

【指令篇】

第 1 章：PLC 梯形图程序基本原理及简码指令的转译法则

本章将介绍 PLC 梯形图程序的基本原理，以及将梯形图程序转换成简码指令（MNEMONIC）的转译法则。

1.1 梯形图工作原理

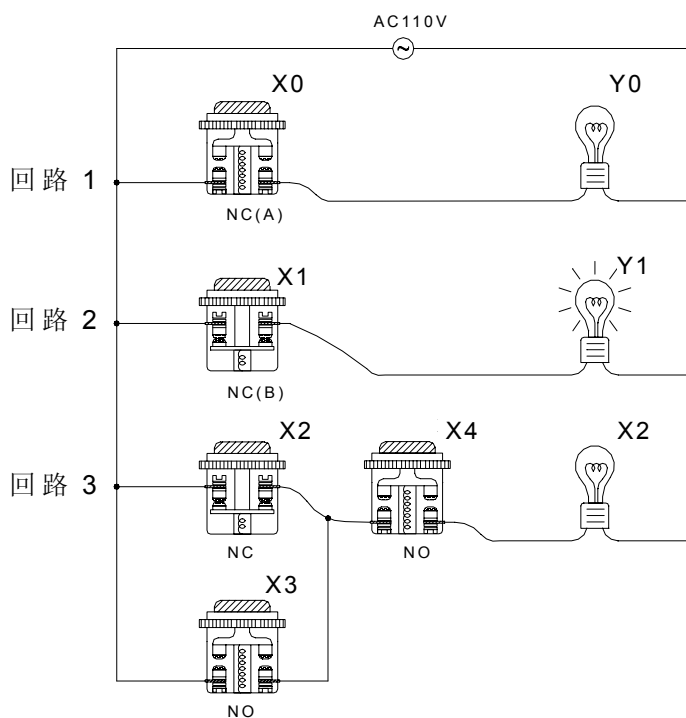
梯形图为二次世界大战期间所发展出来的自动控制图形语言，是历史最久、使用最广泛的自动控制语言，最初只有 A（常开）接点、B（常闭）接点、输出线圈、定时器、计数器等基本机构组件（今日仍在使用的配电盘即是），直到微电脑 PLC 出现后，梯形图的组件（语言）除上述组件外还增加了诸如微分接点、保持线圈等组件（请参考 1-6 页的组件类别）以及传统配电盘无法实现的应用指令。

无论是传统梯形图还是 PLC 梯形图其工作原理都相同，只是在符号表示上传统梯形图比较接近实体的符号表示，而 PLC 则采用较简明且易于计算机或报表上表示的符号表示。在梯形图逻辑方面可分为组合逻辑和顺序逻辑两种，现分述如下：

1.1.1 组合逻辑

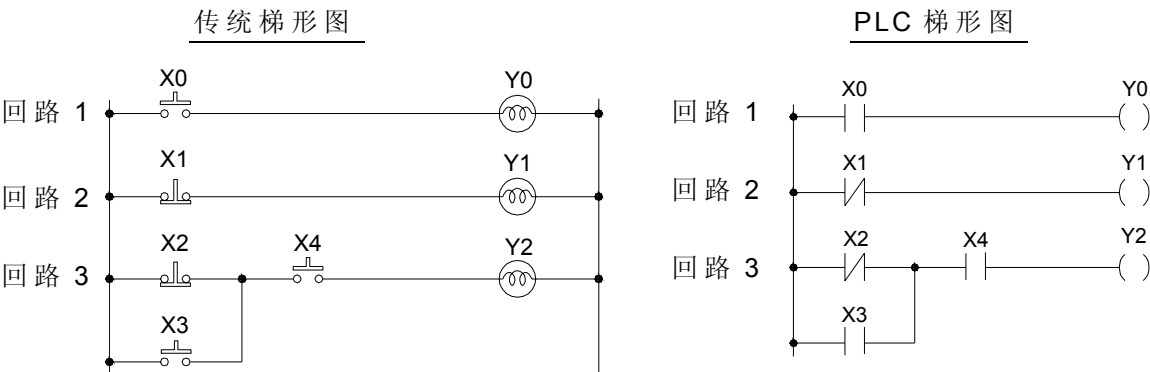
组合逻辑的梯形图是单纯地将单一或一个以上的输入组件组合（串、并联等）后再将结果送到输出组件（线圈、计时 / 计数器或应用指令等）的回路结构。

