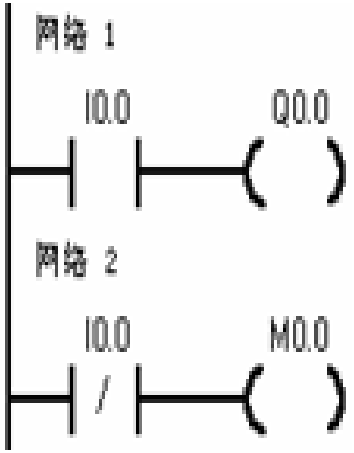
可编程控制器原理及应用

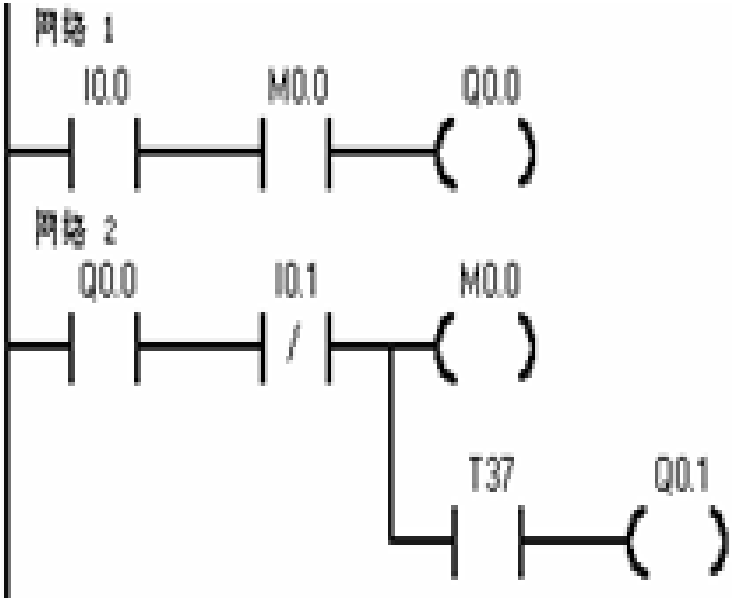
梯形图



语句表

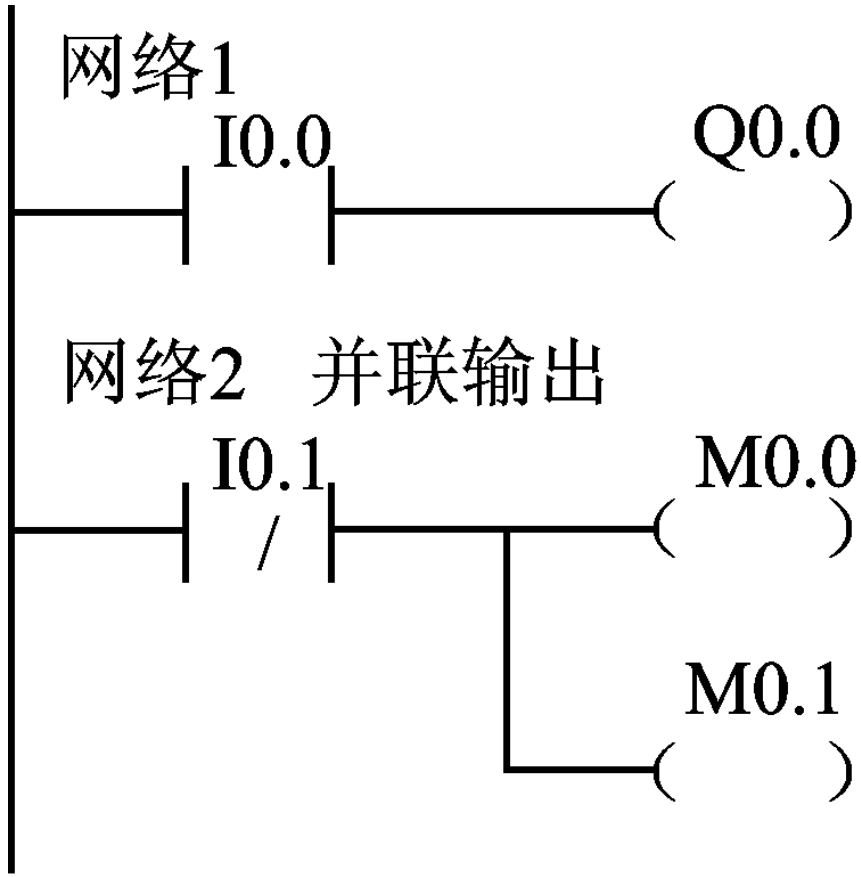
```
网络 1
LD    I0.0     //装载常开触点
=     Q0.0     //输出线圈
网络 2
LDN   I0.0     //装载常闭触点
=     M0.0     //输出线圈
```

可编程控制器原理及应用




```
网络 1
LD    I0.0      //装载常开触点
A     M0.0      //与常开触点
=     Q0.0      //输出线圈
网络 2
LD    Q0.0      //装载常开触点
AN    I0.1      //与常闭触点
=     M0.0      //输出线圈
A     T37       //与常开触点
=     Q0.1      //输出线圈
```

可编程控制器原理及应用



(a) 梯形图

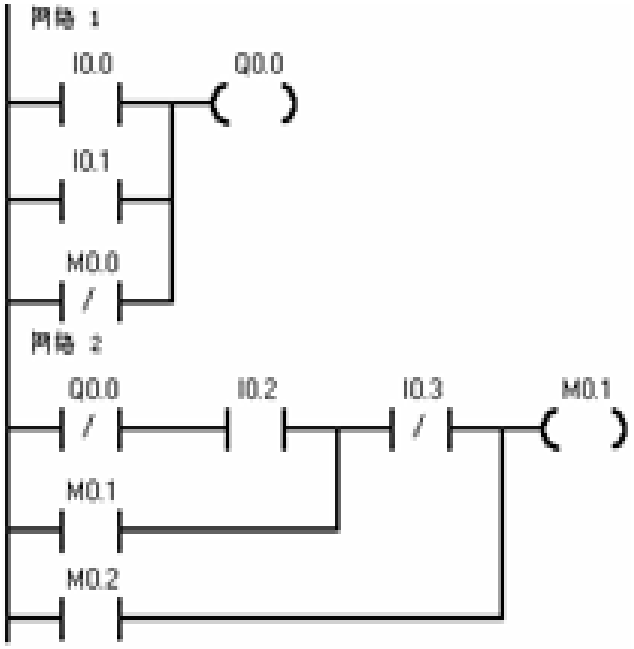
```
LD    I0.0
=     Q0.0

LDN   I0.1
=     M0.0
=     M0.1
```

(b) 语句表

可编程控制器原理及应用

梯形图



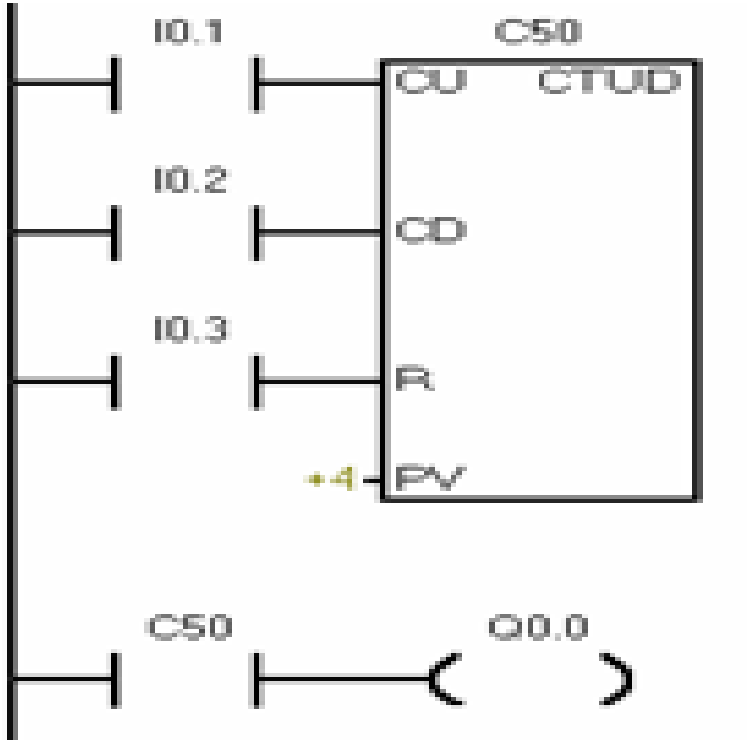
网络1

```
LD      I0.0
O       I0.1
ON      M0.0
=       Q0.0
```

网络2

```
LDN     Q0.0
A       0.2
O       M0.1
AN      I0.3
O       M0.2
=       M0.1
```

可编程控制器原理及应用




```
LD    I0.0
LD    I0.1
LD    I0.3
CTUD  C50,+4
LD    C50
=     Q0.0
```

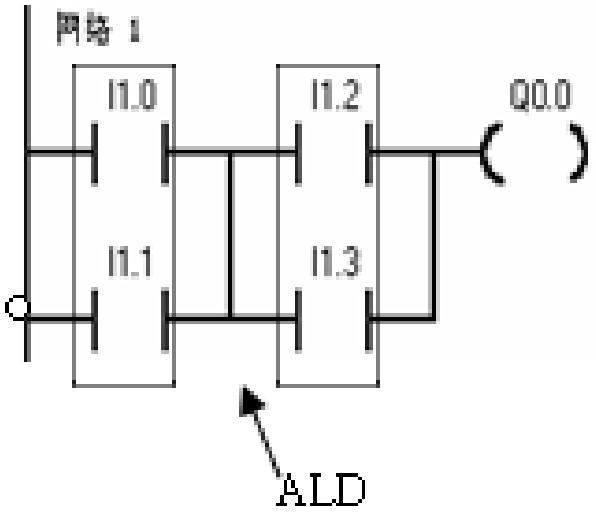
电路块的串联指令ALD

ALD：块“与”操作
用于并联电路块的串联连接

什么是并联电路块？
两条以上支路并联形成的电路叫并联电路块

可编程控制器原理及应用

- 梯形图



- 语句表

```
LD   I1.0   //装入常开触点
O    I1.1   //或常开触点
LD   I1.2   //装入常开触点
O    I1.3   //或常开触点
ALD         //块与操作
=    Q0.0   //输出线圈
```

