使用手册

之

PLC 编程篇

指令说明 1

第一章 顺序程序编制流程..... 1
1.1 PLC 规格..... 2
1.2 顺序程序的概念..... 2
1.3 分配接口..... 3
1.4 编制梯形图..... 3
1.5 PLC 指令表..... 3
1.6 调试顺序程序..... 3
第二章 顺序程序..... 1
2.1 顺序程序的执行过程..... 1
2.2 循环执行..... 2
2.3 执行的优先顺序（第一级，第二级）..... 2
2.4 顺序程序结构..... 3
2.5 输入／输出信号的处理..... 4
2.6 互锁..... 6
第三章 地址..... 1
3.1 机床→PLC 的地址（X）..... 1
3.2 PLC→机床的地址（Y）..... 2
3.3 PLC→CNC 的地址（G）..... 3
3.4 CNC→PLC 的地址（F）..... 3
3.5 内部继电器地址（R）..... 3
3.6 信息显示请求地址（A）..... 3
3.7 保持型继电器地址（K）..... 3
3.8 计数器地址（C）..... 3
3.9 计数器预置值地址（DC）..... 3
3.10 定时器地址（T）..... 4
3.11 计数器预置值地址（DT）..... 4
3.12 数据表地址（D）..... 4
3.13 标记地址（L）..... 4
3.14 子程序号（P）..... 4
第四章 基本指令..... 1
4.1 LD，LDI，OUT，OUTN 指令..... 1
4.2 AND，ANI 指令..... 2
4.3 OR，ORI 指令..... 2
4.4 ORB 指令..... 3
4.5 ANB 指令..... 4
第五章 功能指令..... 1
5.1 END1（第一级顺序程序结束）..... 1
5.2 END2（第二级顺序程序结束）..... 2
5.3 SET（置位）..... 2
5.4 RST（复位）..... 3
5.5 CMP（二进制数据比较）..... 3
5.6 TMRB（定时器）..... 4
5.7 CTRC（二进制计数器）..... 5
5.8 DECB（二进制译码）..... 6
5.9 CODB（二进制代码转换）..... 7
5.10 JMPB（标号跳转）..... 9
5.11 LBL（标号）..... 9
5.12 CALL（调用子程序）..... 10
5.13 SP（子程序）..... 11

5.14 SPE（子程序结束）.....11
5.15 ROTB（二进制旋转控制）.....12
5.16 MOVE（二进制逻辑乘数据传送）.....14
5.17 PARI（奇偶校验）.....15
5.18 ADDB（二进制数据相加）.....16
5.19 SUBB（二进制数据相减）.....16
5.20 DIVB（二进制数据相除）.....17
5.21 MULB（二进制数据相乘）.....18
5.22 MOVB（单字节数据传递）.....19
5.23 MOVW（双字节数据传递）.....19
5.24 MOVN（1、2、4 字节数据传递）.....20
5.25 COIN（比较相等）.....20
5.26 ORE（异或）.....21
5.27 ORF（或）.....23
5.28 ANDF（与）.....24
5.29 NOT（非）.....25
5.30 ALT（交替输出）.....26
5.31 SFT（寄存器移位）.....26
5.32 DSCH（数据检索）.....27
5.33 DIFU（上升沿检测）.....28
5.34 DIFD（上升沿检测）.....28
5.35 COM（公共线控制）.....29
5.36 COME（公共线控制结束）.....30
第六章 梯形图编辑限制.....1

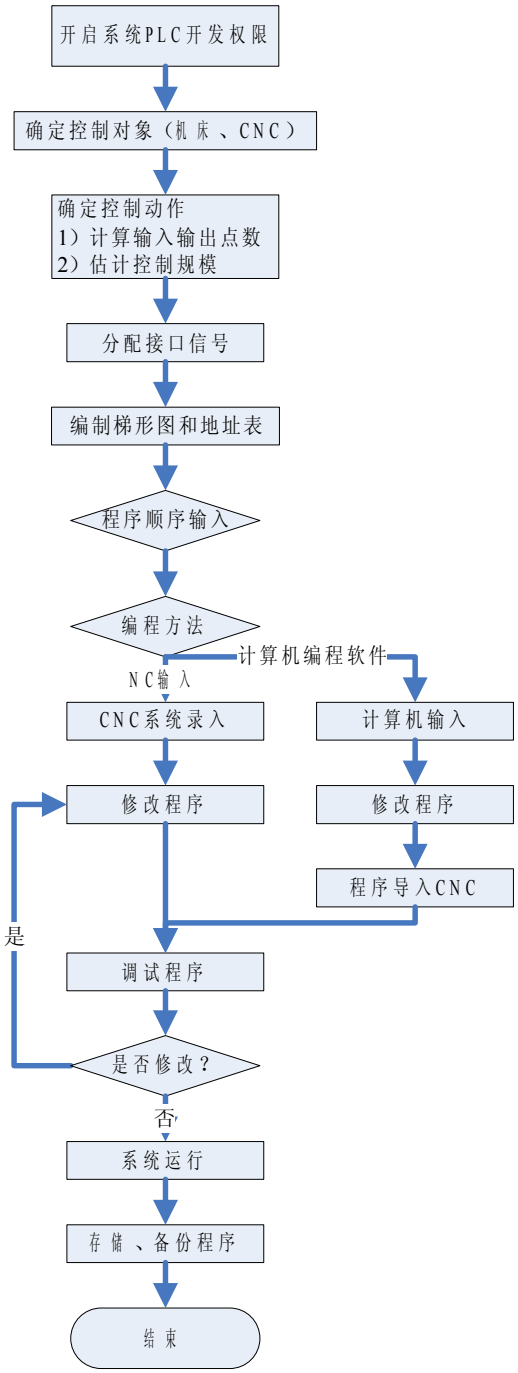
附 录 2

附录一 X 地址.....2
附录二 Y 地址.....5
附录三 F 地址.....8
附录四 G 地址.....9

指令说明

第一章 顺序程序编制流程

需要 PLC 控制的数控机床，一般的编制流程如下：



1.1 PLC 规格

不同系统的 PLC，其程序容量，处理速度，功能指令数量，非易失性存储区地址均不同。

CNC PLC 规格如下：

规格	CNC PLC
编程语言	Ladder
编程软件	VisualPLC.exe
文件格式	Ldx
程序级数	2
第一级程序执行周期	8ms
基本指令平均处理时间	3μs
程序容量	1000 步
第二级程序执行周期	1000 条/8ms
程序容量	4000 步
指令	基本指令＋功能指令
内部继电器 （R） 信息显示请求位 （A） 保持型存储区 ＊ 定时器 （T） ＊ 计数器 （C） ＊ 数据表 （D） ＊ 保持型继电器（K） ＊ 计数器预置值数据寄存器 （DC） ＊ 定时器预置值数据寄存器 （DT） 子程序 （P） 标号 （L）	1024 字节 32 字节 512 字节(128 个 4 字节) 512 字节(128 个 4 字节) 1024 字节(256 个 4 字节) 32 字节 512 字节(128 个 4 字节) 512 字节(128 个 4 字节) — —
I/O 模块 （X） （Y） （F） （G）	 64 字节 64 字节 256 字节 256 字节

1.2 顺序程序的概念

所谓顺序程序是指对机床及相关设备进行逻辑控制的程序。

在将程序转换成某种格式后，CPU 即可对其进行译码和运算处理，并将结果存储在 RAM 中。CPU 高速读出存储在存储器中的每条指令，通过算术运算来执行程序。

顺序程序的编制由编制梯形图开始，即把控制逻辑转换为相应的算术运算，通过特定的格式描述出来。

CNC 可通过 PC 版梯形图编辑器或者 CNC 的 PLC 软件进行顺序程序编辑。

1.3 分配接口

在确定了控制对象并计算出对应的输入 / 输出信号的点数后，即可分配接口。

在分配接口时，请参考系统说明书的输入 / 输出接口信号表。

1.4 编制梯形图

可选择 PC 机版梯形图编辑器或 CNC 的 PLC 在线编辑功能，用梯形图将机床所需的控制逻辑动作表示出来。对于无法用继电器符号表示的定时器、计数器等功能，用指定的功能指令符号表示。

为了缩短开发周期，建议使用系统自备的 PLC 程序。另外，为了方便广州数控公司、机床制造厂商和最终用户的维修人员理解，建议在编写 PLC 增加输入/输出地址、网络注释，CNC 允许网络注释可输入最多 64 个字符或 32 个中文汉字，地址最多允许 16 个字符或汉字。

编辑好的梯形图，需要进行校验和编译，转换成相应的 PLC 指令；若是在 PC 机上编辑，还需要导入到 CNC 中，以便 CNC 的读取与执行。

1.5 PLC 指令表

PLC 指令表是把梯形图的控制逻辑内容转换成另外一种系统识别的语言（格式）。系统读取和执行便是相应的指令编码。

一般情况下，只有高级用户才能在此基础上进行修改。

1.6 调试顺序程序

可用下列方法调试顺序程序：

1） 用仿真器调试

用一个仿真器（有灯和开关组成）替代机床。用开关的开和闭表示机床的输入信号状态，用灯的亮和灭来表示输出信号的状态。

2） 通过实际运行调试

在实际机床上调试。由于可能会发生意想不到的情况，因此在调试前应做好防范措施。

第二章 顺序程序

由于 PLC 顺序控制由软件来实现，所以和一般的继电器电路的工作原理不尽相同。
因此在设计 PLC 顺序程序时应充分理解顺序控制的原理。

2.1 顺序程序的执行过程

在一般的继电器控制电路中，各继电器在时间上完全可以同时动作，在下图所举例中，当继电器 A 动作时，继电器 D 和 E 可同时动作（当触点 B 和 C 都闭合时）。在 PLC 顺序控制中，各个继电器依次动作，当继电器 A 动作时，继电器 D 首先动作，然后继电器 E 才动作（见下图）。即各个继电器按梯形图中的顺序（编辑次序）动作。

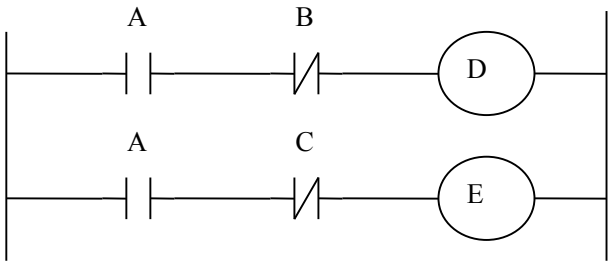


图 2.1(a)　电路举例

下图 2.1（b）和（c）图指出了继电器电路与 PLC 程序动作之间的区别。

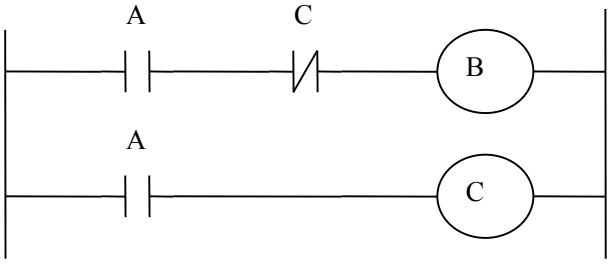


图 2.1(b)

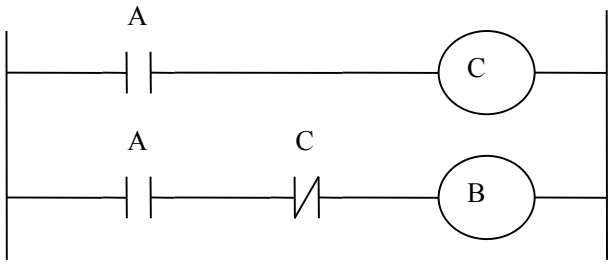


图 2.1(c)

(1) 继电器电路
图（A）和（B）中的动作相同。A 接通后，B 和 C 接通。C 接通之后 B 断开。

(2) PLC 程序

图（A）中，同继电器电路一样，A 接通后，B 和 C 接通。经过 PLC 程序的一个循环之后 B 断开。但图（B）中，接通 A 后，C 接通，但 B 并不接通。

2.2 循环执行

PLC 从梯形图的开头执行直至梯形图的结束。梯形图结束之后，再次从梯形图的开头重新开始执行。这被称作循环执行。

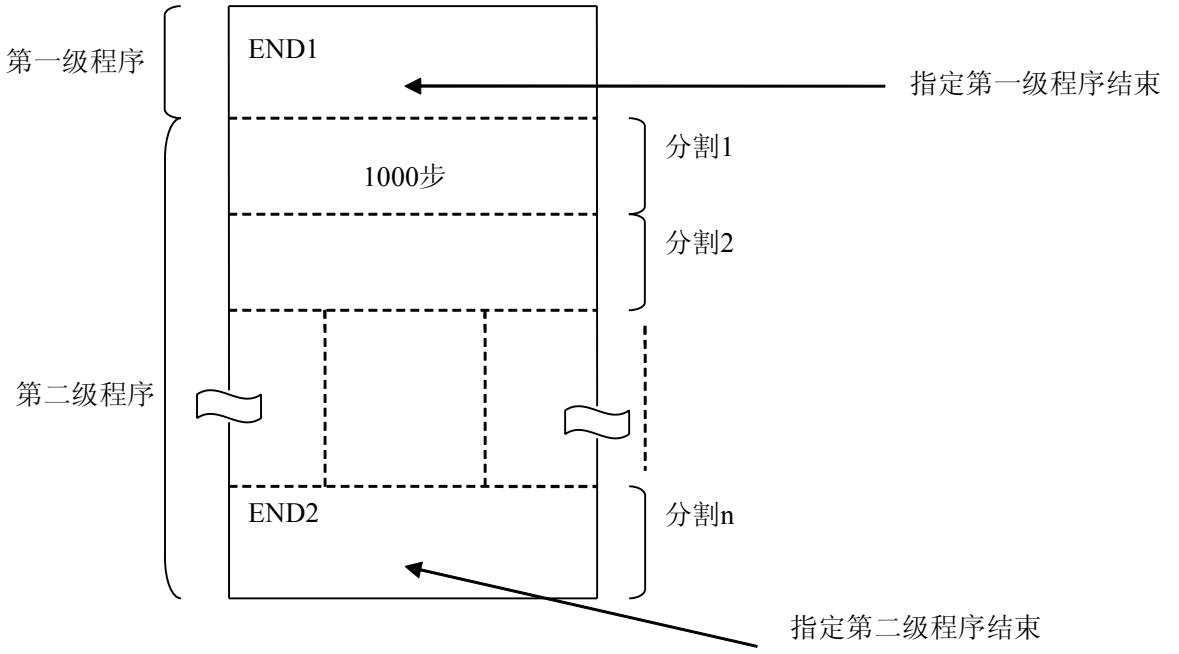
从梯形图的开头直至结束的执行时间简称为循环处理周期。处理周期越短，信号的响应能力就越强。