实际配线图



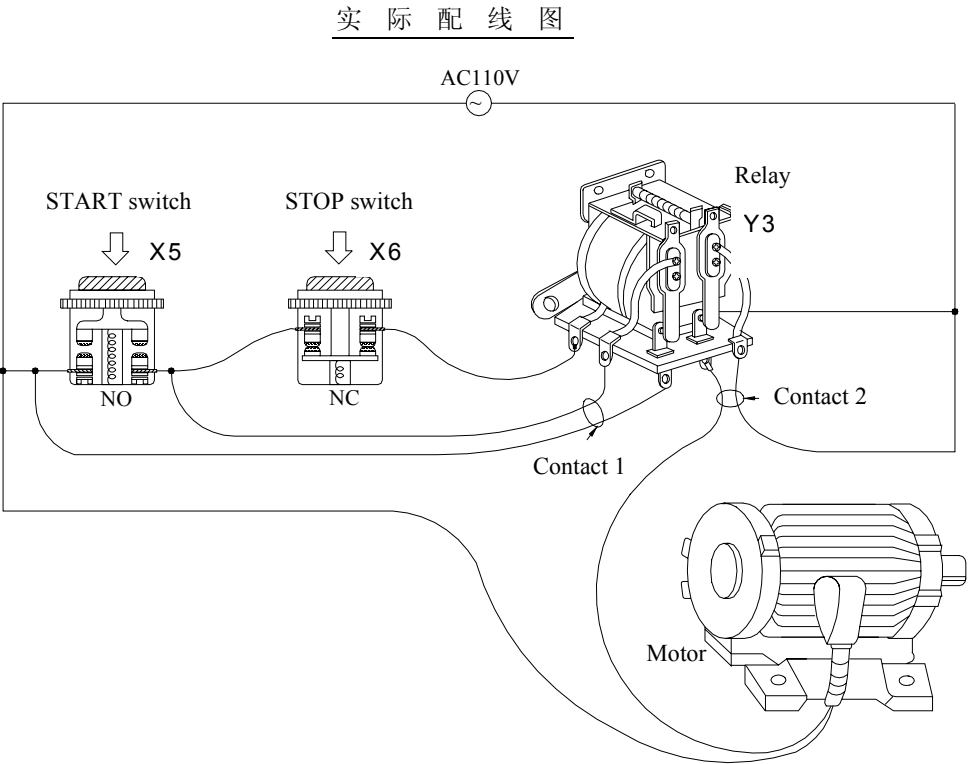
本例为组合逻辑分别以实际配线、传统梯形图及 PLC 梯形图表示的范例，其中回路 1 使用一个常开开关（NO: Normally Open）也就是一般所谓的“A”开关或接点。其特性是在平常（未压下）时其接点为开路（OFF）状态，故灯泡不亮，而在开关动作（压下按钮）时其接点变为导通（ON），故灯泡点亮。相对地，回路 2 使用一个常闭开关（NC: Normally Close）也就是一般所称的“B”开关或接点，其特性是在平常时其接点为导通，故灯泡点亮，而在开关动作时其接点反而变成开路，故灯泡熄灭。

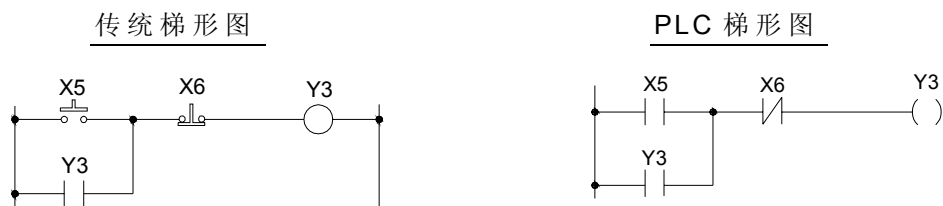
回路 3 为一个以上输入组件的组合逻辑输出范例，其输出 Y2 灯泡只有在 X2 不动作或 X3 动作且 X4 为动作时才会点亮。



1.1.2 顺序逻辑

顺序逻辑为具有回授结构的回路，也就是将回路输出结果拉回当输入条件，这样在相同输入条件下，会因前次状态或动作顺序的不同，而得到不同的输出结果，现就下图具自保持功能的电机启动 / 停止回路作说明。





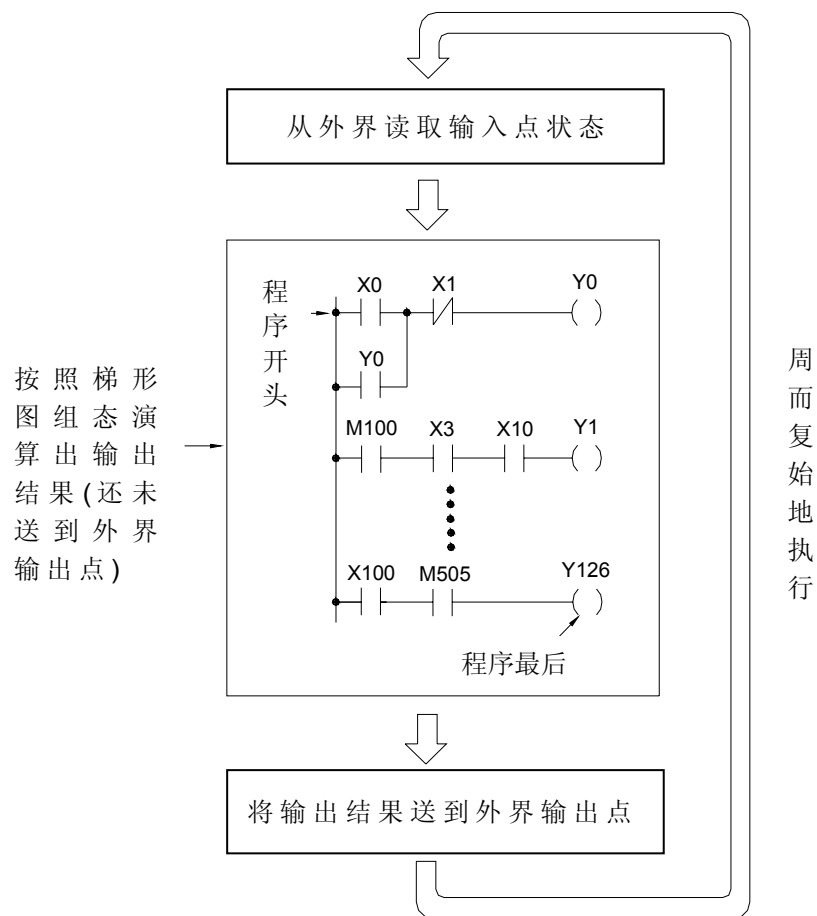
在此回路刚接上电源时，虽 **X6** 开关为 **ON**，但 **X5** 开关为 **OFF**，故继电器不动作，而继电器的输出接点 1 和接点 2 都为 **A** 接点（继电器动作时才 **ON**），故接点 1 和接点 2 都不导通，电机在停止状态。在启动开关 **X5** 按下后，继电器动作，接点 1 及接点 2 同时 **ON**，电机开始运转，一旦继电器动作后，即使放开启动开关（**X5** 变成 **OFF**）继电器电源因为自身的接点 1 回授而仍可继续保持动作（即为自保持回路），其动作可以下表表示：