使用说明

在块电路开始时要使用LD和LDN指令

在每完成一次块电路的串联连接后要写上ALD指令

ALD指令无操作数

电路块的并联指令OLD

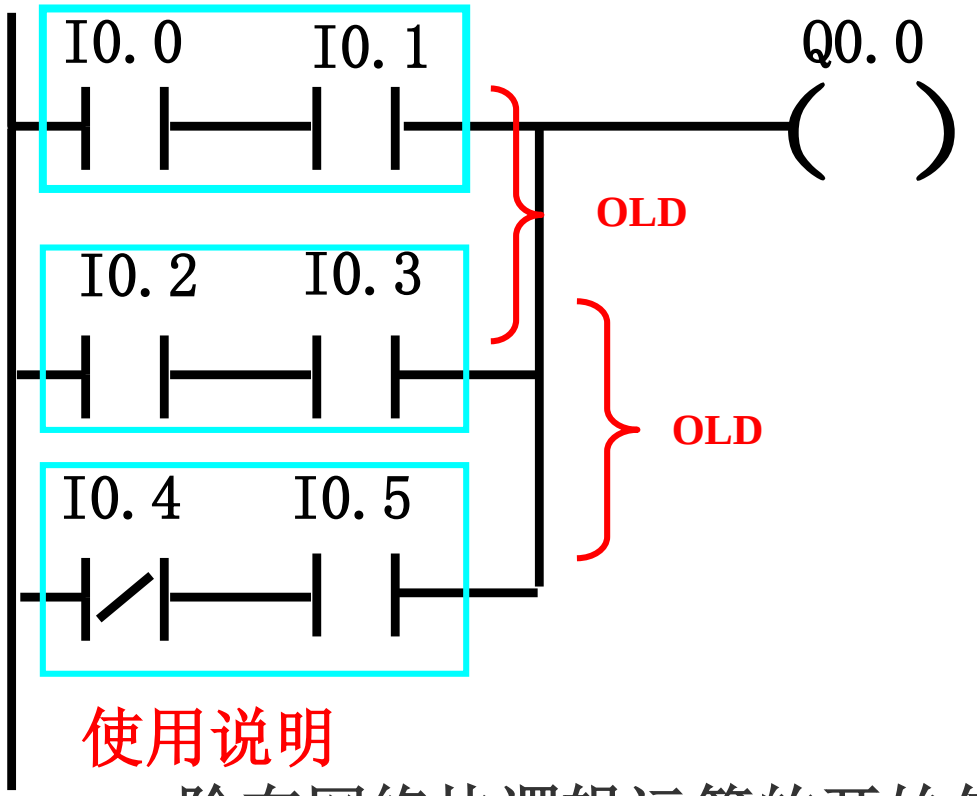
OLD：块“或”操作

用于串联电路块的并联连接

什么是串联电路块？

两个以上触点串联形成的支路叫串联电路块

可编程控制器原理及应用



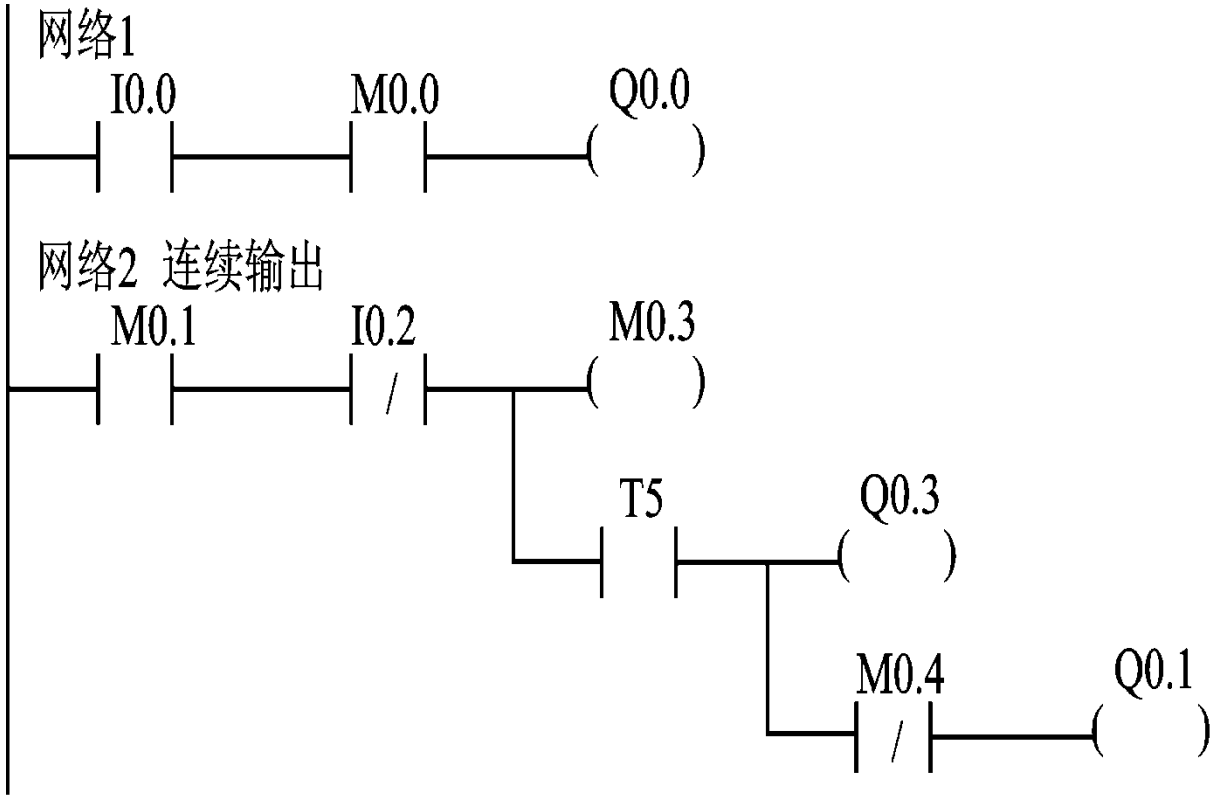

```
LD    I0.0
A     I0.1
LD    I0.2
A     I0.3
OLD
LDN   I0.4
A     I0.5
OLD
=     Q0.0
```

使用说明

除在网络块逻辑运算的开始使用LD或LDN指令外，在块电路的开始也要使用LD和LDN指令

每完成一次块电路的并联时要写上OLD指令

OLD指令无操作数



(a) 梯形图

```
LD    I0.0
A     M0.0
=     Q0.0

LD    M0.1
AN    I0.2
=     M0.3
A     T5
=     Q0.3
AN    M0.4
=     Q0.1
```

(b) 语句表

堆栈指令

又称多分支回路指令

S7-200CN系列PLC使用一个9层堆栈来处理所有逻辑操作。堆栈是一组能够存储和取出数据的暂存单元，其特点是“先进后出”。每一次进行入栈操作，新值放入栈顶，栈底值丢失；每一次进行出栈操作，栈顶值出栈，第2级堆栈内容上升到栈顶，栈底自动生成随机数。逻辑堆栈指令主要用来完成对触点进行复杂的连接

堆栈指令

程序中使用堆栈指令的目的：

处理2路以上的多分支电路

LD装载指令是从梯形图最左侧母线画起的

如果要生成一条分支的母线

则需要利用语句表的栈操作指令来描述

LPS：入栈指令(分支电路开始指令)

作用：运算存储。

从梯形图中的分支结构中可以形象地看出，它用于生成一条新的母线，其左侧为原来的主逻辑块，右侧为新的从逻辑块，因此可以直接编程。从堆栈使用上来讲，LPS指令的作用是把栈顶值复制后压入堆栈。

LRD：读栈指令

作用：读出存储。中间分支电路使用。

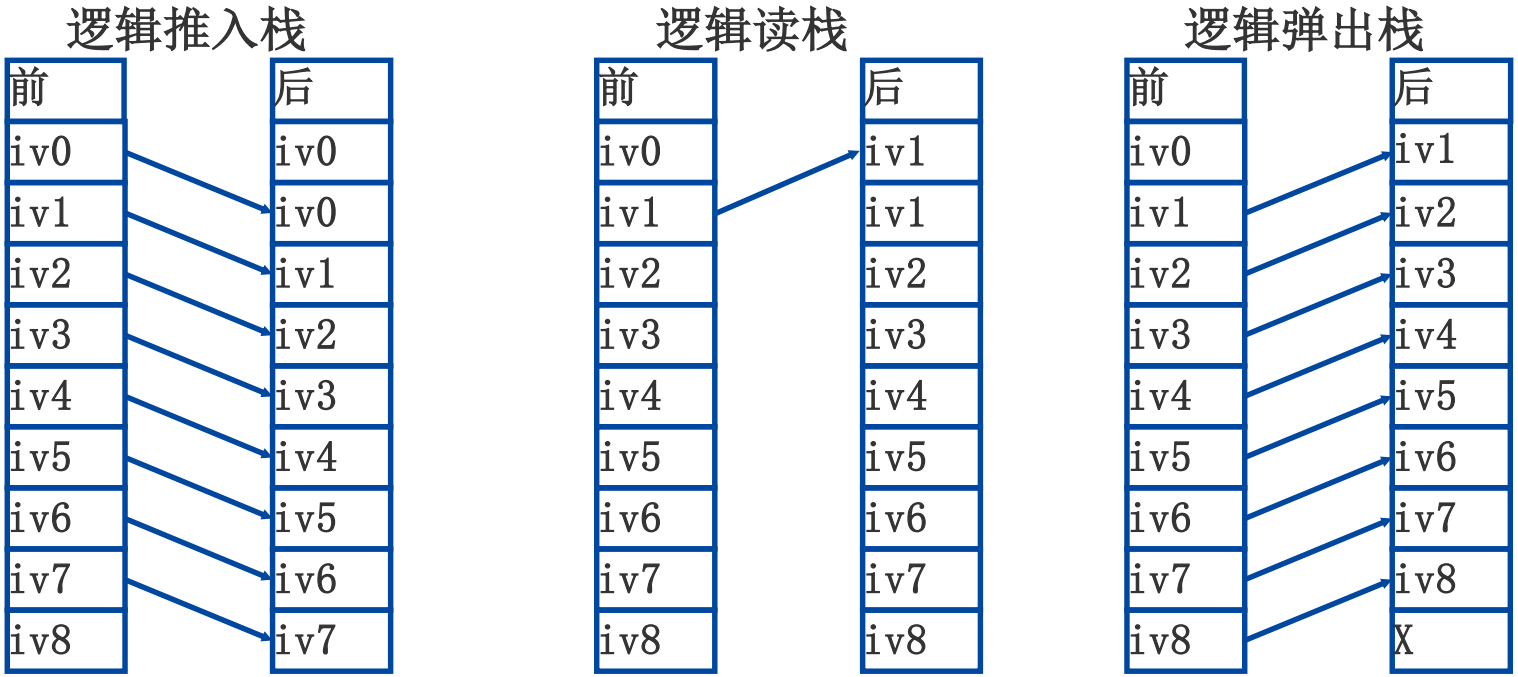
在梯形图分支结构中，当新母线左侧为主逻辑块时，LPS开始右侧的第一个从逻辑块编程，LRD开始第二个以后的从逻辑块编程。从堆栈使用上来讲，LRD读取最近的LPS压入堆栈的内容，而堆栈本身不进行Push和Pop工作。

LPP：出栈指令

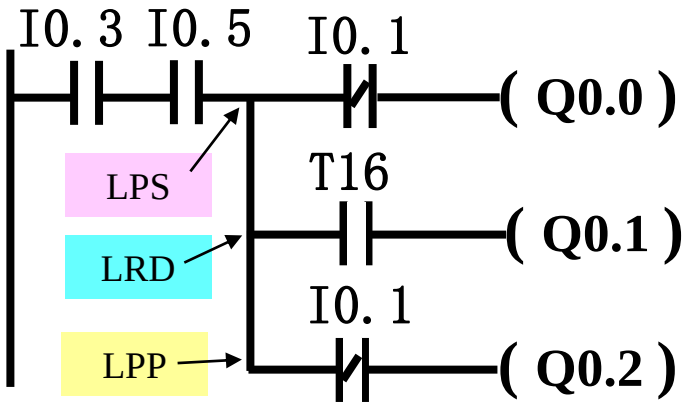
作用：读出存储或复位。 分支电路结束指令。

在梯形图分支结构中，LPP用于LPS产生的新母线右侧的最后一个从逻辑块编程，它在读完离它最近的LPS压入堆栈内容同时复位该条新母线。从堆栈使用上来讲，LPP把堆栈弹出一级，堆栈内容依次上移。

图5.4 LPS，LRD，LPP指令的操作过程



可编程控制器原理及应用



LPS：复制栈顶第0层的值，向下压一层

LRD：复制第1层的值，装到第0层

LPP：将第0层的值弹出，其他层依次上移一层

助记符语句表

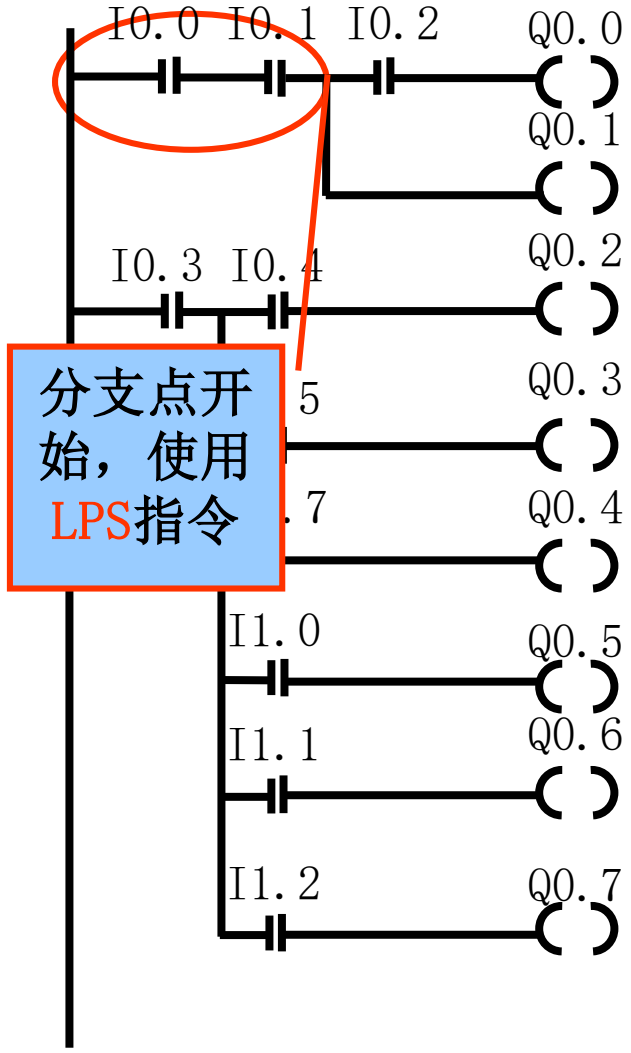
```
LD   I0.3
A    I0.5
LPS
AN   I0.1
=    Q0.0
LRD
A    T16
=    Q0.1
LPP
AN   I0.1
=    Q0.2
```

逻辑堆栈操作指令

- LPS（Logic Push）
  逻辑入栈指令（分支电路开始指令）
- LRD（Logic Read）
  逻辑读栈指令。
- LPP（Logic Pop）
  逻辑出栈指令（分支电路结束指令）

使用说明

- 由于受堆栈空间的限制（9层堆栈），LPS、LPP指令连续使用时应少于9次
- LPS和LPP指令必须成对使用，它们之间可以使用LRD指令。
- LPS、LRD、LPP指令无操作数
- 使用LPS、LPP指令时
  如果其后是单个触点
  　　使用A或AN指令
  如果其后是电路块
  　　则在电路块的起始点用LD或LDN指令
  　　然后使用块指令