2.3 执行的优先顺序（第一级，第二级）

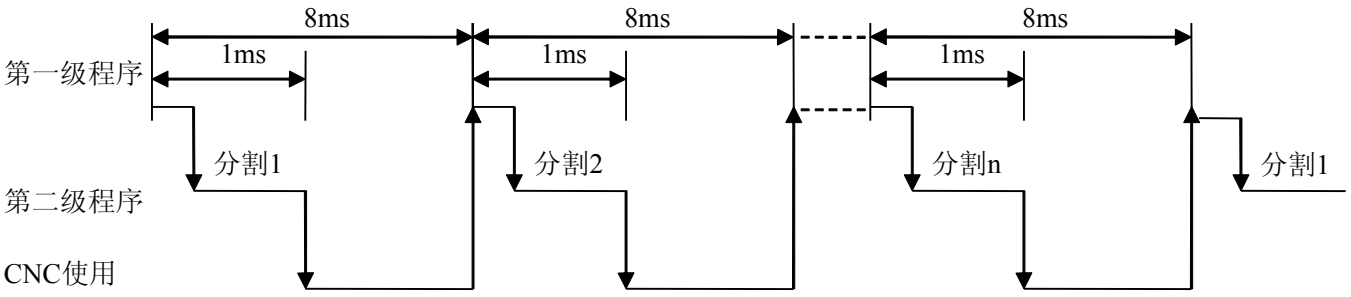
PLC 程序分为两部份：第一级程序和第二级程序。它们在执行周期上不一致。

第一级程序每 8ms 执行一次。可以处理一些要求响应快的短脉冲信号。

第二级程序每 8*n ms 执行一次。N 为第二级程序的分割数。在开始执行第二级程序时，PLC 会把二级程序分割成 1000 步/份。每个 8ms 只执行一份。



CNC 中，8ms 当中的 1ms 用于执行第一和第二级程序，剩余时间由 NC 使用。



当最后分割数为 n 的二级程序执行完后，程序又从头开始执行。这样当分割数为 n 时，一个循环的执行时间为 8*n ms。第一级程序每 8ms 执行一次，第二级程序每 8*n ms 执行一次。如果第一级程序的步数

增加，那么在 8ms 内第二级程序执行的步数就要相应的减少，这样分割数就要变多，整个程序的处理时间就要变长。因此，第一级程序应编得尽可能地短。

2.4 顺序程序结构

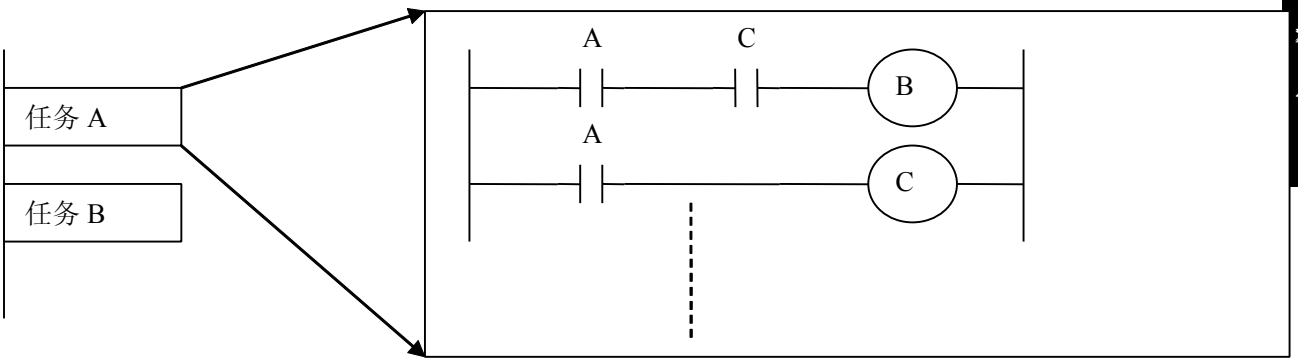
在传统的 PLC 中，梯形图顺序编制。而在允许结构化编程的梯形图语言中，具有以下优点：

- 程序易于理解，便于编制
- 更加方便找出编程错误
- 出现运行错误时，易于找出原因

主要的结构化编程方法有以下三种：

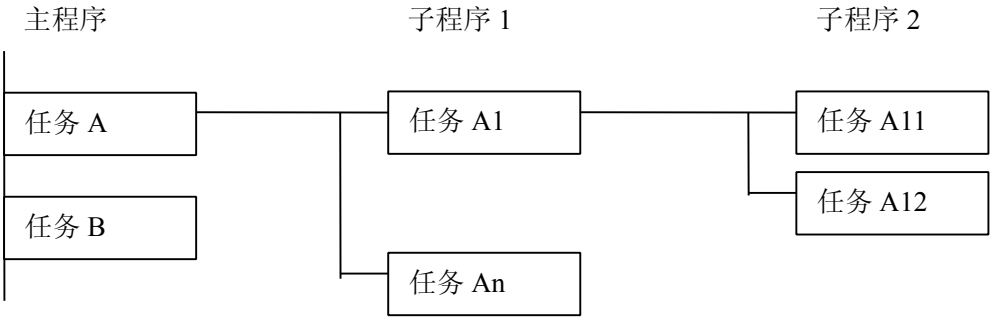
1） 子程序

子程序以梯形图作为处理单元。



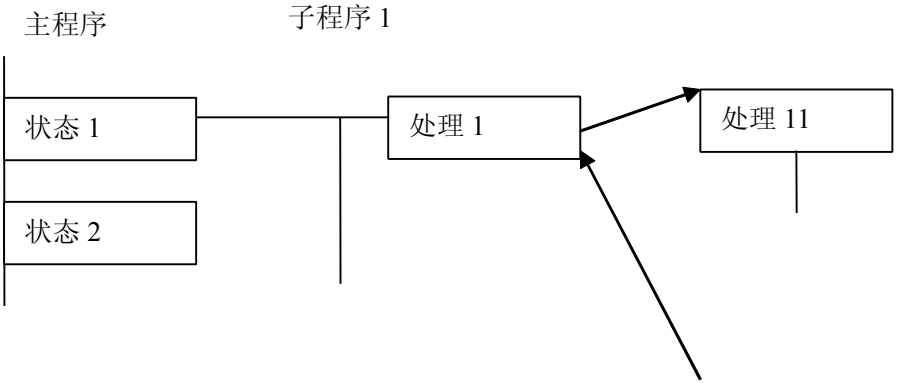
2) 嵌套

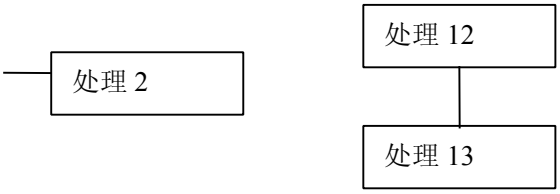
子程序可以调用其它子程序来完成任务。



3)条件分支

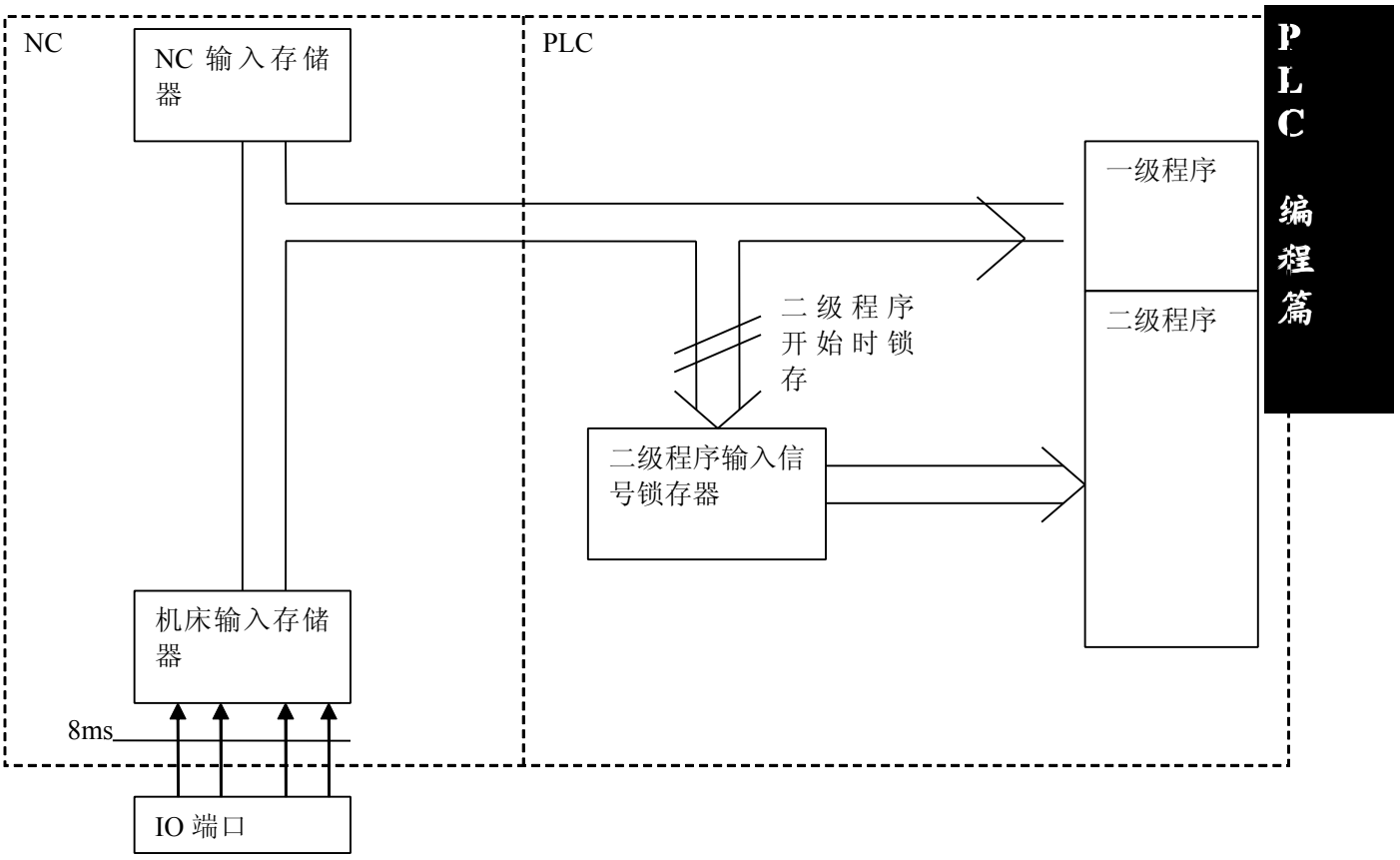
主程序循环执行并检测条件是否满足。如果条件满足，执行相应的子程序。如果条件不满足，不执行相应的子程序。



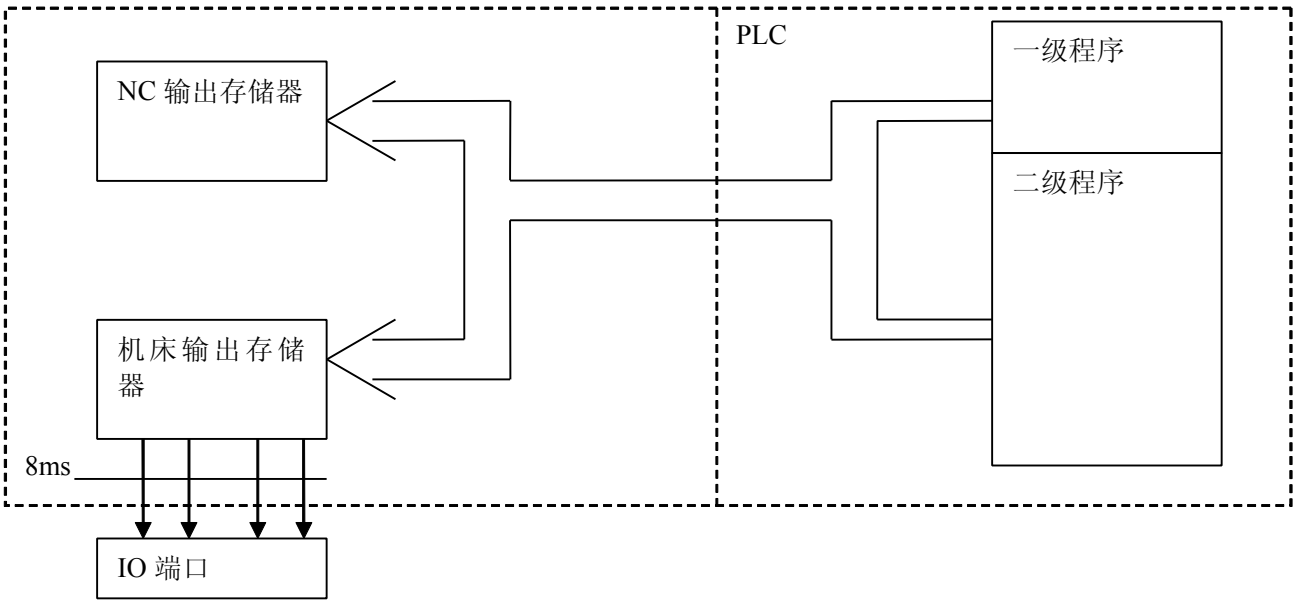


2.5 输入／输出信号的处理

输入信号处理：



输出信号处理：



2.5.1 输入信号处理

（1）NC 输入存储器

来自 NC 的输入信号存放在 NC 输入存储器中，每隔 8ms 传送至 PLC 中。一级程序直接引用这些信号的状态，执行相应的处理。

（2）机床输入存储器

机床输入存储器每隔 8ms 扫描和存储来自机床的输入信号。一级程序也是直接引用这些信号的状态，执行相应的处理。

（3）二级程序输入锁存器

二级程序输入信号锁存器，也叫二级程序同步输入信号存储器。其中存储的输入信号专门供二级程序处理。此存储器中的信号状态与第二级的信号状态是同步的。

只有在开始执行第二级程序时，NC 输入存储器和机床输入存储器中的信号才会被锁存到二级程序输入锁存器中。并且在整个第二级程序执行过程中，此锁存器中的信号状态保持不变。

2.5.2 输出信号的处理

（1）NC 输出存储器

输出信号每隔 8ms 由 PLC 传送至 NC 的输出存储器中。

（2）机床输出存储器

存储在机床输出存储器中的信号每隔 8ms 传送至机床。

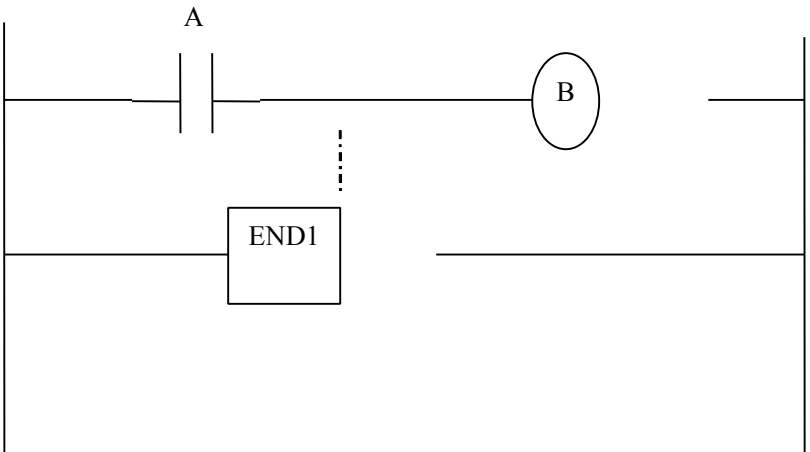
注 意

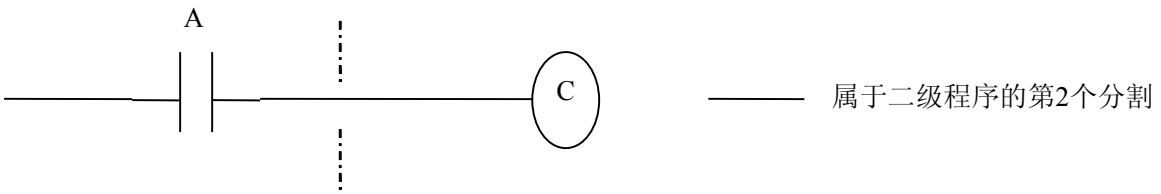
NC 输入存储器，NC 输出存储器，机床输入存储器，机床输出存储器的信号状态可用自诊断功能显示。诊断号就是顺序程序中的地址号。

2.5.3 第一级和第二级程序中信号状态的区别

同一个输入信号，在一级和二级程序中其状态也有可能不同。这是因为两级程序中使用不同的输入存储器。即“二级程序使用的输入信号是经锁存的一级程序的输入信号”。因此二级程序中的信号要比一级的输入信号滞后。在最坏的情况下，可滞后一个二级程序执行周期。

编制梯形图时应牢记这一点。





第一个 8ms 时，A＝1，执行一级程序 则 B＝1。且开始执行二级程序把 A＝1 锁存给二级程序，并执行二级程序的第一个分割。

第二个 8ms 时，A 变为了 0，执行一级程序，则 B＝0。接着执行二级程序的第二个分割，但此时 A 状态仍为上次锁存时的状态 1 。故 C＝1。

如此，B，C 的状态不相同。

2.6 互锁

在顺序控制中，从安全方面考虑，互锁是非常重要的。

在顺序控制程序中必须采取必要的互锁。同时在机床侧的强电柜的继电器控制电路中也应该采取必要的硬互锁。这是因为即使在顺序程序（软件）中逻辑上采取了互锁，但在执行顺序程序的硬件发生故障时，互锁会失效。因此，在机床侧的强电柜中采取互锁可保障操作者的安全并防止机床的损坏。

第三章 地址

地址用来区分信号。不同的地址分别对应机床测的输入 / 输出信号，CNC 侧的输入 / 输出信号，内部继电器，计数器，定时器，保持型继电器和数据表。每个地址由地址号和位号组成。其编号规则如下：

地址编号规则:

地址编号由地址类型、地址号、位号组成.

