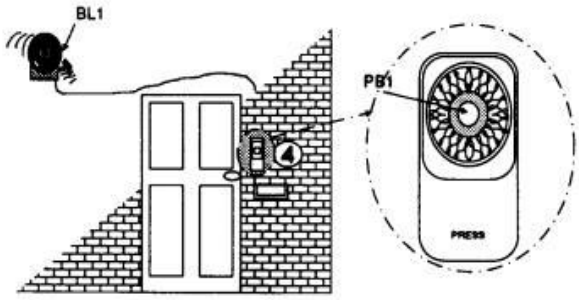
目录

1.1一个开环电路

梯形图

1.1 一个开环电路

用在门铃上的一个小开环电路。



器件	PC 软元件	说明
PB1	X000	门铃按钮
BL1	Y000	门铃

说明：

只有在门铃按钮 PB1 被按下时，门铃 BL1 才响。BL1 只能在 PB1 工作的同一时间段内工作，事实上，BL1 操作完全依赖于 PB1。

C写法

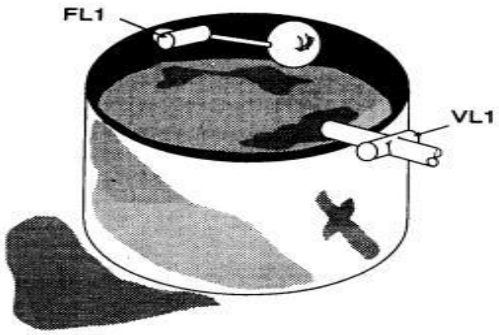
```
void PLC_Task(void)
{

        if(X0==1)SET(Y0); //如果X0接通，Y0就接通
        else      RST(Y0); //否则Y0就断开
 }
```

1.2信号非允许动作

梯形图

1.2 信号非允许动作

当浮阀发一个信号时，容器停止注水。



器件	PC 软元件	说明
FL1	X003	浮标传感器-测水位
VL1	Y003	进水阀

说明：

一个空容器的自然状态是：浮阀 FL1 "悬" 空，进水阀 VL1 打开。这样水就流入并注满容器。当容器逐渐地注满了水，浮阀的浮标抬起。

最后，浮阀到达一点，在该点它接通一个开关，由此传送一个信号给 PC 中的 X003，使常闭触点无效，线圈 Y003 掉电，从而关闭进水管的阀门。当水位降低，浮阀下降，供水阀重新打开。

C写法1

C写法

```
void PLC_Task(void)
{
        if(X3==0)SET(Y3);  //如果X3接通，Y3就接通
        else       RST(Y3);  //如果X3断开，Y3就断开
 }
```

C写法2

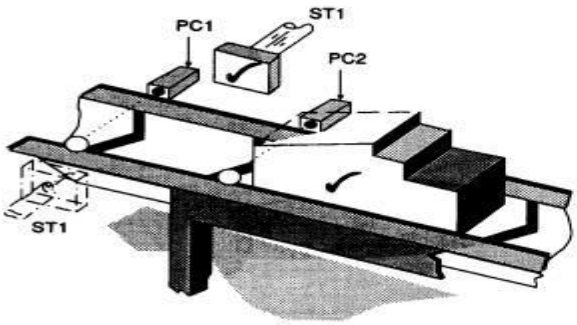
```
void PLC_Task(void)
{
        if(X3==1)RST(Y3);  //如果X3接通，Y3就断开
        else       SET(Y3); //如果X3没有接通，Y3就接通
 }
```

1.3引起一个动作的串联与输入

梯形图

1.3 引起一个动作的串联与输入

这个电路检测随传送带运动的产品的位置。



器件	PC 软元件	说明
PC1	X002	定位光电管 1
PC2	X003	定位光电管 2
ST1	Y002	贴邮票执行结构

说明：

当包裹从传送带上送过来时，经过两个光电管 PC1 和 PC2。这两个光电管用来检测传送线上包裹的位置。当信号 X002 和 X003 同时被接收到，即：两个光电管同时被激活时，包裹贴邮票的操作就能顺利完成，即 Y002 被驱动，把邮票 ST1 移向包裹

附注：贴邮票操作的移开和复位不在这里细述。

C写法

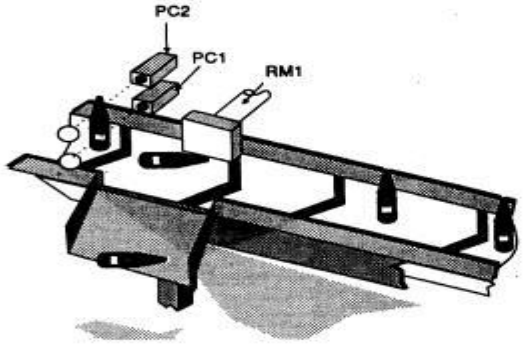
```
void PLC_Task(void)
{
    if(X2==1 && X3==1)SET(Y2);  //如果X2、3提示接通，Y0就接通
    else                RST(Y2);  //否则Y0就断开
 }
```

1.4引起一个动作的串联与非输入

梯形图

1.4 引起一个动作的串联与非输入

这个电路检测瓶子是否直立，如果不，则它被抛到传送带外。



器件	PC 软元件	说明
PC1	X004	光电管自动检测瓶底
PC2	X005	光电管自动检测瓶顶
RM1	Y001	推出杆

说明：

当瓶子从传送带上移过来时，它被两个光电管 PC1 和 PC2 检测。这两个光电管经安排能检测瓶子是否直立。从它们那儿得到两个输入 X004 和 X005。如果瓶子不处于直立状态，光电管 PC2 就不能给输入 X005 信号。这意味着推出杆 RM1 开始工作，因为程序驱动输出 Y001：输入 X004 为 ON，而又因为输入 X005 为 OFF，常闭触点使输出 Y001 被驱动。

