目　　录

第一章　FX2N PLC 编程简介

1.1　FX2N PLC 简介

1.1.1　FX2N PLC 的提出

基于以下观点，提出 FX2N PLC 的概念：

①、软件和硬件独立设计。

在规定好硬件和软件接口的前提下，各自独立设计，以提高开发效率。

②、简化硬件设计。

只需进行外形设计和电气接口设计，功能设计由软件设计取代。

③、简化软件设计。

依托功能强大的软件平台，只需设计个体产品与平台间的软件接口。

④、产品应用可二次编程。

根据工艺要求，用标准梯形图语言进行二次编程。

1.1.2　FX2N PLC 的特点

①、什么是 FX2N PLC?

将 PLC 语言（梯形图语言）嵌入到专用芯片中，获取了梯形图编程平台所提供的各种强大的应用功能。我们称用于 PLC 专用芯片产品开发，自身具有强大功能的梯形图语言编程软件为 FX2N PLC。FX2N PLC 能广泛应用于各种工业控制产品中。

②、FX2N PLC 产品有哪些特点？

利用 FX2N PLC 软件开发出的应用产品，我们称之为 FX2N PLC 产品。FX2N PLC 产品具有以下特点：

- 用梯形图语言编写应用程序。
- 能与多家人机界面连接，如台达、EView 等。
- 支持 CANBUS 网络结构。
- 与其它厂家 PLC 并联运行。

1.1.3　FX2N PLC 产品举例

①、可编程控制器 FX2N-40MR

本产品有开关量输入 24 点、开关量输出 16 点，除具有可编程逻辑控制功能之外。每

台产品均支持人机界面。

②、空压机控制器

具有用户要求的外观和接口，用户可根据自己的意图，用梯形图编写不同的控制程序，便于工艺保密和系列产品的标准化。每台控制器均可支持人机界面。

③、供水控制器

预留较多的富余接口，可适应各种复杂的供水要求，是供水行业的通用型控制器。应用人员可用梯形图编写控制程序，满足用户的不同需要。该产品支持人机界面。

1.1.4 关于本手册

编写本手册的目的是帮助 FX2N PLC 产品的用户，正确使用梯形图语言编程，充分发挥 FX2N PLC 所提供的强大功能。

“编程简介”简要介绍梯形图的指令集和资源集，使有一定梯形图语言编程基础的用户参照指令集和资源集后可立即编写通用控制程序。

第二章到第五章，详细介绍了指令集和资源集，通过这些章节的学习，使初学者也能用梯形图编写各种应用程序。

第六章介绍 PID 专用控制算法，属专家成果应用。

1.2　　编程简介

1.2.1　指令集简介

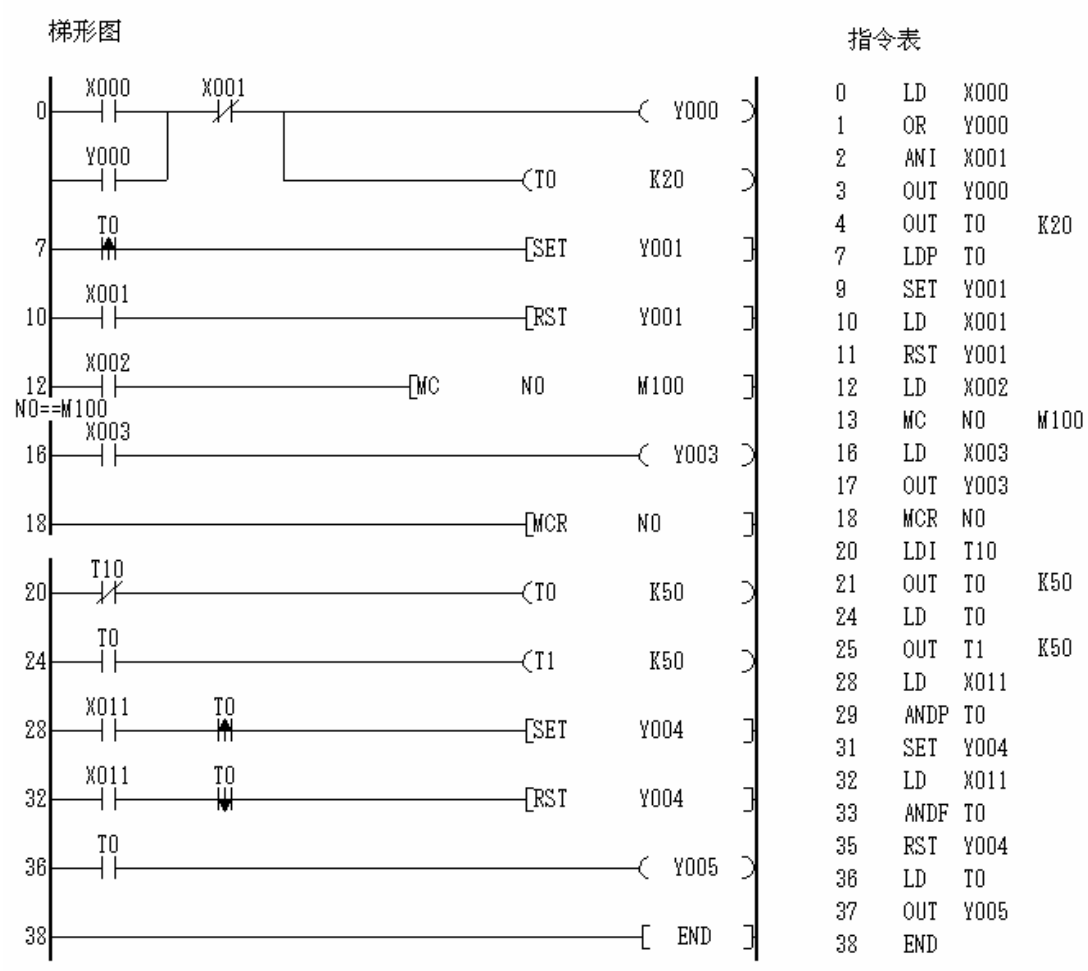
①、基本逻辑指令：

- 助记符及名称：

  LD：读取常开点。
  LDI：读取常闭点。
  AND：串入常开点。
  ANI：串入常闭点。
  OR　并入常开点。
  ORI：并入常闭点。
  ANB：电路块串联。
  ORB：电路块并联。
  OUT：线圈输出。
  SET：线圈输出保持。
  RST：清除线圈输出。
  PLS：上升沿输出脉冲。
  PLF：下降沿输出脉冲。
  LDP　读取上升沿。
  LDF　读取下降沿。
  ANDP：上升沿接通，串联连接。
  ANDF：下降沿接通，串联连接。
  ORP：上升沿接通，并联连接。
  ORF：下降沿接通，并联连接。
  INV：运算触点取反。
  MPS：压栈。
  MRD：读栈。
  MPP：出栈。
  MC：主控。
  MCR：主控结束。
  NOP：空操作。
  END：程序结束。

- 梯形图与指令表：

  梯形图是电气控制的专业语言，方便编程人员编程。
  专用芯片是按指令表执行控制。
  梯形图与指令表二者自动相互转换。下例是二者相互转换示意图。

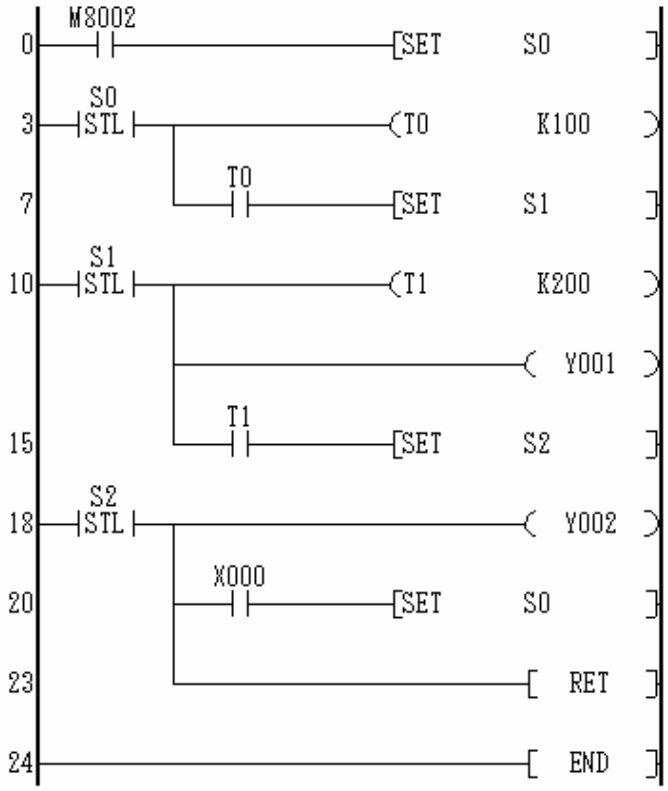


想对基本逻辑指令进一步了解，请参看《第二章 基本逻辑指令说明及应用》。

②、步进顺控指令：

- 助记符及名称：
  STL：步进梯形图开始。仅对状态继电器 S。<u>步序间状态转移必须使用 **SET S**</u>，不能用 OUT S。
  RET：步进梯形图结束。
- 梯形图与指令表：
  梯形图是电气控制的专业语言，方便编程人员编程。
  专用芯片是按指令表执行控制。
  梯形图与指令表二者自动相互转换。下例是二者相互转换示意图。

步进梯形图



指令表

```
0    LD    M8002
1    SET   S0
3    STL   S0
4    OUT   T0     K100
7    LD    T0
8    SET   S1
10   STL   S1
11   OUT   T1     K200
14   OUT   Y001
15   LD    T1
16   SET   S2
18   STL   S2
19   OUT   Y002
20   LD    X000
21   SET   S0
23   RET
24   END
25   NOP
26   NOP
```

想对步进顺控指令进一步了解，请参看《第三章 步进顺控指令说明及应用》。

③、基本功能指令：

- 助记符及名称：

  CJ： 条件跳转。

  CALL：子程序调用。

  SRET：子程序返回。

  FEND：主程序结束。

  FOR：循环开始。

  NEXT：循环结束。

  ****

  CMP：比较。

  ZCP：区域比较。

  MOV：传送。

  CML：取反传送。

  BCD：BIN 向 BCD 转换。

  BIN：BCD 向 BIN 转换。

****

ADD：加法。
SUB：减法。
MUL：乘法。
DIV：除法。
INC：自加 1 运算。
DEC：自减 1 运算。
WAND：字与运算（按位）。
WOR：字或运算（按位）。
WXOR：字异或运算（按位）。
NEG：取补运算。
SQR：开方运算。

****

ROR：循环右移。
ROL：循环左移。
RCR：带进位循环右移。
RCL：带进位循环左移。

****

DECMP：二进制浮点数比较。
DEZCP：二进制浮点数区域比较。
DEBCD：二进制浮点数向十进制浮点数转换。
DEBIN：十进制浮点数向二进制浮点数转换。
DEADD：二进制浮点数加法。
DESUB：二进制浮点数减法。
DEMUL：二进制浮点数乘法。
DEDIV：二进制浮点数除法。
DESQR：二进制浮点数开方。

****

INT：二进制浮点数取整。
FLT：整数转换为二进制浮点数。

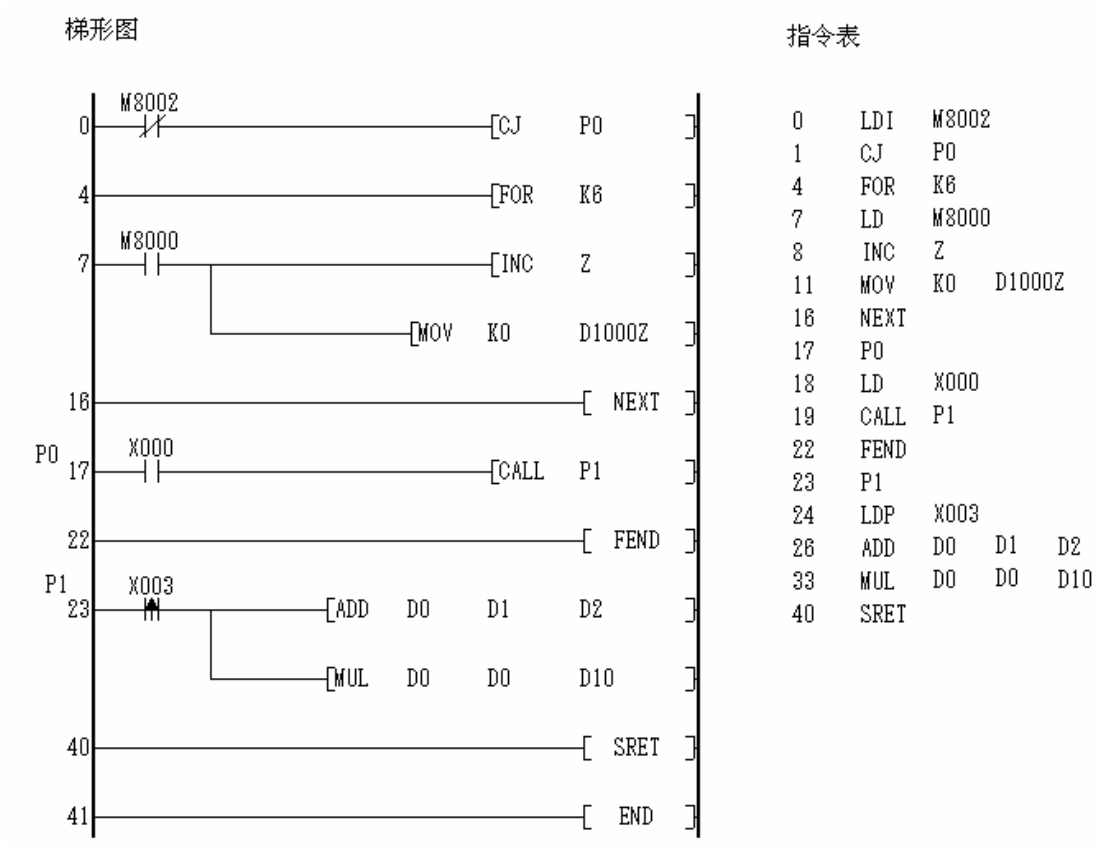
****

LD=：读取“等于比较节点”。
LD>：读取“大于比较节点”。
LD<：读取“小于比较节点”。
LD<>：读取“不等于比较节点”。
LD<=：读取“小于等于比较节点”。
LD>=：读取“大于等于比较节点”。
AND=：串联“等于比较节点”。
AND>：串联“大于比较节点”。
AND<：串联“小于比较节点”。
AND<>：串联“不等于比较节点”。

AND<=：串联“小于等于比较节点”。
AND>=：串联“大于等于比较节点”。
OR=：并联“等于比较节点”。
OR>：并联“大于比较节点”。
OR<：并联“小于比较节点”。
OR<>：并联“不等于比较节点”。
OR<=：并联“小于等于比较节点”。
OR>=：并联“大于等于比较节点”。

- 梯形图与指令表：

梯形图是电气控制的专业语言，方便编程人员编程。
专用芯片是按指令表执行控制。
梯形图与指令表二者自动相互转换。下例是二者相互转换示意图。

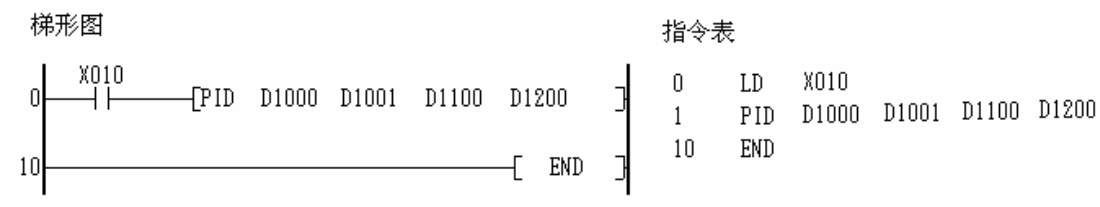


想对基本功能指令进一步理解，请参看《第四章 基本功能指令说明及应用》。

④、专家功能指令：

- 助记符及名称：
  PID：PID 控制算法。
- 梯形图与指令表：

梯形图是电气控制的专业语言，方便编程人员编程。
专用芯片是按指令表执行控制。
梯形图与指令表二者自动相互转换。下例是二者相互转换示意图。



想对专家功能指令进一步了解，请参看《第六章 专家功能指令说明及应用》。

1.2.2 资源集简介

①、输入继电器 X：
扩展数量：128 点。
标号范围：X000-----X177；标号为 8 进制。
实际产品的数量和范围：由 FX2N PLC 产品确定。
如 K-40MR，范围：X000-X027，数量：24 点。

②、输出继电器 Y：
扩展数量：128 点。
标号范围：Y000-----Y177；标号为 8 进制。
实际产品的数量和范围：由 FX2N PLC 产品确定。
如 K-40MR，范围：Y000-Y017，数量：16 点。

③、辅助继电器 M：
数量：1536 点
标号范围：M0---M1535；标号为十进制。
一般用：M0---M1023，计 1024 点。
停电保持用：M1024---M1535，计 512 点。

④、状态继电器 S：
数量：1000 点
标号范围：S0---S999；标号为十进制。
一般用：S0---M499，计 500 点。
停电保持用：M500---M999，计 500 点。

⑤、时间继电器 T：
数量：256 点
标号范围：T0---T255；标号为十进制。

一般用：T0---T199，100 ms 型，计 200 点；
T200---T245，10 ms 型，计 46 点；
累积用：T246---T249，1 ms 型，计 4 点；
T250---T255，100 ms 型，计 6 点；
累积用的时间继电器在停电时，计时数据保持，必须用 RST 清零。

⑥、计数器 C：
数量：256 点
标号范围：C0---C199；C200---C255；标号为十进制。
一般用：C0---C99，16 bit，计 100 点。
停电保持用　C100---C199，16 bit，计 100 点。
C200---C255，32bit 可逆计数器，计数方向由 M8200-M8255 确定，ON 时减计数。

⑦、数据寄存器 D：
数量：6000 点
标号范围：D0---D5999；标号为十进制。
一般用：D0---D199，计 200 点。
停电保持用：D200---D5999，计 5800 点。

⑧、变址寄存器 V：
数量：8 点。
标号范围：V0---V7；标号为十进制，无停电保持功能。

⑨、变址寄存器 Z：
数量：8 点。
标号范围：Z0---Z7；标号为十进制，无停电保持功能。

⑩、程序位置指针 P：
数量：128 个
标号范围：P0---P127；标号为十进制。

⑾、十进制常数标记 K、H：
标号 K 后的常数为十进制常数。
标号 H 后的常数为十六进制常数。如 H10=K16。

⑿、特殊软元件：
M8000：程序运行时 ON；
M8002：程序开运行时第一个扫描周期时 ON；
M8020：零标志；
M8021：　借位标志；

M8022：进位标志；
M8200---M8255：32 bit 可逆计数器方向指定。

想对资源更进一步了解，请参看《第五章 资源详细说明及应用》。

1.2.3 编程及应用简介

①、编程软件

- 梯形图编程软件 SLJDWin：
  支持梯形图编程、下载、监控，可对 FX2N PLC 产品设置加密口令。
- 网络设置软件上位机软件
  支持网络构建、下载，经上位机软件设置的主节点与从节点能自动交换网络数据。
  网络构建支持第三方设备。

②、编程设备

个人计算机：
SLJDWin 运行于 Windows 操作系统。操作系统可以是：
Windows 95，Windows 98，Windows 2000，Windows XP。

③、编程及应用流程说明

- 产品编程。

  一般有以下步骤：
  了解 FX2N PLC 产品的硬件接口（X，Y，D）和功能要求；
  编写梯形图程序；
  程序检查及下载；
  程序监控及调试；
  批量应用于嵌入式产品；

第二章　基本逻辑指令说明及应用

2.1　基本逻辑指令一览表

助记符、名称	功能	可用软元件	程序步
LD 取	常开触点逻辑运算开始	X,Y,M,S,T,C	1
LDI 取反	常闭触点逻辑运算开始	X,Y,M,S,T,C	1
LDP 取脉冲上升沿	上升沿检出运算开始	X,Y,M,S,T,C	2
LDF 取脉冲下降沿	下降沿检出运算开始	X,Y,M,S,T,C	2
AND 与	常开触点串联连接	X,Y,M,S,T,C	1
ANI 与非	常闭触点串联连接	X,Y,M,S,T,C	1
ANDP 与脉冲上升沿	上升沿检出串联连接	X,Y,M,S,T,C	2
ANDF 与脉冲下降沿	下降沿检出串联连接	X,Y,M,S,T,C	2
OR 或	常开触点并联连接	X,Y,M,S,T,C	1
ORI 或非	常闭触点并联连接	X,Y,M,S,T,C	1
ORP 或脉冲上升沿	上升沿检出并联连接	X,Y,M,S,T,C	2
ORF 或脉冲下降沿	下降沿检出并联连接	X,Y,M,S,T,C	2
ANB 块与	并联回路块的串联连接		1
ORB 块或	串联回路块的并联连接		1
OUT 输出	线圈驱动	Y,M,S,T,C	注 1
SET 置位	动作保持	Y,M,S	注 2
RST 复位	清除动作保持，寄存器清零	Y,M,S,T,C,D,V,Z	
PLS 上升沿脉冲	上升沿输出	Y,M（特殊 M 除外）	1
PLF 下降沿脉冲	下降沿输出	Y,M（特殊 M 除外）	1
MC 主控	公共串联点的连接线圈指令	Y,M（特殊 M 除外）	3
MCR 主控复位	公共串联点的消除指令		2
MPS 压栈	运算存储		1
MRD 读栈	存储读出		1
MPP 出栈	存储读出与复位		1
INV 取反	运算结果的反转		1
NOP 空操作	无动作		1
END 结束	输入输出及返回到开始		1

- 软元件为 Y 和一般 M 的程序步为 1，S 和特殊辅助继电器 M 的程序步为 2，定时器 T 的程序步为 3，计数器 C 的程序步为 3－5。
- 软元件为 Y 和一般 M 的程序步为 1，S 和特殊辅助继电器 M、定时器 T、计数器 C 的程序步为 2，数据寄存器 D 以及变址寄存器 V 和 Z 的程序步为 3。

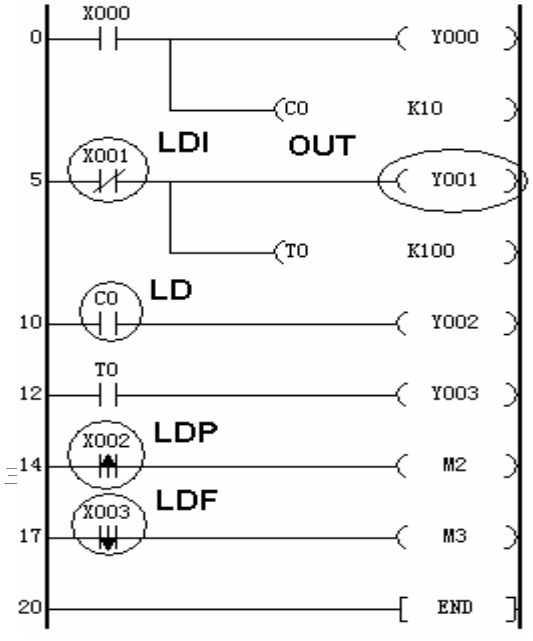
2.2　[LD],[LDI],[LDP],[LDF],[OUT] 指令

2.2.1 指令解说

助记符、名称	功能	可用软元件	程序步
LD 取	常开触点逻辑运算开始	X,Y,M,S,T,C	1
LDI 取反	常闭触点逻辑运算开始	X,Y,M,S,T,C	1
LDP 取脉冲上升沿	上升沿检出运算开始	X,Y,M,S,T,C	2
LDF 取脉冲下降沿	下降沿检出运算开始	X,Y,M,S,T,C	2
OUT 输出	线圈驱动	Y,M,S,T,C	见说明

- LD,LDI,LDP,LDF 指令将触点连接到母线上。多个分支用 ANB,ORB 时也使用。
- LDP 指令在上升沿（软元件由 OFF 到 ON 变化时）接通一个周期；LDF 指令在下降沿（软元件由 ON 到 OFF 变化时）接通一个周期。
- LD,LDI,LDP,LDF 指令的重复使用次数在 8 次以下。即与后面的 ANB,ORB 指令使用时串并连使用的最多次数为 8 个。
- 软元件为 Y 和一般 M 的程序步为 1，S 和特殊辅助继电器 M 的程序步为 2，定时器 T 的程序步为 3，计数器 C 的程序步为 3－5。
- OUT 指令各种软元件的线圈驱动，但对输入继电器不能使用。并列的 OUT 可多次连续使用。
- OUT 指令驱动计数器时，当前面的线圈从 ON 变成 OFF，或者是从 OFF 变成 ON 时，计数器才加一。

2.2.2 编程示例




```
0   LD    X000
1   OUT   Y000
2   OUT   C0    K10
5   LDI   X001
6   OUT   Y001
7   OUT   T0    K100
10  LD    C0
11  OUT   Y002
12  LD    T0
```

```
      13   OUT     Y003
14    LDP     X002
16    OUT     M2
17    LDF     X003
19    OUT     M3
20    END
```

- 用 LD,LDI,LDP,LDF 指令与母线连接。输出使用 OUT 指令驱动线圈。
- 使用 OUT 指令驱动定时器的计时线圈或者计数器的计数线圈时，必须设定定时和计数的时间和计数的值，可以是常数 K，或者由数据寄存器间接指定数值。
- 每个程序结束必须要有 END 指令，关于 END 指令详见后面的 END 指令介绍。

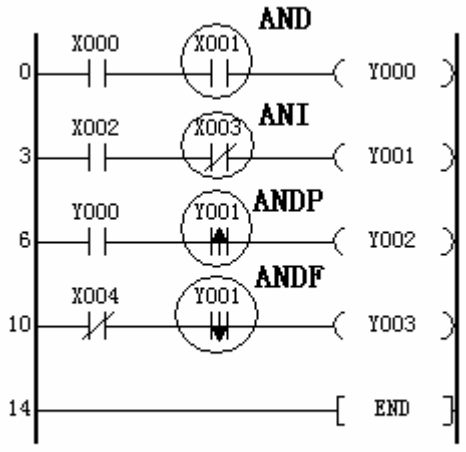
2.3　[AND],[ANI],[ANDP],[ANDF] 指令

2.3.1 指令解说

助记符、名称	功能	可用软元件	程序步
AND 与	常开触点串联连接	X,Y,M,S,T,C	1
ANI 与非	常闭触点串联连接	X,Y,M,S,T,C	1
ANDP 与脉冲上升沿	上升沿检出串联连接	X,Y,M,S,T,C	2
ANDF 与脉冲下降沿	下降沿检出串联连接	X,Y,M,S,T,C	2

- AND,ANI,ANDP,ANDF 指令只能串接一个触点，两个以上的并联回路串联时使用后面的 ANB 指令。串联次数不受限制。
- ANDP,ANDF 指令在上升沿（即软元件由 ON 到 OFF 变化时）和下降沿即（软元件由 OFF 到 ON 变化时）接通一个周期。

2.3.2 编程示例




```
0     LD      X000
1     AND     X001
2     OUT     Y000
3     LD      X002
4     ANI     X003
5     OUT     Y001
6     LD      Y000
7     ANDP    Y001
9     OUT     Y002
10    LDI     X004
```

```
11   ANDF    Y001
13   OUT     Y003
14   END
```

- 实例中 X001，X003，Y001 作为串联触点与前面的触点相连。

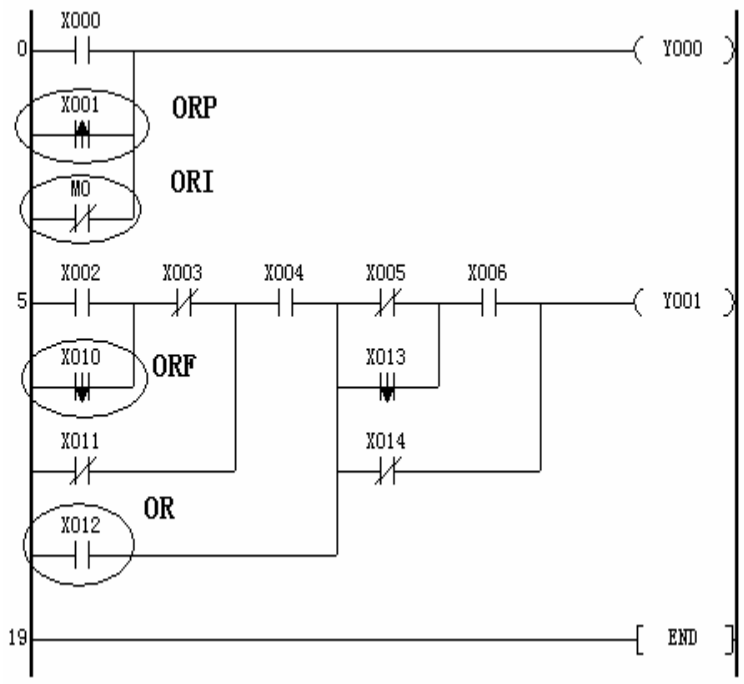
2.4　[OR],[ORI],[ORP],[ORF] 指令

2.4.1　指令解说

助记符、名称	功能	可用软元件	程序步
OR 或	常开触点并联连接	X,Y,M,S,T,C	1
ORI 或非	常闭触点并联连接	X,Y,M,S,T,C	1
ORP 或脉冲上升沿	上升沿检出并联连接	X,Y,M,S,T,C	2
ORF 或脉冲下降沿	下降沿检出并联连接	X,Y,M,S,T,C	2

- OR,ORI,ORP,ORF 指令只能并接一个触点，两个以上的串联回路并联时使用后面的 ORB 指令。
- ORP,ORF 指令在上升沿（即软元件由 OFF 到 ON 变化时）和下降沿（即软元件由 ON 到 OFF 变化时）接通一个周期。
- OR,ORI,ORP,ORF 指令和前面的 LD,LDI,LDP,LDF 指令一起使用，并联次数不受限制。

2.4.2　编程示例



```
0   LD      X000
1   ORP     X001
3   ORI     M0
4   OUT     Y000
5   LD      X002
6   ORF     X010
8   ANI     X003
9   ORI     X011
10  AND     X004
11  OR      X012
12  LDI     X005
13  ORF     X013
15  AND     X006
16  ORI     X014
17  ANB
18  OUT     Y001
19  END
```

- 使用 OR,ORI,ORP,ORF 与前面的 LD,LDI,LDP,LDF 并联连接，在程序步 12 到 16 中，由于是两个并联回路块的串联，所以使用 ANB 指令，关于 ANB 指令详见后面的说明。

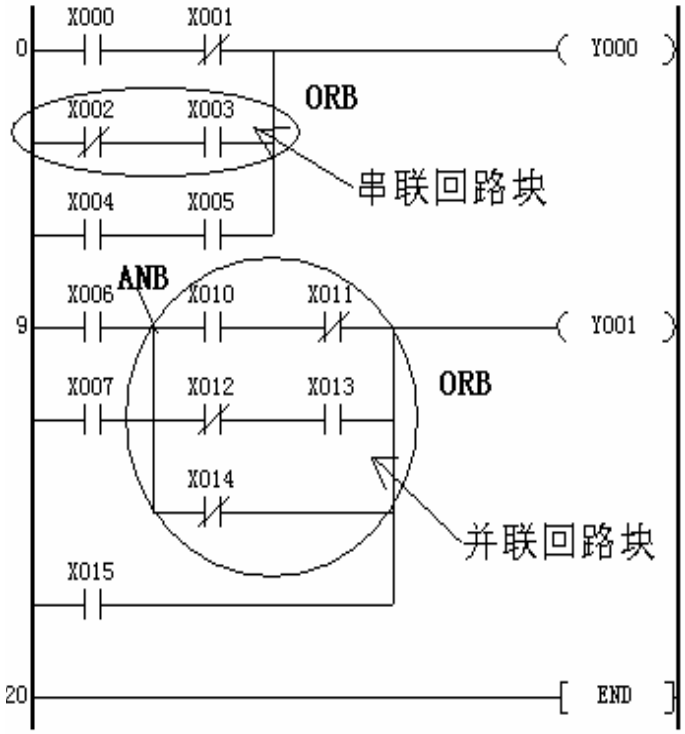
2.5　[ANB],[ORB] 指令

2.5.1　指令解说

助记符、名称	功能	可用软元件	程序步
ANB 块与	并联回路块的串联连接		1
ORB 块或	串联回路块的并联连接		1

- 当多分支回路与前面的回路串联连接时，使用 ANB 指令。分支以 LD,LDI,LDP,LDF 指令作为起点，使用 ANB 指令与前面以 LD,LDI,LDP,LDF 指令作为起点的分支串联连接。
- 当 2 个以上的触点串接的串联回路块并联连接时，每个分支使用 LD,LDI 指令开始，ORB 指令结束。
- ANB,ORB 指令都是不带软元件的指令。
- ANB,ORB 使用的并串联回路的个数不受限制，但是当成批使用时，必须考虑 LD,LDI 的使用次数在 8 次以下。

2.5.2　编程示例



```
0   LD      X000
1   ANI     X001
2   LDI     X002
3   AND     X003
4   ORB
5   LD      X004
6   AND     X005
7   ORB
8   OUT     Y000
9   LD      X006
10  OR      X007
11  LD      X010
12  ANI     X011
13  LDI     X012
14  AND     X013
15  ORB
16  ORI     X014
17  ANB
18  OR      X015
19  OUT     Y001
20  END
```

- 在每个分支的最后使用 ORB 指令，不要在所有的分支后面使用 ORB 指令，如程序步 4 和 7 所示。
- ORB 和 ANB 指令只是对块的连接，如果不是块就不能使用，如程序步 16 和 18 不是块就不能使用。如图所示，串联回路块和并联回路块的示例。

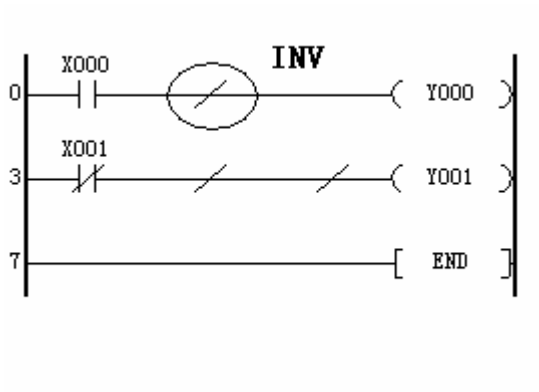
2.6　[INV] 指令

2.6.1　指令解说

助记符、名称	功能	可用软元件	程序步
INV 取反	运算结果的反转		1

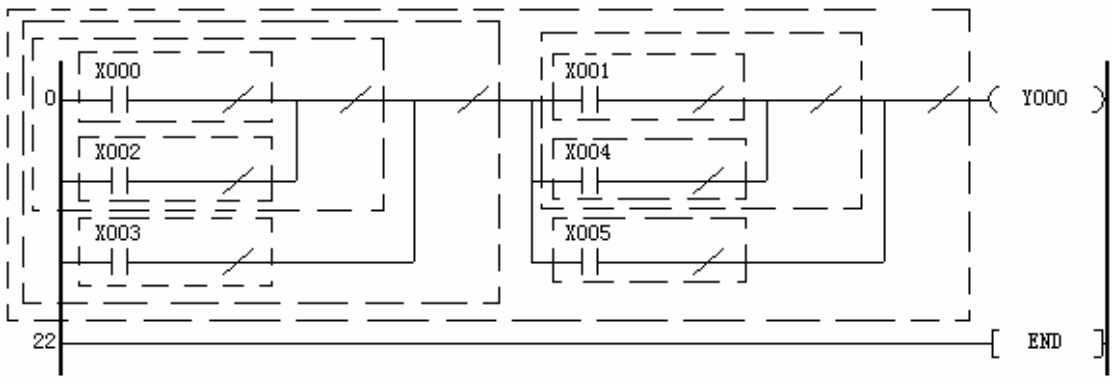
- INV 指令是将 INV 指令之前，LD,LDI,LDP,LDF 指令之后的运算结果取反的指令，没有软元件。

2.6.2　编程示例




```
0    LD     X000
1    INV
2    OUT    Y000
3    LDI    X001
4    INV
5    INV
6    OUT    Y001
7    END
```

INV 指令的动作范围如图：



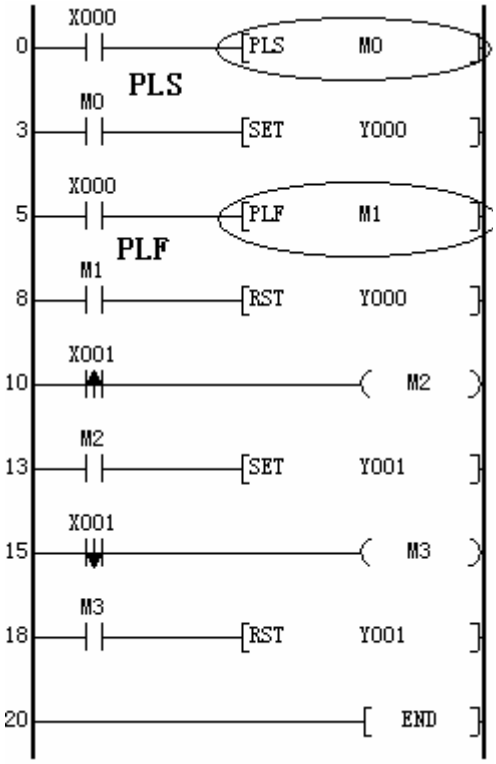
2.7　[PLS],[PLF] 指令

2.7.1 指令解说

助记符、名称	功能	可用软元件	程序步
PLS 上升沿脉冲	上升沿输出	Y,M（特殊 M 除外）	1
PLF 下降沿脉冲	下降沿输出	Y,M（特殊 M 除外）	1

