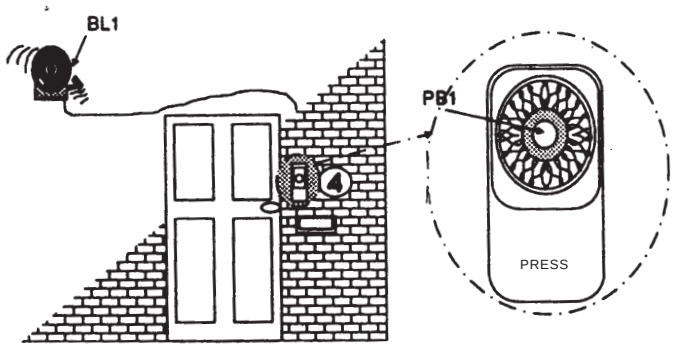
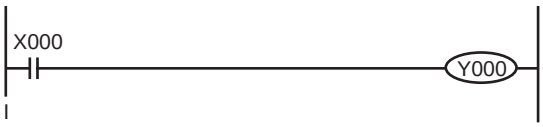
1.1 一个开环电路

用在门铃上的一个小开环电路。





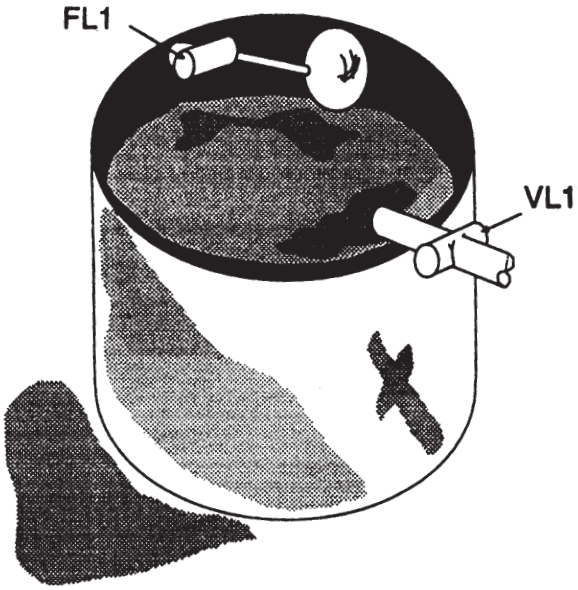
器件	PC 软元件	说明
PB1	X000	门铃按钮
BL1	Y000	门铃

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 工作完全依赖于 PB1。

1.2 信号断开允许动作

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

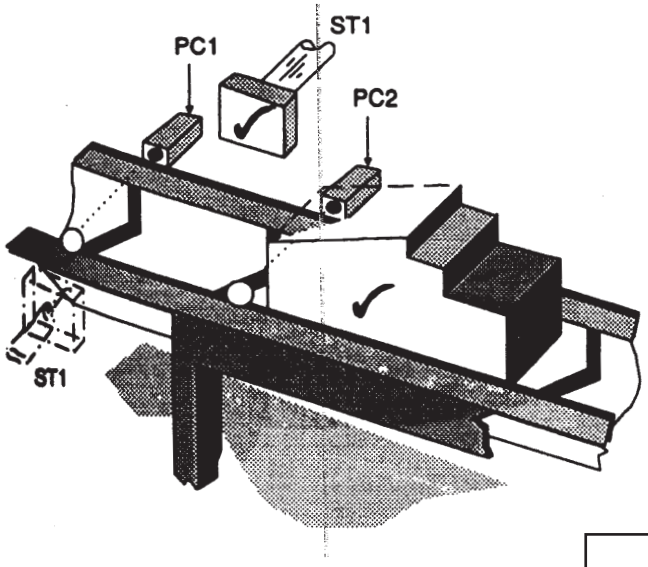
说明：

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。

1.3 串联常开触点引起一个动作

这个电路检测随传送带运动的物体的位置。



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

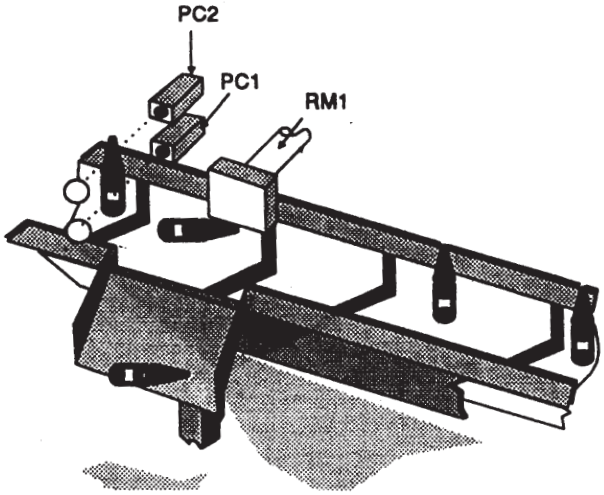
说明：

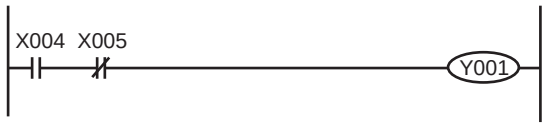
当包裹从传送带上送过来时，经过两个光电管 PC1 和 PC2，这两个光电管用来检测传送带上包裹的位置。当信号 X002 和 X003同时被接收到，即：两个光电管同时被激活时，给包裹贴邮票的操作就能顺利完成，即 Y002 被驱动，把邮票 ST1 移向包裹

附注：贴邮票操作的移开和复位不在这里细述。

1.4 串联常闭触点引起一个动作

这个电路检测瓶子是否直立，如果不是，则它被抛到传送带外。





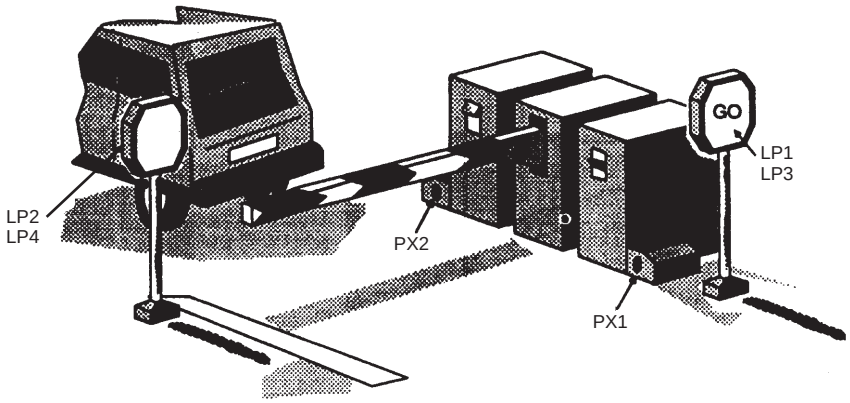
器件	PC 软元件	说明
PC1	X004	光电管自动检测瓶座
PC2	X005	光电管自动检测瓶顶
SM1	Y001	推出杆

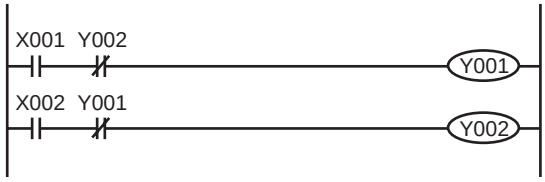
说明：

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。

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	Y002	STOP-正要进入的车辆
LP4		GO-正要离开的车辆

说明：

这个程序说明了一种互锁输出的简单方法，这也是确保安全的简便方法。在安全要求较高的控制场合，还应该使用机械的和/或物理的互锁。程序操作如下：

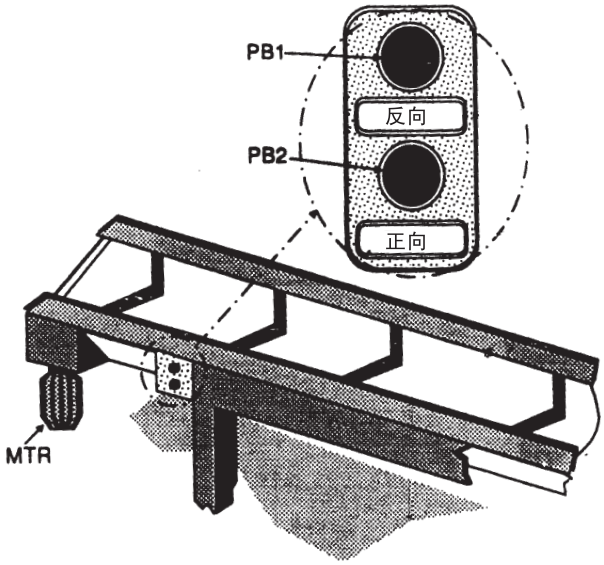
一辆车接近检票栏前，触发一个接近开关，是 PX1 还是 PX2，这决定于车来的方向。当 PX1 或 PX2 被触发时，输入 X001或 X002 信号被 PC 接收，每个输入触发一个输出 Y001 或 Y002。

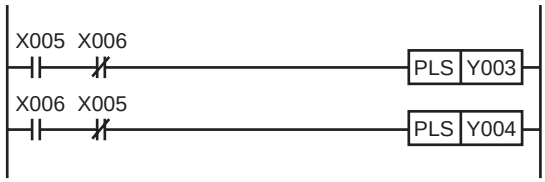
接着被驱动的输出使交通指示灯接通，允许车通过，也就是说，一盏灯指示 GO，而另一盏灯也由同一个输出信号控制，指示 STOP。如果有两辆车同时要通过检票栏，互锁结构会保证在任何时刻只有一辆车通过。

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。

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。

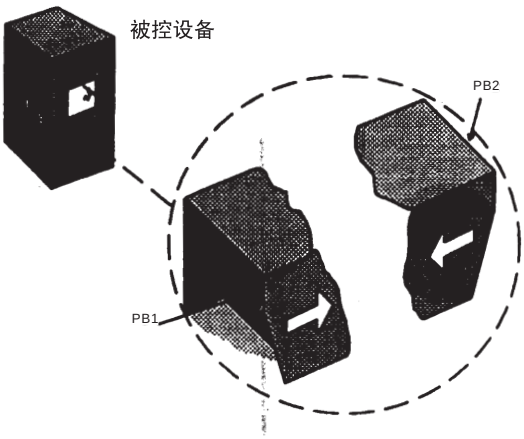
为了正确定位传送带，要求对电机有一个很好的控制。按钮 PB1 (X005) 和 PB2 (X006)能使电机短暂地反向 (Y003) 或正向 (Y004) 旋转。可以这样实现：给定一个按钮输入，则产生所选择的输出脉冲，从而驱动电机。

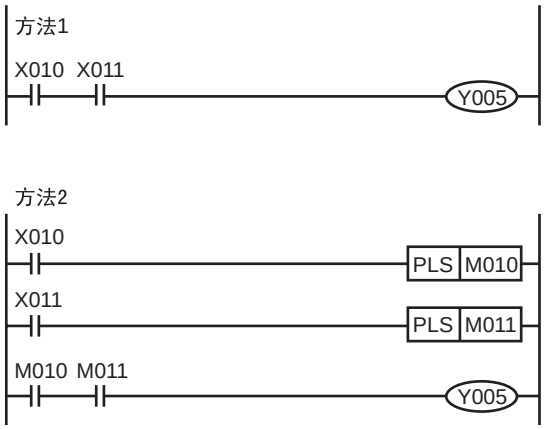
使用脉冲功能确保每按一次按钮，只输出一个程序扫描周期，这也意味着按钮输入时间的长短与电机被驱动的时间没关系。

通过使用脉冲指令的 PLS 形式，一有按钮输入，立即驱动输出。

1.7 “无暇手柄”安全系统

对于系统控制工程师，一个常用的安全手段是使操作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用作者在远处操作。





器件	PC 软元件	说明
PB1	X010	左手按钮
PB2	X011	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：

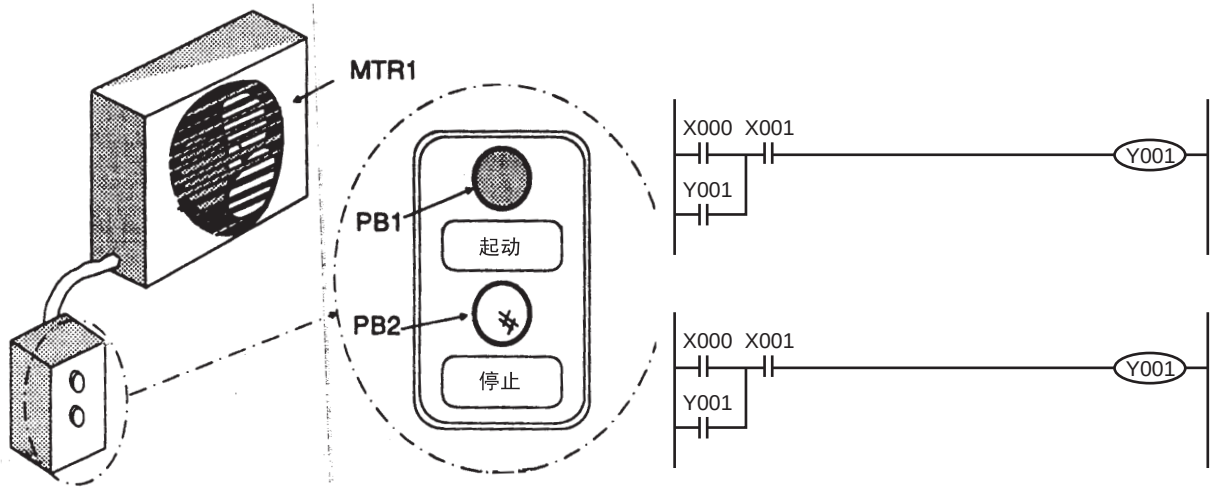
这一节讲述的安全系统被许多工程师称为“无暇手柄”这是一个很简单但非常实用的控制方法。