C写法

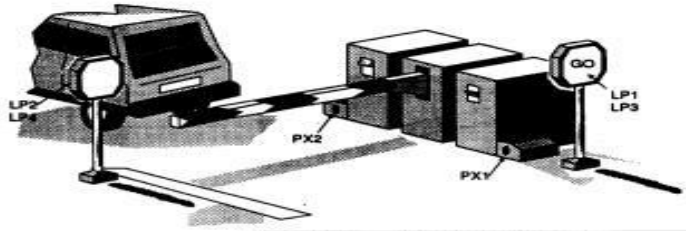
```
void PLC_Task(void)
{
    if(X4==1 && X5==0)SET(Y1);  //如果X4按下、X5断开，Y1就接通；
    else                      RST(Y1);  //否则Y1就断开
 }
```

1.5互锁电路

梯形图

1.5 互锁电路

在检票栏旁的交通控制信号。



器件	PC 软元件	说明
PX1	X001	车进入停车场
PX2	X002	车离开停车场
LP1	Y001	GO-正要进入的车辆
LP2		STOP-正要离开的车辆
LP3		STOP-正要进入的车辆
LP4	Y002	GO-正要离开的车辆

说明：

这个程序说明了一种互锁程序输出的简单方法，这是个确保安全性的快捷方法。在安全要求高的控制场合，也应该使用机械的和/或物理的互锁。程序操作如下：

一辆车接近检票栏前，触发一个接近开关，是 PX1 还是 PX2，这决定于车来的方向。当 PX1 或 PX2 被触发时，输入 X001 和 X002 信号被 PC 接收。每个输入触发一个输出，Y001 或 Y002。

接着被驱动的输出使交通指示灯接通，允许车通过，也就是说，一盏灯指示 GO，而另一盏灯也由同一个输出信号控制，指示 STOP。如果有两辆车同时要通过检票栏，互锁结构会保证在任何时刻只有一辆车通过。

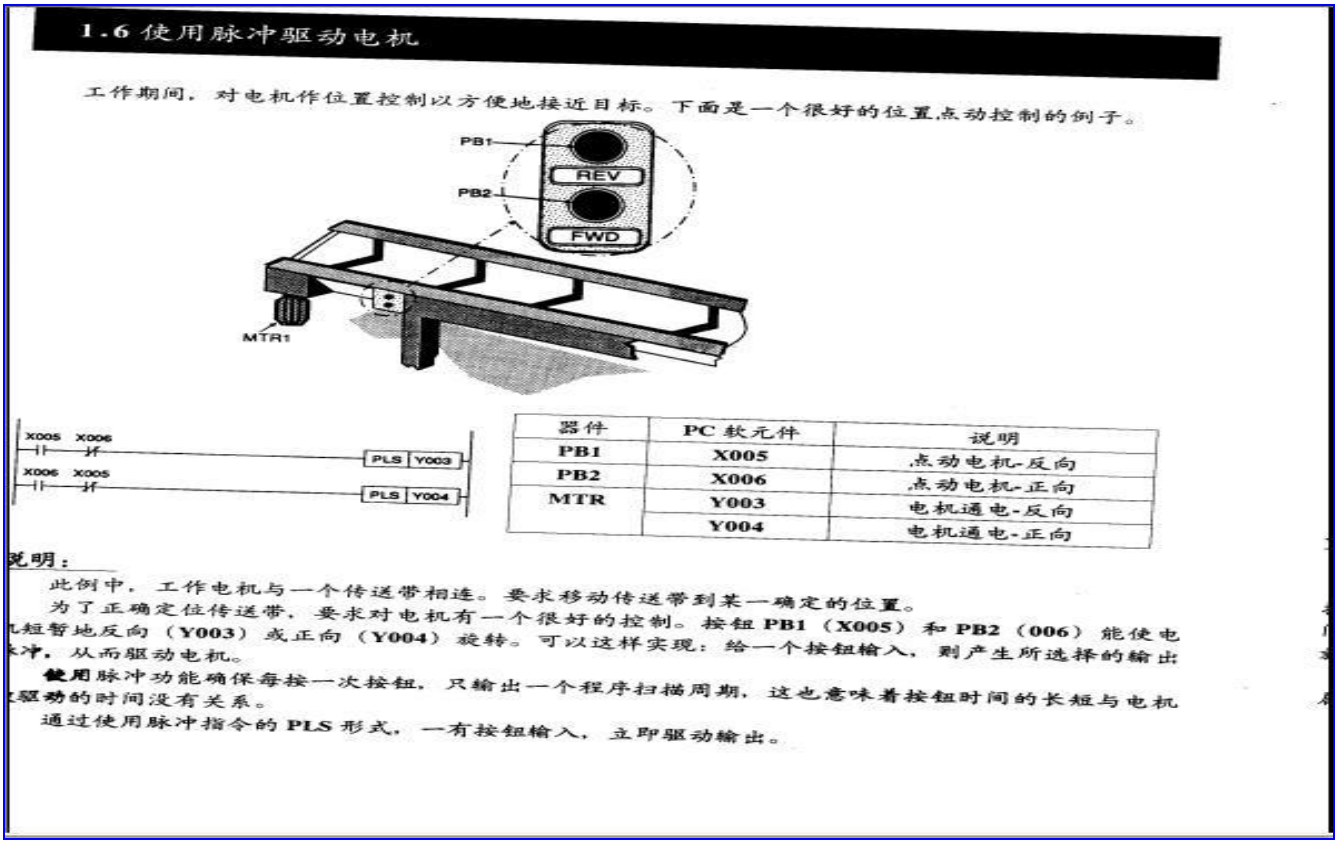
先触发的接近开关“锁定”另一个接近开关驱动的输出。程序非常简单地规定动作为“单一/或”方式，永远不同时发生。

