其來龍去脈有條有理的表現出來，便利設計者做查核整理的工作，也可提供旁人對此程式的瞭解。流程圖的繪製可粗可細，粗至一個方塊符號代表一段副程式，細至一個符號僅表示一個指令，通常一個複雜的程式可先用粗略的系統流程圖來描述此程式的大綱，其次再就各部份作詳細的流程圖。流程圖最大的好處是能將程式執行的順序以箭線表示，並且用不同的符號表示程式在該階段所執行的性質，對於熟練的程式設計者，設計簡單的程式可能不必繪製流程圖即能著手程式設計之工作，圖 2 - A - 2 爲流程圖使用之一般符號及範例。





3. 程式設計

程式的種類很多譬如簡單的數學運算方程式，輸入信號及輸出信號的轉換與產生程式，資料的編碼或解碼，週邊設備之趨動程式，或者繁複的編譯程式，監督程式，系統控制以及其他特殊應用之程式。每一類程式均有其特殊的設計技巧，勤練與充實是獲致成功的唯一途徑。

一般程式設計應攷慮的事項如下：

(1) 輸入信號或資料的獲取。
(2) 輸出資料或信號的安排與產生。
(3) 主程式之邏輯分析及算法式。
(4) 主程式與副程式之關係。
(5) 暫存器之使用。
(6) 主程式之記憶位址分配。
(7) 副程式之記憶位址分配。
(8) 數列資料之記憶位址分配及指標索引方式。
(9) 程式初值及常數之設定。
(10) 程式變數之設定。
(11) 執行時間與時序的攷慮。
(12) 記憶容量大小的限制。
(13) 資料之精確度與長度。
(14) 是否有現成程式可資應用或參攷。
(15) 其他應特殊攷慮事項。

4. 程式撰寫

程式撰寫以組合語言爲主，Z80之指令與組合語言格式將在實習準備(B)及(C)中分別敍述本節僅說明程式撰寫的格式及程式之整理。程式分成四大部份分別爲標名（ label），操作碼（ Opcode ），操作元（ Operand ）及說明（ Comment ）如下例

所示：

```
標  名  操作碼與操作元          說    明
LABEL  OPCODE & OPERAND    COMMENT

DTB4   LD B,16
DB3    SRL H
       RR L
       RR D
       RR E          ; ROTATE .HL.DE. RIGHT
       LD A,H
       CALL DB1
       LD H,A        ; CORRECT .H.
       LD A,L
       CALL DB4
       LD L,A        ; BINARY CORRECT .L
       DJNZ DB3
       RET

;BINARY CORRECT ROUNTINE
DB4    BIT 7,A
       JR Z DB1      ; IF BIT 7 OF .A. =1 ,SUB. FROM 30H
       SUB 30H
DB1    BIT 3,A
       JR Z DB2      ; IF BIT 3 OF .A. =1 ,SUB. FROM 03H
       SUB 3
DB2    RET
```

上例的程式若未加特別說明，不易看出其性質，若加上說明，則一望生義，對於繁複的程式而言，說明更屬重要，下面爲程式加上程式名稱及功能說明後的結果：

從附加之說明事項可以明白的看出此段程式之功能，便於呼叫（CALL）及查核整理。

5.程式編譯組合

利用微型電腦發展系統之組合程式（Assembler）來編排組合我們的原始程式，固然是最有效的方法，但是初學者或沒有發展系統可以運用的設計者仍須自行編譯，以人工編譯的程序如下：

(1) 查閱組合語言與機器語言對照表，將程式中每一指令逐一譯成機器語言，說明部份不必譯。

```
; ** 4 DIGIT BCD TO BINARY CONVERTION ROUNTINE **
; ENTRY   BCD DATA IN  HL
; EXIT    BINARY DATA IN .DE.
; REGISTER CHANGED : AF BC DE HL

DTB4  LD B,16
DB3   SRL H
      RR L
      RR D
      RR E          ; ROTATE .HL.DE.  RIGHT
      LD A,H
      CALL DB1
      LD H,A        ; CORRECT .H.
      LD A,L
      CALL DB4
      LD L,A        ; BINARY CORRECT .L.
      DJNZ DB3
      RET

;BINARY CORRECT ROUNTINE
DB4   BIT 7,A
      JP Z DB1      ; IF BIT 7 OF .A. =1 ,SUB. FROM 30H
      SUB 30H
DB1   BIT 3,A
      JP Z DB2      ; IF BIT 3 OF .A. =1 ,SUB. FROM 03H
      SUB 3
DB2   RET
```