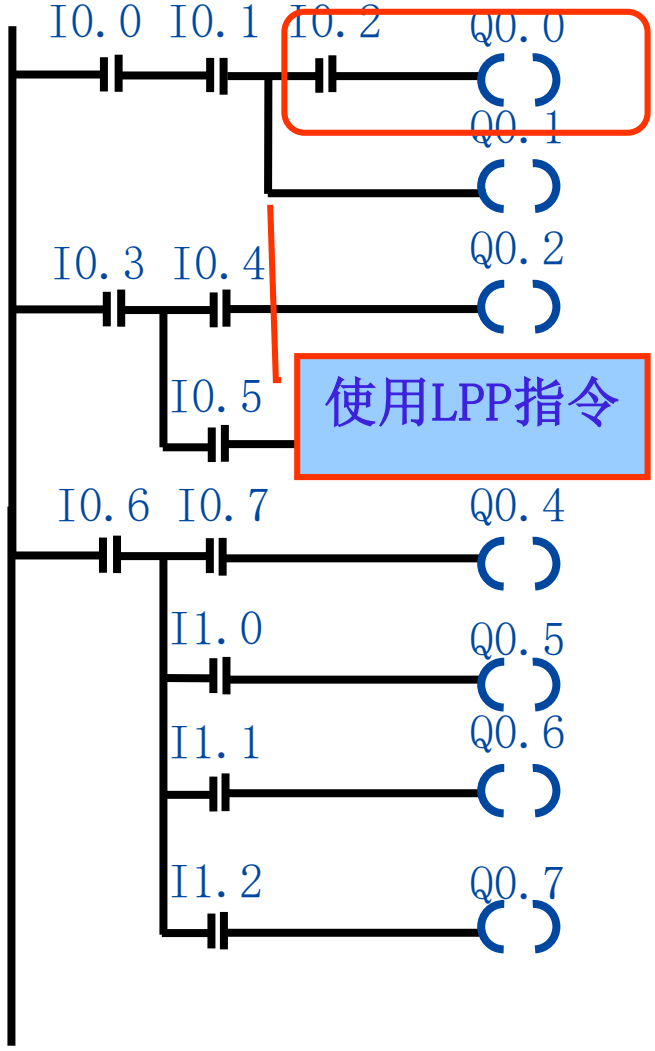


例1：一层堆栈

指令表

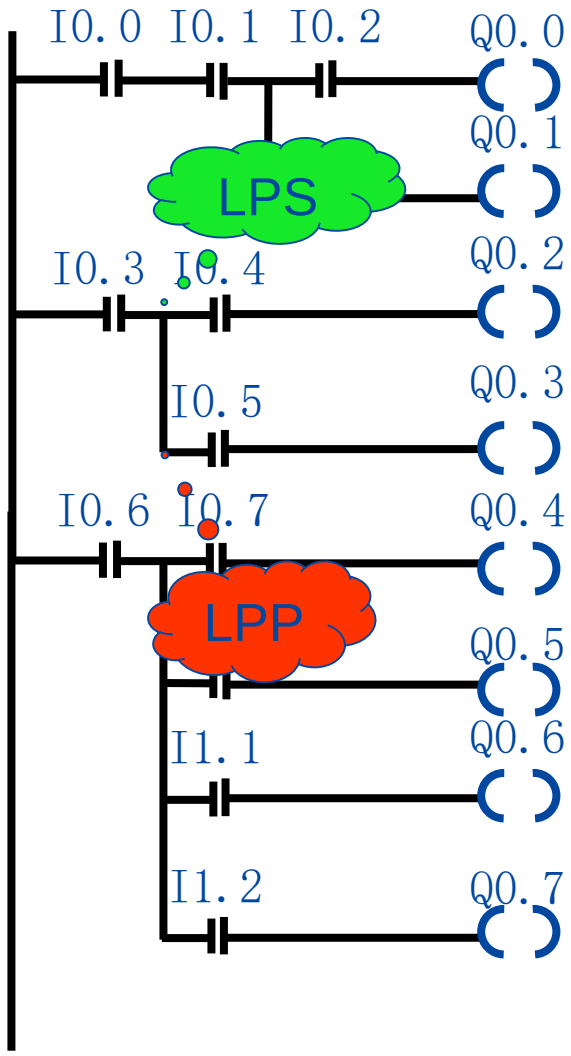
1	LD	I0.0
2	A	I0.1
3	LPS	→ 入栈

例1：一层堆栈



指令表

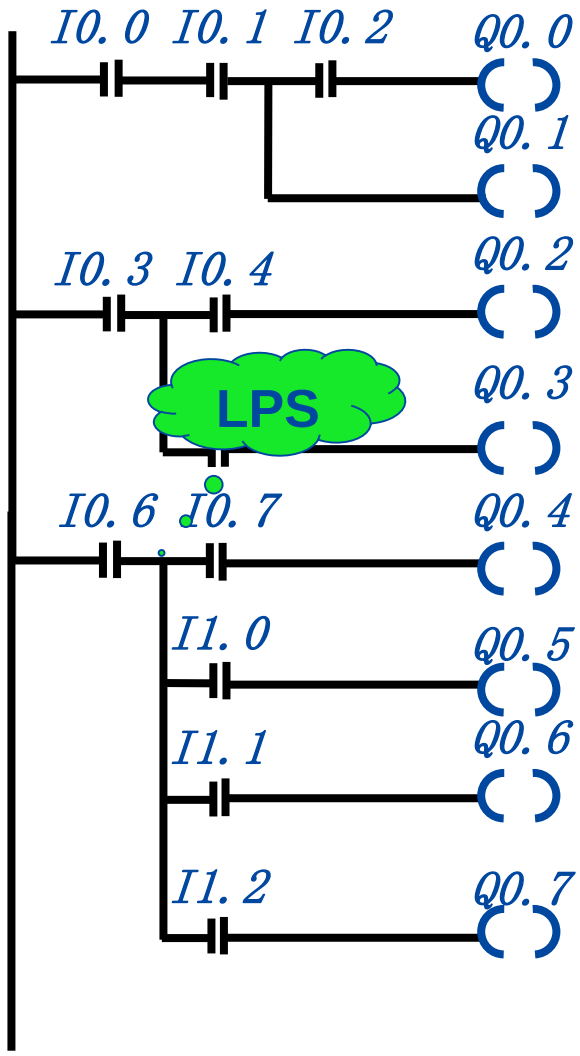
1	LD	I0.0
2	A	I0.1
3	LPS	
4	A	I0.2
5	=	Q0.0
6	LPP	→ 出栈
7	=	Q0.1



例1：一层堆栈

指令表

1	LD	I0.0	13	A	I0.5
2	A	I0.1	14	=	Q0.3
3	LPS				
4	A	I0.2			
5	=	Q0.0			
6	LPP				
7	=	Q0.1			
8	LD	I0.3			
9	LPS				
10	A	I0.4			
11	=	Q0.2			
12	LPP				

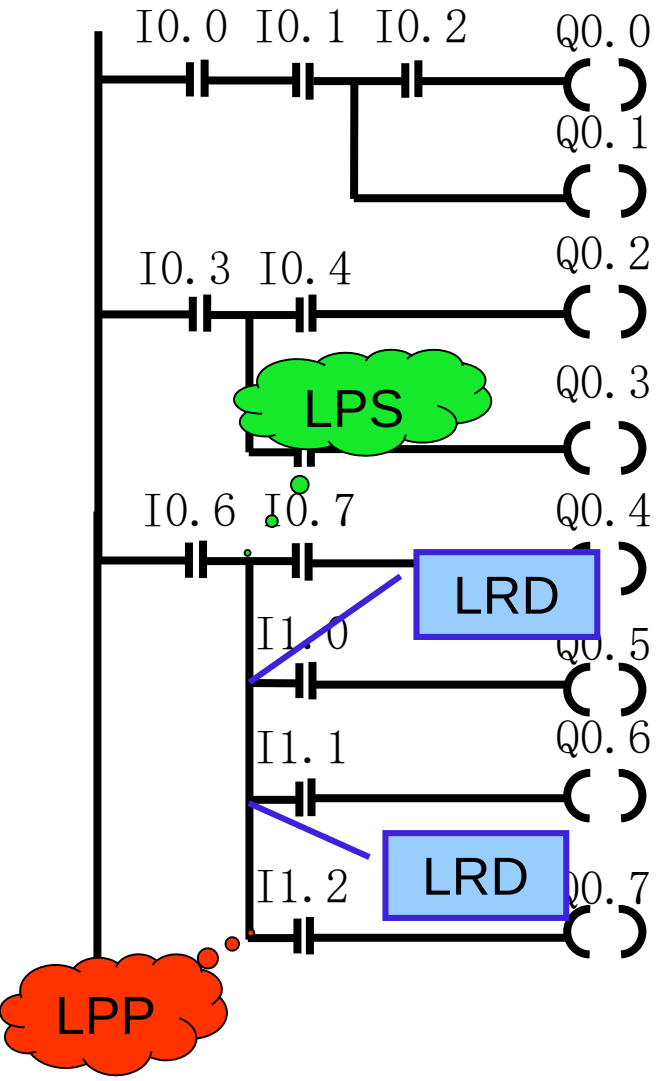


例1：一层堆栈

指令表

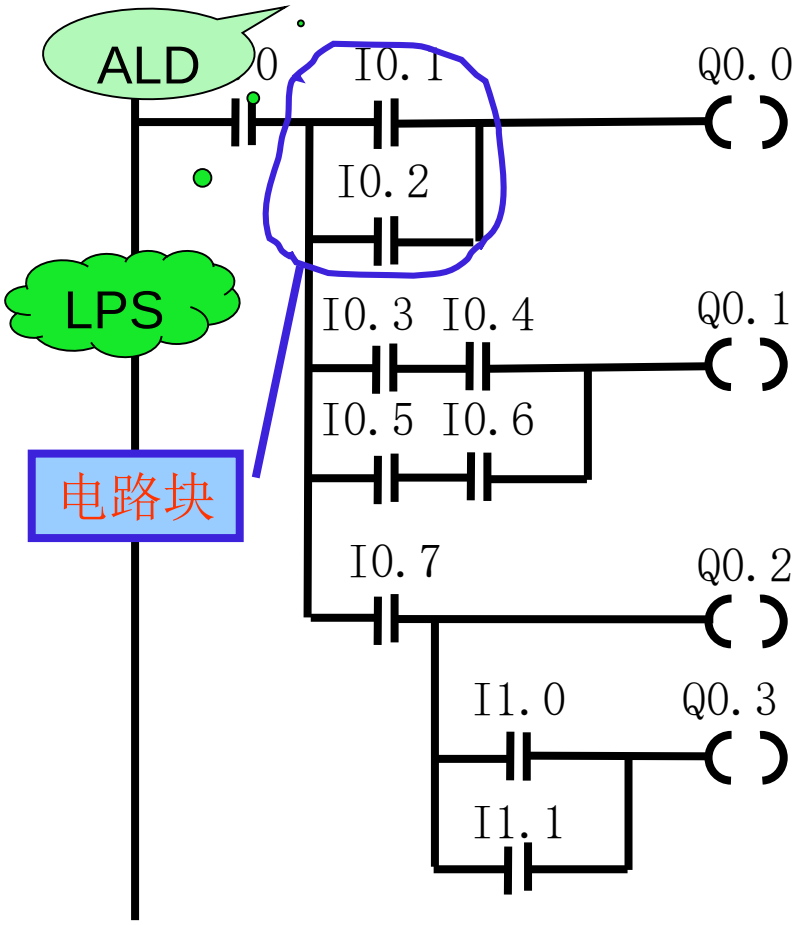
1	LD	I0.0
2	A	I0.1
3	LPS	
4	A	I0.2
5	＝	Q0.0
6	LPP	
7	＝	Q0.1
8	LD	I0.3
9	LPS	
10	A	I0.4
11	＝	Q0.2
12	LPP	
13	A	I0.5
14	＝	Q0.3
15	LD	I0.6
16	LPS	
17	A	I0.7
18	＝	Q0.4