C写法

```
void PLC_Task(void)
{
    if(X1==1 && Y2==0)SET(Y1);  //如果X1接通、Y2断开，Y1就接通
    else                RST(Y1);  //否则Y1就断开

   if(X2==1 && Y1==0) SET(Y2);  //如果X2接通、Y1断开，Y2就接通
    else                RST(Y2);  //否则Y2就断开
 }
```

1.6使用脉冲驱动电机

梯形图



1.6 使用脉冲驱动电机

工作期间，对电机作位置控制以方便地接近目标。下面是一个很好的位置点动控制的例子。

器件	PC 软元件	说明
PB1	X005	点动电机-反向
PB2	X006	点动电机-正向
MTR	Y003	电机通电-反向
	Y004	电机通电-正向

说明：

此例中，工作电机与一个传送带相连。要求移动传送带到某一确定的位置。

为了正确定位传送带，要求对电机有一个很好的控制。按钮 PB1（X005）和 PB2（006）能使电机短暂地反向（Y003）或正向（Y004）旋转。可以这样实现：给一个按钮输入，则产生所选择的输出脉冲，从而驱动电机。

使用脉冲功能确保每按一次按钮，只输出一个程序扫描周期，这也意味着按钮时间的长短与电机驱动的时间没有关系。

通过使用脉冲指令的 PLS 形式，一有按钮输入，立即驱动输出。

C写法

```
void PLC_Task(void)
{
    if(X5==1 && X6==0 && M0==0){SET(Y1);M0=1; } //如果X1接通、Y2断开，Y1就接通
    else                                RST(Y1);  //否则Y1就断开
    if(X5==0 && M0==1)M0=0;

   if(X6==1 && X5==0) SET(Y2);  //如果X2接通、Y1断开，Y2就接通
    else                      RST(Y2);  //否则Y2就断开
 }
```

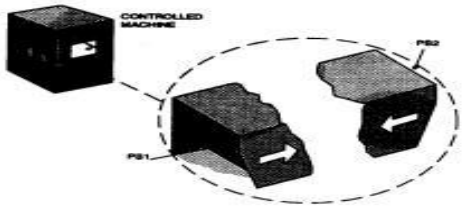
个人理解：Y3、Y4在条件成立下，只接通一个扫描周期时间；

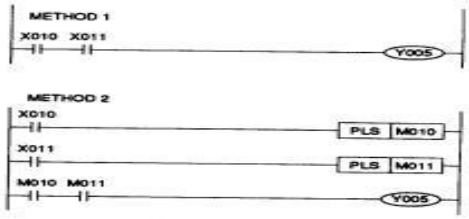
1.7“无暇手柄”安全系统

梯形图

1.7 “无暇手柄”安全系统

对于控制系统工程师，一个常用的安全手段是使操作者必须处在一个相对任何控制设备都很安全的位置。其中最简单的方法是使操作者在远处操作。





器件	PC 软元件	说明
PB1	X010	左手按钮
PB2	X006	右手按钮
	Y005	预定作用
	M010，M011	同时操作按钮的控制软元件

说明：

这一节讲述的安全系统被许多工程师称为“无暇手柄”这是一个很简单但非常实用的控制方法。“柄”是用来指初始化和操作被控机器的方法。用两个按钮构成一个“无暇手柄”。两钮必须同时按下。用此方法能防止只用一手就能进行控制的情况。常把按钮放在控制板上直接相对的两端。按钮间的距离保持在300mm（12英寸）左右。为了防止操作者误碰按钮，或者采取某种方式使得一只手就能操作按钮，每个按钮都凹放在一个金属罩下。

最后作用是使操作者位于一个没有危险的位置。两个操作者的手都在忙于控制按钮。按钮上的金属罩使手得到保护，而且，也不容易更改对专用设施的安排。

程序“方法1”是一个简单的两键控制，而“方法2”进了一步，要求按钮同时按下。

C写法

```
void PLC_Task(void)
{
    if(X10==1 && X11==1)        {SET(Y5);} //如果X10、Y11同时按下，Y5就接通
    else                        RST(Y5);  //否则Y1就断开
 }
```

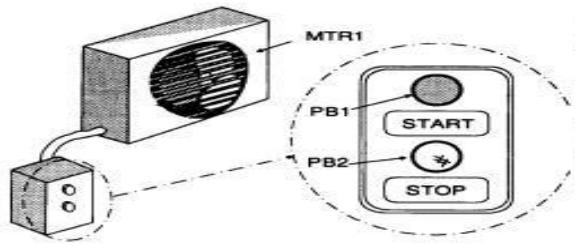
```
void PLC_Task(void)
{
    if(X10==1 && M0==0)  {SET(M0);SET(M10);} //X10按下,M10输出一个扫描周期,M0当按键防重复起动用
    else                 RST(M10);  //否则M10就断开
    if(X10==0 && M0==1)  RST(M0);
    if(X11==1 && M1==0)  {SET(M1);SET(M11);} //X10按下,M10输出一个扫描周期,M0当按键防重复起动用
    else                 RST(M11);  //否则M10就断开
    if(X10==0 && M1==1)  RST(M1);

    if(M10==1&& M11==1)SET(Y5);
    else                RST(Y5);
 }
```