X 10 .6

类 型 地址号 位号

地址类型: 包括 X，Y，R，F，G，K，A

地址号 ：十进制编号，表示一个字节

位号 ：八进制编号，0～7 分别表示前面地址号代表的字节的 0～7 位

PLC 中的地址类型如下：

地 址	地址说明	长 度
X	机床→PLC(64)	INT8U
Y	PLC→机床(64)	INT8U
F	CNC→PLC(256)	INT8U
G	PLC→CNC(256)	INT8U
R	中间继电器(128)	INT8U
D	数据寄存器(256)	INT32U
C	计数器(128)	INT32U
T	定时器(128)	INT32U
DC	计数器预置值数据寄存器(0～127)	INT32S
DT	定时器预置值数据寄存器(0～127)	INT32S
A	信息显示请求信号 （32 字节）	INT8U
K	保持型继电器（32 字节）	INT8U
L	跳转标号 (L1~L9999)	
P	子程序标号 (P1~P9999)	

3.1 机床→PLC 的地址（X）

PLC 的 X 地址分为五部分：

1. X 地址分配于系统的 I/O 单元上。
2. X 地址分配于系统高速输入端口上。
3. X 地址分配于系统的 MDI 面板的输入按键上。
4. X 地址分配于伺服控制单元的输入状态。

5. X 地址分配于手脉功能地址。

3.1.1 I/O单元上的X地址

地址从 X8 到 X15。定义类型为：INT8U，共 8 个字节。它们分布在扩展的两个 I/O 单元输入端口。这些 I/O 口的 X 地址，用户可根据实际情况自行定义它们的信号含义，用来连接机床和编制对应的梯形图。

3.1.2 高速输入端口上的X地址

地址从 X20 到 X21。定义类型为：INT8U，共 2 个字节。用于系统高速信号的处理，响应周期小于 1ms，常用于高速跳转信号等。

3.1.3 MDI面板上的X地址

地址从 X0 到 X7，共 8 个字节。这些 X 地址与 MDI 面板上的按键输入一一对应。一般情况下，用户不能更改其中的信号定义。详细地址见附录一。

3.1.4 伺服控制单元的X地址

地址从 X16 到 X19 以及 X28 到 X35，共 12 个字节。这些 X 地址与伺服控制器上的状态一一对应。用户不能更改其中的信号定义。详细地址见附录一。

其中，X16 到 X19 为模拟量控制的地址；X28 到 X35 为总线伺服的地址。

3.1.5 手脉的X地址

地址从 X22，共 1 个字节。这些 X 地址与外挂手脉地址定义对应。一般情况下，用户不能更改其中的信号定义。详细地址见附录一。

3.2 PLC→机床的地址（Y）

PLC 的 Y 地址分为两类：

1. Y 地址分配于系统 I/O 单元的输出口上。
2. Y 地址分配于系统的 MDI 面板上的各个提示灯上。
3. Y 地址分配于伺服控制单元的输出状态。

3.2.1 I/O输出口上的Y地址

地址从 Y8 到 Y15。定义类型为：INT8U，共 8 个字节。它们分布在扩展的两个 I/O 单元输入端口。这些 I/O 口的 Y 地址，用户可根据实际情况自行定义它们的信号含义，用来连接机床和编制对应的梯形图。

3.2.2 MDI面板上的Y地址

地址从 Y0 到 Y7，共 8 个字节。这些 Y 地址与 MDI 面板上的提示灯一一对应。用户不能更改其中的信号定义。详细地址见附录。

3.2.3 伺服控制单元上的Y地址

地址从 Y24 到 Y31，共 8 个字节。这些 Y 地址与伺服控制器上的状态一一对应。用户不能更改其中的信号定义。详细地址见附录。

3.3 PLC→CNC 的地址（G）

地址从 G0 到 G255。定义类型为：INT8U，共 256 个字节。

3.4 CNC→PLC 的地址（F）

地址从 F0 到 F255。定义类型为：INT8U，共 256 个字节。

3.5 内部继电器地址（R）

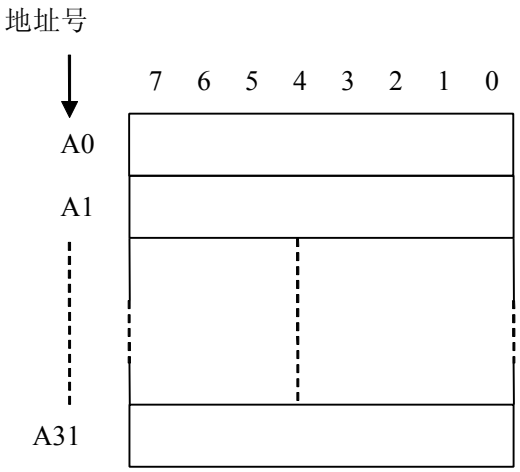
此地址区域在系统上电时被清零。

定义类型为：INT8U，共 128 个字节。

3.6 信息显示请求地址（A）

此地址区域在系统上电时被清零。

定义类型为：INT8U，共 32 个字节。



3.7 保持型继电器地址（K）

此地址区域用作保持型继电器和设定 PLC 参数。此区为非易失性存储区域，即使系统掉电，存储器中的内容也不会丢失。0~7 字节是可以在梯形图上作用输出。

定义类型为：INT8U，共 32 个字节。

3.8 计数器地址（C）

此地址区域用来存放计数器当前计数值。。

定义类型为：INT32U，共 512 个字节。

3.9 计数器预置值地址（DC）

此地址区域用来存放计数器预置值。此区为非易失性存储区域，即使系统掉电，存储器中的内容也不会丢失。

定义类型为：INT32S，共 512 个字节。

输入数值范围： 2147483648~2147483647

3.10 定时器地址（T）

此地址区域用来存放定时器当前数值。此地址区域在系统上电时被清零。

定义类型为：INT32U，共 512 个字节。

3.11 计数器预置值地址（DT）

此地址区域用来存放定时器预置值。此区为非易失性存储区域，即使系统掉电，存储器中的内容也不会丢失。

定义类型为：INT32U，共 512 个字节。

输入数值范围：-2147483648~2147483647。

3.12 数据表地址（D）

D000～D063 为非易失性存储区域，即使系统掉电，存储器中的内容也不会丢失。D064～D255 地址在系统上电时被清零。

定义类型为：INT32U，共 1024 个字节。

3.13 标记地址（L）

用来指定 JMPB 指令中的跳转目标标号和 LBL 指令的标号。

范围：L0～L9999

3.14 子程序号（P）

用来指定 CALL 指令中调用的目标子程序号和 SP 指令的子程序号。

范围：P0～P9999

第四章 基本指令

顺序程序的设计从编制梯形图开始。梯形图由继电器触点，功能指令构成。梯形图中所表示的逻辑关系构成顺序程序。输入顺序程序的方法有两种：一种输入方法使用助记符语言（LD，AND，OR 的 PLC 指令)；别一种方法使用继电器符号。使用继电器符号，可以使用梯形图格式并且不用理解 PLC 指令格式即可进行编程。

实际上，即使顺序程序由继电器符号方法输入，在系统内部也被转换成相应的 PLC 指令。

基本指令是设计顺序程序时最常用到的指令，它们执行一位运算。

CNC 的基本功能指令如下：

指令名	功能	可操作元件
LD	读取常开触点状态	X，Y，F，G，R，K，A
LDI	读取常闭触点状态	X，Y，F，G，R，K，A
OUT	驱动输出线圈	Y，G，R，K，A
AND	常开触点串联	X，Y，F，G，R，K，A
ANI	常闭触点串联	X，Y，F，G，R，K，A
OR	常开触点并联	X，Y，F，G，R，K，A
ORI	常闭触点并联	X，Y，F，G，R，K，A
ORB	串联电路的并联	无
ANB	并联电路块的串联	无

4.1 LD，LDI，OUT，OUTN 指令

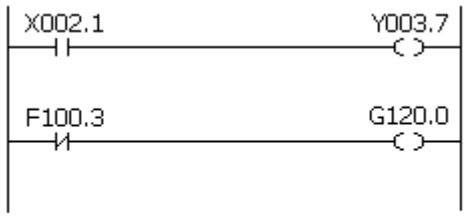
助记符与功能

助记符	功能	梯形图符号
LD	读取常开触点状态	
LDI	读取常闭触点状态	
OUT	驱动输出线圈	
OUTN	驱动输出取反线圈	

指令说明

- LD，LDI 指令用于将触点连接到母线上。其他用法与后述的 ANB 指令组合，在分支起点处也可使用。
- OUT 指令是对输出继电器，内部继电器的线圈驱动指令。对输入继电器不能使用。
- 并列的 OUT 命令能多次连续使用。

编程




```
LD    X002.1
OUT   Y003.7
LDI   F100.3
OUT   G120.0
```


4.2 AND，ANI 指令

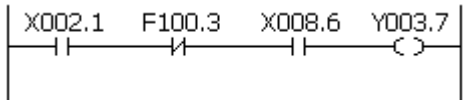
助记符与功能

助记符	功能	梯形图符号
AND	常开触点串联	
ANI	常闭触点串联	

指令说明

- 用 AND，ANI 指令可串联连接 1 个触点。串联触点数量不受限制，该指令可多次使用。

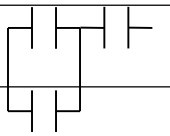
编程

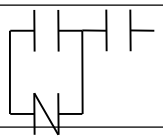



```
LD    X002.1
ANI   F100.3
AND   X008.6
OUT   Y003.7
```

4.3 OR，ORI 指令

助记符与功能

助记符	功能	梯形图符号
OR	常开触点并联	

ORI	常闭触点并联	

指令说明

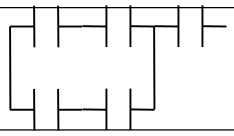
- 用 OR，ORI 指令可并联连接 1 个触点。如果有两个以上的触点串联连接，并将这种串联回路块与其他回路并联连接时，采用后述的 ORB 指令。
- OR，ORI 是指从该指令的步开始，与前述的 LD，LDI 指令步，进行并联连接。

编程


```
LD    X002.1
ORI   F100.3
OUT   Y003.7
```

4.4 ORB 指令

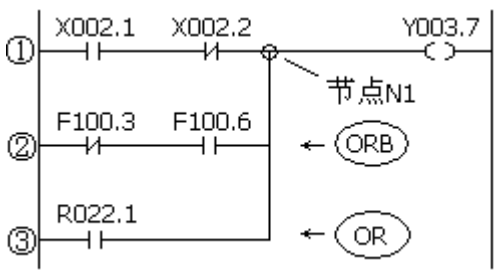
助记符与功能

助记符	功能	梯形图符号
ORB	串联电路的并联	

指令说明

- 由两个以上的触点串联连接的回路被称为串联回路块。将串联回路块并列连接时，分支开始用 LD，LDI 指令，分支结束用 ORB 指令。
- ORB 指令是不带地址的独立指令。

编程




```
LD    X002.1
ANI   X002.2
LDI   F100.3
AND   F100.6
ORB
OR    R022.1
```

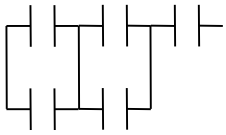
```
OUT   Y003.7
```

如图从左边母线至节点 N1 有三条支路①，②，③，支路①和②都为串联电路块，当母线至节点或节点与节点间有并联的串联电路块时，除第一个分支，在以后的分支结束使用 ORB 指令。支路③不是串联电路块，用 OR 指令即可。

ORB 和 ANB 为无操作元件的指令，表示电路块间的或、与关系。

4.5 ANB 指令

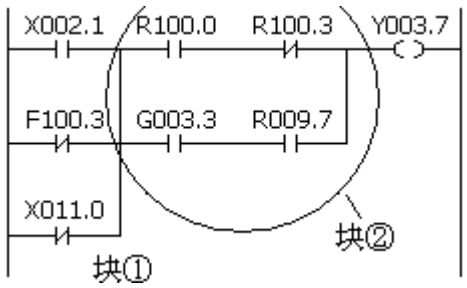
助记符与功能

助记符	功能	梯形图符号
ANB	并联电路的串联	

指令说明

- 当分支回路（并联回路块）与前面的回路串联连接时，使用 ANB 指令。分支的起点用 LD，LDI 指令，并联回路块结束后，使用 ANB 指令与前面的回路串联连接。
- ANB 指令是不带地址的独立指令。

编程




```
LD    X002.1
ORI   F100.3
ORI   X011.0
LD    R100.0
ANI   R100.3
LD    G003.3
AND   R009.7
ORB                ← (1)
ANB                ← (2)
OUT   Y003.7
```

如上梯形图及指令表，(1)ORB 表示块②中的串联电路块并联，(2)ANB 表示电路块①与电路块②的串联。

第五章 功能指令

在用基本指令难于编制某些机床动作时，可使用功能指令来简化编程。

CNC 功能指令如下：

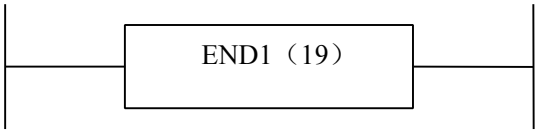
序号	指令名	功　能
1	END1	一级程序结束
2	END2	二级程序结束
3	END	整个 PLC 程序结素
4	SET	置位
5	RST	复位
6	CMP	二进制数据比较
7	TMRB	定时器
8	CTRC	二进制计数器
9	MOVN	任意数据字节传送
10	DECB	二进制译码
11	CODB	二进制转换
12	JMPB	条件跳转
13	LBL	标号
14	CALL	调用子程序
15	SP	子程序开始
16	SPE	子程序结束
17	ROTB	二进制旋转控制
18	MOVE	逻辑乘数据传送
19	MOVOR	逻辑或数据传送
20	PARI	奇偶校验
21	COMP	数值大小判别
22	COIN	一致性检测
23	SFT	寄存器移位
24	DSCH	二进制数据检索
25	MOVB	1 字节数据传送
26	MOVW	2 字节数据传送
27	XMOV	变址数据传送
28	ADD	二进制加法
29	SUB	二进制减法
30	ANDF	逻辑与
31	ORF	逻辑或
32	EOR	异或
33	NOT	逻辑非
34	DIFU	上升沿触发
35	DIFD	下降沿触发