(2) 決定程序之起始位址，按序將各指令編上位址（各指令之第一個Byte ），此時將JR，DJNZ之相對位移數及JP，CALL之目的位址預留應佔有之Byte位置。計算各相對位移數將其填上則程式編譯完成，相對位移數之簡易計算公式。

位移數之簡易計算公式：

位移數△△＝跳越目的位址－下一指令位址

若其值爲正則直接取其值，若結果爲負則以 100 H減去其值（取 2 之補數），將結果做爲此指令之操作元。

例如本例位置 0014H之指令DJNZ DB3 初步編譯爲 10 △△，然後計算△△值

△△ = 0002H（目的位址）－ 0016H（下一指令位址）

= － 14（負數）

∴△△= 100H － 14H = ECH

因此DJNZ DB3 應編譯成 10 EC。

另外　位址 0019H 之指令 JR　Z，DB 1 初步編譯 28 △△

△△＝ 001 DH（目的位址）－ 001BH（下一指令位址）＝ 2

因此 JR　Z, DBL 應編譯成 2802 。

位址	機器語言	標名	操作碼及操作元	說　　明

```
                        ; ** 4 DIGIT BCD TO BINARY CONVERTION ROUTINE **
                        ; ENTRY : BCD DATA IN . HL.
                        ; EXIT  : BINARY DATA IN . DE.
                        ; REGISTER CHANGED : AF BC DE HL

0000    0610    DTB4    LD B,16     ;B=BIT COUNT
0002    CB3C    DB3     SRL H
0004    CB1D            RR L
0006    CB1A            RR D
0008    CB1B            RR E        ; ROTATE . HL. DE.  RIGHT
000A    7C              LD A,H
000B    CD1D00          CALL DB1
000E    67              LD H,A      ; CORRECT . H.
000F    7D              LD A,L
0010    CD1700          CALL DB4
0013    6F              LD L,A      ; BINARY CORRECT . L.
0014    10EC            DJNZ DB3
0016    C9              RET
                        ;
                        ;BINARY CORRECT ROUNTINE
0017    CB7F    DB4     BIT 7,A
0019    2802            JR Z DB1    ; IF BIT 7 OF . A.  =1 ,SUB  FROM 30H
001B    D630            SUB 30H
001D    CB5F    DB1     BIT 3,A
001F    2802            JR Z DB2    ; IF BIT 3 OF  A.  =1 ,SUB. FROM 03H
0021    D603            SUB 3
0023    C9      DB2     RET
```

6. 程式之輸入

將程式輸入到MPF－I微型電腦預定之記憶位址內，監督程式將協助我們做這項工作（請參閱MPF－I 之使用），可從鍵盤按入或從磁帶讀入。程式輸入後應加以查核，否則任何一個 Byte 錯誤都將使程式執行發生錯誤，多餘的指令或資料得以"NOP"指令彌補其空位，遺漏的指令或資料則以集體資料傳送之方式騰出空位再將遺漏的指令按入，或者重按，當程式修正再輸入時應特別注意位址變動後是否影響及跳越指令（JP、JR、DJNZ、CALL等），如果涉及應同時修正之。

7.程式之執行與偵錯

執行程式之第一步驟是設定程式之初值，然後將PC（Program Counter ）設定到程式之起始位址，再按執行鍵“ GO ”，卽可執行此程式，程式執行後查核其結果，如果錯誤，則應用監督程式之功能分段執行逐步查核修正，或者重新檢查定義及程式找出問題，修正後重新執行之。

8.程式完成

程式執行無誤後應留下正確的書面程式（ Program List ），並編上號碼錄入磁帶以資運用，必要時加以簡化以節省記憶組佔用位址，或者加以修正以節省執行時間。

實習2-1資料傳送練習

主旨：1.練習資料傳送指令之使用

2.練習資料之初值設定

3.練習程式之編譯與輸入執行

需用時間：四小時

相關知識：

1. 資料傳送指令以LD（LOAD）指令爲主，可以一次傳送8 Bit或16 Bit之資料。此外EX，EXX，PUSH，POP指令一次傳送16 Bit資料，至於集體資料傳送指令LDI，LDIR，LDD，LDDR則可以一次傳送較多資料。
2. LD指令應有兩個操作元（Operands），第一個操作元代表資料存入之位置（暫存器或記憶組）；稱之爲目的位置（Destination），後一個操作元代表欲傳送之資料（Immediate 資料）或資料原來存在之位置（暫存器或記憶組），稱之爲來源（Source）。