1.8一个简单的自锁电路

梯形图

1.8 一个简单自锁电路

一个用以控制风扇开始/停止操作的简单自锁电路。



器件	PC 软元件	说明
PB1	X000	起动按钮
PB2	X001	停止按钮
MTR1	Y001	电机电源

说明：

按下按钮 **PB1**，风扇起动。按钮 **PB1** 在 PC 中表示为输入 **X000**。**X000** 接通，输出 **Y001** 被接通，这样使得电扇 **MTR1** 运行。若为使电扇运转而一直按着 **PB1**，那是很不方便的。一个小的自锁电路由此产生，它通过把程序输出 **Y001** 当作一个输入条件来实现。这意味着按钮 **PB1** 只需按一下，电扇就能连续运转起来。但是，电扇怎么停呢？使用与停止按钮 **PB2** 相连的常闭触点 **X001** 输入，锁定输出 **Y001** 就被“关断”，电扇就停止了。

安全注意：

对一个要求更安全控制的程序，停止按钮的输入应是一个常开触点，即 **X001** 为常开。实际按钮应该为常闭类型的物理接点。这意味着如果停止按钮失效，输入 **X001** 会消失，系统将停止，即变得安全。这称之为失效安全。

C写法

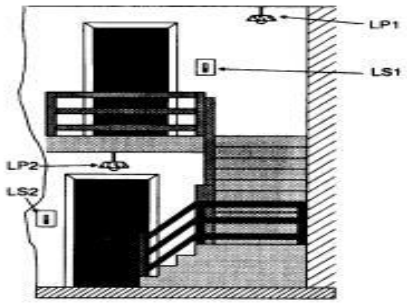
```
void PLC_Task(void)
{
    if((X0==1 || Y1==1) && X1==0) {SET(Y1);}   //X0按下、Y1接通,X0松开，Y1自锁保持；
    else                          RST(Y1);     //X1按下断开Y1；
}
```

1.9一个简单的双重控制系统

梯形图

1.9 一个简单双重控制系统

最简单的双重控制电路是一个楼上/楼下照明控制系统。



器件	PC 软元件	说明	
LS1	X002	电灯开关-楼梯顶	
LS2	X003	电灯开关-楼梯底	
LP1	Y001	灯接点	楼梯顶
LP2			楼梯底

说明：

本例模拟楼上/楼下硬布线的照明控制。每个开关连接一个输入点：LS1-X002 和 LS2-X003。要使灯动作，两个开关必须在同一状态中：都是“ON”或都是“OFF”，与硬布线解决方案一样，灯的 ON/OFF 控制能从任何一个开关产生。

附注：两盏灯 LP1 和 LP2 由同一输出 Y003 驱动。

C写法

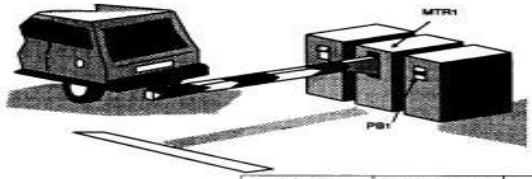
```
void PLC_Task(void)
{
    if((X2==1 && X3==1) ||(X2==0 && X3==0)) {SET(Y1);}   //两处的X2/ X3接通,Y1接通；
    else                                                 RST(Y1);    //否则Y1断开；
}
```

1.10一个定时关开关

梯形图

1.10 一个定时关电路

下面的定时关结构用来延迟检票栏的关闭。



器件	PC 软元件	说明
PB1	X000	收停车票
MTR1	Y000	升起栏杆
	T000	栏杆复位到水平位置前的时间延迟

说明：

当一辆车到达检票栏时，按钮 PB1 被司机按下，接收一张停车票后，允许车进入停车场。

一收到 X000（PB1）信号，输出驱动 MTR1，栏杆升起。因为 PB1 的初始输入是瞬时的，所以用来驱动栏杆升起的输出自锁。定时器计时 10 秒后，自锁关断。

这时，输出 Y000 关断，栏杆回到水平位置，等待下一位顾客。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100);                                    //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                        //复位定时器0
 }
```

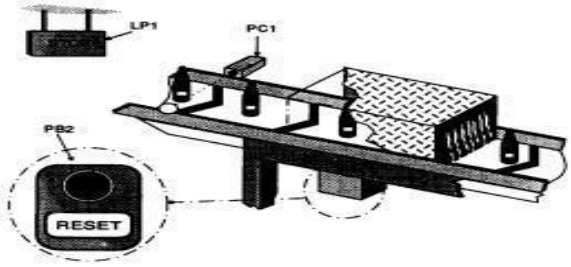
1.11一个简单计数器

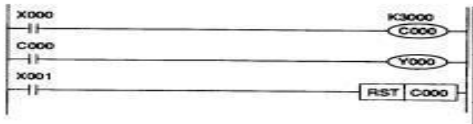
梯形图

梯形图

1.11 一个简单计数器

这个计数程序用来累计随传送带移动的瓶子数量。





器件	PC 软元件	说明
PC1	X000	瓶子计数光电管
LP1	Y000	停止装载指示灯
PB2	X001	复位计数器按钮
	C000	计数器

说明：

当瓶子在传送带上移过来时，它们挡住光电管 PC1 的光线。每次光线被挡住，代表 PC1 的输入 X000 变为 ON，程序起动一个计数器。这里，C000 用来“记录”经过 PC1 的瓶子数量。C000 事先设定一个计数上限，这样就能提供一天或一班次处理的瓶子总数。本例中上限定为 3000。

一旦计数器达到限度值，C000 的输出线圈闭合。为了向外部表示计数任务已完成，计数器 C000 的一个触点用来激活输出 Y000，接着，起动“STOP”灯 LP1，从而使操作者知道目的已达到。

因为计数器要保存它的数据，所以需要一种复位当前计数值的方法，可以用“复位”按钮 PB2 实现。PB2 对应于输入 X001，它使计数器设定为 0。“STOP”灯关断，整个系统准备下一批 3000 个瓶子经过。

C写法