例1：一层堆栈



指令表

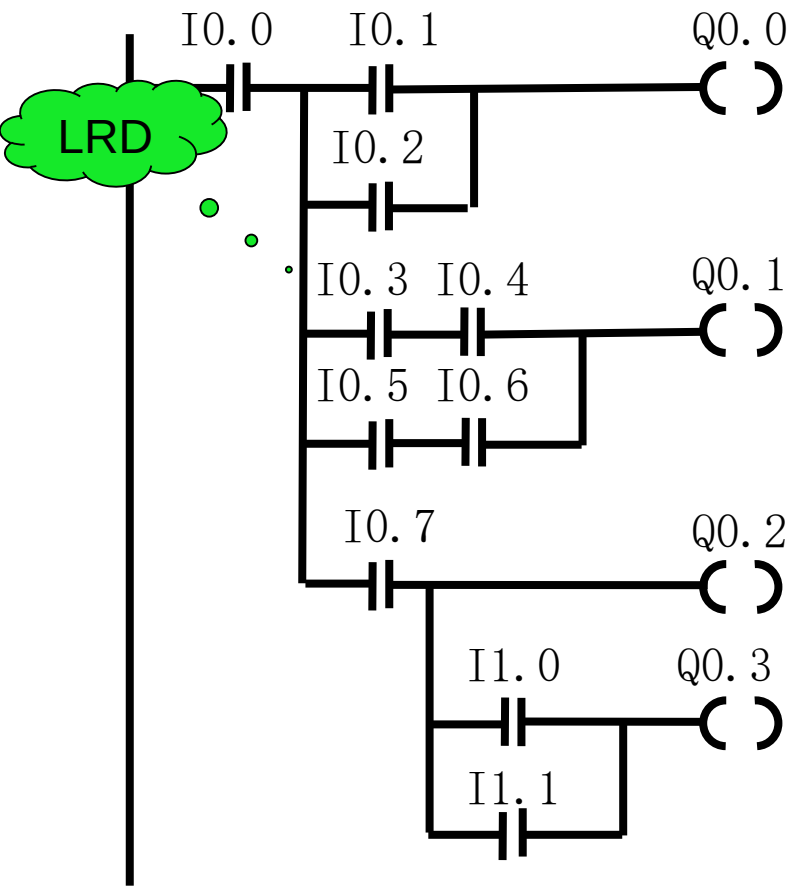
1	LD	I0.0
2	A	I0.1
3	LPS	
4	A	I0.2
5	=	Q0.0
6	LPP	
7	=	Q0.1
8	LD	I0.3
9	LPS	
10	A	I0.4
11	=	Q0.2
12	LPP	
13	A	I0.5
14	=	Q0.3
15	LD	I0.6
16	LPS	
17	A	I0.7
18	=	Q0.4
19	LRD	→ 读栈
20	A	I1.0
21	=	Q0.5
22	LRD	→ 读栈
23	A	Q1.1
24	=	Q0.6
25	LPP	
26	A	I1.2
27	=	Q0.7



例2：一层堆栈(并用ALD、OLD指令）

指令表

```
1   LD   I0.0
2   LPS
3   LD   I0.1
4   O    I0.2
5   ALD
6   =    Q0.0
```

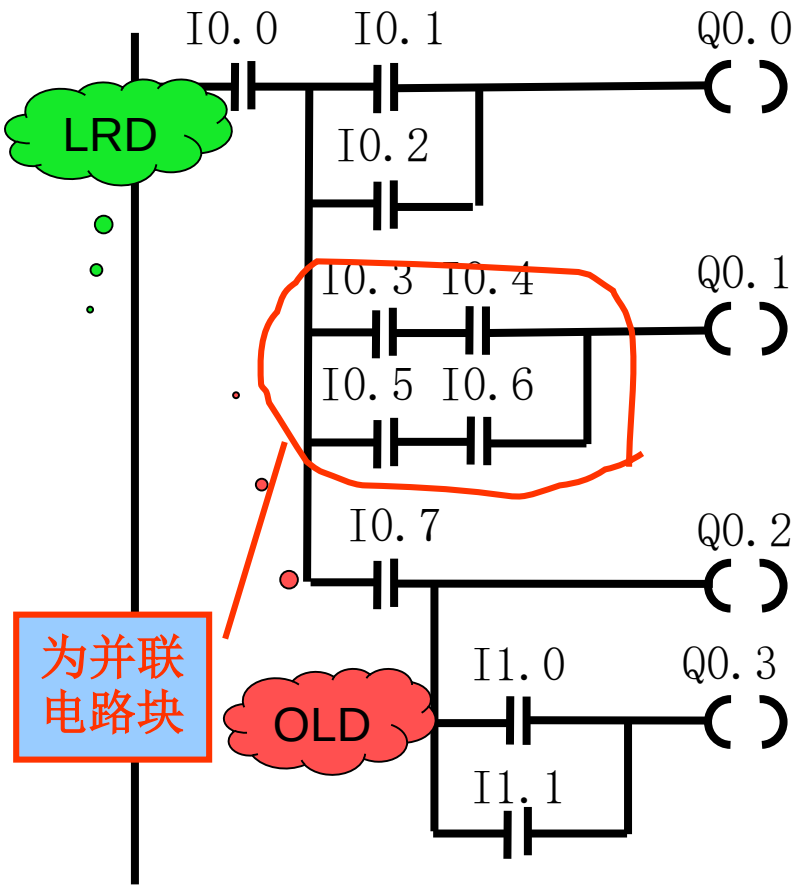


例2：一层堆栈(并用ALD、OLD指令）

指令表

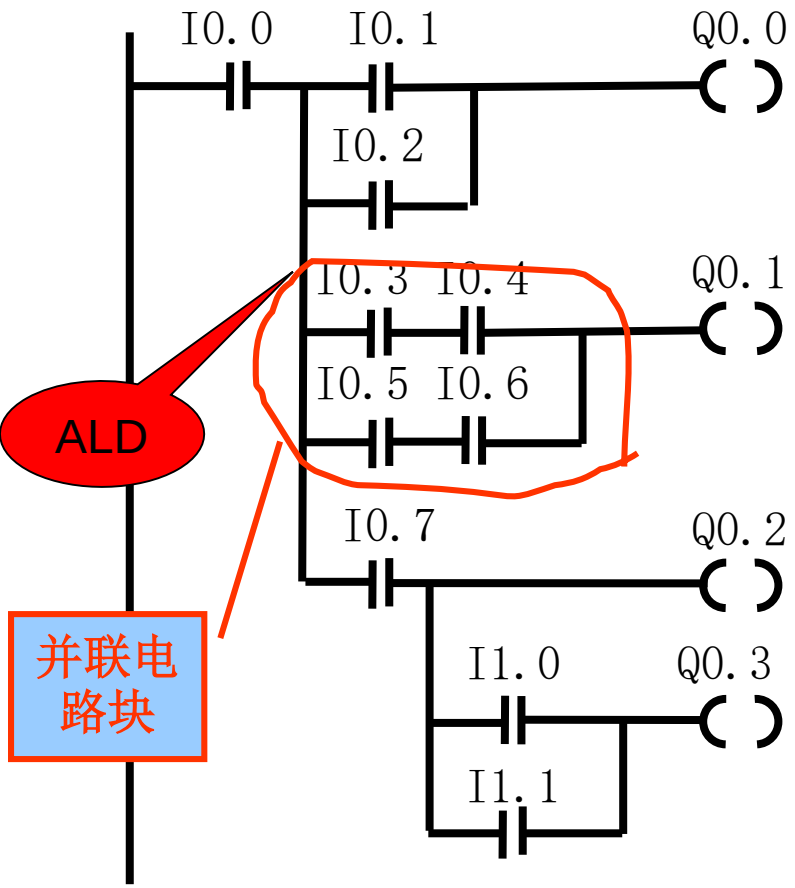
1	LD	I0.0
2	LPS	
3	LD	I0.1
4	O	I0.2
5	ALD	
6	=	Q0.0

例2：一层堆栈(并用ALD、OLD指令）



指令表

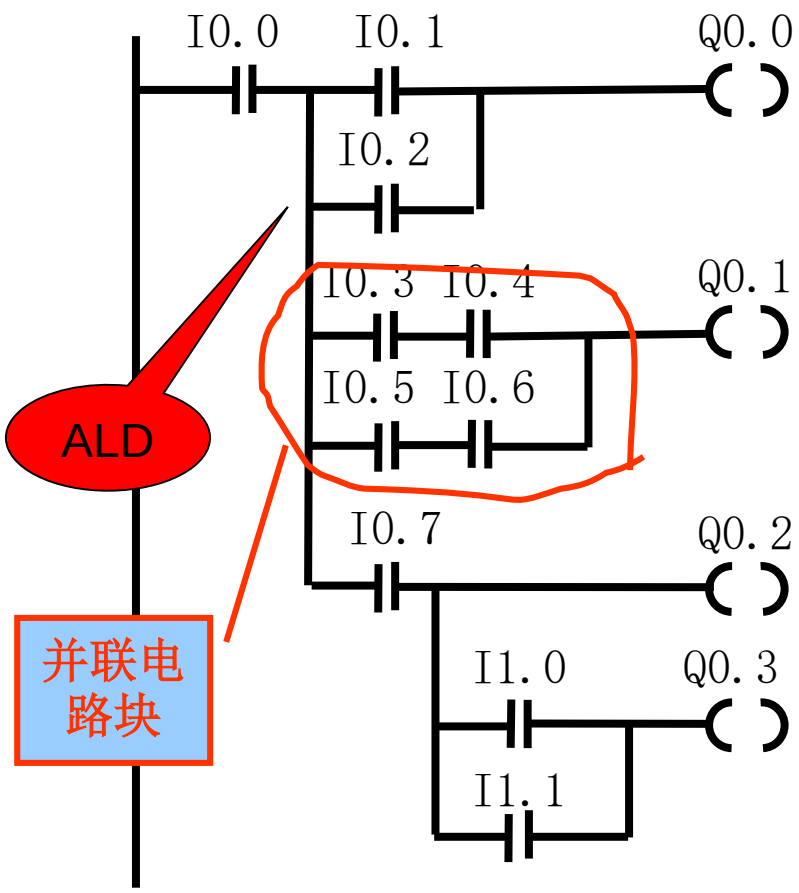
1	LD	I0.0
2	LPS	
3	LD	I0.1
4	O	I0.2
5	ALD	
6	＝	Q0.0
7	LRD	
8	LD	I0.3
9	A	I0.4
10	LD	I0.5
11	A	I0.6
12	OLD	



例2：一层堆栈(并用ALD、OLD指令）

指令表

```
1   LD    I0.0
2   LPS
3   LD    I0.1
4   A     I0.2
5   ALD
6   =     Q0.0
7   LRD
8   LD    I0.3
9   A     I0.4
10  LD    I0.5
11  A     I0.6
12  OLD
```

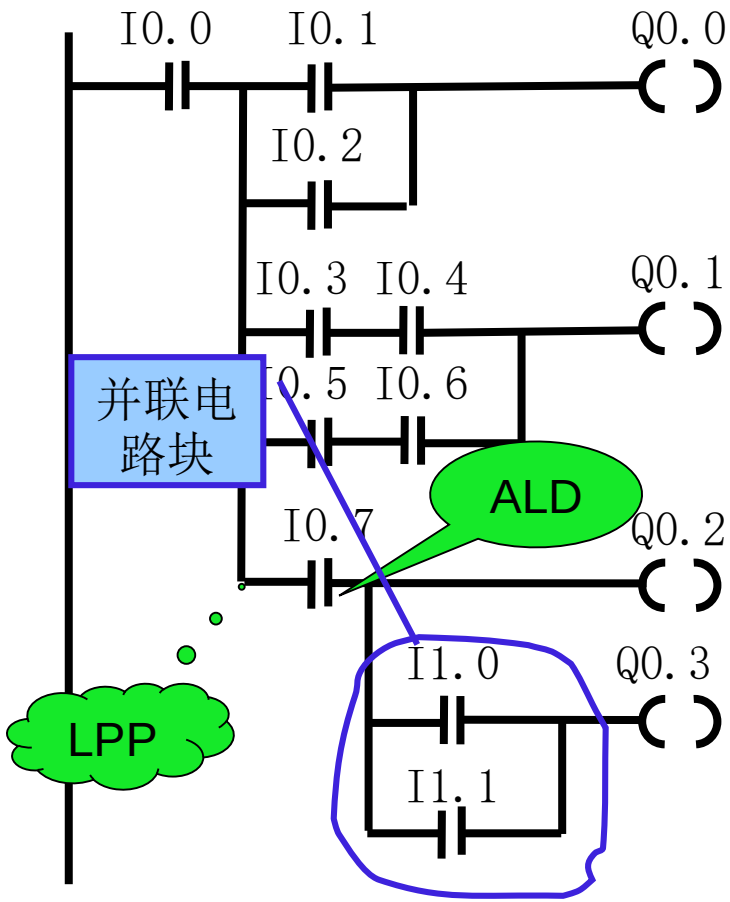


例2：一层堆栈(并用ALD、OLD指令）

指令表

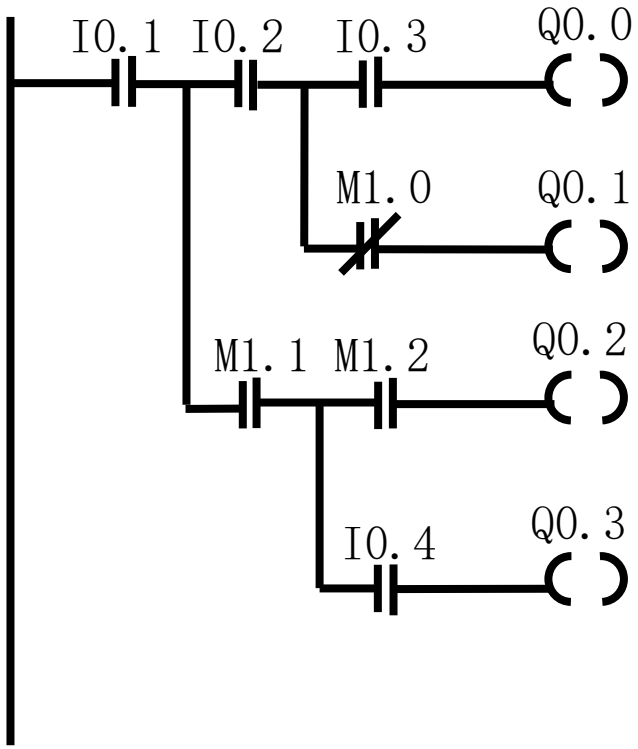
1	LD	I0.0	13	ALD	
2	LPS		14	＝	Q0.1
3	LD	I0.1			
4	A	I0.2			
5	ALD				
6	＝	Q0.0			
7	LRD				
8	LD	I0.3			
9	A	Q0.4			
10	LD	Q0.5			
11	A	Q0.6			
12	OLD				