   例如：LD A，B 表示將暫存器B之內容傳送至暫存器A。暫存器A爲目的位置，暫存器B爲來源。
3. 資料傳送的方向不外乎下列各種：

   (1)暫存器←暫存器 例如：LD A，B； LD HL，BC

   (2)暫存器←記憶組 例如：LD A，(HL)； POP AF

   (3)暫存器←immediate data 例如：LD A，25H；LD HL，125AH

   (4)記憶組←暫存器 例如：LD(HL)，A；PUSH BC

   (5)記憶組←記憶組 例如：LDD； LDIR

   (6)記憶組←immediate data 例如：LD（HL），5BH

練習2－1－1

試用組合語言做一程式分別設定各暫存器之內容如下：A＝0，B＝1，C＝2，D＝3，E＝4，H＝5，L＝6（採用8BIT LD指令，一次傳送1 BYTE）

步驟1：撰寫組合語言程式填入下列表格，最後一條指令RST 38 H使程式執行至此即回到監督程式。

步驟2：參考8-BIT LD指令表將此程式自1800H之位址開始編譯成機器語言，並編上各指令之位址。

步驟3：備妥MPF－I微型電腦，將此程式從鍵盤上輸入，重新檢查無誤後，將PC（PROGRAM COUNTER）設至起始位址1800H，並執行此程式。

步驟4：按Reg鍵檢查各暫存器內容是否正確，否則回到步驟1重新檢查修正之。

記憶位址	機器語言	組合語言
1800 H	3E00	LD A,0
	FF	RST 38H

練習2－1－2

試用組合語言做一程式分別設定各暫存器之內容如下：B＝12，C＝34，D＝56，E＝78，H＝9，L＝A（採用16BIT LD指令，一次傳送兩個BYTES）。

步驟1：同上題（撰寫組合語言程式）。

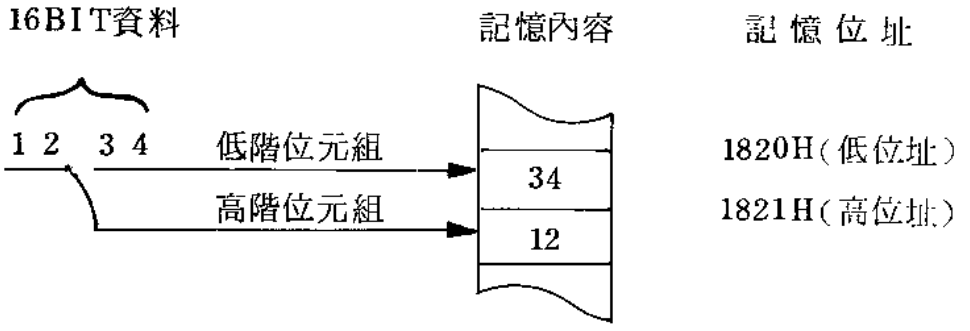
步驟2：參考16BIT LD指令表，將此程式自1820H之位址開始編譯成機器語言並編上各指令之位址。

步驟3：同上題（輸入程式）設PC為1820H並執行之。

步驟4：同上題（檢查結果）

附註：16BIT之資料分成2組8 BIT數據，高階位元組（HIGH ORDER BYTE)

在記憶組內之較高位址，低階位元組（LOW ORDER BYTE）在較低記憶組位址，例如一組16BIT資料1234H存在位址1820H及1821H之情形如下：



位　　址	機器語言	組合語言
1820 H	01 34 12	LD BC,1234H
1823 H		
	FF	RST 38H

例題2－1－1

作一程式將記憶位址1850H～186FH之內容設定爲0。

說明：1.假設以8BIT LD指令將0逐一傳送至所有的目的位址需要時傳送32次(20H)故以環路（LOOP）方式來設計程式較爲簡便。

2.以Reg B做爲環路計數器（LOOP COUNTER），在環路程式執行前先設定其值等於所要執行之次數（20H），每一次環路分別設定一個記憶位址之內容爲0，故以HL做爲記憶位址之指標，設定其初值爲所要清除之記憶組的起始位址1850H，每次環路均將HL加1使得下一次LOOP（環路）時能指向下一個記憶位址，並且將B值減1，如果B值等於0表示所有LOOP已經做完否則再度執行此LOOP。

