第4章

應用指令介紹及設計

單元八　應用指令之傳送指令
單元九　應用指令之算數、比較指令
單元十　應用指令之位移、旋轉指令
單元十一　資料暫存器的應用
單元十二　應用指令之邏輯、數碼算指令
單元十三　應用指令之資料處理、便利指令
單元十四　應用指令之外部設定、顯示指令

本章單元八到單元十四是將常用的位元及字元指令，以分類的方式做介紹，雖然指令很多，但不必馬上全部學會。可以在需要時，再翻開來參考即可。本章所介紹的指令是以 FX2 為主，至於其它廠牌，請自行參閱各機種的操作手冊。

單元八　應用指令之傳送指令介紹

壹 學習目標

- 由下列的介紹，你能了解 PLC 所使用的資料格式。你也能任意的轉換各種資料格式。
- 利用所附圖表，你能了解各種傳送指令的意義及使用方法。
- 你能應用傳送指令設計程式。

貳 相關知識

一、基礎觀念介紹

1. **應用指令**：由於可程式控制器的按鍵數量的限制，無法將所有的指令都以一個按鍵來代替。所以就將常用的、較簡單的指令分別使用一個按鍵來代表，這些指令就稱之為基本指令，大部份在第三章已介紹過。其餘較複雜的指令則以功能鍵的方式存在記憶體中。要得到這些指令，必須先按 FNC（功能）鍵，再連續按兩或三位數字即可。例如 FNC 1 0 為 CMP 指令、FNC 1 2 為 MOV 指令等等，這些指令稱之為應用指令。
2. **位元指令**：應用指令數量多於基本指令，且應用廣泛，在功能較為複雜的設計當中，往往都必須以這些指令來設計，才能符合電路的要求。因此在學會了基本指令之後，本章將以分類的方式，繼續介紹應用指令。應用指令可以分成兩類：一類為與基本指令有相同格式的位元指令；另一類為可同時對多個位元做運算或控制的字元指令。所謂位元指令，就是該指令一次祇能針對單一元件來作控制或驅動。例如 ALT Y0 就是祇能驅動或復歸 Y0 這單一元件而已。
3. **字元指令**：基本及位元指令一次祇能針對單一元件來作控制或驅動。但是，想想看，假使有一系統具有 64 個輸入元件，若要分別檢查所有的輸入元件的狀態，那不是須要用 LD 指令 64 次嗎？如果是這樣，那就太沒效率了。因應這個缺失，PLC 另外提供了字元指令。所謂字元指令，就是指該指令可以同時對多個元件（多

個位元）來作控制或驅動。以前述的例子來說，若使用 MOV 指令，則一次可載入16個元件（16位元），所以總共只須四個MOV指令就夠了。其實字元指令的取用元件（位元）數量，是因各機種的CPU而異的。早期的PLC是8位元，而現今則大多為16或32位元。FX2系列的字元使用方式彈性較大，它一次可以以4的倍數位元(4、8、12、16、20、24、28、32)來做控制，例如：K2Y0 表示可同時對 Y0～Y7(K2 表示 2 × 4 = 8 位元)做控制。

4. **應用指令格式的定義**：一行程式通常是由運算碼（指令）及運算元所組成的，例如 LD X0，其中的 LD 稱為運算碼，而 X0 則稱為運算元。在基本指令中，運算元都祇有一項，一般都是指示元件編號-例如X0、Y0、T0...等。但在應用指令中的運算元可能有兩項以上，且運算元可能是元件編號，也可能是一數值資料。

實例 MOV K25,K1Y0　　K25 = 25_{10} 為一數值資料

第一運算元 10 進位數值　第二運算元 位元號碼　　K1Y0=Y0～Y3

5. **一應用指令**，通常都是分成若干行鍵入的，例如 MOV K25,K1Y0 要分成三行來鍵入。

位　址	指　　令
0	MOV
	K25
	K1Y0

如此一來，程式的列表就比較長，因此本章為了簡化列表，將只以一行來顯示各種指令，且運算元間以','隔開。讀者在實際鍵入時，須依上表的方式來鍵入。

位　址	指　　令
0	MOV K25,K1Y0

6. 有些指令具有處理 32 位元的能力，使用時只要在指令前加上'D'就可以了。

實例 DMOV K0,K8Y0。

7. **上緣觸發**：當只要輸入信號為 ON 時，指令就會動作的稱之為準位觸發，而當輸入信號須由OFF變為ON時，指令才會動作的則稱之為上（前）緣觸發。應用指令中的某些指令只要在指令的後端

加上 P 即可當成上（前）緣觸發指令使用(微分指令)，例如 MOVP、INCP。圖 4-1 是只有在按下 X0 開關（由 OFF 變 ON）的時候，D0 才會被加 1（INC 的功能），之後就算你一直壓著 X0 開關，INC 也不會再動作（加 1），除非你放掉 X0 開關之後，再重新按 X0 開關，那 D0 才會再被加 1。

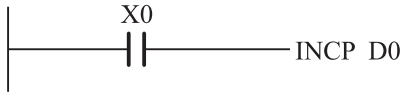


圖 4-1

8. **掃描時間**：PLC 程式執行的方式是藉由 CPU 不斷的、依序的、重覆的掃描每一行指令，再根據指令完成所需的動作。CPU 由第一行指令一一掃描到最後一行指令所須的時間稱為掃描時間，掃描時間過長（程式太長），將會遺漏某些輸出、入動作，影響 PLC 的效率。還好新一代的 PLC 已使用速度極快的 CPU，所以這種問題，已很少發生。

9. **資料暫存器的（D）應用**：FX2 提供了大量的記憶體區來做為暫存器，讓使用者可以暫存資料。它是以 D 之後加上一個號碼來代表，例如 D0、D200 等。每一個記憶體位置均有 16 位元的容量。但是它與一般電驛不同的是，一般電驛可以以位元（一次一位元）或字元（一次多個位元）為單位來存取，D 只能以一次十六位元的方式來存取。FX2 總共提供了從 D0～D511 等 512 個資料暫存器位置供使用者使用。其中D200～D511 是僅讀（只能讀取）暫存器，D0～D199 為讀寫（可讀可寫）暫存器。

二、數值資料

1. 數值資料表示

⑴十進制：以數字 0～9 及位階加權的方式來表示數值大小的方法。

實例 $856 = 8 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$ 10^2、10^1、10^0 為各位階加權值，所以 8 稱為百位數（$10^2 = 100$），5 稱為十位數（$10^1 = 10$），6 稱為個位數（$10^0 = 1$）。

⑵二進制：祇以數字 0、1 及位階加權值的方式來表示數值大小的方法。

實例 $101011_2 = 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$ 其中 2^5、2^4、2^3、2^2、2^1、2^0 為各位階加權值。二進制是以

bitn 來表示各位階，即 bit5、bit4、bit3 、bit2、bit1、bit0 ，bit0 稱之為 LSB（最低位元），bit5 稱之為 MSB（最高位元），上例中的 bit5 = 1，bit4 = 0，bit3 = 1，bit2 = 0，bit1 = 1，bit0 = 1。

⑶**十六進制**：以數字 0～9，A～F 及位階加權的方式來表示數值大小的方法。例: $A8E = A \times 16^2 + 8 \times 16^1 + E \times 16^0$ 16^2、16^1、16^0 為各位階加權值。其中 A 即十進制的 10，E 即十進制的 14。

⑷**二進制的十進位（BCD）**：與二進制一樣，以 0、1 及位階加權值的方式來表示數值大小的方法。但不同的是將數值分成每 4 個位元一組（一個 nibble 或一個字），每一字皆為 0000～1001 的某一數值（十進位），當數值超過 1001 時則以 2 個字來表示，即 00010000。

實例 $15_{10} = 0001111_2 = \underline{0001}\ \underline{0101}_{BCD}$。

$108_{10} = 1101100_2 = \underline{0001}\ \underline{0000}\ \underline{1000}_{BCD}$。

2. 數值資料轉換

⑴二進制化成十進制

實例 $10111_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 23_{10}$

⑵十六進制化成十進制

實例 $2A1_{16} = 2 \times 16^2 + A \times 16^1 + 1 \times 16^0 = 2 \times 16^2 + 10 \times 16^1 + 1 \times 16^0 = 673_{10}$

⑶十進制化成二進制

實例 $84_{10} = ?_2$

2 | 84
2 | 42……0
2 | 21……0
2 | 10……1
2 | 5……0
2 | 2……1
1……0

$\therefore 84_{10} = 1010100_2$

⑷十進制化成十六進制

$625_{10} = ?_{16}$

16 | 625
16 | 39……1
2……7

$\therefore 625_{10} = 271_{16}$

⑸二進制化成十六進制：將每一個 nibble（4bits）的二進位值轉成十六進制值即可。

實例 $1011011011010111_2=?_{16}$

$\therefore \underline{1011}\ \underline{0110}\ \underline{1101}\ \underline{0111}_2=B6D7_{16}$

B 6 D 7

⑹十六進制化成二進制：將每一個十六進制值轉成一個 nibble（4bits）的二進位值。

實例 $8A9_{16}=?_2$

$\underline{8}\ \underline{A}\ \underline{9}_{16}=100010101001_2$

1000 1010 1001

$\therefore 8A9_{16}=100010101001_2$

⑺ BCD 與十進制的轉換：將每一個十進制值對應成一個 nibble（4bits）的二進位值。

實例 $839_{10}=?_{BCD}$

$\underline{8}\ \underline{3}\ \underline{9}_{10}=100000111001_{BCD}$

1000 0011 1001

$\therefore 839_{10}=100000111001_{BCD}$

$001101110101_{BCD}=?_{10}$

$\underline{0011}\ \underline{0111}\ \underline{0101}_{BCD}=375_{10}$

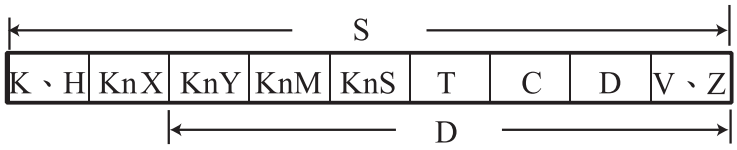
3 7 5

$\therefore 001101110101_{BCD}=375_{10}$

三、指令介紹

1. 傳送指令：MOV/MOVP（FNC 12）-------DMOV/DMOVP(32 位元)

⑴格式：MOV S,D　S：來源元件或數值　D：目標元件



⑵意義：MOV 是將一來源元件的狀態或數值傳送（拷貝）到目標元件。

⑶舉例：

①階梯圖

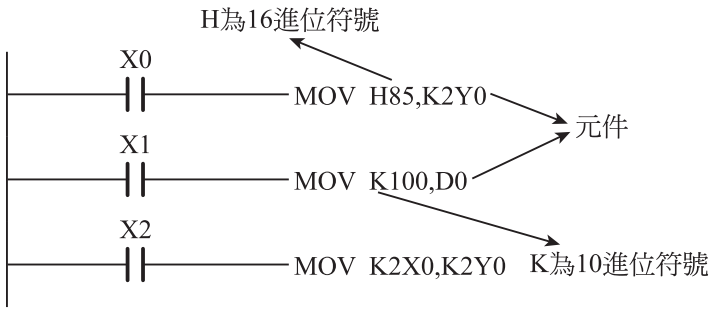


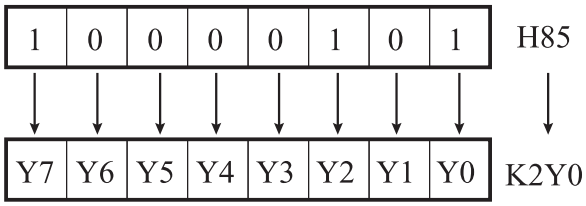
圖 4-2

②程式

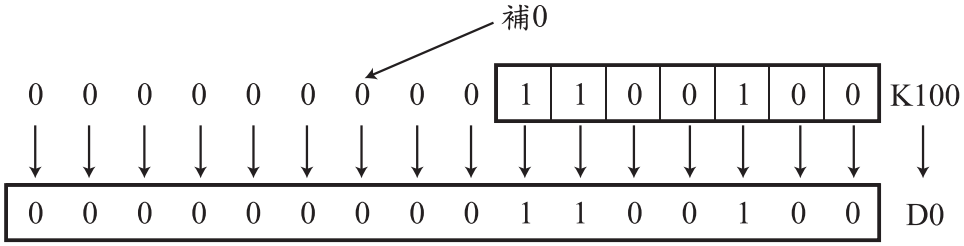
位址	指　　令
0	LD　　X0
1	MOV　　H85,K2Y0
6	LD　　X1
7	MOV　　K100,D0
12	LD　　X2
13	MOV　　K2X0,K2Y0
18	END

③說明

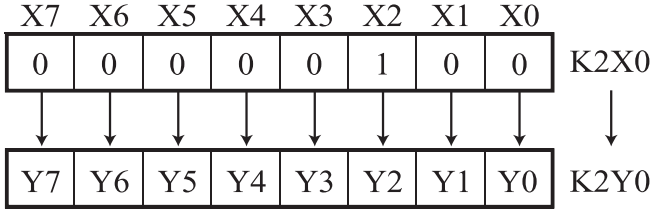
A.當按下 X0 開關時，MOV 指令將十六進制數值 85 輸出到 Y0～Y7，結果為 Y0、Y2、Y7 動作，其餘的輸出復歸。



B.當按下X1開關時，MOV指令將十進制數值100輸出到D0，結果為 D0=K100。

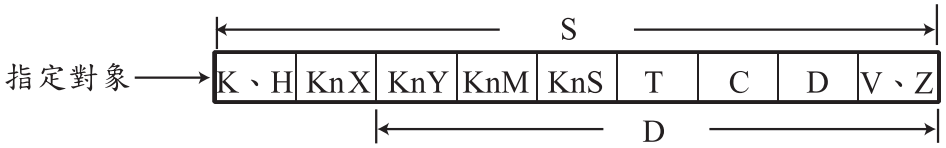


C.當按下 X2 開關時，MOV 指令將 X0～X7 的狀態取入，再輸出到 Y0～Y7。結果為除了 Y2 動作外，其餘的輸出皆復歸。



2. 反相傳送指令：CML/CMLP（FNC 14）-------DCML/DCMLP（32位元）

⑴格式：CML S,D　S：來源元件或數值 D：目標元件



⑵意義：CML 是將一來源元件的狀態或數值先反相後再傳送（拷貝）到目標元件。

⑶舉例：

①階梯圖

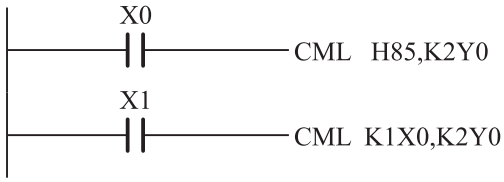


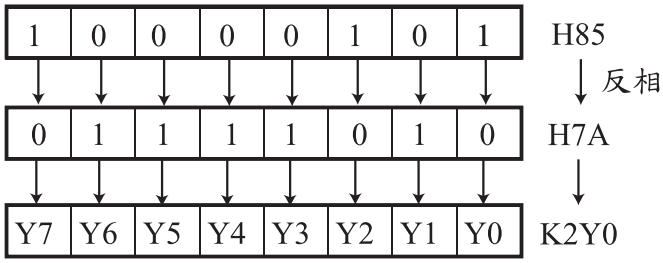
圖 4-3

②程式

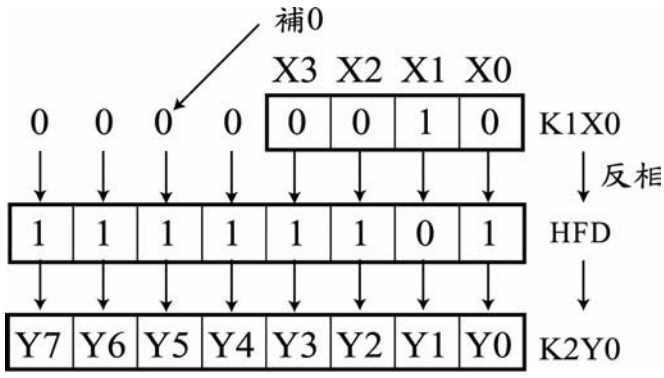
位址	指　令
0	LD　X0
1	CML　H85,K2Y0
6	LD　X1
7	CML　K1X0,K2Y0
12	END