- 使用 PLS 指令时，只在线圈由 OFF 变成 ON 的一个扫描周期内，驱动软元件。
- 使用 PLF 指令时，只在线圈由 ON 变成 OFF 的一个扫描周期内，驱动软元件。
- 对具有停电保持功能的软元件，它只在第一次运行时产生脉冲动作。

2.7.2 编程示例




```
0   LD    X000
1   PLS   M0
3   LD    M0
4   SET   Y000
5   LD    X000
6   PLF   M1
8   LD    M1
9   RST   Y000
10  LDP   X001
12  OUT   M2
13  LD    M2
14  SET   Y001
15  LDF   X001
17  OUT   M3
18  LD    M3
19  RST   Y001
20  END
```

- 程序段 0－2 和 10－12 的动作相同，都是在线圈闭合的上升沿，驱动一个扫描周期的输出。同样，程序段 5－7 和 15－17 的动作相同，都是在在线圈闭合的下降沿，驱动一个扫描周期的输出。
- 关于 SET,RST 指令的作用详见后面的说明。

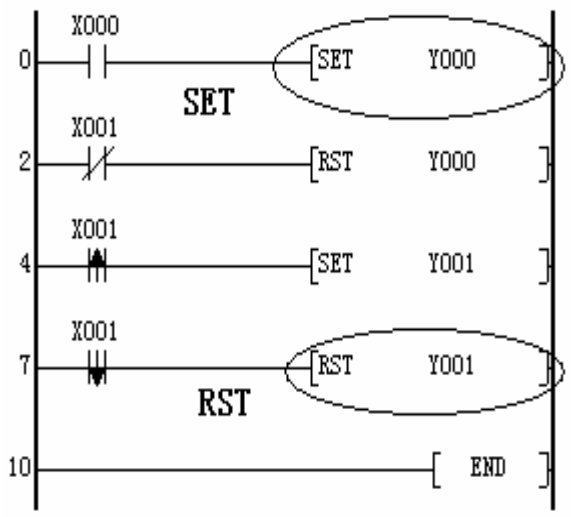
2.8　[SET],[RST] 指令

2.8.1 指令解说

助记符、名称	功能	可用软元件	程序步
SET 置位	动作保持	Y,M,S	见说明
RST 复位	清除动作保持，寄存器清零	Y,M,S,T,C,D,V,Z	

- 软元件为 Y 和一般 M 的程序步为 1，S 和特殊辅助继电器 M、定时器 T、计数器 C 的程序步为 2，数据寄存器 D 以及变址寄存器 V 和 Z 的程序步为 3。
- SET 指令在线圈接通的时候就对软元件进行置位，只要置位了，除非用 RST 指令复位，否则将保持为 1 的状态。同样，对 RST 指令只要对软元件复位，将保持为 0 的状态，除非用 SET 指令置位。
- 对同一软元件，SET,RST 指令可以多次使用，顺序随意，但是程序最后的指令有效。
- RST 指令可以对数据寄存器(D)，变址寄存器(V,Z)，定时器(T)和计数器(C)，不论是保持还是非保持的都可以复位置零。

2.8.2 编程示例




```
0   LD    X000
1   SET   Y000
2   LDI   X001
3   RST   Y000
4   LDP   X001
6   SET   Y001
7   LDF   X001
8   RST   Y001
10  END
```

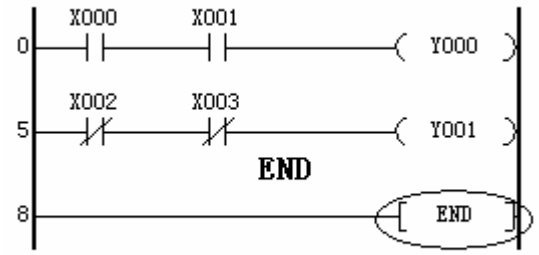
2.9　[NOP],[END] 指令

2.9.1 指令解说

助记符、名称	功能	可用软元件	程序步
NOP 空操作	无动作		1
END 结束	输入输出及返回到开始		1

- 程序清除时指令变为 NOP 指令，指令之间加入 NOP 指令，程序对他不做任何事情，继续向下执行，只是增加了程序的步数。
- 每个程序必须有一个且只有一个 END 指令，表示程序的结束。PLC 不断反复进行如下操作：输入处理，从程序的 0 步开始执行直到 END 指令，程序处理结束，接着进行输出刷新。然后开始循环操作。

2.9.2 编程示例




```
0    LD     X000
1    AND    X001
2    OUT    Y000
3    NOP
4    NOP
5    LDI    X002
6    ANI    X003
7    OUT    Y001
8    END
```

2.10　[MPS],[MRD],[MPP] 指令

2.10.1 指令解说

助记符、名称	功能	可用软元件	程序步
MPS 压栈	运算存储		1
MRD 读栈	存储读出		1
MPP 出栈	存储读出与复位		1

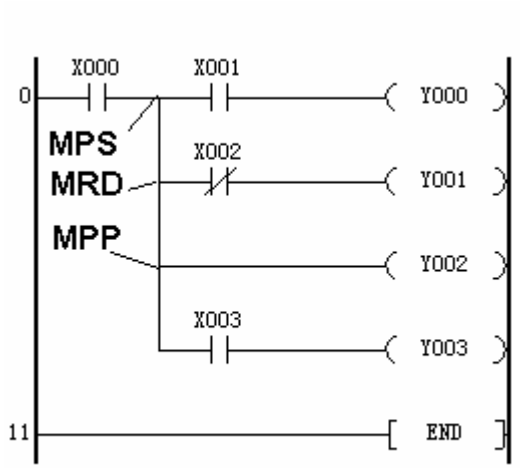
- 嵌入式 PLC 中有 11 个栈空间，也就是说可以压栈的最大深度为 11 级。每使用一次 MPS 将当前结果压入第一段存储，以前压入的结果依次移入下一段。MPP 指令将第一段读出，并且删除它，同时以下的单元依次向前移。MRD 指令读出第一段，但并不删除它。其他单元保持不变。使用这三条指令可以方便多分支的编程。
- 在进行多分支编程时，MPS 保存前面的计算结果，以后的分支可以利用 MRD,MPP

从栈中读出前面的计算结果，再进行后面的计算。最后一个分支必须用 MPP，保证 MPS,MPP 使用的次数相同。注意，使用 MPP 以后，就不能再使用 MRD 读出运算结果，也就是 MPP 必须放在最后的分支使用。

- MRD 指令可以使用多次，没有限制。MPS 连续使用的最多次数为 11，但是可以多次使用。每个 MPS 指令都有一个 MPP 指令对应，MPP 的个数不能多于 MPS 的个数。

2.10.2 编程示例

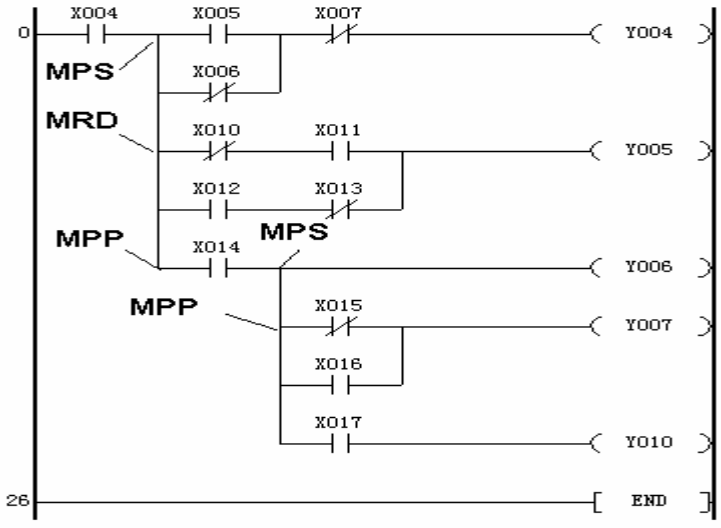
实例 1：




```
0   LD    X000
1   MPS
2   AND   X001
3   OUT   Y000
4   MRD
5   ANI   X002
6   OUT   Y001
7   MPP
8   OUT   Y002
9   AND   X003
10  OUT   Y003
11  END
```

- 该实例只使用一级堆栈，使用一个 MPS 指令压栈，一个 MRD 指令读栈，一个 MPP 指令出栈。

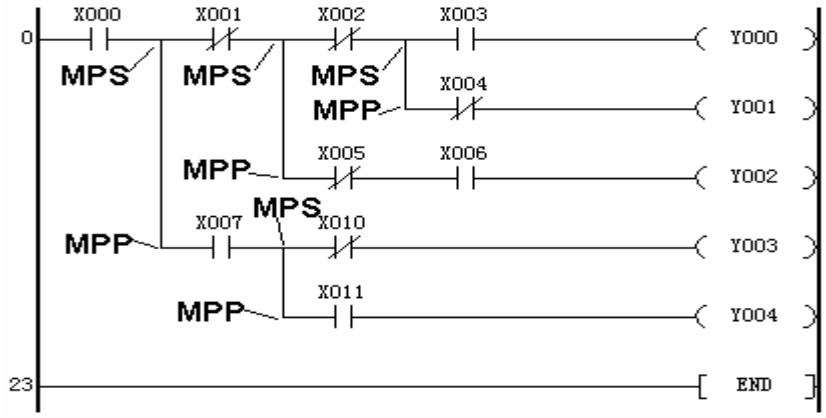
实例 2：



```
0   LD    X004
1   MPS
2   LD    X005
3   ORI   X006
4   ANB
5   ANI   X007
6   OUT   Y004
7   MRD
8   LDI   X010
9   AND   X011
10  LD    X012
11  ANI   X013
12  ORB
13  ANB
14  OUT   Y005
15  MPP
16  AND   X014
17  OUT   Y006
18  MPS
19  LDI   X015
20  OR    X016
21  ANB
22  OUT   Y007
23  MPP
24  AND   X017
25  OUT   Y010
26  END
```

- 该实例使用一级两段堆栈，并且跟 OR,ORB,ANB 指令混合使用。

实例 3



```
0    LD     X000
1    MPS
2    ANI    X001
3    MPS
4    ANI    X002
5    MPS
6    AND    X003
7    OUT    Y000
8    MPP
9    ANI    X004
10   OUT    Y001
11   MPP
12   ANI    X005
13   AND    X006
14   OUT    Y002
15   MPP
16   AND    X007
17   MPS
18   ANI    X010
19   OUT    Y003
20   MPP
21   AND    X011
22   OUT    Y004
23   END
```

该实例使用三级堆栈，即堆栈嵌套三级。

2.11　[MC],[MCR] 指令

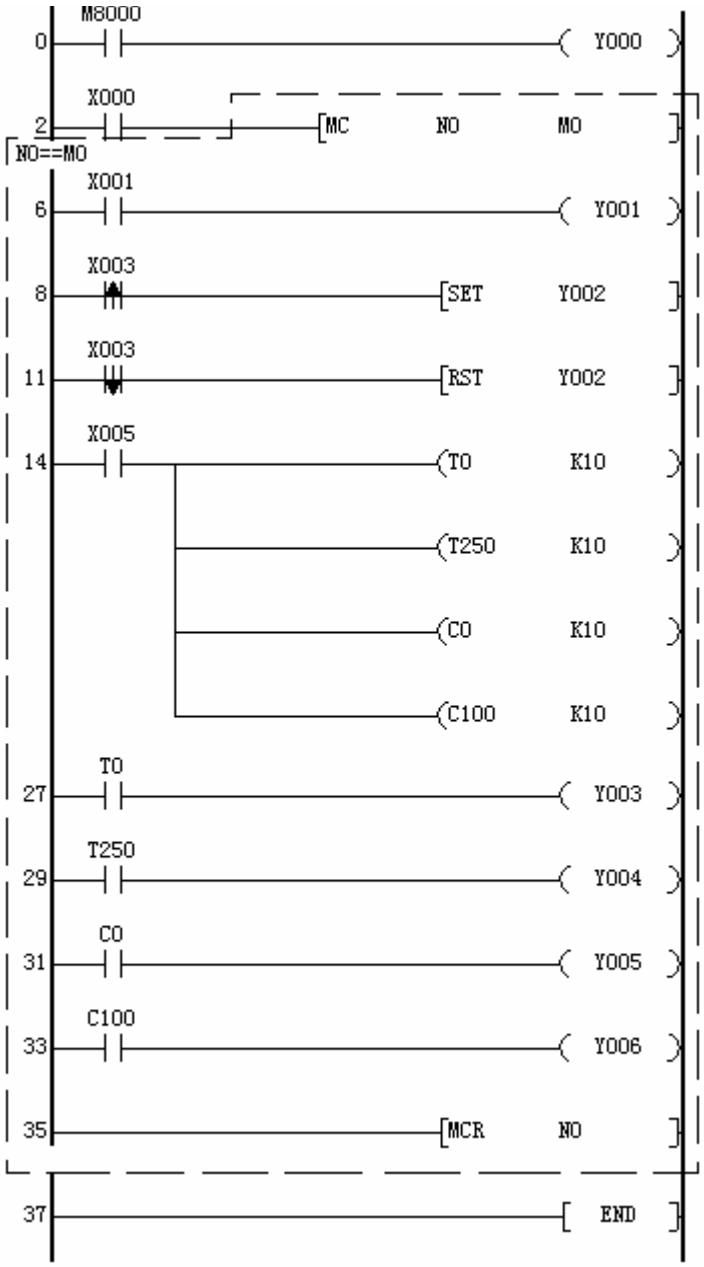
2.11.1 指令解说

助记符、名称	功能	可用软元件	程序步
MC 主控	公共串联点的连接线圈指令	Y,M（特殊 M 除外）	3
MCR 主控复位	公共串联点的消除指令		2

- 当前面的触点接通时，就执行 MC 到 MCR 的指令。执行 MC 指令时，母线向 MC 触点后移动，执行 MCR 指令返回母线。
- 使用 MC 指令时，嵌套级 N 的编号按顺序依次增大，也就是说只有使用 N0，才能嵌套 N1。相反使用 MCR 指令时，必须从大往小返回母线。最大嵌套级数为 7 级（N6）。
- 通过不同的软元件 Y,M，可以多次使用 MC 指令，如果使用相同的软元件，将同 OUT 指令一样，会出现双线圈输出。

2.11.2 编程示例

- 该实例只使用一个 MC,MCR 指令，嵌套级数也是 1，可以进行 7 级嵌套。
- 该实例中当 X000 接通时，执行 MC,MCR 之间的指令，当 X000 断开时，成为如下两种形式。
  现状保持：累积定时器的值，计数器的值，用 SET/RST 指令驱动的软元件。
  变为断开的元件：非累积定时器的值，用 OUT 指令驱动的软元件。



0	LD	M8000	
1	OUT	Y000	
2	LD	X000	
3	MC	N0	M0
6	LD	X001	
7	OUT	Y001	
8	LDP	X003	
10	SET	Y002	
11	LDF	X003	
13	RST	Y002	
14	LD	X005	
15	OUT	T0	K10
18	OUT	T250	K10
21	OUT	C0	K10
24	OUT	C100	K10
27	LD	T0	
28	OUT	Y003	
29	LD	T250	
30	OUT	Y004	
31	LD	C0	
32	OUT	Y005	
33	LD	C100	
34	OUT	Y006	
35	MCR	N0	
37	END		

第三章　步进顺控指令说明及应用

3.1 步进顺控指令说明

3.1.1 指令解说

助记符、名称	功能	可用软元件	程序步
STL	步序动作开始	S	1
RET	步序动作结束	无	1

步进控制方式(STL)是将控制被划分为多个工序状态（S），依据条件进行状态转移（SET ），逐步完成控制过程。

步进控制方式的特点是将复杂控制分步后，分别考虑好每一步的控制，从而降低了各步的关联，降低编程的复杂程度。

各状态内执行的动作由梯形图其它指令编写。

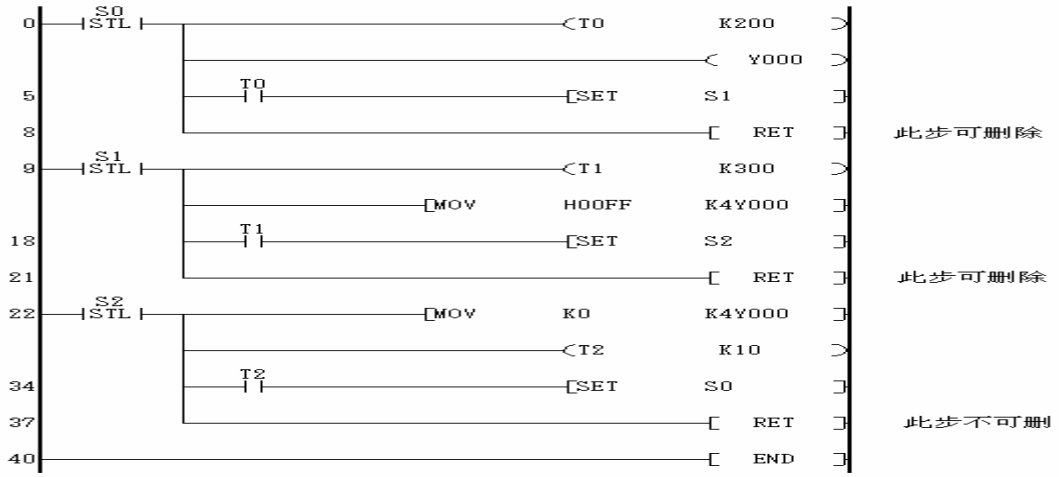
STL 是一个步序动作的开始指令。

RET 是一个步序动作的结束指令，其后指令返回母线。

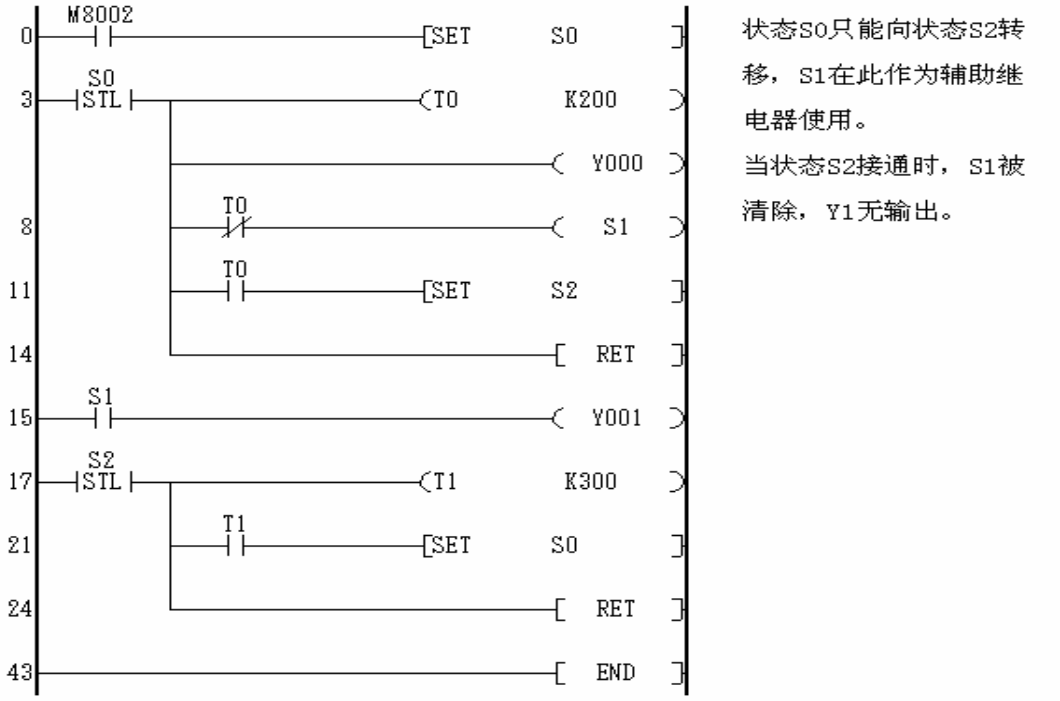
- **SET S_i** 是 **STL** 状态发生转移的唯一指令
- 规定：子程序内不能使用 STL----RET 指令。
- 当前状态（S0）向下一个状态（S1）转移时，该扫描周期两个状态内的动作均得到执行；下一扫描周期执行时，当前状态（S0）被下一状态（S1）所复位，当前状态（S0）内的所有动作不被执行，所有 OUT 元件的输入均被断开。
- 步序与步序之间一般省去 RET，因此看起来是多个 STL 可共用一个 RET。有 STL 而没有 RET，程序检查出错。

3.1.2 编程示例

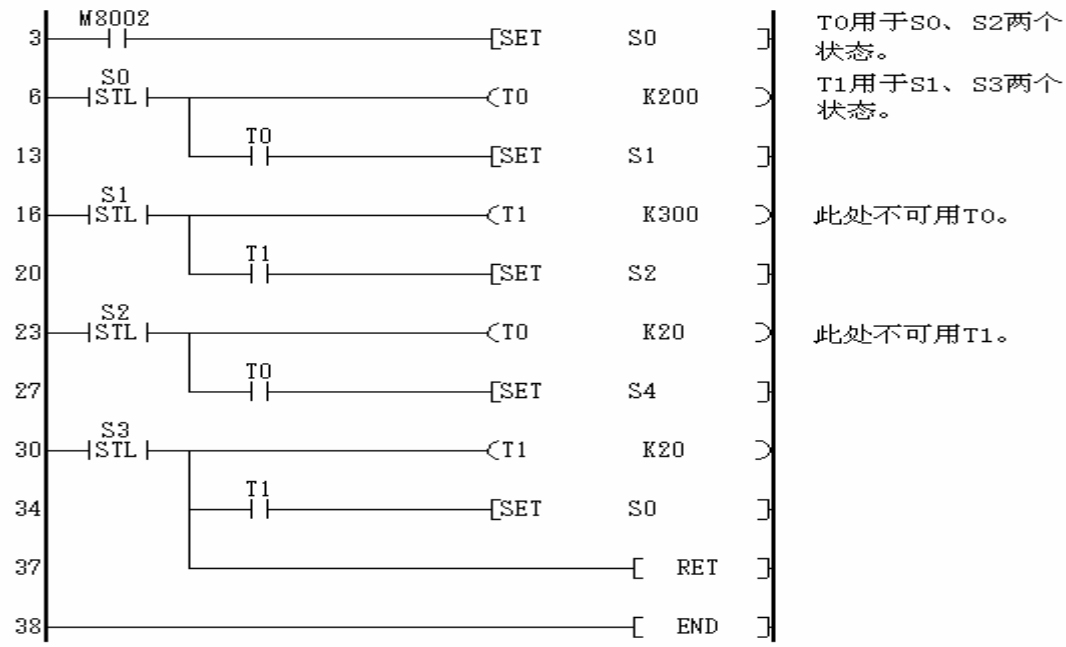
- 步序与步序之间一般省去 RET，因此看起来是多个 STL 可共用一个 RET。有 STL 而没有 RET，程序检查出错。



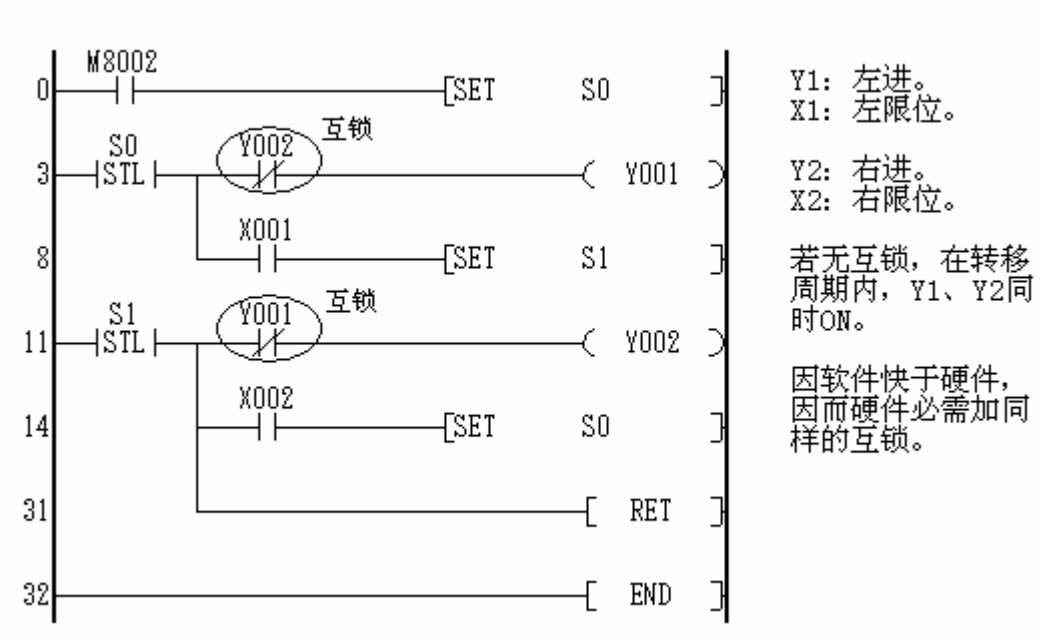
- 状态转移只能用 SET 指令，不能用 OUT 指令。
- 使用 OUT S 时，S 作为辅助继电器使用，而不是状态寄存器。



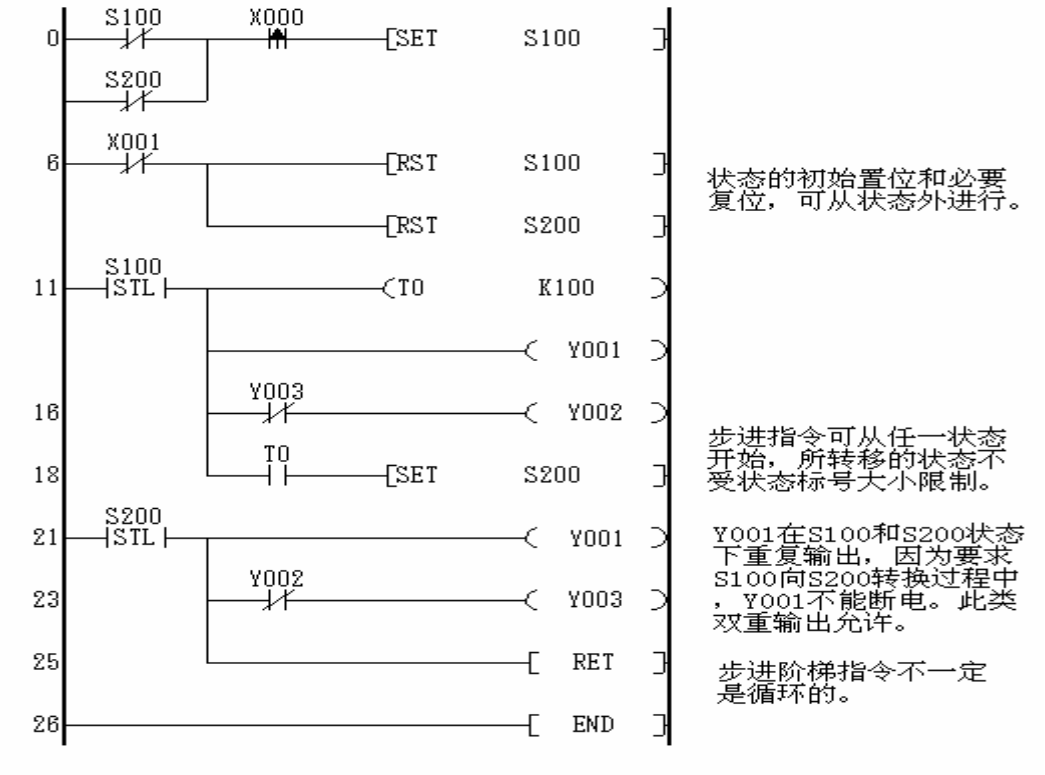
- 时间继电器 T 可重复使用，但相邻两个状态不能重复使用同一时间继电器。



- 两个矛盾继电器输出时，必需加软件互锁。考虑软件快于硬件，相矛盾的硬件输出也必需互锁。



- 允许同一继电器在不同状态下输出，其实际输出视状态转移的位置确定。



3.2　　步进顺控指令应用

3.2.1　单一流程示例

示例说明：

该程序描述一个自行葫芦自进入工位到走出工位的步序过程，若在葫芦升降过程中发生停电，来电后继续停电前的动作，并保证升或降动作总时间不变。

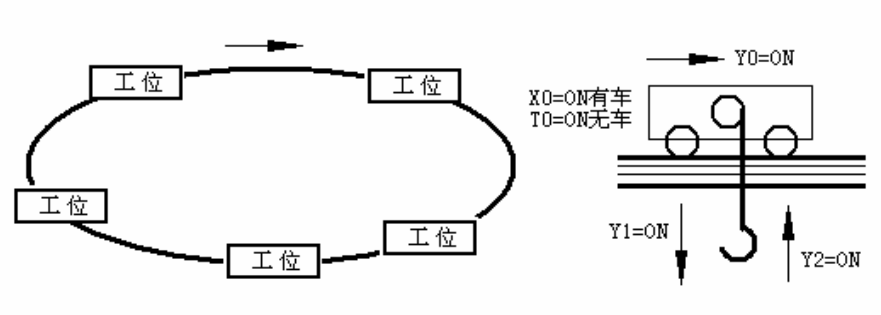
S500---S503 为停电保持型状态寄存器；C100---C101 为停电保持型计数器；

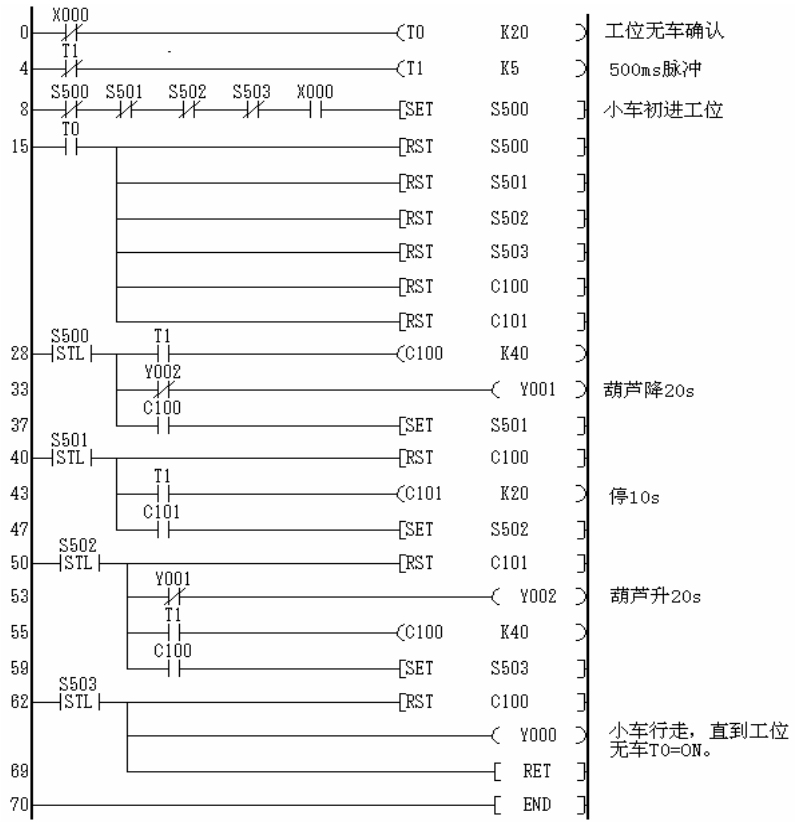
T0 延时 2 秒，作信号确认用；T1 作为 500 ms 脉冲发生器；

X0=ON 时，表示工位上停有自行葫芦；

T0=ON 时，表明工位上无自行葫芦；因信号由滑触线供给，因而 X0=OFF 时，不一定确定工位无车，需延时确认。

Y0 为驱动进车；Y1 驱动葫芦下降；Y2 驱动葫芦上升。





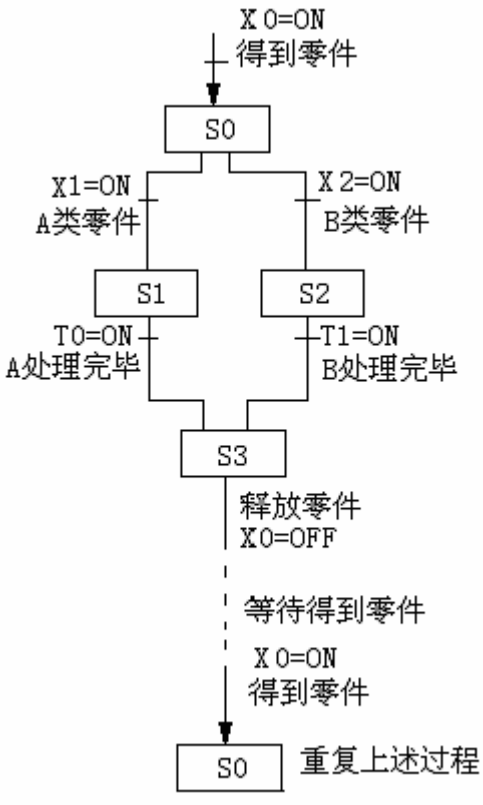
3.2.2 选择性分支与汇合示例

在步进顺控指令中，多个条件均可导致状态转移，但多个条件是互斥的，当一个条件成立时，另外条件便不能成立。这样的分支是选择性分支。
各选择性分支最终进行到一个共同的状态，我们称这一过程为选择性分支的汇合。

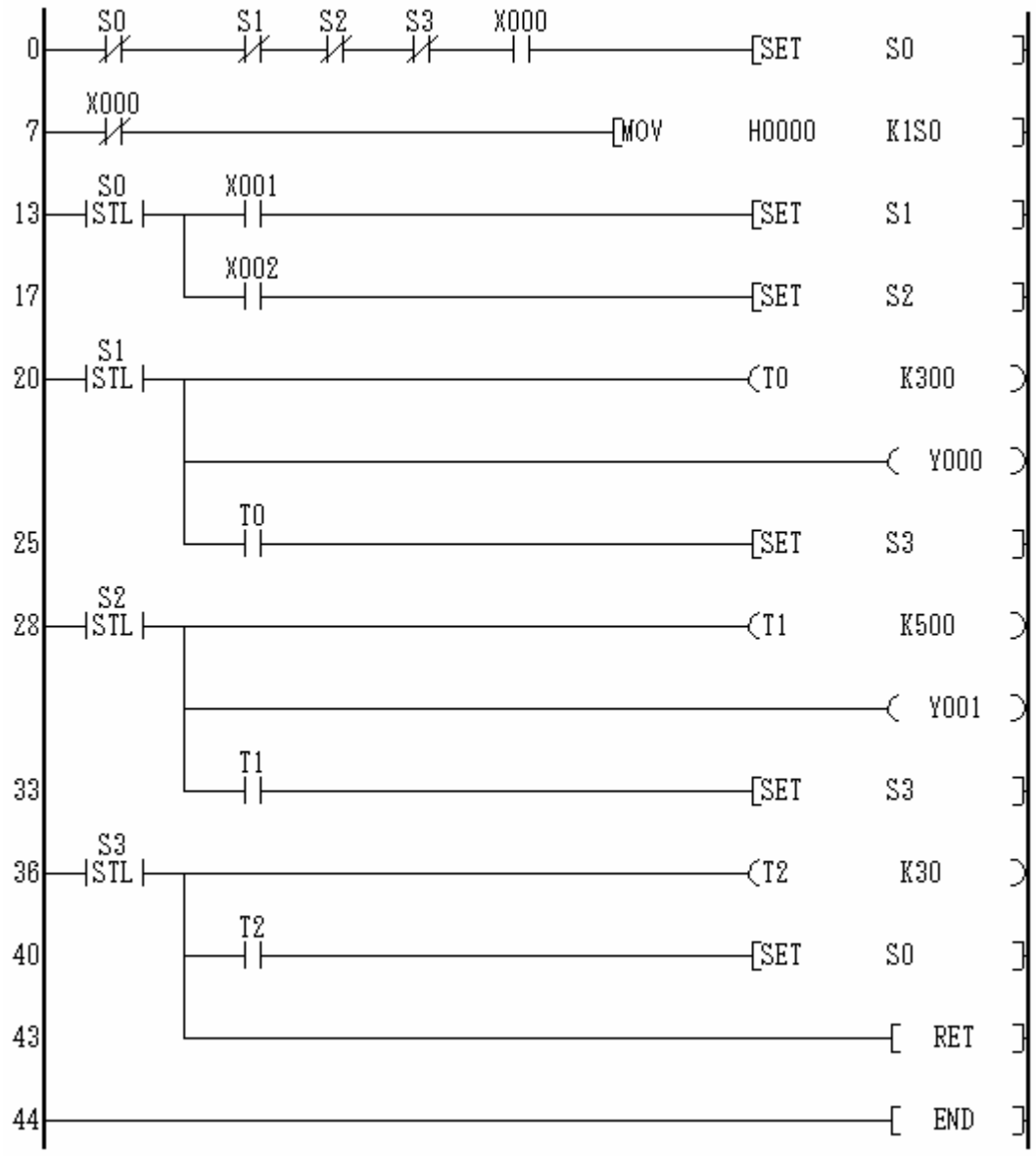
- 选择性分支分支数规定不能超过 8 路。

示例说明：
如产品输送线上有 A、B 两种产品，当机械手识别为 A 类产品时，进入 A 类流程处理；若识别为 B 类产品，进入 B 类流程处理，处理完后，放回输送线，进入下一工序。