5.1 END1（第一级顺序程序结束）

功能

在顺序程序中必须给出一次，可在第一级程序末尾，或当没有第一级程序时，排在第二级程序开头。

图形格式



指令格式

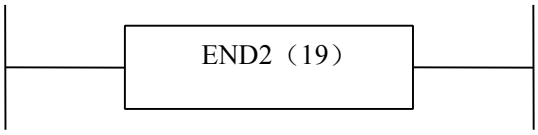
END1

5.2 END2（第二级顺序程序结束）

功能

在第二级程序末尾给出。

图形格式



指令格式

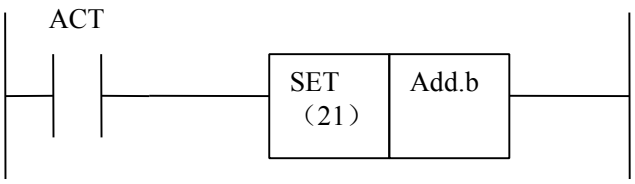
END2

5.3 SET（置位）

功能

在指定地址上置 1。

图形格式



指令格式

SET	Add.b

控制条件

ACT＝0，add.b 的状态保持不变。

ACT＝1，add.b 置 1。

参数

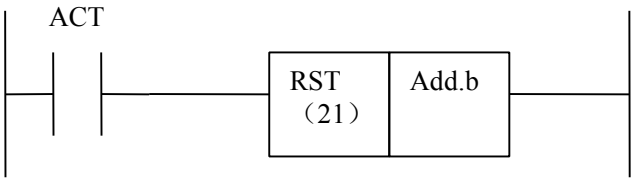
Add.b：置位元件地址位，可以为触点、输出线圈， Add= Y，G，R，A,K。

5.4 RST（复位）

功能

在指定地址上置 0。

图形格式



指令格式

RST	Add.b

控制条件

ACT＝0，add.b 的状态保持不变。

ACT＝1，add.b 置 0。

参数

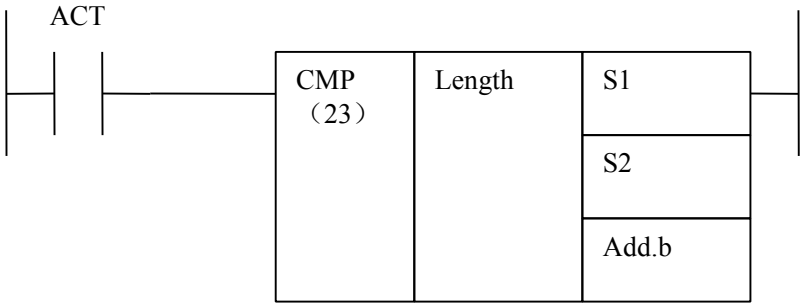
Add.b：复位元件地址位，可以为触点、输出线圈， Add= Y，G，R， A,K。

5.5 CMP（二进制数据比较）

功能

比较两个二进制数据的大小，输出比较结果。

图形格式



指令格式

CMP	Length	S1	S2	Address.b

控制条件

ACT＝0，address.b 保持原值

ACT＝1，比较 S1，S2 的大小。其输出结果如下：

	address.(b+2)	address.(b+1)	address.(b+0)
S1>S2	0	0	1
S1=S2	0	1	0
S1<S2	1	0	0

参数

length， 指定数据长度

1：1 字节

2：2 字节

S1、S2， 比较源 1 和比较源 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D,DT、DC,T,C 类。

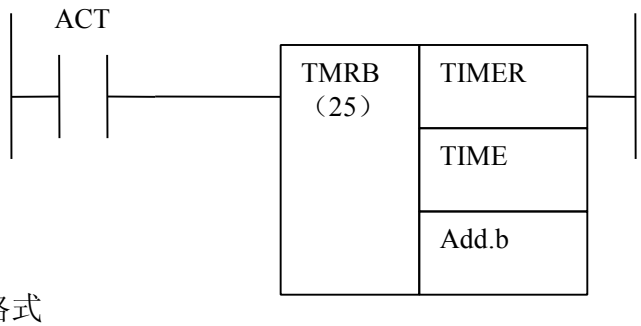
address.b ， 为比较输出结果。可为 R，Y，G， A ,K 类。

5.6 TMRB（定时器）

功能

延时导捅定时器。

图形格式



指令格式

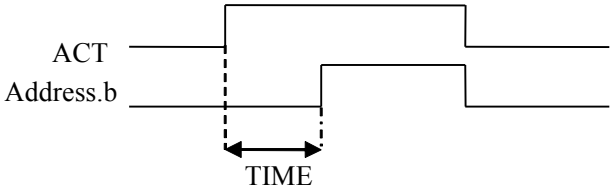
TMRB	TIMER	TIME	Address.b

控制条件

ACT＝0，TIMER 与 address.b 复位。

ACT＝1，TIMER ，从 0 开始计时。且到达 TIME 预置时间时 ，address.b=1。

具体工作情况如下：



参数

TIMER : 定时器编号，以 Txxx 表示，xxx 为数字(0～127)。

TIME : 定时常数或以 DT 开头的数据寄存器
(以 10ms 为单位)

address.b ： 定时器输出地址。可为 R，Y，G， A ,K 类。

注：

定时器 TIMER，每 8ms 执行一次，以 8ms 为单位计时。

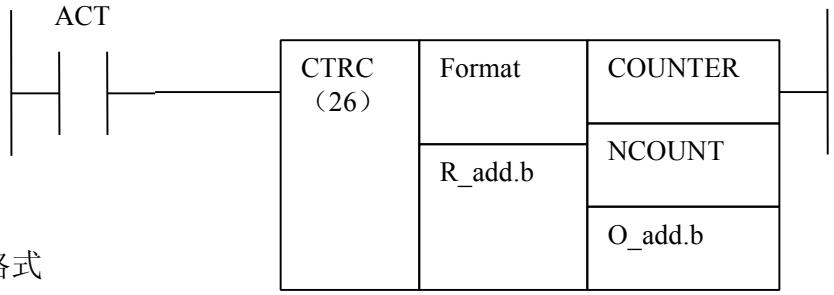
5.7 CTRC（二进制计数器）

功能

此计数器中的数据都是二进制的，根据应用情况有下列功能。

1) 预置型计数器
对计数值进行预置，如果计数达到预置值输出信号。
2) 环形计数器
计数器到达预置值时，输入计数信号，复位到初始值，并重新计数。
3) 加，减计数器
这是可逆计数器，既可用于做加，也可用于做减。
4) 初始值的选择
初始值可为 0 或 1.

图形格式



指令格式

CTRC	Format	R_add.b	COUNTER	NCOUNT	O_add.b

控制条件

ACT，为上升沿时：

加计数时，COUNTER ，从现有值开始计数，到达 NCOUNT 预置计数值时 ，O_add.b =1。而 COUNTER 与 NCOUNT 不等是，O_add.b =0； 且 COUNTER 大于 NCOUNT 预置计数值时，COUNTER 再次恢复初始值 CN0，开始计数。

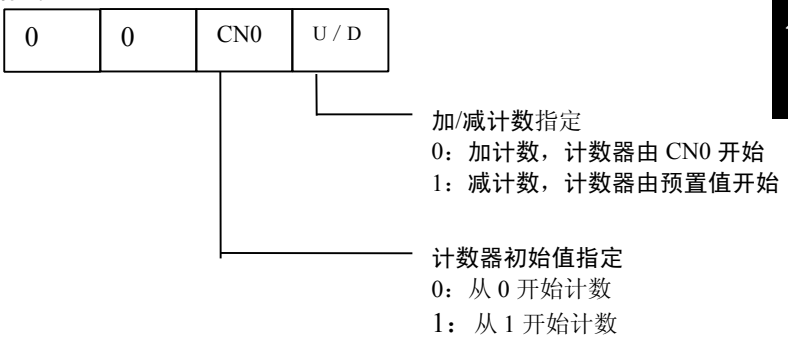
减计数时，计数器 从现有值开始减计数，到达 CN0 时，O_add.b =1；不然 O_add.b =0；且 COUNTER 小于初始值 CN0 时，COUNTER 恢复 NCOUNT 预置值，重新开始计数。

ACT＝0 时：

COUNTER 与 O_add.b 保持原值

参数

Format ： 数据格式



R_add.b ： 为 1 时，不论 ACT 为何状态，COUNTER＝CN0；
O_add.b =0。可为：X，Y，G，F，R，K，A。

COUNTER ： 指定计数器编号，以 Cxxx 表示，xxx 为数字(0～127)

NCOUNT ： 计数器预置值，可为常数值，也可为以 DC 开头的数据寄存器. 若为常数值。

O_add.b ： 到达计数值时输出位置 1。可为 R，Y，G， A ,K 类。

＊ 计数器是在上升沿时，进行计数。

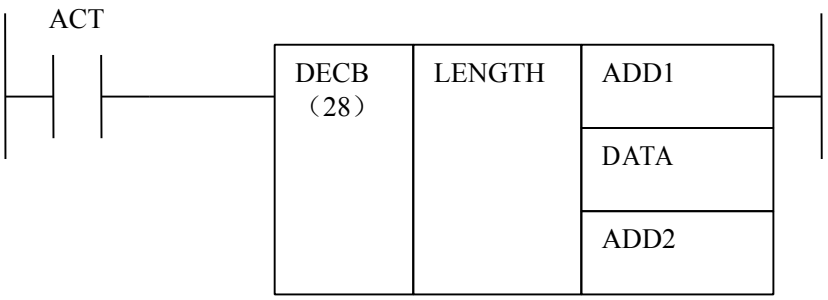
5.8 DECB（二进制译码）

功能

DECB 可对二进制代码数据译码，所指的八位连续数据之一与代码数据相同时，对应的输出数据位为 1。没有相同的数时，输出数据为 0。

此指令用于 M 或 T 功能的数据译码。

图形格式



指令格式

DECB	LENGTH	ADD1	DATA	ADD2

控制条件

ACT＝0，ADD2 的 8 个数据位全复位。。

ACT＝1，把译码地址（ADD1）的内容值，与以 DATA 为开头的 8 个连续的数据相比较。若 ADD1 的内容值与 8 个数据中的任一个相等时，而此相等的数据在这 8 个数据中排在第几位，则输出地址（ADD2）对应的第几位，将被置 1。

参数

length， 指定 ADD1 地址的长度（1，2，4 字节）

ADD1 : **译码起始地址**。地址号为，R，X，Y，F，G，K，A，D,DT,DC,T,C 类

DATA : **比较常数的基值**

ADD2 : **比较结果输出**. 。地址号为：R，Y，G， A,K,T,C 类

如：

DECB	1	F10	8	R4

当 ACT＝1，F10＝8 时， R4＝0000，0001；

当 ACT＝1，F10＝9 时， R4＝0000，0010；

……

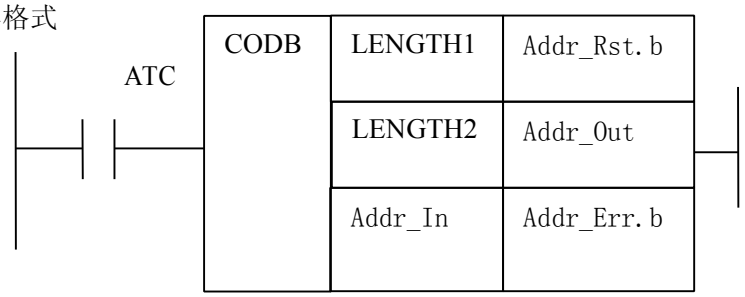
当 ACT＝1，F10＝15 时， R4＝1000，0000；

5.9 CODB（二进制代码转换）

功能

此指令用于二进制数据的转换。

图形格式



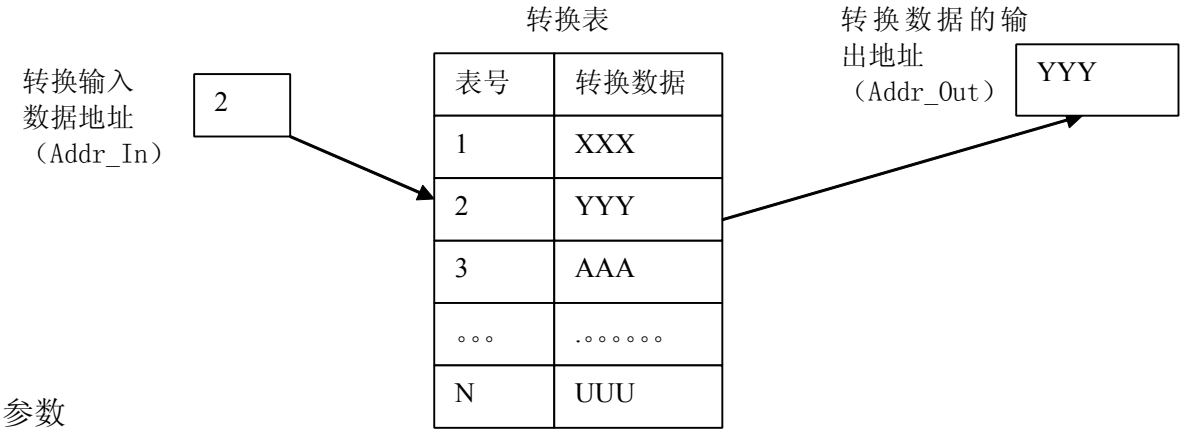
指令格式

CODB	LENGTH1	LENGTH2	Addr_In	Addr_Out	Addr_Rst.b	Addr_Err.b	D1	D2	…..	Dn

控制条件

ACT＝0，Addr_Out 中的值保持不变。

ACT＝1，以“转换输入数据地址（Addr_In）“的值作为转换表的表号，从转换表中取出该表号对应的转换数据，输出给转换数据的输出地址(Addr_Out)。输入转换数据的值超过表号时，输出错误地址 Addr_Err.b 为 1，Addr_Rst.b 复位地址为 1 时，Addr_Err.b 地址才清零.



参数

length1， 转换表中转换数据的二进制数据长度和转换数据的输出地址长度

1：1 字节

2：2 字节

4：4 字节

length2， 转换表的长度

1：2 个

2：4 个

3：8 个

4：16 个

5：32 个

6：64 个

7：128 个

8：256 个

Addr_In：转换数据的输入地址。此地址只需一个字节的数据。地址为 R，X，Y，G，F， A， K，D,T,C,DT,DC 类。

Addr_Out：转换数据的输出地址。地址为 R， Y，G， A， D,K、T、C 类。

Addr_Rst.b：为1 时，Addr_Err.b 复位为零，Addr_Out 不变。地址号为R、X、Y、F、G、A 以及K 等。

Addr_Err.b：运算结果错误输出地址，地址可为 Y、G、R、A ,K。

如：

CODB	1	3	F20	R30	F129.7	Y0.0	12	22	32	42	52	62	72	82

当

ACT＝1, F20＝1 时 ： R30＝12

ACT＝1, F20＝2 时 ： R30＝22

……..