```
void PLC_Task(void)
{
    if(X0==1 && M0==0) {SET(M0); C_T(0, 3000);}  //X0按下 ,计数器0计数，设定值为3000个；
    if(X0==0 && M0==1) {RST{M0;}

    if(C_Bit[0]==1)SET(Y0); //计数值到达3000个，Y0接通；
    else               RST(Y0);

     if(X1==1)C_T_Rst(0);//X1按下，复位计数器0
}
```

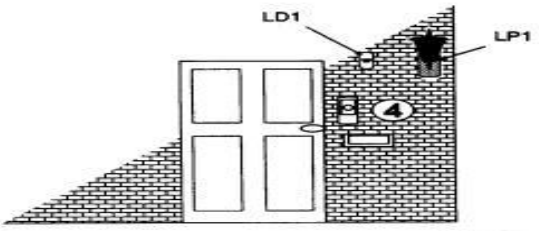
1.12一个定时操作

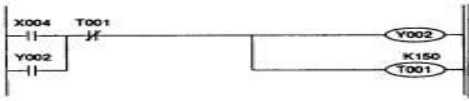
梯形图

梯形图

1.12 一个定时操作

这种家居安全使用的“示警灯”是很常见的。





器件	PC 软元件	说明
LD1	X004	活动检测器
LP1	Y002	灯
	T001	灯亮的时间

说明：

传感器 LD1 提供初始输入给电路。如果某个活动被 LD1 检测到，程序输入 X004 变为 ON，接着安全灯 LP1 在 Y002 驱动下变为 ON。安全灯会持续 15 秒，这通过锁定灯输出（Y002）来实现，但是接着用一个来自定时器 T001 的触点关断锁定。

因为定时器与灯线圈同时起动，这样可保证灯保持开状态 15 秒。计时结束时，定时器的常闭触点断开，因而切断锁定电路。

C写法

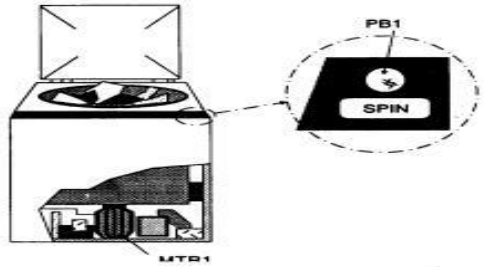
```
void PLC_Task(void)
{
    if((X4==0 || Y2==1) && T_100MS_Bit[1]==0)) {SET(Y2); T100MS_Set(1,150) }   //X0按下 Y1自锁；
    else                                        {RST(Y2); T100MS_Rst(1); //定时时间到Y0断开；
 }
```

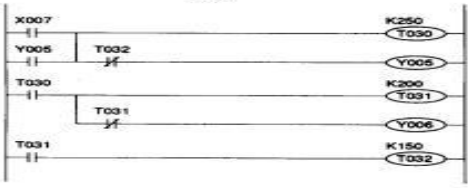
1.13时序操作

梯形图

1.13 时序操作

如下所示的洗衣机的旋转过程从开始到结束都是使用定时操作来顺序进行的。





器件	PC 软元件	说明
PB1	X007	手动选择-旋转启动
MTR1	Y005	低速旋转
MTR2	Y006	高速旋转
	T030	低速旋转时间
	T031	低速旋转时间
	T032	减速时间

说明：

按下“SPIN”按钮 PB1，使得 PC 输入点 X007 有效。这样就启动了一系列完全由定时器控制的旋转操作。

当 X007 有效时，洗衣机的电机开始低速旋转，初始化由输出 Y0005，MTR1 完成。这个输出线圈接着锁定整个程序，进行一个周期的运行。

25 秒后，定时器 T030 的常开触点闭合，输出 Y006，MTR2 使洗衣机再高速旋转 20 秒，并使定时器 T031 有效。

定时器 T031 定时结束后，高速输出 Y006，MTR2 被取消。接着定时器 T032 提供 15 秒时间使洗衣机减速。

这个定时过程一结束，整个序列复位，等待下一次“SPIN”钮的按下。

C写法

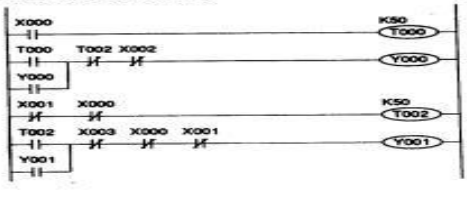
```
void PLC_Task(void)
{
    if((X7==1 || Y5==1) T_100MS_Set(0,250);      //X7接通 Y5自锁，定时器0打开设定25秒
    else                T_100MS_Rst(0);          //Y5断开,定时器0复位
   if((X7==1 || Y5==1) && T_100MS_Bit[2]==0)) {SET(Y5);}  //X7按下 Y5动作自锁；
    else                                        RST(Y5);   //定时2时间到Y5断开；

    if(T_100MS_Bit[0]==1) T_100MS_Set(1,200);//定时器0时间到,打开定时器1设定20秒
     else                 T_100MS_Rst(1);    // ；定时器0复位,复位定时器1

     if(T_100MS_Bit[0]==1 && T_100MS_Bit[1]==0 )SET(Y6);//定时器0时间到,Y6接通
     else                                        RST()Y6;//定时器1时间到,Y6断开