流程示意图：



梯形图：



3.2.3　并行分支与汇合示例

在步进顺控指令中，一个条件导致多个状态发生，每个状态都按自已的流程独立进行状态转移，这些各自独立的状态流程称步进指令的并行分支。

多个同时独立进行状态转移的分支，当各分支状态同时有效时，整体才能进行到下一状态，我们称这一过程为并行分支的汇合。

并行分支汇合梯形图上表示为多个状态连续使用 STL 指令，连续使用 STL 的个数就是并行分支汇合的支路数。

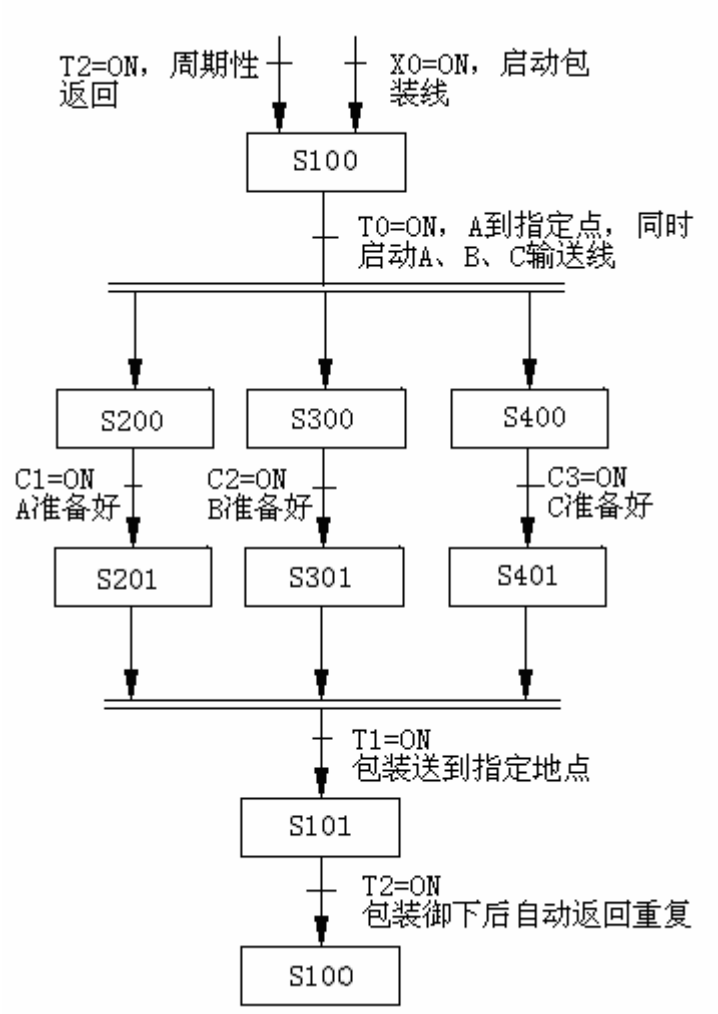
连续使用 STL 的个数规定不超过 8 个。

示例说明：

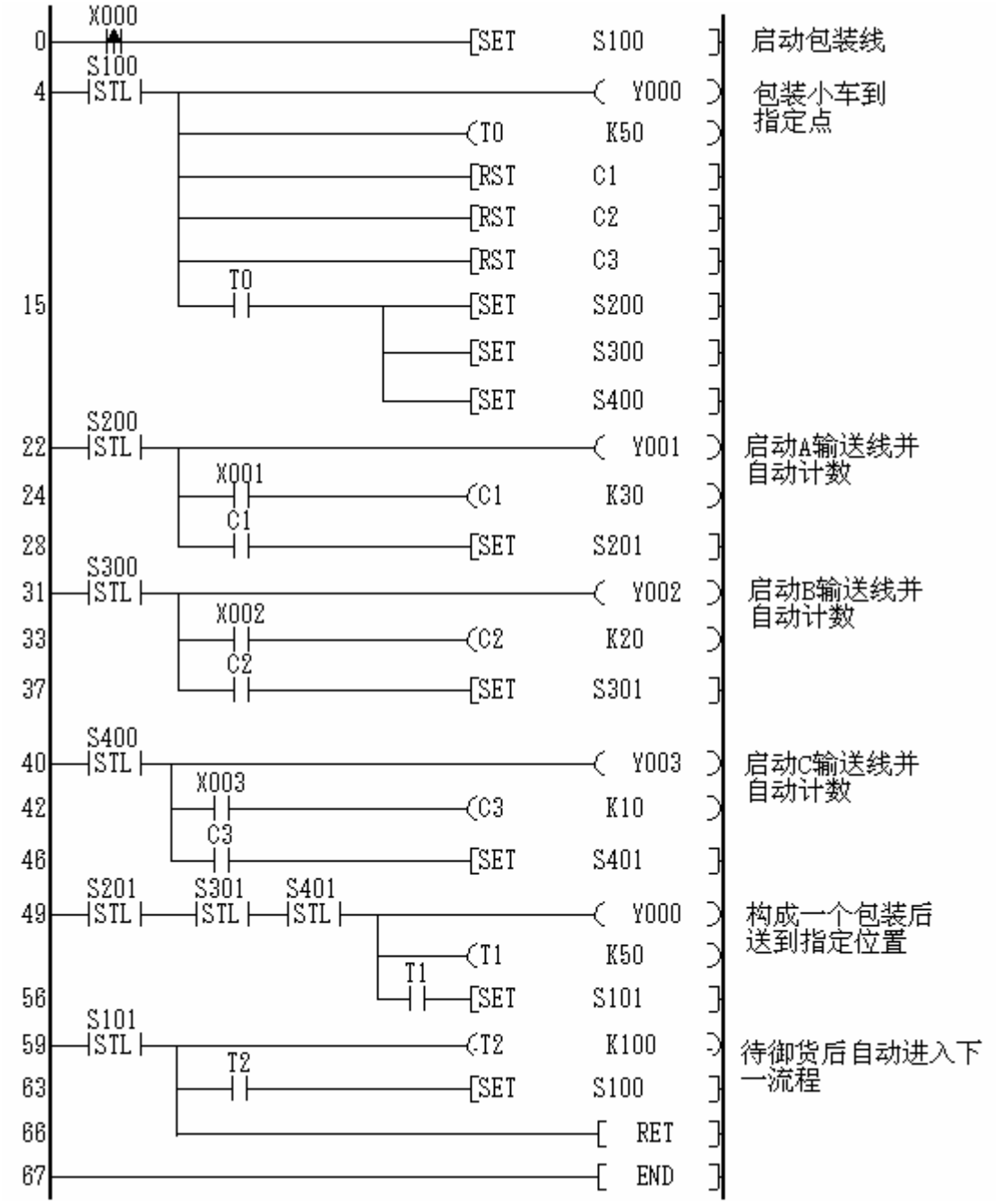
三条独立的产品线上，分别生产 A、B、C 三类产品，但包装入库必须按 30 件 A、20 件

B、10 件 C 组成一个包装。当任一产品数量不够时就不能构成一个包装。

流程示意图：



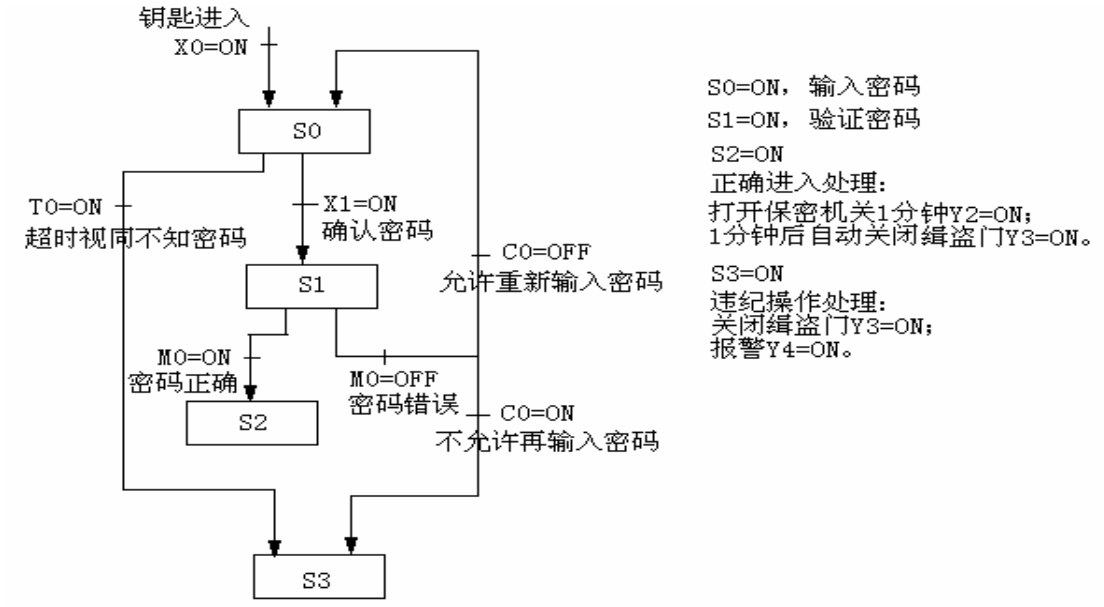
梯形图：



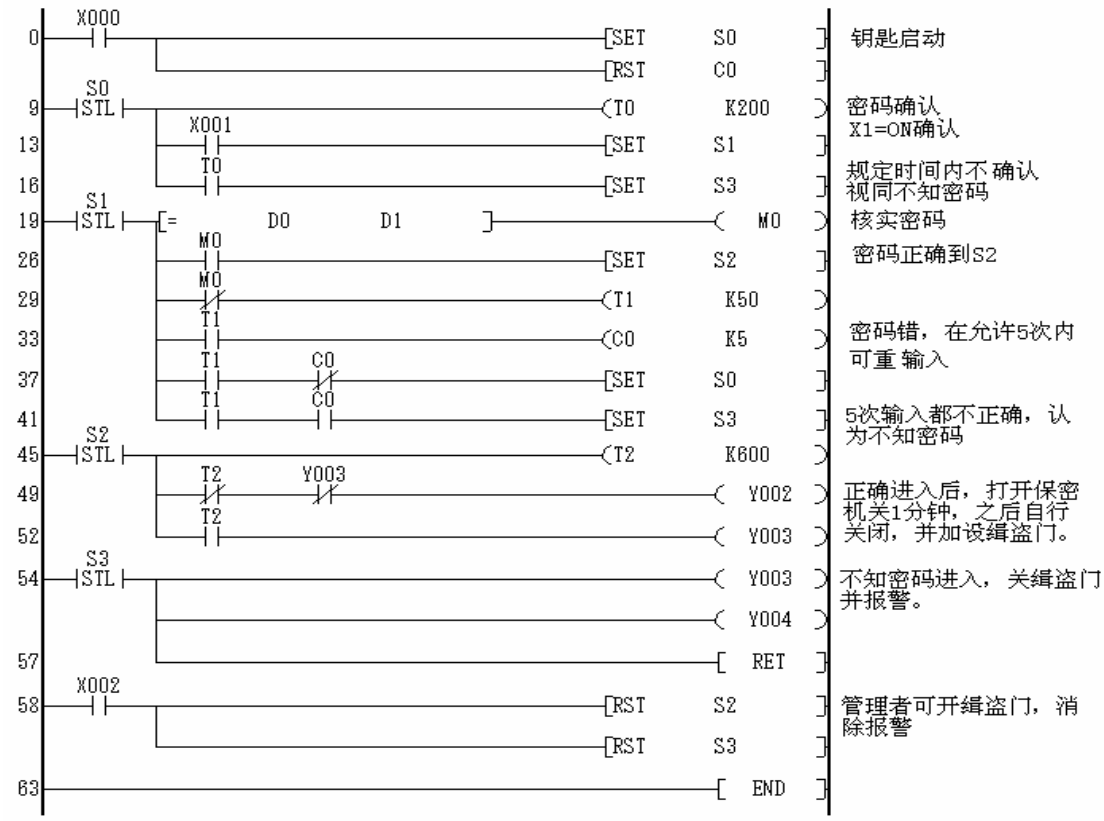
3.2.4 循环和跳转示例

实际使用过程中，常常涉及到很多混合形式，如：
选择性分支导致循环（第一个循环）和跳转（进入一个新的循环）。而每个大的循环内又有并行分支和汇合情况。
示例说明：
一个简易保安系统，在规定次数（如 5 次）的范围内，若密码不正确将启动报警系统，并关闭安全通道。若规定的次数内密码正确，进入密级操作。

流程示意图：


钥匙进入
X0=ON
S0
T0=ON
超时视同不知密码
X1=ON
确认密码
C0=OFF
允许重新输入密码
S1
M0=ON
密码正确
M0=OFF
密码错误
C0=ON
不允许再输入密码
S2
S3
S0=ON，输入密码
S1=ON，验证密码
S2=ON
正确进入处理：
打开保密机关1分钟Y2=ON；
1分钟后自动关闭缉盗门Y3=ON。
S3=ON
违纪操作处理：
关闭缉盗门Y3=ON；
报警Y4=ON。

梯形图：


X000
SET S0
RST C0
钥匙启动
S0
STL
T0 K200
密码确认
X1=ON确认
X001
SET S1
T0
SET S3
规定时间内不 确认
视同不知密码
S1
STL
[= D0 D1]
M0
核实密码
M0
SET S2
密码正确到S2
M0
T1 K50
T1
C0 K5
密码错，在允许5次内
可重 输入
T1 C0
SET S0
T1 C0
SET S3
5次输入都不正确，认
为不知密码
S2
STL
T2 K600
T2 Y003
Y002
正确进入后，打开保密
机关1分钟，之后自行
关闭，并加设缉盗门。
T2
Y003
S3
STL
Y003
不知密码进入，关缉盗门
并报警。
Y004
RET
X002
RST S2
RST S3
管理者可开缉盗门，消
除报警
END

第四章　功能指令说明及应用

4.1 功能指令一览表

分类	指令助记符	功能	页码
程序流程	CJ	条件跳转	
	CALL	子程序调用	
	SRET	子程序返回	
	FEND	主程序结束	
	FOR	循环范围开始	
	NEXT	循环范围结束	
传送与比较	CMP	比较	
	ZCP	区域比较	
	MOV	传送	
	CML	反向传送	
	BCD	BCD 转换	
	BIN	BIN 转换	
四则逻辑运算	ADD	BIN 加法	
	SUB	BIN 减法	
	MUL	BIN 乘法	
	DIV	BIN 除法	
	INC	BIN 加 1	
	DEC	BIN 减 1	
	WAND	逻辑字与	
	WOR	逻辑字或	
	WXOR	逻辑字异或	
	NEG	求补码	
	SQR	BIN 开方	
循环与移位	ROR	循环右移	
	ROL	循环左移	
	RCR	带进位循环右移	
	RCL	带进位循环左移	

基本功能指令一览表（续）

分类	指令助记符	功能	页码
浮点数运算	ECMP	2 进制浮点数比较	
	EZCP	2 进制浮点数区域比较	
	EBCD	2 进制浮点数转 10 进制浮点数	
	EBIN	10 进制浮点数转 2 进制浮点数	
	EADD	2 进制浮点数加法	
	ESUB	2 进制浮点数减法	
	EMUL	2 进制浮点数乘法	
	EDIV	2 进制浮点数除法	
	ESQR	2 进制浮点数开方	
	INT	2 进制浮点数转 BIN 整数	
	FLT	BIN 整数转 2 进制浮点数	
接点比较	LD=	（S1）=（S2）	
	LD>	（S1）>（S2）	
	LD<	（S1）<（S2）	
	LD<>	（S1）<>（S2）	
	LD≦	（S1）≦（S2）	
	LD≧	（S1）≧（S2）	
	AND=	（S1）=（S2）	
	AND>	（S1）>（S2）	
	AND<	（S1）<（S2）	
	AND<>	（S1）<>（S2）	
	AND≦	（S1）≦（S2）	
	AND≧	（S1）≧（S2）	
	OR=	（S1）=（S2）	
	OR>	（S1）>（S2）	
	OR<	（S1）<（S2）	
	OR<>	（S1）<>（S2）	
	OR≦	（S1）≦（S2）	
	OR≧	（S1）≧（S2）	

4.2 程序流程

4.2.1 条件跳转 [CJ]

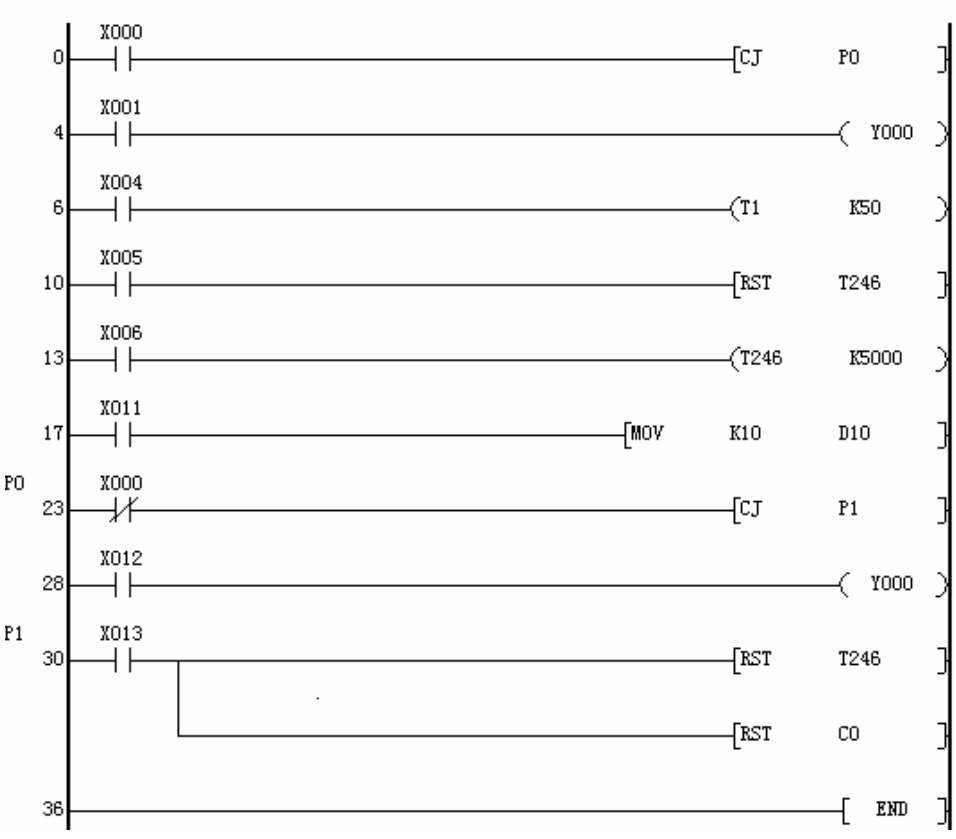
	CJ	
		P

16 位指令 CJ（连续执行型）
3 步　　CJP （脉冲执行型）

适用软元件	指针（P）可以指定下列编号 • P0~P127 • 指针编号可作变址修改

功能和动作

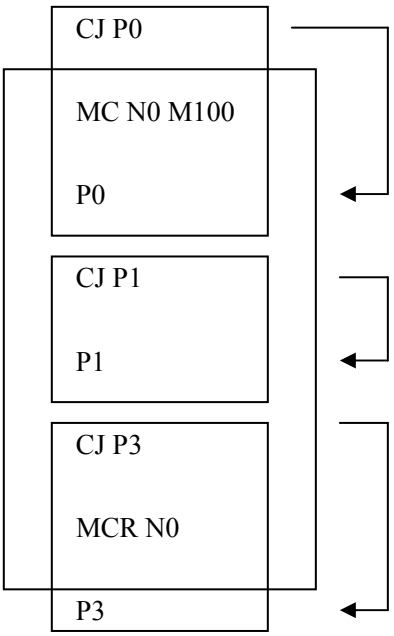
作为执行序列的一部分的指令，有 CJ、CJP 指令，可以缩短运算周期。



在上图示例中，如果 X000“ON”，则从 0 步跳到 23 步（标记 P0 的后一步）。X000“OFF”时，不进行跳转，顺序执行。

当 X000“ON”时，进行跳转，跳转中的线圈动作如下：

- Y、M、S 保持以前动作；
- T 在跳转前若没有触发，跳转后即使触发，定时器也不动作。若被触发，时钟继续运行，但触点不动作，当 X000“OFF”时，触点立即动作；
- C 在跳转前若没有触发，跳转后即使触发，计数器不动作。若被触发，计数中断，当 X000“OFF”时继续计数；
- 功能指令跳转后不动作；
- 定时器及计数器的复位指令在跳转外时，计时线圈及跳转的计数线圈复位（接点复位及当前值的清除）有效；
- 对 END 步跳转，需标明标号（P0~P127 都可以），线圈动作如上。
- 主控制指令和跳转指令的关系及动作如下，



• 从 MC 外向 MC 内跳转时，与 MC 的动作无关，即使 M100 处于“OFF”状态下，P0 以下 M100 视为“ON”；

• 从 MC 内向 MC 内跳转时，M100 处于“OFF”时，不能跳转；

• 从 MC 内向 MC 外跳转时，M100 处于“OFF”时，不能跳转，当 M0“ON”时，可跳转，但 MCR 无效

4.2.2 子程序调用 [CALL]

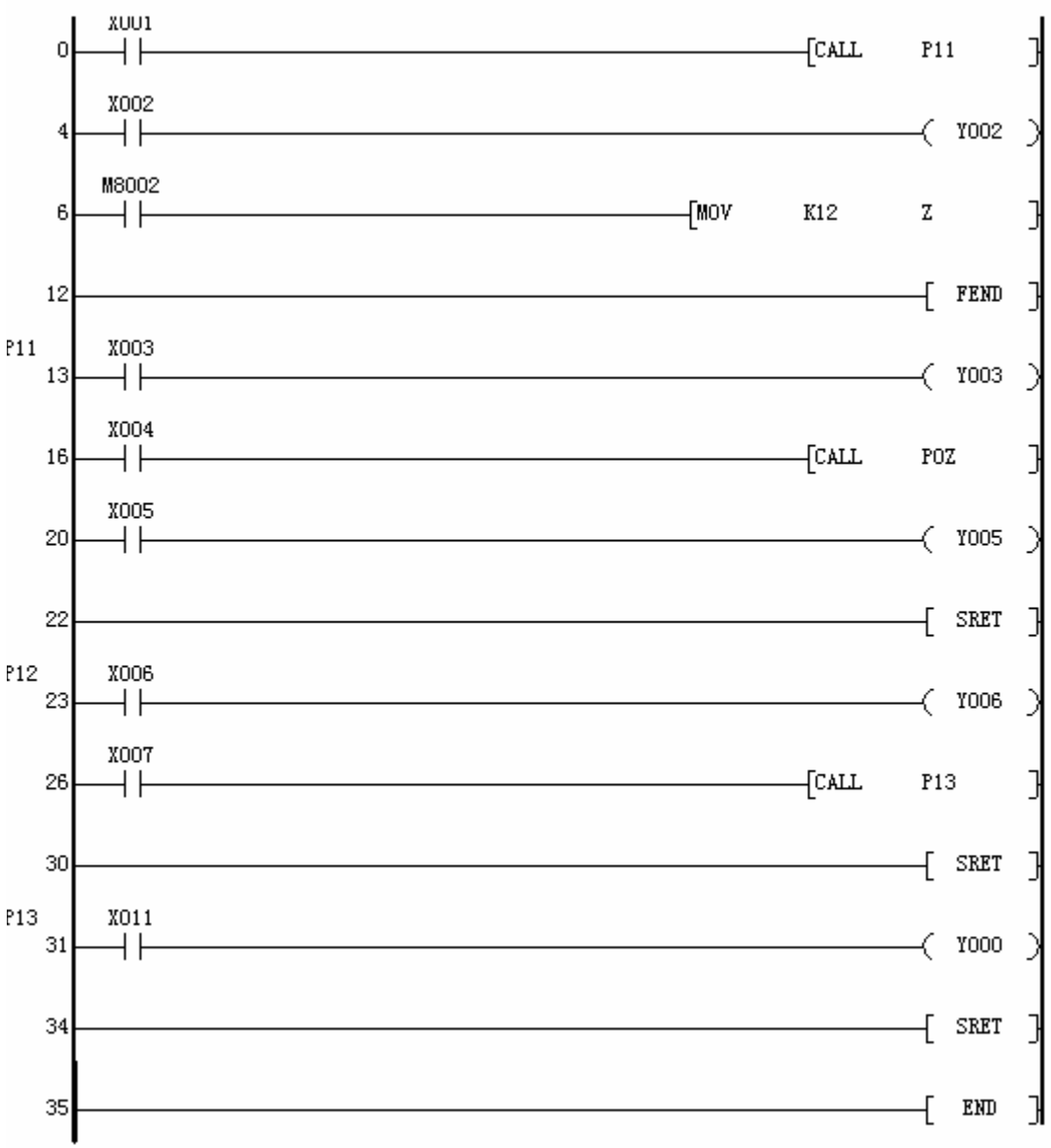
16 位指令 CALL （连续执行型）
3 步　　CALLP （脉冲执行型）

4.2.3 子程序返回 [SRET]

单独指令 SRET
1 步　不需要触点驱动的指令

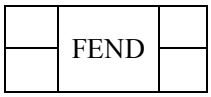
适用软元件	子程序调用的指针（P）可以指定下列编号 • P0~P127； • 指针编号可作变址修改； • 嵌套最多可为 5 层； • 对子程序返回无适用软元件。

功能和动作



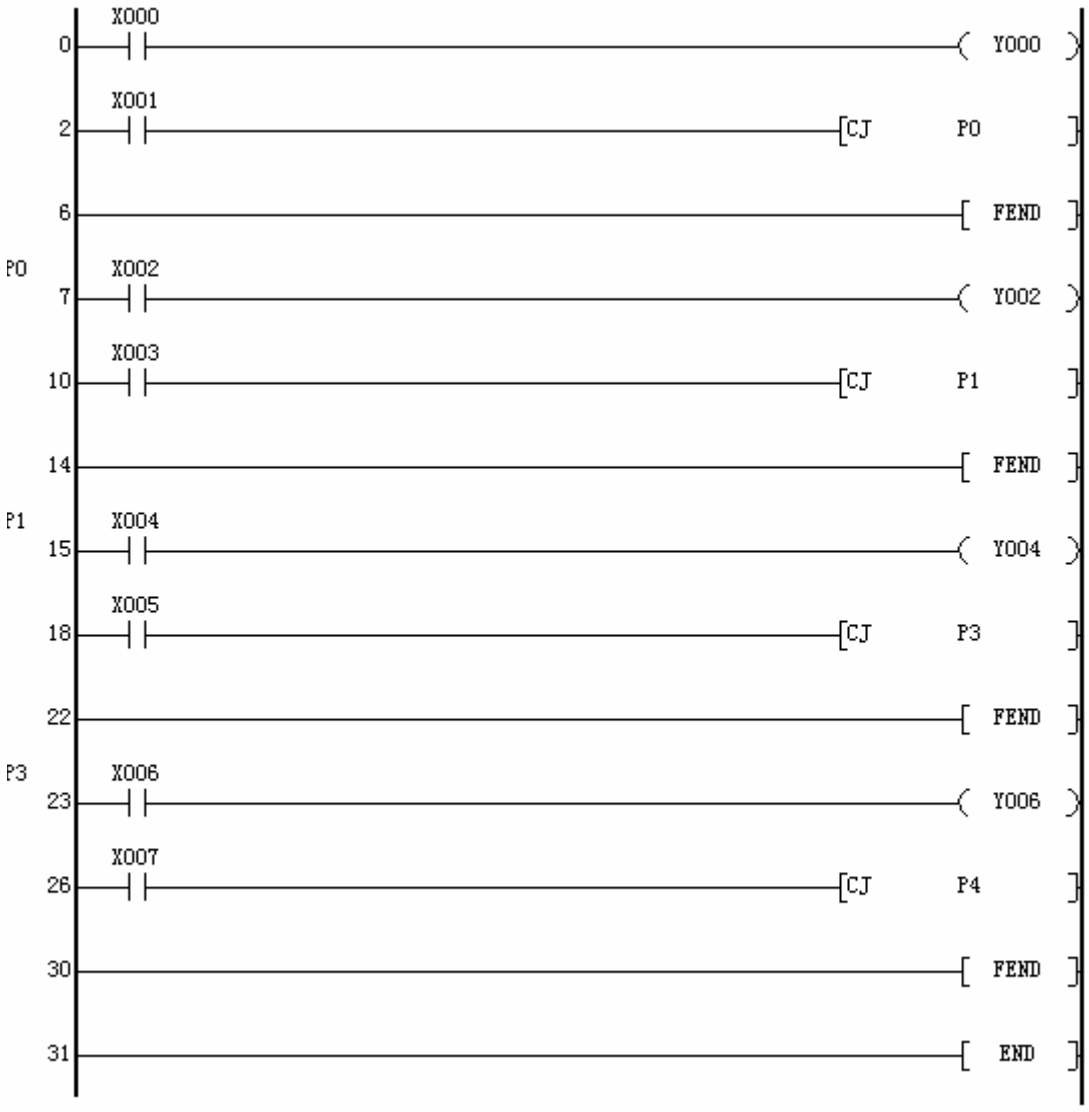
- 若 X001“ON”，则执行调用指令跳转到标记 P11 步，执行完通过执行 SRET 指令返回原来的步，再往下执行；
- 在 FEND 指令后对标记(子程序)编程；
- CJ 指令的标记和子程序的标记不能重复编号；
- 在子程序内最多可以允许有四层嵌套，如上例，还可增加 2 层，整体而言可做 5 层；
- 指针编号可作变址修改，如 P0Z（0+12=12），如果变址得出的编号没有，嵌入式 PLC 停止工作。

4.2.4 主程序结束 [FEND]



单独指令 FEND
1 步　　不需要触点驱动的指令

功能和动作



- 当程序使用多个 FEND 指令时，子程序请在最后的 FEND 指令与 END 指令之间编写。

4.2.5 循环范围开始［FOR]

	FOR	

16 位指令 FOR （连续执行型）
3 步

适用软元件	• 字软元件 K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z， • 可作变址修改。

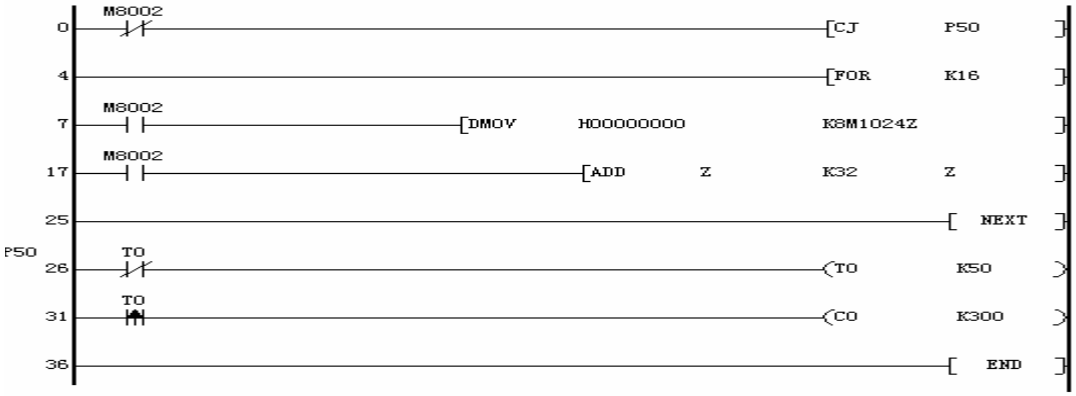
4.2.6 循环范围结束 [NEXT]

	NEXT	

单一指令 NEXT
1 步　　不需要触点驱动的指令。

功能和动作

只在 FOR~NEXT 指令之间的处理（利用源数据指定的次数），执行完后，才处理 NEXT 指令以后的程序。



- 上图是通电时对保持用辅助继电器复位的程序；
- 从 4 步至 25 步之间的程序执行了 16 次，执行完后 Z 的值为 512；

- FOR ~NEXT 嵌套最多 5 层；
- 循环次数多时扫描周期会延长，请务必注意；
- NEXT 指令在 FOR 指令之前，或无 NEXT 指令，或在 FEND、END 指令以后有 NEXT 指令，或 FOR 指令与 NEXT 指令个数不相等，都会出错；
- 若不想执行 FOR~NEXT 之间的程序时，利用 CJ 指令，使之跳转。如在上图所例，在 25 步前插入 LDI　M0　　CJ　P50 则 Z 的值为 32，即只执行了一次。

4.3　传送与比较

4.3.1 比较指令 [CMP]

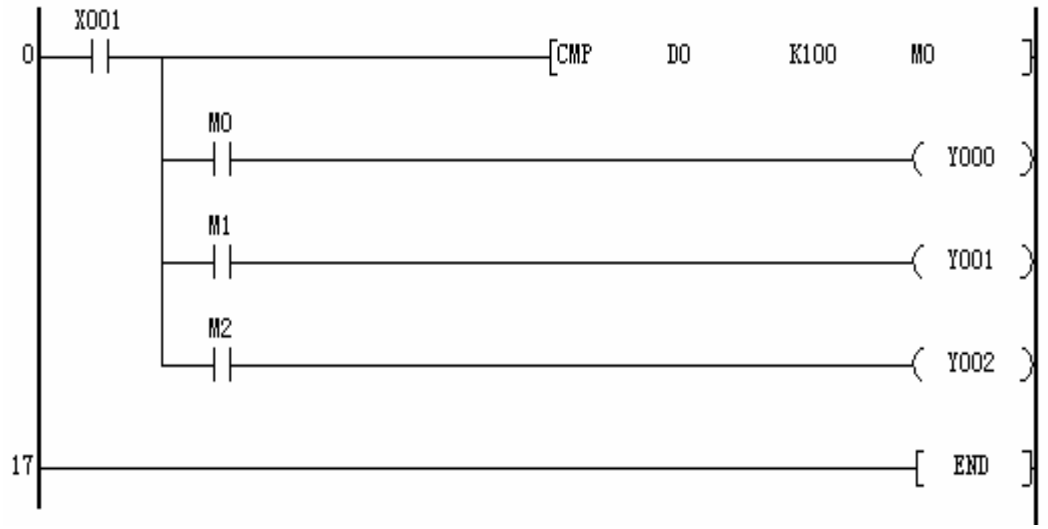
	CMP	
D		P

16 位指令 CMP （连续执行型）
7 步　　CMPP （脉冲执行型）

32 位指令 DCMP （连续执行型）
13 步　　DCMPP （脉冲执行型）

适用软元件	• 字软元件（S1 • 、S2 •） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z • 位软元件（D •） Y、M、S

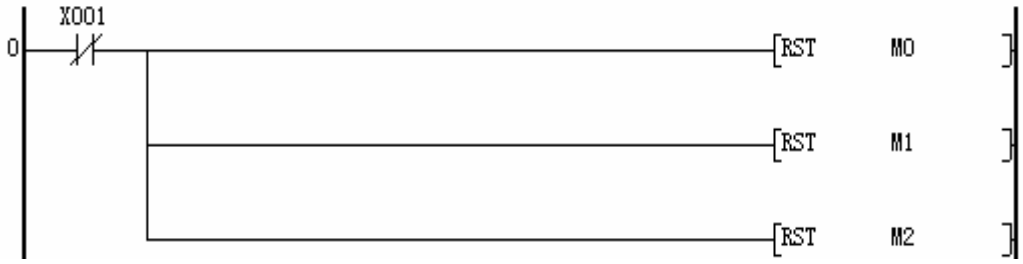
功能和动作



- 上图示例是 D0 的内容与常数 100 进行比较，大小比较是按代数形式进行的（-8<0）；

- 所有数据都以 2 进制值处理；
- 当 D0>100，M0“ON” ，当 D0=100，M1“ON” ，当 D0<100，M2“ON”；
- 目标地址指定 M0，则 M1、M2 被自动占用；
- 当 X001“OFF”时，M0、M1、M2 仍保持以前状态。如当 D0 的内容为 50，则 50<100，M2“ON” ，M0、M1 都“OFF” ，X001“OFF”时，M2 仍“ON” 。

指令不执行时，想要清除比较结果，可使用复位指令。



4.3.2 区域比较［ZCP]

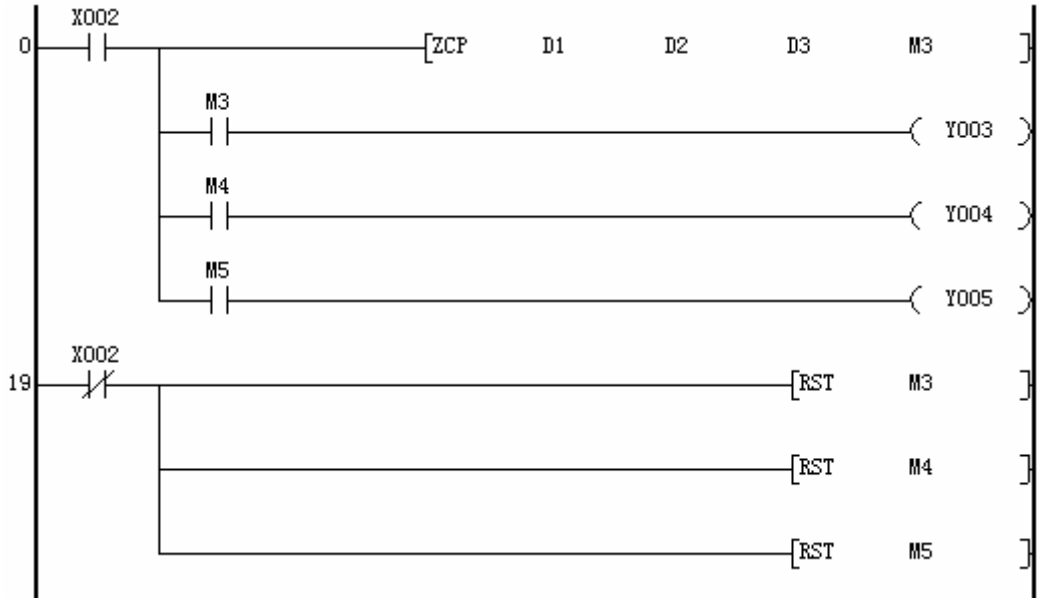
	ZCP	
D		P

16 位指令 ZCP （连续执行型）
7 步　　ZCPP （脉冲执行型）

32 位指令 DZCP （连续执行型）
13 步　　DZCPP （脉冲执行型）

适用软元件	• 字软元件（S1 •、S2 •、S •）　K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z • 位软元件（D •）　　Y、M、S