③說明

A.當按下 X0 開關時，CML 指令將十六進制數值 85 取入，經反相後，再輸出到 Y0～Y7，結果為 Y1、Y3、Y4、Y5、Y6 動作，其餘的輸出復歸。



B. 當按下 X1 開關時，CML 指令將 X0～X3 的狀態取入，經反相後，再輸出到 Y0～Y7。結果為除了輸出 Y1 復歸外，其餘的輸出皆動作。



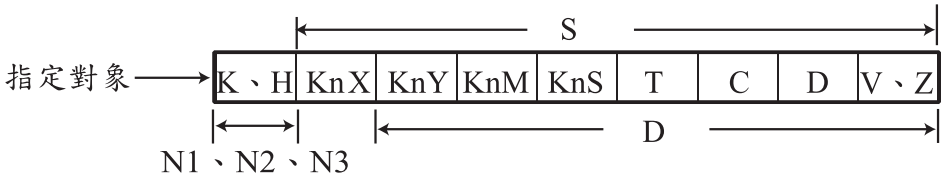
3. 字傳送指令：SMOV/SMOVP（FNC 13）

⑴格式：SMOV S,N1,N2,D,N3　S：來源元件 D：目標元件

N1：來源元件的啟始字(nibble)

N2：欲傳送字(nibble)數

N3：指定目標元件的啟始字



⑵意義：SMOV 是將一來源元件的某些字的值傳送（拷貝）到目標元件的某些字中。

⑶舉例

①階梯圖

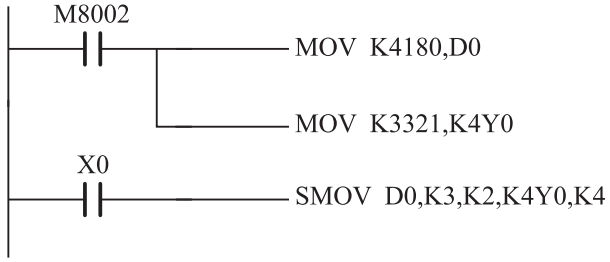


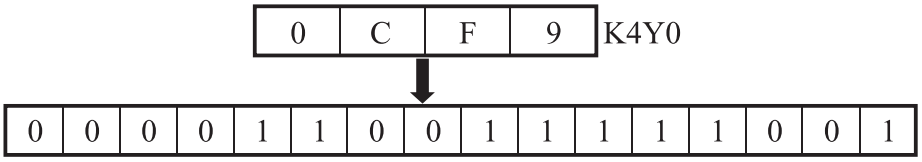
圖 4-4

②程式

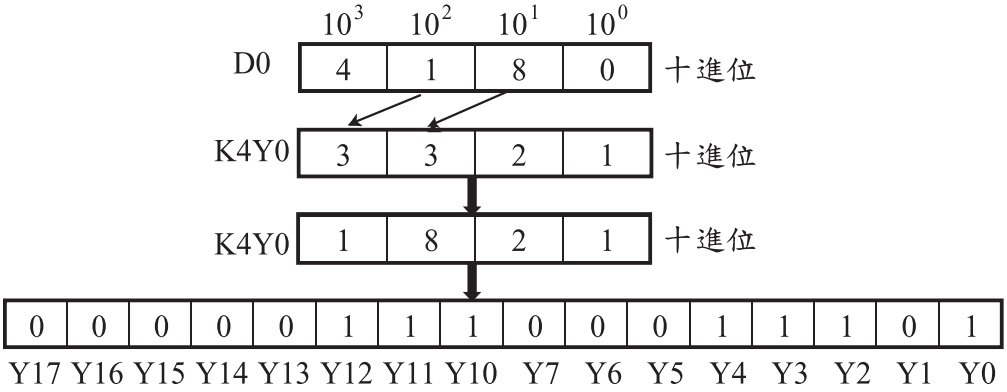
位址	指　令	
0	LD	M8002
1	MOV	K4180,D0
6	MOV	K3321,K4Y0
11	LD	X0
12	SMOV	D0,K3,K2,K4Y0,K4
23	END	

③說明

A.當 RUN 後，D0 被放入數值 K4180，Y0～Y17 放入 K3321 (HCF9)。即 Y0、Y3、Y4、Y5、Y6、Y7、Y12、Y13 動作，其餘的輸出不動作。



B.當按下X0 開關時，SMOV 指令將D0 的第 3 字開始的兩個字傳送(拷貝)到 K4Y0 的第 4 字開始的兩個字中。



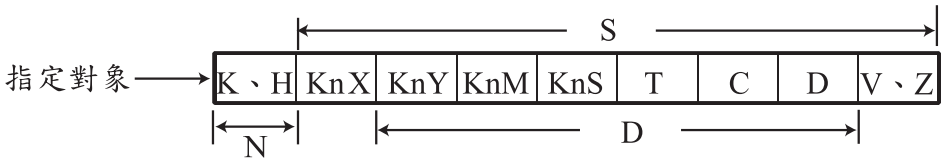
C.結果為除了Y0、Y2、Y3、Y4、Y10、Y11、Y12動作外，其餘的輸出皆復歸。

*4.*複數傳送指令：BMOV/BMOVP（FNC 15）

⑴格式：BMOV S,D,N　S：來源元件啟始編號

D：目標元件啟始編號

N：傳送的數量



⑵意義：BMOV 是將某些來源元件的狀態傳送（拷貝）到某些目標元件中，傳送的數量由 N 值決定。

⑶舉例

①階梯圖

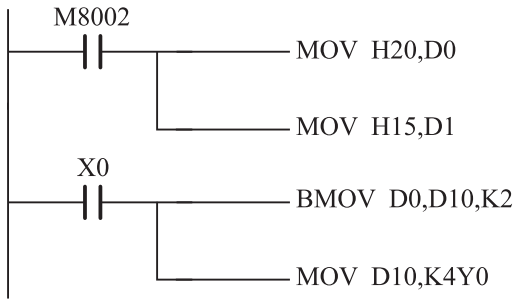


圖 4-5

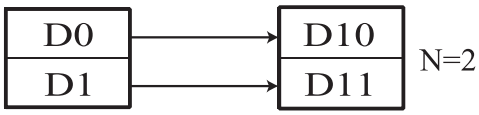
②程式

位址	指　令
0	LD　M8002
1	MOV　H20,D0
6	MOV　H15,D1
11	LD　X0
12	BMOV　D0,D10,K2
19	MOV　D10,K4Y0
24	END

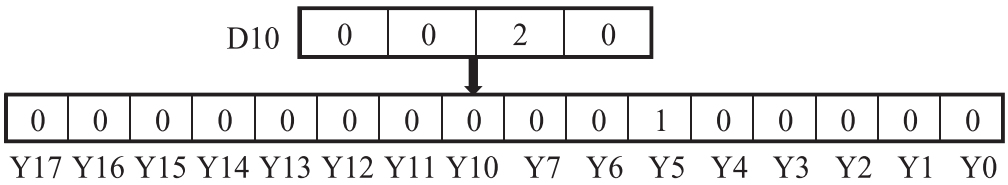
③說明

A.當 RUN 後，D0、D1 分別被放入數值 H20、H15。

B.當按下 X0 開關時，BMOV 指令將 D0 及 D1 兩個來源元件傳送(拷貝)到 D10、D11 兩個目標元件中。



C.再將 D10 傳送到 Y0～Y17 中，結果為除了 Y5 動作外，其餘的輸出皆復歸。

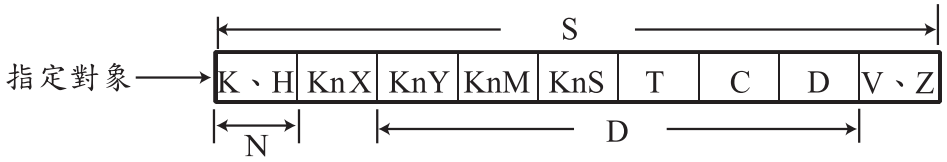


5.多點傳送指令：FMOV/FMOVP（FNC 16）

⑴格式：FMOV S,D,N　S：來源元件或數值

D：目標元件啟始編號

N：傳送的數量



⑵意義：FMOV 是將一來源元件的狀態或數值同時傳送（拷貝）到多個目標元件中，傳送的數量由 N 值決定。

⑶舉例：

①階梯圖

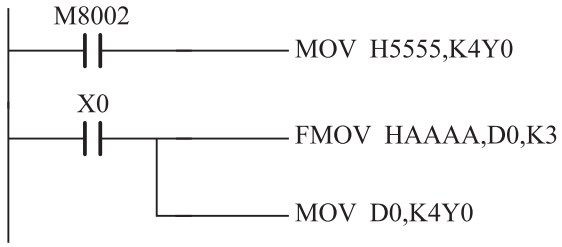


圖 4-6

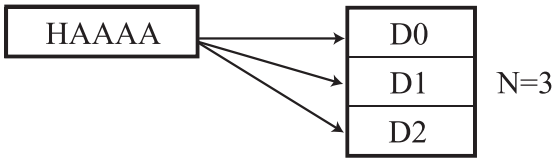
②程式

位址	指　　令	
0	LD	M8002
1	MOV	H5555,K4Y0
6	LD	X0
7	FMOV	HAAAA,D0,K3
14	MOV	D0,K4Y0
19	END	

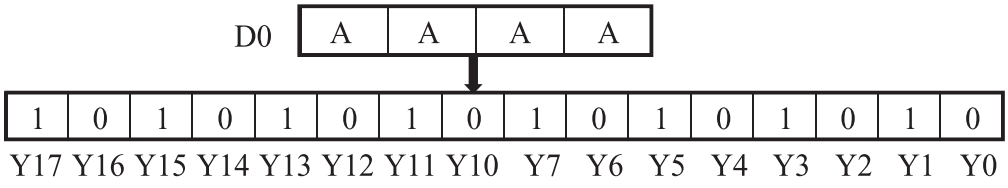
③說明

A.當 RUN 後，Y0～Y17 放入 H5555。即 Y0、Y2、Y4、Y6、Y10、Y12、Y14、Y16 動作，其餘的輸出不動作。

B.當按下 X0 開關時，FMOV 指令將數值 HAAAA 同時傳送（拷貝）到 D0～D2 三個目標元件中。

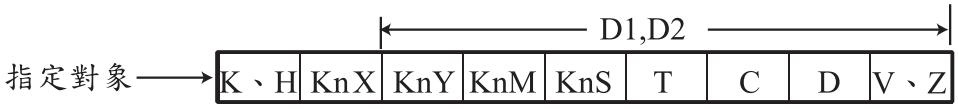


C.再將 D0 傳送到 Y0～Y17 中，結果為 Y1、Y3、Y5、Y7、Y11、Y13、Y15、Y17 動作，其餘的輸出不動作。



6. 數值交換指令：XCH/XCHP（FNC 17）-------DXCH/DXCHP（32 位元）

⑴格式：XCH D1,D2　　D1,D2：目標元件



⑵意義：XCH 是將兩個目標元件的內容互相交換。

⑶舉例：

①階梯圖

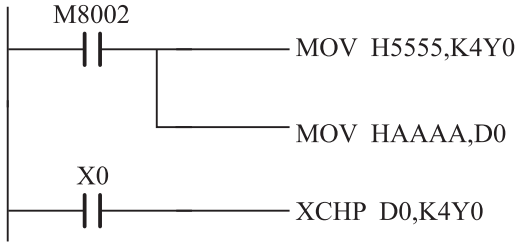


圖 4-7

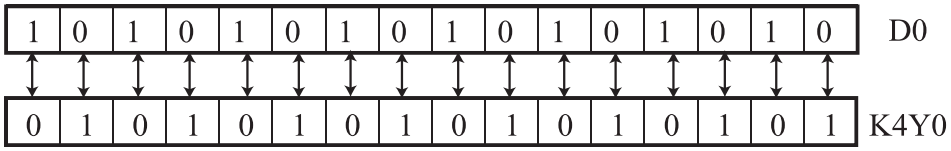
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　H5555,K4Y0
6	MOV　　HAAAA,D0
11	LD　　X0
12	XCHP　　D0,K4Y0
17	END

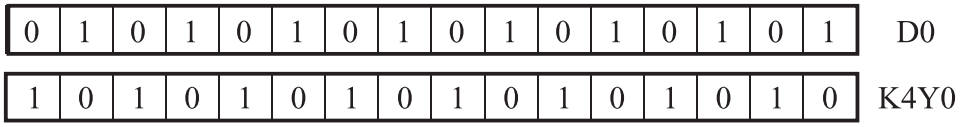
③說明

A.當RUN後，D0被放入數值HAAAA，Y0～Y17放入H5555。即 Y0、Y2、Y4、Y6、Y10、Y12、Y14、Y16 動作，其餘的輸出不動作。

B.當按下 X0 開關時，XCH 指令將數值 D0 與 K4Y0 的內容互相交換。



C.結果 D0=H5555 而 K4Y0=HAAAA，即 Y1、Y3、Y5、Y7、Y11、Y13、Y15、Y17 動作，其餘的輸出不動作。



參 實習

一、題目說明

下面的題目，使用 MOV 來做(M1、M2、M3 為三只馬達)

PB1→M1、M2（ON）

PB2→M2、M3（ON）

PB3→M3、M1（ON）

二、實習步驟

1. 設計觀念：仍以狀態設計法來分析，整個電路包括三種狀態，其設計方法如下:

⑴按 PB1 時為第一種狀態 S1，以 MOV HN1,K1Y0 來代表

⑵按 PB2 時為第二種狀態 S2，以 MOV HN2,K1Y0 來代表

⑶按 PB3 時為第三種狀態 S3，以 MOV HN3,K1Y0 來代表

由上述的分析，可以看出，設計方法不再像使用基本指令一樣，用一電力電驛來代表各狀態的動作，而是改成將各狀態的狀態值以MOV指令傳送到輸出部份。其中的N即為各狀態的輸出狀態值，如下表：

狀態	Y3 ×	Y2 M3	Y1 M2	Y0 M1	N 值 16 進位
S1	0	0	1	1	H3
S2	0	1	1	0	H6
S3	0	1	0	1	H5

2. 元件配置

輸入元件	輸出元件	內部元件
PB1→X0 PB2→X1 PB3→X2 OL1→X3 OL2→X4 OL3→X5	MC1→Y0 MC2→Y1 MC3→Y2	無

3. 繪階梯圖

⑴依狀態分析，初繪如圖 4-8：

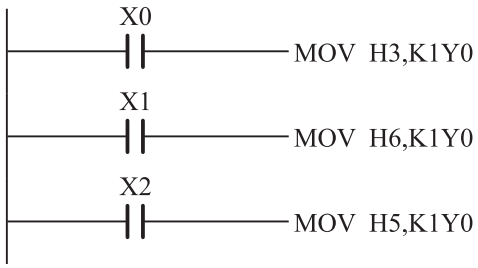


圖 4-8

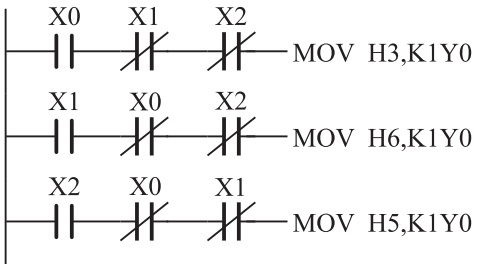


圖 4-9

⑵因為三種狀態中，在一段時間只能有一種存在，所以假若 S1 動作，則 S2、S3 必需切斷；同理 S2 動作，則 S1、S3 必需切斷；S3 動作，則S1、S2 必需切斷。這是一種互鎖電路，最好的方法是利用開關的 B 接點。也就是用 A 接點接通自己而用 B 接點切斷別人，如圖 4-9 所示。