ACT＝1, F20＝8 时 ： R30＝82

ACT＝1, F20＝9 时 ：R30 不变，Y0.0 为 1，

F129.7=1，Y0.0=0；其他不变

5.10 JMPB（标号跳转）

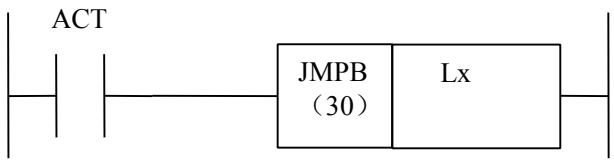
功能

JMPB 立即将控制转移至设置在梯形图程序中的标号后的程序。

除外还有以下特点与限制：

* 多条跳转指令可使用同一标号。
* 跳过 END1 和 END2 是禁止的。
* 跳出子程序也是禁止的。
* 可上跳也可下跳。

图形格式



指令格式

JMPB	Lx

控制条件

ACT＝0，不跳转，执行 JMPB 后的下一条指令。。

ACT＝1，跳转到指定标号后，执行标号后的下一条指令。

参数

Lx ： 指定跳转的目的标号。标号数必须以 L 地址开头指定。可指定由 L1 至 L9999 的一个值。

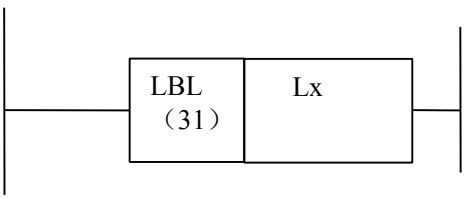
5.11 LBL（标号）

功能

在梯形图中指定一标号，即为 JMPB 指定跳转的目的地。

要注意的是 ：一个 Lx 标号，只能用 LBL 指定一次。多则报警。

图形格式



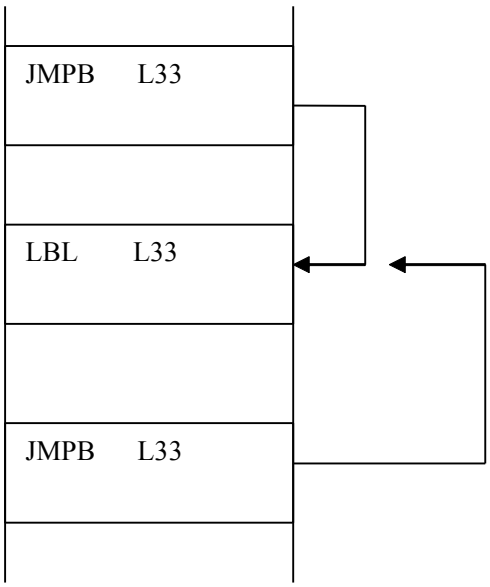
指令格式

LBL	Lx

参数

Lx ： 指定跳转的目的标号。标号数必须以 L 地址开头指定。可指定由 L1 至 L9999 的一个值。

例：



5.12 CALL（调用子程序）

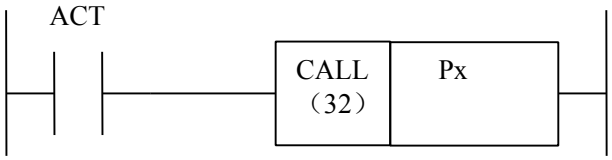
功能

调用一指定子程序。

除外还有以下特点与限制：

- ＊ 多条调用指令可调用同一子程序。
- ＊ 调用指令可嵌套。
- ＊ 不能在第一级程序中调用子程序。
- ＊ 子程序必须在 END2，之后编写。

图形格式



指令格式

CALL	Px

控制条件

ACT＝0，执行 CALL 后的下一条指令。。

ACT＝1，调用指定子程序号的子程序。

参数

Px ： 指定调用的子程序标号。子程序标号数必须以 P 地址开头指定。可指定由 P1 至 P9999 的一个值。

5.13 SP（子程序）

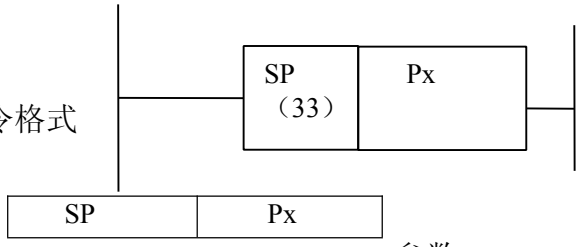
功能

SP 用来生成一个子程序。子程序号作为子程序的名称。 SP 指令与后述的 SPE 指令一道使用来指定子程序的范围。

要注意的是 ：子程序必须在 END2，之后编写。

图形格式

指令格式



参数

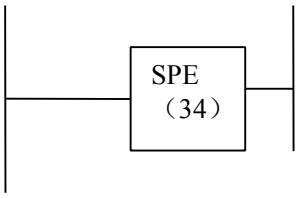
Px ： 指定调用的子程序标号。子程序标号数必须以 P 地址开头指定。可指定由 P1 至 P9999 的一个值

5.14 SPE（子程序结束）

功能

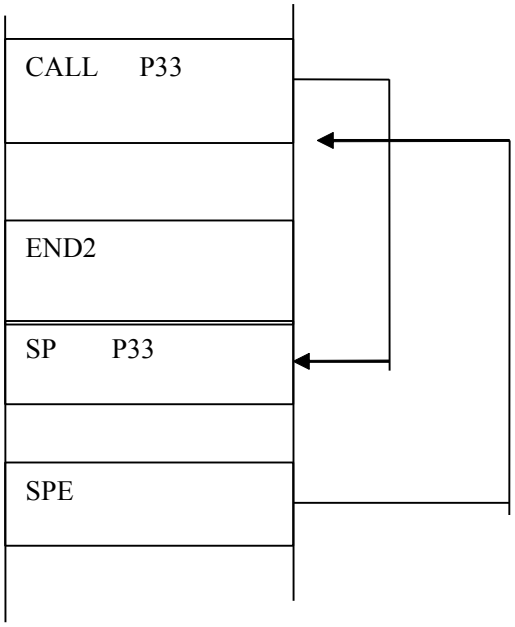
* SPE 与 SP 一起使用，指定子程序的范围。
* 当此功能指令被执行时，控制将返回到调用此子程序的主程序中。
* 子程序必须在 END2，之后编写。

图形格式



指令格式

例：



5.15 ROTB（二进制旋转控制）

功能

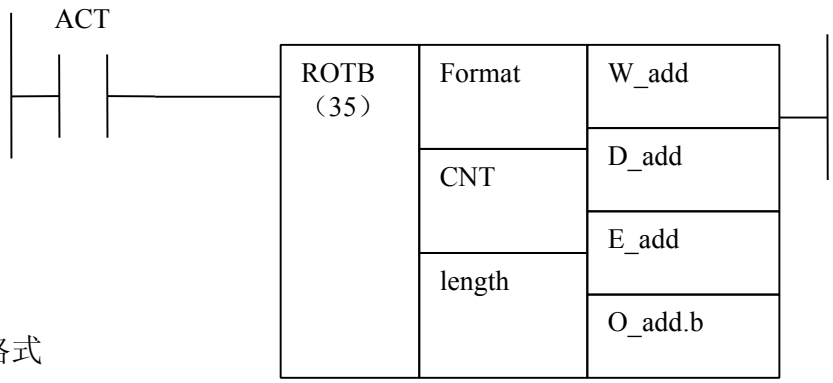
用于回转控制，如刀架，旋转工作台等。指令有如下功能：

× 选择短路径的回转方向。

× 计算由当前位置到目标位置的步数；或计算由当前位置的前一位置到目标位置的前一位置的步数

× 计算目标前一位置的位置号。

图形格式



指令格式

ROTB	Format	CNT	length	W_add	D_add	E_add	O_add.b

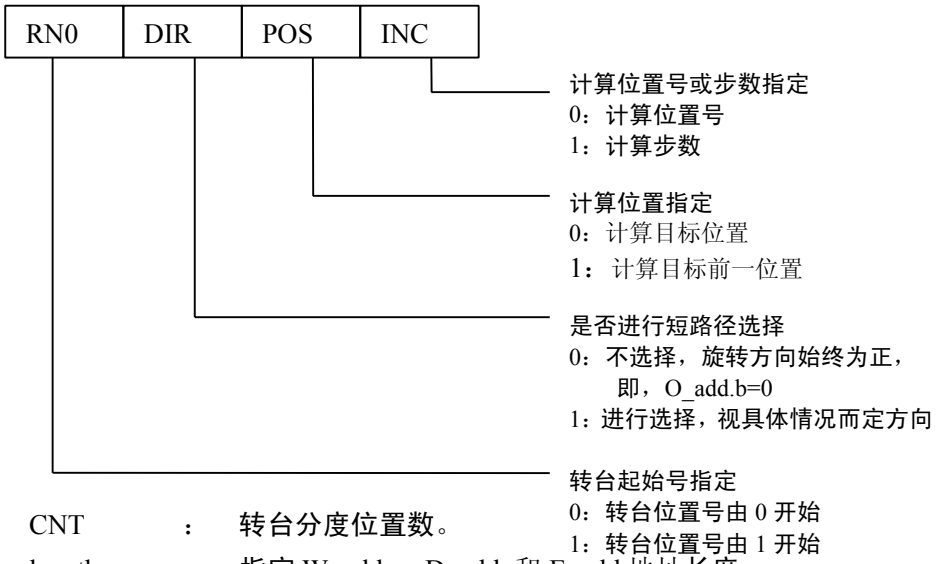
控制条件

ACT＝0，不执行指令，E_add 与 O_add.b 保持原值。

ACT＝1，执行指令，结果输出至 E_add 和 O_add.b 中。

参数

Format ： 数据格式



CNT ： 转台分度位置数。

length ： 指定 W_add ，D_add 和 E_add 地址长度（1，2，4 字节）。

W_add ： 当前位置地址，存放当前位置号。地址号为，R，X，Y，F，G，K，A，D,DT,DC,D,T 类。

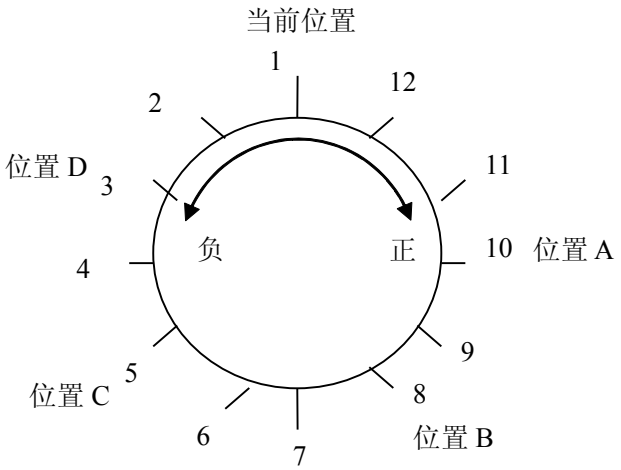
D_add ： 目标位置地址，存放目标位置号。地址号为，R，X，Y，F，G，K，A，D,DT,DC,D,T 类。

E_add ： 计算结果输出地址。地址号为，R，Y，G， A，D,K、T、C 类。

O_add.b ： 旋转方向输出。定义：使转台的位置号增加的方向为，正方向（FOR）；若减少为反方向（REV）。则，当 O_add.b＝0 时，为正向旋转；O_add.b＝1 时，为反向旋转。地址号为，R，Y，G， A 类。

例：

有一转台如下:



控制指令为，

ROTB	1110	12	1	R7	F26	R27	R37.0

进行短路径旋转，计算目标位置的前一位置的位置号。

当前位置号 R7＝1 ，转台分度位置数＝12，

则 ，

F26＝10 目标位置为 A 时 ，在 ACT＝1 下，R27＝11，R37.0＝1

F26＝8 目标位置为 B 时 ，在 ACT＝1 下，R27＝9 ，R37.0＝1

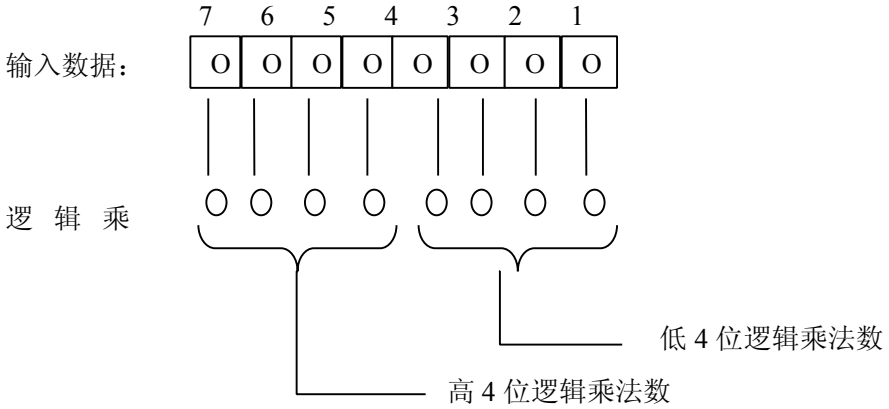
F26＝5 目标位置为 C 时 ，在 ACT＝1 下，R27＝4 ，R37.0＝0

F26＝3 目标位置为 D 时 ，在 ACT＝1 下，R27＝2 ，R37.0＝0

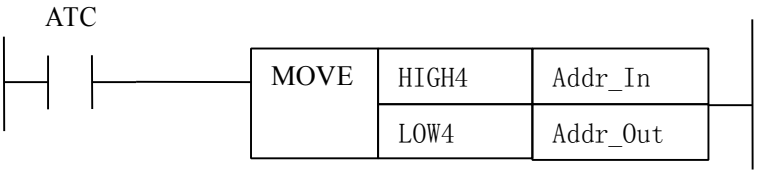
5.16 MOVE（二进制逻辑乘数据传送）

功能：

将逻辑乘数与输入数据进行逻辑乘，将结果输出至指定地址，还可用来从指定地址中一个八位信号中排出不需要的位数。



梯图形模式；

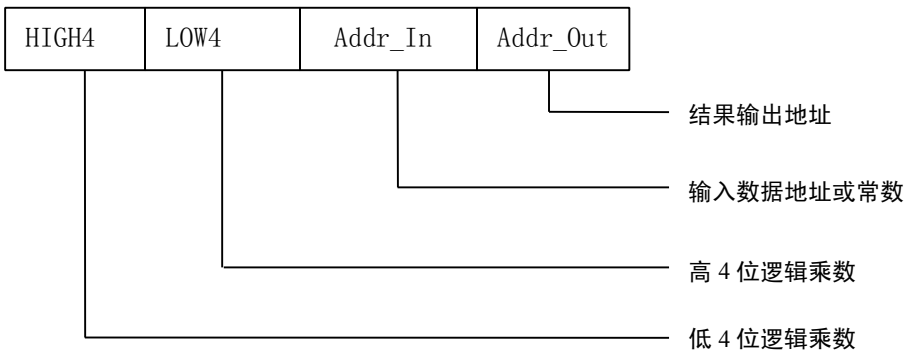


控制条件

当ACT=1 时：将逻辑乘数(HIGH4、LOW4)与输入数据（Addr_In）进行逻辑与运算，将结果输出至指定地址(Addr_Out)。可用来从指定地址中一个8 位的信号中排除不需要的位数。

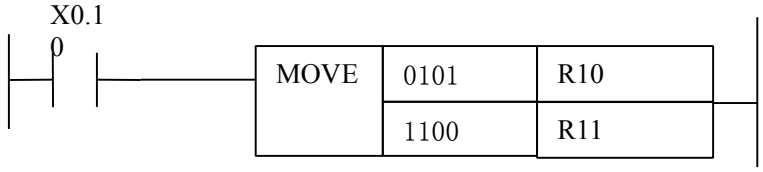
ACT=0 时：Addr_Out 保持原值。

相关参数



HIGH4 ：高四位逻辑乘数
LOW4 ：低四位逻辑乘数
Addr_In ：输入数据地址，地址号为 R、A、K、X、Y、F、G 、D 等。
Addr_Out：输出数据地址，地址号为 R、A、K、Y、G 、D、K 等