功能和动作



- 如上例，D3 的内容与 D1、D2 的内容进行比较；
- D1 的内容应小于等于 D2　的内容，若 D1=100，D2=80，比较时 D2 的内容为 100；
- 按代数形式进行比较（-8<0）；
- 当 D1>D3，则 M3“ON” ；当 D1≦D3≦D2，则 M4“ON” ，当 D2<D3，则 M5“ON” 。

4.3.3 传送指令［MOV]

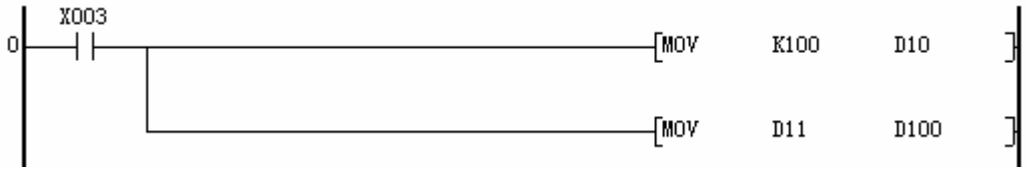
	MOV	
D		P

16 位指令 MOV （连续执行型）
7 步　　MOVP （脉冲执行型）

32 位指令 DMOV （连续执行型）
13 步　　DMOVP （脉冲执行型）

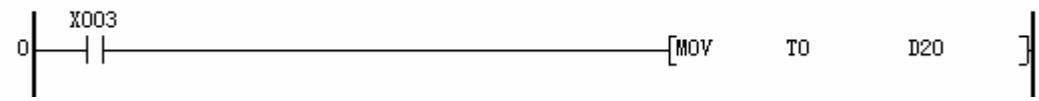
适用软元件	• 字软元件（S •） K、H、KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作　使数据原样传送的指令。



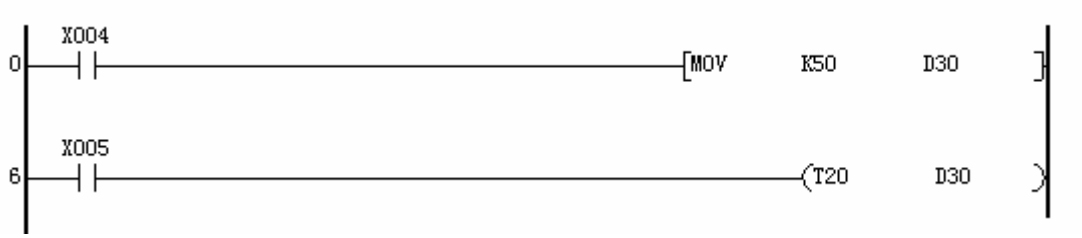
- 将源（S •）的内容向目标（D •）传送，X003“OFF”时，目标（D •）的内容不变化；
- 常数 K100 被自动转换成 BIN 码。

《定时、计数器的当前值读出示例》



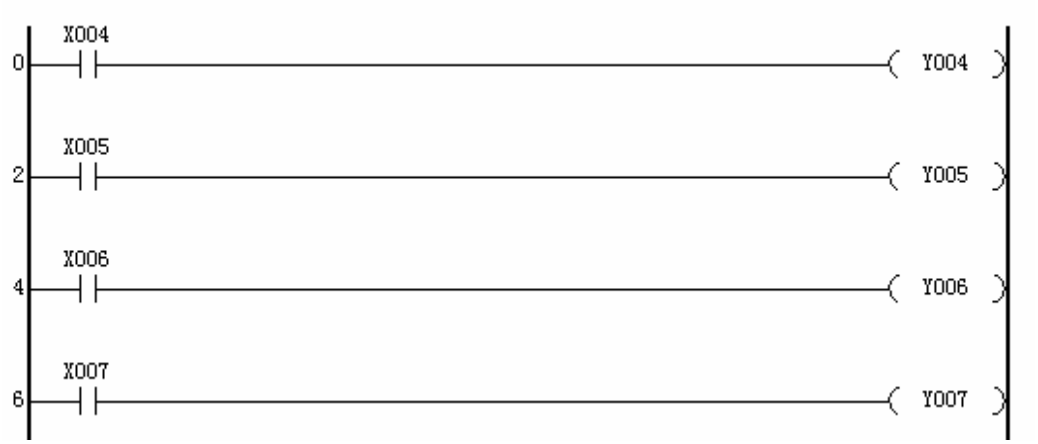
- 将 T0 当前值传送给 D20。

《定时、计数器设定值的间接指定示例》

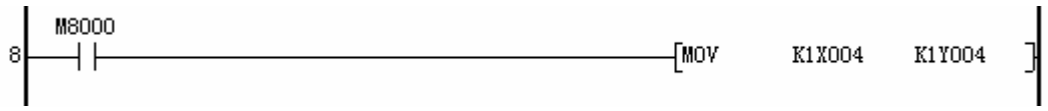


- T20 定时时间为 5 秒。

《位软元件的传送》

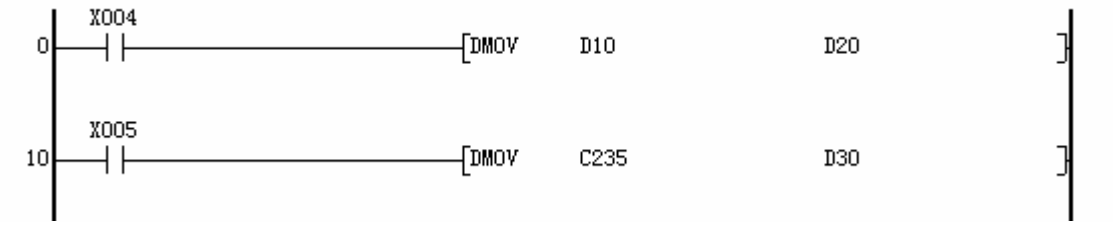


- 上图示例可用下面的 MOV 指令来实现，



《32 位数据的传送》

运算结果是 32 位的应用指令（MUL 等）、32 位数值、32 位软元件或 32 位计数器等 32 位数据的传送，必须使用 DMOV 指令。



- 上例将（D11、D10）的内容传送给（D21、D20），
  （C235 的当前值）传送给（D31、D30）。

4.3.4 反向传送 [CML]

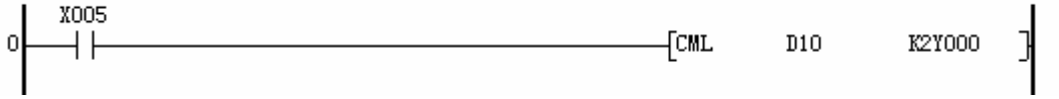
	CML	
D		P

16 位指令 CML （连续执行型）
5 步　　CMLP （脉冲执行型）

32 位指令 DCML （连续执行型）
13 步　　DCMLP （脉冲执行型）

适用软元件	• 字软元件（S •） K、H 、KnX、KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作 将数据反向传送的指令



- 将 D0 的内容每位取反（0 取反为 1，1 取反为 0）后，传送到目标地址，常数 K 被自动转换成 2 进制。如：

D10

1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

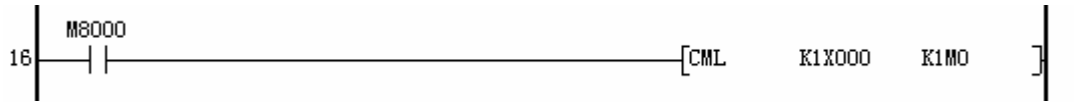
符号位（0=正数， 1=负数）

								0	1	0	1	0	1	0	1

Y17 Y16 Y15 Y14 Y13 Y12 Y11 Y10 Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0

无变化　　反向数据被传送

上例可用 CML 指令来实现。



4.3.5 BCD 转换［BCD］

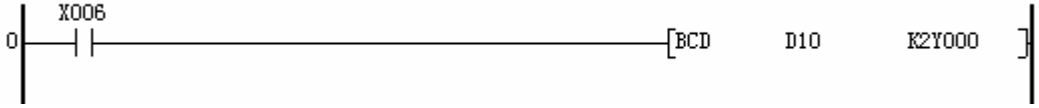
D	BCD	P

16 位指令 BCD （连续执行型）
5 步　　BCDP （脉冲执行型）

32 位指令 DBCD （连续执行型）
9 步　　DBCDP （脉冲执行型）

适用软元件	• 字软元件（S •） KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作　将源（BIN）转换为目标（BCD）的指令。



- 使用 BCD、BCDP 指令，转换结果不能超出 0~9999，使用 DBCD、DBCDP 指令，转换结果不能超出 0~99999999；
- 将 PLC 内的 2 进制数变为七段显示等的 BCD 码向外部输出时使用。

4.3.6 BIN 转换 [BIN]

	BIN	
D		P

16 位指令 BIN （连续执行型）
5 步　　BINP （脉冲执行型）

32 位指令 DBIN （连续执行型）
9 步　　DBINP （脉冲执行型）

适用软元件	• 字软元件（S •） KnX、KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作　将源（BCD）转换为目标（BIN）的指令。

- 使用 BIN、BINP 指令，源数据（S •）不能超出 0~9999，使用 DIND、DBINP 指令，源数据（S •）不能超出 0~99999999；
- 常数 K 能自动转成 2 进制。

4.4　四则逻辑运算

4.4.1 BIN 加法运算 [ADD]

	ADD	
D		P

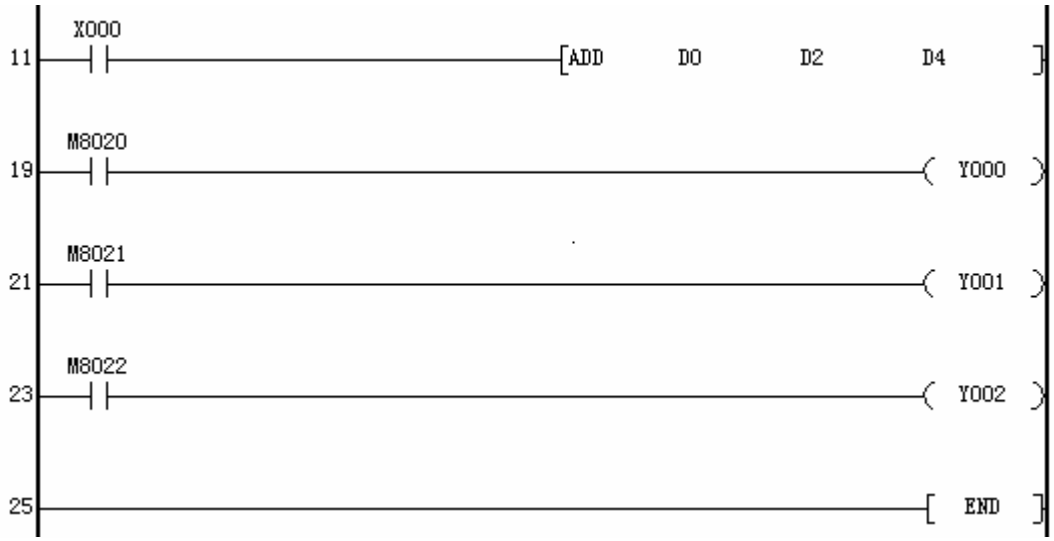
16 位指令 ADD （连续执行型）
7 步　　ADDP （脉冲执行型）

32 位指令 DADD （连续执行型）
13 步　　DADDP （脉冲执行型）

适用软元件	• 字软元件（S1 • 、S2 •） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

标志位	零	M8020
	借位	M8021
	进位	M8022

功能和动作



- 两个源数据进行加法后传送到目标处，各数据的最高位是符号位（正数为 0，负数为 1），数据以代数形式进行加法运算（8+（-8）=0）。
- 运算结果为 0 时，0 标志位 M8020 动作；运算结果超出 32767（16 位运算）或

2147483647（32 位运算）时，进位标志位 M8022 动作；运算结果小于-32768（16 位运算）或-2147483648（32 位运算）时，借位标志位 M8021 动作；

- 进行 32 位运算时，字软元件的低 16 位侧的软元件被指定，紧接着上述软元件编号后的软元件作为高位，为了防止编号重复，建议将软元件指定为偶数编号。
- 对于脉冲型指令，每出现一次 OFF 到 ON 的变化，操作数做一次运算。
- 可以将源（S •）和目标（D •）指定为相同的软元件编号。这种情况下，如使用连续执行型指令（ADD、DADD），则每个扫描周期加一次，请务必注意。

4.4.2 BIN 减法运算［SUB]

	SUB	
D		P

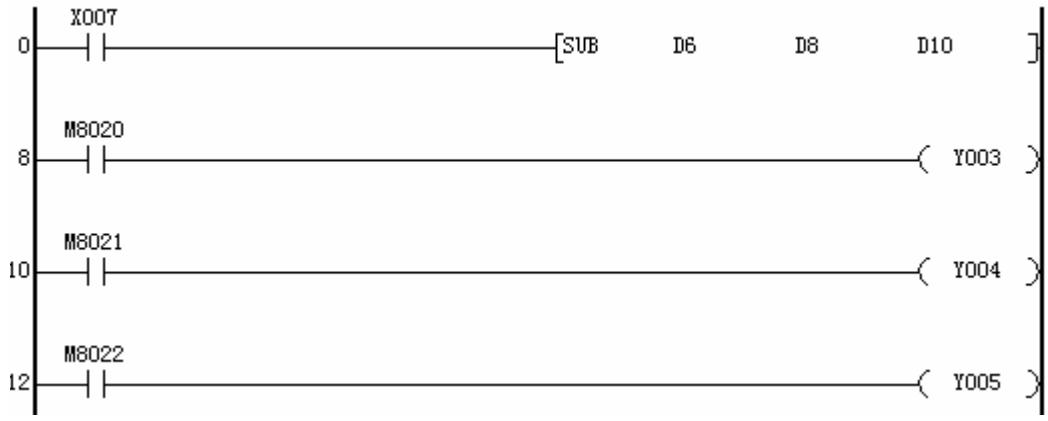
16 位指令 SUB （连续执行型）
7 步　　SUBP （脉冲执行型）

32 位指令 DSUB （连续执行型）
13 步　　DSUBP （脉冲执行型）

适用软元件	• 字软元件（S1 • 、S2 •） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z • 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

标志位	零	M8020
	借位	M8021
	进位	M8022

功能和动作



- （S1 •）指定的内容和（S2 •）指定的内容相减，结果存入（D •）指定的软元件

中。（8-（-8）=16）。

- 各种标志位的动作，32 位运算软元件的指定方法，连续型和脉冲型的差异等都跟 ADD 指令相同。

4.4.3 BIN 乘法运算［MUL]

	MUL	
D		P

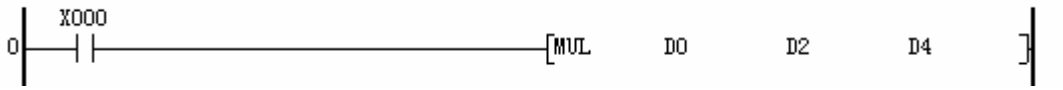
16 位指令 MUL （连续执行型）
7 步　　MULP （脉冲执行型）

32 位指令 DMUL （连续执行型）
13 步　　DMULP （脉冲执行型）

适用软元件	•字软元件（S1• 、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z（V，Z 仅限 16 位计算） •字软元件（D•） KnY、KnM、KnS、T、C、D、V，Z

功能和动作

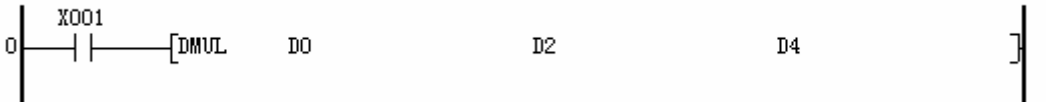
《16 位运算》



BIN　　BIN　　BIN
（D0）×（D2）⟶（D5，D4）
16 位　　16 位　　32 位

- 各源指定的软元件内容的乘积，以 32 位数据形式存入目标地址指定的软元件（低位）和紧接其后的软元件（高位）中，如　（D0）=125，（D2）=8，则（D5，D4）=1000；
- 结果的最高位是符号位，0 为正，1 为负；
- (D•)是位元件时，可以进行 K1∽K8 的位指定。指定为 K4 时，只能求得乘积运算的低 16 位。

《32 位运算》



BIN　　BIN　　BIN
（D1，D0）×（D3，D2）⟶（D7，D6，D5，D4）
32 位　　32 位　　64 位

- 在 32 位运算中，目标地址使用位软元件，只能得到低 32 位的结果，最好先向字元件传送一次后再进行运算；
- 即使使用字元件，也不能一下子监视 64 位数据的运算结果，此种情况下建议进行浮点数运算；
- 不能指定 Z 作为（D•）。

4.4.4 BIN 除法运算 [DIV]

	DIV	
D		P

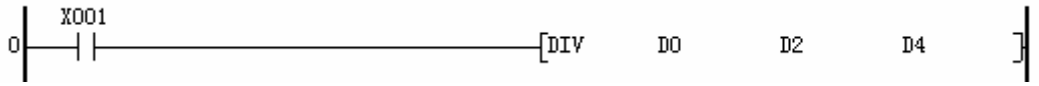
16 位指令 DIV （连续执行型）
7 步　　DIVP （脉冲执行型）

32 位指令 DDIV （连续执行型）
13 步　　DDIVP （脉冲执行型）

适用软元件	•字软元件（S1•、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z（V，Z 仅限 16 位计算） •字软元件（D•） KnY、KnM、KnS、T、C、D、V，Z

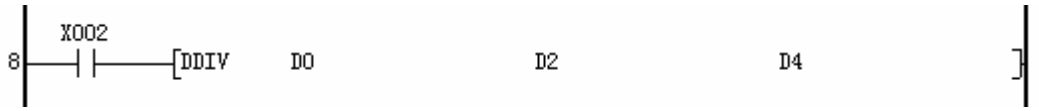
功能和动作

《16 位运算》



被除数　除数　　商　　余数
BIN　　BIN　　BIN　　BIN
（D0）÷（D2）⟶（D4）•••（D5）
16 位　　16 位　　16 位　　16 位

《32 位运算》



被除数　除数　商　余数
BIN　BIN　BIN　BIN
（D1，D0）÷（D3，D2）⟶（D5，D4）•••（D7，D6）
32 位　32 位　32 位　32 位

- 32 位运算不能指定 Z 作为（D •）；
- 除数为 0 时，如果被除数为正数，商为 32767（16 位）或 2147483647（32 位）；如果被除数为 0，商为 0；如果被除数为负数，商为-32768（16 位）或-2147483648（32 位）；
- 商和余数的最高位为符号位，0 为正，1 为负，当被除数或除数中的一方为负数时，商为负，当被除数为负时，余数则为负。

4.4.5 BIN 增 1［INC］

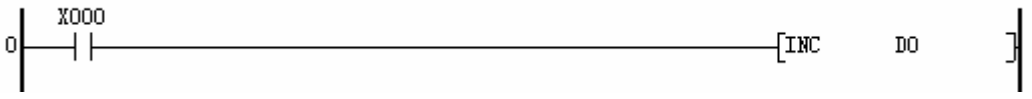
D	INC	P

16 位指令 INC （连续执行型）
3 步　　INCP （脉冲执行型）

32 位指令 DINC （连续执行型）
5 步　　DINCP （脉冲执行型）

适用软元件	•字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作



（D0）+1 ⟶（D0）

- X000 每置“ON”一次，D0 的内容增 1，在连续执行指令中，每个扫描周期执行加 1 运算，所以务必引起注意；
- 16 位运算时，如果 32767 加 1 变为-32768，标志位不动作，32 位运算时，如果 2147483647 加 1 变为-2147483648，标志位不动作；

4.4.6 BIN 减 1 [DEC]

	DEC	
D		P

16 位指令 DEC （连续执行型）
3 步　　DECP （脉冲执行型）

32 位指令 DDEC （连续执行型）
5 步　　DDECP （脉冲执行型）

适用软元件	• 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作

（D0）—1　　（D0）

- X001 每置“ON”一次，D0 的内容减 1，在连续执行指令中，每个扫描周期执行减 1 运算，所以务必引起注意；
- –32768 或-2147483648 减 1 变为 32767 或 2147483647，标志位不动作

4.4.7 逻辑与 [WAND]

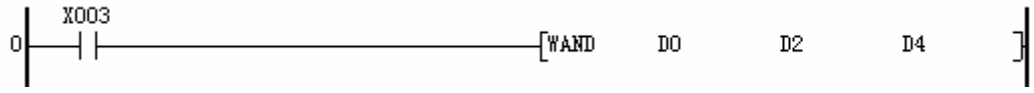
	WAND	
D		P

16 位指令 WAND （连续执行型）
7 步　　WANDP （脉冲执行型）

32 位指令 DWAND （连续执行型）
13 步 DWANDP （脉冲执行型）

功能和动作

《逻辑与》



（D0）∧（D2）⟶　（D4）

- 对各位进行逻辑与运算。

  1∧1=1　　0∧1=0

  1∧0=0　　0∧0=0

4.4.8 逻辑或 [WOR]

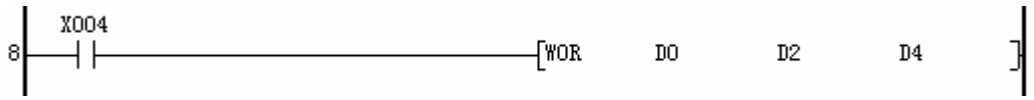
D	WOR	P

16 位指令 WOR （连续执行型）
7 步　　WORP （脉冲执行型）

32 位指令 DWOR （连续执行型）
13 步 DWORP （脉冲执行型）

功能和动作

《逻辑或》



（D0）∨（D2）⟶　（D4）

- 对各位进行逻辑或运算。

  1∨1=1　　0∨1=1

  1∨0=1　　0∨0=0

4.4.9 逻辑异或 [WXOR]

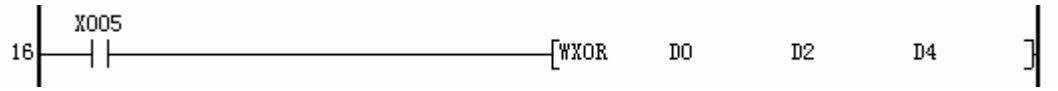
16 位指令 WXOR （连续执行型）
7 步　　WXORP （脉冲执行型）

32 位指令 DWXOR （连续执行型）
13 步 DWXORP （脉冲执行型）

适用软元件	•字软元件（S1• 、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z •字软元件（D•） KnY、KnM、KnS、T、C、D、V，Z

功能和动作

《逻辑异或》



（D0）（（D2）⟶（D4）

- 对各位进行逻辑异或运算；

  1（1=0　　0（1=1

  1（0=1　　0（0=0

- 如果将这个指令与 CML 组合使用，将进行异或非运算。

4.4.10 求补 [NEG]

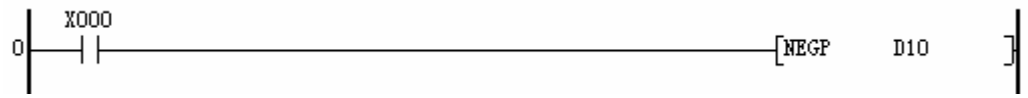
	NEG	
D		P

16 位指令 NEG （连续执行型）
3 步　　NEGP （脉冲执行型）

32 位指令 DNEG （连续执行型）
5 步 DNEGP （脉冲执行型）

适用软元件	•字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z

功能和动作



（$\overline{D10}$）+1 ⟶（D10）

- 将（D •）指定的软元件内容中各位先取反（0 变 1，1 变 0），然后再加 1，将其结果存入原先的软元件中；
- 使用连续执行指令则在每一个扫描周期执行一次，务必引起注意。

4.4.11 BIN 开方运算［SQR]

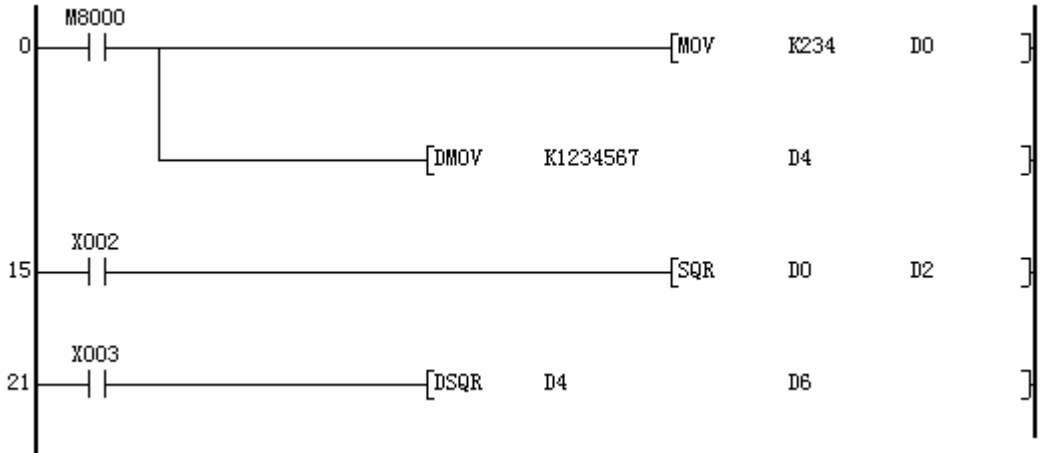
	SQR	
D		P

16 位指令 SQR （连续执行型）
5 步　　SQRP （脉冲执行型）

32 位指令 DSQR （连续执行型）
9 步　　DSQRP （脉冲执行型）

适用软元件	• 字软元件（S •） K、H、D • 字软元件（D •） D

功能和动作



- 进行开方运算的指令；
- 仅在（S •）≧0 时有效，如果（S •）<0，结果为 0；
- 运算结果舍去小数取整数；
- 无标志位。

4.5　循环与移位

4.5.1　循环右移［ROR]

	ROR	
D		P

16 位指令 ROR　（连续执行型）
5 步　　RORP　（脉冲执行型）

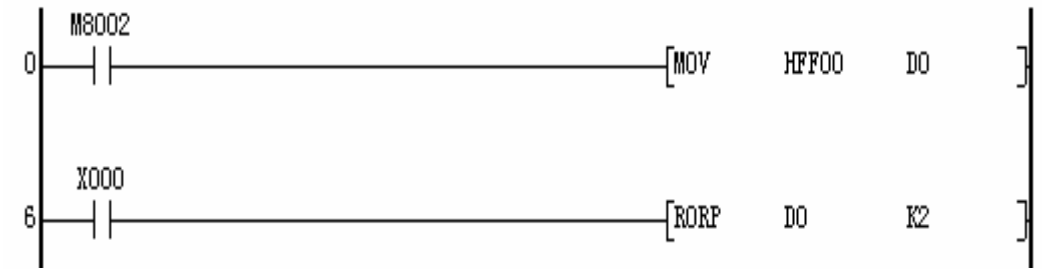
32 位指令 DROR　（连续执行型）
9 步　　DRORP　（脉冲执行型）

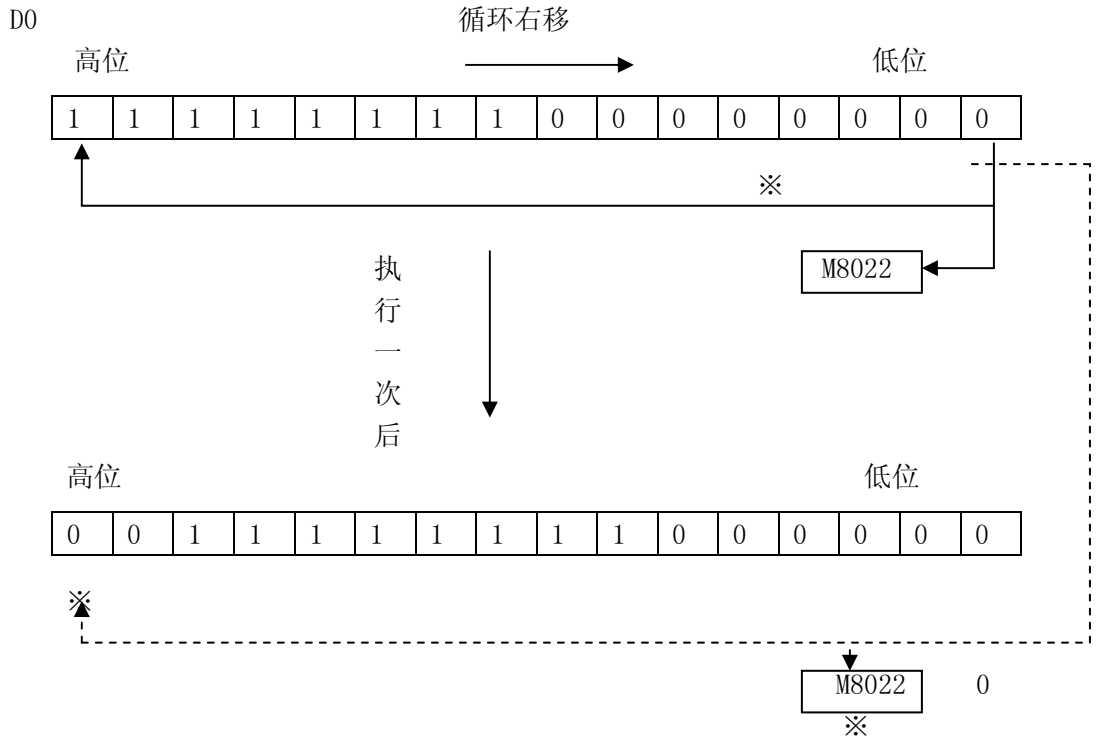
适用软元件	• 字软元件（D •）　KnY、KnM、KnS、T、C、D、V，Z • 字软元件（n）　K、H 回转量：n≦16（16 位指令） n≦32（32 位指令）

标志位		
	进位	M8022

功能和动作　　使 16 位或 32 位数据的各位右移位的指令。

《循环右移》





- X000 从“OFF”变为“ON”每变化一次，右移 2 位，最终位（※）被存入进位标志中。

4.5.2 循环左移 [ROL]

	ROL	
D		P

16 位指令 ROL （连续执行型）
5 步　　ROLP （脉冲执行型）

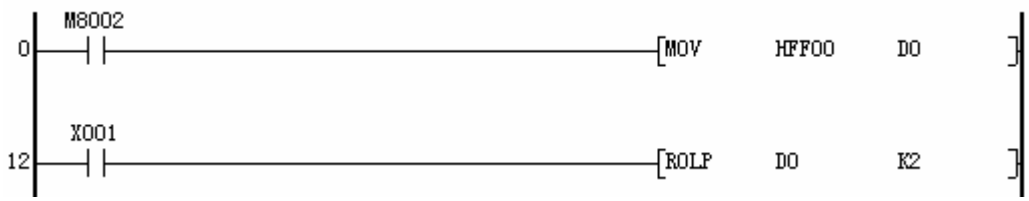
32 位指令 DROL （连续执行型）
9 步　　DROLP （脉冲执行型）

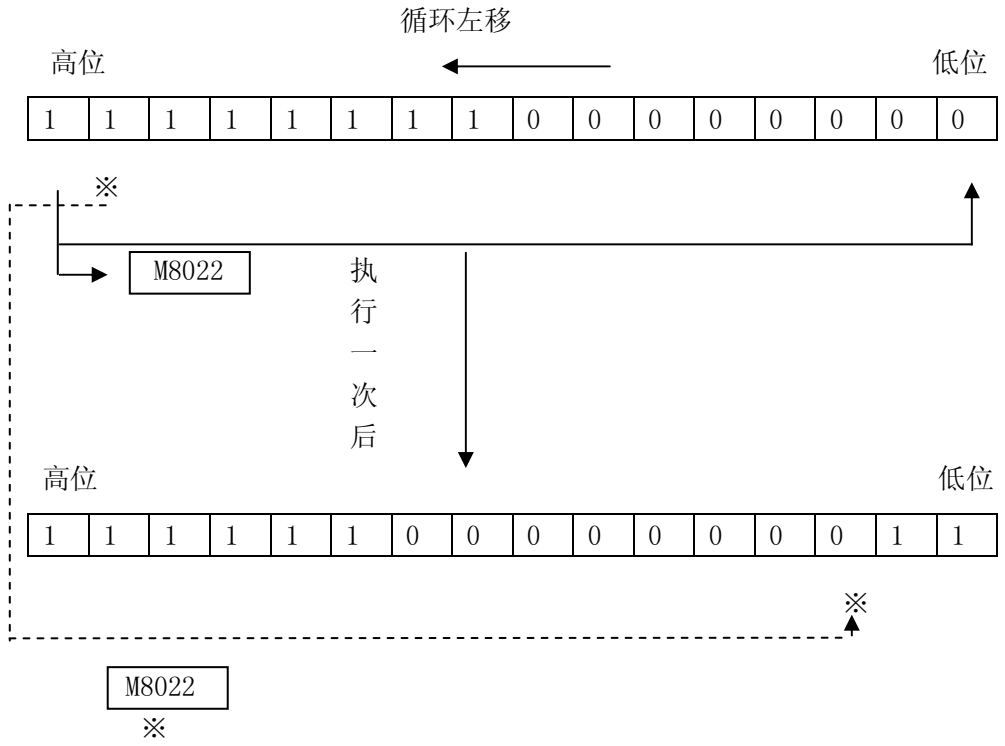
适用软元件	• 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z • 字软元件（n） K、H 回转量：n≦16（16 位指令） n≦32（32 位指令）

标志位		
	进位	M8022

功能和动作　　使 16 位或 32 位数据的各位左移位的指令。

《循环左移》





- X001 从“OFF”变为“ON”每变化一次，左移 2 位，最终位（※）被存入进位标志中。
- 连续执行指令每一个扫描周期进行一次移位，务必引起注意；
- 32 位指令的情况也一样；
- 在位指定软元件时，只有 K4（16 位）和 K8（32 位指令）是有效的（例如 K4Y0，K8M0）。

4.5.3 带进位循环右移［RCR]

	RCR	
D		P

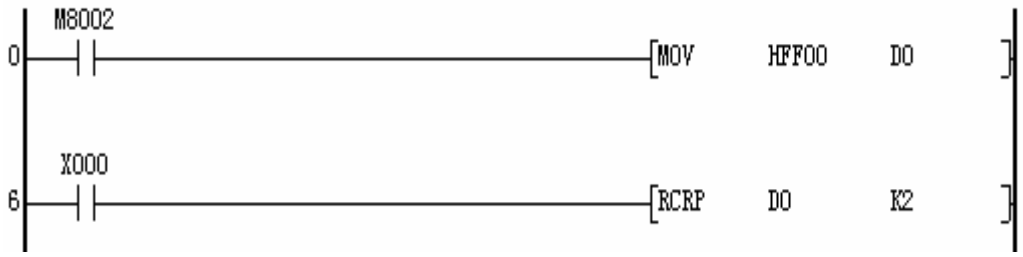
16 位指令 RCR （连续执行型）
5 步　　RCRP （脉冲执行型）

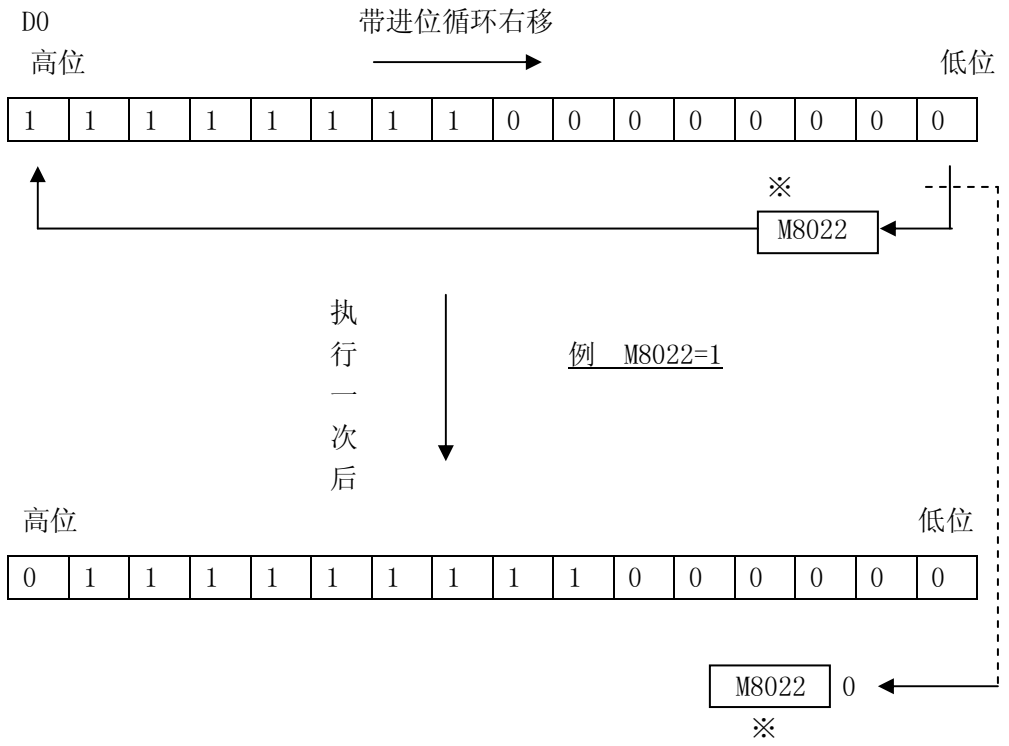
32 位指令 DRCR （连续执行型）
9 步　　DRCRP （脉冲执行型）

适用软元件	• 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z • 字软元件（n） K、H 回转量：n≦16（16 位指令） n≦32（32 位指令）