“柄”是指用来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，没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。操作者的两只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。

1.8 一个简单自锁电路

一个用以控制风扇起动/停止的简单自锁电路。



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：

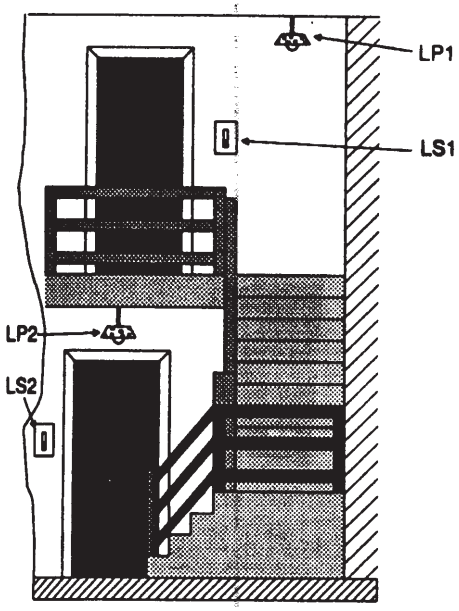
按下按钮 PB1，风扇起动。按钮 PB1 在 PC 中表示为输入 X000。X000 接通，输出 Y001 被接通，这样使得电扇 MTR1 运行。若为使电扇运转而一直按着 PB1，那是很不方便的。一个小的自锁电路由此产生，它通过把程序输出 Y001 当作一个输入条件来实现。这意味着按钮 PB1 只需按一下，电扇就能连续运转。但是，电扇怎么停呢？使用与停止按钮 PB2 相连的常闭触点 X001 输入，锁定输出 Y001 就被断开，电扇就停止了。

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，这被称为失效安全。

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 - 楼上	
LS2	X003	电灯开关 - 楼下	
LP1	Y001	灯	楼上
LP2			楼下

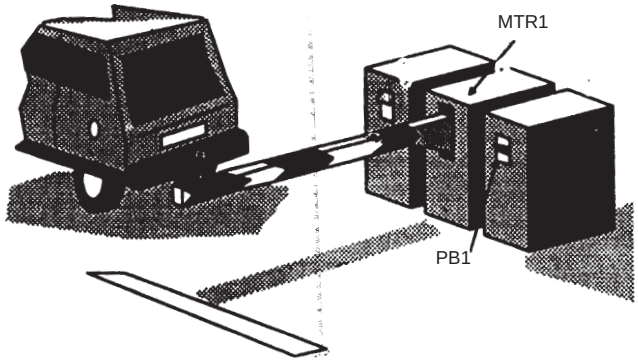
说明：

本例模拟楼上/楼下照明控制的布线，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进行。

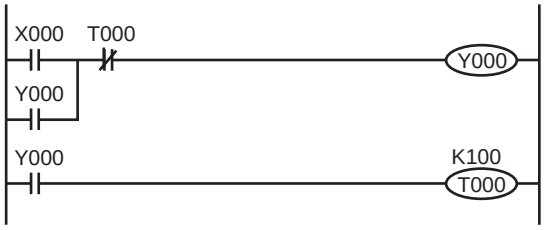
附注：两盏灯 LP1 和 LP2 由同一输出 Y001 驱动。

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	取停车票
MTR1	Y000	升起栏杆
	T000	栏杆复位到水平位置前的时间延迟



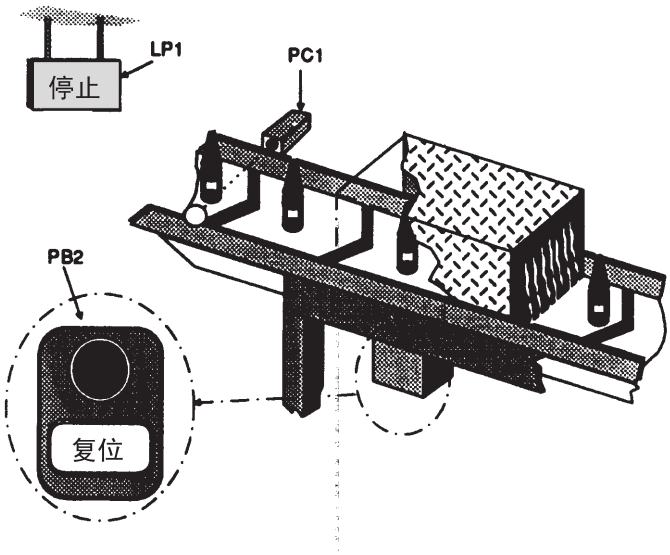
说明：

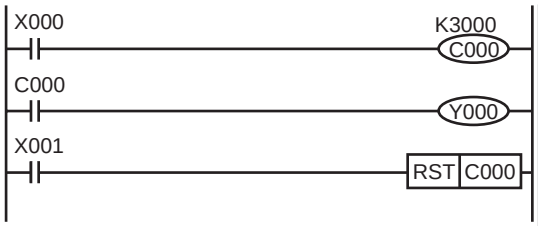
当一辆车到达检票栏时，司机按下按钮 PB1，取一张停车票后，允许车进入停车场。

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，自锁断开。这时，输出 Y000 断开，栏杆回到水平位置，等待下一位顾客。

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：

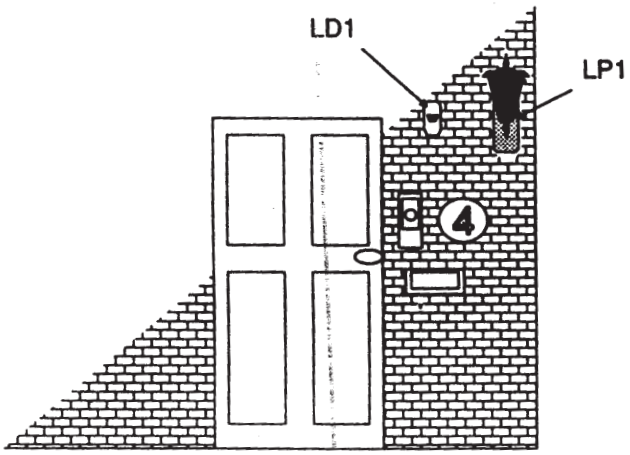
当瓶子在传送带上移过来时，它们挡住光电管 PC1 的光线。每次光线被挡住，代表 PC1 的输入 X000 变为 ON，程序启动计数器。这里，C000 用来记录经过 PC1 的瓶子数量。C000 事先设定一个计数上限，这样就能提供一天或一班次处理的瓶子总数，本例中上限定为 3000。

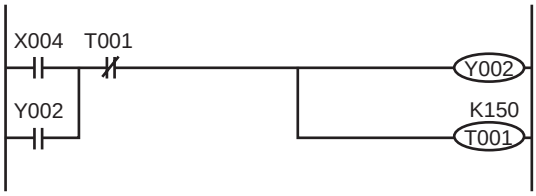
一旦计数器达到上限值，C000 的输出线圈闭合。为了向外部表示计数任务已完成，计数器 C000 的一个触点用来激活输出 Y000，启动“停止”指示灯 LP1，从而使操作者知道目的已达到。

因为计数器会保持它的数据，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，它使计数器当前值复位为 0。“停止”指示灯熄灭，整个系统准备下一批 3000 个瓶子经过。

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。





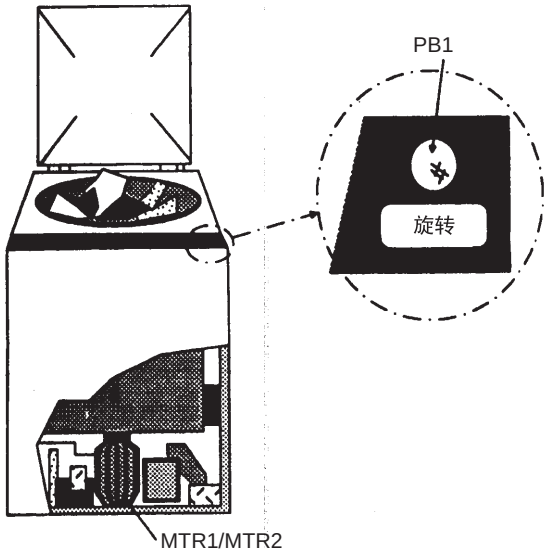
器件	PC 软元件	说明
LD1	X004	活动检测器
LP1	Y002	灯
	T001	灯亮的时间

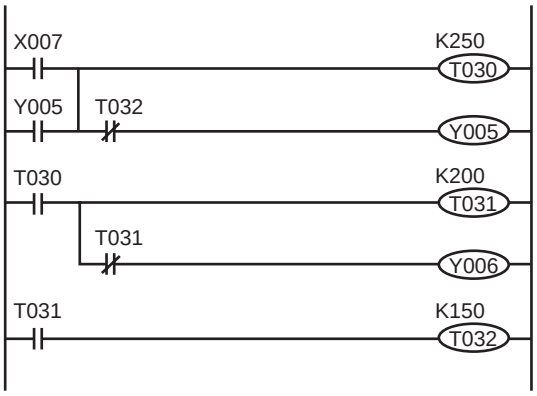
说明：

传感器 LD1 提供初始输入给电路。如果 LD1 检测到某个动作，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，这通过锁定灯输入 (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，这样可保证灯保持接通状态 15 秒。计时结束时，定时器的常闭触点断开，从而切断锁定电路。

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	高速旋转时间
	T032	减速时间

说明：

按下“旋转”按钮 PB1，使得 PC 输入点 X007 有效，由此启动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，由输出 Y005，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。

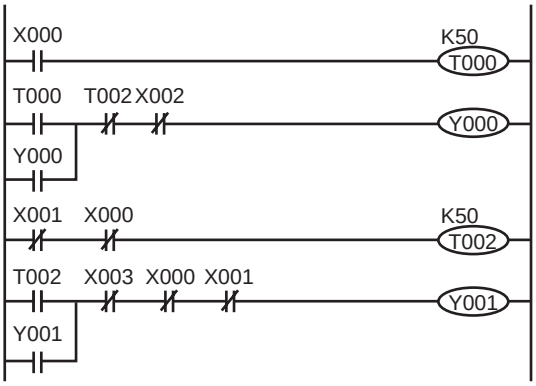
25秒后，定时器 T030 的常开触点闭合，输出 Y006，MTR2 使洗衣机再高速旋转 20 秒，并使定时器 T031 有效。

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

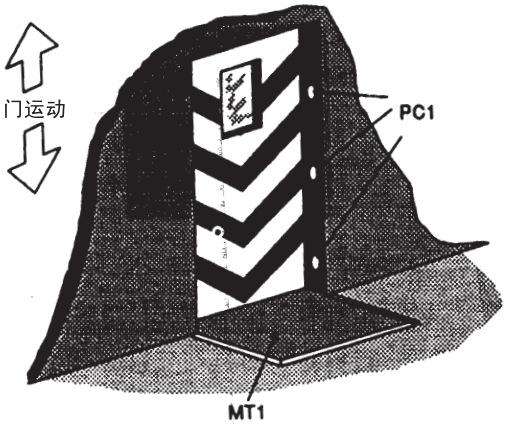
这个定时过程一结束，整个序列复位，等待下一次“旋转”钮的按下。

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，但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
	X002	门在打开位置
	X003	门在关闭位置
	Y000	门打开
	Y001	门关闭
	T000	打开延迟-确认门真的要打开
	T002	关闭延迟-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，即压扇/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。

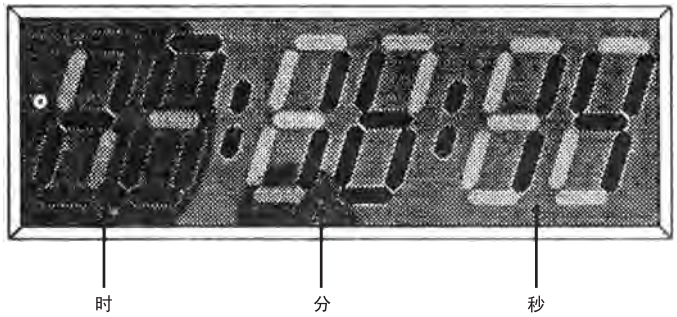
为防止不必要的开和关操作，压力垫信号的接收必须有一段“检测”时间。之后，输出 Y000 将起动开门的过程，这个输出自锁。