    if(T_100MS_Bit[1]==1) T100MS_Set(2,150); //定时器1时间到，打开定时器2，设定15秒；
    else       T100MS_Rst(2);                //定时器1复位，复位定时器2
 }
```

1.14自动门

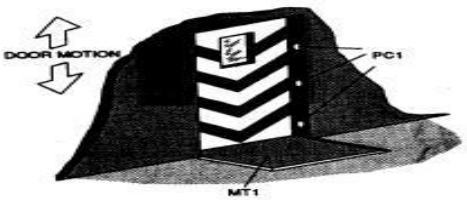
梯形图

1.14 自动门

检测和延时是预防事故发生的有效措施。可以说不可能有绝对完备的安全措施，这里的两项技术可以加强安全措施。



器件	PC 软元件	说明
MT1	X000	压力垫
PC1	X001	门通道光电管（并行连接）
	X002	门在上升位置
	X003	门在关闭位置
	Y000	门升起
	Y001	门下降
	T000	上升延迟-确认门真地将上升
	T002	下降延迟-检查门区域是否无人



说明：

观察一扇门，它的动作看起来非常简单，关上，打开。为什么门需要一个可编程控制器呢？凭经验想一想，便发现门从来不会夹住一个人，即压扁/碰撞用户。在许多情况下，门过早地关上，这是很不方便的。因此，大量控制和安全措施加入到这些看似普通的对象中。在本例所示中，当压力垫有感应时，输入 **X000** 接通，门将会打开。

为防止不必要的开和关操作，压力垫信号的接受必须有一段“检测”时间。之后，输出 **Y000** 将起动开门的过程，这个输出自锁。

当门安全打开时，限位开关 **X002** 被激活，取消“打开”的输出。门一直保持开，直到压力垫和光电管信号消失。由 **T002** 设定的一个小停顿使关门的动作延迟。这个延迟检测是否有人立即跟着第一人经过门道。如果不再有人经过门道，输出 **Y001** 接通，门会继续关，直到“关”限位开关 **X003** 起作用。

在关过程的任何时刻，一个人要经过门道时，开门过程会再次起动。

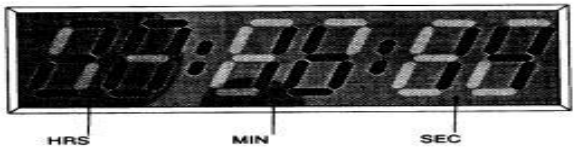
C写法

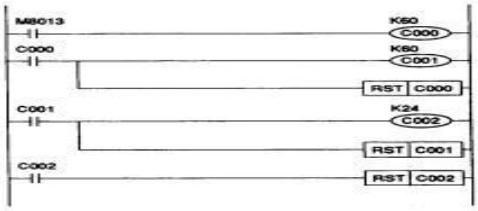
```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                                  //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                    //复位定时器0
 }
```

1.15使用计数器的24小 梯形图 梯形图

1.15 使用计数器的 24 小时钟

下例中使用三个计数器“计时间”。





器件	PC 软元件	说明
SEC	C000	秒计数
MIN	C001	分计数
HRS	C002	时计数
	M8013	1 秒脉冲

说明：

这个小程序在 PC 一投入运行就开始工作。其操作的关键在于专用辅助继电器 M8013。这个元件提供一个周期为 1 秒的脉冲。计数器根据这个“时钟脉冲”对过去的时间以秒计数。如果这个计数器到了 60 就复位，便可知道过去了一分钟。

如果第二个计数器对计数器 C000 复位次数计数，则可知道过去了多少分钟。

因此，接下来的逻辑与前面相同，只是用 C001 来计“分”。再进一步，用 C002 来计“时”，从而完成一个完全独立的时钟。时间可从 3 个计数器上直接读出。使 C002 的“设定”值等于 24，就得到一个 24 小时时钟，或者设定值等于 12 时，一个标准的 12 小时时钟就能工作了。

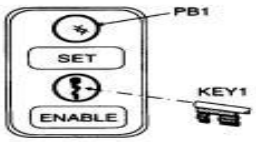
```
void PLC_Task(void)
{
    //定时器0产生1秒脉冲，C0为秒计数，C1为分钟计数，C2为时计数；
     if(M0==0)
      {
                    M0=1;                   //自锁，防止再次设定
                    T100MS_Set(0,10);       //1秒定时
      }
     if(T_100MS_Bit[0]==1)                  //1秒定时到开始秒
      {
                    T100MS_Rst(0);          //复位秒脉冲
                    M0=0;                   //重新秒脉冲定时打开
                    C_T(0, 60);             //60秒计数
      }
     if(C_T_Bit[0]==1)                      //60秒计数到
      {
                    C_T_Rst(0);             //复位秒
                    C_T(1, 60);             60分钟计数
      }
     if(C_T_Bit[1]==1)                      //60分钟计数到
      {
                    C_T_Rst(1);             //复位分钟
                    C_T(2, 24);             //24小时计数
      }
    if(C_T_Bit[2]==1)C_T_Rst(2);            //60秒计时到 复位
}
```

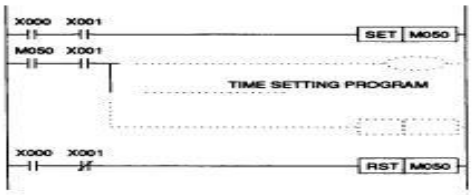
1.16置位和复位一个操作模式

梯形图

1.16 置位和复位一个操作模式

这个灵巧的 SET/RST 程序能使一段程序“转换”为 ON 或 OFF。





器件	PC 软元件	说明
PB1	X000	置位-按钮
KEY1	X001	执行钥匙开关
	M050	内部操作标志

说明：

本例中，SET/RST 程序用来起到一个管理作用，比如设定时间。为了能够进行这种模式操作，受权人必须有一把执行“钥匙”（钥匙开关）。

当需要更改时间时，使用者打开钥匙开关（Key1），接着按“SET”按钮（PB1）。钥匙开关和按钮的动作给可编程控制器（PC）两个输入信号 X000 和 X001。当 PC 一收到这些输入，“时间设定模式”被 SET 指令激活，它将内部标志 M050 置位。

标志 M050 会一直保持 ON，即使输入 X000 和/或 X001 不再出现。不过，考虑到安全，在执行“时间设定”操作时，本程序要求钥匙一直留在钥匙开关中。

当 M050 为 ON，并且钥匙开关有效时，相应的“时间设定”的程序会被处理。当时间设定完毕，取下安全钥匙，并再次按 SET 钮。则内部标志 M050 复位（RST），这样就不能再做“时间设定”了。

C写法

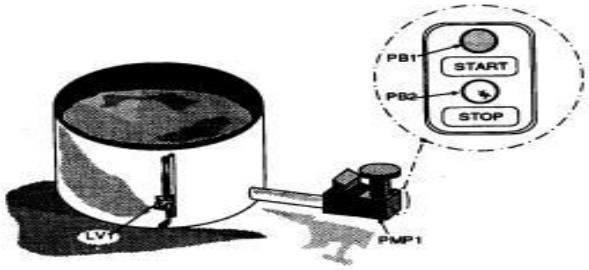
```
void PLC_Task(void)
{
    if(X0==1 &&  X1==1) SET(M0);     //X0 X1同时接通 ，  M5置位=1；
    if(X0==1 &&  X1==0) RST(M0);     // X0 接通 X1断开，M5复位=0；