功能和动作　使 16 位或 32 位数据的各位带进位右移位的指令。

《带进位循环右移》





- X000 从“OFF”变为“ON”每变化一次，右移 2 位，最终位（※）移入进位标志中。

4.5.4 带进位循环左移 [RCL]

D	RCL	P

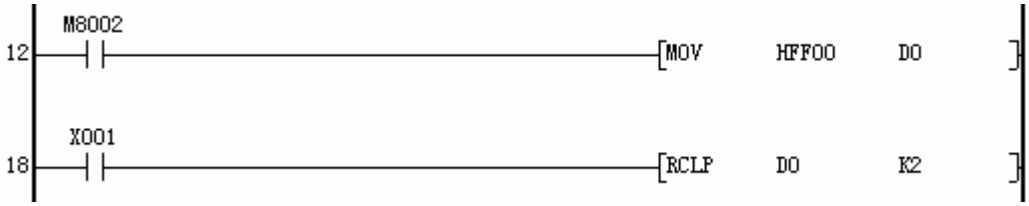
16 位指令 RCL （连续执行型）
5 步　　RCLP （脉冲执行型）

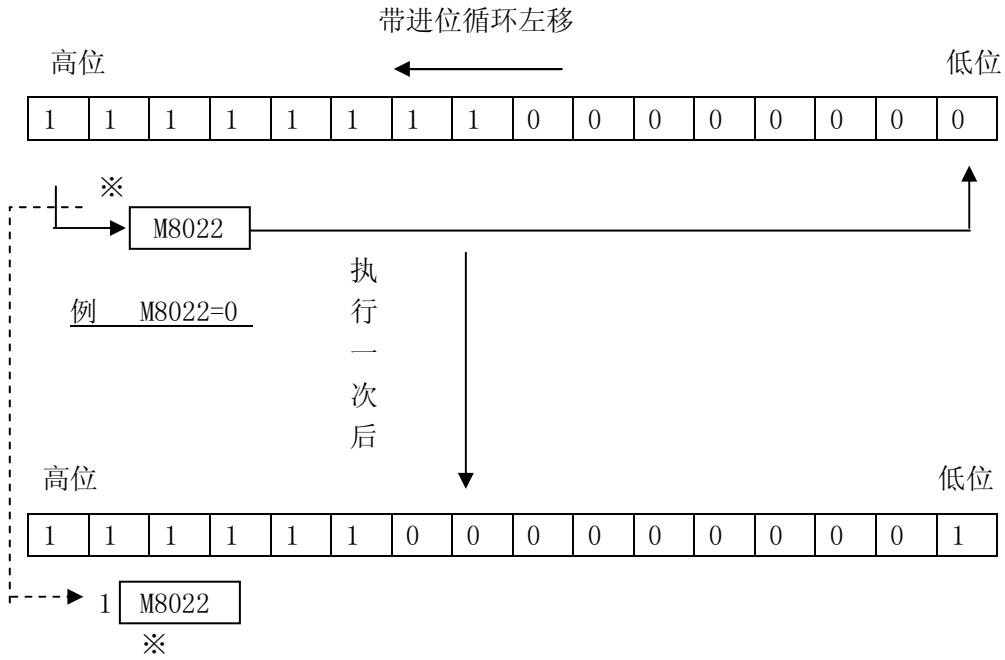
32 位指令 DRCL （连续执行型）
9 步　　DRCLP （脉冲执行型）

适用软元件	• 字软元件（D •） KnY、KnM、KnS、T、C、D、V，Z • 字软元件（n） K、H 回转量：n≦16（16 位指令） n≦32（32 位指令）

功能和动作　使 16 位或 32 位数据的各位带进位左移位的指令。

《带进位循环左移》





- X001 从“OFF”变为“ON”每变化一次，左移 2 位，最终位（※）移入进位标志中。
- 因为带进位循环移位中有进位标志，如果在执行前将驱动 M8022，可以将其送入目标地址中；
- 连续执行指令每一个扫描周期进行一次移位，务必引起注意；
- 32 位指令的情况也一样；
- 在位指定软元件时，只有 K4（16 位）和 K8（32 位指令）是有效的（例如 K4Y0，K8M0）。

4.6　浮点数运算

4.6.1　二进制浮点数比较 [DECMP]

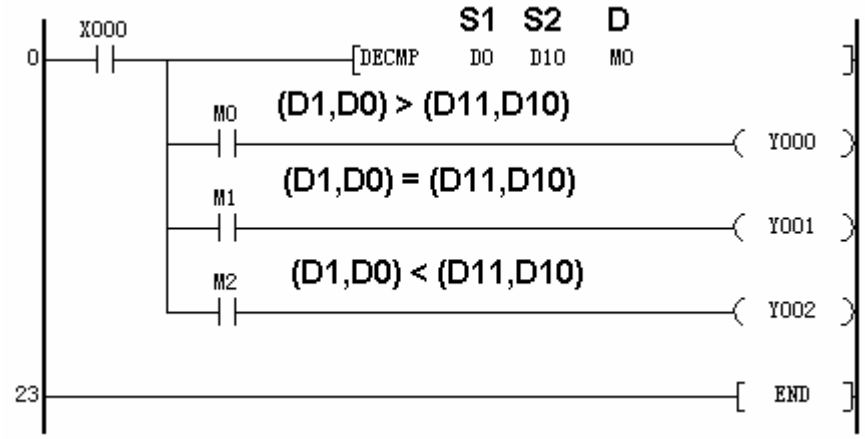
	DECMP	
		P

16 位指令　　无

32 位指令 DECMP　（连续执行型）
13 步　　DECMPP （脉冲执行型）

适用软元件	• 字软元件（S1 • 、S2 •） K、H、D • 位软元件（D •） Y、M、S（D • 占有连续的 3 点）

功能和动作



- 浮点数比较两个二进制浮点数的值，根据比较结果，对应输出 3 个位元件的 ON/OFF 状态，大于时第一个软元件闭合，等于时第二个闭合，小于时第三个闭合，如上图示例所示，但是当 X0 为 OFF 时，指令不执行。
- 源操作数为常数 K，H 时，自动转换为浮点数处理。
- 正常范围的比较：①、比较数据范围：$\pm 1*10^{37}$ ；②、比较数据可分辨范围：$246*10^{-34}$ ；同时符合①②项条件的两数据，可以正确比较。
- 非正常范围数据的处理办法：两数相差的绝对值小于 $246*10^{-34}$ 时，超出比较的

最小范围，因此有：$1000*10^{-34}=1200*10^{-34}$ ；$0*10^{0}=246*10^{-34}$ 。

4.6.2 二进制浮点数区域比较 [DEZCP]

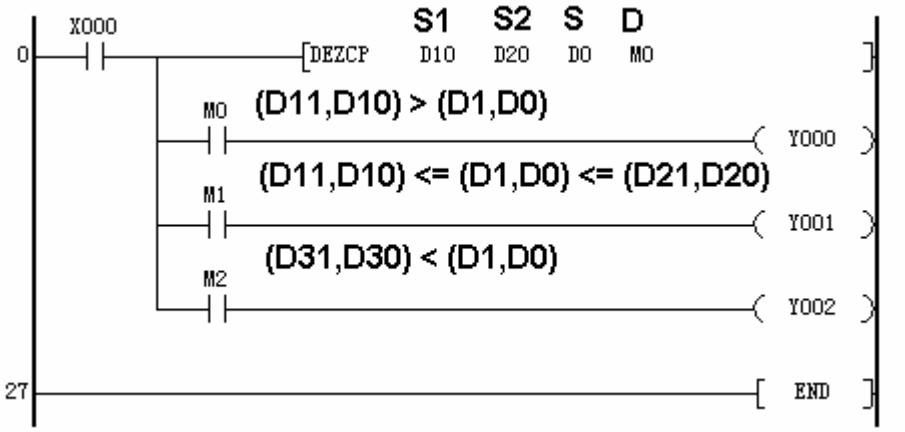
	DEZCP	
		P

16 位指令　　无

32 位指令 DZCP　（连续执行型）
17 步　 DZCPP　（脉冲执行型）

适用软元件	• 字软元件（S1 • 、S2 • 、 S1 •） K、H、D （S1 • ≤S2 •） • 位软元件（D •） Y、M、S（D • 占有连续的 3 点）

功能和动作



- 将 32 位的源操作数 S 与下限 S1 和上限 S2 进行范围比较，对应输出 3 个位元件的 ON/OFF 状态，如上例源操作数 S 小于区间时输出 M0，在区间内时输出 M1，大于区间时输出 M2。
- 源操作数为常数 K，H 时，自动转换为浮点数处理。
- 必须设置 S1 • ≤S2 • 当 S1 • ＞S2 • 时，则将 S1 • 和 S2 • 当作相同进行比较。
- 各个操作数的范围为 $1*10^{-37}$ 到 $1*10^{37}$。

4.6.3 二进制浮点数转十进制浮点数［DEBCD]

	DEBCD	
		P

16 位指令　　无

32 位指令 DEBCD　（连续执行型）
9 步　　DEBCDP　（脉冲执行型）

适用软元件	• 字软元件（S •）　D • 位软元件

功能和动作

- 该指令把二进制浮点数的源操作数转换为十进制浮点数的目标操作数。
- 浮点数的运算在嵌入式 PLC 内部是以二进制浮点数为基础执行的。
- 二进制浮点数数据格式：尾数部分 23 位，指数部分 8 位，符号位 1 位。
- 二进制浮点数所表示的范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。
- 十进制浮点数数据格式：尾数部分为低 16 位，指数部分为高 16 位。
- 十进制浮点数所表示的范围：尾数的输入范围任意，但是不能超过二进制浮点数所能表示的范围,即最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$，指数＝－37～＋37。

4.6.4 十进制浮点数转二进制浮点数［DEBIN]

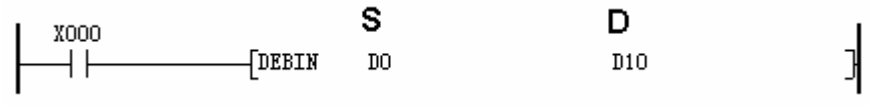
	DEBIN	
		P

16 位指令　　无

32 位指令 DEBIN　（连续执行型）
9 步　DEBINP　（脉冲执行型）

适用软元件	• 字软元件（S •） D • 位软元件

功能和动作



- 该指令把十进制浮点数的源操作数转换为二进制浮点数的目标操作数。
- 二进制浮点数和十进制浮点数的表示范围见 DEBCD 指令的说明。注意指数的输入范围为-37～＋37，尾数的输入范围任意，但是不能超过二进制浮点数所能表示的范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。

4.6.5 二进制浮点数加法［DEADD]

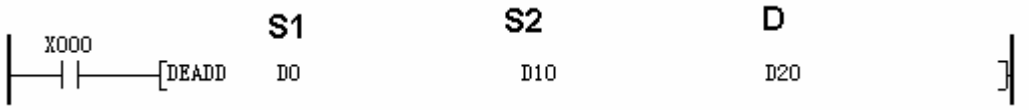
	DEADD	
		P

16 位指令　　无

32 位指令 DEADD　（连续执行型）
13 步　DEADDP　（脉冲执行型）

适用软元件	• 字软元件（S1 • S2 •） K、H、D (D • ） D • 位软元件

功能和动作



- 两个二进制浮点数源数据相加后，存入目的地址中。
- 源操作数为常数 K，H 时，自动转换为浮点数处理。
- 源数据和目的地址可以为指定的同一元件号。
- 正确运算的数据范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。不论是源操作数还是目的操作数和中间结果都不能超过此范围，否则导致运算结果不正确。

4.6.6 二进制浮点数减法［DESUB]

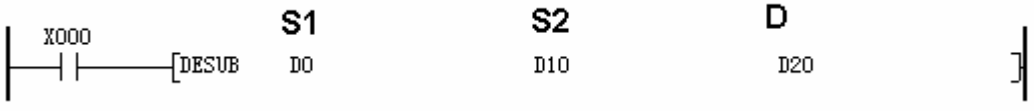
	DESUB	
		P

16 位指令　　无

32 位指令 DESUB　（连续执行型）
13 步　　DESUBP　（脉冲执行型）

适用软元件	・字软元件（S1・S2・）　K、H、D (D・)　D ・位软元件

功能和动作



- 两个二进制浮点数源数据相减后，存入目的地址中。
- 源操作数为常数 K，H 时，自动转换为浮点数处理。
- 源数据和目的地址可以为指定的同一元件号。
- 正确运算的数据范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。不论是源操作

数还是目的操作数和中间结果都不能超过此范围，否则导致运算结果不正确。

4.6.7 二进制浮点数乘法［DEMUL]

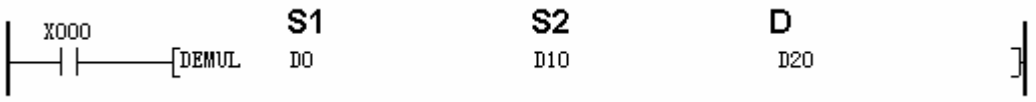
	DEMUL	
		P

16 位指令　　无

32 位指令 DEMUL　（连续执行型）
13 步　　DEMULP　（脉冲执行型）

适用软元件	・字软元件（S1・S2・）　K、H、D (D・)　D ・位软元件

功能和动作



- 两个二进制浮点数源操作数的乘积作为二进制浮点数存入目的地址中。
- 将常数 K、H 作为源操作数时，自动转换为二进制浮点数处理。
- 正确运算的数据范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。不论是源操作数还是目的操作数和中间结果都不能超过此范围，否则导致运算结果不正确。因此也不存在零的二进制浮点数。

4.6.8 二进制浮点数除法［DEDIV]

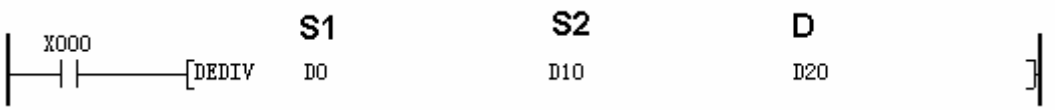
	DEDIV	
		P

16 位指令　　无

32 位指令 DEDIV　（连续执行型）
13 步　 DEDIVP　（脉冲执行型）

适用软元件	• 字软元件（S1 • S2 •） K、H、D (D • ） D • 位软元件

功能和动作



- 两个二进制浮点数源操作数相除的结果作为二进制浮点数存入目的地址中。
- 将常数 K、H 作为源操作数时，自动转换为二进制浮点数处理。
- 正确运算的数据范围：最小绝对值 $1*10^{-37}$，最大绝对值 $1*10^{37}$。不论是源操作数还是目的操作数和中间结果都不能超过此范围，否则导致运算结果不正确。因此也不存在零的二进制浮点数，当除数 S2 为 0 时，则运算结果错误。

4.6.9 二进制浮点数开方［DESQR]

	DESQR	
		P

16 位指令　　无

32 位指令 DESQR　（连续执行型）
9 步　　DESQRP　（脉冲执行型）

适用软元件	• 字软元件（S •） K、H、D (D • ） D • 位软元件

功能和动作

- 将二进制浮点数源操作数开平方的结果作为二进制浮点数存入目的地址中。
- 将常数 K、H 作为源操作数时，自动转换为二进制浮点数处理。
- 正确运算的数据范围：最小绝对值 $3.16*10^{-30}$，最大绝对值 $1*10^{37}$。当不在此范围内的二进制浮点数运算结果不正确。

4.6.10 二进制浮点数转 BIN 整数变换［INT]

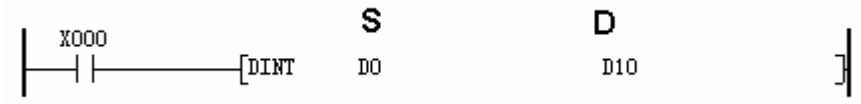
	INT	
D		P

16 位指令 INT（连续执行型）
5 程序步 INTP（脉冲执行型）

32 位指令 DINT （连续执行型）
9 步 DINTP （脉冲执行型）

适用软元件	• 字软元件（S • D •）D • 位软元件

功能和动作



- 将二进制浮点数源操作数取整后的结果作为 BIN 整数存入目的地址中，舍去小数点后面的值。
- 正确运算的数据范围：16 位指令对在位于－32768 到 32767 间的任意浮点数，能正确取整；32 位指令对在位于－2147483648 到 2147483647 间的任意浮点数，能正确取整。
- 该指令是指令 FLT 的逆变换。

4.6.11 BIN 整数转二进制浮点数［FLT]

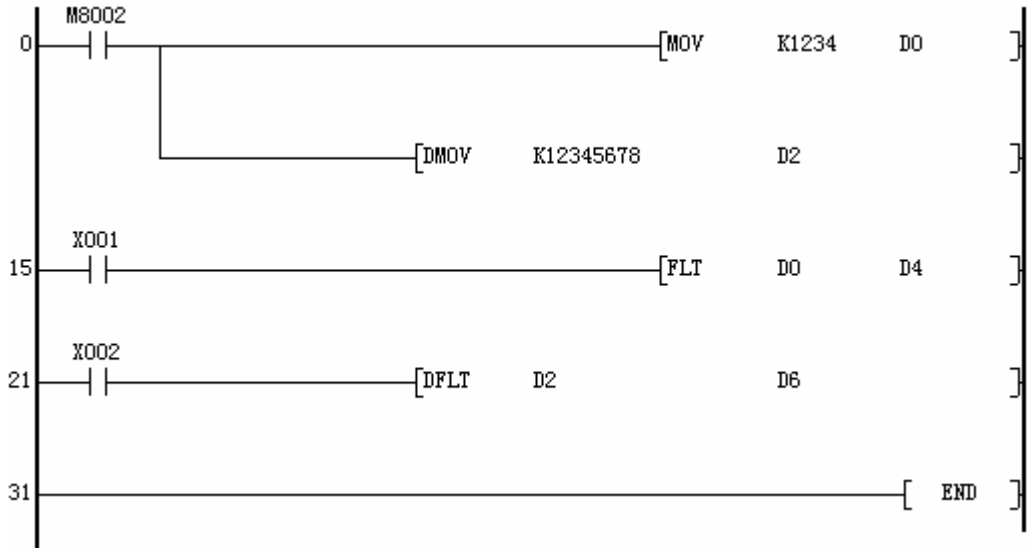
	FLT	
D		P

16 位指令 FLT （连续执行型）
FLTP （脉冲执行型）

32 位指令 DFLT （连续执行型）
9 步 DFLTP （脉冲执行型）

适用软元件	• 字软元件（S •） D • 字软元件（D •） D

功能和动作



（D0） ⟶ （D5，D4）
BIN 整数　　2 进制浮点数

（D2） ⟶ （D7，D6）
BIN 整数　　2 进制浮点数

- BIN 整数转 2 进制浮点数的指令，常数 K、H 在各浮点运算中被自动转换，因此在 FLT 指令中不能使用；
- FLT 指令的逆变换指令是 INT 指令。

4.7　触点比较指令

4.7.1 接点比较指令［LD※］

D	LD※	

※ 表示：=、>、<、<>、≦、≧ 。

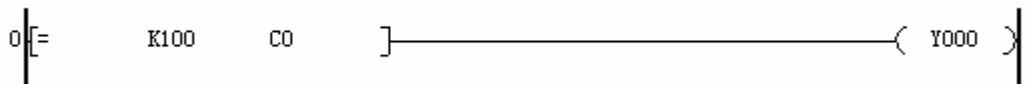
16 位指令 LD※（连续执行型）
5 步

32 位指令 LDD※　（连续执行型）
9 步

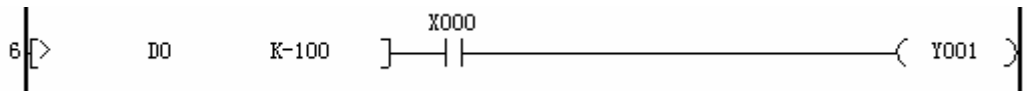
适用软元件	•字软元件（S1• 、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z

指令形式与功能　对源数据进行 BIN 比较，对应其结果执行后段的运算

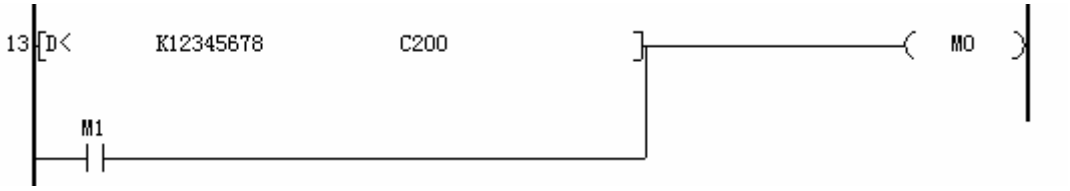
16 指令	32 位指令	导通条件	非导通条件
LD=	LDD=	(S1•) = (S2•)	(S1•) ≠ (S2•)
LD>	LDD>	(S1•) > (S2•)	(S1•) ≦ (S2•)
LD<	LDD<	(S1•) < (S2•)	(S1•) ≧ (S2•)
LD<>	LDD<>	(S1•) ≠ (S2•)	(S1•) = (S2•)
LD≦	LDD≦	(S1•) ≦ (S2•)	(S1•) > (S2•)
LD≧	LDD≧	(S1•) ≧ (S2•)	(S1•) < (S2•)



- 当计数器 C0 的当前值为 100 时，驱动。



- 当 D0 的内容大于-100，且 X000 处于“ON”时，驱动 Y1。



- 当计数器 C200 的内容大于 12345678，或者 M1 处于“ON”时，驱动 M0。

注意事项

- 当源数据的最高位（16 位指令：b15，32 位指令：b31）为 1 时，将该数值作为负数进行比较；
- 32 计数器（C200-C255）的比较，必须以 32 位指令来进行。

4.7.2 接点比较指令［AND※］

D	AND※	

※表示 ：=、>、<、<>、≦、≧ 。

16 位指令 AND※（连续执行型）
5 步

32 位指令 ANDD※ （连续执行型）
9 步

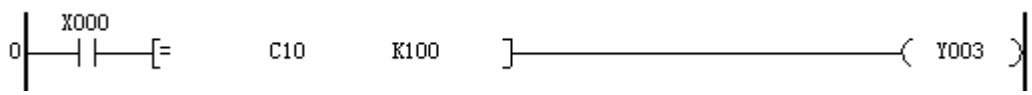
适用软元件	•字软元件（S1• 、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z

指令形式与功能

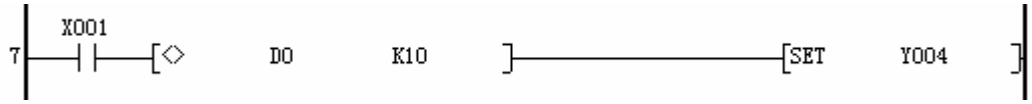
对源数据进行 BIN 比较，对应其结果执行后段的运算。

16 指令	32 位指令	导通条件	非导通条件
AND=	ANDD=	（S1 •）= （S2 •）	（S1 •）≠（S2 •）

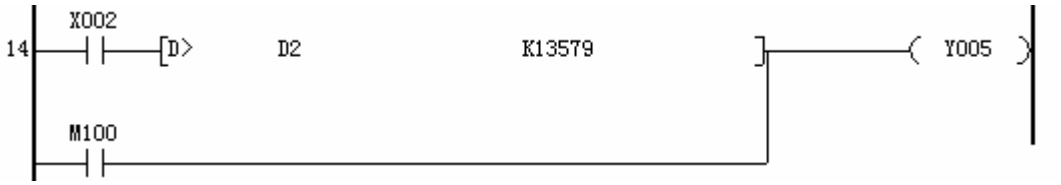
AND>	ANDD>	（S1 •）>（S2 •）	（S1 •）≦（S2 •）
AND<	ANDD<	（S1 •）<（S2 •）	（S1 •）≧（S2 •）
AND<>	ANDD<>	（S1 •）≠（S2 •）	（S1 •）=（S2 •）
AND≦	ANDD≦	（S1 •）≦（S2 •）	（S1 •）>（S2 •）
AND≧	ANDD≧	（S1 •）≧（S2 •）	（S1 •）<（S2 •）



- 当 X000 处于“ON”时，且计数器 C10 的当前值等于 100 时，驱动 Y3。



- 当 X001 处于“ON”时，且 D0 的内容不等于 10 时，置位 Y4。



- 当 X002 处于“ON”，且（D3，D2）的内容大于 13579 时，或者 M100 处于“ON”时，驱动 Y5。

注意事项

- 当源数据的最高位（16 位指令：b15，32 位指令：b31）为 1 时，将该数值作为负数进行比较；
- 32 计数器（C200-C255）的比较，必须以 32 位指令来进行。

4.7.3 接点比较指令［OR※］

D	OR※	

※表示 ：=、>、<、<>、≦、≧ 。

16 位指令 OR※（连续执行型）
5 步

32 位指令 ORD※　（连续执行型）
9 步

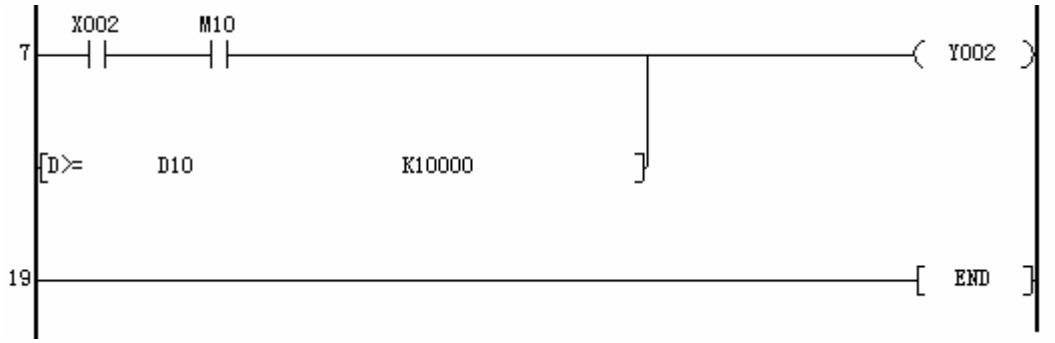
适用软元件	•字软元件（S1• 、S2•） K、H、KnX、KnY、KnM、KnS、T、C、D、V，Z

指令形式与功能

对源数据进行 BIN 比较，对应其结果执行后段的运算。

16 指令	32 位指令	导通条件	非导通条件
OR=	ORD=	(S1•) = (S2•)	(S1•) ≠ (S2•)
OR>	ORD>	(S1•) > (S2•)	(S1•) ≦ (S2•)
OR<	ORD<	(S1•) < (S2•)	(S1•) ≧ (S2•)
OR<>	ORD<>	(S1•) ≠ (S2•)	(S1•) = (S2•)
OR≦	ORD≦	(S1•) ≦ (S2•)	(S1•) > (S2•)
OR≧	ORD≧	(S1•) ≧ (S2•)	(S1•) < (S2•)

- 当 X001 处于“ON”，或计数器 C10 的当前值等于 100 时，驱动 Y1



- 当 X002 和 M10 处于“ON”时，或者（D11，D10）的内容大于等于 10000 时，驱动 Y2。

注意事项

- 当源数据的最高位（16 位指令：b15，32 位指令：b31）为 1 时，将该数值作为负数进行比较；
- 32 计数器（C200-C255）的比较，必须以 32 位指令来进行。

4.8　功能指令的基本规则

本节叙述可编程控制器功能指令的表示方法与基本规则。在使用功能指令编程时，需要大致了解指令中有关软元件的使用及其执行形式。

4.8.1. 功能指令的表示与执行形式

1、指令与操作数

①、功能指令用助记符表示。

②、有些功能指令仅有指令段（助记符），但更多的有操作数。

③、指令中的操作数符号表示方法及解释。

[S]：表示数据源。内容不随指令执行而变化的操作数称为源。

在可变址修改软元件编号的情况下，加上“ •”符号的[S •]表示。

源的数量多时，以[S_1 •][S_2 •]等表示。

[D]：表示目标操作数。内容随指令执行而改变的操作数被称作目标。

可作变址修饰时，加上“ •”符号的[D •]表示.

在目标数量多时，以[D_1 •]、[D_2 •]等表示。

[n •] [m •]：以[m •]或[n •]表示既不做源，也不做目标的操作数。

这样的操作数数量很多时，以[m1 •]、[m2 •]、[n1 •] [n2 •]等表示。

2、可用作操作数的软元件

①、X，Y，M，S 等位元件。

②、位元件组合。以 KnX，KnY，KnM，KnS 等形式表示，作为数值处理。

③、数据寄存器 D、定时器 T 的当前值寄存器、计数器 C 的当前值寄存器。

- 数据寄存器 D 为 16 位，在处理 32 位数据时使用一对数据寄存器的组合。

例如，将数据寄存器 D0 指定为 32 位指令的操作数时，处理(D1，D0)32 位数据(D1 为高 16 位，D0 为低 16 位)。

- T、C 的当前值寄存器也可作为一般寄存器处理。

- C200—C255 为 32 位计数器，处理 32 位的数据，不能作 16 位指令的操作数。

3、指令的形态与执行形式

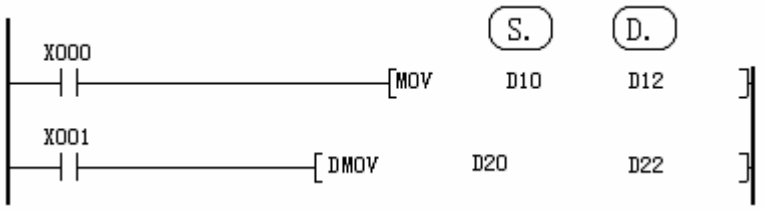
根据功能指令处理数值的大小，分为《16 位指令》和《32 位指令》。

根据功能指令的执行形式，分为《连续执行型》与《脉冲执行型》。

功能指令可将这些形式组合使用或单独使用。

①、16 位指令和 32 位指令

- 在数值处理的功能指令中，根据数值数据的位长分为 16 位与 32 位。

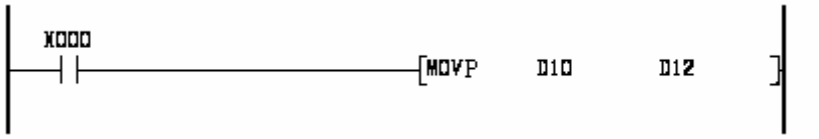


16 位指令：通过 MOV 将 D10 的内容传送到 D12 中的指令。
32 位指令：通过 DMOV 将(D21，D20)的内容传送到(D23，D22)中。

- 32 位计数器(C200—C255)的一个软元件为 32 位，不可用作 16 位指令的操作数。

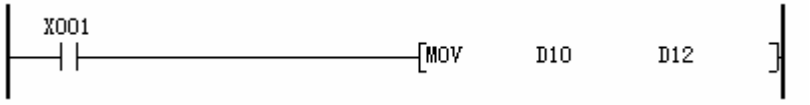
②、脉冲执行形式和连续执行形式

- 脉冲执行型



如图所示，在 X000 从 OFF→ON 变化时，指令执行一次。
指令在不执行时的处理时间快，建议尽量采用脉冲执行型指令。
符号 P 表示脉冲执行型命令。

- 连续执行型



图为连续执行型指令，X001 接通时，每个扫描周期都执行。

- 1NC、DEC 等指令根据程序要求的内容而采取不同的执行型式。

  如果采用连续执行型指令，则每个扫描周期，其操作数的内容都发生变化。这种指令采用连续形式指令时，必须注意。

- 在功能指令解说时，使用下图符号以示区别。

←使用连续执行命令时，每一扫描周期“源”的内容都发生变化。

INCREMENT

4、标志位的处理

①、一般标志

根据功能指令的种类，有下述标志动作。

M8020：零标识

M8021：借位标识

M8022：进位标识

这些标志在每次各种指令为 ON 时，出现接通或断开动作，但是在 OFF 时，或出现错误时不变化。

4.8.2　功能指令内的数值处理

1、位元件的处理

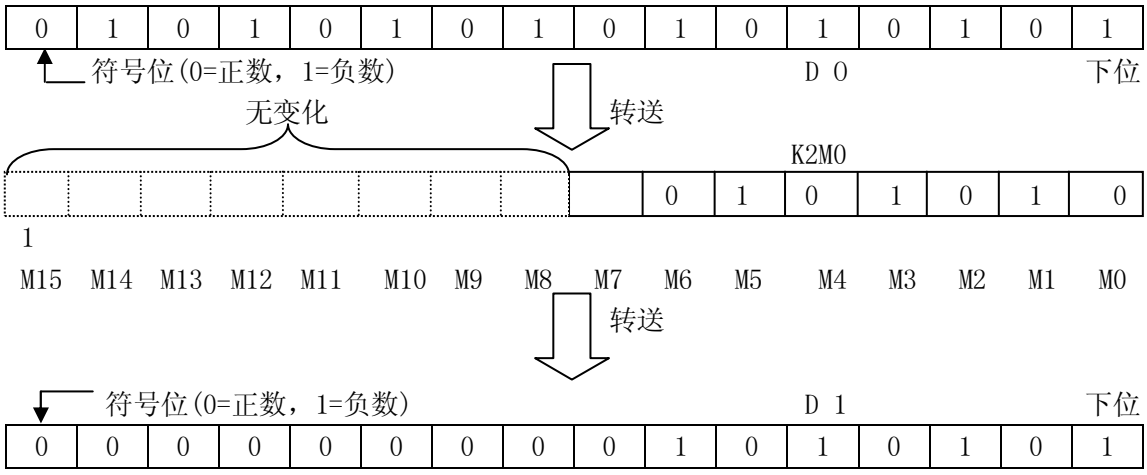
①、位元件：象 X，Y，M，S 等只处理 ON/OFF 信息的软元件

字元件：象 T，C，D 等处理数值的软元件。

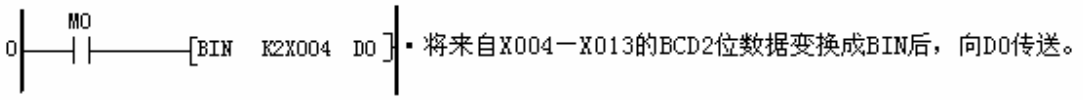
位元件组合使用也可处理数值：以位数 Kn 和起始的软元件号的组合来表示。

采用 4 位为单位；16 位数据用 K1—K4 表示；32 位数据用 K1—K8 表示。

例如，K2M0，由于是 M0—M7，为 2 单位数据。



- 若向 K1M0-K3M0 传送 16 位数据，则数据长度不足的高位部分不被传送。32 位数据亦同样。
- 在 16 位(或 32 位)运算中，对应位元件的位指定是 K1-K3(或 K1-K7)时，长度不足的高位通常被视为 0。因此，通常将其作为正数处理。



- 被指定的位元件编号，一般可自由指定。建议：

在 X，Y 的场合，最低位的编号尽可能设定为 0，如 X000，X010，X020…Y000，Y010，Y020…。

在 M，S 的场合，最低位的编号设定为 8 的倍数，或设定为 0，如 M0，M10 等。

②、连续字的指定

- 所谓以 D1 为开头的一系列数据寄存器就是 D1，D2，D3，D4…等。
- 通过位指定，在字的场合，也可将其作为一系列的字处理。如下所示。

  K1X000　K1X004　K1X010　K1X014…，

  K2Y010　K2Y020　Y2X030…

  K3M0　K3M12　K3M24　K3M36…

  K4S16　K4S32　K4S48…

  也就是说，按照各位的进制且不跳过软元件，连续使用软元件。
- 在 32 位运算中采用 K4Y000 时，则将高 16 位看作 0。在需要 32 位数据时，要用 K8Y000。

2、 浮点运算的数值处理

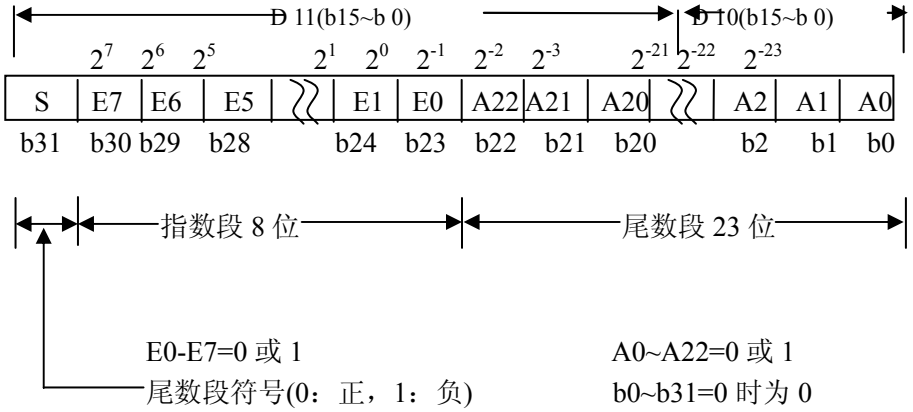
在可编程控制器中，常采用浮点数运算。

①、10 进制浮点数

- 二进制浮点数是用户难于判断的数值，但可编程运算采用二进制浮点数。
- 十进制浮点数为用户所接受，但可编程不能直接运算，因此二者必须相互转换。
- 十进制浮点数表示方法：利用编号连续的一对数据寄存器表示十进制浮点值，编号小的一侧为尾数段，编号大的一侧为指数段。

②、二进制浮点数

二进制浮点值也采用编号连续的一对数据寄存器。例如(D11，D10)的场合，其具体含义如下：



2 进制浮动值=$\pm(2^0+A22\times2^{-1}+A21\times2^{-2}+\cdots\cdots+A0\times2^{-23})$
$\times2$（$E7\times2^7+E6\times2^6+\cdots\cdots+E0\times2^0$）$/2^{127}$