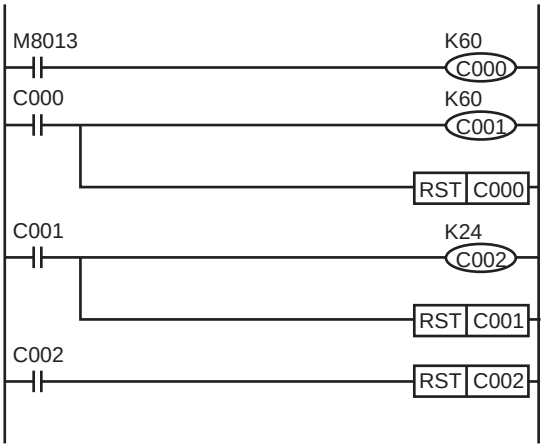
当门完全打开时，限位开关 X002 起作用，断开“打开”的输出。门一直保持开，直到压力垫和光电管信号消失。由 T002 设定的一个小停顿使关门的动作延迟，这个延迟检测是否有人立即跟着第一人经过门道。如果不再有人经过门道，输出 Y001 接通，门会继续关，直到“关”限位开关 X003 起作用。

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。

1.15 使用计数器的 24 小时钟

下列中使用三个计数器“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：

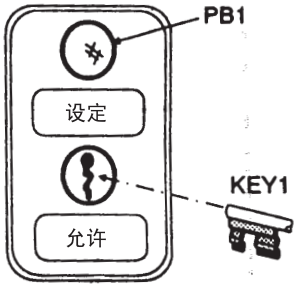
这个小程序在PC 一投入运行就开始运行。其工作的关键在于特殊辅助继电器M8013，这个元件提供一个周期为 1 秒的脉冲。计数器根据这个“时钟脉冲”对过去的时间以秒计数。如果这个计数器到了 60 就复位，便可知道过去了一分。

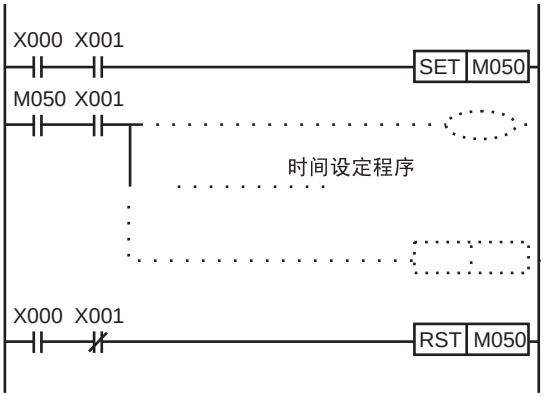
如果第二个计数器对计数器C000 动作的次数计数，则可知道过去了多少分钟。

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。

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	设定按钮
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)。

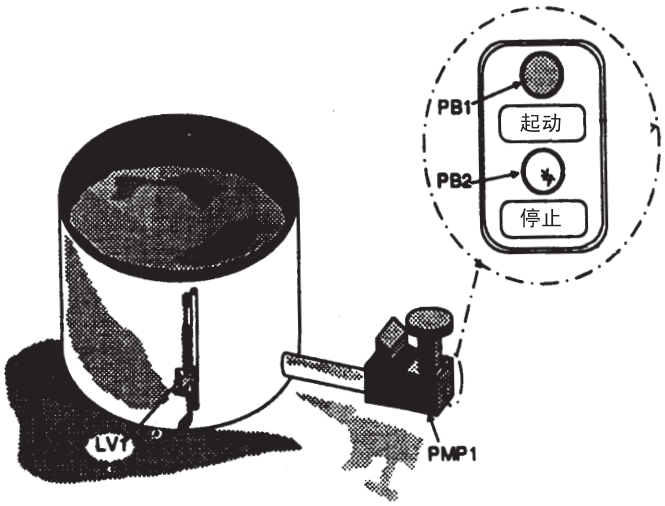
当需要更改时间时，使用者打开钥匙开关(KEY1)，接着按“设定”按钮(PB1)。钥匙开关和按钮的动作给可编程控制器(PC)两个输入信号X000 和X001。当PC 一收到这些输入，“时间设定模式”被SET 指令激活，它将内部标志M050 置位。

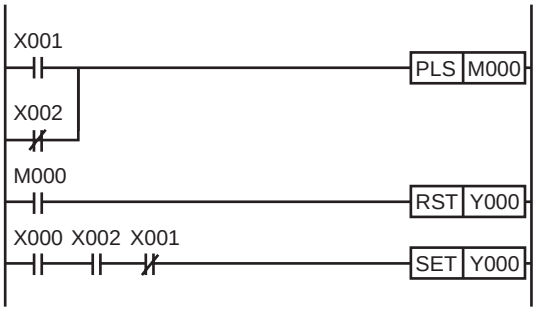
标志M050 会一直保持ON，即使输入X000 和/或X001 不再出现。不过，考虑到安全，在执行“时间设定”操作时，本程序要求钥匙一直留在钥匙开关中。

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

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	容器水位低
PMP1	Y000	泵电机

说明：

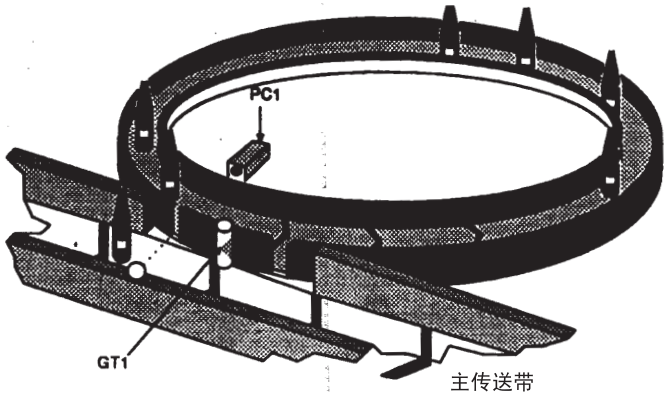
泵从容器中抽水，按下标明为“起动”的按钮PB1 (X000)，泵开始工作。

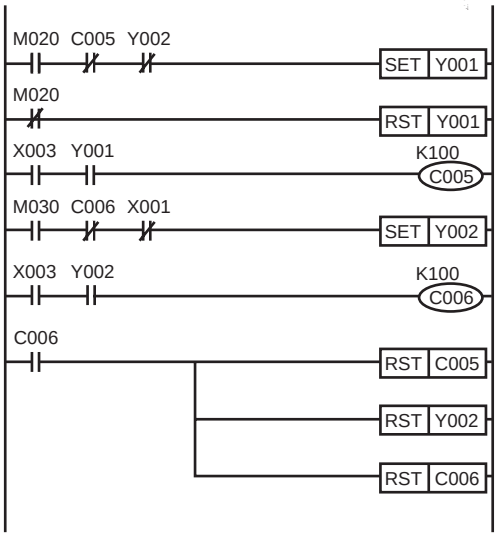
只要容器中有水(X002 为ON)，而且标明为“停止”的按钮PB2 (X001) 没有按下，泵会一直工作，PMP1 由输出Y000驱动。如果在泵工作期间的某一时刻，容器抽干了，即低水位检测器LV1 被触发(X002 会变为OFF)，泵会自动停止。实际上，低水位检测器提供可编程控制器常闭触点输入，也就是说，有水时，输入X002 为ON，实际的低水位检测器为OFF。因此，若低水位检测器失效，即总是ON (X002 会变为OFF)，系统不允许泵工作，故在故障时是安全的。

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重起动泵。

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	主传送带门设置
	C005	进入再循环线的瓶子个数
	C006	进入主传送带的瓶子个数
	M020	允许瓶子进入再循环线
	M030	允许瓶子进入主传送带

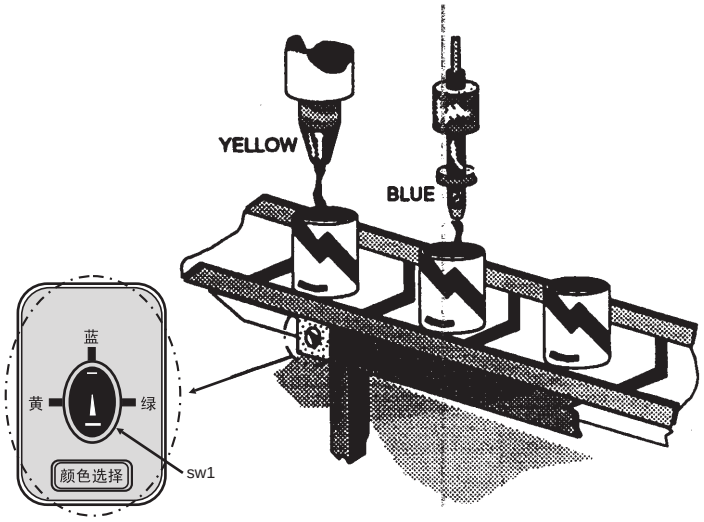
说明：

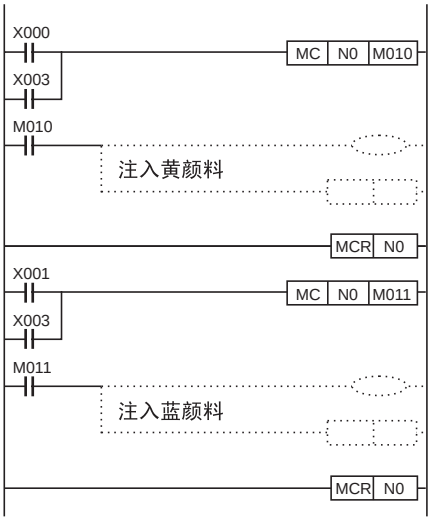
如果要使用再循环线路，则激活内部标志M020，就能使用再循环线。然而，如果再循环线满了(即C005 动作)或者正处于清空过程中(Y002 动作)，再循环线忽略M020 标志。当Y001 动作时，门GT1 反时针旋转，允许瓶子从主传送带转到再循环线上。当瓶子完成这次转移，光电管PC1 光线被遮断，这样给出一个输入X003，这个输入与计数器一起作用可以确定再循环线上的瓶子数目。当标志M020 不动作时，输出Y001 复位，门GT1 返回到中间位置(弹簧返回)。

瓶子离开再循环线时，标志M030 必须动作。在进入和清空的选择之间又有一个互锁，门GT1 被输出Y002 驱动，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。

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(N0)	黄颜料注入程序
	M011(N0)	蓝颜料注入程序

说明：

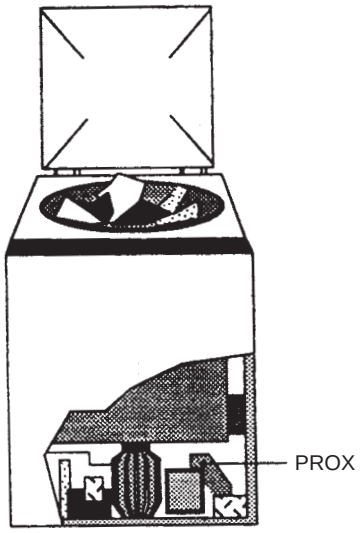
本例说明一段特定程序，必须要配合更大的主程序才能正常使用。颜料罐注入规定的颜料，黄或蓝色。如果两种颜料都使用，则产生绿色。

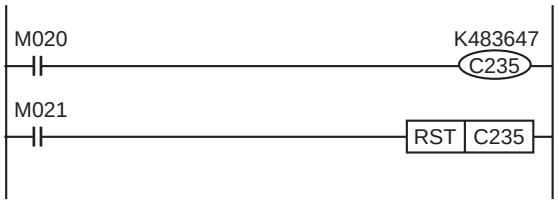
颜色的选定是通过旋转开关SW1 到适当的位置，输入X000、X001 和X003 会被接收到。因为把选定的颜料加到罐中的机器各不相同，每种选择要求对应一个特定程序。当颜色选定时，相应程序就“转换”到接通或断开。

当绿色被选中，两个程序顺序地运行，使用主控指令控制应该执行哪个程序。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”条件满足时，才能执行这些MC 和MCR 指令之间的程序。

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)

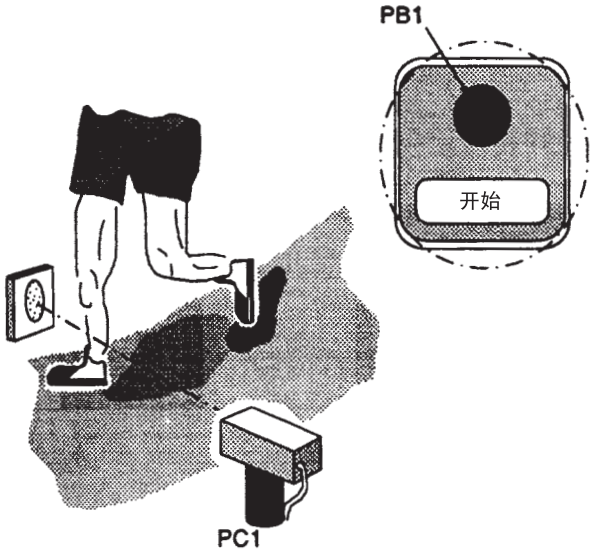
说明：

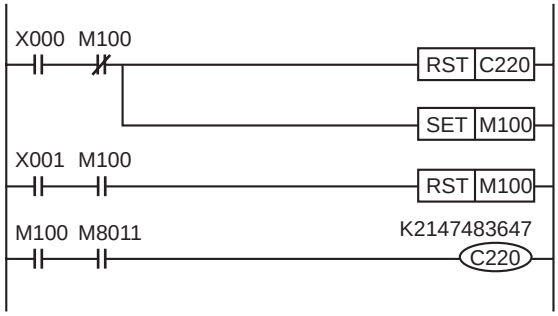
当洗衣机滚筒旋转时，固定在滚筒底部的一个小标记或指示器经过一个接近传感器(PROX)，这个传感器驱动输入X000，因为使用了高速计数器，输入X000 自动转为此计数器的输入。辅助继电器M020 用来控制计数操作，而辅助继电器M021 用来复位计数器。