⑶加上 OL 得到圖 4-10。

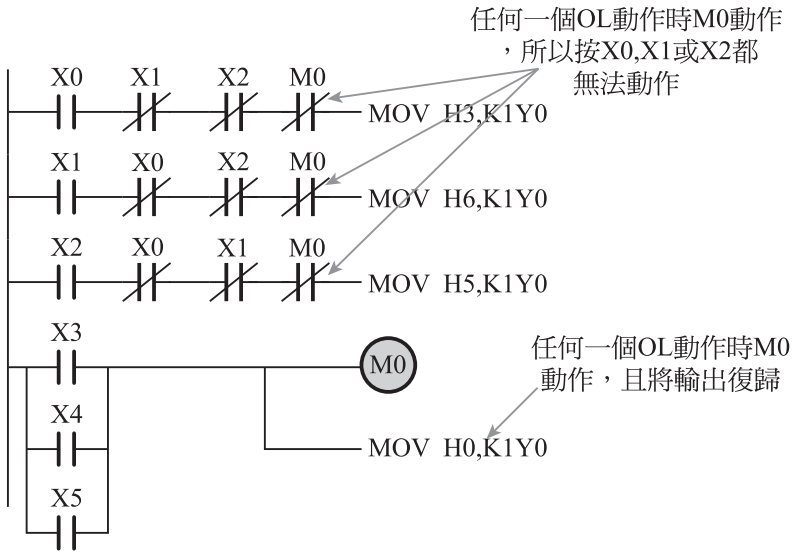


圖 4-10

4. 撰寫程式並鍵入 PLC 中

位址	指　令		位址	指　令	
0	LD	X0	19	ANI	X0
1	ANI	X1	20	ANI	X1
2	ANI	X2	21	ANI	M0
3	ANI	M0	22	MOV	H5,K1Y0
4	MOV	H3,K1Y0	27	LD	X3
9	LD	X1	28	OR	X4
10	ANI	X0	29	OR	X5
11	ANI	X2	30	OUT	M0
12	ANI	M0	31	MOV	H0,K1Y0
13	MOV	H6,K1Y0	36	END	
18	LD	X2			

5. 接線

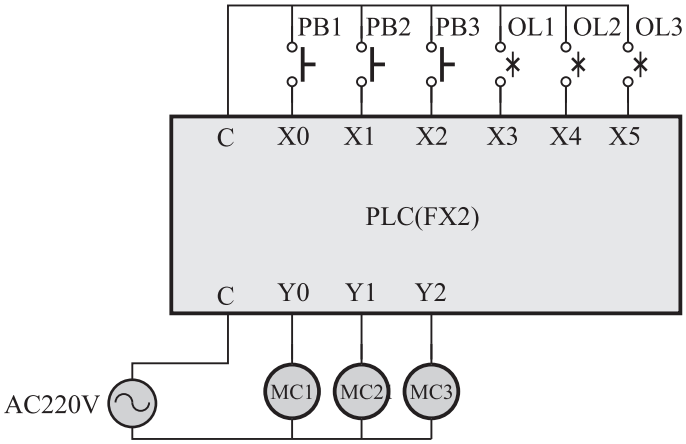


圖 4-11

6. 執行

單元九　應用指令之算數、比較指令

壹 學習目標

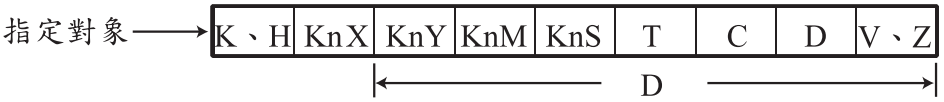
- 利用所附圖表，你能了解各種算數指令的意義及使用方法。
- 利用所附圖表，你能了解各種比較指令的意義及使用方法。
- 你能應用算數及比較指令設計一程式。

貳 相關知識

一、算數指令

1. **加 1 指令**：INC/INCP（FNC 24）-------DINC/DINCP（32 位元）

⑴格式： INC D　D：某一目標元件



⑵意義：INC 是將某一目標元件的值加 1 之後，再存回目標元件中。為了不使加 1 持續進行，因此通常使用微分符號 P。

⑶舉例：

①階梯圖

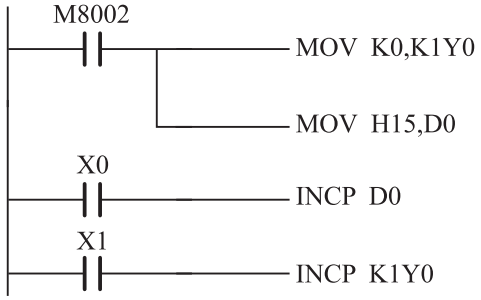


圖 4-12

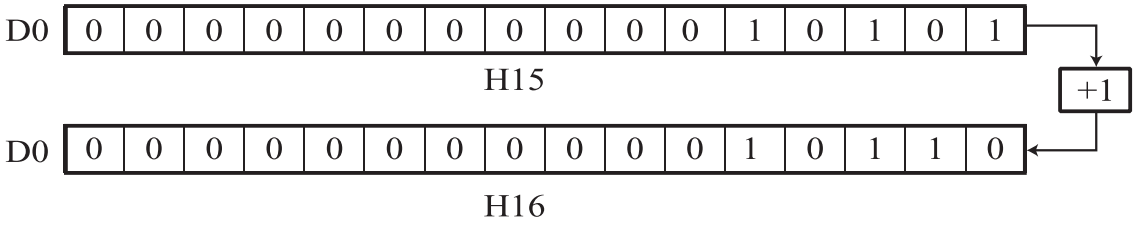
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　K0,K1Y0
6	MOV　　H15,D0
11	LD　　X0
12	INCP　　D0
15	LD　　X1
16	INCP　　K1Y0
19	END

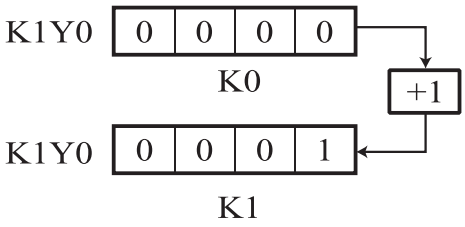
③說明：

A.一 RUN 之後，Y0～Y3 皆被復歸，而 D0 被放入數值 H15。

B.當按下 X0 開關時，D0 的值被加 1 後，再放回 D0 中，故 D0=H16，若放掉 X0 開關後，再按一次 X0 開關，則 D0=H17。



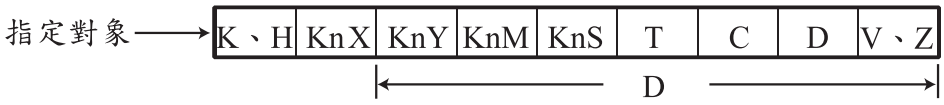
C.當按下 X1 開關時，Y0～Y3 的值被加 1 後，再放回 Y0～Y3 中，故 K1Y0=K1，即只有 Y0 動作，其餘不動作。若放掉 X1 開關後，再按一次 X1 開關，則 K1Y0=K2。



D.16 位元的最大值為 32767(H7FFF)，若加 1 後，結果變成-32768，但是進位旗標及負旗標並不動作。

2.減 1 指令：DEC/DECP（FNC 25）-------DDEC/DDECP(32 位元)

⑴格式: DEC D　D：某一目標元件



⑵意義：DEC 是將某一目標元件的值減 1 之後，再存回目標元件中。為了不使減 1 持續進行，因此通常使用微分符號 P。

⑶舉例：

①階梯圖

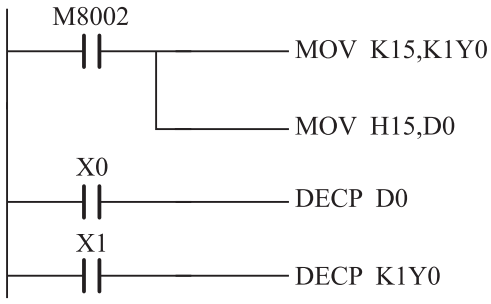


圖 4-13

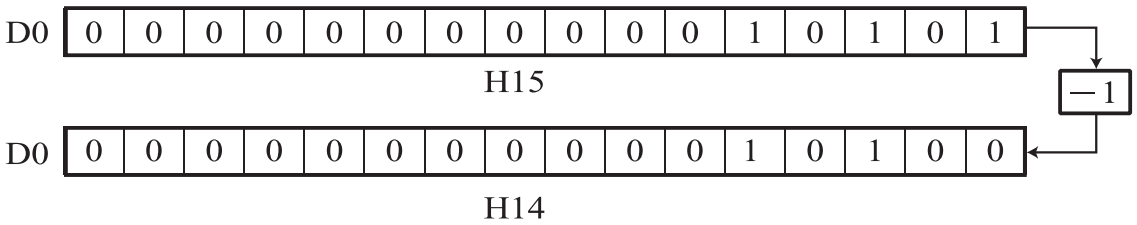
②程式

位址	指　　令	
0	LD	M8002
1	MOV	K15,K1Y0
6	MOV	H15,D0
11	LD	X0
12	DECP	D0
15	LD	X1
16	DECP	K1Y0
19	END	

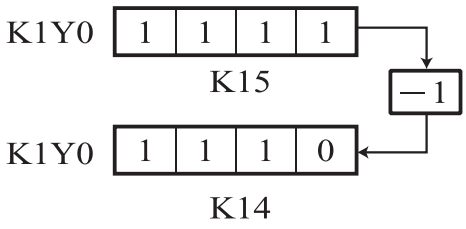
③說明

A.一 RUN 之後，K1Y0 被放入數值 K15(Y0～Y3 皆動作)，而 D0 被放入數值 H15。

B.當按下 X0 開關時，D0 的值被減 1 後，再放回 D0 中，故 D0=H14，若放掉 X0 開關後，再按一次 X0 開關，則 D0=H13。



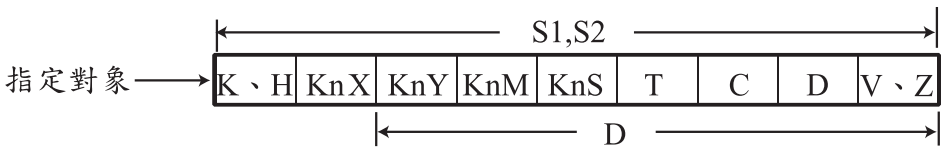
C.按下 X1 開關時，Y0～Y3 的值被減 1 後，再放回 Y0～Y3 中，故 K1Y0=K14，即只有 Y0 復歸，其餘皆動作。若放掉 X1 開關後，再按一次 X1 開關，則 K1Y0=K13。



D.若原值為-32768，經減 1 後，結果變成 32767，但進位旗標 CF 並不動作。

3. 加法指令：ADD/ADDP（FNC 20）-------DADD/DADDP(32 位元)

⑴格式：ADD S1,S2,D S1：被加數
S2：加數
D：目標元件



⑵意義：ADD 是將指定之被加數與加數相加，並將結果存入目標元件中。

⑶舉例：

①階梯圖

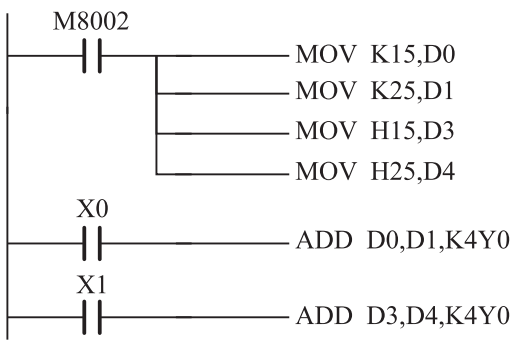


圖 4-14

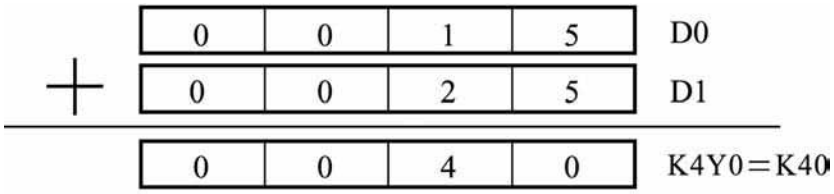
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　K15,D0
6	MOV　　K25,D1
11	MOV　　H15,D3
16	MOV　　H25,D4
21	LD　　X0
22	ADD　　D0,D1,K4Y0
29	LD　　X1
20	ADD　　D3,D4,K4Y0
37	END

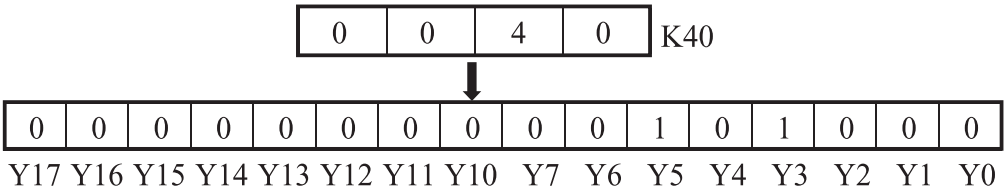
③說明：

A.RUN之後，D0 被放入數值K15；D1 被放入數值K25；D3 被放入數值 H15；D4 被放入數值 H25。

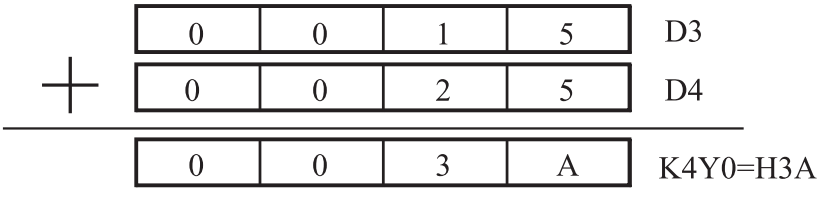
B.當按下 X0 開關時，K4Y0 = (D0) + (D1) = K40。



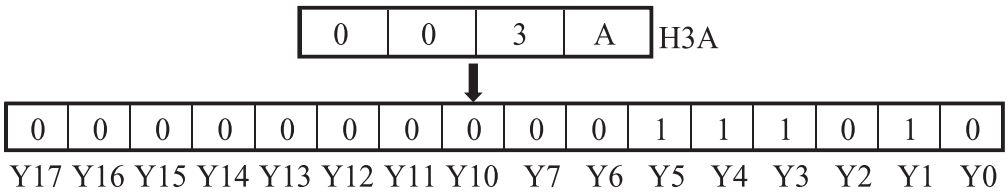
結果如下：Y0～Y17 當中 Y3、Y5 動作，其餘的輸出復歸。



C.當按下 X1 開關時，K4Y0=(D3)+(D4)=H3A。



結果如下：Y0～Y17 當中 Y1、Y3、Y4、Y5 動作，其餘的輸出復歸。



D.相加後的值超過 32767 時，進位旗標 CF（M8022）則為 1。

E.相加後的值等於零時，零位旗標 ZF（M8020）則為 1。

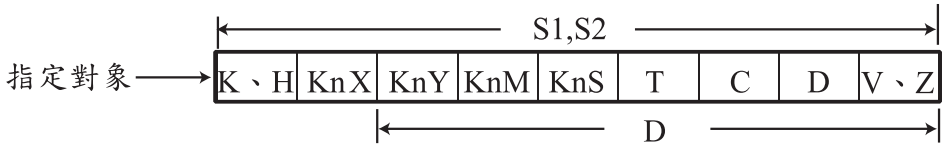
F.相加後的值為負時，負數旗標 NF（M8021）則為 1。

4. 減法指令：SUB/SUBP（FNC 21）-------DSUB/DSUBP(32 位元)

⑴格式: SUB S1,S2,D S1：被減數

S2：減數

D：目標元件



⑵意義：SUB 是將指定之被減數與減數相減，並將結果存入目標元件中。

⑶舉例：

①階梯圖

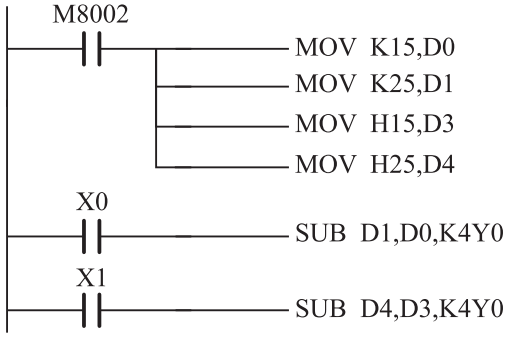


圖 4-15

②程式

位址	指　　令
0	LD　　M8002
1	MOV　　K15,D0
6	MOV　　K25,D1
11	MOV　　H15,D3
16	MOV　　H25,D4
21	LD　　X0
22	SUB　　D1,D0,K4Y0
29	LD　　X1
20	SUB　　D4,D3,K4Y0
37	END