例2：一层堆栈(并用ALD、OLD指令）



指令表

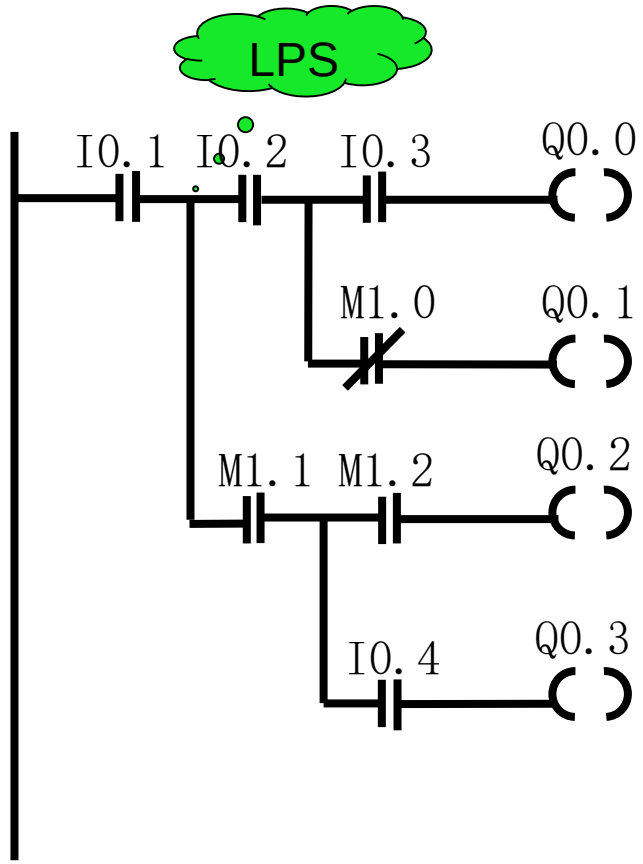
1	LD	I0.0	13	ALD		
2	LPS		14	=	Q0.1	
3	LD	I0.1	15	LPP		
4	O	I0.2	16	A	I0.7	连续输出形式
5	ALD		17	=	Q0.2	
6	=	Q0.0	18	LD	I1.0	
7	LRD		19	O	I1.1	
8	LD	I0.3	20	ALD		
9	A	I0.4	21	=	Q0.3	
10	LD	I0.5	22	MEND		
11	A	I0.6				
12	OLD					



例3：二层堆栈

指令表

1	LD	I0.1

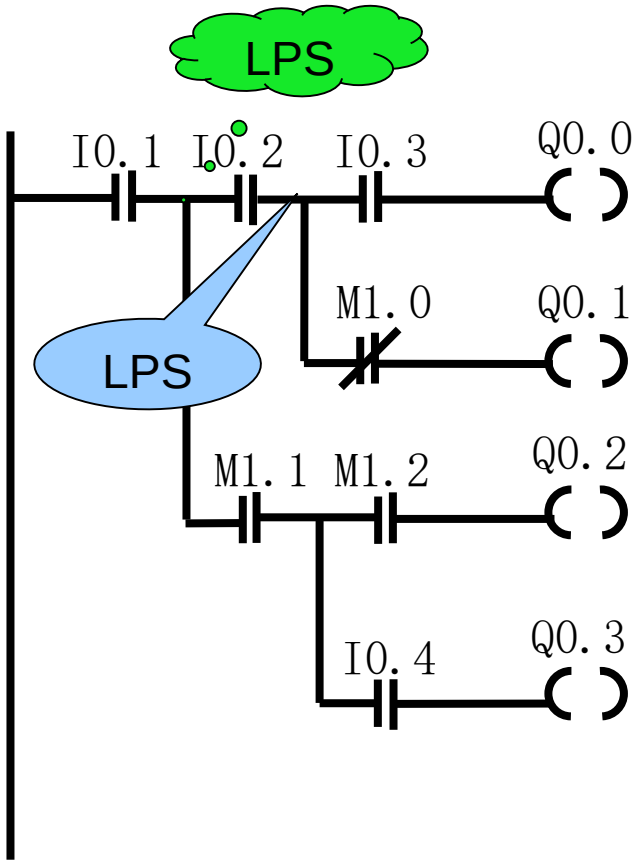


例3：二层堆栈

指令表

```
1    LD    I0.1
```

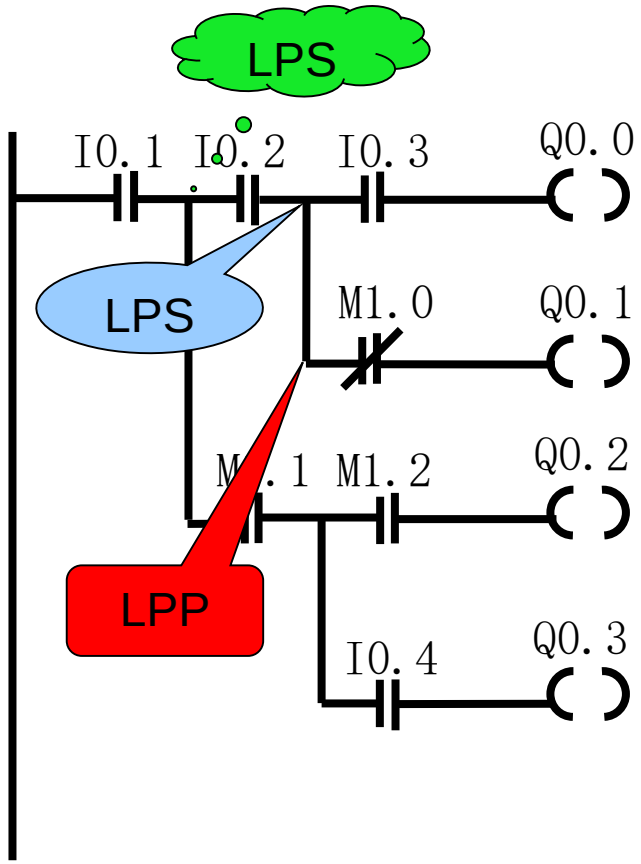
例3：二层堆栈



指令表

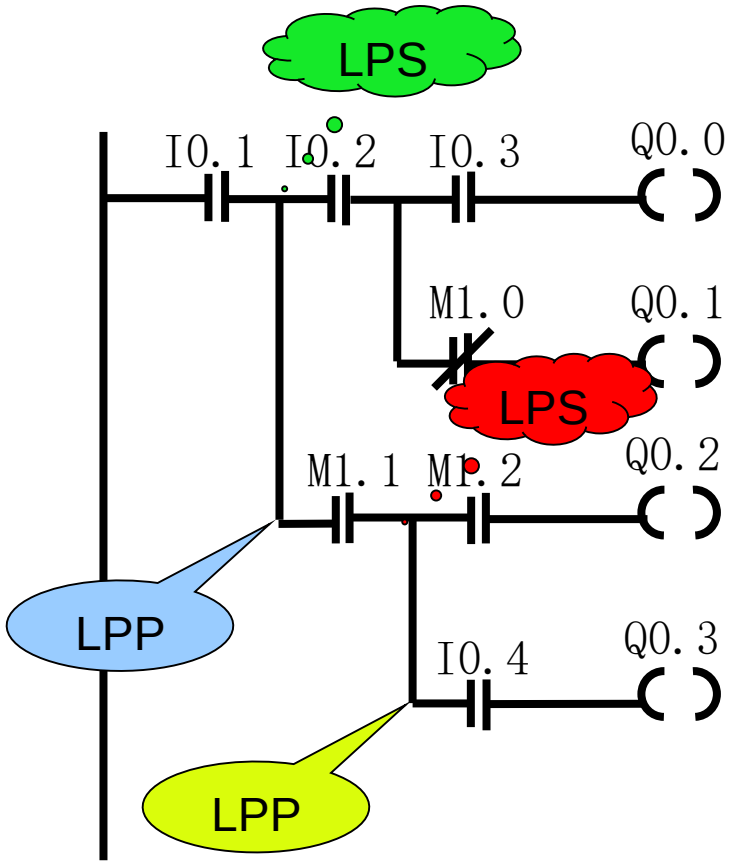
1	LD	I0.1
2	LPS	→ 第一层入栈
3	A	I0.2

例3：二层堆栈



指令表

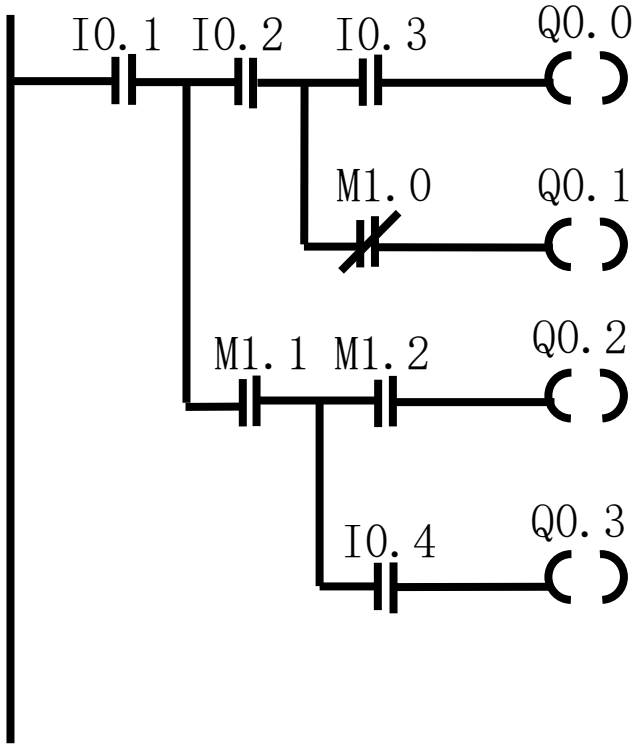
1	LD	I0.1	
2	LPS		→ 第一层入栈
3	A	I0.2	
4	LPS		→ 第二层入栈
5	A	I0.3	
6	=	Q0.0	
7	LPP		→ 第二层出栈
8	AN	M1.0	
9	=	Q0.1	



例3：二层堆栈

指令表

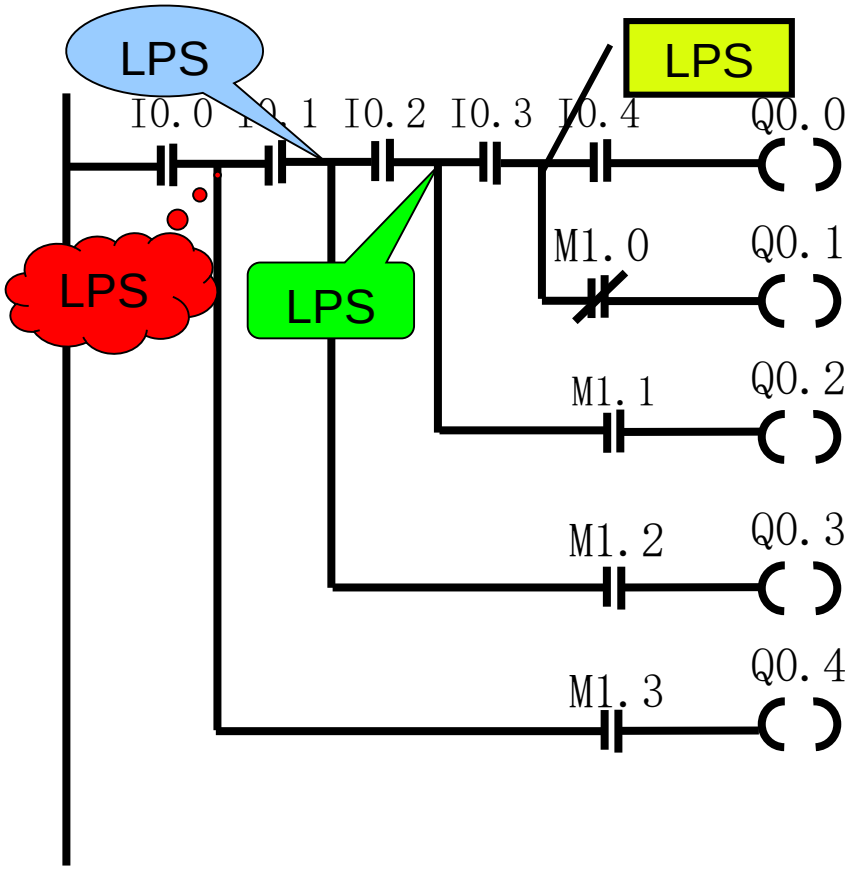
```
1   LD    I0.1        12  LPS   ──→ 第二层入栈
2   LPS               13  A     M1.2
3   A     I0.2        14  =     Q0.2
4   LPS               15  LPP   ──→ 第二层出栈
5   A     I0.3        16  A     I0.4
6   =     Q0.0        17  =     Q0.3
7   LPP               18  MEND
8   AN    M1.0
9   =     Q0.1
10  LPP   ──→ 第一层出栈
11  A     M1.1
```