	X5 开关 (NO)	X6 开关 (NC)	电机（继电器）状态
①	放开	放开	停止
↓			
②	压下	放开	动作
↓			
③	放开	放开	动作
↓			
④	放开	压下	停止
↓			
⑤	放开	放开	停止

由上表可知在不同顺序下，虽然输入状态完全一致，但输出结果也可能不一样，如表中的状态①和③其 **X5** 和 **X6** 开关都为放开，在①状态下电机为停止，但状态③时电机却为运转，这种继电器输出状态拉回当输入（即所谓的回授）而使回路具有顺序控制效果是梯形图回路的主要特性，因此有人称梯形图为“顺序控制回路”，而将 **PLC** 称为顺序控制器（**Sequencer**）。在本节范例中仅列举 **A**、**B** 接点和输出线圈作说明，其它组件的用法和此相同，请参考第 5 章“顺序指令说明”。

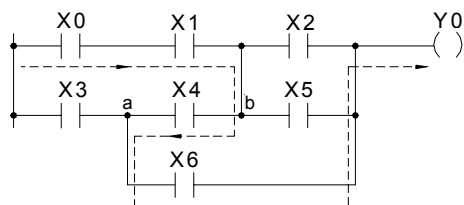
1.2 传统梯形图和 PLC 梯形图的差异

虽然传统梯形图和 **PLC** 梯形图的工作原理是完全一致的，但实际上 **PLC** 仅是利用微电脑（**CPU**）来模拟传统梯形图的动作，也就是利用扫描的方式逐一地查看所有输入组件及输出线圈的状态，再将这些状态按照梯形图的组态逻辑来演算出和传统梯形图一样的输出结果，但因 **CPU** 只有一个，只能逐一地查看梯形图程序，并按照该程序及输入/出状态演算输出结果，再将结果送到输出界面，然后又重新读取输入状态、演算、输出，如此周而复始地循环执行上述动作，这一完整的循环动作所费的时间称为扫描时间，其时间会随着程序的增大而加长，此扫描时间将造成 **PLC** 从输入检测到输出反应的延时，延时时间越长对控制所造成的误差越大，甚至造成无法胜任控制要求的情况，此时就必须选用扫描速度更快的 **PLC**，因此 **PLC** 的扫描速度是 **PLC** 的重要规格，只有靠计算机及 **ASIC**（特定用途 **IC**）技术的进步，现在的 **PLC** 在扫描速度上都有极大的改善，以 **FBs-PLC** 为例 **1K step** 接点的扫描时间只需 **0.33ms**，下图为 **PLC** 的梯形图程序扫描的示意图。



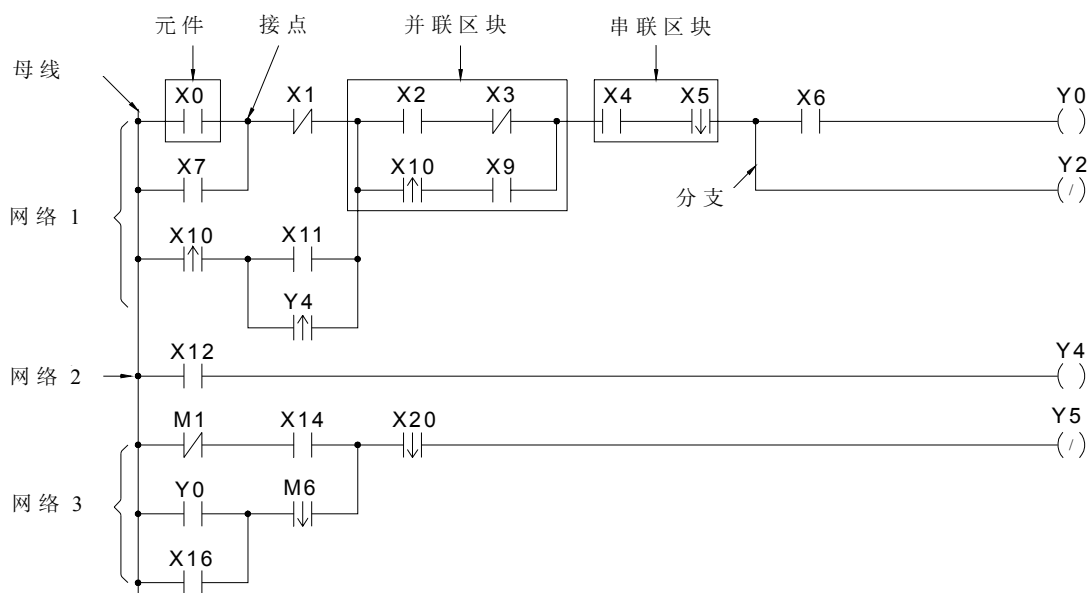
除上述扫描时间差异外，PLC 梯形图 and 传统梯形图还有如下“逆向回流”的差异，如下图所示图中若 X0，X1，X4，X6 为导通，其它为不导通，在传统的梯形图回路上输出 Y0 会如虚线所示形成回路而为 ON，但在 PLC 梯形图因 PLC 的 CPU 在演算梯形图程序的结果时，是由左而右，由上而下地扫描。在同样输入条件下，本图例中的 a 点状态因 X3 接点 OFF 故 CPU 认定为 OFF，虽然 a 点经由 X4 接至 b 点都为 ON，但因 PLC 梯形图只由左至右扫描，CPU 无法察觉，故 Y0 输出为 OFF。