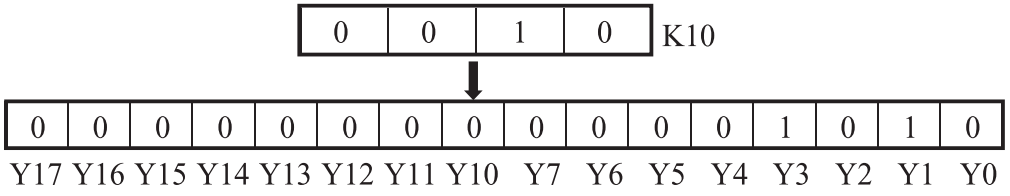
③說明

A.RUN之後，D0被放入數值K15；D1被放入數值K25；D3被放入數值H15；D4被放入數值H25。

B.當按下X0開關時，K4Y0=(D1)-(D0)=K10。

	0	0	2	5	D1
−	0	0	1	5	D0
	0	0	1	0	K4Y0=K10

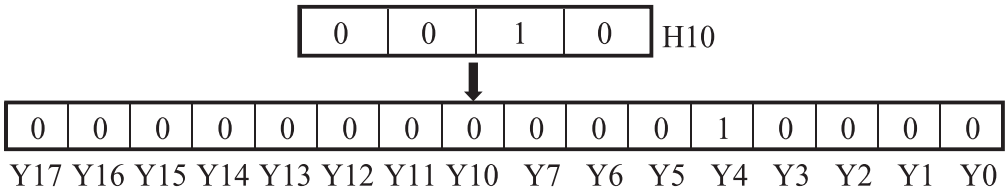
結果如下：Y0～Y17當中Y1、Y3動作，其餘不動作。



C.當按下X1開關時，K4Y0=(D4)-(D3)=H10。

	0	0	2	5	D4
−	0	0	1	5	D3
	0	0	1	0	K4Y0=H10

結果如下：Y0～Y17 當中只有 Y4 動作，其餘不動作。



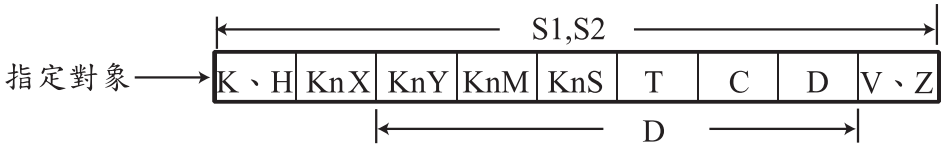
D.相減後的值等於零時，零位旗標 ZF（M8020）則為 1。

5. 二進制乘法指令：MUL/MULP（FNC 22）-------DMUL/DMULP(32 位元)

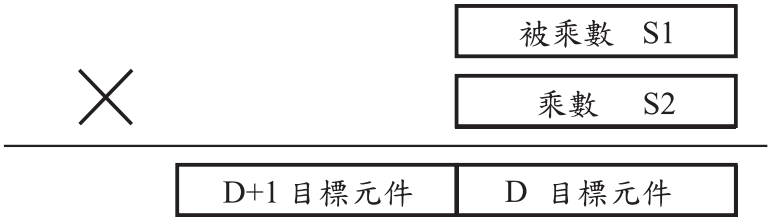
⑴格式: MUL S1,S2,D S1：被乘數

S2：乘數

D：目標元件



⑵意義：MUL 是將兩個 16 位元的二進制數值(被乘數與乘數)相乘，並將結果存入目標元件中。



⑶舉例：

①階梯圖

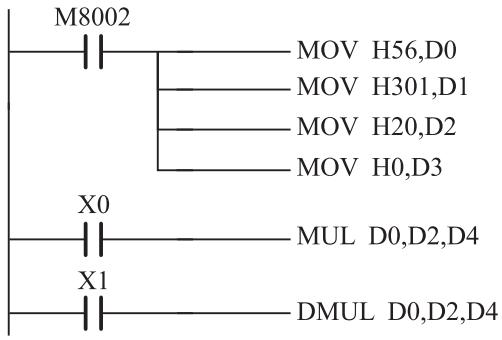


圖 4-16

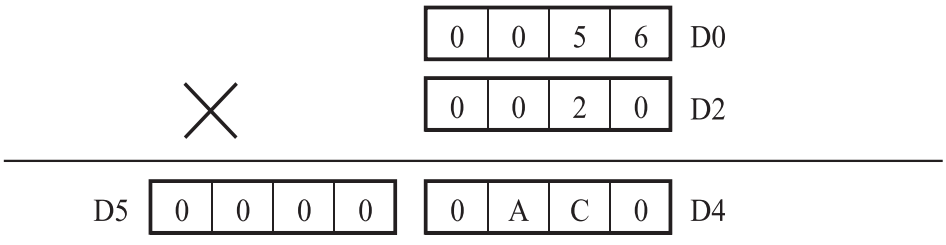
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　H56,D0
6	MOV　　H301,D1
11	MOV　　H20,D2
16	MOV　　H0,D3
21	LD　　X0
22	MUL　　D0,D2,D4
29	LD　　X1
30	DMUL　　D0,D2,D4
43	END

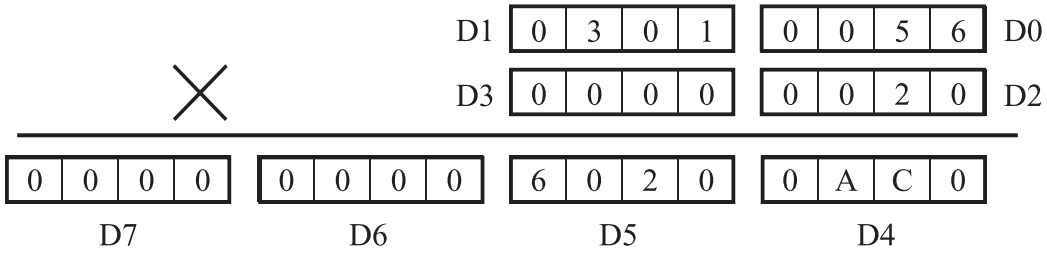
③說明

A.RUN 之後，D0、D1、D2、D3 被放入數值 H56、H301、H20、H0。

B.當按下 X0 開關時，(D5，D4)=(D0) × (D2)=HAC0。下面的算式是以 nibble 為單位來計算。結果 D4=HAC0、D5=H0（使用監視模式來看結果）。



C.當按下 X1 開關時，則執行 32 位元的乘法運算，即(D7，D6，D5，D4)=(D1，D0) × (D3，D2)=HAC0。下面的算式是以 nibble 為單位來計算。結果 D4=HAC0、D5=H602、D6=H0、D7=H0 (使用監視模式來看結果)。

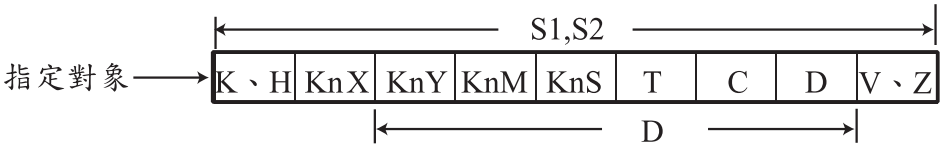


5.二進制除法指令：DIV/DIVP（FNC 23）-------DDIV/DDIVP（32 位元）

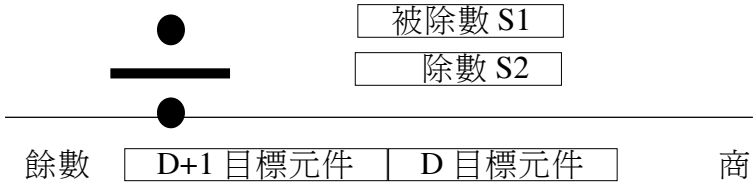
⑴格式: DIV　S1,S2,D　S1：被除數

S2：除數

D：目標元件



⑵意義：DIV 是將兩個 16 位元的二進制數值(被除數與除數)相除，並將結果存入目標元件中。結果須要兩個位置，一個用來存商，另一個用來存餘數。



⑶舉例：

①階梯圖

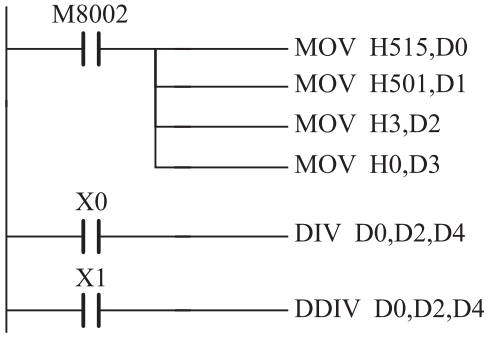


圖 4-17

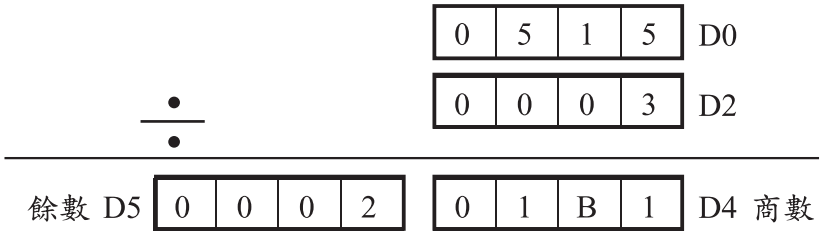
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　H515,D0
6	MOV　　H501,D1
11	MOV　　H3,D2
16	MOV　　H0,D3
21	LD　　X0
22	DIV　　D0,D2,D4
29	LD　　X1
30	DDIV　　D0,D2,D4
43	END

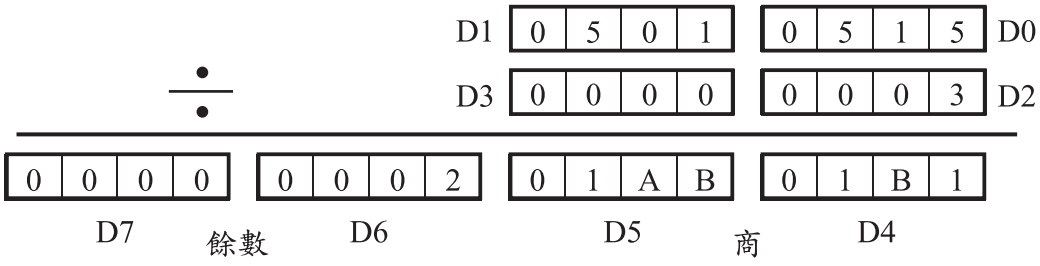
③說明：

A.RUN 之後，D0、D1、D2、D3 被放入數值 H515、H501、H3、H0。

B.當按下 X0 開關時，(D4)...(D5) = (D0) ÷ (D2)。下面的算式是以 nibble 為單位來計算。結果商 D4 = H1B1，餘數 D5 = H2（使用監視模式來看結果）。



C.當按下 X1 開關時，則執行 32 位元的除法運算，即(D5，D4)...(D7，D6)=(D1，D0) ÷ (D3，D2)。下面的算式是以 nibble 為單位來計算。結果商 D4=H1B1、D5=H1AB，餘數 D6=H2、D7=H0（使用監視模式來看結果）。

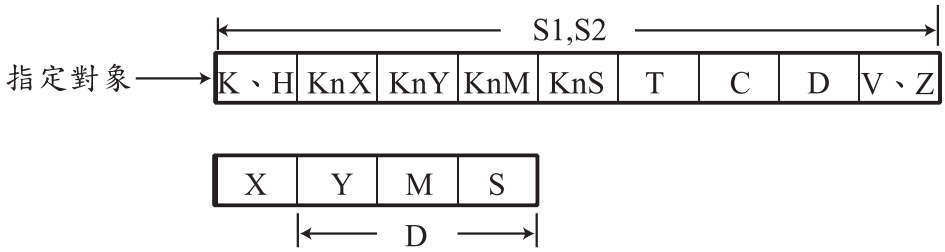


二、比較指令

1. CMP/CMPP（FNC 10）-------DCMP/DCMPP（32 位元）

⑴格式：CMP S1,S2,D　S1，S2：比較用的兩來源元件或數值

D：目標元件



⑵意義：將 S1 及 S2 互相比較，並指定目標元件來反應大於、等於、小於動作。

⑶舉例：

①階梯圖

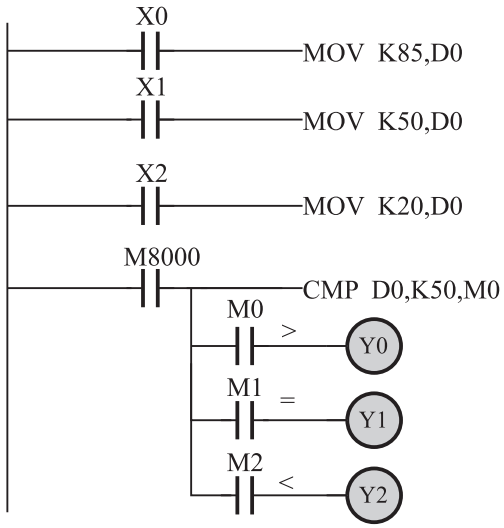


圖 4-18

②程式

位址	指　令	位址	指　令
0	LD X0	27	AND M0
1	MOV K85,D0	28	OUT Y0
6	LD X1	29	MRD
7	MOV K50,D0	30	AND M1
12	LD X2	31	OUT Y1
13	MOV K20,D0	32	MPP
18	LD M8000	33	AND M2
19	MPS	34	OUT Y2
20	CMP D0,K50,M0	35	END

③說明

A.當一RUN之後，因D0之初始值為0，所以Y2動作(<K50)。

B.當按下 X0 開關時，MOV 指令將十進制數值 85 輸出到 D0（D0=K85）。經 CMP 比較後，因為 K85>K50，所以 M0 動作，輸出 Y0 動作。

C.當按下 X1 開關時，MOV 指令將十進制數值 50 輸出到 D0（D0=K50）。經 CMP 比較後，因為 K50=K50，所以 M1 動作，輸出 Y1 動作。

D.當按下 X2 開關時，MOV 指令將十進制數值 20 輸出到 D0（D0=K20）。經 CMP 比較後，因為 K20<K50，所以 M2 動作，輸出 Y2 動作。

④時序圖

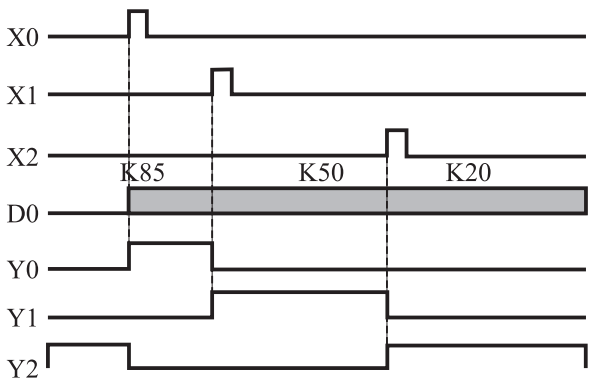
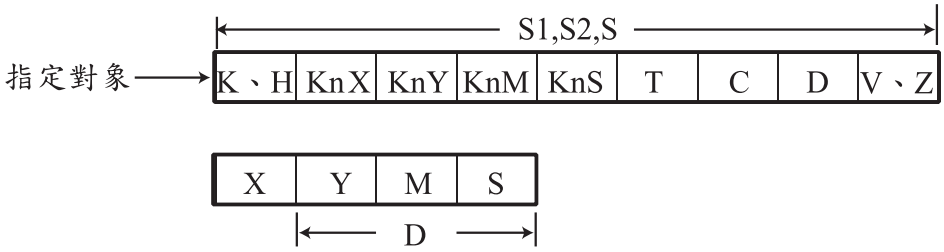