例3：二层堆栈

指令表

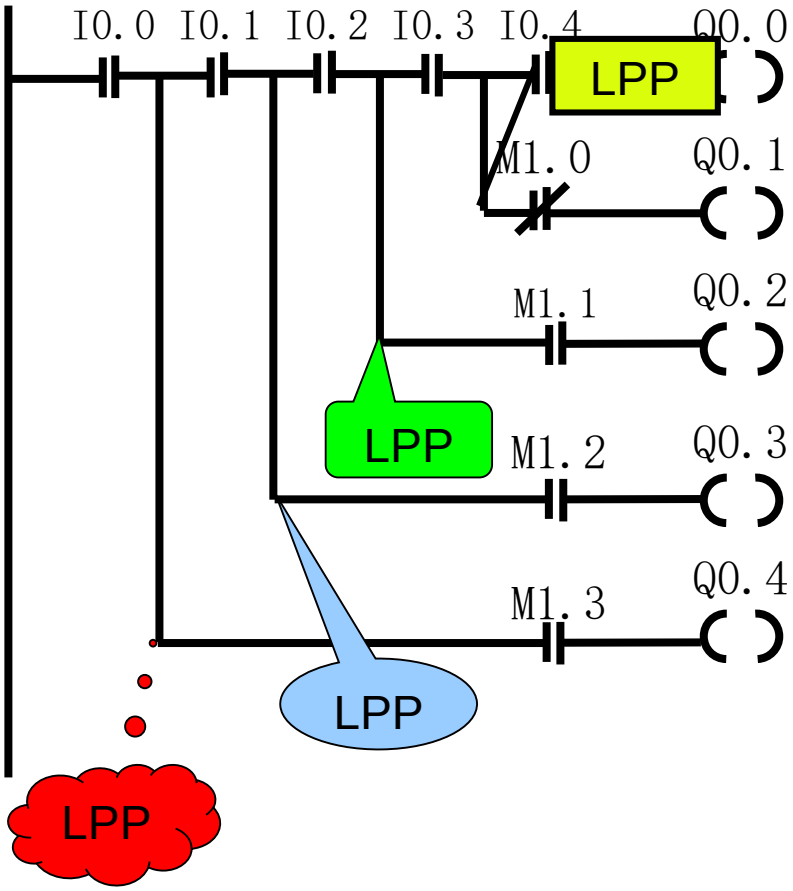
```
1   LD    I0.1
2   LPS
3   A     I0.2
4   LPS
5   A     I0.3
6   =     Q0.0
7   LPP
8   AN    M1.0
9   =     Q0.1
10  LPP
11  A     M1.1
12  LPS
13  A     M1.2
14  =     Q0.2
15  LPP
16  A     I0.4
17  =     Q0.3
18  MEND
```



例4：四层堆栈

指令表

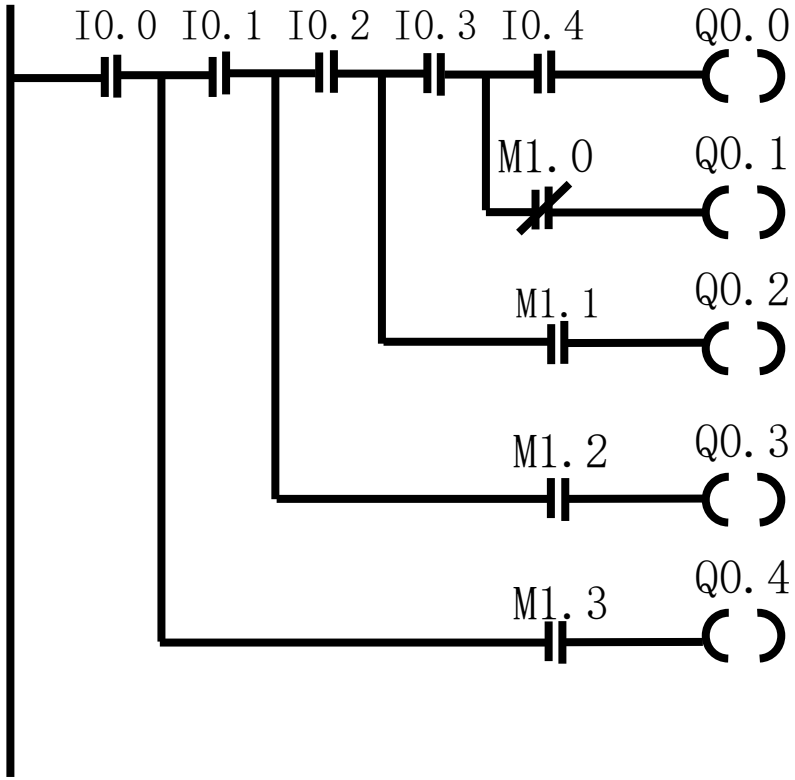
1	LD	I0.0
2	LPS	
3	A	I0.1
4	LPS	
5	A	I0.2
6	LPS	
7	A	I0.3
8	LPS	
9	A	I0.4
10	=	Q0.0



例4：四层堆栈

指令表

1	LD	I0.0
2	LPS	
3	A	I0.1
4	LPS	
5	A	I0.2
6	LPS	
7	A	I0.3
8	LPS	
9	A	I0.4
10	=	Q0.0
11	LPP	
12	AN	M1.0
13	=	Q0.1
14	LPP	
15	A	M1.1
16	=	Q0.2
17	LPP	
18	A	M1.2
19	=	Q0.3
20	LPP	
21	A	M1.3
22	=	Q0.4

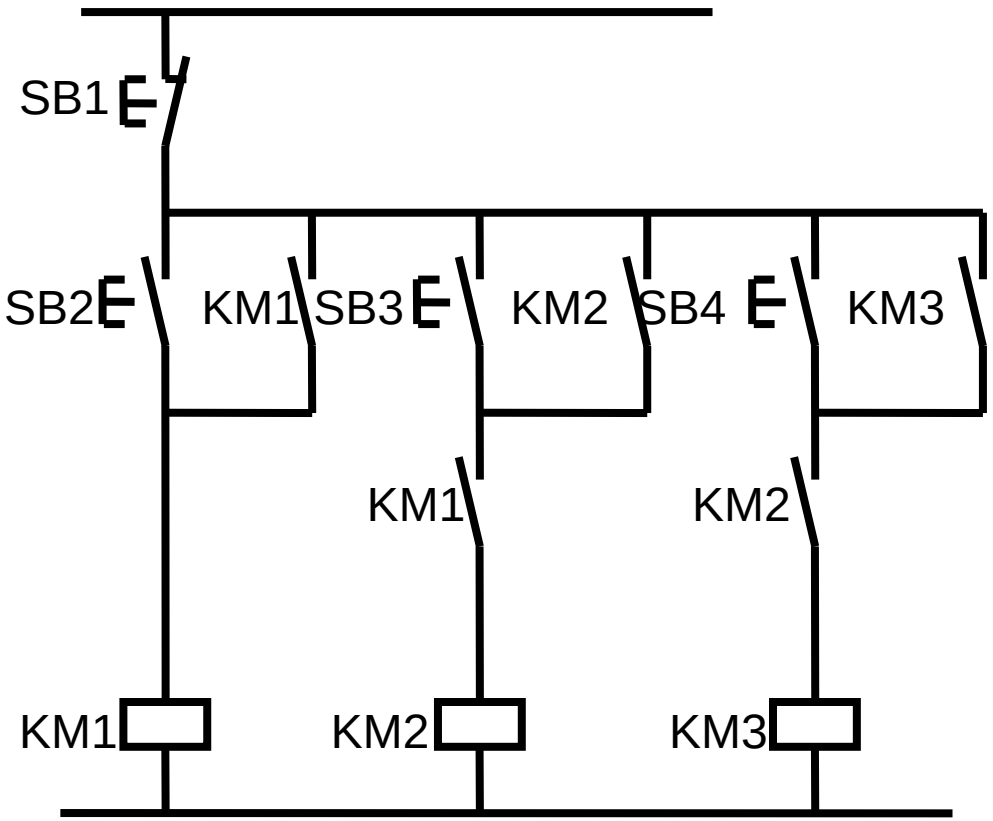


例4：四层堆栈

指令表

1	LD	I0.0	11	LPP	
2	LPS		12	AN	M1.0
3	A	I0.1	13	=	Q0.1
4	LPS		14	LPP	
5	A	I0.2	15	A	M1.1
6	LPS		16	=	Q0.2
7	A	I0.3	17	LPP	
8	LPS		18	A	M1.2
9	A	I0.4	19	=	Q0.3
10	=	Q0.0	20	LPP	
			21	A	M1.3
			22	=	Q0.4

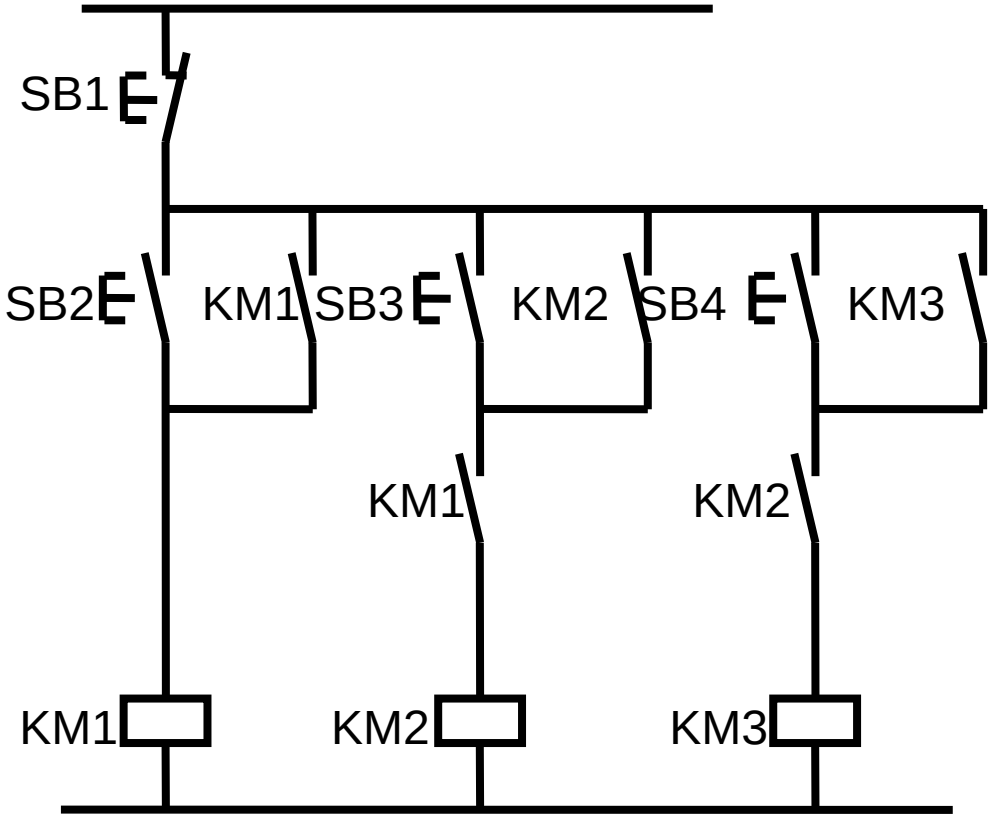
例5：练习



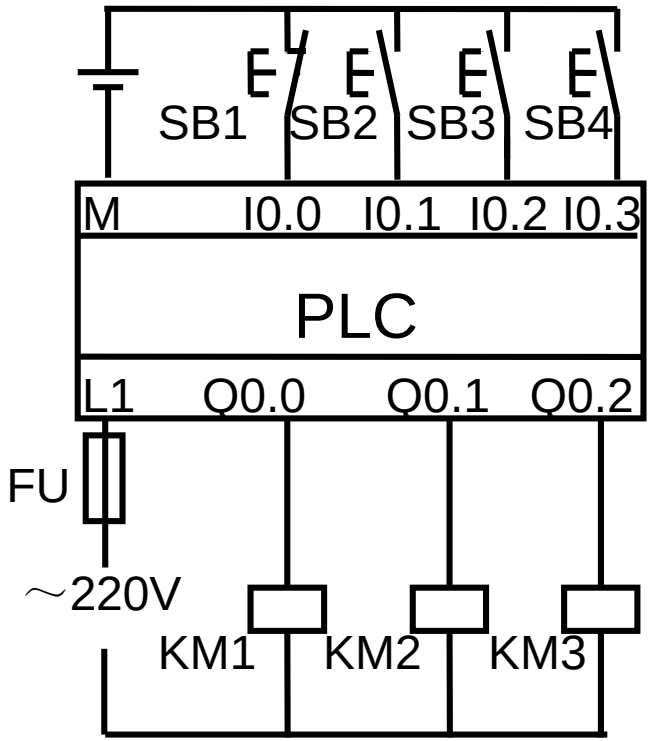
I/O分配表

SB1	停止按钮	I0.0
SB2	启动按钮1	I0.1
SB3	启动按钮2	I0.2
SB4	启动按钮3	I0.3
KM1	接触器	Q0.0
KM2	接触器	Q0.1
KM3	接触器	Q0.2