传统梯形图的逆向回流



1.3 梯形图组成及其术语定义

图一：梯形图程序范例



(注：FBs 系列 PLC 的网络最大为 22 行 × 16 列)

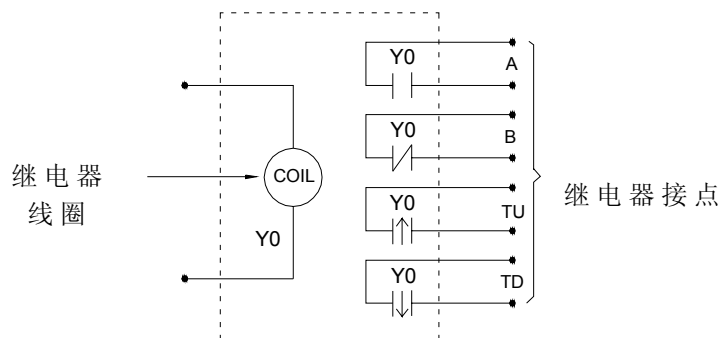
如上梯形图程序可分为一个个小方块（本图例为 8 列 × 11 行 = 88 个小方块），每个小方块都可以放置一个组件，将所有组件按照控制需求作成各种不同的连结即构成所谓的梯形图程序，现就梯形图程序相关的术语及其意义，分述如下：

① 接点（Contact）

接点为表示导通（ON）与不导通（OFF）状态的组件，共有两类。一为“输入接点”（编号以 X 开头的），其状态是来自外界（端子台上的输入点）。另一为“继电器附属的接点”（请参考②项说明），其状态是反应（来自）继电器线圈的状态。FBs 系列 PLC 所提供的接点有 A 接点、B 接点、上 / 下微分接点、开 / 短路接点 6 种，请参考④组件的说明。

② 继电器（Relay）

正如同传统继电器，它包含线圈（Coil）和接点（Contact），如下图例所示。

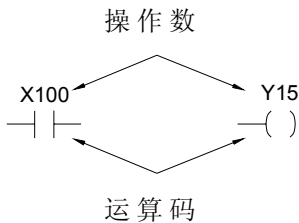


如图示继电器必有线圈，要使继电器动作，需驱动其线圈（用 OUT 指令驱动），在线圈被驱动后，其接点状态会受到影响。如上图例若将 Y0 以 1 驱动（使它为 ON），则继电器的 A 接点为 1，B 接点为 0，TU 接点只 ON 一个扫描时间，TD 接点为 0。当 Y0 变成 OFF 时，A 接点为 0，B 接点为 1，TU 接点为 0，而 TD 接点只 ON 一个扫描时间（A、B、TU、TD 接点的动作请参考第 5 章“顺序指令说明”）。

FBs-PLC 的继电器有四种，分别为 Y△△△（输出继电器），M△△△△（内部辅助继电器），S△△△（步进继电器）和 TR△△（暂存继电器），其中输出继电器 Y△△△的状态会被送到外部（端子台上的输出点）去。

③ 母线（Origin）：梯形图最左侧的起始线。

④ 组件（Element）：组件（即线圈或接点）为组成梯形图程序的最基本单位。组件的表示分为两部分，一为组件的符号，称之为运算码（OP Code），另一为数字部分，称之为操作数（Operand），如下图所示。



FBs 系列 PLC 的组件有下列 9 种：

组 件 类 别	符 号	简 码 指 令 表 示 方 式	备 注
A 接点 (常开接点)		(ORG、LD、AND、OR) □△△△△	□可为 X、Y、M、S、T、C (请参考 3.2 节说明)
B 接点 (常闭接点)		(ORG、LD、AND、OR)NOT □△△△△	
上微分接点		(ORG、LD、AND、OR)TU □△△△△	
下微分接点		(ORG、LD、AND、OR)TD □△△△△	
开路接点		(ORG、LD、AND、OR)OPEN	
短路接点		(ORG、LD、AND、OR)SHORT	
输出线圈		OUT □△△△△	□可为 Y、M、S
倒相输出线圈		OUT NOT □△△△△	
保持型外部输出线圈		OUT L Y△△△△	

注：X、Y、M、S、T、C 等接点或线圈范围请参考 3.2 节、其组件特性请参考 5.2 节。

另外尚有三个特殊顺序指令（OUT TRn、LD TRn 及 FOn）也属组件的一种，但却不显示在梯形图上，请参考第 1.6 节“暂存继电器（TR）的使用”及第 5.1.4 节“功能输出 FO”的说明。

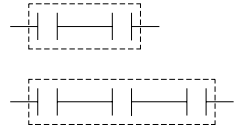
⑤节点（Node）：任意两个或两个以上组件相连接的点（FBs-PLC 可对节点状态作运作，请参考第 4.3 节“节点运作指令”的说明）。