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。

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	开始按钮
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，从而设定跑步标志 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 被接通，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。

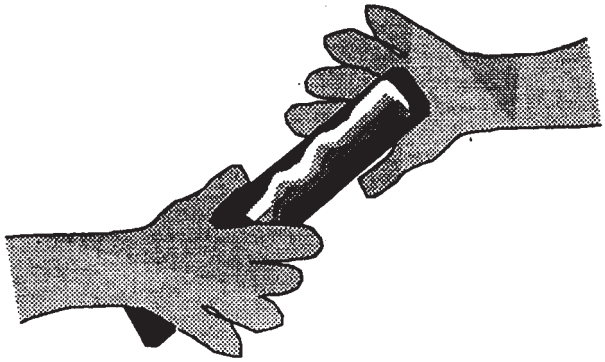
如果运动员都是跑得特别快的选手，那么就可以用一个 16 位计数器如 C0 或 C10 代替，如果这样，设定值应该是 32767。

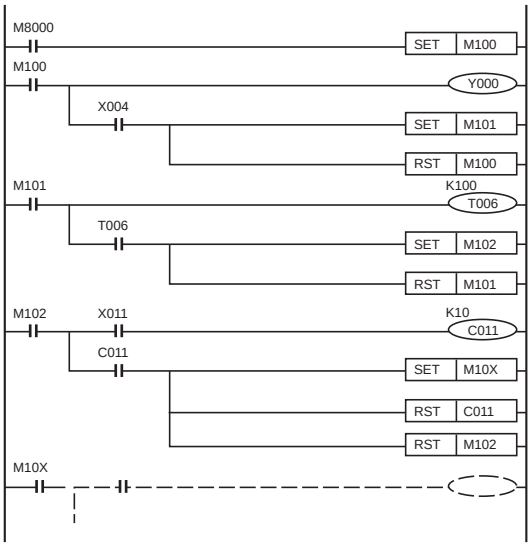
顺序控制

有关如何进行顺序控制的一些思想

2.1 接力棒式控制

“每个问题都有一万种解法”——编写有关可编程控制器的程序时，有多种方法都能达到预期结果。正确的方法是：程序设计者和系统装配工程师都能理解的……当然是能运行的！下面的方法是对于程序中顺序事件的另一种解决方法。





器件	PC 软元件	说明
	M8000	PC 运行常闭触点
	M100	用户程序“步骤1”
	M101	用户程序“步骤2”
	M102	用户程序“步骤3”
	M10X	用户程序“步骤X”

说明：

这种方法包括简单的“手传手”程序传递标志，程序控制如同接力跑般传递下去。

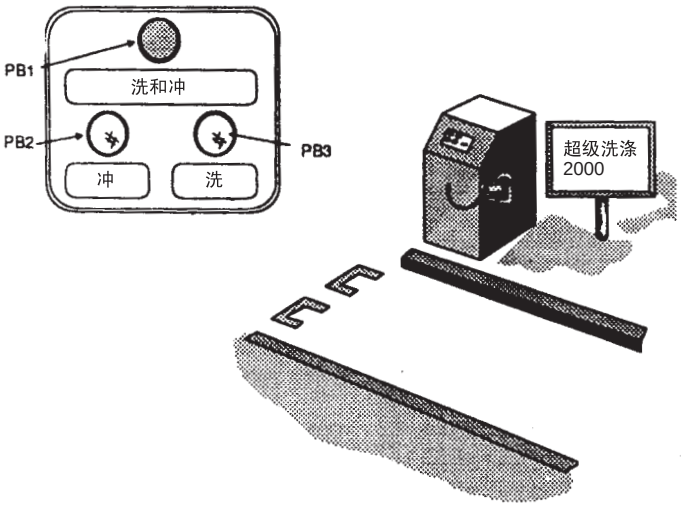
当每个“赛跑者”完成他们的规定路程，他们把“控制棒”传给一个等待着的“赛跑者”。这个程序的过程也正是这样，当每段程序结束时，使用一系列的 SET 和 RST 指令，控制被强制传递到程序的下一部分。

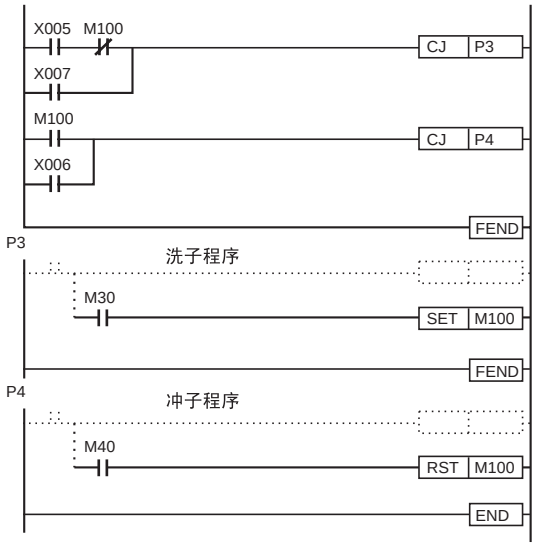
这种方法对控制步的数量没有限制，只要每部分程序有一个独立/唯一的“棒”标志。

M 和 S 线圈都能使用，甚至 Y 输出也能提供所期望的控制。

2.2 使用条件跳转

压力洗涤器有三个选项，“子程序”代替写3个分数的程序，缩短了1 / 2的程序长度。





器件	PC 软元件	说明
PB1	X005	选择冲和洗
PB2	X006	只选择冲
PB3	X007	只选择洗
CJ	FNC 00	CJ 应用指令
FEND	FNC 06	FEND 应用指令

说明：

洗车器有三种设置。其中两种提供单一服务，即冲或洗，第三种两项服务全部提供，即冲和洗。

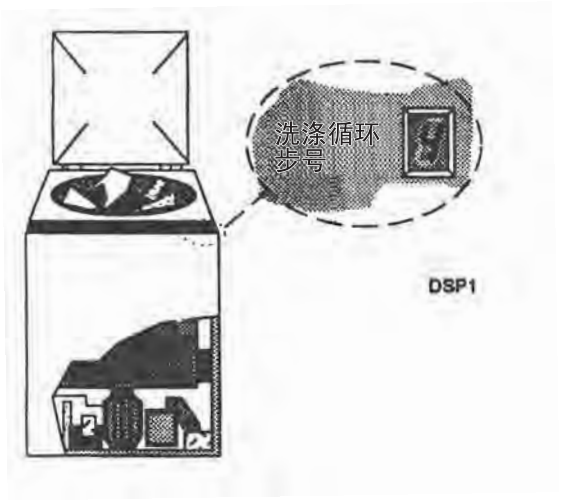
因为提供每项服务的程序是一样的，不管是单项的，还是整体服务的，所以使设计者可省去 1 / 2 的程序。这是因为不用再编 写整体处理的每项服务，设计者简单地调用或再使用已编写的程序，这通过指定单项服务为一个子程序来实现。

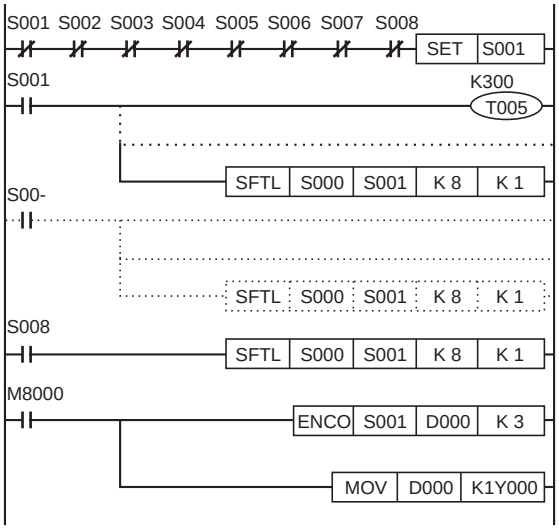
由此，当选中整体处理时，不是只有一个子程序运行，而是两个都运行。

例中，当选择冲时，X006接通，子程序 P4 运行。当要求洗时，输入 X007 接通，子程序 P3 运行。当选中整体处理时，两个程序都执行。为确保被调用的子程序只在前一个子程序结束时开始、标志 M030、M040 和 M100 用来“接通”要运行的程序。

2.3 一种可选的程序控制方法

程序编排有多种方法，下面方法的优点是编程工程只需最少的工作/控制。





器件	PC 软元件	说明
DSP1	Y000-Y003	有效程序步显示
	S000	虚构源(位)器件
	S001-S008	位栈(移位寄存器)
SFTL	FNC 35	SFTL 应用指令
ENCO	FNC 42	ENCO 应用指令
MOV	FNC 12	MOV 应用指令

说明：

主程序可以分为一些更小的程序，执行一系列任务。从程序/现场的维护角度来看，很难确切地知道一个程序是怎么工作的，尤其在维护工程师不是程序编制者时。如果使用结构化编程技术，例如本例方法或 SFC、STL 或任何其它已知的方法，维护就容易得多。

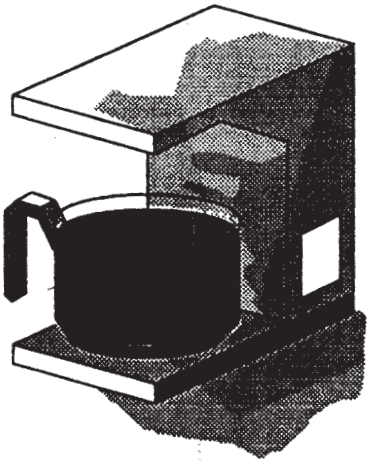
本例编程方法是控制程序序列的每一步。这通过在位栈移动控制位来实现(本例中位元件 S000 到 S008 作为栈)。当某一位动作时，相应的程序部分会工作。

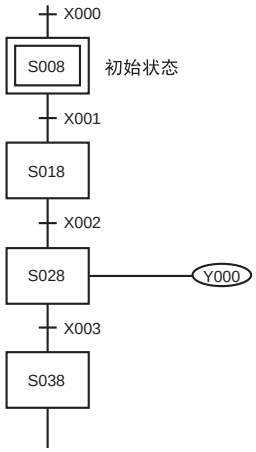
位栈里的每位依次有效时，它被编码并移出到一个 7 段显示(S001 到 S008 被编码为 D000，接着直接移至 Y000 到 Y003)。位 S000 用作正确操作移位指令的一个源位：S000 不能用于主程序的其它部分。

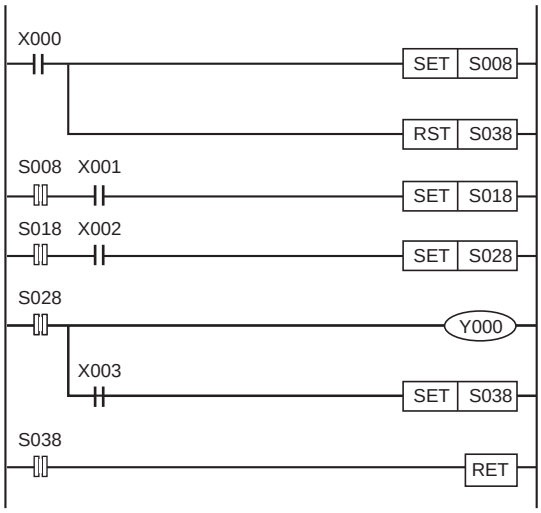
通过将有效步号显示在7段数码管，任何场合的维护工程师能够检测程序情况，即使他们没有编程经验。洗衣机实例表明：在这里，重要的是知道程序处于哪一步，以便能在适当时刻放入洗衣粉、柔软剂、染料或漂白剂。

2.4 使用步进梯形图编程

程序控制可用多种方法实现，最常用的一种方法是建立一个顺序功能图或更为人知的SFC程序。







器件	PC 软元件	说明
	X000	开始煮咖啡过程
	X001	加水
	X002	加咖啡
	X003	水煮沸
	Y000	运行加热器件
	S008	步骤1-有水吗
	S018	步骤2-有咖啡吗
	S028	步骤3-水热时关闭加热器
	S038	步骤4-结束加热过程

说明：

本例说明了一个煮咖啡的简单 SFC 程序，“流程图”是程序的一种 SFC 表示方法，图表说明 SFC 如何把一个程序分为独立步骤的。可以说，一个 SFC 程序是一系列较小的独立程序“串”起来的。

为进行这种编程，要用到 STL 指令。STL 指令的作用类似于流程图中的封闭方框。本例中程序被输入 X000 初始化，驱动第一步或初始步（例中为 S008）。此时，只有 SFC 程序被激活的步骤工作。假如输入 X003，什么也不会发生，因为步骤 S028尚未被激活。

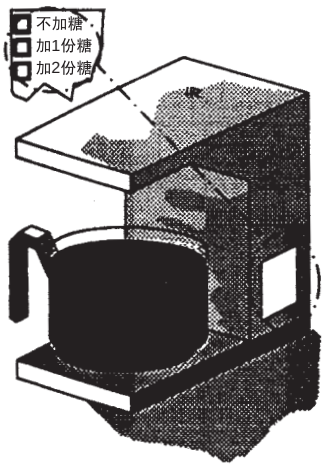
为使程序运行到 S018，必须接通输入 X001，使 STL 指令置位 S018 置位，并且自动复位步骤 S008，并继续完成 STL 程序的剩余部分。