（例）　A22＝1，A21＝0，A20＝1，A19~A0＝0
E7＝1，E6~E1=0，E0=1

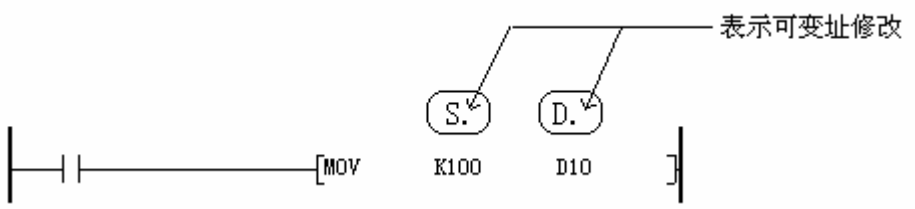
2 进制浮动值=$\pm(2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}+\cdots\cdots+0\times2^{-23})$
$\times2$（$1\times2^7+0\times2^6+\cdots\cdots+1\times2^0$）$/2^{127}$
$=\pm1.625\times2^{129}/2^{127}=\pm1.625\times2^2$

正负是由 b31 的符号决定的，不是补码处理。

4.8.3　利用变址寄存器的操作数修改

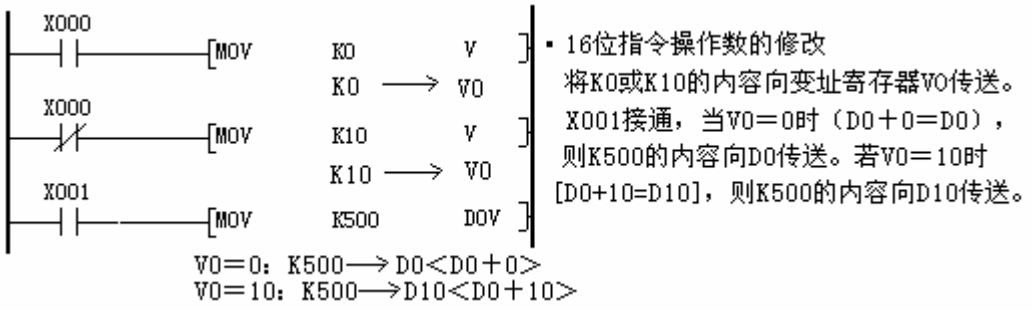
1、 操作数变址表示方法

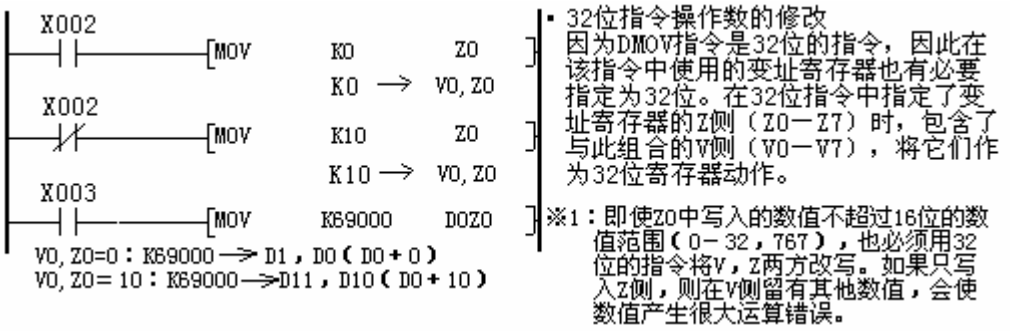
在功能指令的说明中，表示源 S 或目标 D 的符号中加[・]标记，以示操作数可变址修饰。



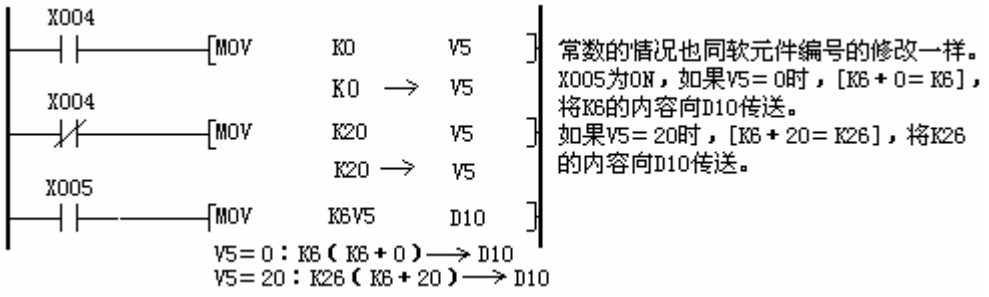
2、变址修改示例

①、数据寄存器编号的修改

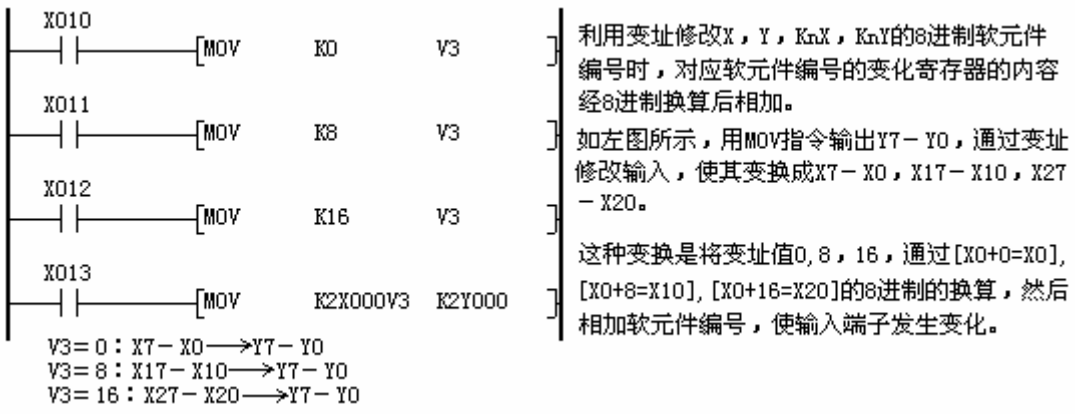




②、常数 K 的修改



③、输入输出继电器(8 进制软元件编号)的修改



3、注意事项

- 利用变址修改的 16 计数器不能作为 32 位计数器使用。作为变址修改的结果，需要使用 32 位计数器的场合，请在计数器 C200 以后附加上 Z0-Z7。
- V，Z 自身或位指定用 Kn 的“n”不可修改。(K4M0Z0 有效，K0Z0M0 无效)
- LD，AND，OUT 等可编程控制器的基本顺序指令和步进梯形图指令不可变址修改。

第五章　资源说明及应用

鉴于已全面学习了指令集的内容，因而在资源应用上，不受指令限制。

5.1 变址寄存器 V、Z 说明及应用

5.1.1 变址寄存器 V、Z 说明

V、Z 的显著作用是能够和其它软元件或数值组合使用，从而动态修改软元件编号或数值内容。

变址寄存器共 16 个：V0----V7；Z0---Z7。

每个变址寄存器都是 16 bit 数据寄存器，可作普通数据寄存器使用。

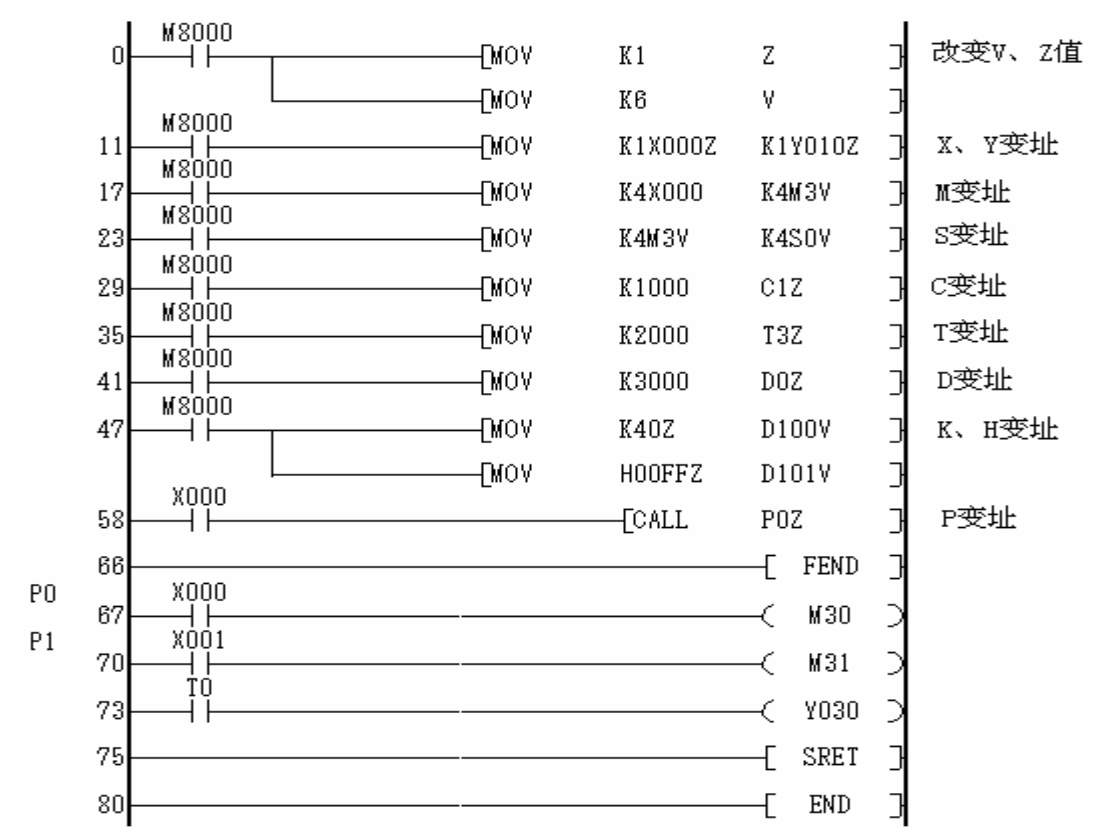
用它们组合成 32 bit 数据寄存器，必须同序号 V、Z 配对，Z 为低字，V 为高字。可配成 8 个 32bit 数据寄存器：Z0（V0）------Z7（V7）。

V、Z 变址功能不能用于基本指令（如 LD、AND、OUT 等）、步进阶梯指令（STL）。

V、Z 变址功能主要用在功能指令中，灵活改变资源的编号。

5.1.2 变址寄存器在梯形图中的应用

①、各种资源的变址访问

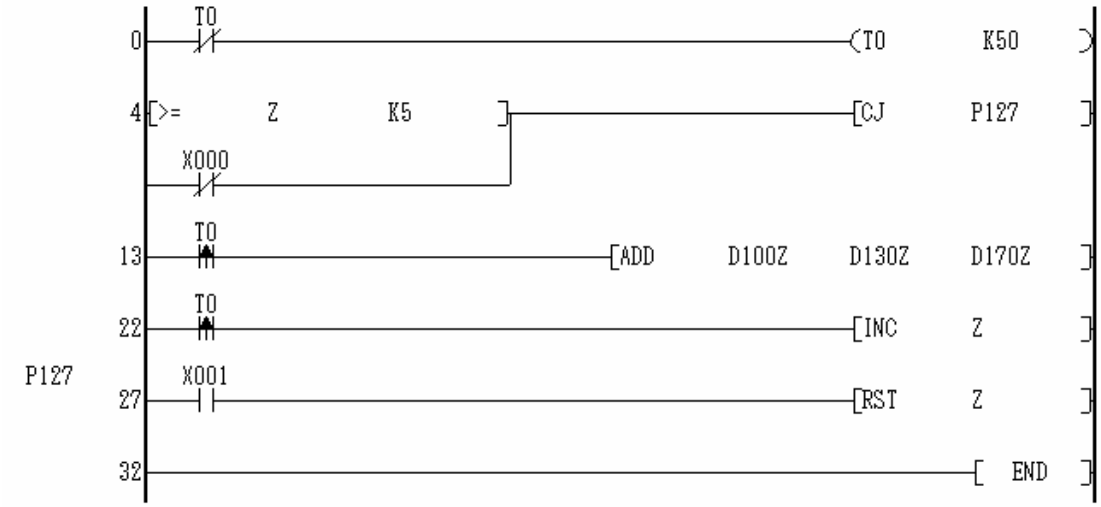


修改说明：Z=1，V=6。

```
MOV    K1X000Z     K1Y010Z 等同于  MOV    K1X001      K1Y011  ；
MOV    K4X000      K4M3V 等同于    MOV    K4X000      K4M9    ；
MOV    K4M3V       K4S0V 等同于    MOV    K4M9        K4S6    ；
MOV    K1000       C1Z 等同于  MOV    K10000      C2  ；
MOV    K2000       T3Z 等同于  MOV    K20000      T4  ；
MOV    K3000       D0Z 等同于  MOV    K30000      D1  ；
MOV    K40Z    D100V  等同于  MOV    K41     D106    ；
MOV    H00FF   D101V  等同于  MOV    H0100   D107    ；
CALL   P0Z      等同于  CALL   P1  ；
```

②、使用变址功能示例

示例说明：一个简易运算程序，将 D100-----D104 的 5 个整数与 D130----D134 的 5 个整数对应相加，存放在 D170---D174 的寄存器中。为观察运算结果，控制 5 秒运行一次，并设重新运算键 X1，允许运算键 X0。该程序使用变址寄存器 Z，从而使程序简化。



5.1.3　使用变址功能的注意事项

①、正确计算变址寄存器的取值范围

变址寄存器理论取值范围：

16bit：-32768------+32767；

32bit：-2147483648------+2147483647；作普通 32bit 数据寄存器使用。

变址寄存器实际取值范围：

作 16bit 使用时，才有变址功能。

当 V、Z 与其它资源组合，并修改其它资源编号时，实际取值不得突破所修饰资源的编号范围。否则，程序在运行过程中找不到资源而发生错误。

如，指针变址 P10Z（设 Z=-3，则 P10Z 等同于 P7），如程序无对应的指针标号（P7），程序不能定位到正确位置而导致错误。

正确计算不同情况下变址寄存器的取值范围，避免取值不当而导致程序错误。

如，上例中指令 ADD　D100Z　D130Z　　D170Z；

Z 最小保证　D100Z=D0，Z=-100　；

Z 最大保证　D170Z=D5999，Z=5829 ；

如，程序中只有指针标号 P0、P3、P66，则指令 CJ P3Z 中，Z 只能取-3、0、63 三个值。Z 取其它值时（Z=3），程序发生错误后，即使 Z 再取正确值（Z=-3），　程序仍不能恢复。需重新 STOP→RUN 或重新上电。

②、变址功能不能应用于基本顺控指令及步进阶梯指令 STL 中。

如：基本指令 LD C1Z ，OUT C0Z ；

步进指令 STL S0Z ；

都是错误地应用了变址寄存器。

但在 STL S0 的状态步序中，可以应用功能指令编程，当然也可以使用变址组合访问。

③、16bit 计数器 32bit 计数器不能作为同一组设备变址。

16bit 计数器变址编号组合值应在 0----199 范围内；

32bit 计数器变址编号组合值应在 200-----255 范围内。

如果 16bit 计数器变址编号到 32bit 或 32bit 计数器变址编号到 16bit 时，程序在运行过程中作越界处理，中止运行。

如　C100Z，Z 取值应在-100-------+99 范围。

C233V，V 取值应在-33------+22 范围。

④、变址寄存器本身不能变址。

如 MOV　　K2　　V0Z　不被梯形图认可。

⑤、位元件组合成字元件 K_nM0 型的下标 n 不能变址。

如 MOV　　K2　　K1VM10　不被梯形图认可。

5.2　输入输出继电器 X、Y 说明及应用

内部软元件是具有明确含义的存储器单元，可供 CPU 快速访问。但作为控制器又必需与外部设备打交道，因此必需将一些软元件对应到外部硬件接口上。我们把接口规划为以下三类：

开关量输入输出型。

模拟量输入输出型。

通讯输入输出型。

本节就专供开关量输入映射软元件 X，开关量输出映射软元件 Y 作详细说明

5.2.1　输入输出继电器 X、Y 说明

①、X、Y 编号及数量

X：8 进制编号，范围：X000-----X177；数量：128 点。

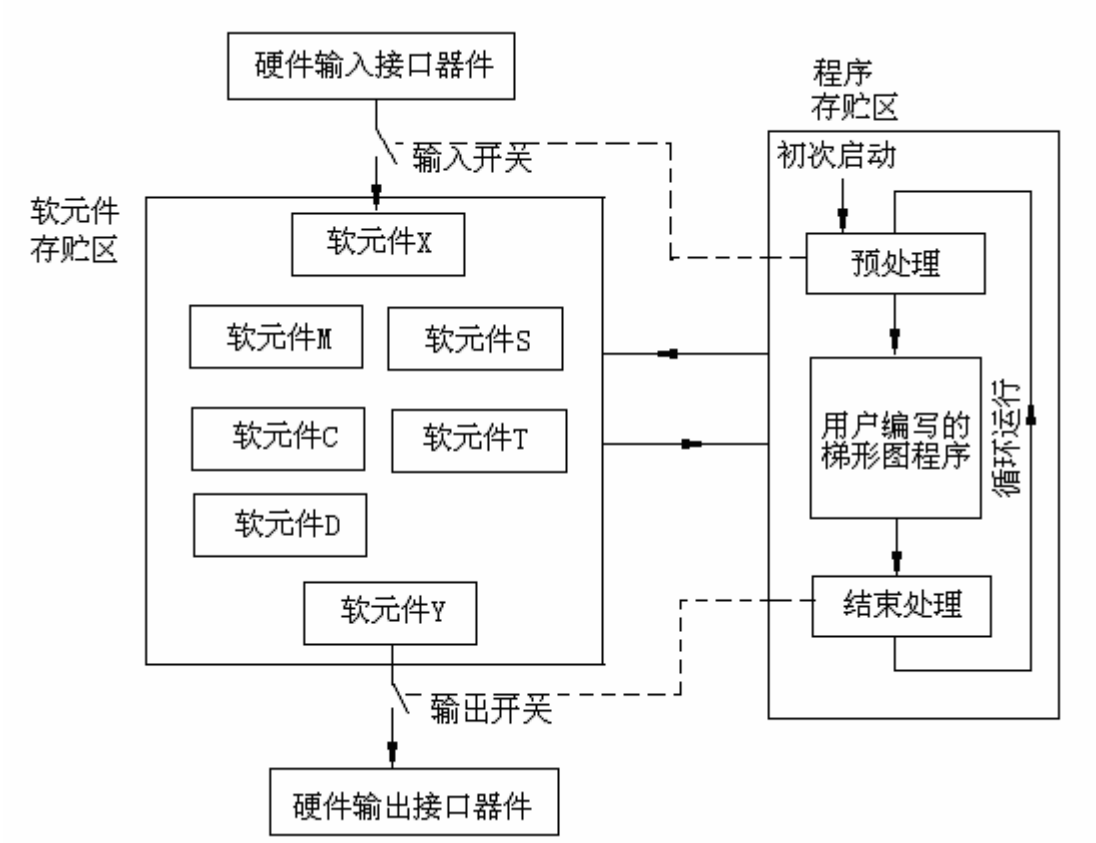
Y：8 进制编号，范围：Y000-----Y177；数量：128 点。

具体嵌入式 PLC 产品 X、Y 范围参看硬件手册。

②、PLC 一般程序流程：

看图说明。

- 预处理：在进入用户程序之前，进行必要的信息处理，由嵌入式 PLC 系统自动完成。控制输入开关，读取输入信号是预处理任务之一。所读取的信号是输入开关合上瞬间外界信号的状态，在开关合上以外的状态不被读入。
- 用户程序处理：控制器按用户所编写的梯形图程序读写软元件的处理过程。
- 结束处理：在用户程序结束后所进行的信息处理，由嵌入式 PLC 系统自动完成。控制输出开关，将程序运行结果输出到外部接口是结束处理的任务之一。程序的中间处理结果并不直接对外输出。



扫描周期：把程序从预处理开始经程序处理、结束处理后，回到预处理起点的时间，称一个扫描周期。

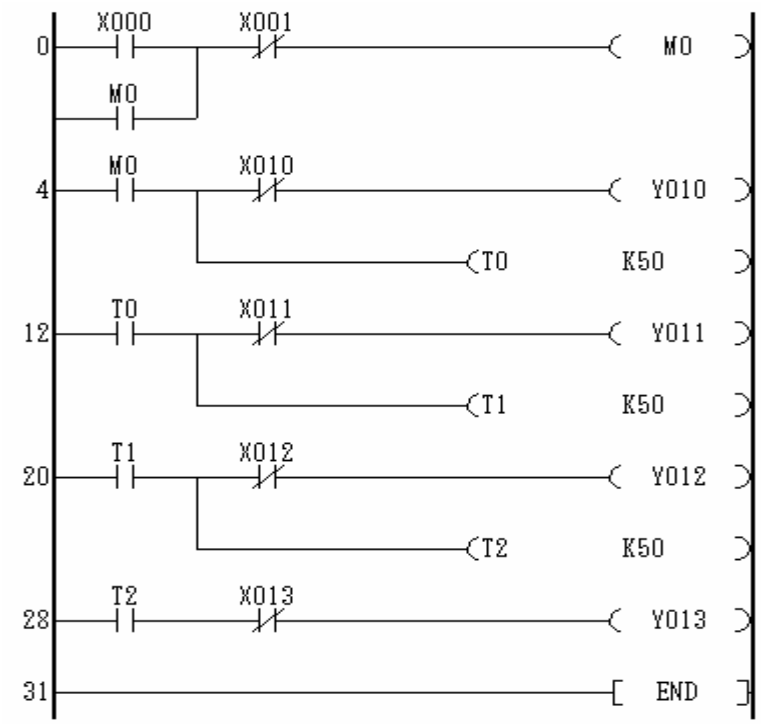
嵌入式 PLC 对扫描时间没有限制，扫描周期由一次执行指令的类型和数量决定。

5.2.2　输入输出继电器应用

①、基本指令中应用

程序访问外部设备的接口，用逻辑指令编程，主要用作位型设备。

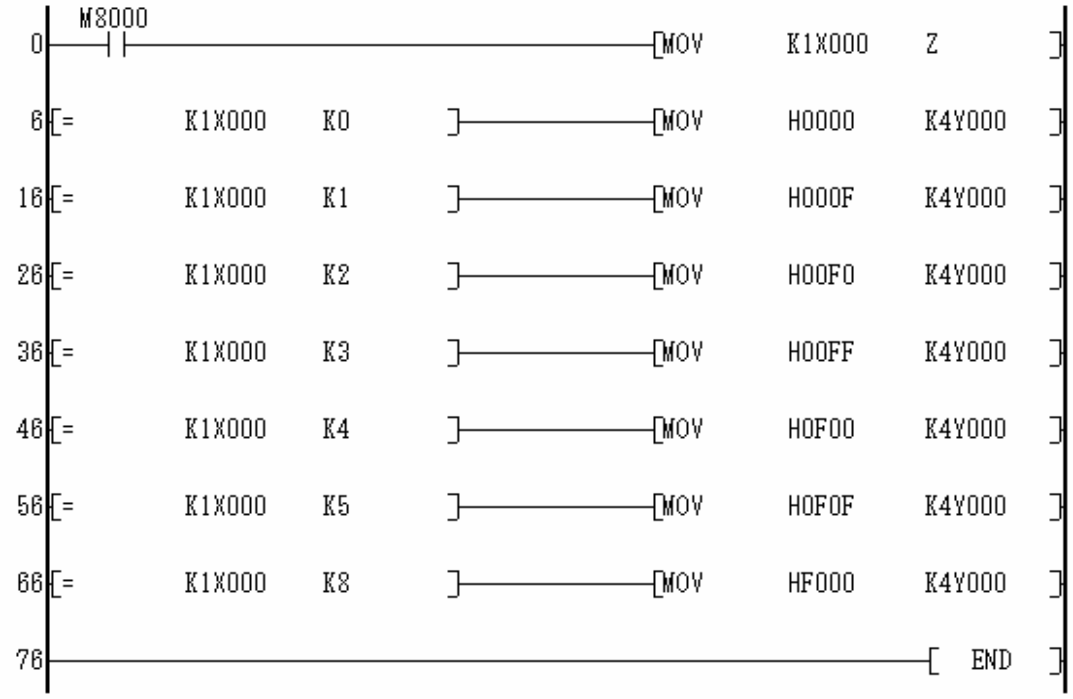
例中，要求 4 台电机依次间隔 5 s 启动。X0 启动，X1 停止，X10---X13 为电机保护输入，Y10---Y13 分别驱动四台电机。



②、功能指令中应用

X、Y 可组合成字来进行输入和输出。

例：K1X000 作数字输入时，可构成 K0----K15 共 16 个不同命令，根据不同命令，Y0---Y15 可组成 16 个不同的输出方式。



5.3 辅助中间继电器 M 说明及应用

5.3.1 辅助中间继电器 M 说明

辅助中间继电器 M 与输出继电器 Y 类似，但它没有与硬件连接，因此不能直接控制硬件。

①、M 的标号范围、数量

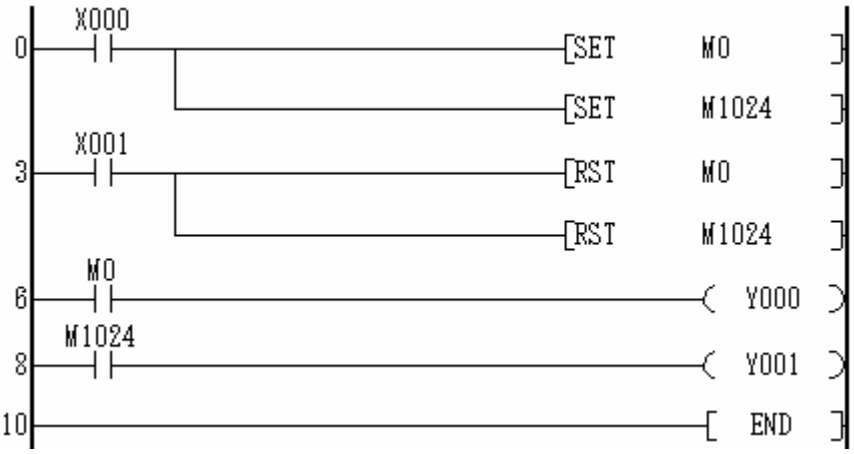
标号范围：M0-----M1535 ，十进制，共计 1536 个。

一般用：　　M0----M1023，计 1024 个。

停电保持用：M1024----M1535，计 512 个。

②、一般型 M 与停电保持型 M 的区别

- 一般型 M：在程序运行时，设备停电后再送电，M 不能记忆停电前的状态，只与当前控制条件相关。
- 停电保持型 M：在程序运行时，设备停电后再送电，M 的状态不仅与当前控制条件有关，还与停电前状态相关。如，



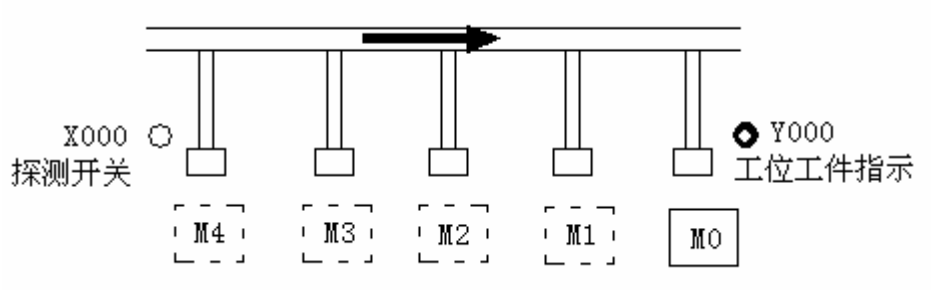
程序运行时，点动 X000=ON，则 M0=ON，M1024=ON；断电后观察发现 M0=OFF，M1024=ON。说明 M1024 保持停电前状态。

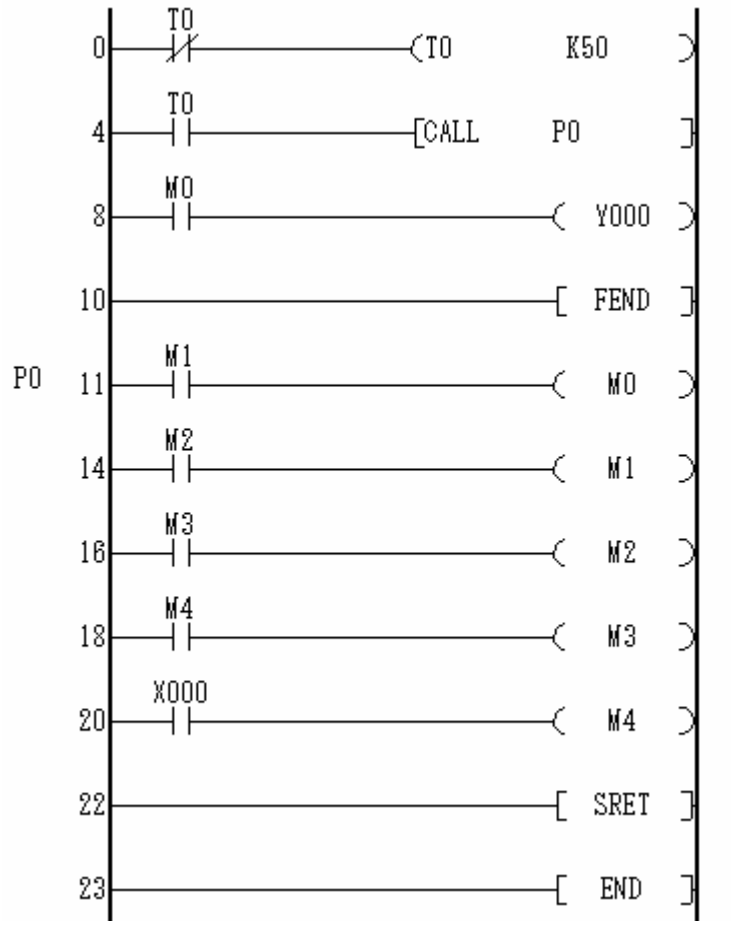
程序在运行时，点动 X000=ON→OFF，则 M0=ON，M1024=ON；将切换开关由 RUN 打到 STOP，此时不断电，又回到 RUN 状态，发现 M0=OFF，M1024=ON。说明切换 RUN→STOP 切换对 M 有相同效果。

5.3.2 辅助中间继电器 M 应用

①、用作位元件

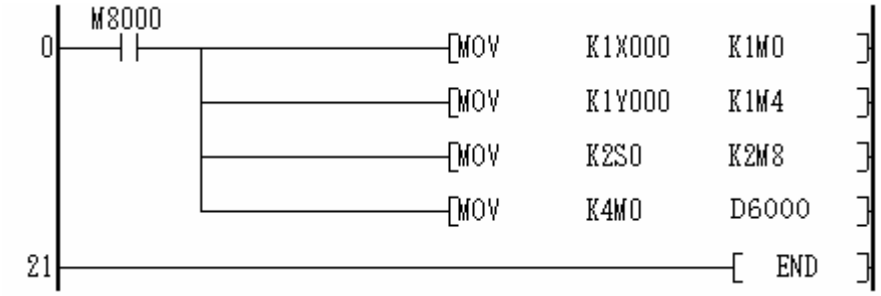
输送机每节距运行时间设为 5s，而检测点 X000 与处理点 Y000 间有四个节距间隔。若处理点无工件，将不启动处理，有工件才进行处理。如图，用 M0、M1、M2、M3、M4 分别对应图上位置，ON 时表示该位置有工件。





②、用作字元件

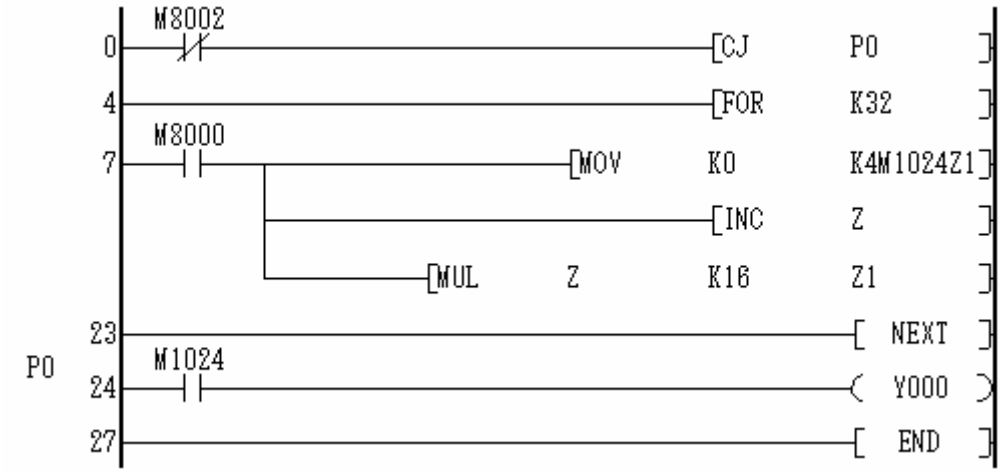
在嵌入式 PLC 的网络通讯过程中，只对数据寄存器 D 进行传输，此时常用 M 作字元件。如要求传送 X0-X4，Y0-Y4，S0-S7 的数据到主站。我们利用 M 拼成字后送到 D6000，D6000 是嵌入式 PLC 从站中的特殊数据寄存器，由网络自动发送到主站。



③、特殊应用及注意事项

- 当使用停电保持型 M 时，注意上电后对初始状态的影响。一般情况下在第一个周期内，对不必要保持的 M 作 RST 处理或用类于 MOV K0 K4M1024 作批清零处理。

  下例中，利用 M8002 一个周期对 M1024---M1535 进行清零。
- 使用 M8002 进行初始化处理。其它程序为保证硬件信号上电匹配，可延时执行。
- 辅助继电器 M 用作字访问时，访问范围可达 M3071，但 M1535 以上不能进行位访问。



5.4 状态继电器 S 说明及应用

5.4.1 状态继电器 S 说明

状态继电器 S 表示一个工序状态时，主要用于步进顺控指令中。STL 指令作用的元件只能是状态继电器 S，并不允许 S 作变址指定。

在非 STL 指令中，状态继电器 S 作为通用辅助继电器使用。

①、S 的标号范围、数量

标号范围：S0-----S999 ，十进制，共计 1000 个。

一般用：　　S0----S499，计 500 个。

停电保持用：S500----S999，计 500 个。

②、一般型 S 与停电保持型 S 的区别

- 一般型 S：在程序运行时，设备停电后再送电，S 不能记忆停电前的状态，只与

当前控制条件相关。

- 停电保持型 S：在程序运行时，设备停电后再送电，S 的状态不仅与当前控制条件有关，还与停电前状态相关。

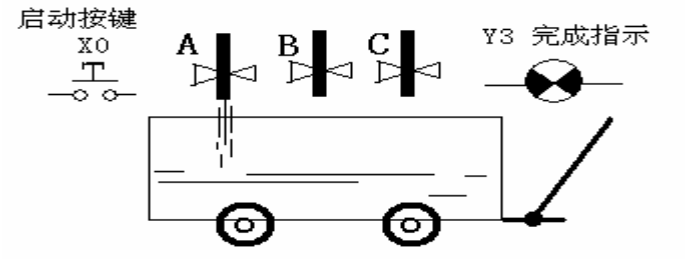
在复杂的工序步中，停电保持功能显得非常重要。

5.4.2 状态继电器 S 应用

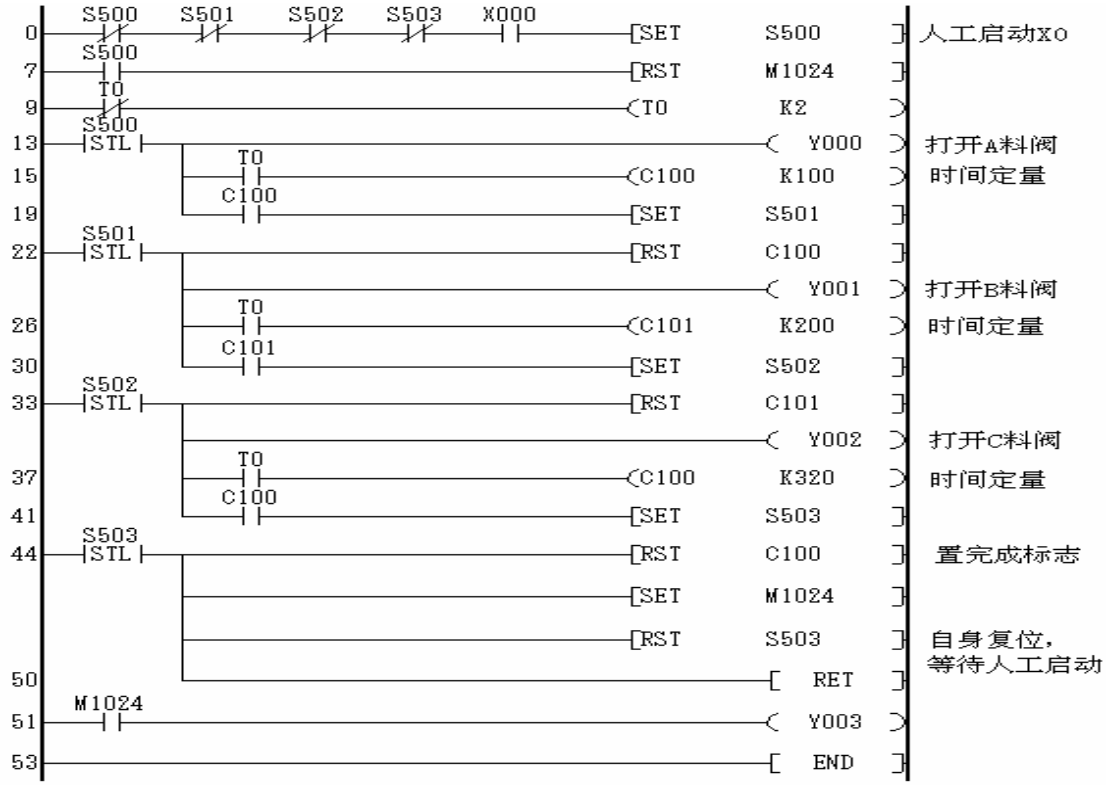
①、应用 S 停电保持功能。

示例说明：化工生产过程中，A、B、C 三种原料按一定顺序和一定数量混合。人工启动混合过程，混合完后，机器给出完成信号，同时可手工启动下一轮混合。在混合过程中停电，并不影响混合的顺序和数量。