⑥区块（Block）：两个或两个以上的组件组合成的回路称之为区块。

基本的区块有两种：

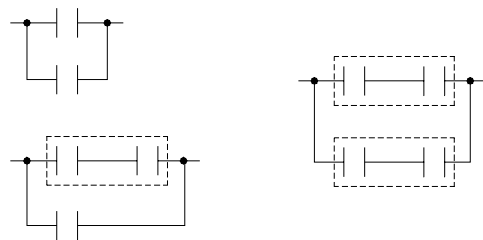
- 串联区块：两个或两个以上组件串接而成的单列回路。

例：



- 并联区块：由组件或串联区块并联组成的平行（矩形）封闭回路称之为并联区块。

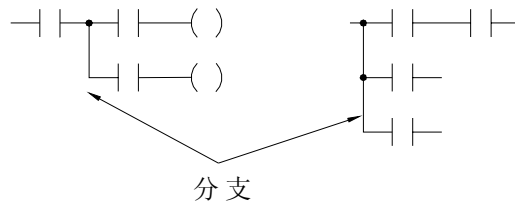
例：



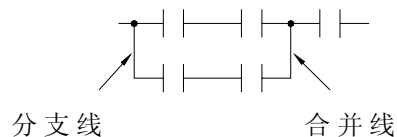
注：由组件、串联区块及并联区块等三种基本单元可以组成许多更复杂的串并联区块回路。在梯形图程序输入时，若以简码指令输入，必须先将所有网络拆分成上述的组件、串联区块、并联区块等基本单元后才能输入，请参考 1.5 节“梯形图网络的拆解”说明。

⑦分歧（Branch）：任一个网络中的垂直线右方有两列或两列以上的回路连接，这就是分歧，而该垂直线称为分歧线或称为支线。

例：

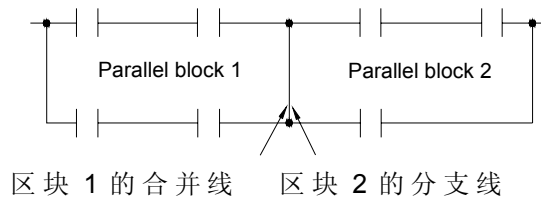


分歧线的右边若有另一垂直线将分歧的两列回路合并（该垂直线称为合并线），则此回路即形成一个封闭的回路（形成并联区块），此回路即非分歧回路。



若垂直线左、右边都有两列以上的回路连接，则该垂直线既是合并线，又是分歧线。

如下例：



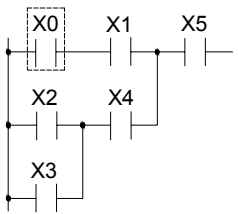
⑧网络（Network）：由组件、分歧、区块组成一个能执行特定功能的回路，称为网络。网络是梯形图程序中能执行完整功能的基本单位，而梯形图程序就是由一连串网络所组成。网络的起始必须由母线开始，任一个无垂直线连接的两列回路即属于不同的两个网络（有垂直线相连的则属于同一网络）。根据该法则，如图一可区分成网络 1~3 三个网络。

1.4 梯形图程序转成简码指令的转译法则 (梯形图大师用户请跳过本节)

FBs-PLC 如果以梯形图大师软件包当规划工作，则可由 CRT 屏幕直接以梯形图输入，使用简易、方便。但如果使用 FP-07C 当输入工具，则因 FP-07C 没有 CRT 屏幕来供绘图输入，用户必须按照本节至 1.6 节所述的法则以人工方式先将梯形图转译成等效的简码指令（MNEMONIC）后才能输入。以下为其转译法则：

- 程序编辑由左而右、由上而下，故网络的开头一定在回路的最左上角，网络开头指令必须用 ORG 指令，且一个网络只能有一个 ORG 指令（无输入控制的应用指令除外，请参考第 6.1.1 节的说明）。

例：

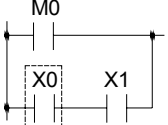




ORG	X	0
AND	X	1
LD	X	2
OR	X	3
AND	X	4
ORLD		
AND	X	5

- 接在垂直线（母线或支线）的指令用 LD 指令（网络的开头除外）。

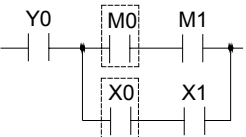
例 1：





ORG	M	0
LD	X	0
AND	X	1
ORLD		

例 2：

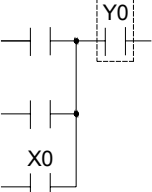




AND	Y	0
LD	M	0
AND	M	1
LD	X	0
AND	X	1
ORLD		

注 1：如果支线上仅串接一系列组件则直接用 AND 指令。

例：

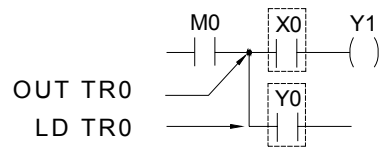




AND	X	0
ORLD		
AND	Y	0

注 2: 如果支线上已使用 **OUT TR** 指令将节点状态暂存起来(分歧回路用), 则也用 **AND** 指令。

例:



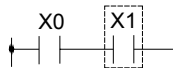
```

AND    M    0
OUT TR 0
AND    X    0
OUT    Y    1
LD     TR 0
AND    Y    0

```

- 单一组件串联用 **AND** 指令。

例:



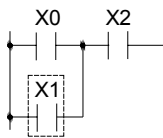
```

ORG    X    0
AND    X    1

```

- 单一组件并联用 **OR** 指令。

例 1:

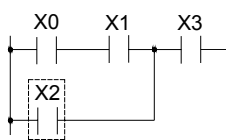


```

ORG    X    0
OR     X    1
AND    X    2

```

例 2:



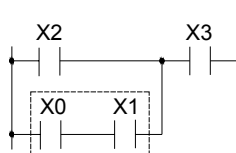
```

ORG    X    0
AND    X    1
OR     X    2
AND    X    3

```

- 并联组件为串联区块时必须使用 **ORLD** 指令。

例:



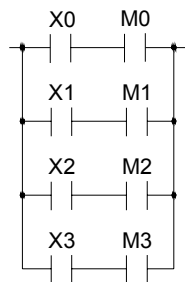
```

ORG    X    2
LD     X    0
AND    X    1
ORLD
AND    X    3