程序示例：

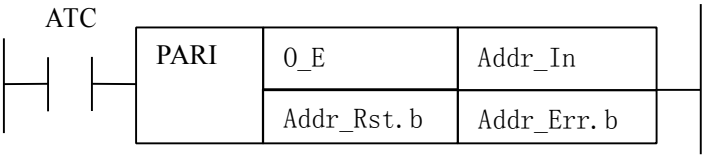


说明：当 X0.1=1 时，R10 和 01011100 进行与，将结果存放在 R11 中。

5.17 PARI（奇偶校验）

指令功能：
对输入数据进行奇偶校验，输入的数据为 1 个字节（8 位）。

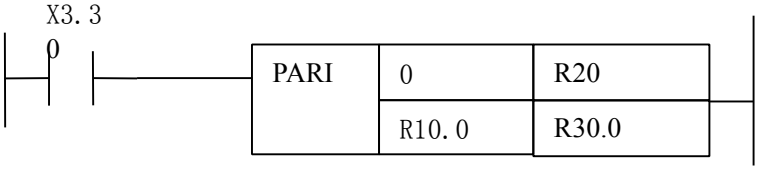
梯形图格式：



控制条件
当ACT=1 时：对输入数据进行奇偶校验，若输入数据与O_E 的指定不符，则Addr_Err.b 为1；否则Addr_Err.b 为0。
ACT=0：不执行指令，Addr_Err.b 保持原值。

相关参数
O_E =0：输入数据中的“1”的个数为偶数
=1：输入数据中的“1”的个数为奇数
Addr_Rst.b：为1 时，Addr_Err.b 复位为0，地址为X、Y、G、R、F、A 以及K 等。
Addr_In ：输入数据地址，地址可为X、Y、G、R、F、A、K 以及D。
Addr_Err.b ：校验结果输出地址，地址可为 Y、G、R、A、K 等。

程序示例：



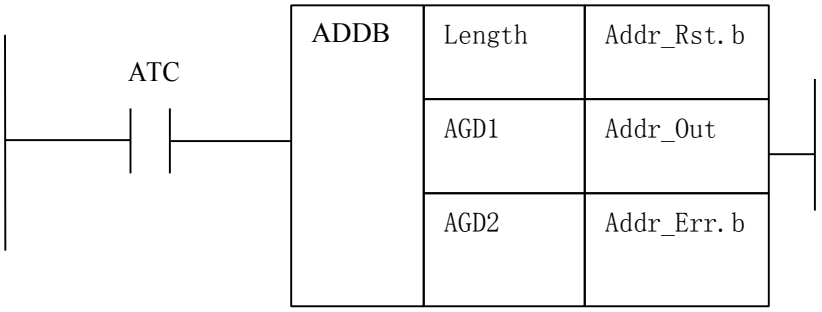
说明：当R3.3 为1 时执行PARI 指令，O_E=0，进行偶校验。当R10.0 为1，R0030.0 复位为0，不进行校验。当R10.0 为0 时，进行校验，当R20 的数值中含偶数个1 时，R30.0 为0，当R20 的数值中含奇数个 1 时，R30.0 为 1。

5.18 ADDB（二进制数据相加）

指令功能

二进制数据相加。

梯形图格式：



控制条件

当ACT=1 时：执行Addr_Out=AGD1+AGD2。若运算出错Addr_Err.b 为1；否则Addr_Err.b 为0。

ACT=0 时：不执行指令，Addr_Out 和 Addr_Err.b 保持不变。

相关参数

Length ：1、2、4字节长。

AGD1 ：被加数，可为常数或地址。地址号为R、X、Y、F、G、A、K、D，DT,TC,C,T等。

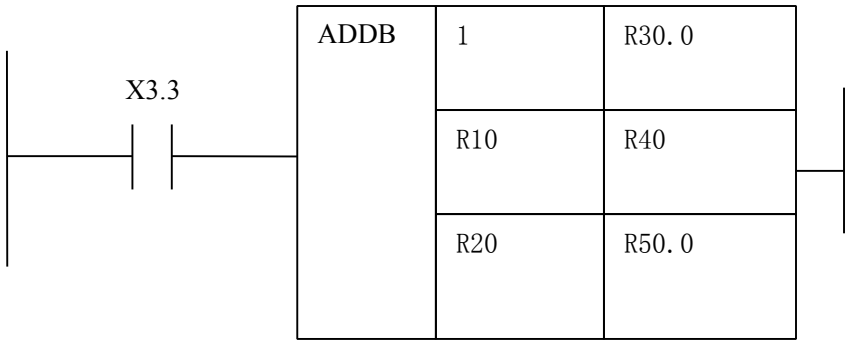
AGD2 ：加数，可为常数或地址。地址号为R、X、Y、F、G、A、K、D，DT,TC,C,T等。

Addr_Rst.b：为1 时，Addr_Err.b 复位为零，Addr_Out 不变。地址号为R、X、Y、F、G、A 以及K 等。

Addr_Out ：运行结果输出数据地址。地址可为Y、G、R、A、D、K、T、C等

Addr_Err.b：运算结果错误输出地址，地址可为 Y、G、R、A 、K。

程序示例：



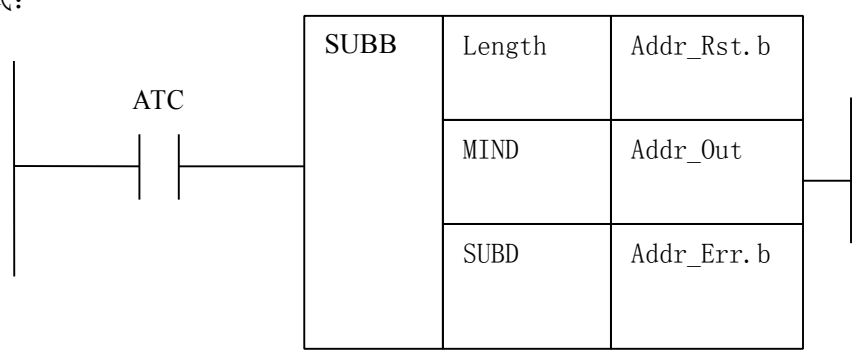
说明：当X3.3=1 时，执行ADDB 指令。R40=R10+R20，若运算出错，则R50.0 为1，否则为 0。当 R30.0 为 1 时，R40 状态不变，R50.0 复位为 0。

5.19 SUBB（二进制数据相减）

指令功能：

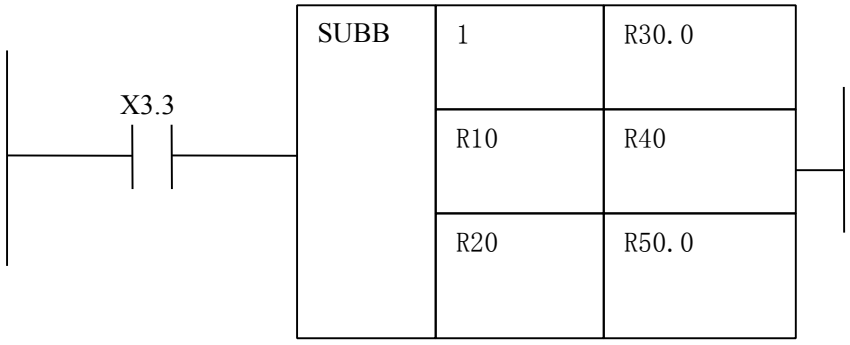
二进制数据相减。

梯形图格式：



控制条件
当ACT=1 时：执行Addr_Out= MIND - SUBD。若运算出错Addr_Err.b 为1；否则Addr_Err.b 为0。
ACT=0 时：不执行指令，Addr_Out 和 Addr_Err.b 保持不变。
相关参数
Length ：Length ：1、2、4字节长。
MIND：被减数，可为常数或地址。地址号为 R、X、Y、F、G、A、K、D，DT,TC,C,T 等。
SUBD：减数，可为常数或地址。地址号为R、X、Y、F、G、A、K、D，DT,TC,C,T等。
Addr_Rst.b：为1 时，Addr_Err.b 复位为零，Addr_Out 不变。地址号为R、X、Y、F、G、A 以及K 等。
Addr_Out ：运行结果输出数据地址。地址可为Y、G、R、A、D、K、T、C等
Addr_Err.b：运算结果错误输出地址，地址可为 Y、G、R、A 、K。

程序示例：

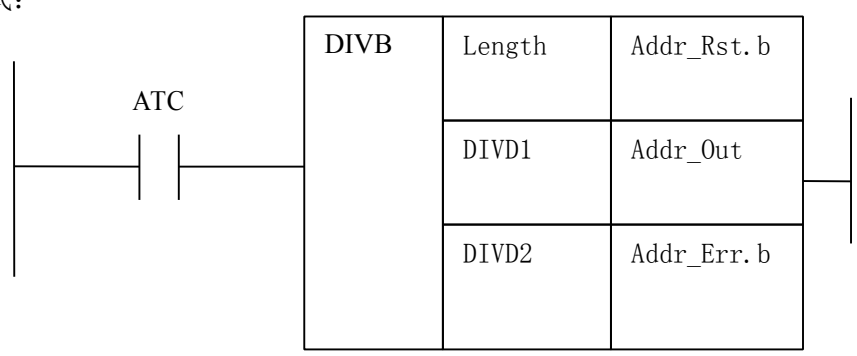


说明：当X3.3=1 时，执行SUBB 指令。R40=R10-R20，若运算出错，则R50.0 为1，否则为 0。当 R30.0 为 1 时，R40 状态不变，R50.0 复位为 0。

5.20 DIVB（二进制数据相除）

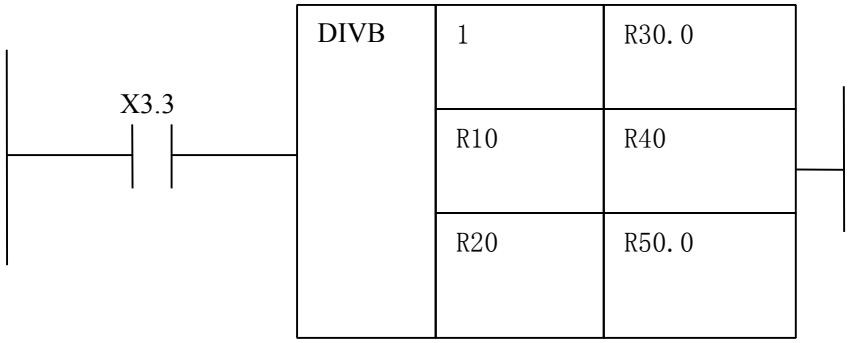
指令功能：
二进制数据相除。

梯形图格式：



控制条件
当ACT=1 时：执行Addr_Out= DIVD1 / DIVD2。若运算出错Addr_Err.b 为1；否则Addr_Err.b 为0。
ACT=0 时：不执行指令，Addr_Out 和 Addr_Err.b 保持不变。
相关参数
Length ：：1、2、4字节长。
DIVD1：被除数，可为常数或地址。地址号为 R、X、Y、F、G、A、K、D，DT,TC,C,T 等。
DIVD2：除数，可为常数或地址。地址号为R、X、Y、F、G、A、K、D，DT,TC,C,T等。
Addr_Rst.b：为1 时，Addr_Err.b 复位为零，Addr_Out 不变。地址号为R、X、Y、F、G、A 以及K 等。
Addr_Out ：运行结果输出数据地址。地址可为Y、G、R、A、D、K、T、C等
Addr_Err.b：运算结果错误输出地址，地址可为 Y、G、R、A 、K。

程序示例：



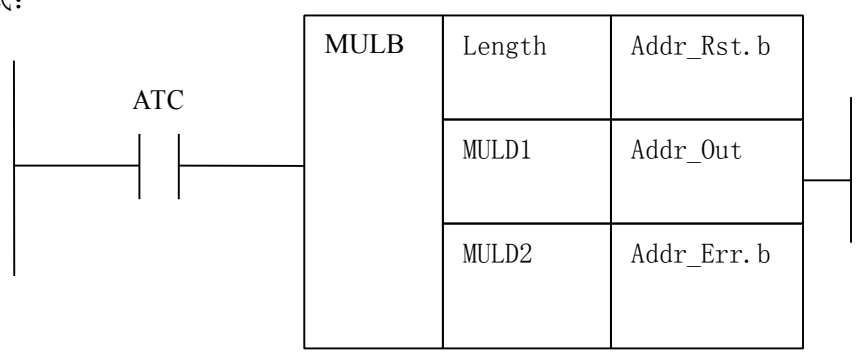
说明：当X3.3=1 时，执行DIVB 指令。R40=R10/R20，余数放在D255中。若运算出错，则R50.0 为1，否则为0。当R30.0 为1 时，R40 状态不变，R50.0 复位为0。

5.21 MULB（二进制数据相乘）

指令功能：

二进制数据相除。

梯形图格式：



控制条件

当ACT=1 时：执行Addr_Out= MULD1 * MULD2。若运算出错Addr_Err.b 为1；否则Addr_Err.b 为0。

ACT=0 时：不执行指令，Addr_Out 和 Addr_Err.b 保持不变。

相关参数

Length ：1、2、4字节长。

MULD1：被乘数，可为常数或地址。地址号为 R、X、Y、F、G、A、K、D，DT,TC,C,T 等。

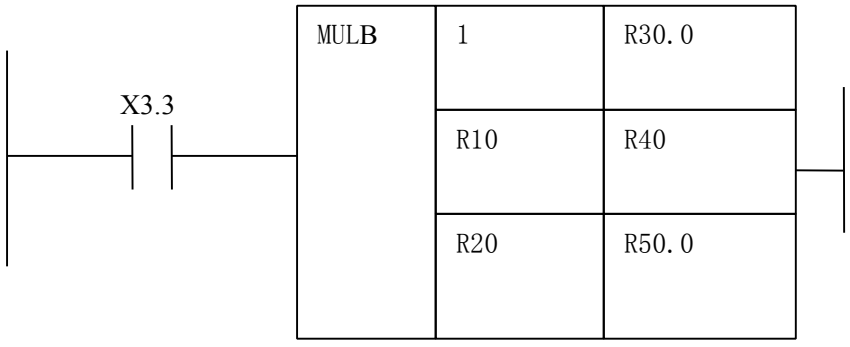
MULD2：乘数，可为常数或地址。地址号为R、X、Y、F、G、A、K、D，DT,TC,C,T等。

Addr_Rst.b：为1 时，Addr_Err.b 复位为零，Addr_Out 不变。地址号为R、X、Y、F、G、A 以及K 等。

Addr_Out ：运行结果输出数据地址。地址可为Y、G、R、A、D、K、T、C等

Addr_Err.b：运算结果错误输出地址，地址可为 Y、G、R、A 、K。

程序示例：

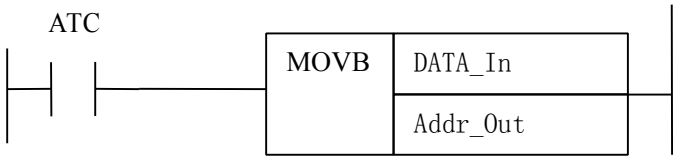


说明：当 X3.3=1 时，执行 MULB 指令。R40=R10*R20。若运算出错，则 R50.0 为 1，否则为 0。当 R30.0 为 1 时，R40 状态不变，R50.0 复位为 0。