工艺示意图：



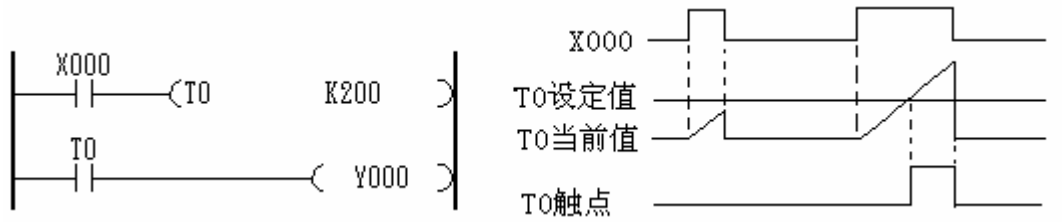
梯形图：例中 S，C，M 均选停电保持型。



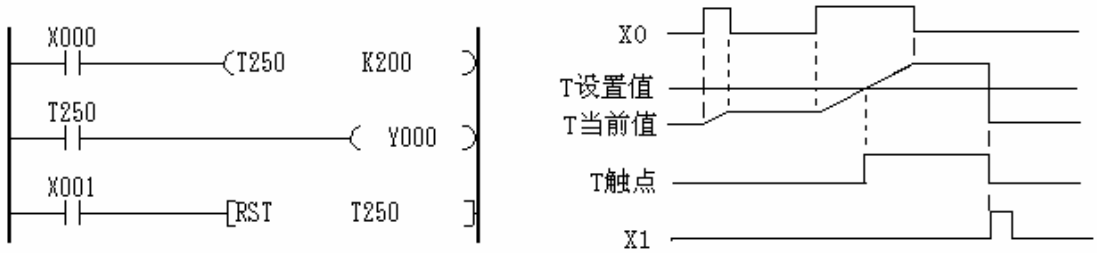
5.5 定时器 T 说明及应用

5.5.1 定时器 T 说明

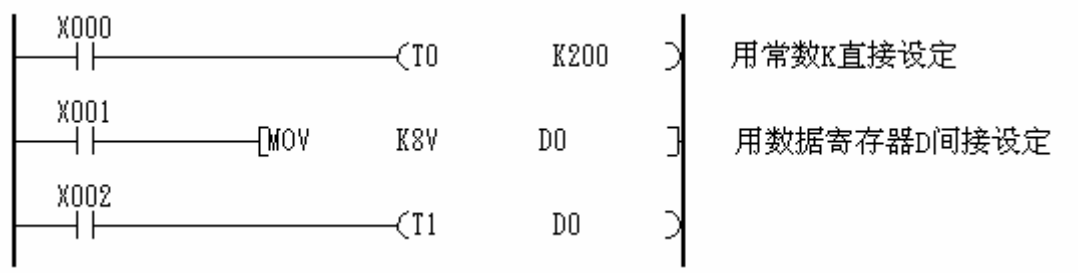
定时器 T 也可称时间继电器，当计时值达到所设时间后，继电器线圈吸合，对外以触点方式输出，触点在程序中可作无限次的使用。在程序中，主要起定时控制作用。
定时器可用作数据寄存器。
通过赋值改变计时器的当前值，从而影响触点输出，程序方法改变计时长度。
①、T 的类型、标号范围、数量
100ms 型：
一般用：T0------T199，200 点。
累积用：T250----T255，6 点。
10ms 型： T200-----T245，46 点，只作一般用。
1ms 型：T246------T249，4 点，只作累积用。
如定时器设定值 K200，对于 100ms 型，计时长度=200***100ms**=20s；对于 10ms 型，计时长度=200***10ms**=2s；对于 1ms 型，计时长度=200***1ms**=0.2s。
设定计时值可直接指定常数 K，也可由数据寄存器 D 间接指定。
②、一般定时器



X0=ON，T0 每 100ms 计数 1 次，当计数值达到 200 时，T0 常开触点 ON，当前值继续计数，当计到最大值 K32767 时保持不变。
X0=OFF，计时器 T0 复位，T0 当前值=0，T0 常开触点 OFF。
断电后，计时器 T0 复位。
③、累积型定时器



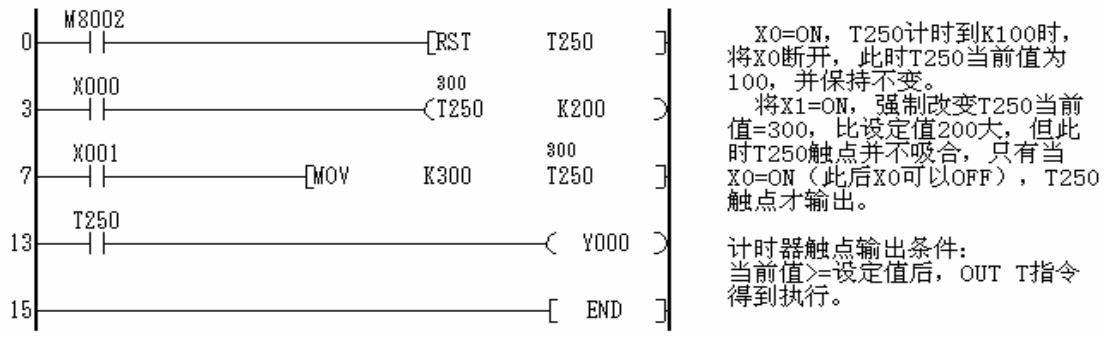
X0=ON，T250 每 100ms 计数 1 次，当计数值达到 200 时，T250 常开触点 ON，当前值继续计数，当计到最大值 K32767 时保持不变。
X0=OFF，T250 当前值保持不变，T250 触点状态保持不变。
X1=ON，定时器 T250 复位，T250 当前值=0，T250 常开触点=OFF。
断电后，重新上电 T250 保持断电前状态。
④、设定定时常数的方法

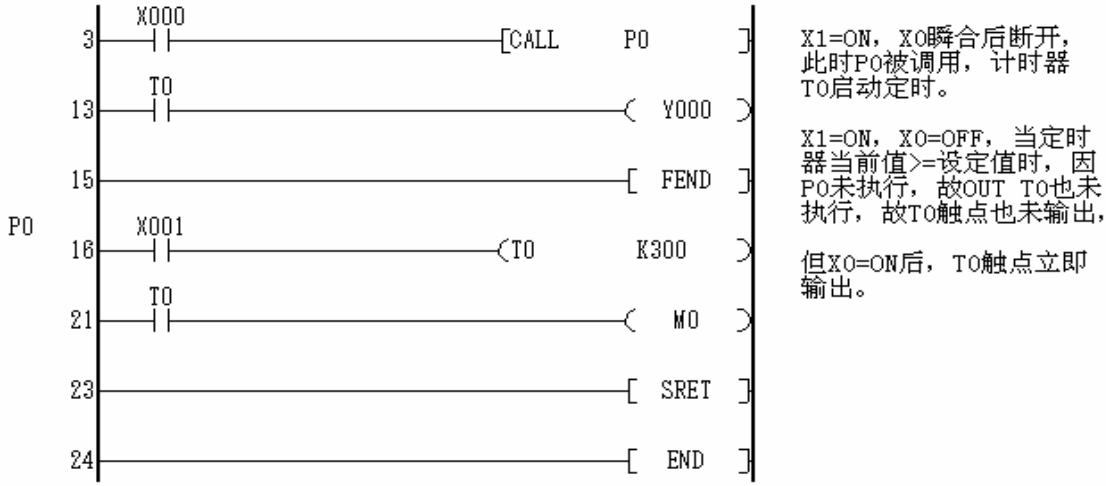


直接设定时，只能用常数 K，不能用常数 H。

⑤、定时器触点输出

定时器是中断方式计时的，当当前值>=设定值时，若没有执行 OUT T 指令，则计时器不能输出。因为当前值与设定值比较是在 OUT 指令中执行的。因此会出现下面情况：定时器启动后，OUT 指令总未执行，当前值远大于设定值，但触点并未输出。

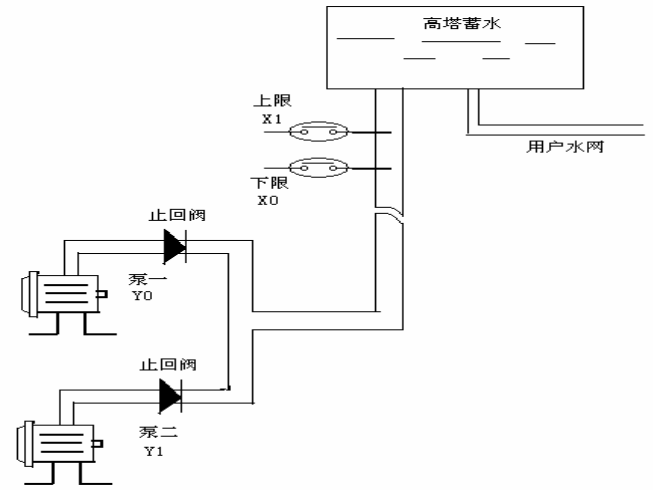




由此看出，定时器执行精度与 OUT 指令输出密切相关。

5.5.2　定时器 T 应用

定时器在控制中应用很广，如电机的顺次延时启动，出门延时关灯，温度分时段控制等。



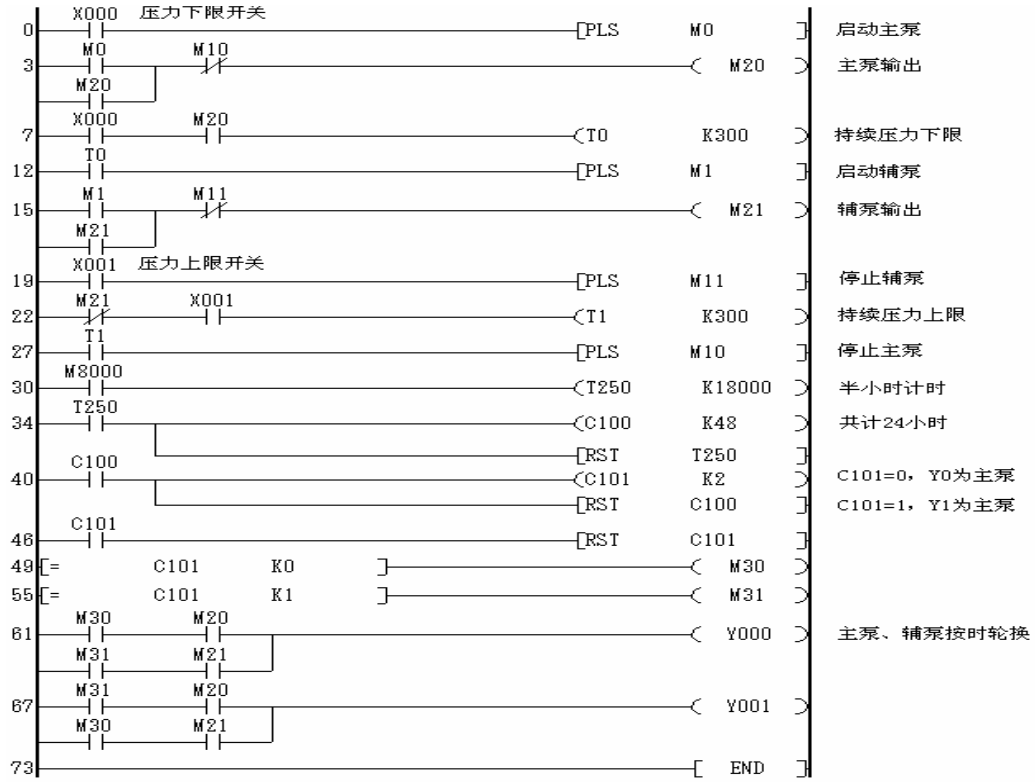
示例说明：用压力开关控制供水。供水泵共有二台，分为主泵、辅泵。

当压力低于下限时（X0=ON），先启动主泵，若压力低持续 5 分钟，再启动辅泵；

当压力高于上限时（X1=ON），先停辅泵，若压力高持续 5 分钟，再停主泵。

主泵和辅泵并不是固定不变的，要求一天变换一次主、辅泵的角色，保证均衡使用二泵。

梯形图：



5.6 计数器 C 说明及应用

计数器接收脉冲信号，并根据要求记录脉冲数，根据脉冲数与设定值关系，输出触点控制信号。

计数器按计数范围分，有 16bit 长和 32bit 长两种不同结构的计数器。尽管两类计数器都用 C 开头，但却不能用变址方式从头到尾进行访问，实际上是结构不同的两类元件。因此在下面的说明中分开进行。

5.6.1 16 bit 计数器 C 说明

16 bit 计数器 C 对脉冲进行计数，当计数值达到或超过设定值时，计数器常开触点=ON，触点在程序中可作无限次的使用。

16 bit 计数器 C 是加法计数器。

计数设定值可由常数 K 直接指定，也可由数据寄存器 D 间接指定。

16bit 计数设定值范围：1-----+32766，理论上可扩展到-32766------+32766。

计数器可作数据寄存器使用，可在 –32768------+32767 内任意取值。

通过赋值改变计数器的当前值，从而影响触点输出。

当计数脉冲是时钟脉冲时，计数器可用作扩展定时器。

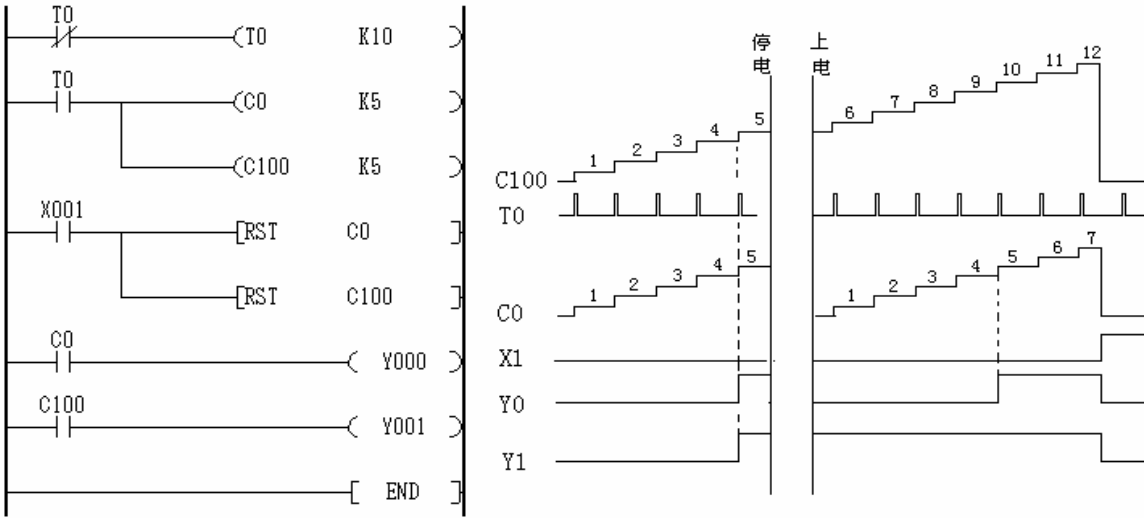
①、16bit 的类型、标号范围、数量

标号：C0-C199，十进制，共计 200 个。

一般用：C0------C99，100 个。

保持用：C100---C199，100 个。

②、16 bit 计数器一般型和保持型的区别

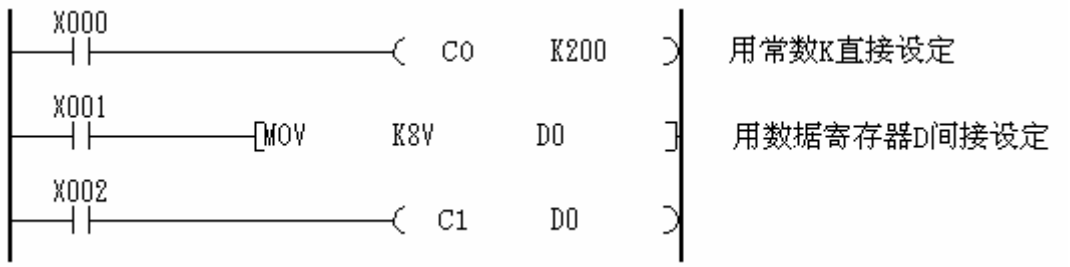


一般计数器 C0 停电后，计数值被清除，从零开始计数。

保持型计数器 C100 停电后，再上电，计数值在原有值上增加。

RUN→STOP 一次，和停电效果相同。

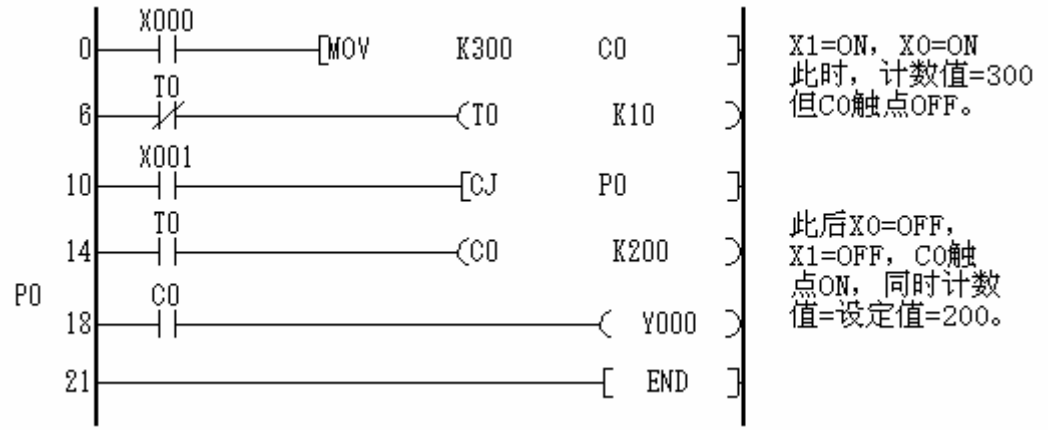
③、16 bit 计数器设定值的设定方法



直接设定时，只能用常数 K，不能用常数 H。

④、16 bit 计数器触点输出

计数值>=设定值时，若没有执行 OUT C 指令，则计数器不输出。因为计数值与设定值比较是在 OUT 指令中执行的。当 OUT C 执行时，若计数值大于给定值，触点输出，同时计数值更改为设定值。



5.6.2　32 bit 计数器 C 说明

32 bit 计数器 C 是环形可逆计数器。

增计数时，计数值由（设定值-1）→设定值时，计数器触点置位。

减计数时，计数值由设定值→（设定值-1）时，计数器触点复位。

无论增减计数，计数值在其他情况下，都不影响计数器触点。

RST C 指令，复位计数器触点，计数值置 0。

计数设定值可由常数 K 直接指定，也可由 32 bit 数据寄存器 D_{N-1}（D_N）间接指定。

32bit 计数设定值范围：K-2147483648-------K2147483647。

环形计数器特点：K-2147483648 减 1 后变为 K2147483647；

K2147483647 加 1 后变为 K-2147483648。

计数器可作 32bit 数据寄存器使用。

通过赋值改变计数器的当前值，从而影响触点输出。

①、32bit 的类型、标号范围、数量

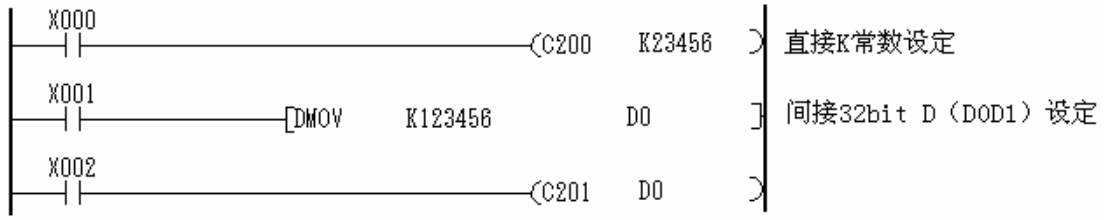
标号：C200-C255，十进制，共计 56 个。

32bit 计数器都是停电保持型的，即计数值、触点状态断电后均保持断电前状态。

②、设置 32 bit 计数器设定值

直接 K 常数设定；

用 2 个相邻的数据寄存器来间接设定。



③、控制 32 bit 计数器的计数方向

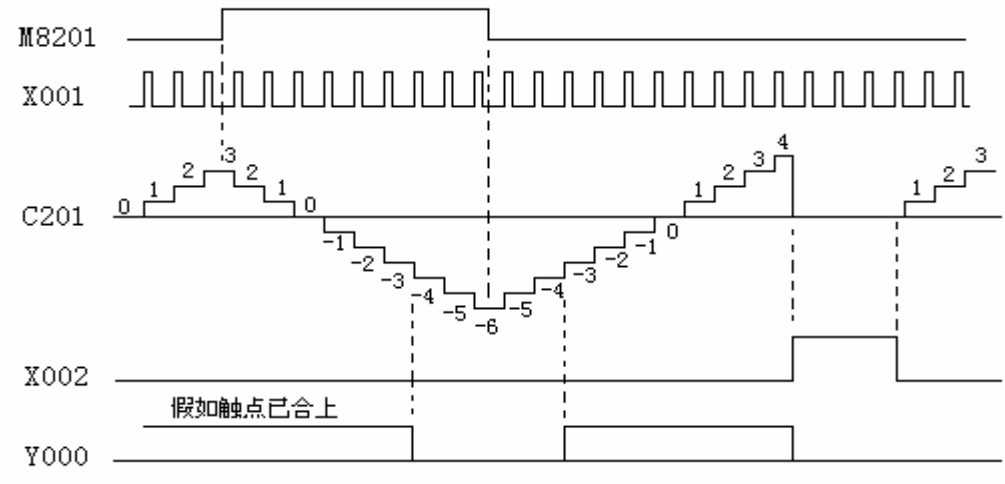
32 bit 计数器只有一个计数端，其增减方向控制是通过驱动特殊功能继电器来实现的。

C200-----C255 对应特殊功能继电器为 M8200-----M8255。

④、32bit 计数器触点动作过程

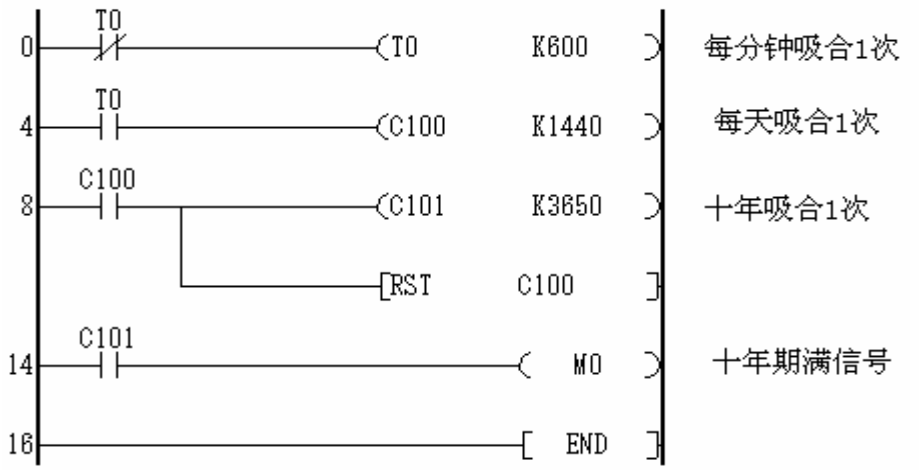
以③中梯形图为例。增计数：计数值由 K-4 到 K-3 时，C201 触点置位；

减计数：计数值由 K-3 到 K-4 时，C201 触点复位；

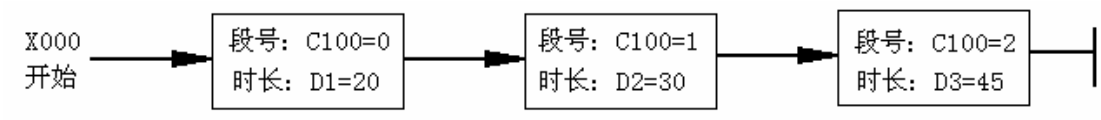


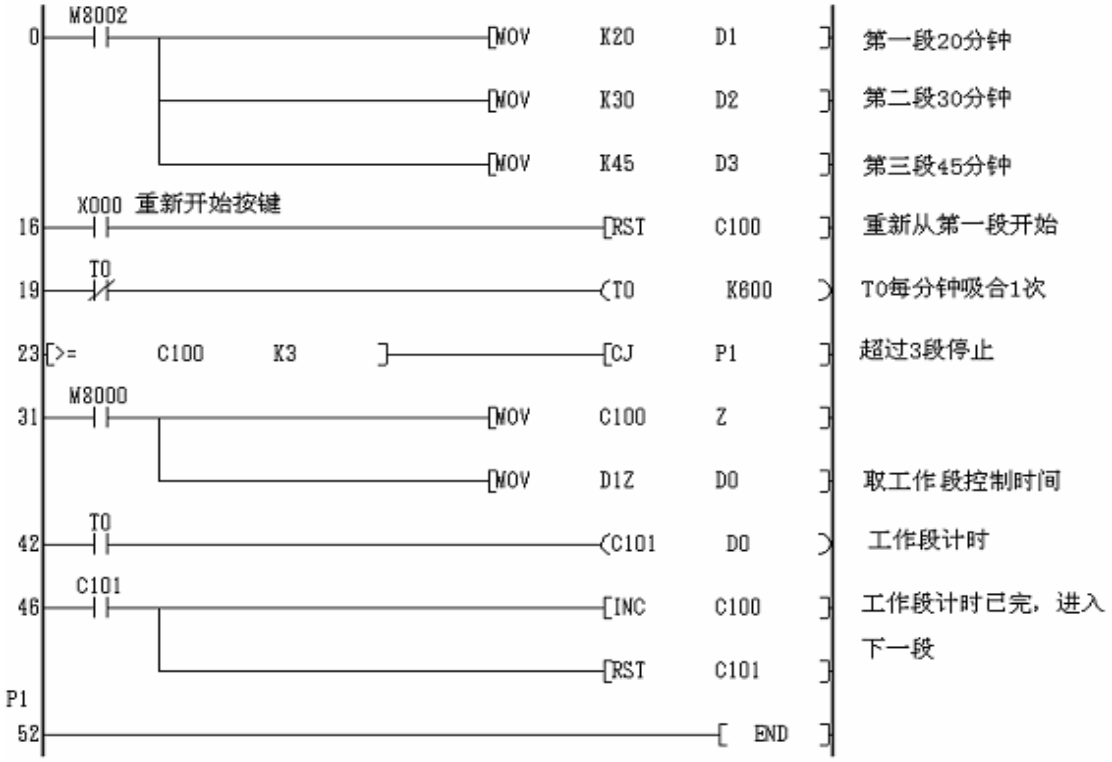
5.6.3 16 bit 计数器 C 应用

设备厂家往往想知道系统运行的总时间，可用计数器扩展计时方法得到。



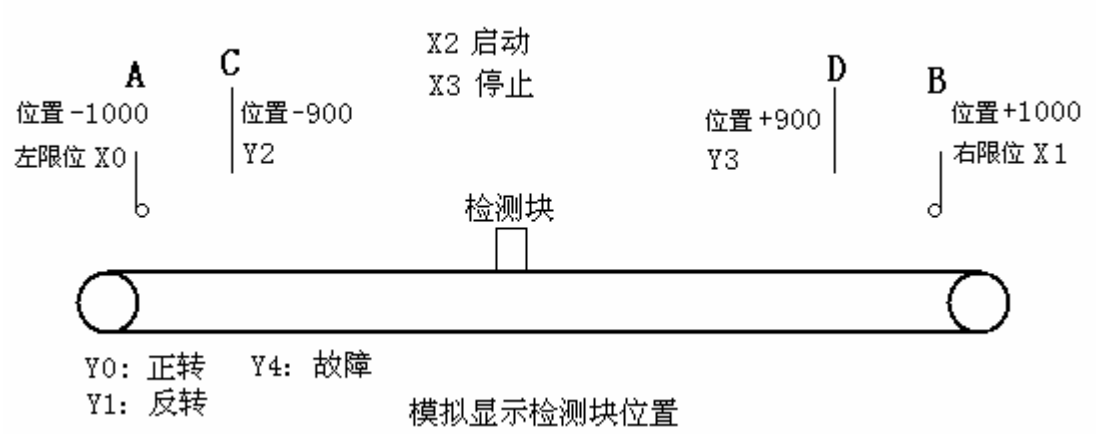
温度控制系统中，常常根据工艺要求，分成多段进行控制。每段段号和时间可以使用计数器。例中分三段控制，C100 表示段号（0、1、2），C101 表示工作段进行时间，根据段号变化，分别从 D1、D2、D3 取出工作时间。工作时间事先预置，也可通过人机界面修改。





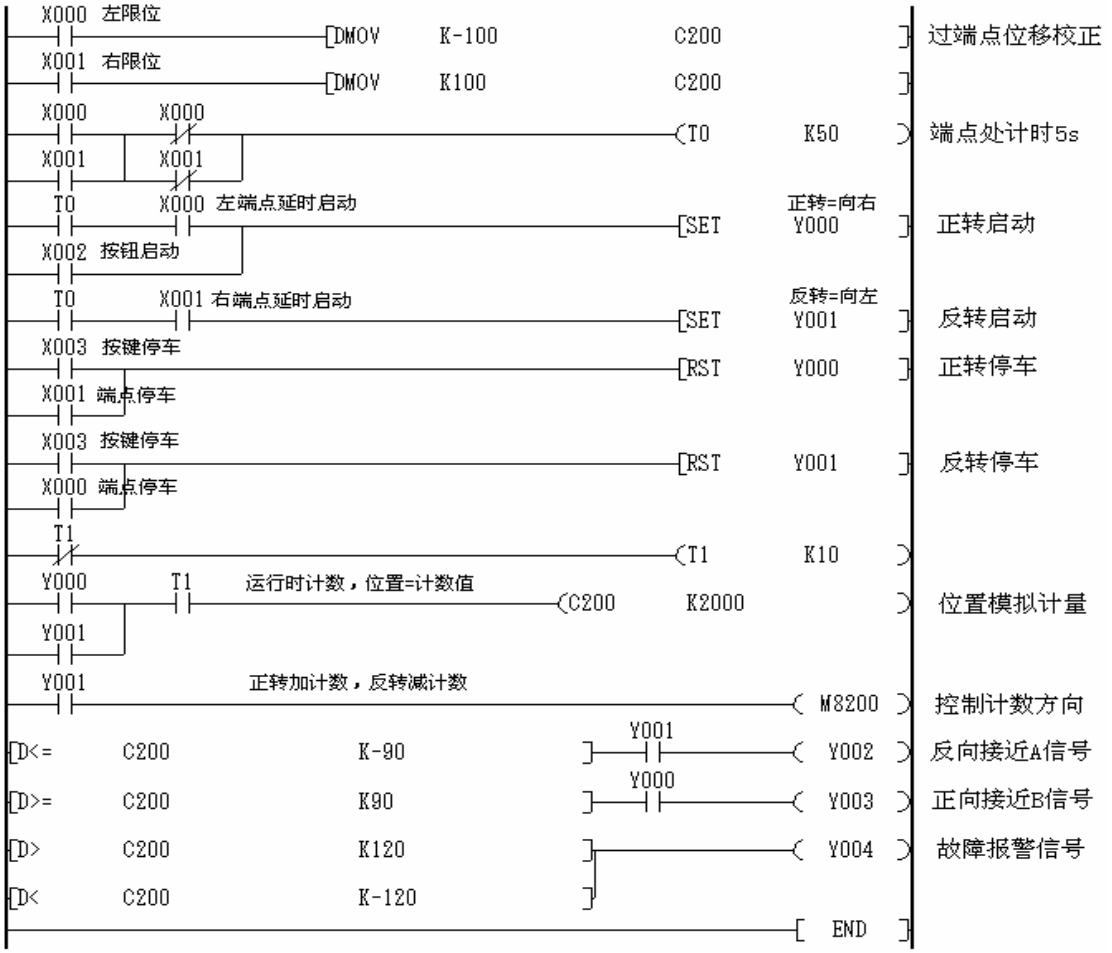
5.6.4　32 bit 计数器应用

有一往复式机构，在 A、B 两处之间往复，但要求反向接近 A 或正向接近 B 时，发出警告信号（Y2、Y3）；到达 A（X0=ON）或 B（X1=ON）后，停 5s（T0）后改变方向；考



模拟显示检测块位置

考虑最大允许范围，如果没有达到 A 或 B，发出故障报警；对机构位置要有模拟显示功能，为保模拟准确，在 A、B 处重置位移量，消除累积误差。



5.7 数据寄存器 D 说明及应用

5.7.1 数据寄存器 D 说明

一个数据寄存器 D 又称为一个字，字长为 16bit，主要用来存储数据。以字长为单位，可以组成双字、三字、四字等数据单元。

数据存储主要以单字和双字存储为主，其最高 1 位是符号位。

数据寄存器 D 只是一个数据表达的形式，用来表示整数、二进制浮点数、BCD 等格式，以及用户根据需要，对每位赋予不同的含义。

数据寄存器 D 在数据运算、网络通讯方面等起极为重要的作用。

①、数据寄存器 D 的类型、标号范围、数量

标号：通用数据寄存器：D0-D5999，十进制，共计 6000 个。

一般用：D0------D199，200 个。

保持用：D200---D5999，5800 个。

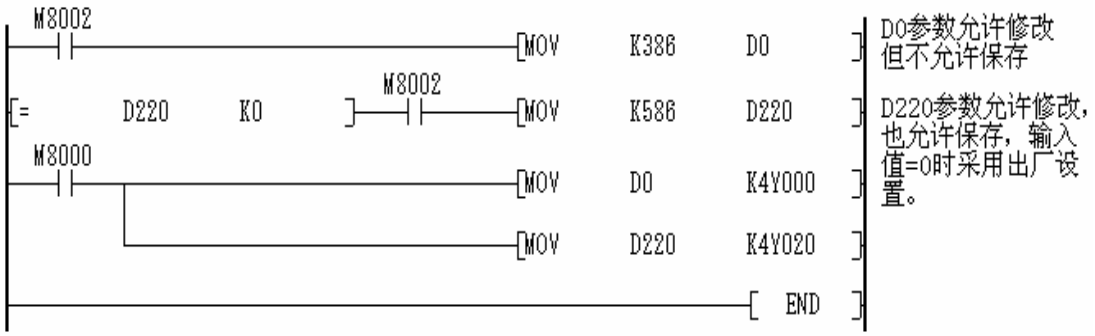
网络功能用：D6000-----D7019 ，十进制，共计 1020 个。网络功能用数据寄存器为停电保持型。

②、一般型和保持型的区别

一般型，断电后，数值为 0；保持型，断电后仍保持断电前状态。

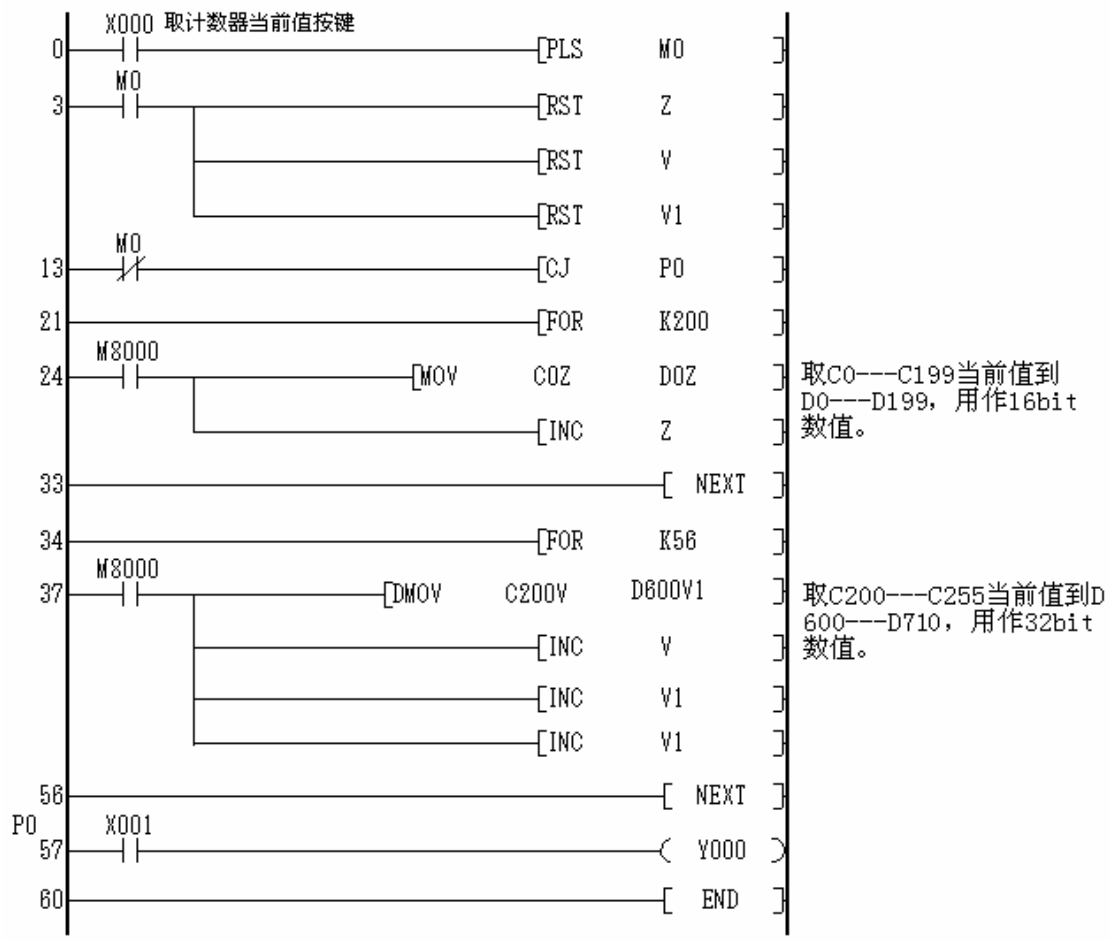
控制系统有些参数允许用户调整，调整后被保存。

在调整后，不知对错情况下，可采用出厂设置。



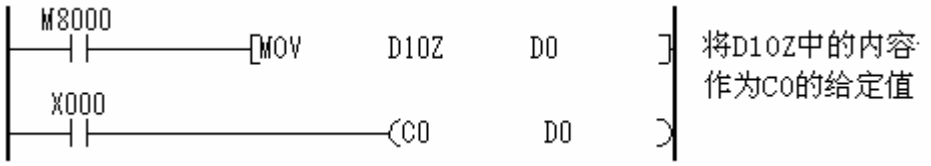
改变 D 中数值办法：在监控程序（EasyWin 或人机界面）上直接修改当前值。