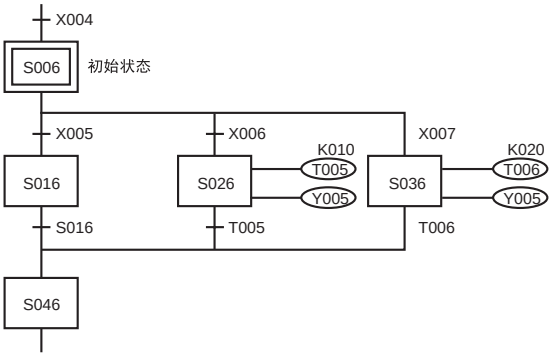
请注意 STL 步骤 S028，此步骤有效期间，Y000 会连续输出。例中输出驱动煮咖啡的加热器件，一旦水沸腾，X003 接通，程序转至 S038，同时 S028 和 Y000 断开。

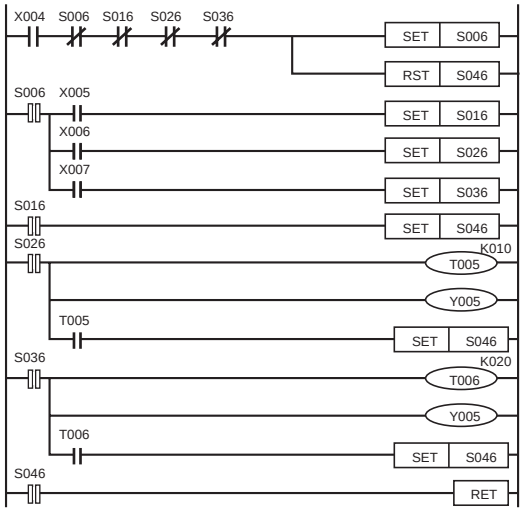
STL 程序能在标准编程部分之间被编程和放置。当它返回到标准编程时，最后的 STL 步骤（例中为 S038）必须包含 RET 或返回指令。

2.5 使用STL程序做多重选择

做一个选择时，有时还包括不同参数甚至不同程序的选择，用 SFC 程序很容易做到这一点，因为 SFC 的本质就是控制程序流程和隔离未被激活的程序段。







器件	PC 软元件	说明
不加糖	X005	按钮-要求不加糖
1份糖	X006	按钮-要求1份糖
2份糖	X007	按钮-要求2份糖
	S006	起动放糖过程
	S016	要求不加糖
	S026	要求一份糖
	S036	要求两份糖
	S046	结束放糖过程
	Y005	放糖

说明：

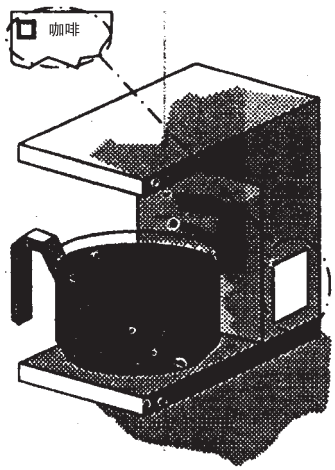
咖啡机能加3种不同份量的糖：不加，1份，2份。控制放糖的程序在这三种情况中略有不同，但是起始点和选择糖量后的结果相对这三个选择都是相同的。从 SFC 程序中可清楚看到这一点（见本页上侧的流程图）。由图中可知，一旦 SFC 程序被输入 X004 激活，初始状态 S006 将为 ON，此时，用户可有三种选择，按下其中一个选择按钮。

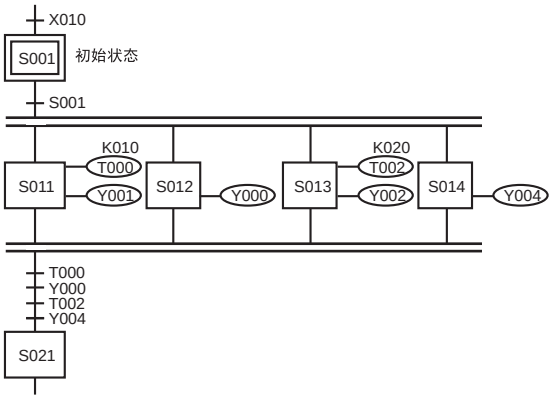
“不加糖”钮接通输入 X005，激活状态 S016，最后激活 S046。如果要求一份糖，则接通输入 X006，从而激活状态 S026，在 T005 限定的时间内输出 Y005 放糖，设定时间到后，程序强制转到状态 S046。最后，如果要求两份糖，则接通输入 X007，从而激活状态 S036，在 T006 定时器设定时间内放糖，同样地，定时完成时，激活状态 S046。

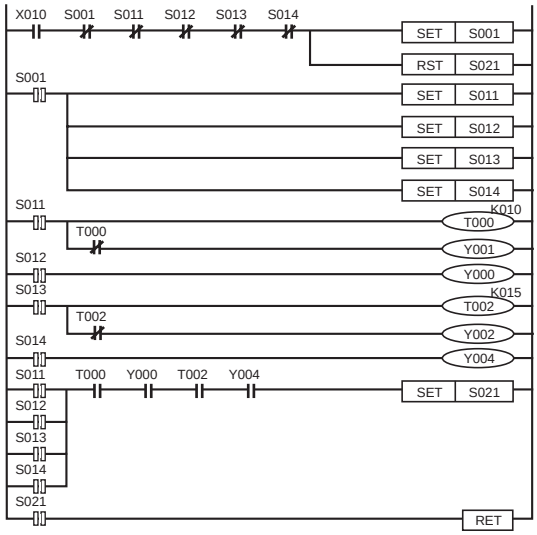
应该注意的是状态 S026 和 S036 都使用 Y005。在一个标准形式程序中，必须写成“OR”形式来驱动单个 Y005 输出。不过，SFC 类型的程序隔离了程序中所有的未激活部分，它允许使用双线圈输出。

2.6 带保持的多任务控制

常有这样的情况：在任何时刻，都有一个以上的任务要求完成。执行多任务时，程序控制需要很小心。SFC 类型的编程思想能快速地实现安全控制。







器件	PC 软元件	说明
咖啡	X010	按钮-咖啡的混合
	S001	咖啡混合过程
	S011	加热水
	S012	加糖
	S013	加牛奶
	S014	加咖啡
	S021	结束混合过程

说明：

这部分描述的程序说明了 4 个操作是如何同时开始的，及所有任务都完成后，程序是如何继续的。

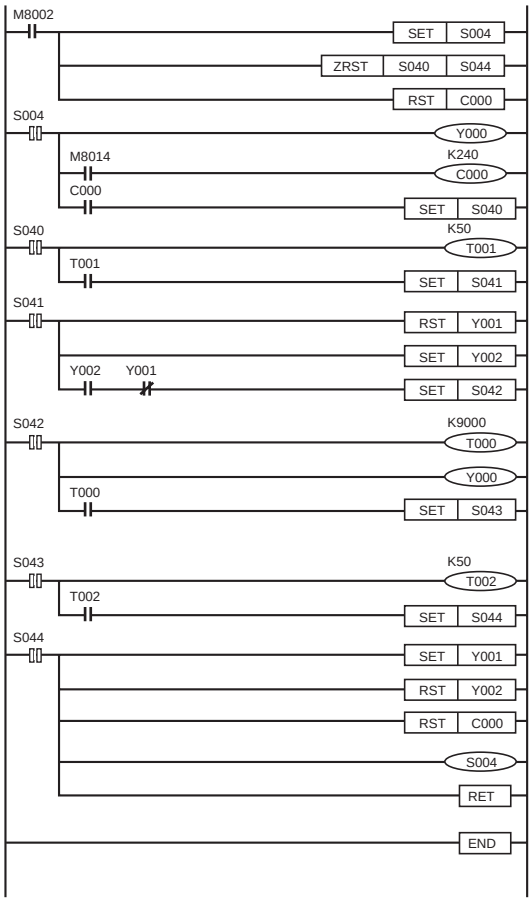
用来同时起动所有任务的方法叫做并行分支（图中表示为并行水平线）。STL 程序图显示，一旦唯一的条件 S001 满足，状态 S011、S012、S013和 S014 就都被激活。

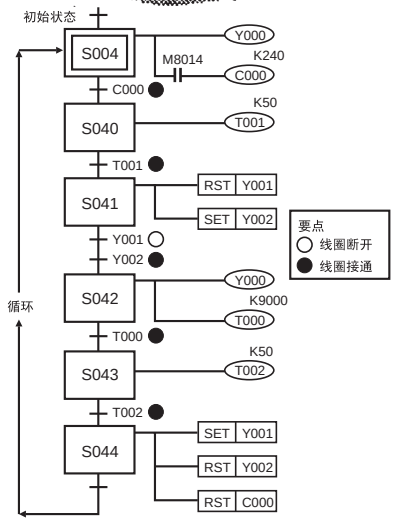
并行汇合是使程序“等待”汇合的所有条件都满足的一个方法。在 STL 程序图上，后跟 4 个条件（垂直串联）的并行线形成本例中的并行汇合。例中汇合只在定时器 T000 和 T002 定时结束，并且输出 Y000 和 Y004 接通时才发生。

程序的功能是：选中“咖啡”后，把泡咖啡所需的各种成分混合在一起，即热水、牛奶、咖啡和糖。

2.7 在某一时间做一个操作

使用 SFC/STL 类型的编程，很容易实现在指定间隔处理一些工作的循环。下面的例子描述了一个游泳池的过滤系统，每 4 小时，水阀换向以清理堵塞物。





器件	PC 软元件	说明
	M8014	1分钟脉冲
	Y000	起动泵
	Y001	正常运转阀门设置
	Y002	反向运转阀门设置
	C000	1分钟脉冲计数=4小时
	T000	泵反向时间=15分钟
	T001	泵电机设定时间
	T002	
ZRST	FNC 40	ZRST应用指令

说明：

程序按一个很简单的过程进行：4小时（S004）过滤池水，停止泵工作，使它转速慢下来（S040），改变水阀（S041）的运转方向，重起动水泵以清理堵塞物-只有 15 分钟（S042），再次暂停泵（S043），复位水阀（S044）。

最后的也是最重要的是 SFC/STL 程序再次返回步骤 S004。本例中创建了一个连续循环控制，即： 每四小时运行一次，使游泳池过滤器不堵塞。程序中这个重要步骤是由状态 S044 中的输出 S004 指令来实现。

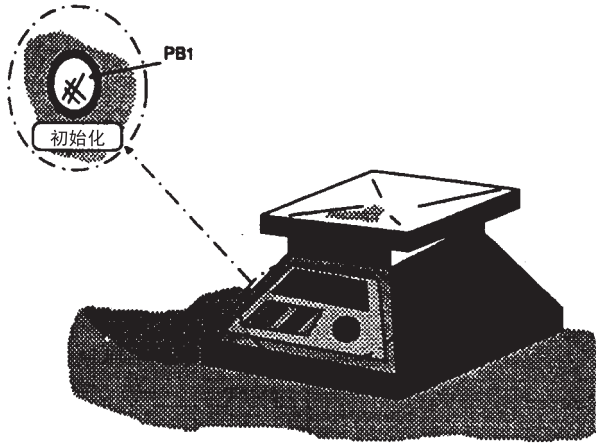
这些“跳转”可使程序跳向任何一个前面的状态，甚至跳向另一个 SFC/STL 事件链，在任何情况下，“跳转”被视为一条输出指令。

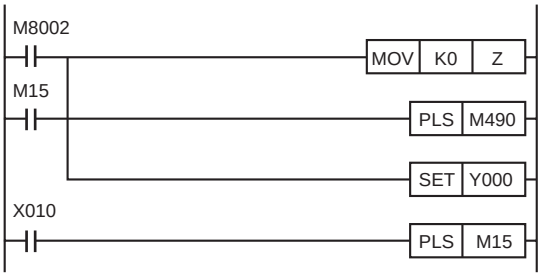
保持定时器，初始化控制和中断

编写自己的特殊定时器的一些建议，以及初始化和中断的使用。

3.1 通电时的自动初始化

在很多情况下，机器或设备要求某种或某一格式的初始化参数，以便它能进入基本准备状态。每次用手输入这些数据是很慢的，而且容易出错。特殊标志 M8002 可用来创建自动初始化程序。





器件	PC 软元件	说明
PB1	X010	手动初始化（按钮）
	M8002	PC 运行初始脉冲
	M15	手动初始化标志

说明：

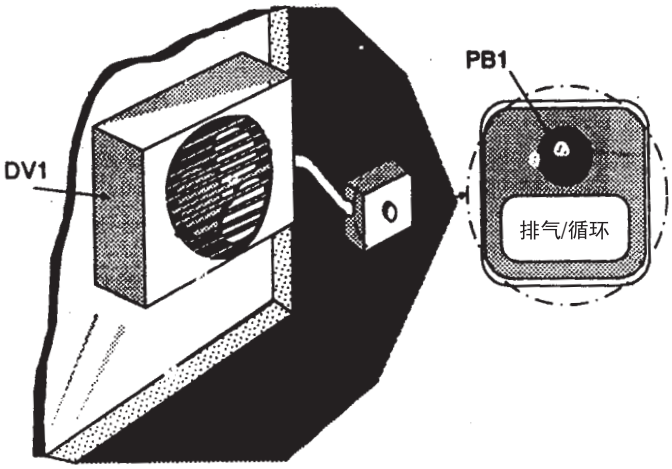
程序初始扫描期间特殊继电器 M8002 接通一次，把它记成在通电时产生的一个脉冲比较方便些。这样就意味着与 M8002 输入相联的任何指令在通电期间只执行一次，这用于一个初始化过程是理想的。这个过程可用来复位数据寄存器、设置元件接通或关断，同时也可复位计数器。