```

注: 若并联区块不只两列, 则应由上而下, 先并联第 1、第 2 列后再和第 3 列并联, 其余依此类推。

例:



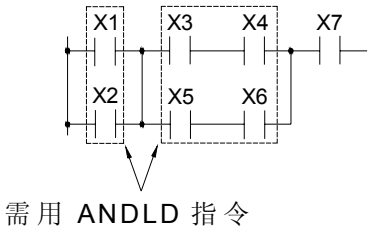
```

LD     X    0
AND    M    0
LD     X    1
AND    M    1
ORLD
LD     X    2
AND    M    2
ORLD
LD     X    3
ORLD M    3

```

- 并联块和并联块串联需用 **ANDLD** 指令。

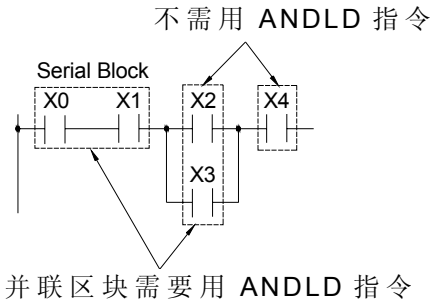
例：



ORG	X	1
OR	X	2
LD	X	3
AND	X	4
LD	X	5
AND	X	6
ORLD		
ANDLD		
AND	X	7

- 组件或串联块和并联块串联时，如果组件或串联块在前，并联块在后须用 **ANDLD** 指令。如果并联块在前，组件或串联块在后则直接用 **AND** 指令将并联块和组件或串联块 **AND** 起来即可。

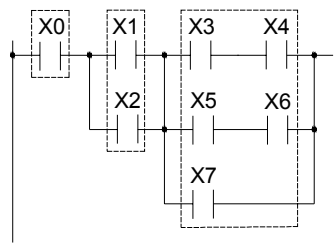
例：



ORG	X	0
AND	X	1
LD	X	2
OR	X	3
ANDLD		
AND	X	4

注：如果块的串联不只两个，则应由左到右先将第 1、第 2 个串联起来后，再和第 3 个块串联，其余依此类推。

例：



ORG	X	0
LD	X	1
OR	X	2
ANDLD		
LD	X	3
AND	X	4
LD	X	5
AND	X	6
ORLD		
OR	X	7
ANDLD		

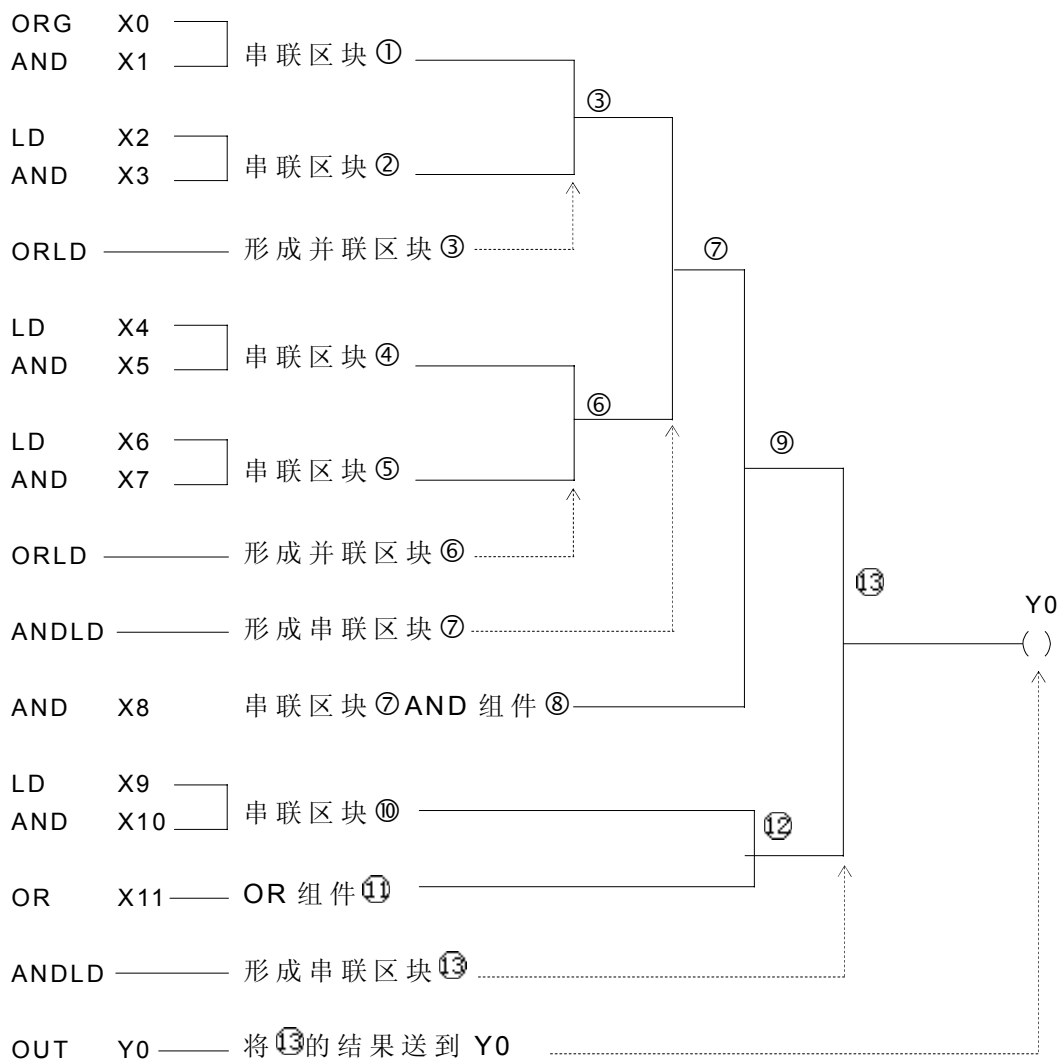
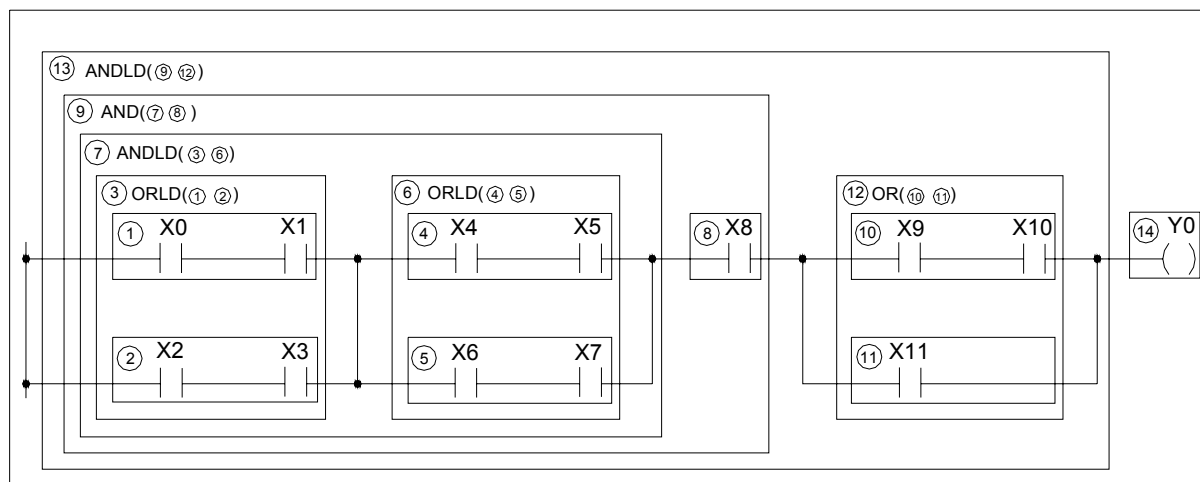
- 输出线圈指令（**OUT** 指令）只能放在网络的最后（最右边），即其后不能再接任何组件。输出线圈不能直接接在母线。若有此需求可用短路接点串接。如下例：