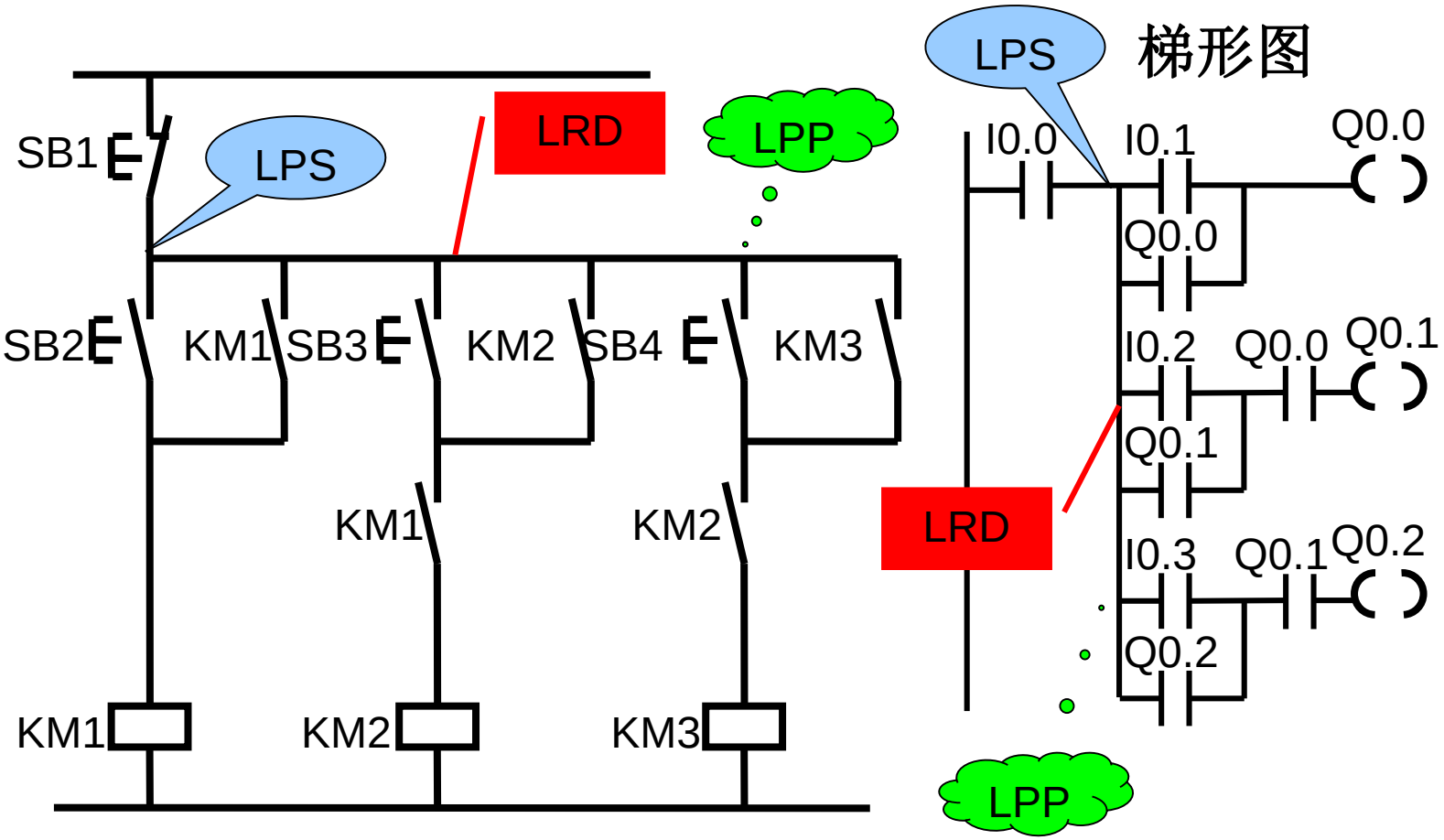
例5：练习


SB1
SB2
KM1
SB3
KM2
SB4
KM3
KM1
KM2
KM1
KM2
KM3

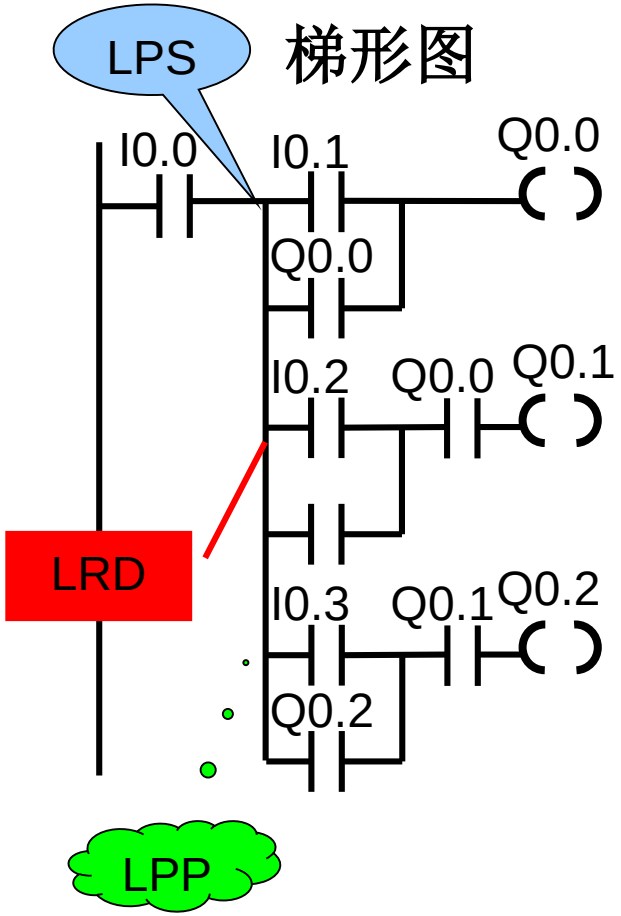
接线图


SB1
SB2
SB3
SB4
M
I0.0
I0.1
I0.2
I0.3
PLC
L1
Q0.0
Q0.1
Q0.2
FU
~220V
KM1
KM2
KM3

例5：练习


SB1
LPS
LRD
LPP
SB2
KM1
SB3
KM2
SB4
KM3
KM1
KM2
KM1
KM2
KM3
LPS
梯形图
I0.0
I0.1
Q0.0
Q0.0
I0.2
Q0.0
Q0.1
Q0.1
LRD
I0.3
Q0.1
Q0.2
Q0.2
LPP

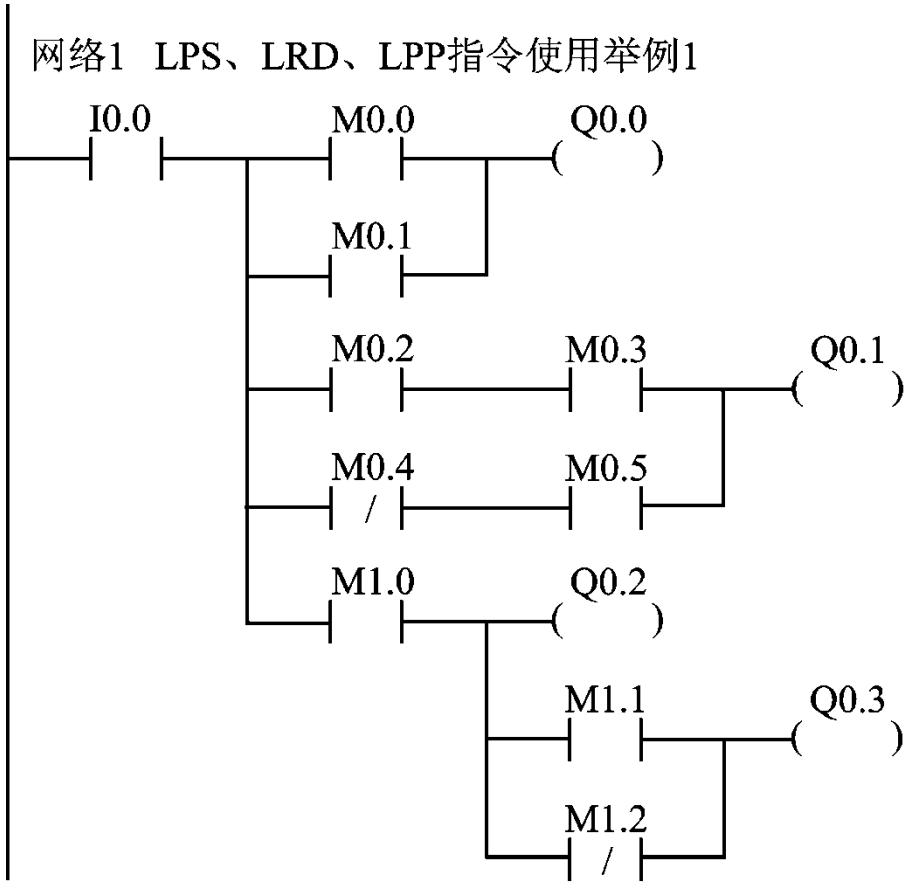
例5：练习



梯形图

```
1  LD   I0.0
2  LPS
3  LD   I0.1
4  O    Q0.0
5  ALD
6  =    Q0.0
7  LRD
8  LD   I0.2
9  O    Q0.1
10 ALD
11 A    Q0.0
12 =    Q0.1
13 LPP
14 LD   I0.2
15 O    Q0.2
16 ALD
17 A    Q0.1
18 =    Q0.2
19 MEND
```

可编程控制器原理及应用

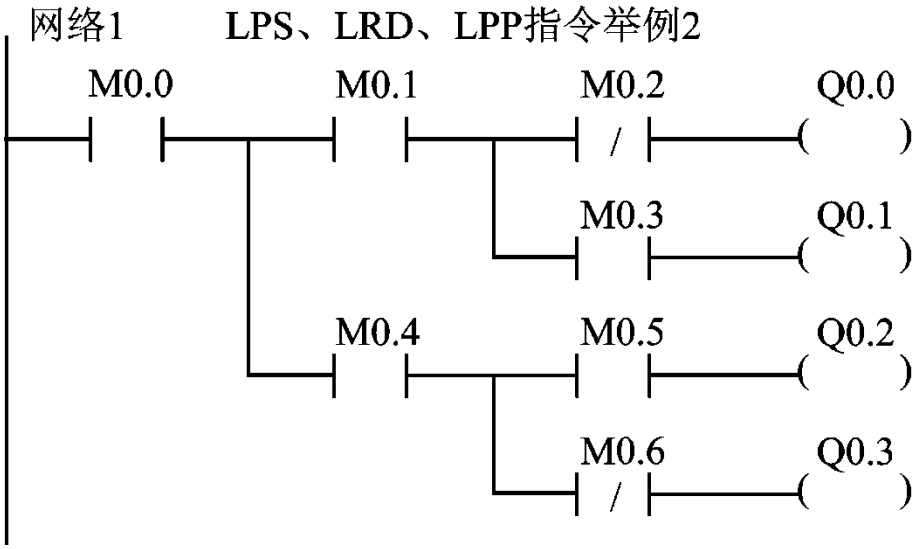


(a) 梯形图

```
LD      I0.0
LPS
LD      M0.0
O       M0.1
ALD
=       Q0.0
LRD
LD      M0.2
A       M0.3
LDN     M0.4
A       M0.5
OLD
ALD
=       Q0.1
LPP
A       M1.0
=       Q0.2
LD      M1.1
ON      M1.2
ALD
=       Q0.3
```