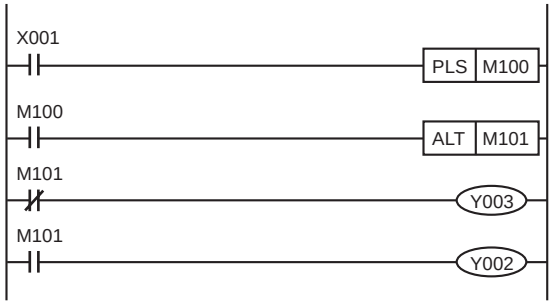
例子也显示一个与 M8002 并行安置的手动按钮（输入 X010 由按钮 PB1 接通），它允许用户在程序运行中的任何时刻强行初始化—结果类似于程序的复位，不过它不是由程序设计者决定。

特殊继电器 M8002的其他用法包括强制起动 SFC 程序、使可编程控制器初始化其它相连的外部设备、清除显示等。

3.2 使用 ALT 在两模式间进行选择

用一个按钮进行两种运行模式的选择。





器件	PC 软元件	说明
PB1	X001	排气/循环按钮
DV1	Y003	移动阀门到循环设置位
	Y002	移动阀门到排气设置位
ALT	FNC 66	ALT 应用指令

说明：

风扇有两种运行模式（循环和排气），可用一个开关进行切换。两模式间的切换由ALT 指令完成。当按钮 PB1（X001）接通时，辅助继电器 M100 接通一个程序扫描周期。

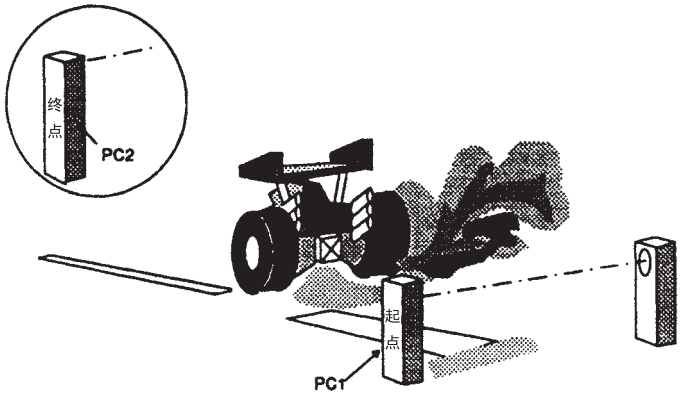
M100 常开触点接通时，起动 ALT 指令。

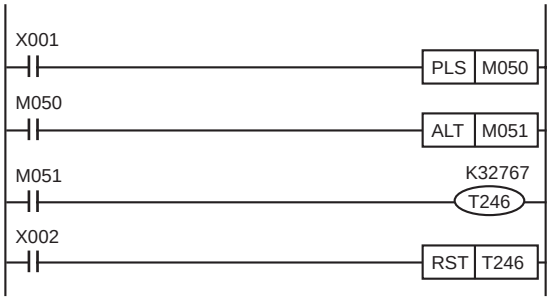
ALT 指令取反输出相关元件，本例中为 M101。第一次按 PB1 会使 M101 变为 ON，第二次按 PB1 则变为 OFF。

利用这种特性，ALT 指令相应继电器的常开和常闭触点用来在两“模式”间进行选择。本例中，当 M101 为 ON，风扇处于排气模式，Y002 起作用；当 M101 为 OFF，风扇处于循环模式，Y003 起作用。

3.3 使用 ALT 开始/结束一个动作

一个事件的计时可以由一个输入来控制，用这个输入改变定时器 ON 状态到 OFF 状态。





器件	PC 软元件	说明
PC1	X001	开始计时
PC2		停止计时
	X002	复位定时器
	M050	起动/停止控制
	M051	定时器运行标志
	T246	比赛定时器（1ms 保持定时器）
ALT	FNC66	ALT应用指令

说明：

为了得到一个所用时间的准确测量，要求某个信号触发开始/结束状态。这里给出的例子是一个极高速度的短程赛车比赛，时间需要用毫秒测量。如果定时器的开始/结束控制由肉眼来处理，当然会出错。因此，用光电管来检测赛车经过起点或终点的准确时刻。

光电管 PC1 和 PC2 都连接到输入 X001，这个输入使辅助继电器 M050 有效，接着 M050 使得应用指令 ALT（FNC66）改变标志 M051 的状态。

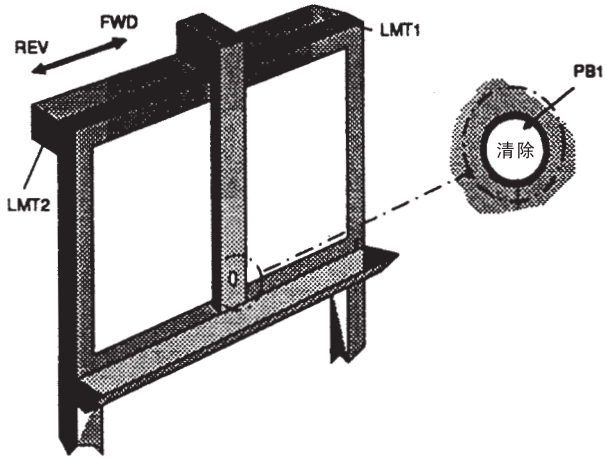
上述过程的最后结果是起动或停止定时器 T246。

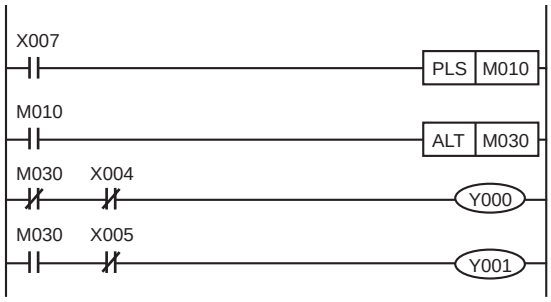
这个定时器是一个特殊定时器，它不仅有 1 毫秒测量精度，还具有保持功能。这意味着赛车冲过终点后，时间仍被保存以作参考。

请注意：时间是以毫秒给出，因此，需要除以 1000 得到时间秒数。定时器必须用输入 X002 来复位。

3.4 另外……

一些任务是往复性的，这就需要判断当前状态，这是一种典型的操作。要记住的是，什么构成一个模式？程序是怎样分配使得它满足两个要求？使用 ALT 指令能处理一种简单的这个/那个的情况。





器件	PC 软元件	说明
PB1	X007	“清除”按钮
LMT	X004	清除臂右极限
LMT	X005	清除臂左极限
FWD	Y000	清除臂右移
REV	Y001	清除臂左移
	M010	起动操作
	M030	方向控制标志
ALT	FNC66	ALT 应用指令

说明：

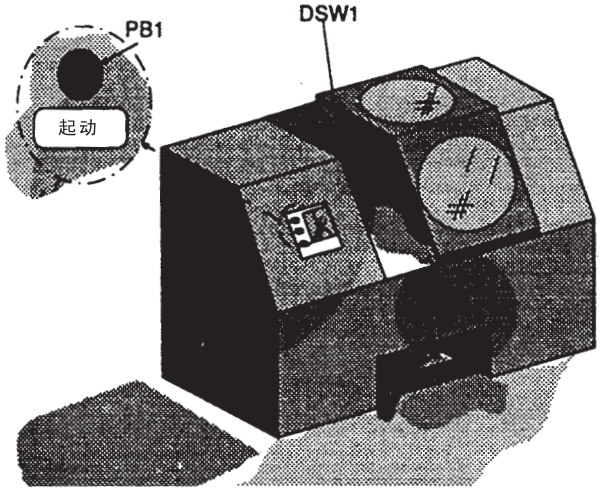
这种编程形式可以在很多情况中见到，不过使用时一般都略有不同。在一个场合中，一台机器可能被起动；在另一场合中，一个排气扇可能在循环与排气间转换。不同情况下，问题的初始表现并不能让人想起相同的解决方法。

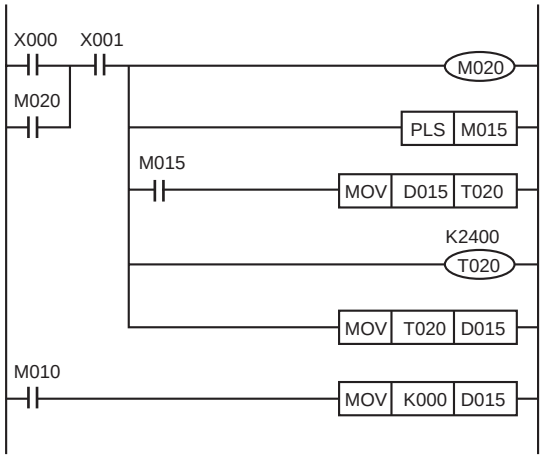
对于本节的例子黑板擦来说，也是如此。编程者的初始反映是它与起动一台机器或改变一个模式不一样。然而，如果忽略实际应用，只研究对象运行所要求的事件或过程，那么在这些不同的应用中能发现相似之处。

当然不能光看能否达到目的，因为实际问题可能会妨碍某些理想方法的实现。要记住的是，解决一个问题的方法不止一种，这个非常简短的擦黑板程序就是其中一种方法。

3.5 用户定义的保持定时器

这个定时过程允许中断，同时确保实际过程在整个时间段运行。





器件	PC 软元件	说明
PB1	X000	起动/重起动按钮
DSW1	X001	安全门开关
	M020	开始运行
	M015	存入定时器上次累计的时间
	M010	复位数据寄存器
	T020	定时器
	D015	保存时间值的数据寄存器

说明：

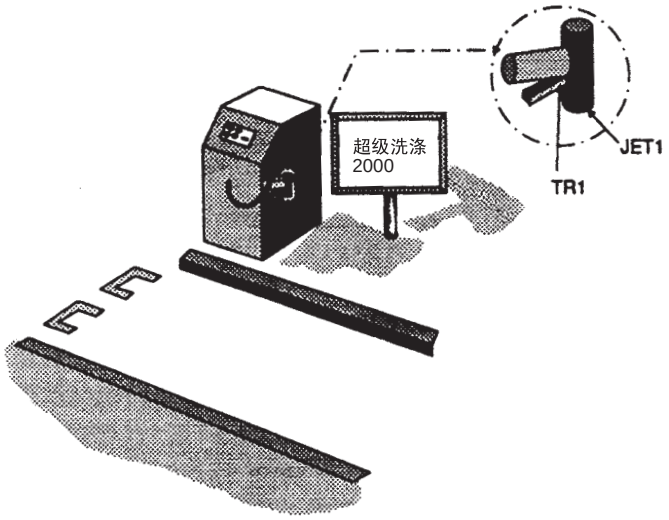
当滑动（安全）门关上，并按下起动按钮时，动作过程开始。当门关上时，开关 DSW1 动作，接通输入 X001。起动按钮 PB1 输入 X000 给可编程控制器。

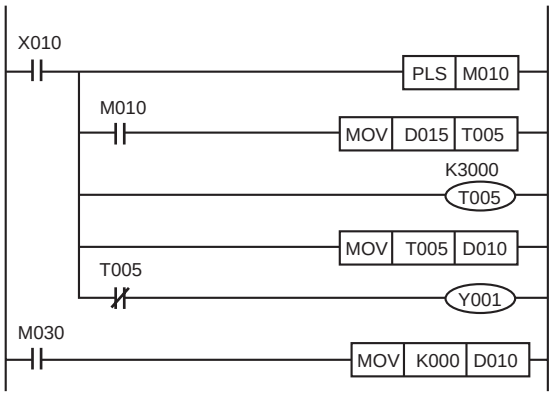
当两个输入都接通时，辅助继电器 M020 接通并自锁。定时器 T020 开始计时，同时当前时间数据送入寄存器 D015。在 T020 运行期间的任何时刻，如果滑门打开，锁定的 M020 继电器断开，定时器停止。

当前时间值不会丢失，再次起动时，即滑门关上并按下启动按钮，中断前存在 D015 中的时间值送回 T020 中，定时器 T020 接着从它断开的地方继续计时。定时器中断次数没有限制，不过，当定时器完成它的计时过程，数据寄存器 D015 需要“强制复位”为0。

3.6 可中断的定时事件

洗车器允许顾客使用喷水器 5 分钟。





器件	PC 软元件	说明
TR1	X010	触发喷水控制
JET1	Y001	喷水阀
	M030	复位数据寄存器
	M010	存入定时器上次累计时间
	T005	定时喷水使用
	D010	保存时间值的数据寄存器
MOV	FNC 12	MOV 应用指令