圖 4-19

2. 區塊（範圍）比較：ZCP/ZCPP（FNC 11）-------DZCP/DZCPP（32位元）

⑴格式：ZCP S1,S2,S,D S1：比較區塊啟始端元件或數值
S2：比較區塊終端元件或數值
S：欲比較元件或數值
D：目標元件



⑵意義：ZCP 是將一元件或數值(S)與一區塊互相比較(S1～S2)，當 S 小於 S1 時，D 動作；當 S 介於 S1 與 S2 之間(含 S1、S2) 時， D+1 動作；當 S 大於 S2 時，D+2 動作。

⑶舉例：

①階梯圖

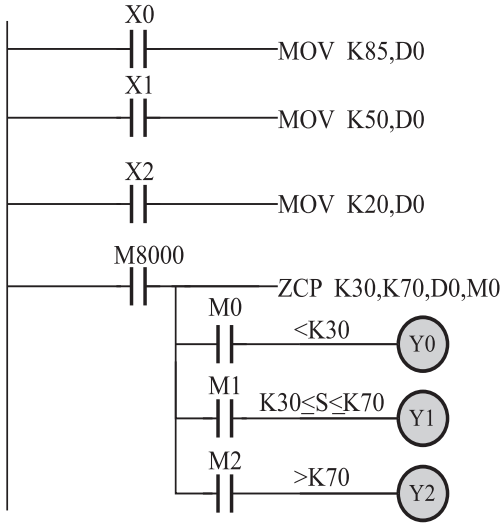


圖 4-20

②程式

位址	指　　令		位址	指　　令	
0	LD	X0	29	AND	M0
1	MOV	K85,D0	30	OUT	Y0
6	LD	X1	31	MRD	
7	MOV	K50,D0	32	AND	M1
12	LD	X2	33	OUT	Y1
13	MOV	K20,D0	34	MPP	
18	LD	M8000	35	AND	M2
19	MPS		36	OUT	Y2
20	ZCP	K30,K70,D0,M0	37	END	

③說明：

A.當一RUN之後，因D0之初始值為0，所以Y0動作(<K30)。

B.當按下 X0 開關時，MOV 指令將十進制數值 85 輸出到 D0（D0=K85）。經 ZCP 比較後，因為 K85>K70，所以 M2 動作，輸出 Y2 動作。

C.當按下 X1 開關時，MOV 指令將十進制數值 50 輸出到 D0（D0=K50）。經 ZCP 比較後，因為 K30<K50<K70，所以 M1 動作，輸出 Y1 動作。

D.當按下 X2 開關時，MOV 指令將十進制數值 20 輸出到 D0（D0=K20）。經 ZCP 比較後，因為 K20<K30，所以 M0 動作，輸出 Y0 動作。

④時序圖

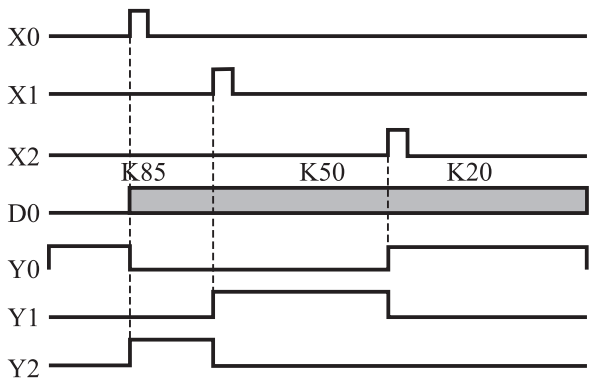


圖 4-21

參 實習

一、題目說明：算術指令的應用

相信各位讀者都曾經去過木柵動物園，它裏面有一間夜行動物館，展出的是各種夜行性動物。當然本題並不是在介紹這些可愛的小動物（非本人專長），而是如何設計一控制電路，在不用人看管下，能隨時控制館內的人數不會超過 100 人（避免人數過多，干擾動物的生活）。

二、實習步驟

1. 狀態分析

⑴前門使用一電磁鎖來控制門的開與關，另外前後門各加裝一個檢測開關（LS_F及LS_B）用來記錄進出人數，當人數少於 100 人時，電磁鎖不動作，前門可打開。若人數已達 100 人，則電磁鎖動作鎖住，前門打不開。

⑵加裝 PL0、PL1 兩指示燈，PL0 為額滿指示燈，PL1 為可進入指示燈。

⑶加啟動(PB_{ON})及停止(PB_{OFF})按鈕。

2. 元件編號

輸入元件	輸出元件	內部元件
PB_{OFF} →X0	電磁鎖→Y0	D200
PB_{ON} →X1	PL0→Y1	M10
LS_F →X2	PL1→Y2	M0，M1，M2
LS_B →X3		

3. 繪出階梯圖

⑴使用具有停電保持功能的資料暫存器(D200)來存放人數累計值，這是為了停電之後再送電時能正常工作（D200 的說明，留待專門單元說明，此時暫且不談）。當有人進入時（開關 LS_F 由 OFF 變 ON 時），則使用 INC 指令將人數加 1，當有人出來時（開關 LS_B 由 OFF 變 ON 時），則使用 DEC 指令將人數減 1。如圖 4-22 所示。

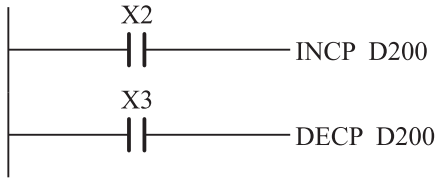


圖 4-22

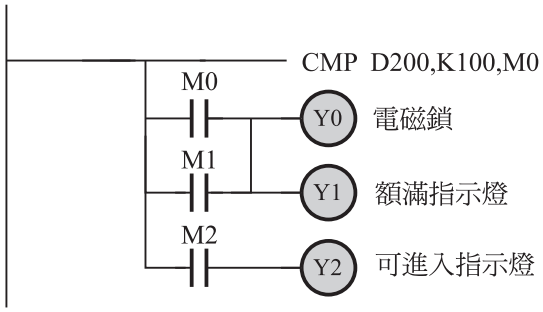


圖 4-23

⑵使用 CMP 指令來判斷人數是否超過 100 人，如圖 4-23 所示。
⑶加上啟動及停止按鈕後全圖如圖 4-24 所示。

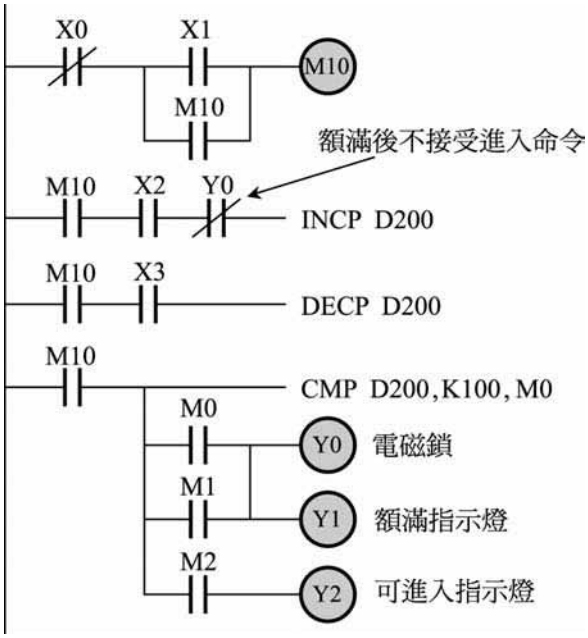


圖 4-24

⑷本例也可使用 ADD 及 SUB 來做，如圖 4-25 所示。

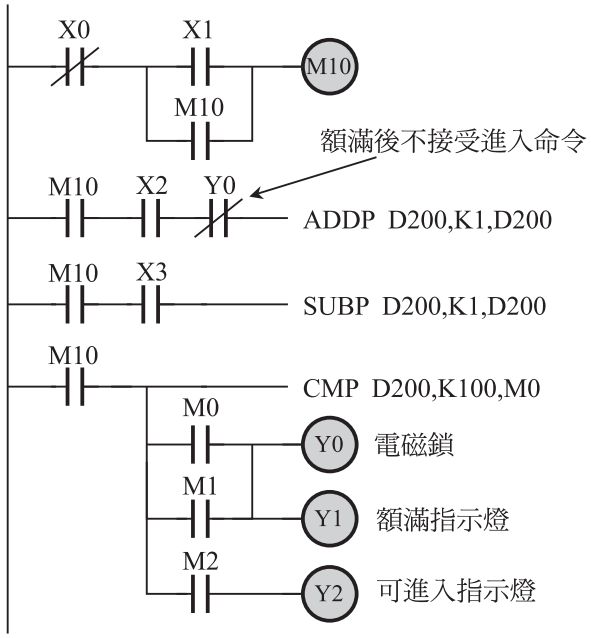


圖 4-25

4. 撰寫程式並鍵入 PLC 中
圖 4-24 之程式

位址	指　令	位址	指　令
0	LD　X1	16	MPS
1	OR　M10	17	CMP　D200,K100,M0
2	ANI　X0	24	LD　M0
3	OUT　M10	25	OR　M1
4	LD　M10	26	ANB
5	AND　X2	27	OUT　Y0
6	ANI　Y0	28	OUT　Y1
7	INCP　D200	29	MPP
10	LD　M10	30	AND　M2
11	AND　X3	31	OUT　Y2
12	DECP　D200	32	END
15	LD　M10		

圖 4-25 之程式

位址	指　　令	位址	指　　令
0	LD　X1	24	MPS
1	OR　M10	25	CMP　D200,K100,M0
2	ANI　X0	32	LD　M0
3	OUT　M10	33	OR　M1
4	LD　M10	34	ANB
5	AND　X2	35	OUT　Y0
6	ANI　Y0	36	OUT　Y1
7	ADDP　D200,K1,D200	37	MPP
14	LD　M10	38	AND　M2
15	AND　X3	39	OUT　Y2
16	SUBP　D200,K1,D200	40	END
23	LD　M10		

5. 接線

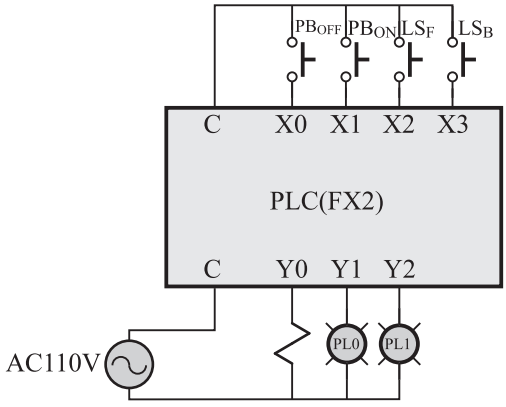


圖 4-26

6. 執行

單元十　應用指令之位移、旋轉指令

壹 學習目標

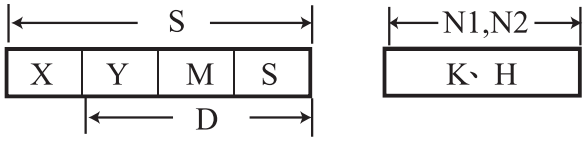
- 利用所附圖表，你能了解各種位移指令的意義及使用方法。
- 利用所附圖表，你能了解各種旋轉指令的意義及使用方法。
- 你能應用位移及旋轉指令設計一程式。