 }
```

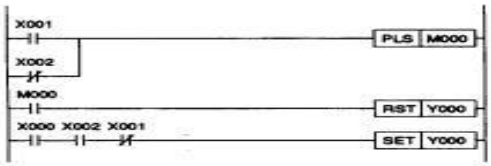
1.17失效安全

梯形图

1.17 失效安全

如果容器为空或按下停止按钮，抽水泵停止工作。





器件	PC 软元件	说明
PB1	X000	起动按钮
PB2	X001	停止按钮
LV1	X002	容器低水位
PMP1	Y000	泵电机

说明：

泵从容器中抽水，按下标明为 Start 的按钮 PB1（X000），泵开始工作。

只要容器中有水（X002 为 ON），而且标明为 Stop 的按钮 PB2（X001）没有按下，泵会一直工作；PMP1 由输出 Y000 驱动。

如果在泵工作期间的某一时刻，容器抽干了，即低水位检测器 LV1 被触发（X002 会变为 OFF），泵会自动停止。实际上，低水位检测器提供可编程控制器常闭触点输入，也就是说，有水时，输入 X002 为 ON，实际的低水位检测器为 OFF。因此，若低水位检测器失效，即总是 ON（X002 会变为 OFF），系统不允许泵工作，故在故障时是安全的。

如果按下停止按钮，泵也会停止。泵已经停止时，必须按下起动按钮来复位/重起动泵。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || X2==1)      //X0按下 Y1自锁；
    ;   //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                    //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                       //复位定时器0
 }
```

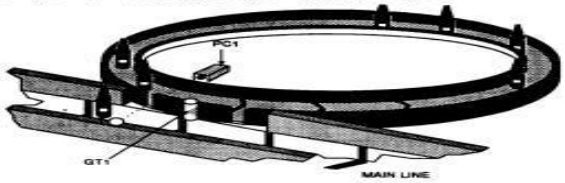
1.2信号非允许动作

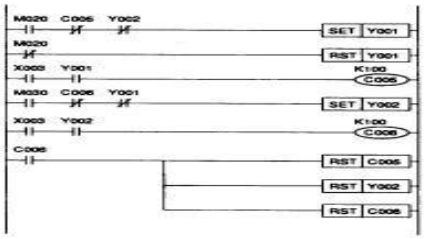
梯形图

梯形图

1.18 置位和复位操作

主传送带上有障碍物时，瓶子会被传送到再循环线上。





器件	PC 软元件	说明
PC1	X003	检测瓶子光电管
GT1	Y001	再循环线门设置
	Y002	主线门设置
	C005	进入再循环线的瓶子个数
	C006	进入主线的瓶子个数
	M020	允许瓶子进入再循环线
	M030	允许瓶子进入主线

说明：

如果要使用再循环线路，则激活内部标志 M020，就能使用再循环线。然而，如果再循环线满了（即 C005 起作用）或者正处于清空过程中（Y002 起作用），再循环线忽略 M020 标志。当 Y001 有效时，门 GT1 反时针旋转，允许瓶子从主传送带转到再循环线上。当瓶子完成这次转移，光电管 PC1 光线被遮断，这样给出一个输入 X003，这个输入与计数器一起作用可以确定再循环线上的瓶子数目。当标志 M020 不起作用时，输出 Y001 复位，门 GT1 返回到中间位置（弹簧返回）。