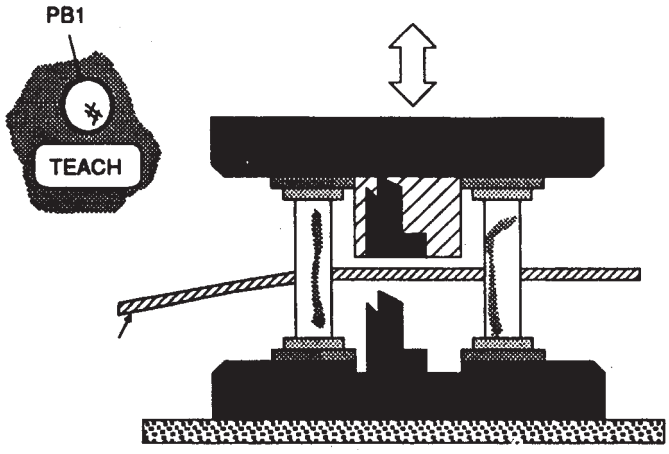
说明：

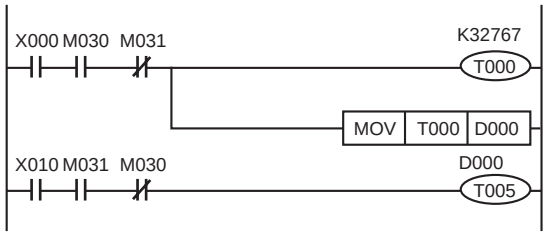
顾客把他们的车开到喷水器前，投入适当的钱后，辅助继电器 M030 被激活，从而复位数据寄存器 D010，它保存着定时器 T005 的时间值。

接下来顾客使用喷水器。当水管上的闸柄按下时，可编程控制器接收到一个信号（X010），这个信号使定时器 T005 开始工作。当定时器“运行”时，它的当前时间值送入数据寄存器 D010，此信号也允许水流出：输出 Y001 打开喷水阀 JET1。如果闸柄放开，定时器停止计时，当前数据被保存。当再次按下闸柄时，定时器从上次保存的时间值开始继续计时，这是因为保存在寄存器 D010 中的时间数据送会到计时器 T005 中，由此 T005 从停止的地方继续运行，这样能确保顾客得到准确的 5 分钟洗车时间。一旦 5 分钟时间到，水停止流出，直至下一位顾客投入正确钱数为止。

3.7 示教定时器

当要求自动设备做人工调整时，“示教”定时器是一种非常有用的工具。





器件	PC 软元件	说明
PB1	X000	示教按钮
	X010	自动模式中触发定时过程的输入
	M030	人工调整模式辅助继电器
	M031	自动运行辅助继电器
	T000	示教定时器
	D000	自动运行时间
	T005	自动模式定时器
MOV	FNC 12	MOV 应用指令

说明：

图示的自动冲压设备必须对每个新任务重新设定，新任务可能使用新的整形模具和不同的材料。

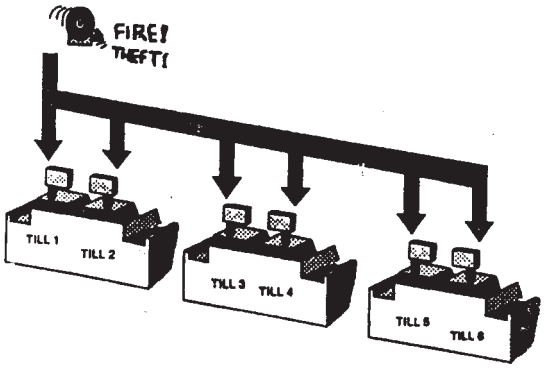
在这个特殊的例子中，冲压设备有一个调整设备的人工模式和生产过程自动运行的自动模式。

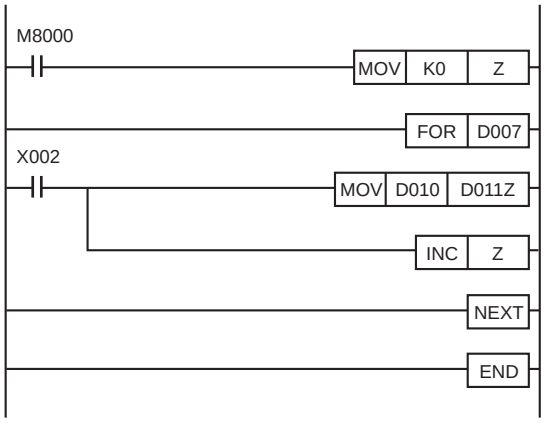
为确保被冲压/塑造的材料有足够的时间“流”成新的形状，控制压机的定时器必须改变。因此在人工模式中，装配技师按住示教钮，保持一段所要求的时间， 该数据自动传送到在自动操作模式中使用的定时器中。

可以这样实现：对按下示教钮的时间计时，把时间值移入一个数据寄存器，接着，那个数据寄存器作为自动模式定时器的设定时间使用。

3.8 将单一值装入一系列元件

每一位和每个数据寄存器的内容都有一个意思，这个意思依赖于用户的解释。有时，用户可能想移一个“意思”进入设备的“扩展范围”内如果这个范围若是可变的，则可能更有用。下面的例子提供了此问题的一种解决方法。





器件	PC 软元件	说明
	X002	移数据到目的地
	Z	变址寄存器
	D007	包含数据，它指定所涉及的收银台数目
	D010	传送到多个目的元件的源数据
	D011	目的寄存器的首地址
MOV	FNC 12	MOV 应用指令
FOR	FNC 8	FOR 应用指令
INC	FNC 24	INC 应用指令
NEXT	FNC 9	NEXT 应用指令

说明：

这里给出的例子描述了超市如何控制收银台，这对防火或防盗是有用的。单个可编程控制器（照看房屋安全）在发生抢劫或火灾情况下，能使一个或所有收银台锁上现金抽屉，或锁住涉及区域的收银台，从而保护里面的现金。

要控制的收银台数量由数据寄存器 D007 设定，D007 控制着要运行的 FOX-NEXT 循环次数。

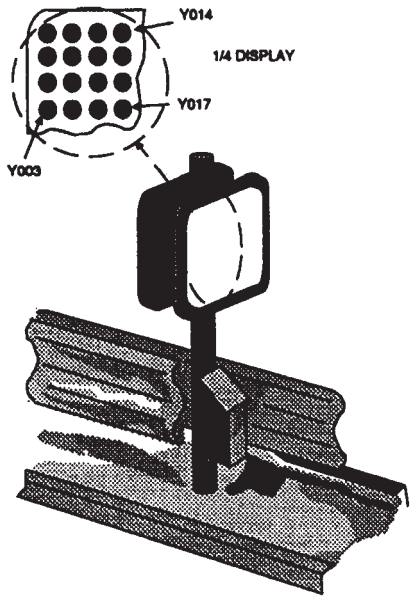
在源数据寄存器中确定的值被写入一个数据堆栈，堆栈首地址是 D011，其长度由 D007 确定。

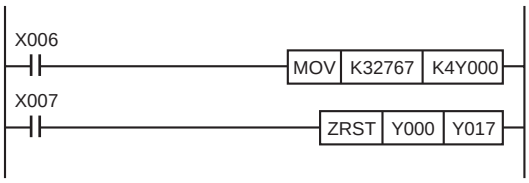
数据寄存器的内容能直接输出到收银台，或者经可编程控制器传回主计算机。

经微小修改，这个程序能直接用于位元件、定时器、计数器等。所有情况下的规则都一样—把一段数据写入很多元件中。

3.9 复位全部设备

程序步是非常宝贵的，为了节省编制一大堆单个复位指令的时间和精力，也为了减少所用的程序步数目，可以使用 ZRST 指令。





器件	PC 软元件	说明
	X006	置位输出 Y000-Y017
	X007	复位输出 Y000-Y017
	Y000-Y017	用以控制交通控制信号的输出
MOV	FNC 12	MOV 应用指令
ZRST	FNC 40	ZRST 应用指令

说明：

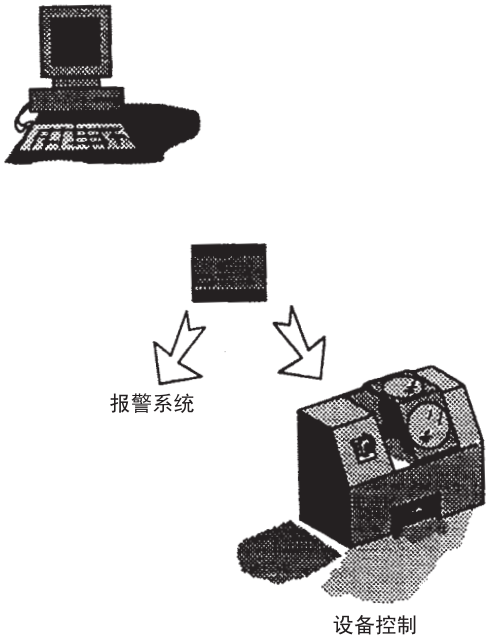
本节给出的例子是一个道路/交通信号，此信号有一段显示区需复位为 OFF，每一个输出驱动一个指示灯。当输入 X006 接通时，数值 32767 移入输出 Y000 到 Y017 ，这个操作结果是使选定区域的所有输出变为 ON。

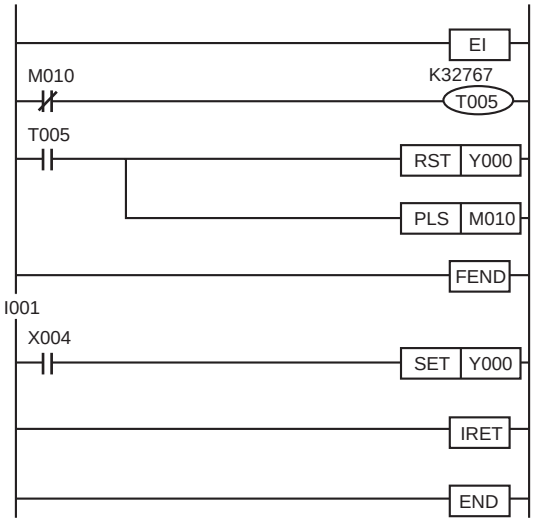
为了复位它们为 OFF，输入 X007 必须接通，它驱动应用指令 ZRST，后者执行复位操作。

本例中可看出，与分别复位所有输出的程序相比，ZRST 指令节省了 11 个程序步。

3.10 即时的系统报警

一个可编程控制器能完成多个不同任务：从操作一台机器到报告整个工厂的状况和形式。有一个关键点有时会被忽视，但它在任何一种情况中都会用到，那就是中断当前任务，执行一项特殊功能的能力。





器件	PC 软元件	说明
	X000	激活中断的输入
EI	FNC 4	EI 应用指令
FEND	FNC 6	FEND 应用指令
	I001	中断指针：X000 从 OFF 到 ON
IRET	FNC 03	IRET 应用指令

说明：

PC 上的中断系统能用于执行多种不同控制，它的目的是中断一个当前正在执行的任务并暂时用一个更即时的操作来代替，所有中断都以高速运行。

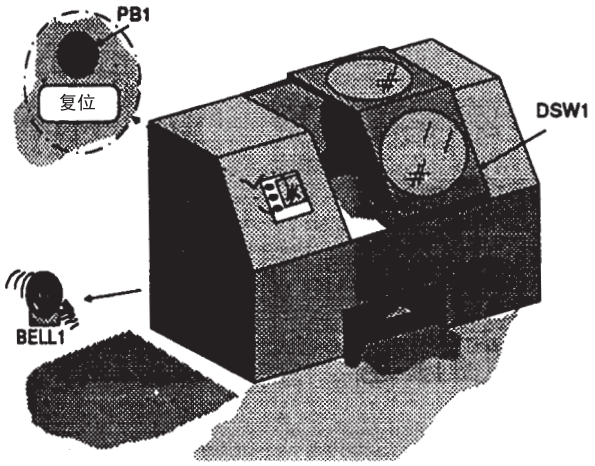
中断可以用于报警系统，标准编程技术考虑了大多数情况。举例来说，工厂管理计算机的电力供应出现尖峰或不足时，必须立即了解到这类情况，这时第二个电源马上切换进来。如果使用不间断电源（UPS），这类警告会使用户存储那些正在使用的计算机文件。起始的供电尖峰或不足可能会在极短时间内发生，这就需要用某种形式的高速“执行”来控制，由此要使用中断元件。

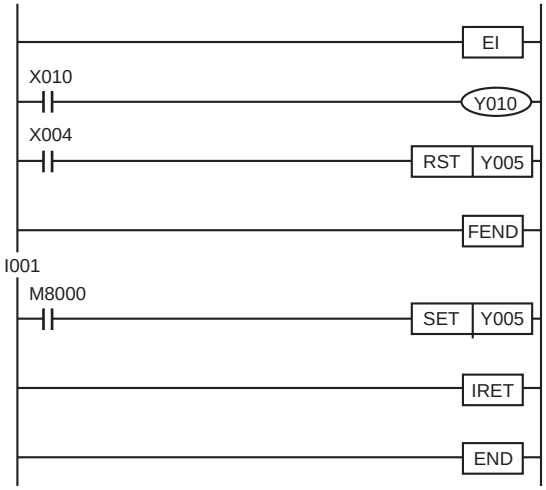
本例中，如果输入 X000 在任意时刻接通，指针 I001 和 IRET 指令间的程序会被执行，与任何别的程序活动无关。