(b) 语句表

可编程控制器原理及应用

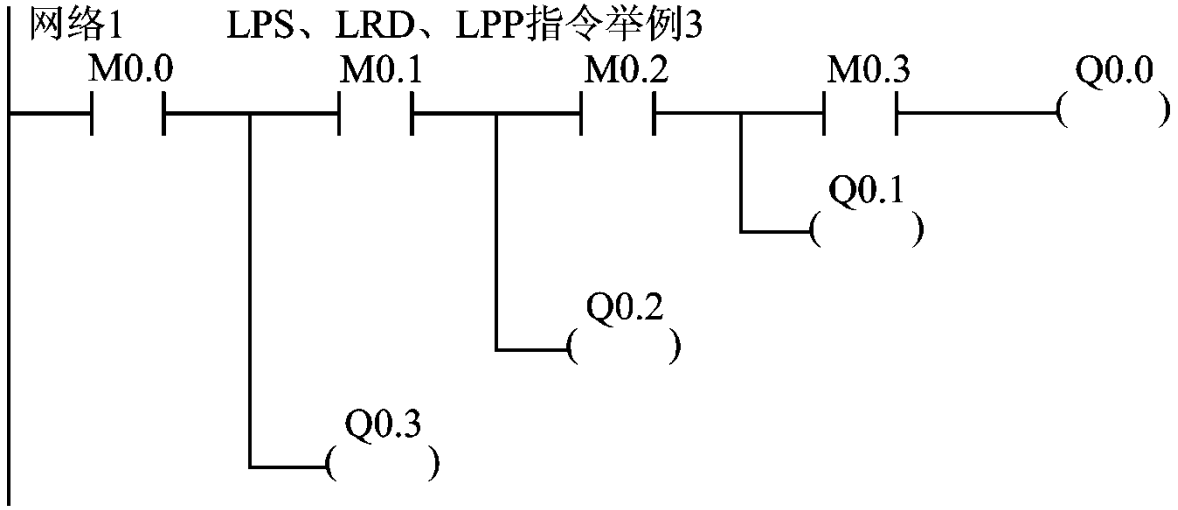


(a) 梯形图

```
LD      M0.0
LPS
A       M0.1
LPS
AN      M0.2
=       Q0.0
LPP
A       M0.3
=       Q0.1
LPP
A       M0.4
LPS
A       M0.5
=       Q0.2
LPP
AN      M0.6
=       Q0.3
```

(b) 语句表

可编程控制器原理及应用



(a) 梯形图

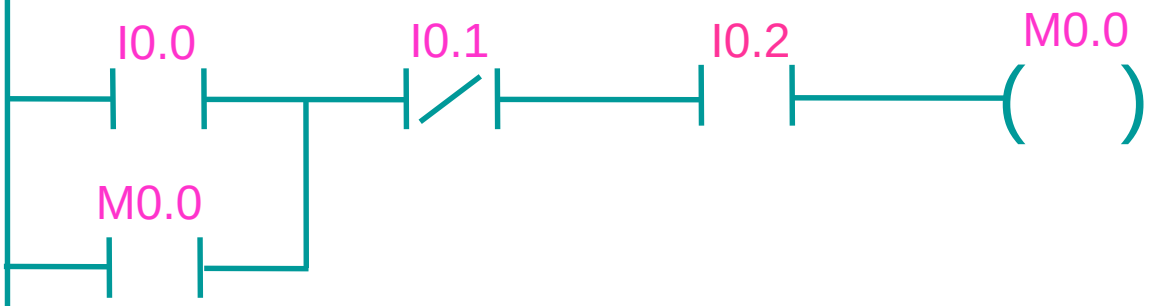
```
LD      M0.0
LPS
A       M0.1
LPS
A       M0.2
LPS
A       M0.3
=       Q0.0
LPP
=       Q0.1
LPP
=       Q0.2
LPP
=       Q0.3
```

(b) 语句表

可编程控制器原理及应用

输入设备		PLC输入继电器	输出设备		PLC继电器输出
代号	功能		代号	功能	
SB1	启动按钮	I0.0	KM1	电源接触器	Q0.1
SB2	停止按钮	I0.1	KM2	星接接触器	Q0.2
KH	热继电器	I0.2	KM3	角接接触器	Q0.3

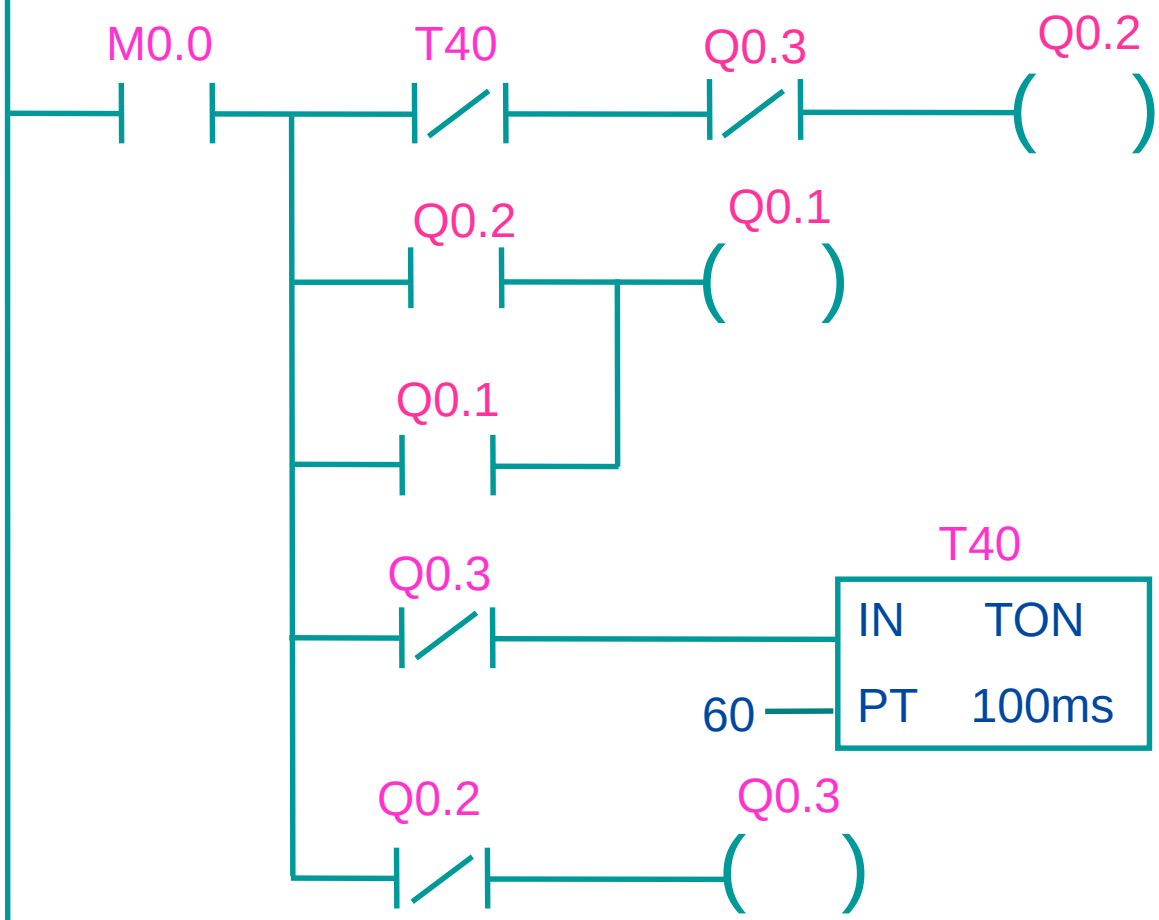
网络1 电动机Y-△启动控制程序



网络1 电动机Y-△启动控制程序

```
LD    I0.0
O     M0.0
AN    I0.1
A     I0.2
=     M0.0
```

网络2



网络2

```
LD    M0.0
LPS
AN    T40
AN    Q0.3
=     Q0.2
LRD
LD    Q0.2
O     Q0.1
ALD
=     Q0.1
LPP
LPS
AN    Q0.3
TON   T40,60
LPP
AN    Q0.2
=     Q0.3
```

标准常开触点指令：

LD　A　　O

标准常闭触点指令：

LDN　AN　ON

立即常开触点指令：

LDI　AI　OI

立即常闭触点指令：

LDNI　ANI　ONI

立即输出指令：

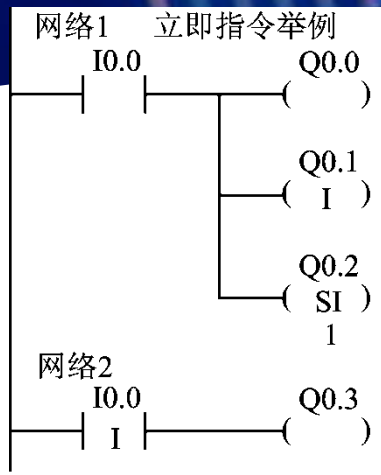
=I　bit

立即置位：

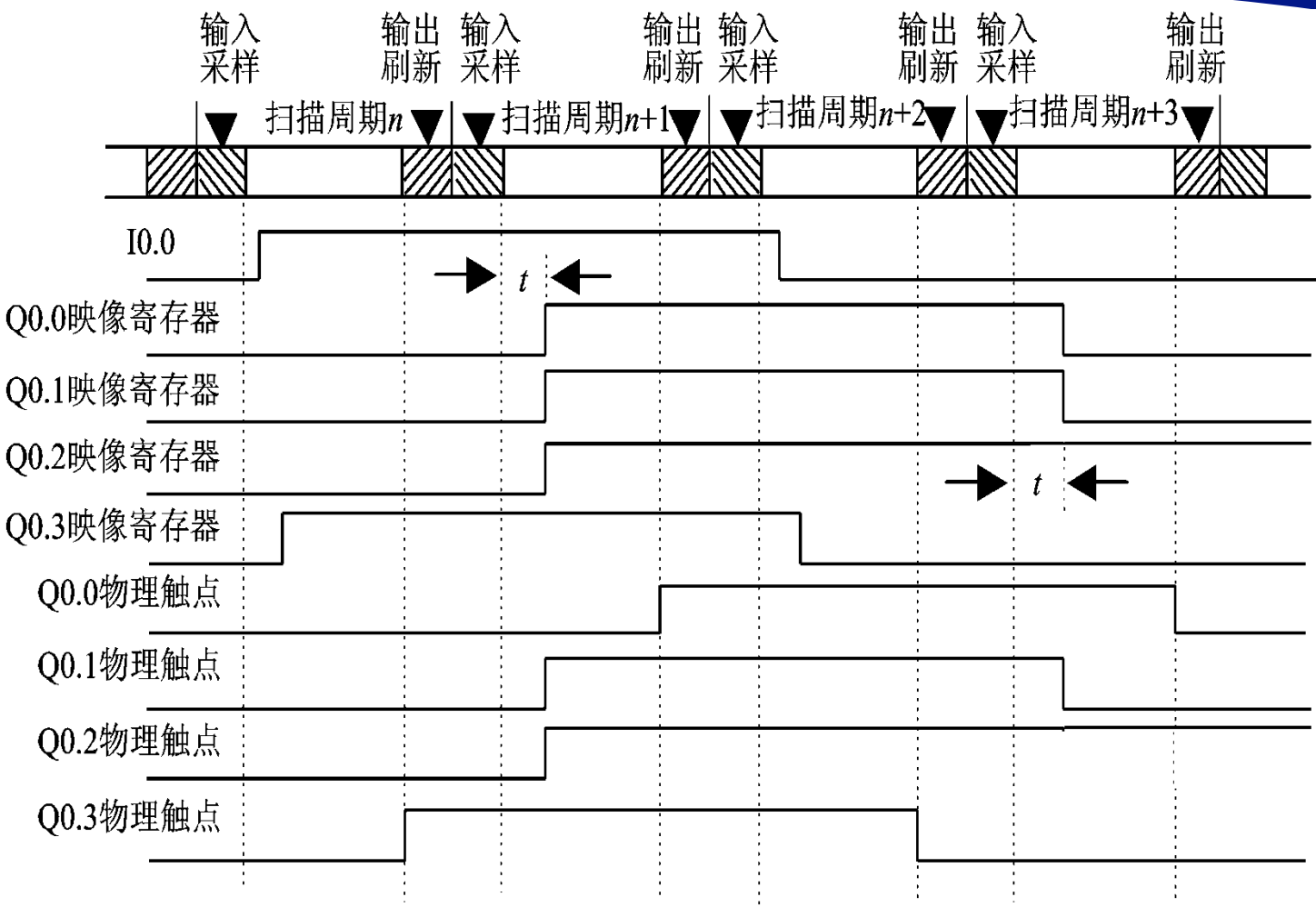
SI　bit，N

立即复位：

RI　bit，N


网络1 立即指令举例
I0.0
Q0.0
Q0.1
I
Q0.2
SI
1
网络2
I0.0
I
Q0.3

(a) 梯形图


输入采样
输出刷新
输入采样
输出刷新
输入采样
输出刷新
输入采样
输出刷新
扫描周期n
扫描周期n+1
扫描周期n+2
扫描周期n+3
I0.0
t
Q0.0映像寄存器
Q0.1映像寄存器
Q0.2映像寄存器
t
Q0.3映像寄存器
Q0.0物理触点
Q0.1物理触点
Q0.2物理触点
Q0.3物理触点

(c) 时序图