貳 相關知識

一、位移指令

1. 位元左位移指令：SFTL/SFTLP（FNC 35）

⑴格式：SFTL S,D,N1,N2　S：補位元件的啟始位元
D：欲移位目標元件的啟始位元
N1：欲移位目標元件的總數量
N2：每次移幾個位元



⑵意義：SFTL 是將欲移位目標元件內的值，向左位移 N2 個位元，左移後所空下的位元由補位元件填入，高位元部份則移出並捨棄不用。

⑶舉例：

①階梯圖

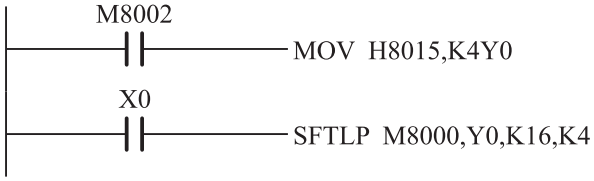


圖 4-27

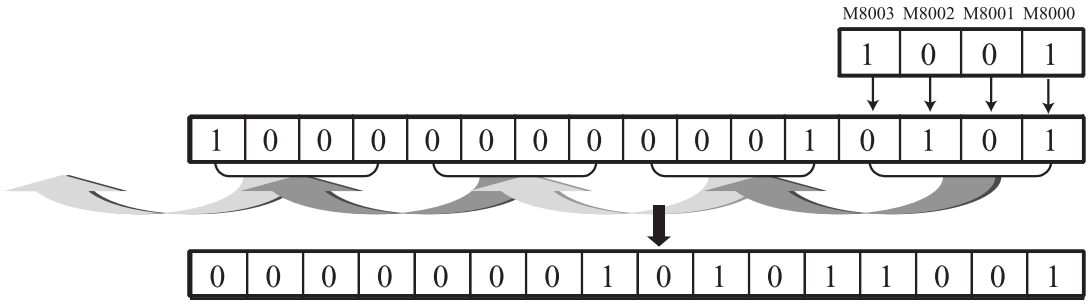
②程式

位址	指　　令	
0	LD	M8002
1	MOV	H8015,K4Y0
6	LD	X0
7	SFTLP	M8000,Y0,K16,K4
16	END	

③說明

A.一 RUN 之後，Y0～Y17 被放入數值 H8015。即 Y0、Y2、Y4、Y17 動作，其餘的輸出不動作。

B.當按下 X0 開關時，Y0～Y17 向左移位 4(N2)個位元，並由 M8000～M8003 補位。



C.結果 Y0、Y3、Y4、Y6、Y10 動作，其餘的輸出不動作。

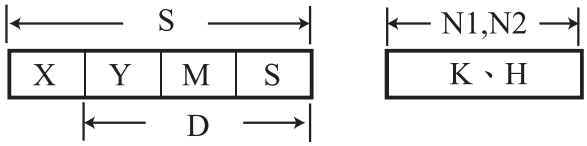
2. **位元右位移指令：SFTR/SFTRP（FNC 34）**

⑴格式：SFTR S,D,N1,N2　　S：補位元件的啟始位元

D：欲移位目標元件的啟始位元

N1：欲移位目標元件的總數量

N2：每次移幾個位元



⑵意義：SFTR 是將欲移位目標元件內的值，向右位移 N2 個位元，右移後所空下的位元由補位元件填入，低位元部份則移出並捨棄不用。

⑶舉例：

①階梯圖

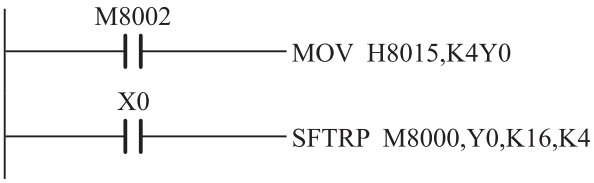


圖 4-28

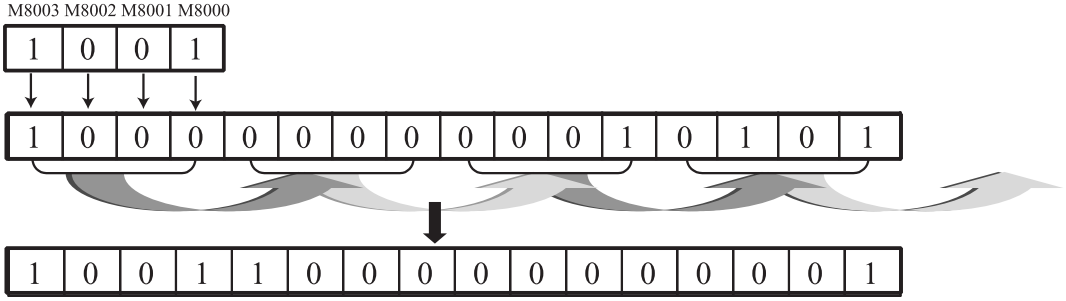
②程式

位址	指　　令	
0	LD	M8002
1	MOV	H8015,K4Y0
6	LD	X0
7	SFTRP	M8000,Y0,K16,K4
16	END	

③說明

A.一 RUN 之後，Y0～Y17 被放入數值 H8015。即 Y0、Y2、Y4、Y17 動作，其餘的輸出不動作。

B.當按下 X0 開關時，Y0～Y17 向右移位 4(N2)個位元，並由 M8000～M8003 補位。



C.結果 Y0、Y13、Y14、Y17 動作，其餘的輸出不動作。

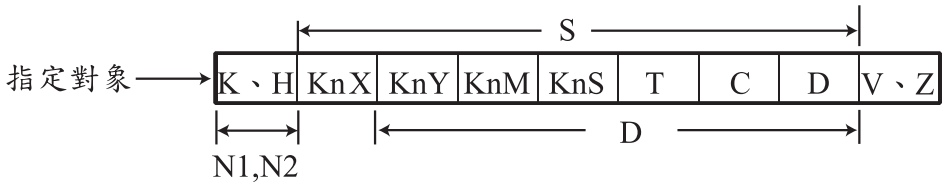
3. **運算元左位移指令**：WSFL/WSFLP（FNC 37）

⑴格式：WSFL S,D,N1,N2　　S：補位元件的啟始編號

D：欲移位目標元件的啟始編號

N1：欲移位目標元件的總數量

N2：每次移幾個運算元



⑵意義：WSFL 是可將多個相鄰的目標元件內的值，向左位移 N2 個元件，左移後所空下的元件值則由補位元件填入，高元件部份則移出並捨棄不用。

⑶舉例：

①階梯圖

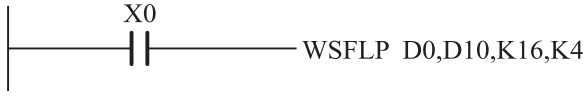


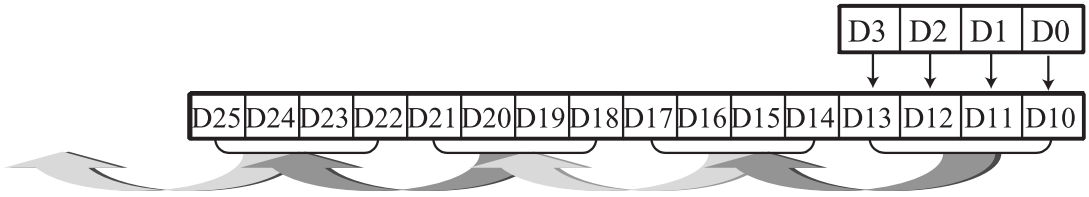
圖 4-29

②程式

位址	指　　令
0	LD　　X0
1	WSFLP　D0,D10,K16,K4
10	END

③說明

A.一RUN之後，當按下X0 開關時，D10～D25 向左移位 4(N2) 個位置，並由 D0～D3 補位。



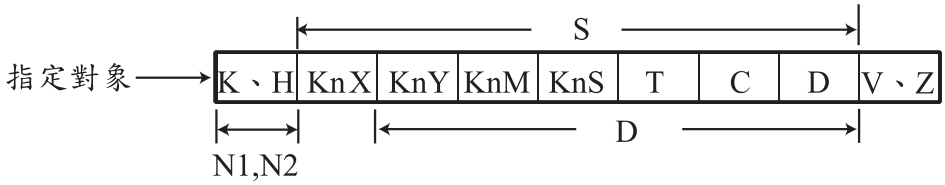
4. **運算元右位移指令：**WSFR/WSFRP（FNC 36）

⑴格式：WSFR S,D,N1,N2　　S：補位元件的啟始編號

D：欲移位目標元件的啟始編號

N1：欲移位目標元件的總數量

N2：每次移幾個運算元



⑵意義：WSFR 是可將多個相鄰的目標元件內的值，向右位移 N2 個元件，右移後所空下的元件值則由補位元件填入，低元件部份則移出並捨棄不用。

⑶舉例：

①階梯圖

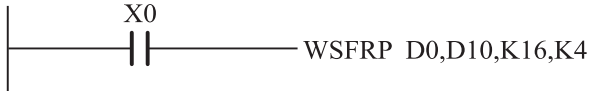


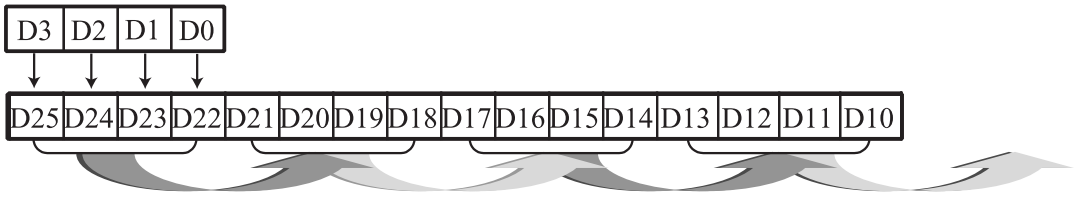
圖 4-30

②程式

位址	指　　令
0	LD　　X0
1	WSFRP　D0,D10,K16,K4
10	END

③說明

A.一RUN之後，當按下X0 開關時，D10～D25 向右移位 4(N2) 個位置，並由 D0～D3 補位。

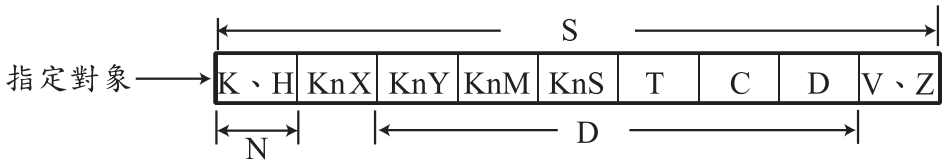


5.**位移寫入指令**：SFWR/SFWRP（FNC 38）

⑴格式：SFWR　S,D,N　　S：元件或數值

D：欲寫入目標元件的啟始編號

N：目標元件的總數量



⑵意義：SFWR 可將來源元件或數值順序寫入多個相鄰的目標元件中。是一種以先入先出(FIFO)的方式將資料順序寫入的指令。

⑶舉例：

①階梯圖

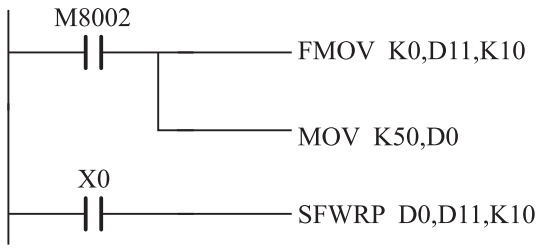


圖 4-31

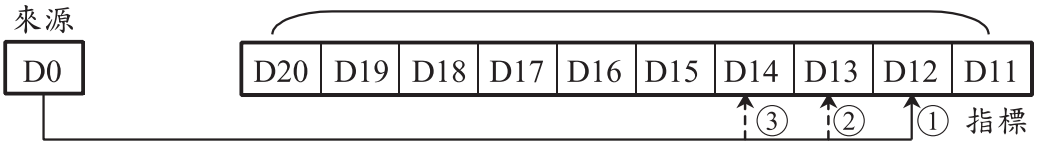
②程式

位址	指　　令	
0	LD	M8002
1	FMOV	K0,D11,K10
8	MOV	K50,D0
13	LD	X0
14	SFWRP	D0,D11,K10
21	END	

③說明

A.一 RUN 之後，D11～D20 被放入數值 K0(D11 指標歸零)。且來源元件 D0 被放入數值 K50。

B.當按下 X0 開關時，D0 的內容(K50)被送至 D12 當中，且指標 D11 自動加 1(=1)。若 X0 開關再次被按下，則 D0 的內容(K50)被送至D13 當中，且指標D11 又自動加 1(=2)，如此的 D0 的內容被順序的寫入 D12～D20 中。



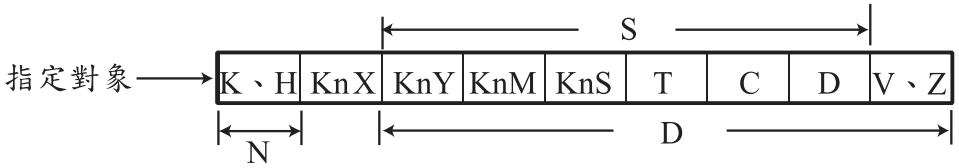
C.N 值應界於 2～512 間。

D.當 D11 的內容超過 N-1 時，本指令不處理，另外 M8022 動作。

E.若隨時變更 D0 的內容，則可將不同的內容寫入目標元件中。

6. **位移讀出指令**：SFRD/SFRDP（FNC 39）

⑴格式：SFRD S,D,N　　S：欲讀出來源元件的啟始編號
D：目標元件
N：來源元件的總數量



⑵意義：SFRD 可將多個相鄰的來源元件順序讀出到目標元件中。是一種以先入先出(FIFO)的方式將資料順序讀出的指令。

⑶舉例：

①階梯圖

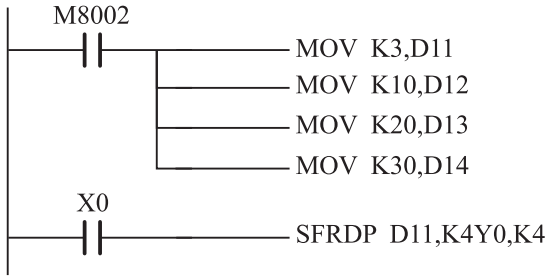


圖 4-32

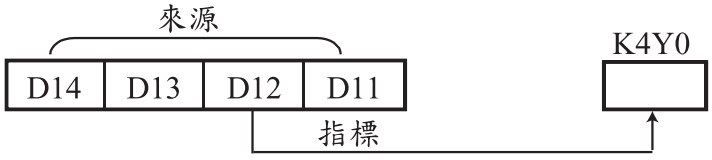
②程式

位址	指　　令	
0	LD	M8002
1	MOV	K3,D11
6	MOV	K10,D12
11	MOV	K20,D13
16	MOV	K30,D14
21	LD	X0
22	SFRDP	D11,K4Y0,K4
29	END	

③說明

A.一 RUN 之後，D11 被放入數值 K3(來源元件範圍 D12～D14)。D12、D13、D14 分別被放入數值 K10、K20、K30。

B.當按下 X0 開關時，D12 的內容(K10)被送至 Y0～Y17 中(Y1、Y3 動作)，且 D14～D12 全部向右移一元件，即 D14=K30、D13=K30、D12=K20，另外指標D11 自動減 1(=2)。若 X0 開關再次被按下，仍然是 D12 的內容(K20)被送至 Y0～Y17 中(Y2、Y4 動作)，且D14～D12 又全部向右移一元件，即 D14=K30、D13=K30、D12=K30，指標 D11 再度減 1(=1)，如此的 D12～D14 的內容被順序的讀出至 Y0～Y17 中。



C.N 值應界於 2～512 間。

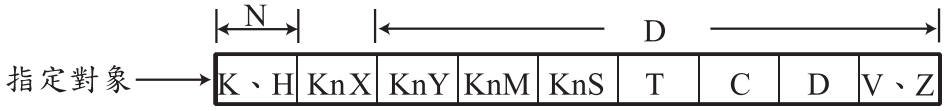
D.當 D11 的內容為 0 時，本指令不處理，另外 M8022 動作。

二、旋轉指令

1. **左旋轉指令**：ROL/ROLP（FNC 31）-------DROL/DROLP(32 位元)

⑴格式：ROL D,N　　D：欲旋轉目標元件

N：每次旋轉幾個位元



⑵意義：ROL 是將欲旋轉目標元件內的值，向左旋轉 N 個位元，而最高位元被移入進位旗標(CF)內。

⑶舉例

①階梯圖

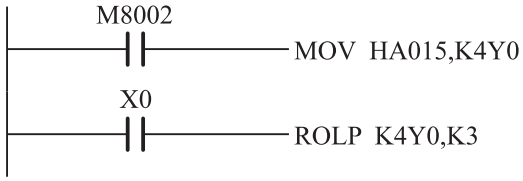


圖 4-33

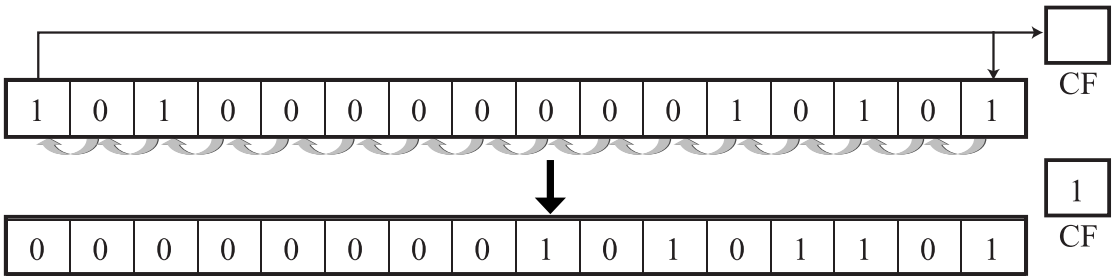
②程式

位址	指　　令	
0	LD	M8002
1	MOV	HA015,K4Y0
6	LD	X0
7	ROLP	K4Y0,K3
12	END	

③說明

A.一 RUN 之後，Y0～Y17 被放入數值 HA015。即 Y0、Y2、Y4、Y15、Y17 動作，其餘的輸出不動作。

B.當按下 X0 開關時，Y0～Y17 被向左旋轉 3(N)個位元。



C.結果如下: Y0、Y2、Y3、Y5、Y7 動作，其餘的輸出不動作。

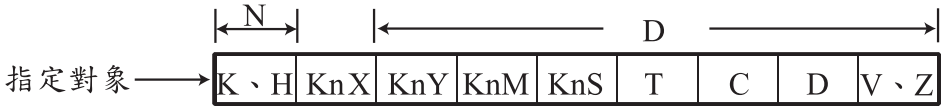
D.進位旗標(CF)電驛 M8022=1。

E. 選擇 M、Y 做為目標元件時，只有 K4(16 位元)及 K8(32 位元)有效。

2. **右旋轉指令**：ROR/RORP（FNC 30）-------DROR/DRORP(32 位元)

⑴格式：ROR D,N　　D：欲旋轉目標元件

N：每次旋轉幾個位元



⑵意義：ROR 是將欲旋轉目標元件內的值，向右旋轉 N 個位元，而最低位元被移入進位旗標(CF)內。

⑶舉例：

①階梯圖

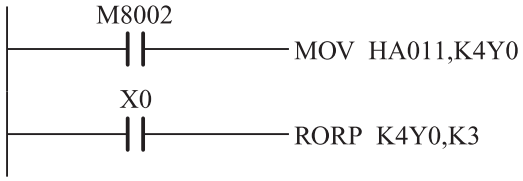


圖 4-34

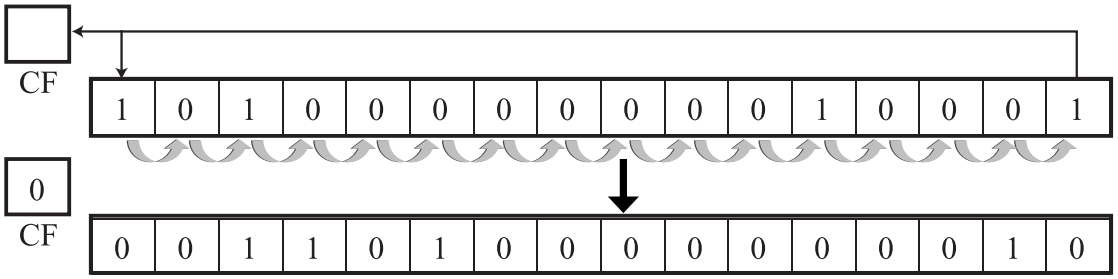
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　HA011,K4Y0
6	LD　　X0
7	RORP　　K4Y0,K3
12	END

③說明

A. 一 RUN 之後，Y0～Y17 被放入數值 HA011。即 Y0、Y4、Y15、Y17 動作，其餘的輸出不動作。

B.當按下 X0 開關時，Y0～Y17 被向右旋轉 3(N)個位元。



C.結果如下: Y1、Y12、Y14、Y15 動作，其餘的輸出不動作。

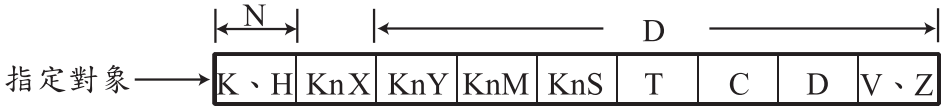
D.進位旗標(CF)電驛 M8022=0。

E. 選擇 M、Y 做為目標元件時，只有 K4(16 位元)及 K8(32 位元)有效。

3. **含CF左旋轉指令**：RCL/RCLP（FNC 33）-------DRCL/DRCLP(32 位元)