5.22 MOVB（单字节数据传递）

指令功能：
单字节数据传递。

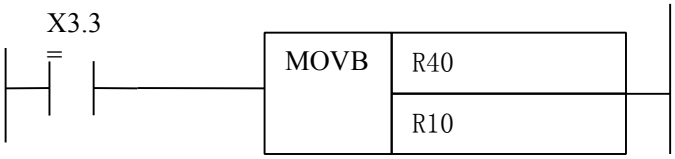
梯形图格式：



控制条件
当 ACT=1 时：执行 Addr_Out= DATA_In

ACT=0 时：不执行指令，Addr_Out 保持不变。
相关参数
DATA_In：输入数据。可以是常数或地址，地址号为 R、X、Y、F、G、A、K、D 等。
Addr_Out：输出数据地址，可为常数或地址。地址号为R、Y、G、A、D、K 等。

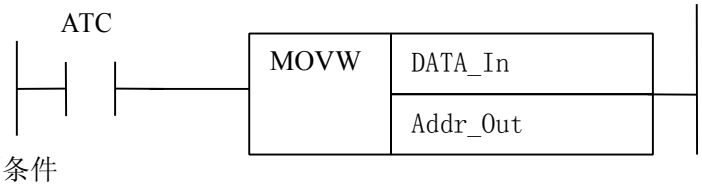
程序示例：



说明：当 X3.3=1 时，执行 MOVB 指令。R40=R10。

5.23 MOVW（双字节数据传递）

指令功能：
双字节数据传递。
梯形图格式：

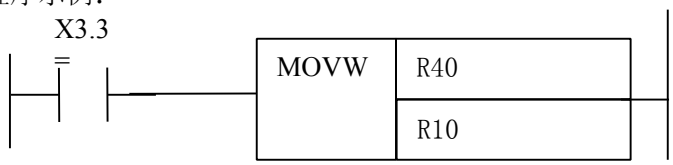


控制条件
当 ACT=1 时：执行双字节传递

ACT=0 时：不执行指令，Addr_Out 保持不变。
相关参数

DATA_In：输入数据。可以是常数或地址，地址号为 R、X、Y、F、G、A、K、D 等。
Addr_Out：输出数据地址，可为常数或地址。地址号为R、Y、G、A、D、K 等。

程序示例：

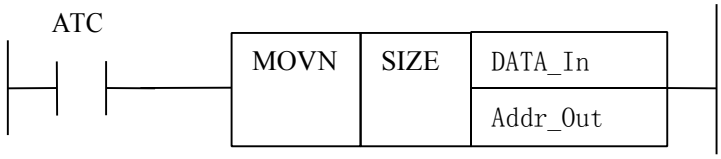


说明：当 X3.3=1 时，执行 MOVW 指令。R40=R10，R41=R11。

5.24 MOVN（1、2、4 字节数据传递）

指令功能：
1、2、4 字节数据传递。

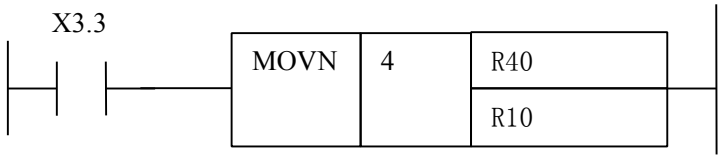
梯形图格式：



控制条件
当 ACT=1 时：执行双字节传递

ACT=0 时：不执行指令，Addr_Out 保持不变。
相关参数
SIZE　：要传递数据长度，1、2、4字节
DATA_In：输入数据。可以是常数或地址，地址号为 R、X、Y、F、G、A、K、D，DT,TC,C,T 等。
Addr_Out：输出数据地址，可为常数或地址。地址号为R、Y、G、A、D、K ，C,T等。

程序示例：



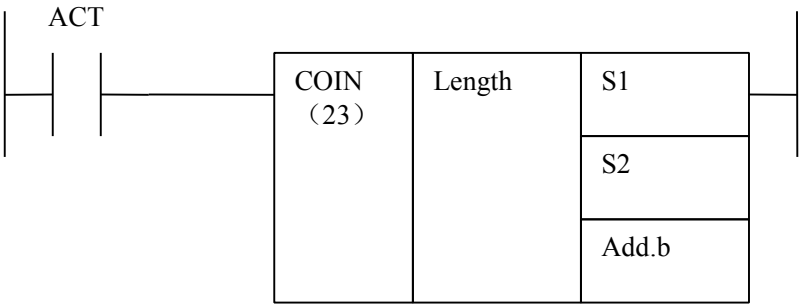
说明：当 X3.3=1 时，执行 MOVN 指令。R40=R10，R41=R11，R42=R12，R43=R13。

5.25 COIN（比较相等）

功能

比较两个二进制数据的是否相等，输出比较结果。

图形格式



指令格式

COIN	Length	S1	S2	Address.b

控制条件

ACT＝0，address.b 保持原值

ACT＝1，比较 S1，S2 的是否相等，如果 S1=S2，address.b=1；否则为 0.

参数

length， 指定数据长度

1：1 字节

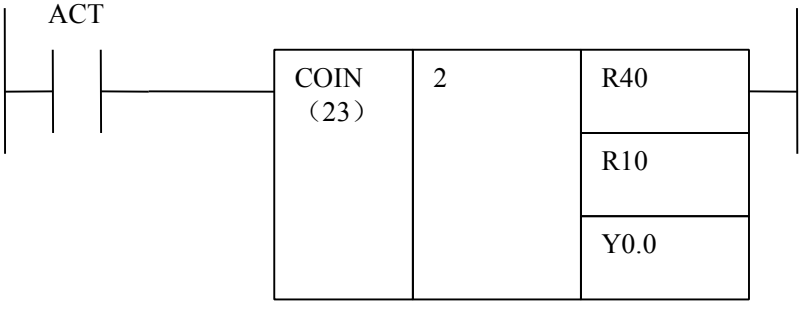
2：2 字节

4：4 字节

S1、S2， 比较源 1 和比较源 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D，DT, TC, C, T 类。

address.b ， 为比较输出结果。可为 R，Y，G， A 、D，K,T,C 类。

程序示例：



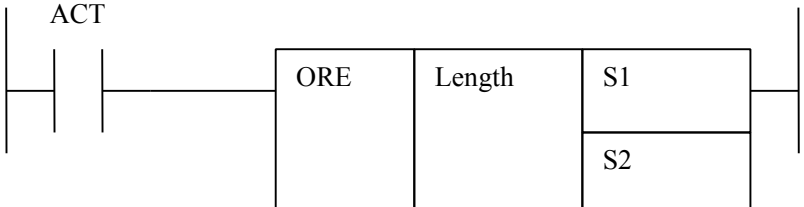
说明：当 X3.3=1 时，执行 COIN 指令。如果 R40 等于 R10 且 R41 等于 R11，Y0.0=1；否则 Y0.0=0。

5.26 ORE（异或）

功能

S1 与 S2 进行位异或，结果送给 Out。

图形格式



指令格式

ORE	Length	S1	S2	Out

控制条件

ACT＝0，Out 保持原值

ACT＝1， S1，S2 进行位异或，结果送给 Out

S1	1	1	0	1	0	1	0	1
S1	0	1	0	1	1	0	1	0
Out	1	0	0	0	1	1	1	1

参数

length， 指定数据长度

1：1 字节

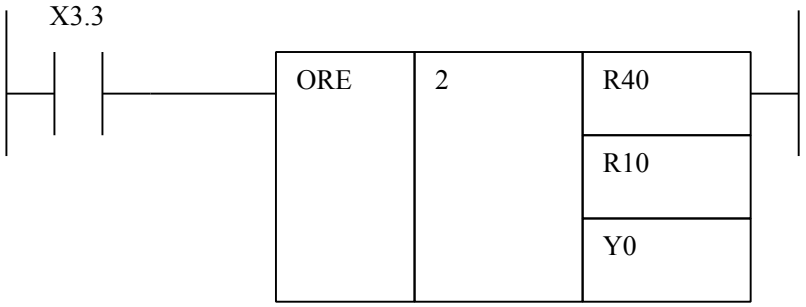
2：2 字节

4：4 字节

S1、S2， 输入数据 1 和输入数据 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D、T、C、DT、DC 类。

Out， 为比较输出结果。可为 R，Y，G， A 、D、K、T、C 类。

程序示例：



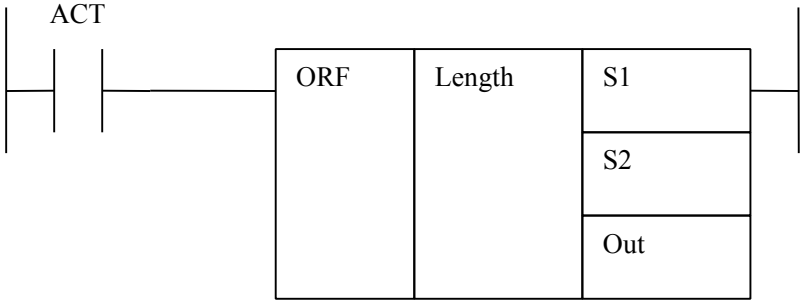
说明：当 X3.3=1 时，执行 ORE 指令。R40 和 R10 异或结果送给 Y0，R41 和 R11 异或结果送给 Y1。

5.27 ORF（或）

功能

S1 与 S2 进行位或，结果送给 Out。

图形格式



指令格式

ORF	Length	S1	S2	Out

控制条件

ACT＝0，Out 保持原值

ACT＝1， S1，S2 进行位或，结果送给 Out

S1	1	0	0	1	0	1	0	1
S1	0	1	0	1	0	0	0	0
Out	1	1	0	1	0	1	0	1

参数

length， 指定数据长度

1：1 字节

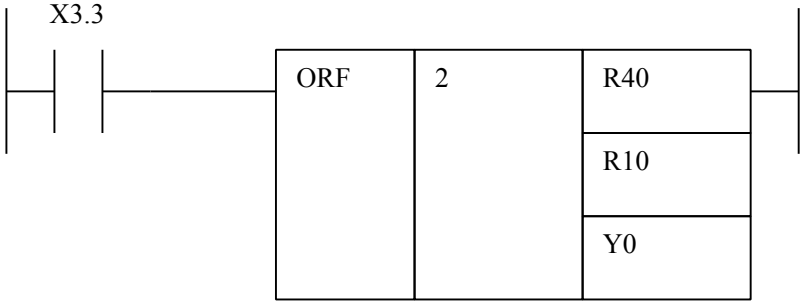
2：2 字节

4：4 字节

S1、S2， 输入数据 1 和输入数据 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D、T、C、DT、DC 类。

Out， 为比较输出结果。可为 R，Y，G， A 、D、K、T、C 类。

程序示例：



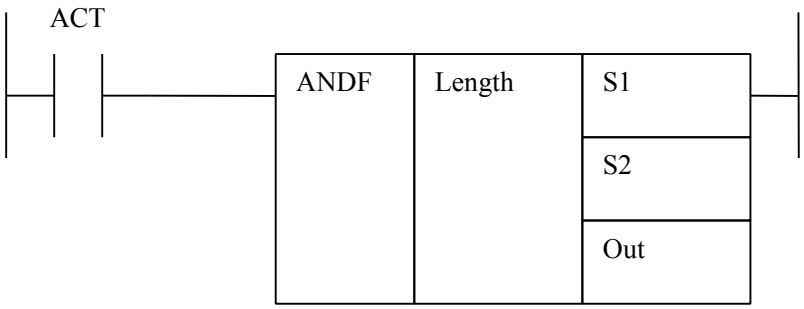
说明：当 X3.3=1 时，执行 ORF 指令。R40 和 R10 相或结果送给 Y0，R41 和 R11 相或结果送给 Y1

5.28 ANDF（与）

功能

S1 与 S2 进行位与，结果送给 Out。

图形格式



指令格式

ANDF	Length	S1	S2	Out

控制条件

ACT＝0，Out 保持原值

ACT＝1， S1，S2 进行位与，结果送给 Out

S1	1	0	1	1	0	1	0	1
S1	0	1	1	1	0	1	0	0
Out	0	0	1	1	0	1	0	0

参数

length， 指定数据长度

1：1 字节

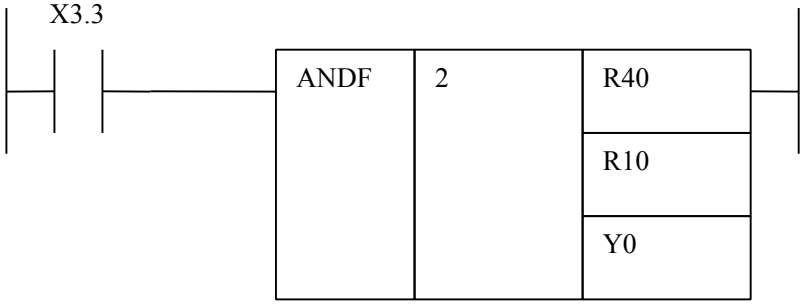
2：2 字节

4：4 字节

S1、S2， 输入数据 1 和输入数据 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D、T、C、DT、DC 类。

Out， 为比较输出结果。可为 R，Y，G， A 、D、K、T、C 类。

程序示例：



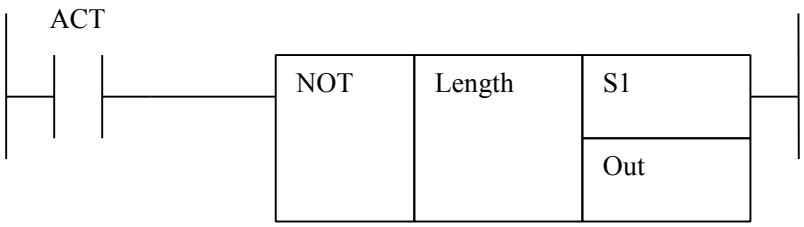
说明：当 X3.3=1 时，执行 ANDF 指令。R40 和 R10 相与结果送给 Y0，R41 和 R11 相与结果送给 Y1。

5.29 NOT（非）

功能

S1 进行位取反，结果送给 Out。

图形格式



指令格式

NOT	Length	S1	Out

控制条件

ACT＝0，Out 保持原值

ACT＝1， S1 取反结果送给 Out

S1	1	0	1	1	0	1	0	1
Out	0	1	0	0	1	0	1	0

参数

length， 指定数据长度

1：1 字节

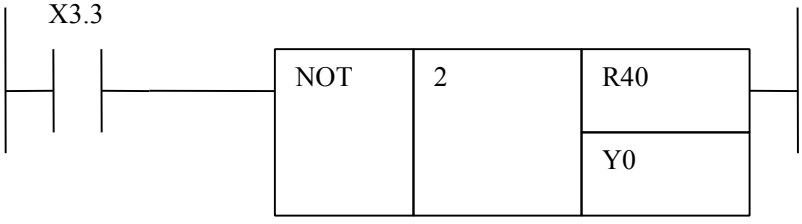
2：2 字节

4：4 字节

S1、S2， 输入数据 1 和输入数据 2 的内容。可为常数(constant)，也可为地址号(address， 注意，address.b 为非法)。地址号为，R，X，Y，F，G，K，A，D、T、C、DT、DC 类。

Out， 为比较输出结果。可为 R，Y，G， A 、D、T、C、K 类。

程序示例：



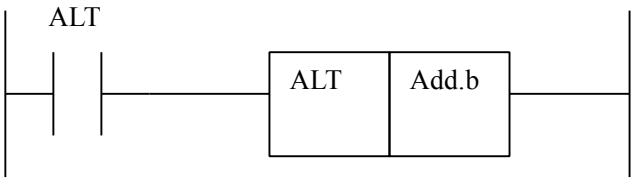
说明：当 X3.3=1 时，执行 NOT 指令。R40 取反结果送给 Y0，R41 取反结果送给 Y1。

5.30 ALT（交替输出）

功能

交替输出给 Add.b。

图形格式



指令格式

ALT	Add.b

控制条件

ACT＝0，add.b 的状态保持不变。

ACT＝1，add.b 取反。

参数

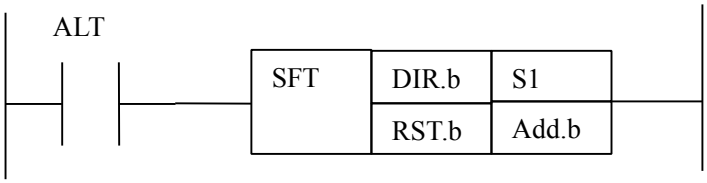
Add.b：置位元件地址位，可以为触点、输出线圈， Add= Y，G，R，A、K。

5.31 SFT（寄存器移位）

功能

该指令可使两字节长的数据左移或右移 1 位，数据‘1’在最左方（第 15 位）或在最右方（第 0 位）移出时，Add.b=1.