3.流程圖及程式如下所示

位　址	機器語言	標　名	操作碼及操作元	說　　明
1800			LD B,20H	設定LOOP COUNTER爲32
			LD HL,1850H	設定清除起始位址
			XOR A	使A＝0
		LOOP	LD(HL),A	將0送入HL所指位址

			INC HL	將HL進1
			DEC B	將B減1
			JR NZ, LOOP	若B不爲0回到LOOP
	FF		RST 38H	程式回到監督程式

練習2－1－3

試將例題2－1－1之程式編譯成機器語言，輸入MPF－I執行之並檢查位址1850H～186FH之內容是否已清除爲0，否則修正錯誤並重新執行之。

練習2－1－4

試修改例題2－1－1之程式，以設定記憶位址1840H～184FH之內容分別爲0, 1,2,3, ……,F。

（提示：變更LOOP COUNTER及起始位址之值，並使Reg.A於每一次LOOP均加1）將程式記入下列表格

位　址	機器語言	標　名	操作碼及操作元

實習2-2基本算術與邏輯操作指令

主旨：1.熟悉8 BIT算術與邏輯操作指令。

2.瞭解記憶位地標示方式。

3.瞭解狀態旗號（ flag ）之意義。

4.練習CPU暫存器與記憶體資料安排。

需用時間：四小時

相關知識：

1. 8 BIT算術與邏輯操作指令

Z80 CPU作8 BIT 算術與邏輯運算時，以A暫存器（ ACCUMULATOR或A REGISTER）爲中心而處理。CPU中A,B,C,D,E,H,L各REGISTER）均可與A REGISTER運算。若欲使記憶體內之資料與A REGISTER作邏輯或算術運算，則位地由HL或 IX, IY指定。下列指令之意義如右方所述。

(1) ADD A ；將A REGISTER之資料自己相加，結果加倍或向左移一位。

(2) ADC B ；將A REGISTER與進位旗號加入A。

(3) SUB C ；將A REGISTER之資料減去C REGISTER之資料。

(4) SBC （HL） ；將A REGISTER之資料減去HL所指位址之記憶內容，再減去進位。組合語言中，括號內之內容代表位址。

(5) AND D ；將D REGISTER內之8 BIT資料與A REGISTER內之8 BIT資料AND起來，結果存入A。

(6) OR OFH ；將A REGISTER資料直接與OFH資料OR，結果置於A。

(7) XOR A ；將A REGISTER資料自己與自己EXCLUSIVE OR。結果存入A。（因A與A必相等，EXCLUSIVE OR後必爲0）。

(8) INC H ；將H REGISTER之資料加1（ INCREMENT ）

(9) INC (IX) ；將用IX指定位址之記憶內容加1。

(10) DEC C ；將C REGISTER減1。

(11) DEC(IY+3) ；將IY內之資料加3，當作位址，將該位址之記憶內容減1。

2 資料位置標示方式(ADDRESSING MODE)

上列組合語言指令中，所處理之資料存放位置標示方法可歸納爲下列數種，其他種方式請參考Z 80 CPU技術手册。

(1) 直接標示暫存器名稱(REGISTER ADDRESSING)

例如ADC A,B指令中ADC代表操作碼，決定做什麼。右方之A字表示資料加到A REGISTER。最右方之B標示資料從B REGISTER加來。

(2) 以16BIT暫存器標示記憶位址，REGISTER INDIRECT ADDRESSING

例SBC A,(HL)，最右方之(HL)並非表示要將HL減入A REGISTER，而是由CPU先去找HL REGISTER，將其中16BIT資料取出作爲住址，再去記憶中找該位址內之8 BIT資料，將其減入A REGISTER。IX與IY稱爲指標暫存器(INDEX REGISTER)。用IX與IY來指定記憶位址時，可加上小於等於+127而大於等於－128之8 BIT資料(以2的補數代表負數)。例如下列兩個指令可將IX所指定位址之記憶內容之資料與IX+2之位址內8 BIT資料相加，結果儲存在A REGISTER。

```
LD A,(IX)
ADD A,(IX+2)
```

(3) 直接數據標示(IMMEDIATE ADDRESSING)

例如OR OFH，OR操作碼右方直接標示OFH之十六進制數字，表示直接將OFH與A REGISTER作邏輯OR運算。因此OFH數據與程式同時儲存在記憶體中。CPU實際上是以PC(PROGRAM COUNTER)當作位址資料去讀取此數據。下列指令均爲直接數據標示法。

```
LD B,8
ADD A,44H
SUB A,φA4H
```

3. 狀態旗號(FLAGS)