⑴格式：RCL D,N　　D：欲旋轉目標元件

N：每次旋轉幾個位元



⑵意義：RCL 是將欲旋轉目標元件內的值，連同進位旗標一起向左旋轉 N 個位元。

⑶舉例：

①階梯圖

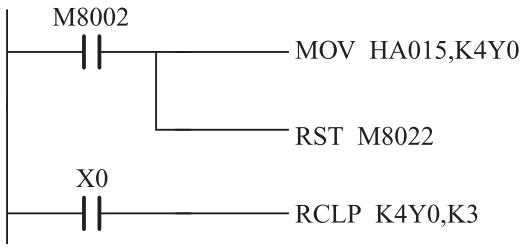


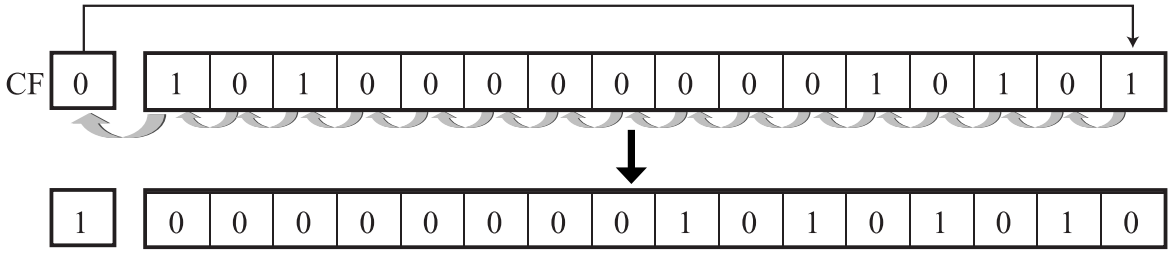
圖 4-35

②程式

位址	指　　令	
0	LD	M8002
1	MOV	HA015,K4Y0
6	RST	M8022
8	LD	X0
9	RCLP	K4Y0,K3
12	END	

③說明：

A.一 RUN 之後，Y0～Y17 被放入數值 HA015。即 Y0、Y2、Y4、Y15、Y17 動作，其餘的輸出不動作，同時進位旗標歸零。



B.當按下X0開關時，Y0～Y17及CF被向左旋轉3(N)個位元。

C.結果如下: Y1、Y3、Y5、Y7動作，其餘的輸出不動作。

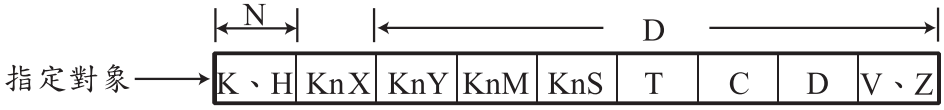
D.進位旗標(CF)電驛M8022=1。

E.選擇 M、Y 做為目標元件時，只有 K4(16 位元)及 K8(32 位元)有效。

*4.*含 CF 右旋轉指令：RCR/RCRP（FNC 32）-------DRCR/DRCRP(32位元)

⑴格式：RCR D,N　　D：欲旋轉目標元件

N：每次旋轉幾個位元



⑵意義：RCR 是將欲旋轉目標元件內的值，連同進位旗標一起向右旋轉N個位元。

⑶舉例：

①階梯圖

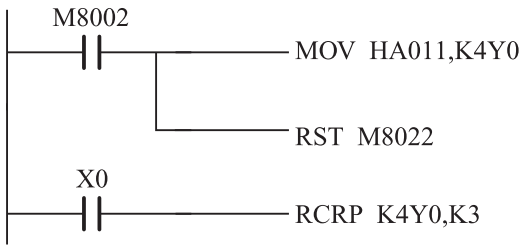


圖 4-36

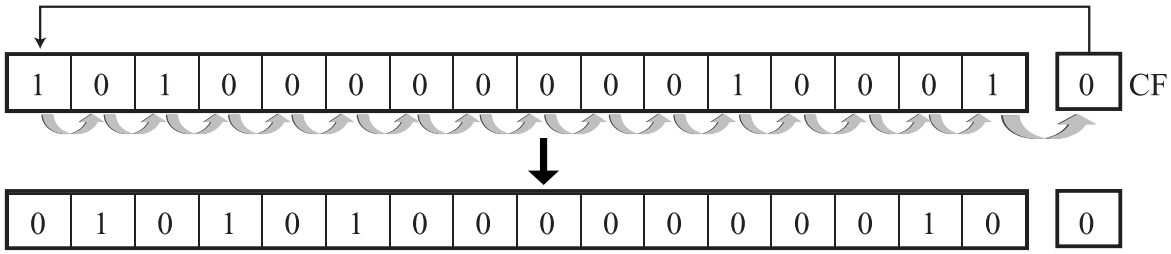
②程式

位址	指　　令
0	LD　　M8002
1	MOV　　HA011,K4Y0
6	RST　　M8022
8	LD　　X0
9	RCRP　　K4Y0,K3
14	END

③說明

A.一 RUN 之後，Y0～Y17 被放入數值 HA011。即 Y0、Y4、Y15、Y17 動作，其餘的輸出不動作，同時進位旗標歸零，

B.當按下X0 開關時，Y0～Y17 及CF被向右旋轉 3(N)個位元。



C.結果如下: Y1、Y12、Y14、Y16 動作，其餘的輸出不動作。

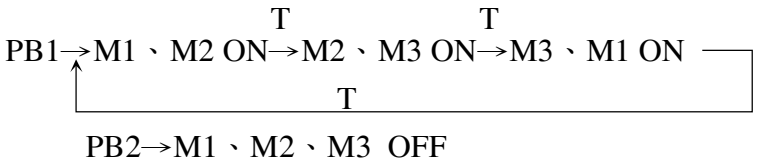
D.進位旗標(CF)電驛 M8022=0。

E.選擇 M、Y 做為目標元件時，只有 K4(16 位元)及 K8(32 位元)有效。

參 實習

工作一

一、題目說明



二、實習步驟

1. 設計觀念:依題意來分析，可以得出下列三個依設定時間而順序執行的狀態。

⑴狀態 S1（M0）→M1 及 M2 動作

⑵狀態 S2（M1）→M2 及 M3 動作

⑶狀態 S3（M2）→M3 及 M1 動作

PB1→S1→S2→S3→

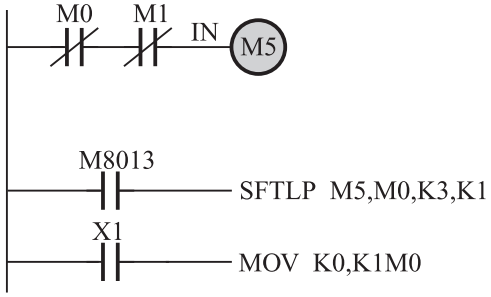
PB2→S1、S2、S3 復歸

2. 元件配置

輸入元件	輸出元件	內部元件
PB1→X0	M1→Y0	S1→M0
PB2→X1	M2→Y1	S2→M1
	M3→Y2	S3→M2
		M4
		M5

3. 繪階梯圖

⑴採用移位指令來產生連續狀態，如下。圖中的 IN 輸入由 M0 及 M1 的 B 接點來做為 SFTL 指令的補位元件(M5)的控制，以產生右下表的狀態循環。



狀態	M2	M1	M0
S1	0	0	1
S2	0	1	0
S3	1	0	0
S1	0	0	1
S2	0	1	0
S3	1	0	0
S1	0	0	1
S2	0	1	0
S3	1	0	0

圖 4-37

① RUN 之後，M0 及 M1 皆為 OFF 狀態，所以補位元件(M5)的狀態為 1，當 CLOCK 來時（M8013 由 0 變 1），補位元件(M5)的狀態，被移入 M0～M2 的最低位元（M0），即產生了狀態 S1。

② S1 狀態發生後，M0 動作，B 接點斷路，所以補位元件(M5)的狀態變為 0，當 CLOCK 來時（M8013 由 0 變 1），補位元件(M5)的狀態，被移入 M0～M2 的最低位元（M0 為 0），而原有 M0 的狀態被移入 M1（為 1），即產生了狀態 S2。

③S2 狀態發生後，M1 動作，B 接點斷路，所以補位元件(M5)的狀態仍為 0，當 CLOCK 來時（M8013 由 0 變 1），補位元件(M5)的狀態，被移入 M0～M2 的最低位元（M0 為 0），而原有 M1 的狀態被移入 M2（為 1），原有 M0 的狀態被移入 M1（為 0），即產生了狀態 S3。

④S3 狀態發生後，M0 及 M1 皆為 OFF 狀態，所以補位元件(M5)的狀態變為 1，當 CLOCK 來時（M8013 由 0 變 1），補位元件(M5)的狀態，被移入 M0～M2 的最低位元（M0 為 1），而原有 M2 的狀態被移入 M3（為 1），原有 M1 的狀態被移入 M2（為 0），原有 M0 的狀態被移入 M1（為 0），因為我們只用到 M0～M2，所以產生的狀態即與 S1 相同，所以另一循環由此開始。

⑤重覆①到③。

⑥當 X1 按下時，M0～M3 被復歸。

⑵利用狀態電驛的 A 接點來驅動輸出，或使用 MOV 指令將動作直接輸出。

①狀態 S1（M0）時，M1、M2（Y0、Y1）動作

②狀態 S2（M1）時，M2、M3（Y1、Y2）動作

③狀態 S3（M2）時，M3、M1（Y2、Y0）動作

得到圖 4-38 或圖 4-39。

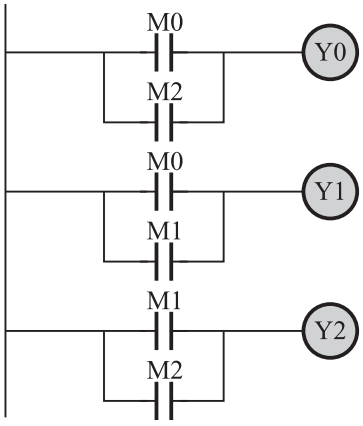


圖 4-38

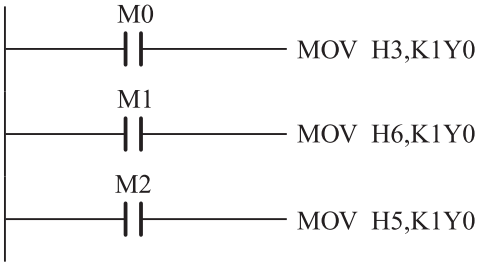


圖 4-39

⑶加上啟動及停止按鈕，完整的階梯圖如圖 4-40 或圖 4-41。

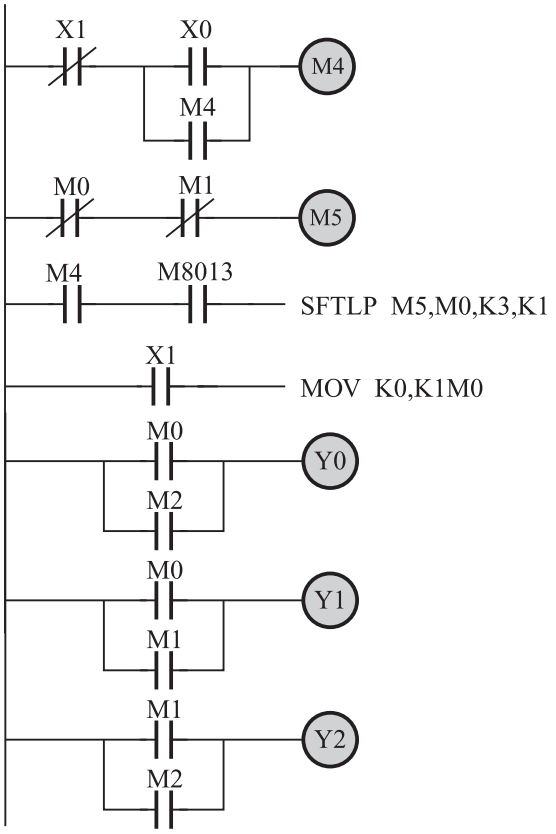


圖 4-40

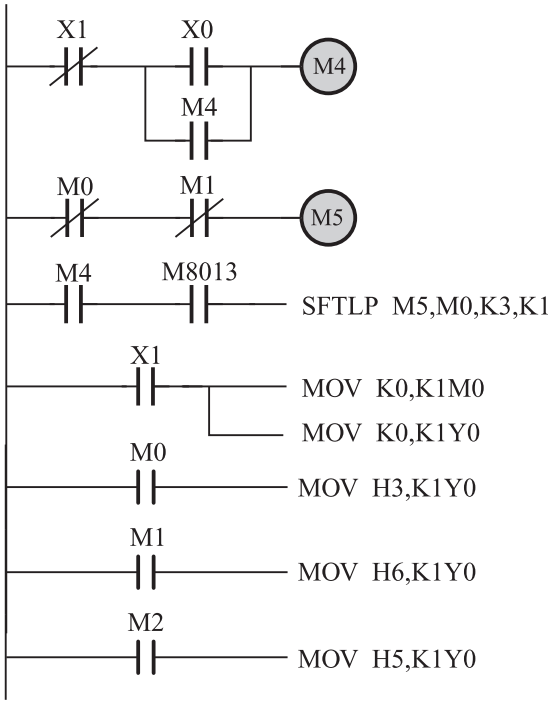


圖 4-41

*4.*撰寫程式並鍵入 PLC 中

位址	指　令	位址	指　令
0	LD　X0	19	MOV　K0,K1M0
1	OR　M4	24	LD　M0
2	ANI　X1	25	OR　M2
3	OUT　M4	26	OUT　Y0
4	LDI　M0	27	LD　M0
5	ANI　M1	28	OR　M1
6	OUT　M5	29	OUT　Y1
7	LD　M4	30	LD　M1
8	AND　M8013	31	OR　M2
9	SFTLP　M5,M0,K3,K1	32	OUT　Y2
18	LD　X1	33	END

5. 接線

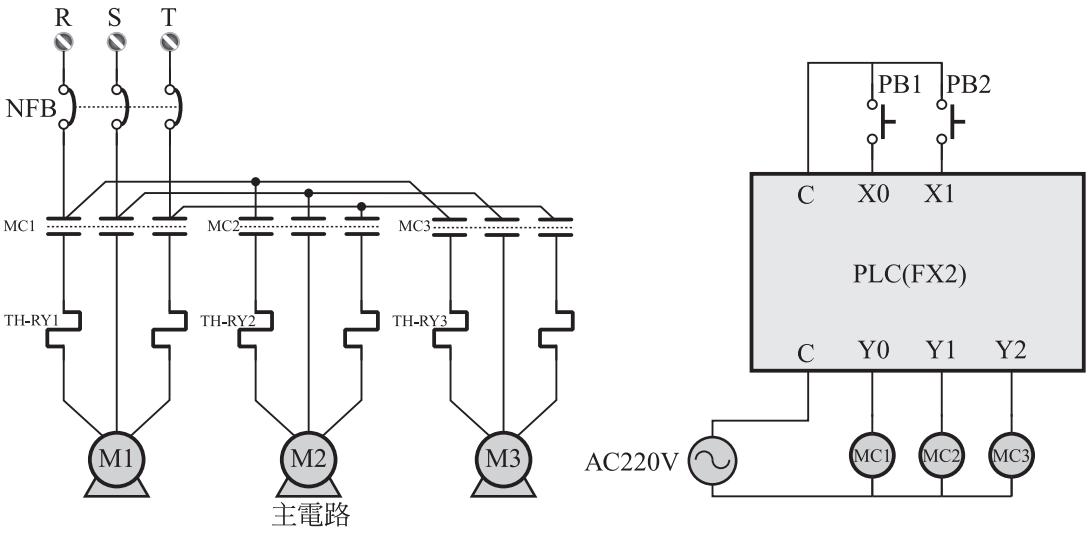


圖 4-42

6. 執行

工作二

一、題目說明：試做一自動閃爍燈（每秒閃動一次）

二、實習步驟

1. 分析：本題的施作方式是先以一 0101010101010101_2（=H5555）值存放於一般電驛中。再以旋轉指令將值做左旋轉後，經由判斷最高位元的狀態，來決定燈亮或不亮。