图形格式



指令格式

SFT	DIR.b	RST.b	S1	Add.b

控制条件

ACT＝0，add.b 的状态保持不变。

ACT＝1，如果 DIR.b 等于 1，S1 的第 0 位的值赋给 Add.b，然后 S1 右移一位。如果 DIR.b 等于 0，S1 的第 15 位的值赋给 Add.b，然后 S1 左移一位。

参数

DIR.b：指定移位方向地址，地址号为，R，X，Y，F，G，K，A 的位地址。

1：右移

0：左移

RST.b：复位地址 ，　地址号为，R，X，Y，F，G，K，A 的位地址

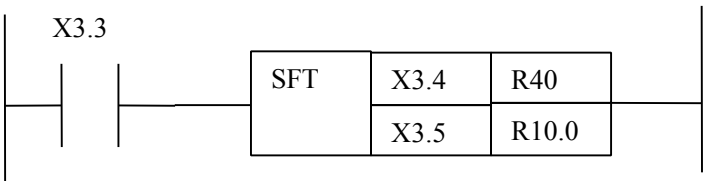
1：Add.b 等于 0；

0：Add.b 保持不变

S1 :移位数据，可以是地址也可以是常数，移位数据是地址时，指定的地址由连续两个地址组成，比如 S1 等于 R40，则移位数据为 R40 和 R41 组成的 16 位数据。R40 为低 8 位，R41 为高 8 位。地址号可为，R，X，Y，F，G，K，D，A。

Add.b：置位元件地址位，可以为触点、输出线圈， Y，G，R，A、K。

程序示例：



说明：当 X3.3=1 时，执行 SFT 指令，如果 X3.4=1 移位数据右移，R40 和 R41 组成的 16 位移位数据的第 0 位赋给 R10.0，然后进行右移一位，如果 X3.4=0 移位数据左移，R40 和 R41 组成的 16 位移位数据的第 15 位赋给 R10.0，然后进行左移一位。当 X3.5=1 时，R10.0=0，否则 R10.0 不变。

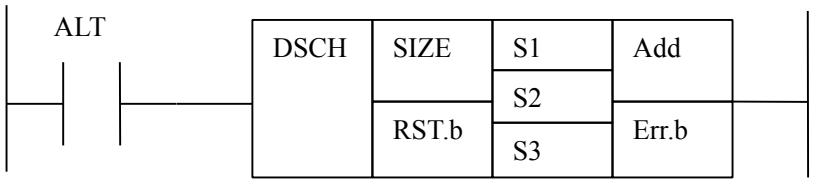
5.32 DSCH（数据检索）

功能

DSCH 指令仅用于 PMC 所使用的数据表。DSCH 指令在数据表中搜索指定的数据，并且输出其表内号。如果未找到指定数据，则 W1=1；

图形格式

指令格式



控制条件

SFT	SIZE	RST.b	S1	S2	S3	Add	Err.b

ACT＝0，Err.b、Add 的状态保持不变。

ACT＝1，执行 DSCH 指令

参数

SIZE：检索数据的字节长度，可以 1、2、4 字节

RST.b：复位地址 ，　地址号为，R，X，Y，F，G，K，A 的位地址

1：Err.b 等于 0；

0：Err.b 保持不变

S1 :输入检索数据地址。地址号可为，R，X，Y，F，G，K，D，A、T、C、DT、DC。

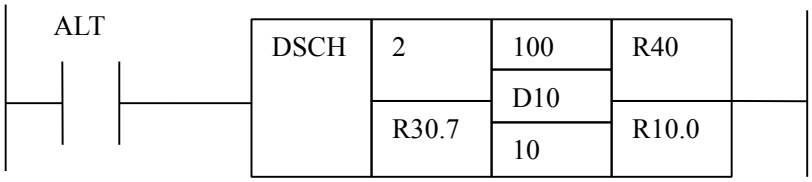
S2：检索数据表的起始地址　地址号：D 数据。

S3：检索数据表长度，地址或常数，地址号为 R，X，Y，F，G，K，D，A、T、C、DT、DC。

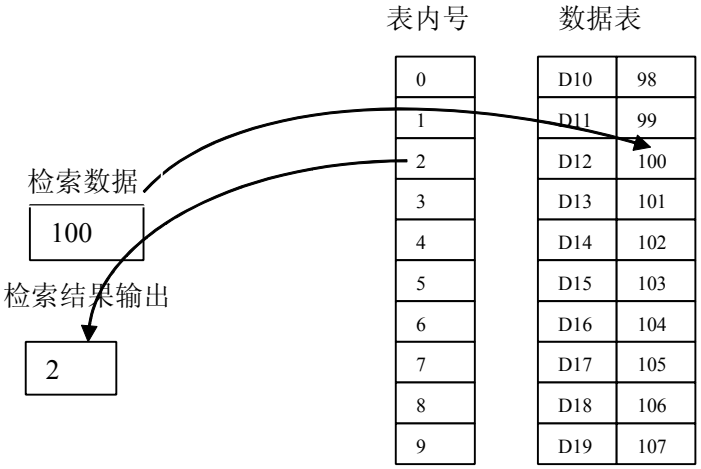
Add:检索结果输出地址

Err.b:输出检索错误地址，未检索到指定数据输出 1，否则输出 0。

程序示例：



说明：



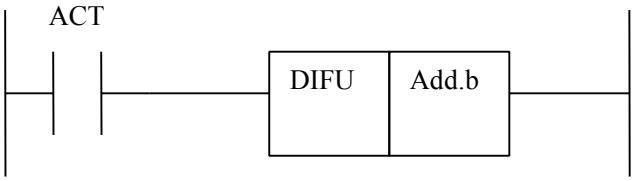
当 ACT=1；执行 DSCH 指令，如果 D12=100，则 R40=2；R10.0=0；

5.33 DIFU（上升沿检测）

功能：

DIFU 指令在输入信号上升沿的扫描周期中将输出信号设置为 1.

图形格式:



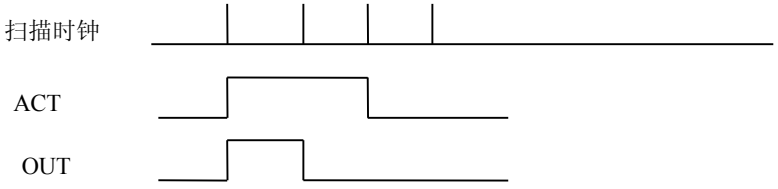
指令格式

DIFU	Add.b

控制条件

在输入信号上升沿（0→1），将输出信号设置为 1，保持一个扫描周期之后清零。

程序示例：

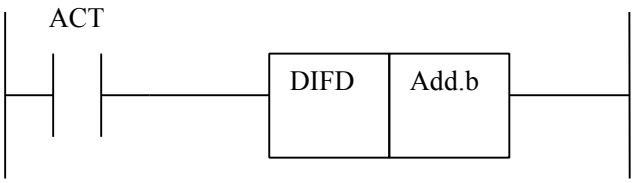


5.34 DIFD（上升沿检测）

功能：

DIFU 指令在输入信号下降沿的扫描周期中将输出信号设置为 1.

图形格式:



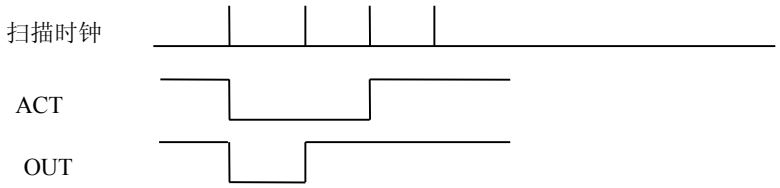
指令格式

DIFD	Add.b

控制条件

在输入信号上升沿（1→0），将输出信号设置为 1，保持一个扫描周期之后清零。

程序示例：

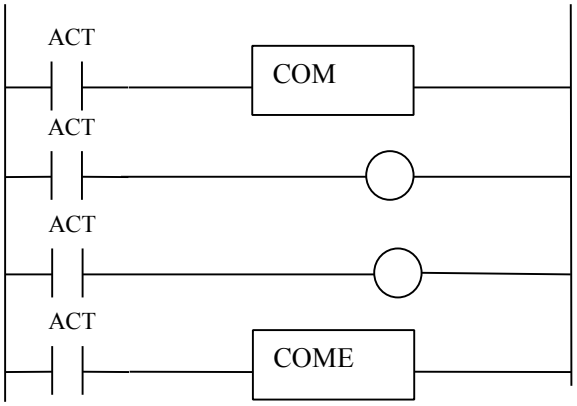


5.35 COM（公共线控制）

功能：

COM 指令控制直至公共结束指令（COME）范围内的线圈工作。COM 和 COME 必须成对出现。

图形格式:



指令格式

COM

控制条件:

ACT=0：指定范围内的线圈无条件断开（设为 0），功能指令不受影响。

ACT=1：与 COM 未执行时的操作一样。

注意：

1：用 COM 指令指定的范围内的功能指令正常执行而不管 COM 指令的 ACT 是何设定

2；在一条 COM 指令指定的范围内不允许指定另外的 COM 指令，即不能嵌套。

3：当 ACT=0 时，指定范围内 WRTNOT 的线圈无条件设为 1。

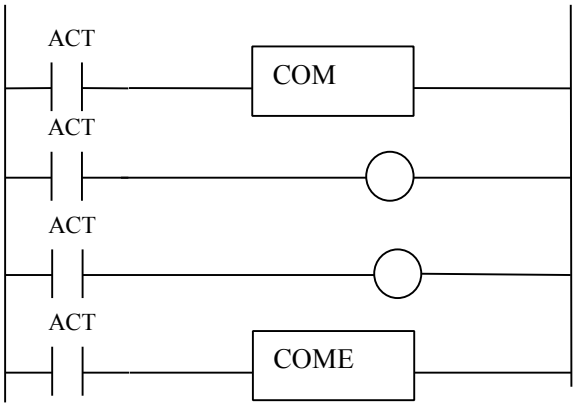
4：在 COM 指令范围内尽量不要使用 JMPB 指令跳出 COM 指令范围。否则会出现逻辑混乱。

5.36 COME（公共线控制结束）

功能：

COME 指令指定公共控制指令（COM）的控制范围。不能单独使用，必须与 COM 成对使用。

图形格式:



指令格式

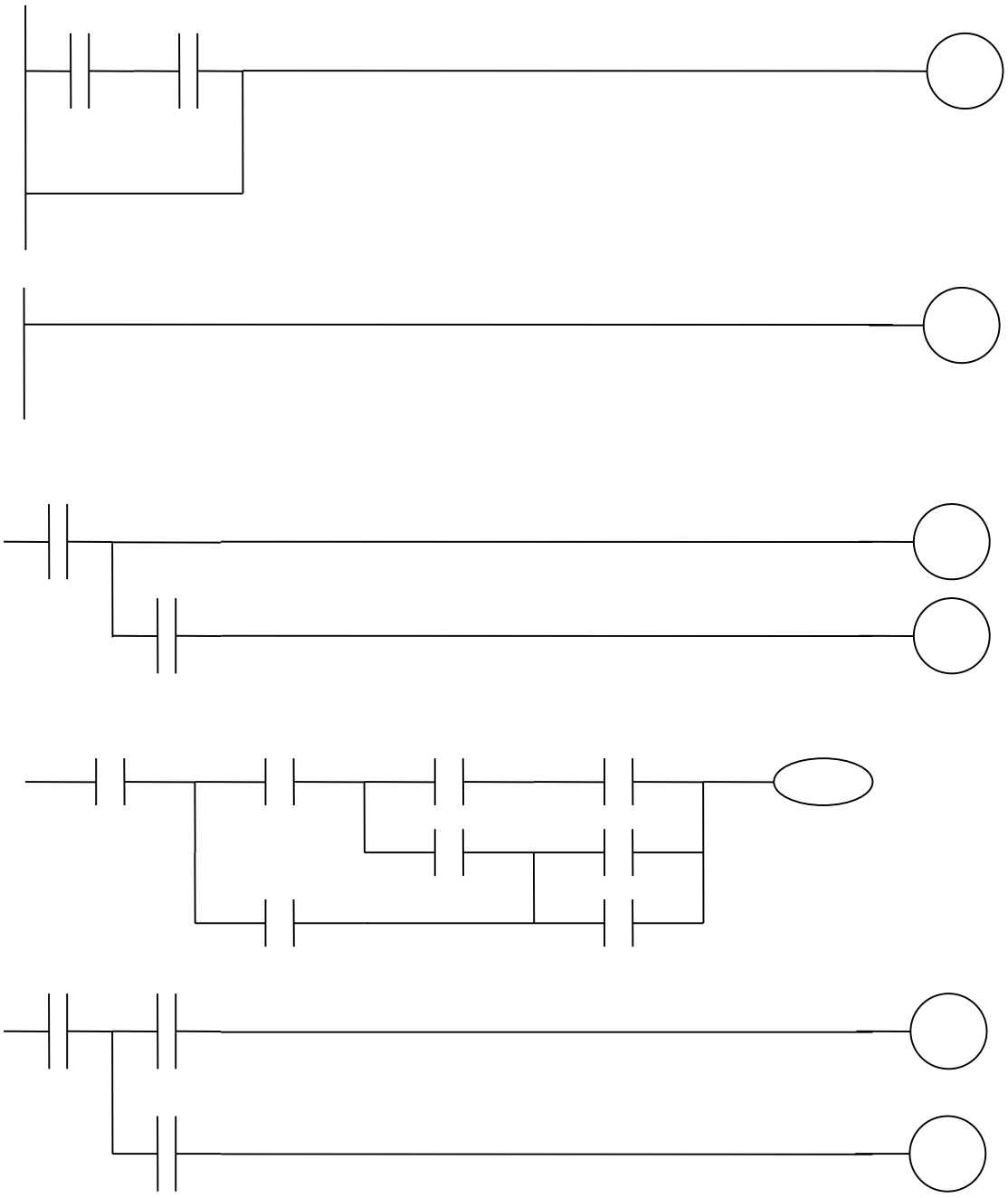
COME

第六章 梯形图编辑限制

1）程序必须有 END1 和 END2 指令，作为第一级和第二级程序的结束标志，且 END1 必须在 END2 之前；

2）只支持并列输出，不支持多级输出；

以下几种情况被视为语法错误:



附 录

附录一 X 地址

一、MDI 面板地址

二、伺服控制单元地址

三、手脉地址

X22.0	手脉轴选(X 轴)
X22.1	手脉轴选(Y 轴)
X22.2	手脉轴选(Z 轴)
X22.3	手脉轴选(4th 轴)
X22.4	手脉倍率选(×1)
X22.5	手脉倍率选(×10)
X22.6	手脉倍率选(×100)
X22.7	手脉倍率选(×1000)

附录二 Y 地址

一、MDI 面板地址

二、伺服控制单元地址

附录三 F 地址

附录四 G 地址