ORG	SHORT	
OUT	Y	0

1.5 梯形图网络的拆解 (阶梯图大师用户请跳过本节)

网络拆解要领是将介于任意两垂直线间的回路区分成独立的组件或串联区块，再按照上节所述的简码转译法则转译成简码指令，再由左而右、由上而下、由小而大将它们连结成并联区块或串并联区块（用 **ANDLD** 或 **ORLD** 指令），直到整个网络都连结完成，如下图范例：



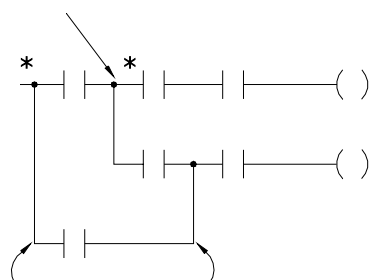
1.6 暂存继电器(TR)的使用 (梯形图大师用户请跳过本节)

对分歧回路或分歧区块而言，无法单纯地利用 1.5 节所述的方法来拆解输入，必须利用暂存接点先将分歧点的节点状态存起来，再利用 1.5 节的方法进行输入。因此回路设计应尽量避免形成分歧回路或区块（请参考下节“程序简化技巧”所述）。现就使用 TR 的两种回路叙述如下：

- 分歧回路：分歧线的右边无合并线的，或虽有合并线但和分歧线不同列的。

例： *表示需要设定 TR 点

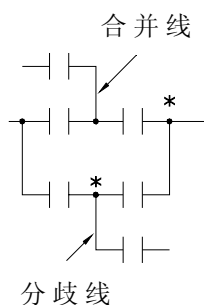
无合并线的



此分歧虽有合并线但不同列，也属于分歧回路

- 分歧区块：虽为平行（矩形）的并联区块，但区块的任意一列有分歧的。

例：



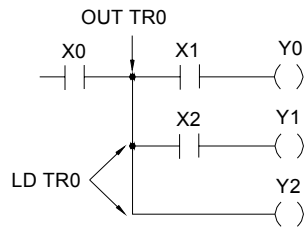
注 1：TR 点的设定必须在分歧回路或分歧区块的分歧线的第一列（最顶端）处，而第二列以后的回路开始前必须先用 LD TRn 指令取回该分歧线的状态后，才开始串接（AND）该列的第一个组件……。（在 OUT TRn 或 LD TRn 指令后的第一个组件必须用 AND 指令，不能用 LD 指令）。

注 2：一个网络最大可有 40 个 TR 点设定。TR 点的号码可任意选用，只要不重复即可（为易读起见最好由 0, 1, 2, ……顺序排起）。同一分歧线其 TR 号码必须一致（例如一分歧线用 OUT TR0，在该分歧线的第二列起必须用 LD TR0 来接续）。