2. 元件編號

輸入元件	輸出元件	內部元件
PB_{OFF} →X0	L0→Y0	M0-M15
PB_{ON} →X1		M20
		M8013

3. 繪出階梯圖

⑴將 H5555 值存於 M0～M15 中，再以左旋轉指令將值左旋轉一次，經由判斷 M15 位元來決定輸出 Y0 的狀態。

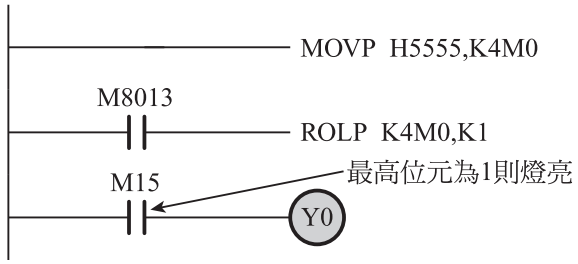


圖 4-43

⑵加上啟動及停止按鈕後全圖如圖 4-44

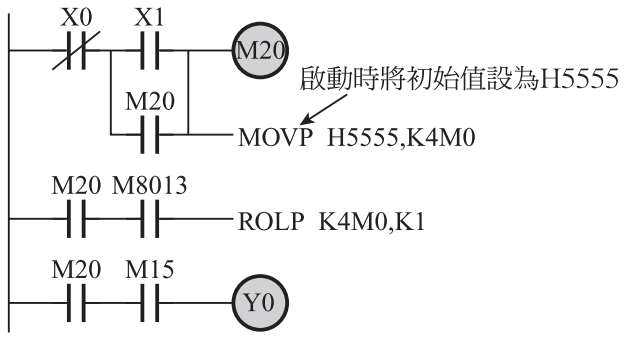


圖 4-44

4. 撰寫程式並鍵入 PLC 中

位址	指　令		位址	指　令	
0	LD	X1	10	AND	M8013
1	OR	M20	11	ROLP	K4M0,K1
2	ANI	X0	16	LD	M20
3	OUT	M20	17	AND	M15
4	MOVP	H5555,K4M0	18	OUT	Y0
9	LD	M20	19	END	

5. 接線

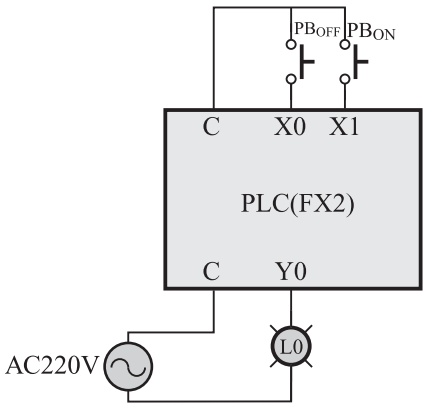


圖 4-45

6. 執行

單元十一　資料暫存器的應用

壹 學習目標

- 藉由實習例題的解說，你能熟悉各種資料暫存器的使用法，也讓你的設計能力逐步加強。

貳 相關知識

一、資料暫存器的應用

資料暫存器（D）的應用：FX2 提供了大量的記憶體區來做為暫存器，讓使用者可以暫存資料。它是以 D 之後加上一個號碼來代表，例如 D0、D200 等。每一個暫存器位置均有 16 位元的容量。但是它與一般電驛不同的是，它不像一般電驛可以以一次一位元或一次十六位元為單位來存取，D 只能以一次十六位元的方式來存取。FX2 總共提供了從 D0～D511 等 512 個資料暫存器位置供使用者使用，其中 D200～D511 具有停電保持的功能。

二、資料暫存器的使用

1. 將資料輸入到暫存器的方式有兩種

⑴在程式中以 MOV 指令來輸入，如圖 4-46，此種方法的缺點是程式過長，不但書寫時間加長，相對的掃描時間也加長，容易造成輸出入存取的不正常（來不及掃描到外界的輸入信號）。另外假如狀態的顯示需要隨時改變（廣告燈不能一成不變的），則勢必要進入程式，一條條的修改、刪減或插入，真是曠日費時。所以它並不適合狀態較多的場合。

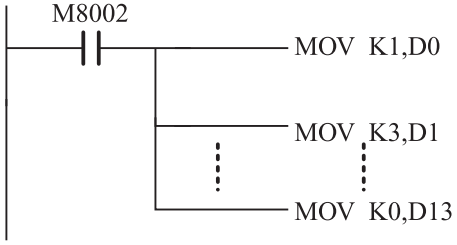


圖 4-46

⑵另外一種方式是不必寫入程式，而直接將數據輸入到 D 中，稱之為直接輸入法。因為它不用以程式指令來輸入，所以如上圖的程式就省下來了。另外當要改變顯示狀態時，則只要改變 D 的值就可以了，根本不用動到程式。所以一般都採用此種方

法。例如欲將下列數值資料分別存入到 D 的連續位址上，則其過程如下：

數值　　D 位址

K1 ⟶ D0

K5 ⟶ D1

K30 ⟶ D2

K8 ⟶ D3

按 鍵 順 序	螢幕顯示
①選擇監視模式 MNT/TEST	M ▶ 0 LD X 001 1 OR Y 001 2 ANI X 000 3 OUT Y 000
②輸入欲輸入元件 ,/SP → D → 0 → GO	M ▶ D 0 K 0
③選擇測試模式 MNT/TEST	T ▶ D 0 K 0
④輸入數值 ,/SP → K → 1 → GO	T ▶ D 0 K 1
⑤往下跳到 D1 位置 ↓	T D 0 K 1 ▶ D 1 K 0
⑥重覆④到⑤直到如右表所示	T ▶ D 0 K 1 D 1 K 5 D 2 K 30 D 3 K 8

圖 4-47

2. 輸出程式的撰寫也可以分成兩種：

⑴直接輸出法：就是只要將剛剛鍵入的資料，再一一取出輸出即可，如圖 4-48。一樣的，此種方法的缺點也是程式過長、書寫時間及掃描時間也長、容易造成輸出入存取的不正常等。另外假如狀態的數量及顯示需要隨時改變，則勢必要進入程式，大

量更動，這樣是不具效率的。

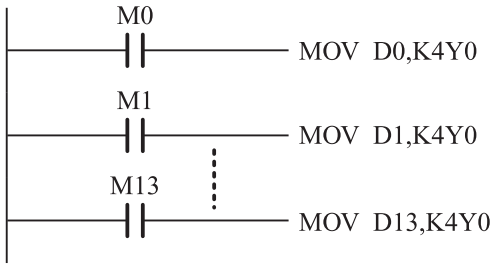
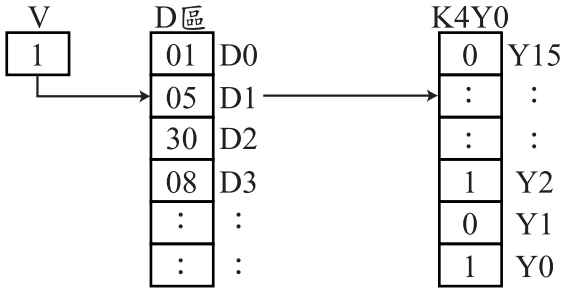


圖 4-48

⑵間接定址輸出法：

①就是選用一個索引暫存器（V 或 Z）來存放位置指標，其內的值為位置指標值而非數據值。使用時只要在 D 後面加上 V 或 Z，即可指出暫存器位置，如下表若 V 之內容若為 1，則 MOV D0V,K4Y0 的意思是將 D1(D[0+1]=D1)的內容（05）輸出給 K4Y0。所以 Y0 及 Y2 動作。



②根據間接定址法，我們只要每次改變 V 內的值，就可以到不同的資料暫存器位置取出狀態數據並予以輸出。當然每輸出一個狀態數據後，即應將指標 V 自動加 1，以便繼續輸出下一個狀態。階梯圖如圖 4-49。空白部份為定時的部份。

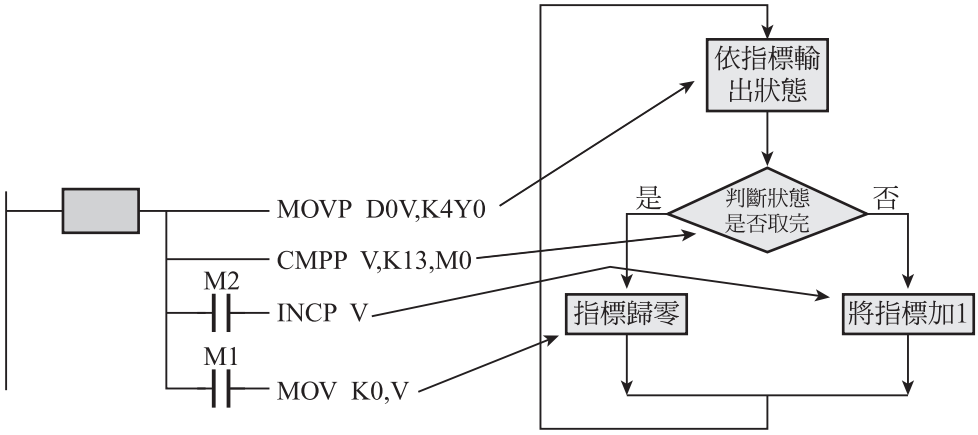


圖 4-49

參 實習

一、題目說明：試設計一廣告燈電路，總共有六個燈，各以 L0～L5 來代表，希望各燈的變化情形如下：

T T T T T
L0→L0,L1→L0,L1,L2→L0,L1,L2,L3→L0,L1,L2,L3,L4→L0,L1,L2,L3,
T T T T T T
L4,L5→L0,L1,L2,L3,L4→L0,L1,L2,L3→L0,L1,L2→L0,L1→L0 →全
T T
熄→全亮→全熄

二、實習步驟

1. 狀態分析：我們的設計方法是先將每一種狀態的對應值存入具有停電保持功能的資料暫存器(D200～)中，如下表。最後再依一定的時間間隔來將存在各暫存器位置的資料（數值）輸出給輸出元件即可。

狀態	暫存器位置	二進位值						16進位值
		L5	L4	L3	L2	L1	L0	
0	D200	0	0	0	0	0	1	H0001
1	D201	0	0	0	0	1	1	H0003
2	D202	0	0	0	1	1	1	H0007
3	D203	0	0	1	1	1	1	H000F
4	D204	0	1	1	1	1	1	H001F
5	D205	1	1	1	1	1	1	H003F
6	D206	0	1	1	1	1	1	H001F
7	D207	0	0	1	1	1	1	H000F
8	D208	0	0	0	1	1	1	H0007
9	D209	0	0	0	0	1	1	H0003
10	D210	0	0	0	0	0	1	H0001
11	D211	0	0	0	0	0	0	H0000
12	D212	1	1	1	1	1	1	H003F
13	D213	0	0	0	0	0	0	H0000

2. 元件編號

輸入元件	輸出元件	內部元件
PB_{OFF} →X0	L0→Y0 L3→Y3	D200～D213
PB_{ON} →X1	L1→Y1 L4→Y4	M0～M2，M4
	L2→Y2 L5→Y5	M8013

3. 繪出階梯圖

⑴將資料用直接輸入法輸入到資料暫存器(D200～D213)中。

⑵鍵好暫存器資料後，將資料用間接定址法輸出。

本例使用特殊電驛M8013來產生一個1秒鐘的脈波。如圖4-50。

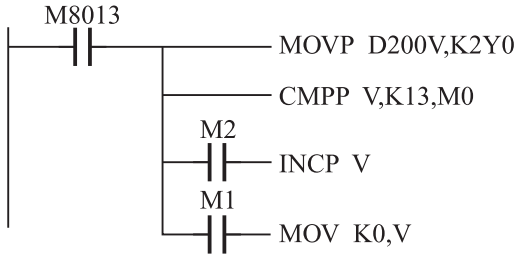


圖 4-50

⑶加上啟動及停止按鈕後，全圖如圖 4-51。當然你一定要先將所有的狀態數據先用直接鍵入法鍵入資料暫存器（D200～D213）中。圖 4-51 的優點是，不管你有多少狀態，程式的長度都不會改變，但需更改次數（圖中的 K13）。

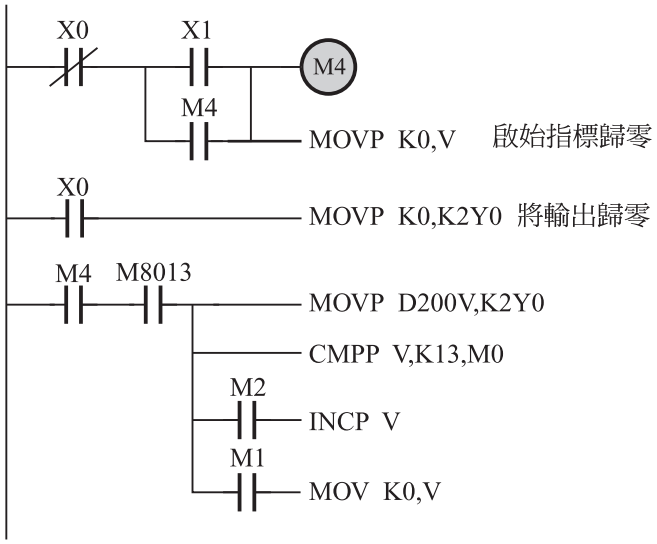


圖 4-51

⑷當然你也可以用另一個資料暫存器位置來放次數（D250-也是用直接鍵入法鍵入）。這樣子就都不用更動程式了。如圖 4-52 所示。

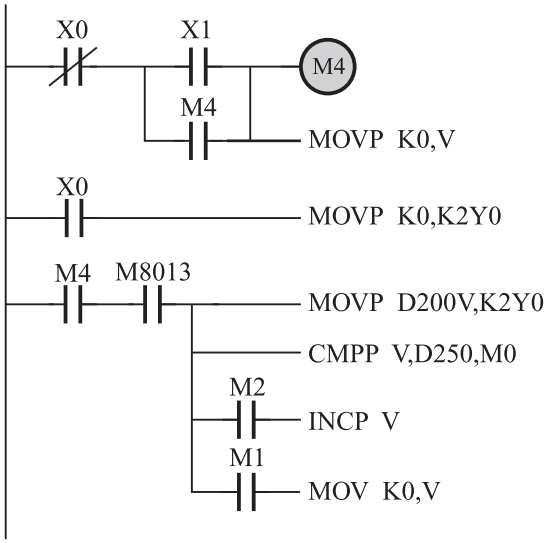


圖 4-52

4. 撰寫程式並鍵入 PLC 中

位址	指　令	位址	指　令
0	LD X1	17	MPS
1	OR M4	18	MOVP D200V,K2Y0
2	ANI X0	23	CMPP V,D250,M0
3	OUT M4	30	AND M2
4	MOVP K0,V	31	INCP V
9	LD X0	34	MPP
10	MOVP K0,K2Y0	35	AND M1
15	LD M4	36	MOV K0,V
16	AND M8013	41	END

5. 接線

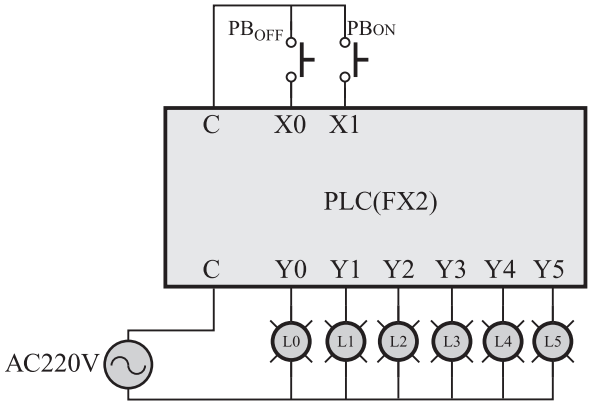

PBOFF
PBON
C
X0
X1
PLC(FX2)
C
Y0
Y1
Y2
Y3
Y4
Y5
L0
L1
L2
L3
L4
L5
AC220V

圖 4-53

6. 執行