③、读取计数器的当前值。

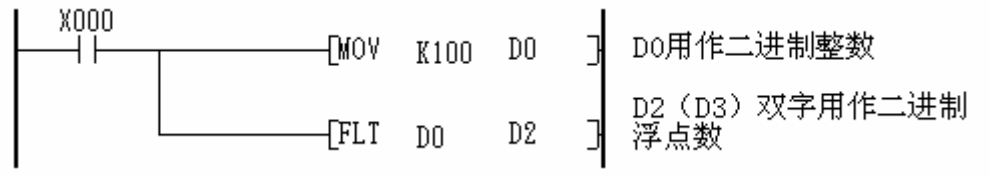


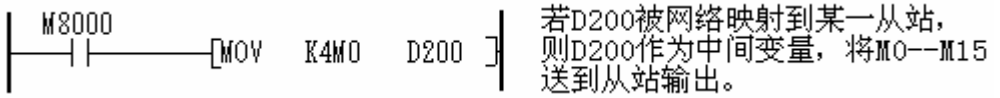
如果进行例中的反向传送，则可改变计数器当前值。

④、通过 D 设定计数器给定值，达到变址给定的目的。



⑤、按其它定义方式使用

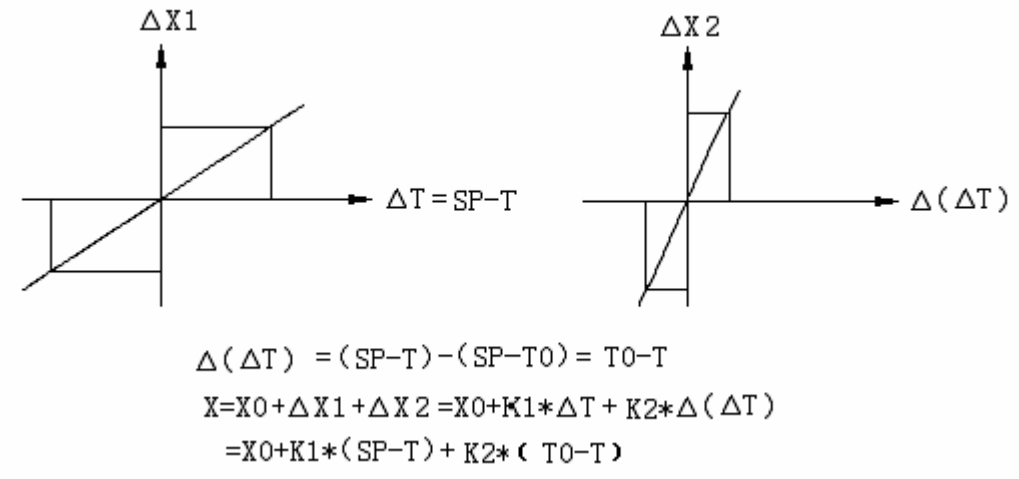




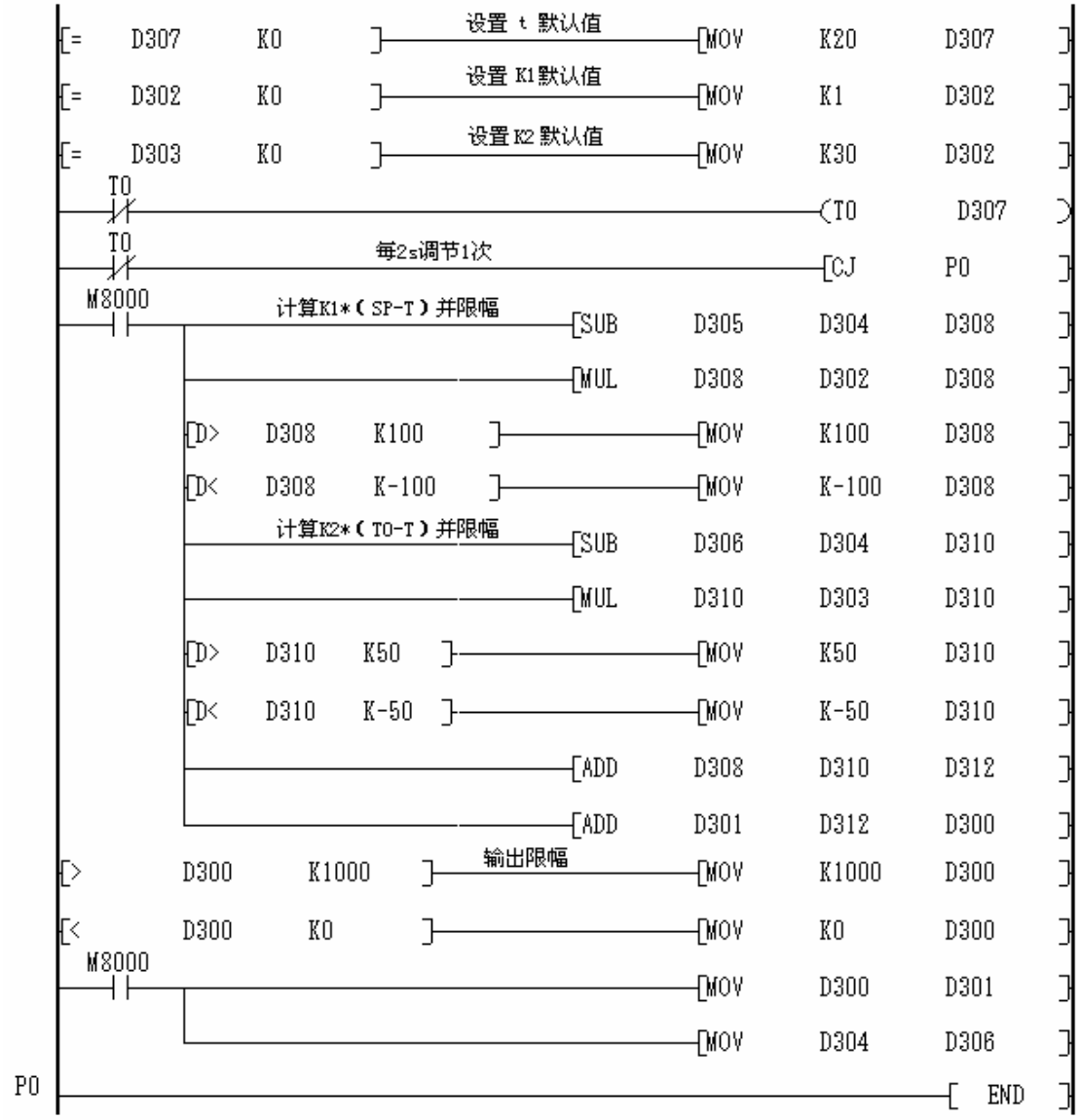
⑥、网络功能用数据寄存器 D6000-----D6999，D7000------D7999 的使用方法，见《网络及网络编程》一章。

5.7.2　数据寄存器 D 应用

在应用指令中，大量使用数据寄寄器。
在很多实际系统中，用户可根据特定的经验编制控制程序。



示例中，一简单温度控制算法（正向调节），
X：当前理论输出（0---1000）；　　X0：前一调节周期的输出；
T：当前实测温度；　　T0：前一周期实测温度；
K1、K2 为调整斜率，K2>=30*K1；如 K1=1，K2=30；
调节周期 t=可设定，出厂默认为 2s。
在梯形图中，对应关系为 X=D300，X0=D301，K1=D302，K2=D303，T=D304，SP=D305，T0=D306，t=D307。
当（给定 SP - 实测 T）>　0，输出 X 渐大，但 X<=1000；
当（给定 SP - 实测 T）<　0，输出 X 渐小，但 X>=0；



同一个例子也可用浮点数编写。

在网络应用中，大量使用数据寄存器 D。主站与从站通讯，主站自身必须使用数据寄存器 D。具体参见第七章《网络及网络编程》。

5.8 程序位置指针 P 说明及应用

5.8.1 程序位置指针 P 说明

在控制程序流程时，作为分支标记或子程序名称。

在 CJ、CALL 指令中使用。

P 作指令操作数时可变址修饰，但在程序中必须找到对应的标识，否则程序运行时出错。

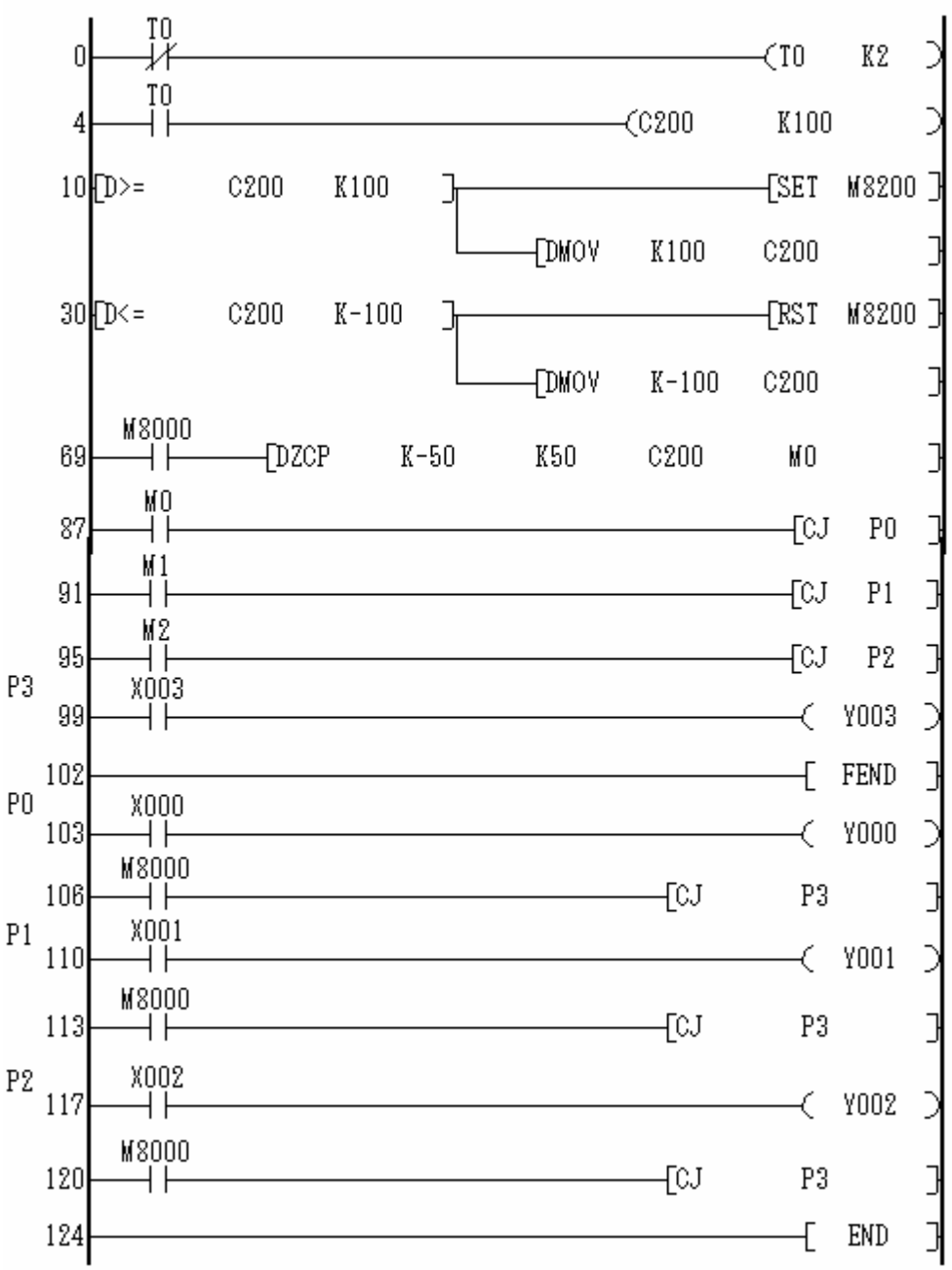
P 标识程序位置时（P 出现于左母线侧时），必须是唯一的，且不带变址修饰。

标号范围：P0---P127；标号为十进制。

数量：128 个。

5.8.2 程序位置指针 P 的应用

①、在 CJ 中的应用（分支标记）。

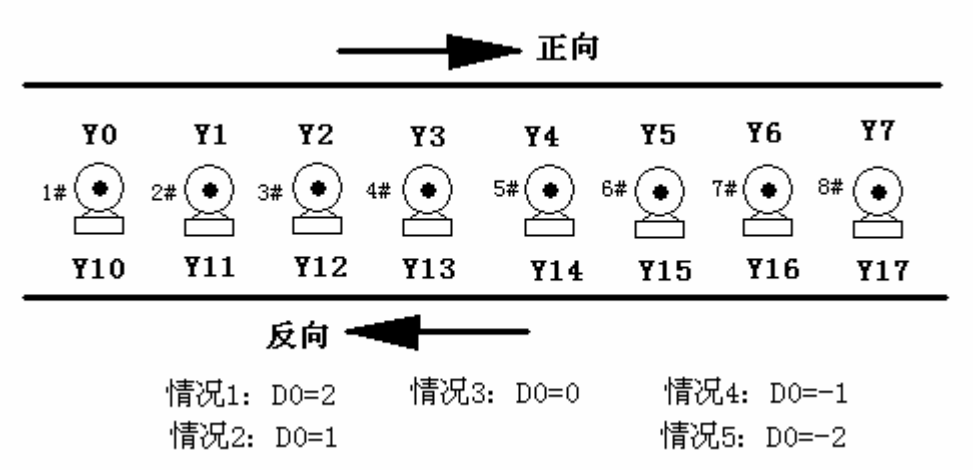


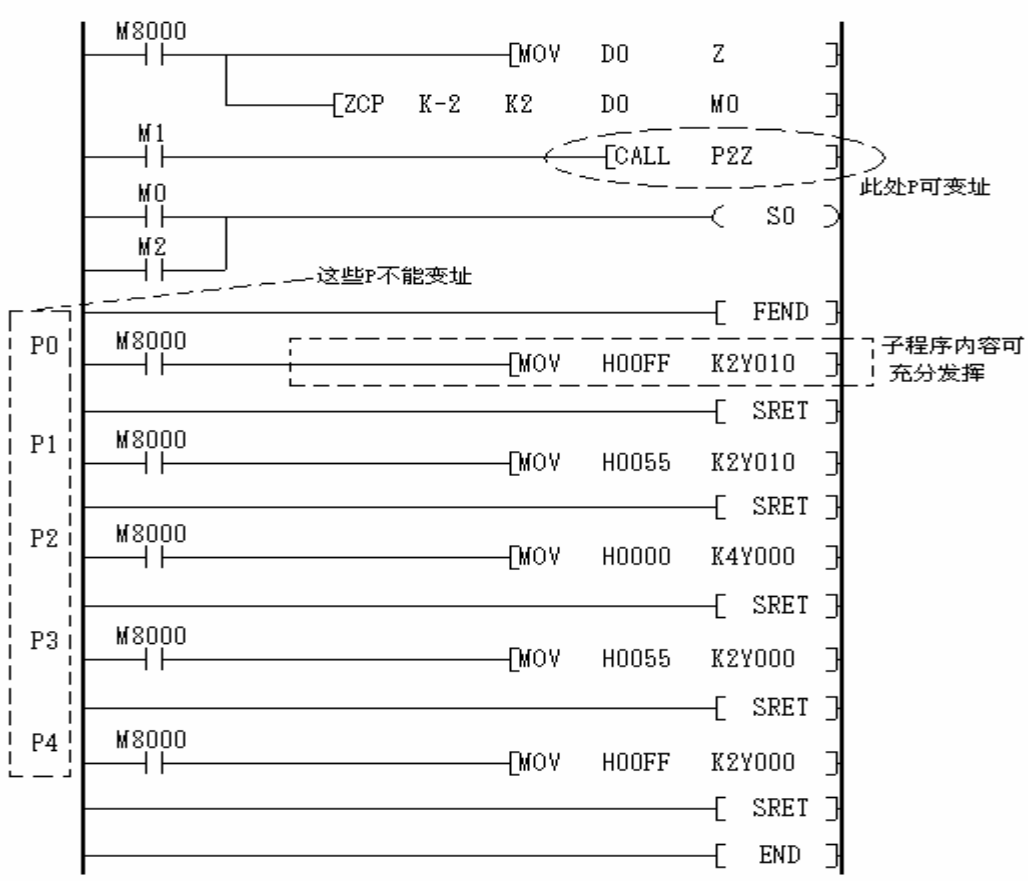
T0作为C200计数脉冲，C200在K-100到K100间来回计数。
C200 < K-50，跳到P0；K-50 <= C200 <= K50，跳到P1；C200 > K50，跳到P2；
P0、P1、P2汇合于P3。

②、在 CALL 中的应用（子程序名称）。

有 8 台电机组成的联合机，有五种工作情况（含停机）：

情况 1：8 台正向同时运转；
情况 2：1#、3#、5#、7# 四台正向同时运转；
情况 3：全部停机；
情况 4：1#、3#、5#、7# 四台反向同时运转；
情况 5：8 台反向同时运转；





5.9 常数标记 K、H 详细说明

5.9.1 常数标记 K

K 后紧接的字符，表示十进制的常数。如 K100 表示十进制 100。在计数器、定时器设定值时使用；在功能指令的操作数中用到常数时使用。

K 常数允许用变址修饰，如 K100V1，表示十进制常数 1200（设 V=1100）；

K1000Z7，表示十进制常数-100（设 Z7=-1100）。

用在基本指令中，常数 K 不能作变址修饰，如 OUT C0 K100V 是不被接受的。

功能指令中，类似 KnM0 组合，表示 n*4 个位元组成的 BIN 常数。

此处 n 不能变址修饰，如 K1VM0 不被接受，而 K1M0V 是对 M 变址，是许可的。

16bit 指令中 n<=4；32bit 指令中 n<=8；

如，K1M0 表示 4 个位元（M0---M3）组合一起表示的 BIN 常数；

K2M0 表示 8 个位元（M0---M7）组合一起表示的 BIN 常数；

K3M0 表示 12 个位元（M0---M11）组合一起表示的 BIN 常数；

5.9.2 常数标记 H

H 后紧接的字符，表示十六进制常数。在应用指令中用作常数操作数。

H 常数允许用变址修饰，如 H0010V1，表示十六进制常数 H001A=K26（设 V=10）；

5.10 特殊软元件说明

常被使用的特殊继电器软元件如下：

M8000：无条件输出继电器，程序运行时 ON；在程序需过渡时使用。

M8002：初始瞬合继电器，程序运行的第一个扫描周期时 ON；在参数初始化时使用。

M8020：零标志；作为某些运算的标志被引用。

M8021： 借位标志；作为某些运算的标志被引用。

M8022：进位标志；作为某些运算的标志被引用。

M8200---M8255：当 32 bit 可逆计数器 C200-C255 作减计数时，必须驱动相应的特殊继电器。如 M8200=ON，C200 作减计数。

第六章　专家指令

6.1 PID 运算

6.1.1 指令解说

16 位指令	32 位指令
PID 9 步	—

适用软元件

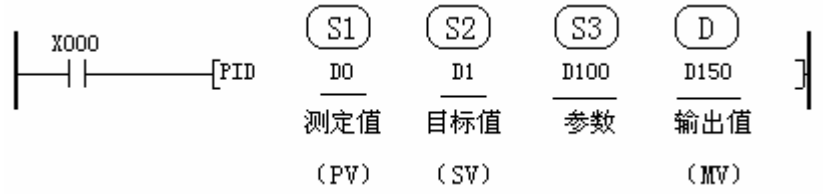
字软	K,H	KnX	KnY	KnM	KnS	T	C	D	V,Z

S1 S2 S3 ⟷ D

S3　D0~D5975

位软				
	X	Y	M	S

指令与操作数



- 用于进行 PID 控制的 PID 运算程序。

  达到采样时间的 PID 指令在其后扫描时进行 PID 运算。

S1　设定目标值(SV)

S2　设定测定现在值(PV)

S3 ~ S3 +6 设定控制参数

} 执行程序时，运算结果(MV)被存入 D

- 对于 D 请根据以下程序，在可编程控制器 RUN 时，务必清除保持的内容。

D 中为指定的停电保持区内的数据寄存器元件号。

- 需占有自 S3 起始的 25 个数据寄存器。

参数设定　控制参数在 PID 运算前必须预先指定。控制参数共有 25 个：

S3　：采样时间(T_S)　1~32767(ms) (但比运算周期短的时间数值无法执行)

S3 +1：动作方向(ACT)　bit0 0：正动作　1：逆动作。

bit1 0：输入变化量报警无　1：输入变化量报警有效

bit2 0：输出变化量报警无　1：输出变化量报警有效

bit3 不可使用

bit4 自动调谐不动作　1：执行自动调谐

bit5 输出值上下限设定无　1：输出值上下限设定有效

bit6~bit15 不可使用

另外，请不要使 bit5 和 bit12 同时处于 ON。

S3 +2：输入滤波常数(α)　0~99[%]　0 时没有输入滤波

S3 +3：比例增益(Kp)　1~32767[%]

S3 +4：积分时间(T1)　0~32767(×100ms) 0 时作为∞处理(无积分)

S3 +5：微分增益(KD)　0~100[%]　0 时无积分增益

S3 +6：微分时间(TD)　0~32767(×10ms)　0 时无微分处理

S3 +7 ~ S3 +19　PID 运算的内部处理占用

S3 +20　输入变化量(增侧)报警设定值 0~32767

S3 +21　输入变化量(增侧)报警设定值 0~32767

S3 +22　输出变化量(增侧)报警设定值 0~32767

S3 +23　输出变化量(减侧)报警设定值有效)

S3 +24　报警输出　　bit0 输入变化量增侧溢出

bit1 输入变化量减侧溢出

bit2 输出变化量增侧溢出

bit3 输出变化量减侧溢出

- PID 指令可多次执行，但请注意运算使用的 S3 或 D 软元件号不要重复。
- PID 指令在子程序、步进梯形图、跳转指令中也可使用。在这种情况下，执行 PID 指令前请清除 S3 +7 后再使用。
- 采样时间 TS 的最大误差为-(1 运算周期+1ms)~+(1 运算周期)。
- 如果采样时间 TS≤可编程控制器的 1 个运算周期，则发生 PID 运算错误。
- 输入滤波常数有使测定值变化平滑的效果。
- 微分增益有缓和输出值急烈变化的效果。
- 动作方向(S3 +1　[bit 0])

S3 +1 的[bit 0]=0：用正动作指定系统的动作方向。

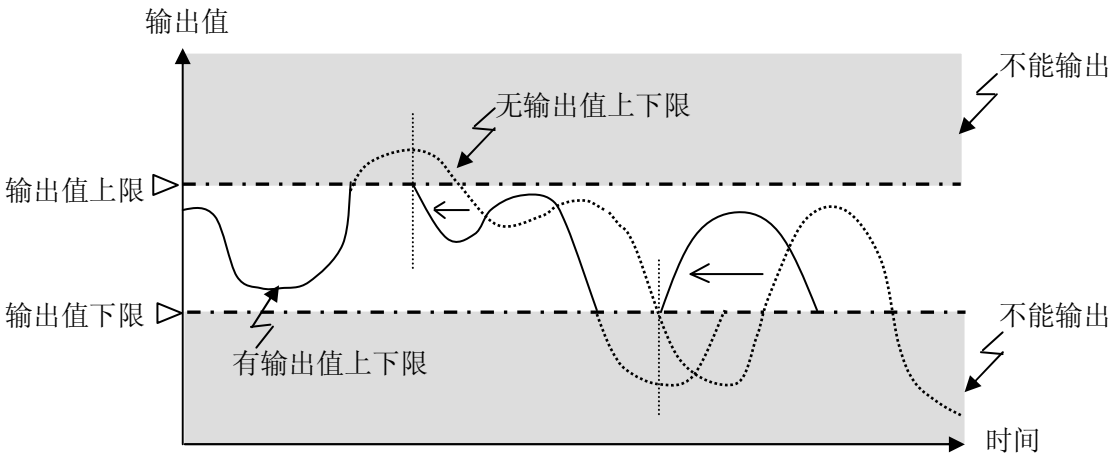
S3 +1 的[bit 0]=1：用逆动作指定系统的动作方向。

- 输出值上下限设定(S3 +1　[bit 5])

S3 +1 的 bit5=1：输出值上下限设定有效。

有抑制 PID 控制的积分项增大的效果。

使用这个功能时，必须 S3 +1 的 bit 2=0。

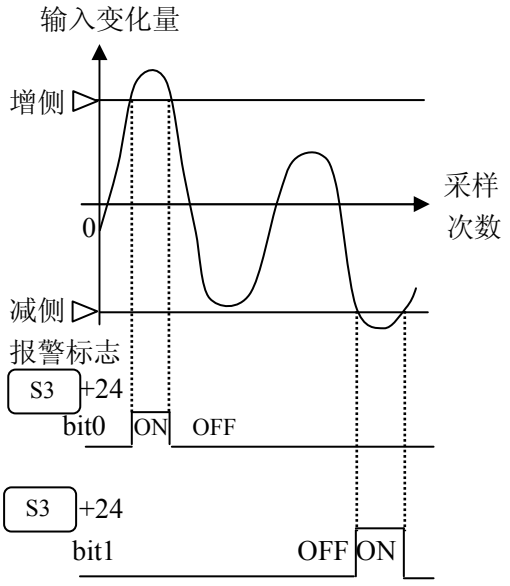


- 输入输出变化量报警设定

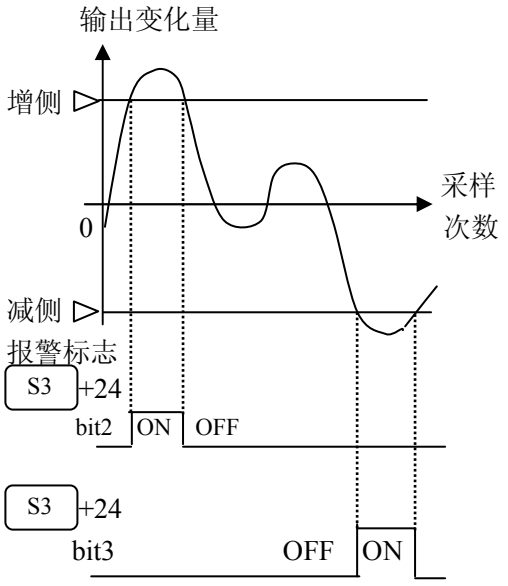
  变化量= (前次的值)-(这次的值)

  报警标志的动作(S3 +24)

i) 输入变化量 (bit1=1)



ii) 输出变化量 (bit2=1)



求 PID 参数

为了执行 PID 控制得到良好的控制结果，必须求得适合于控制对象的各参数的最佳值。这里必须求得 PID 的三个常数：

比例增益Kp ；

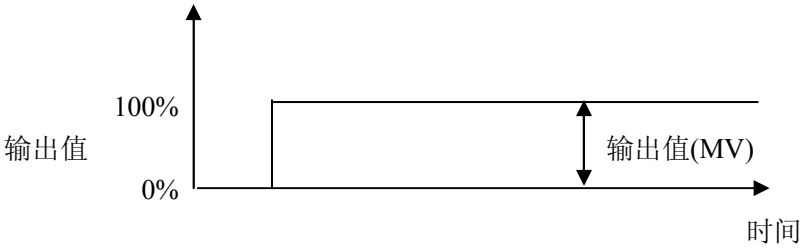
积分时间　　T1 ；

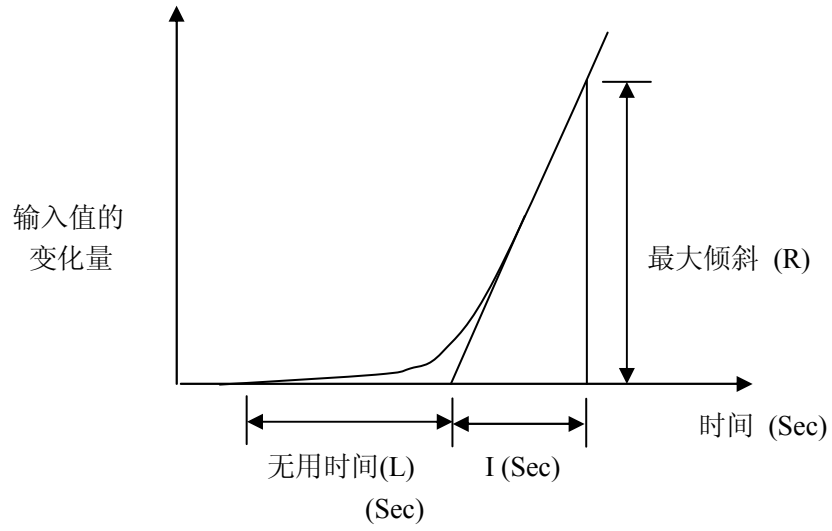
微分时间　　TD 。

这里就该阶路反应法来求取以上三个参数。

阶跃反应法是对控制系统施加 0 → x %的阶跃输出，依据输入变化判断动作特性，从而求得 PID 三个常数的方法。通常取 x=100，75，50 。

<动作特性>





<动作特性和 3 个常数>

	比例增益 (Kp)[%]	积分时间 (T1)[×100ms]	微分时间 (TD)[×100ms]
仅有比例控制(P 动作)	$\frac{1}{RL} \times$ 输出值 (MV)	———	———
PI 控制(PI 动作)	$\frac{0.9}{RL} \times$ 输出值 (MV)	33L	———
PID 控制(PID 动作)	$\frac{1.2}{RL} \times$ 输出值 (MV)	20L	50L

自整定 PID 参数

为了能得到最佳 PID 控制，使用自整定功能。就是用阶跃反应法自动设定重要参数：

动作方向　　S3 +1 的[bit0]

比例增益　　S3 +3

积分时间　　S3 +4

微分时间　　S3 +6

- 传送自整定用输出值至输出值 D 中。

  这个自动调谐用输出值请根据输出设备在输出可能最大值的 50%～100%范围内使用。

- 请设定自整定不能设定的参数(采样时间、输入滤波、微分增益等)以及目标值等。

  若不能满足下述的注意事项，则自整定结果可能不正确。

  注意事项：

  ◆目标值的设定

  自整定开始时的测定值和目标值的差如不是 150 以上则不能正确自整定。

  因此，若不是 150 以上情况时，先设定自整定用目标值，待自整定完成后，再次设定目标值。

◆采样时间

自整定时的采样时间必须在 1 秒(1000ms)以上。

另外本采样时间推荐使用大大长于输出变化周期的时间值。

- [S3]+1(ACT)的 bit 4 设为 ON 后，则自整定开始。
- 变化量达 1/3（开始时的测定值-目标值）以上时，则自整定结束，

  [S3]+1(ACT)的 bit 4 自动变为 OFF。
- 自动调谐请在系统处于稳定状态时开始。如在不稳定的状态开始，则不能正确进行自动调谐。
- 必须在 PID 运算执行前，将正确的测定值读入 PID 测定值(PV)中。特别对模拟量输入模块的输入值进行 PID 运算时，需注意其转换时间。

PID 命令的基本运算式

本指令根据速度形、测定值微分形运算式，进行 PID 运算。PID 控制根据 [S3] 中指定的动作方向的内容，执行正动作或逆动作。运算中的各值是 [S3] 之后单元指定的参数。

PID 基本运算式：

动作方向	PID 运算方式
正动作	$\triangle MV=Kp\{EVn-EVn-1\}+\frac{Ts}{TI}EVn+Dn1$ $EVn=PVnf-SV$ $Dn=\frac{T_D}{Ts+aD\cdot T_D}(-2PVnf-1+PVnf+PVnf-2)+\frac{aD\cdot T_D}{Ts+aD\cdot T_D}\cdot Dn-1$ $MVn=\Sigma\triangle MV$
逆动作	$\triangle MV=Kp\{EVn-EVn-1\}+\frac{Ts}{TI}EVn+Dn1$ $EVn=SV-PVnf$ $Dn=\frac{T_D}{Ts+aD\cdot T_D}(2PVnf-1-PVnf-PVnf-2)+\frac{aD\cdot T_D}{Ts+aD\cdot T_D}\cdot Dn-1$ $MVn=\Sigma\triangle MV\triangle$

符号说明

EVn　：本次采样时的偏差　　Dn：本次的微分项

EVn-1：1 个周期前的偏差　　Dn-1：1 个周期前的微分项

SV　：目标值　　Kp：比例增益

PVnf：本次采样时的测定值(滤波后)　　Ts：采样周期

PVnf-1：1 个周期前的测定值(滤波后)　　TI：积分常数

PVnf-2：2 个周期前的测定值(滤波后)　　TD：微分常数

△MV：输出变化量

MVn：本次的操作量

PVnf 是根据读入的测定值由下列运算式求得的值。

[滤波后的测定值 PVnf]=PVn+L(PVnf-1-PVn)

PVn：本次采样时的测定值

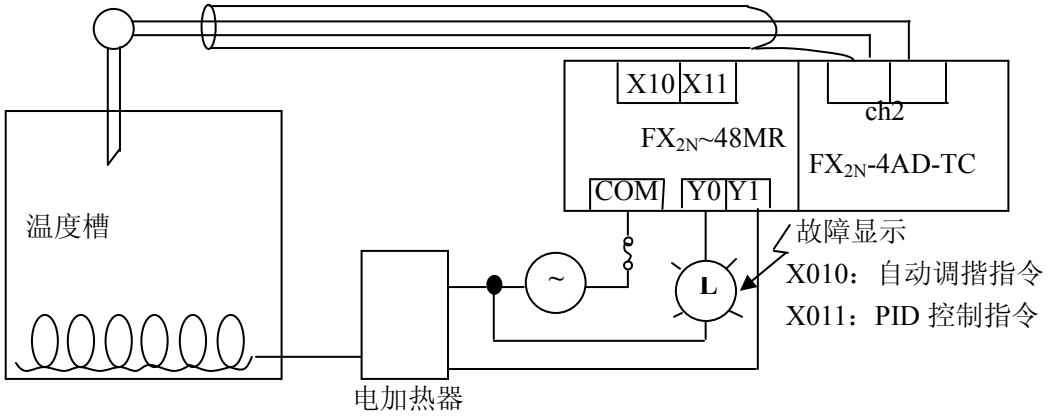
L：滤波系数

PVnf-1：1 个周期前的测定值(滤波后)

6.1.2 应用示例

①、系统构成：

温度传感器
(热电偶)　　带屏蔽的补偿导线



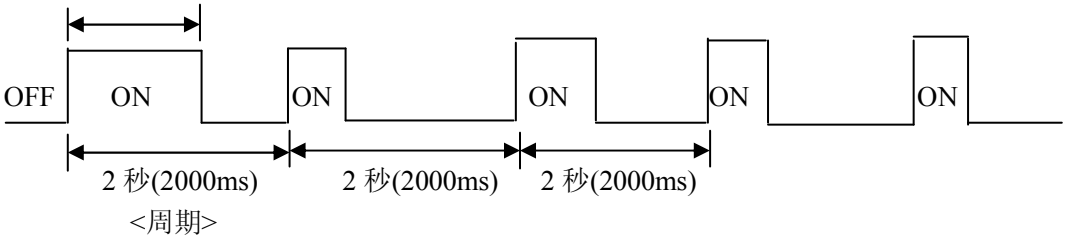
②、设定内容：

			自整定时	PID 控制时
目标值		<S1>	500(+50℃)	500(+50℃)
参数	采样时间(Ts)	<S3>	3000ms	500ms
	输入滤波(a)	<S3 +2>	70%	70%
	微分增益(KD)	<S3 +5>	0%	0%
	输出值上限	<S3 +22>	2000(2 秒)	2000
	输出值下限	<S3 +23>	0	0
	动作方向(ACT)	输入变化量报 <S3 +1bit1	无	无
		输出变化量报警 <S3 +1bit2>	无	无
		输出值上下限设定<S3 +1bit5>	有	有
输出值		<D>	1800	根据运算

③、电热器动作：

- <PID 控制时>

D502X1ms<ON 时间>



- <自动调谐时>　最大输出的 90%时