注 3：分歧回路或分歧区块的分歧线若为母线，则无需使用 TR 接点，直接用 ORG 或 LD 指令即可。

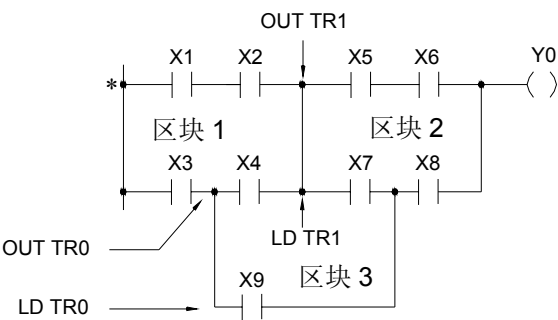
注 4：分歧回路若有任何一列非直接接输出线图（中间有串接组件），且其下方（第二列以后）尚有回路，则该分歧点必须使用 TR 接点。

例 1:



```
AND      X    0
OUT  TR  0
AND      X    1
OUT      Y    0
LD  TR  0  ←第二列开始
AND      X    2
OUT      Y    1
LD  TR  0  ←第三列开始
OUT      Y    2
```

例 2:

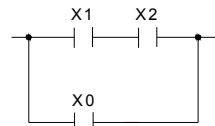
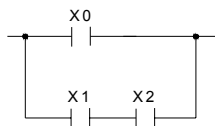


```
ORG      X    1
AND      X    2
LD      X    3
OUT  TR  0
AND      X    4
ORLD
OUT  TR  1
AND      X    5  ←TR 指令后用 AND
AND      X    6
LD  TR  1  ←回 TR 点用 LD TR
AND      X    7
LD  TR  0
AND      X    9  ←TR 指令后用 AND
ORLD
AND      X    8
ORLD
OUT      Y    0
```

- 上图例 2 的区块 1、2 原本为典型的两个并联区块串联。但 X9 组件介入后不但形成区块 3，还使区块 1、2 由原来单纯的并联区块变成分歧区块。
- (*) 处因为是母线，故不需要使用 TR 指令。
- 两区块串联如果已经使用 TR 点作转接，则无须使用 ANDLD 指令。

1.7 程序简化技巧

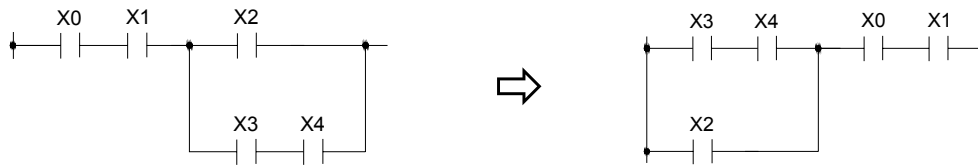
- 单一组件和串联区块并联，请将单一组件放在下方可省去 ORLD 指令。



```
LD      X    0
LD      X    1
AND      X    2
ORLD
```

```
LD      X    1
AND      X    2
OR      X    0
```

- 单一组件或串联区块和并联区块并联时，请将并联区块放在前方可省却 **ANDLD** 指令。



```

ORG   X   0
AND   X   1
LD     X   2
LD     X   3
AND   X   4
ORLD
ANDLD

```

```

ORG   X   3
AND   X   4
OR     X   2
AND   X   0
AND   X   1

```

- 分歧回路的分歧点如果直接接输出线圈，应将该输出线圈放在分歧线的最上面（第一列）。



```

OUT TR 0
AND   X   0
OUT   Y   0
LD TR 0
OUT   Y   1

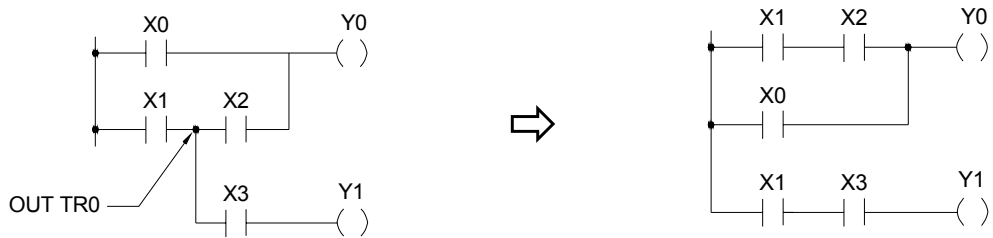
```

```

OUT   Y   1
AND   X   0
OUT   Y   0

```

- 下图例可省去 **TR** 接点及 **ORLD** 的使用。



```

ORG   X   0
LD     X   1
OUT TR 0
AND   X   2
ORLD
OUT   Y   0
LD TR 0
AND   X   3
OUT   Y   1

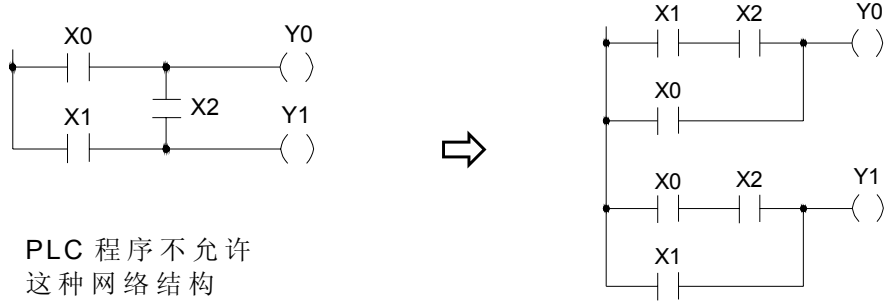
```

```

ORG   X   1
AND   X   2
OR     X   0
OUT   Y   0
ORG   X   1
AND   X   3
OUT   Y   1

```

- 桥式回路须作如下的转换。



```

ORG  X  1
AND  X  2
OR   X  0
OUT  Y  0
ORG  X  0
AND  X  2
OR   X  1
OUT  Y  1

```



MEMO