瓶子离开再循环线时，标志 M030 必须出现。在进入和清空的选择之间又有一个互锁，门 GT1 被输出 Y002 设定，使得产生顺时针旋转。PC1 再一次提供计数信号（C006），将所有的瓶子计数。这个系统设计成在一次运行中或是装满或是卸空再循环线。这就是为什么当最后一瓶离开再循环线时，计数器要复位。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                              //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                //复位定时器0
 }
```

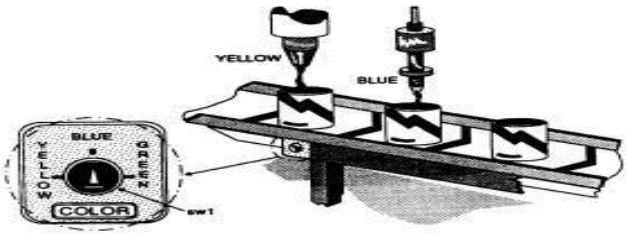
1.2信号非允许动作

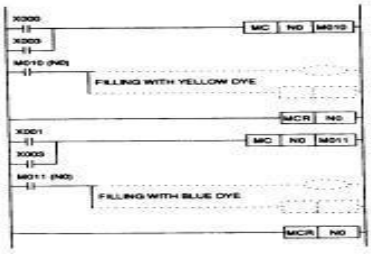
梯形图

梯形图

1.19 选择程序的部分

颜料罐从生产线上移过来，注入两种不同颜料，所以能有三种颜色选择。





器件	PC 软元件	说明
SW1	X000	选择黄颜料
	X001	选择蓝颜料
	X003	选择绿颜料-黄蓝混合颜料
	M010（NO）	黄颜料注入底料混合物的程序
	M011（NO）	蓝颜料注入底料混合物的程序

说明：

本例说明一段特定程序只有在更大的主体程序要求时才能被访问。颜料罐注入规定的颜料，黄或蓝色。如果两种颜料都使用，则产生绿色。

颜色的选定是通过旋转开关 SW1 到适当的位置。接着输入 X000、X001 和 X003 会被接收到。因为把选定的颜料加到罐中的机器各不相同，每种选择要求对应一个特定程序。当颜色选定时，每个程序就“转换”到开或关。

当绿色被选中，两个程序顺序地运行，使用主控制指令控制应该哪个程序运行。MC 和 MCR 指令标注在每个程序的开始和结尾，只有当起控制作用的“主控制”被激活时，才能执行这些指令之间的程序步。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                              //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                //复位定时器0
 }
```

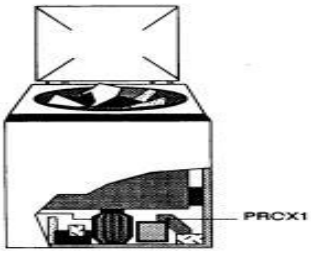
1.2信号非允许动作

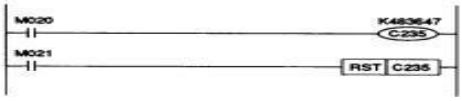
梯形图

梯形图

1.20 高速计数

洗衣机的滚筒旋转被计数，数据可以用在后面编程中控制循环长度，进行速度控制或提供维护帮助。





器件	PC 软元件	说明
PROX	X000	采集滚筒旋转信息的接近传感器
	M020	控制计数操作的触发器
	M021	外部复位触发器
	C235	高速 32 位计数器（注：输入 X000 自动作为此计数器的输入）

说明：

当洗衣机滚筒旋转时，固定在滚筒底部的一个小标记或指示器经过一个接近传感器（PROX1）。这个传感器驱动输入 X000。因为使用了高速计数器，输入 X000 自动转为此计数器的输入。内部 M 线圈或辅助继电器 M020 用来控制计数操作。而辅助继电器 M021 用来复位计数器。

附注：高速计数器是 32 位的元件，这意味着计数器设定值的选择范围可以是 1 到 2，147，483，647。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                              //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                //复位定时器0
 }
```

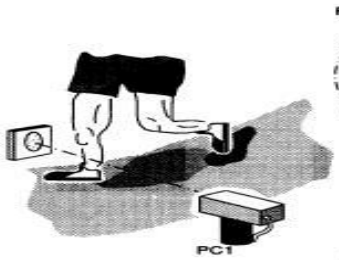
1.2信号非允许动作

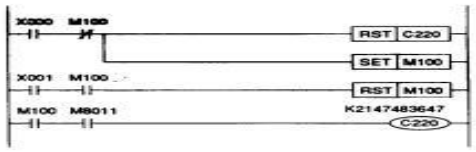
梯形图

梯形图

1.21 精确计时

运动俱乐部里训练课用的跑表。





器件	PC 软元件	说明
PB1	X000	启动按钮
PC1	X001	停止检测光电管
	M100	内部标志-跑步开始
	C220	当前一圈时间，以 10ms 为单位 （注：C220 是 32 位计数器）
	M8011	内部 10msec 时钟脉冲

说明：

按下按钮 PB1，产生输入 X000。程序由 X000 的第一次出现设定跑步标志 M100。

这个标志一设定，就进行赛跑或跑步，计数器用来对特殊 M 线圈 M8011 的脉冲计数。这个 M 线圈是一个 10 毫秒时钟脉冲。计数值与以毫秒计的跑步时间成正比。

当跑步者挡住底线上的光电管 PC1 时，计数和由此得到的计跑时间停止。此时，输入 X001 被接收，并且赛跑标志 M100 复位，从而使计数器 C220 停止工作。为了检查，计数值被保存下来，简单地除以 100 会得出以秒计的时间，再除以 60 则得出分和小时的结果。

在下一次跑步开始时，按下按钮 PB1 时，计数器 C220 复位。

如果运动员都是跑得特别快的选手，那么就可以用一个 16 位计数器如 C0 或 C10 代替。如果这样，设定值应该是 32767。

C写法

```
void PLC_Task(void)
{
    if((X0==1 || Y0==1) && T_100MS_Bit[0]==0)) {SET(Y0);}   //X0按下 Y1自锁；
    else                                                    RST(Y0);    //定时时间到Y0断开；

    if(Y0==1) T100MS_Set(0,100)                              //Y0接通,打开定时器0，设定10秒；
    else      T100MS_Rst(0);                                //复位定时器0
 }
```