请注意：这是个特殊操作，不是所有的元件都能来起动/调用中断程序。

3.11 安全报警系统

紧急系统要求即时的响应。中断编程技术可使某事件在一个程序扫描周期内立即被激活，与它实际在程序中的位置无关。





器件	PC 软元件	说明
PB1	X004	复位按钮
BELL1	Y005	警铃输出
DSW1	X001	门开关
	M8000	PC 运行常闭触点

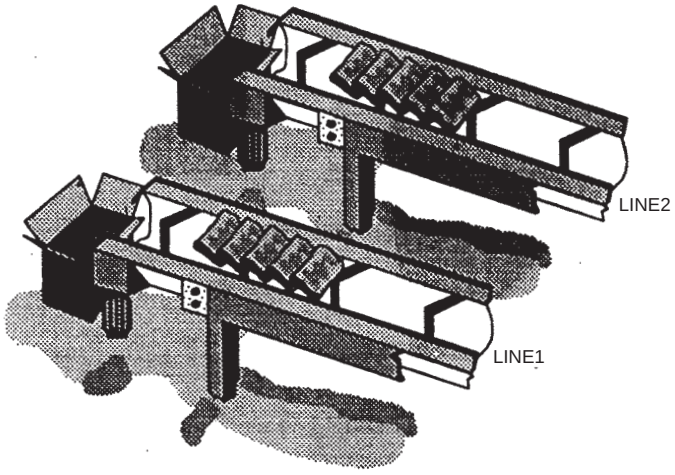
说明：

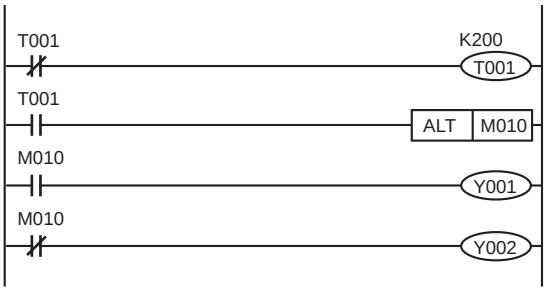
本节给出的这个非常简单的紧急系统说明：在任何时刻，如果安全门被打开，一个专用的中断输入（X001）触发中断程序I101。当输入X001 从OFF 变为ON 时，指针I101 同时作为程序标记和所使用的中断的标记。中断处理完成后，程序控制返回到“紧急情况”前执行的最后一步。这样处理是很有用的，它帮助编程者保持程序的编程顺序。主要的问题是某个特殊功能被激活或者一个子程序运行，并且当控制返回主程序时，操作顺序与机器操作不匹配。使用任何形式的“程序中断”时都应小心。

中断运行完毕，输出Y005 置为ON。本例中，铃声响起，由此作为声音警报。本程序中通过接通输入X004 使警报复位。

3.12 “自关断”定时器

通过切断与驱动程序的联系而自复位的定时器，常被叫做“自关断”定时器，它们是编程者“工具箱”里一个很有用的工具。下面的例子不是一个完整的实际应用，而是用来说明“自关断”定时器操作的部分程序。





器件	PC 软元件	说明
LINE1	Y001	装载机械的传送带
LINE2	Y002	
	T001	自置位/复位的定时器（“自关断”定时器）
	M010	状态改变的标志
ALT	FNC	ALT 应用指令

说明：

定时器T001 连续运行，定时器线圈由它自己的常闭触点驱动。当定时器完成定时过程，线圈被接通，使定时器常闭触点断开，通路被断开，由此线圈不通电，这个新状态也意味着常闭触点不通电。因此，最后情况是定时器复位并且自动地再次开始它的定时过程。这是一个很快的响应，定时器的复位/置位会在程序的大约一次扫描（最多两次扫描）内发生。在如此短的时间内，定时器的连续置位和复位使定时器触点动作如同受脉冲控制使用定时器T001 的常开触点驱动ALT 指令说明了这一点，每过20 秒，Y001 和Y002 的输出状态互换。

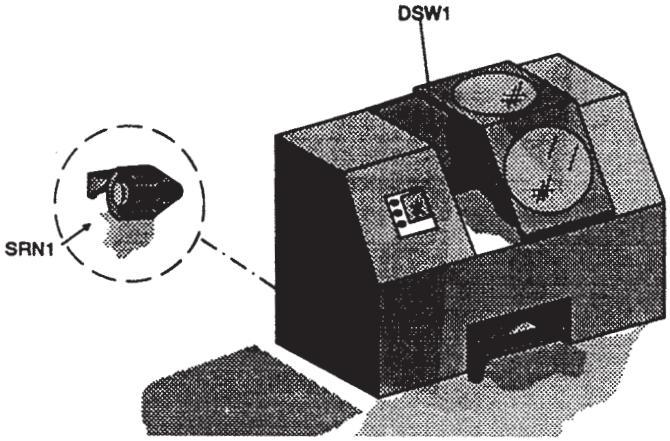
在这个例子中，变化着的输出对输送杂志的线路进行切换，20 秒的停顿用于杂志沿传送带下移并装入包装箱中。这样能保证一个稳定的生产流程，这个过程很容易由照看杂志装箱的一个操作人员管理。

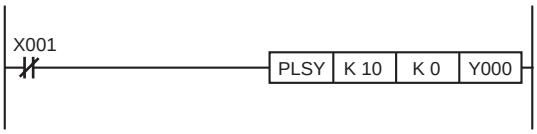
声音、速度、配给和光照亮度

所有这些不同应用可用几个非常简单的指令来控制。

4.1 脉冲输出产生声音

安全门打开时，警报起作用。





器件	PC 软元件	说明
DSW1	X001	安全门开关
SRN1	Y000	警报器输出
	K10	输出频率（脉冲/秒）
	K0	输出脉冲数（0=持续脉冲）
PLSY	FNC 57	PLSY 应用指令

说明：

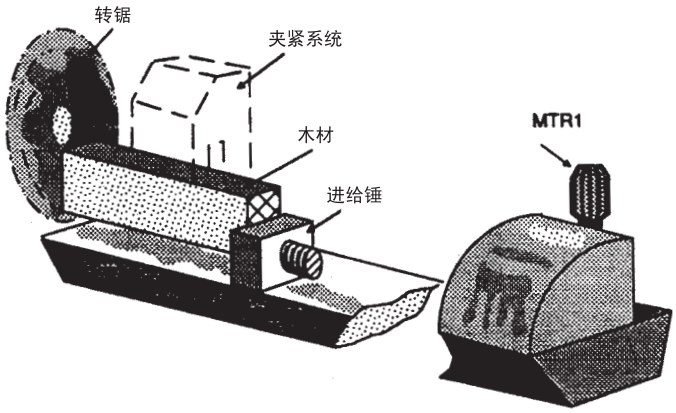
安全门关闭时接通输入X001。如果门打开，则输入断开，警报器开始工作。

使用PLSY 指令控制对警报器的输出，并能控制警报器的音高或“音量”。一系列高频脉冲会发出高而连续的声音，如果使用一个低频输出，则会听到一个较低的、断续的声音。一旦确定相应频率，就能得到警报器声音输出，声音随频率升高降落。收到来自Y000 的输出时，警报器声音的音量和音高都将增大，当来自Y000 的信号不再出现时，警报器的音量和音高会下降，随之声音自然减弱并且/或者中断。

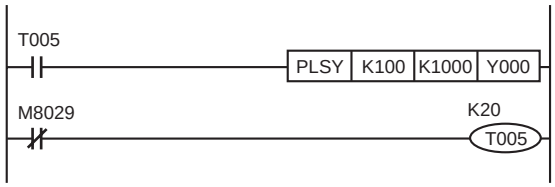
警报器动作过程可以通过限定PLSY 指令发出的脉冲数来控制。本例中，通过设置K0 可得到连续脉冲输出。

4.2 使用脉冲移动一段线性距离

脉冲输出控制木材被切割前移动的距离。



器件	PC 软元件	说明
MTR1	Y000	脉冲输出（到电机）
	T005	切割木材的时间延迟
	M8029	PLSY 输出完成
	K100	脉冲输出频率（脉冲/秒）
	K1000	脉冲输出总数=1000
PLSY	FNC 57	PLSY 应用指令



说明：

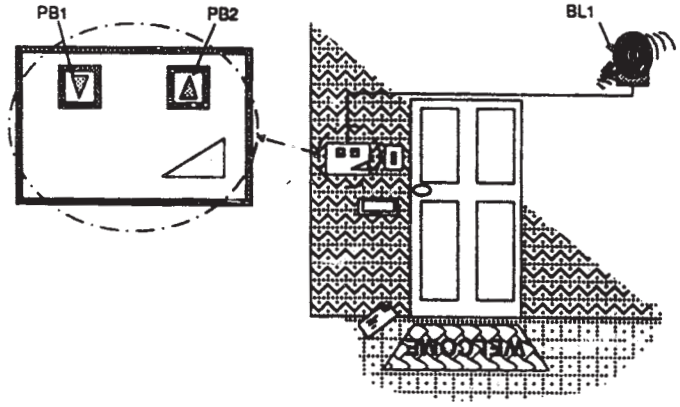
一段木材沿锯床一次移动指定的距离。完成每个移向电锯的移动后，经过一小段时间延迟（T005），木材被切割至规定尺寸。在锯床上的移动由PLSY 指令控制，一系列脉冲送入一台电机，电机的转动通过履带和齿轮传送到螺杆，螺杆旋转并推着活塞直线前进。给电机的脉冲数与最终的线性移动之间有密切联系。了解了这一点，活塞的移位控制就能很容易地实现。

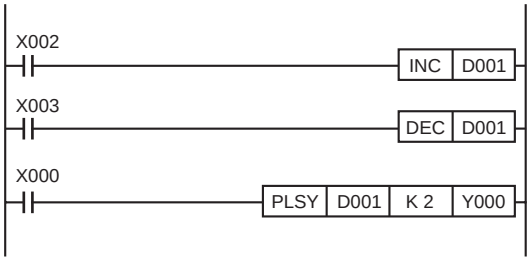
本例中，M8029 的常闭触点用来驱动定时器T005。PLSY 指令在其工作期间需要连续地被驱动。

当标志M8029 出现，PLSY 操作就结束。达到指定的输出脉冲数时，M8029 被接通。M8029 只在PLSY 指令完成后的一个扫描周期内保持接通状态。

4.3 PLSY 指令的可变控制

可以通过升高或降低驱动门铃的输出脉冲频率，以改变门铃的音高。





器件	PC 软元件	说明
PB2	X002	增大门铃声音
PB1	X003	减小门铃声音
	X000	门铃按钮
	D001	当前声音设定
BL1	Y000	门铃
INC	FNC 24	INC 应用指令
DEC	FNC 25	DEC 应用指令
PLSY	FNC 57	PLSY 应用指令

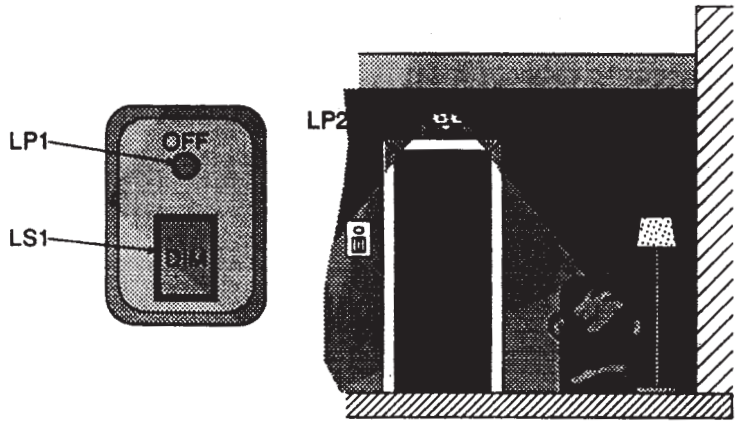
说明：

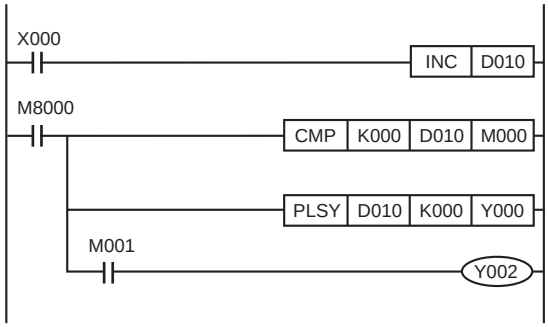
在“铃音”面板上的两个按钮可使驱动门铃的脉冲输出的数目增加或减少，从而改变门铃 BL1“音乐”的音高或音调。输出脉冲频率能在-32768 到 327687 之间变化。当按下按钮时，频率就相应地改变，变化量与按下的时间和程序扫描时间成正比。

脉冲频率的这种变化通过增大或减小数据寄存器的值来完成，本例中的寄存器为 D001。实现方法是：连接输入 X002（PB2）到 INC 指令，输入 X003（PB1）到 DEC 指令，程序每次扫描时，这些指令使数据寄存器的内容加 1或减 1。

4.4 照明控制

改变送给灯的脉冲频率，就能控制灯的亮度。





器件	PC 软元件	说明
LP1	Y002	主灯关指示
LP2	Y000	主灯
LS1	X000	亮度设置按钮
	D010	当前亮度
INC	FNC 24	INC 应用指令
CMP	FNC 10	CMP 应用指令
PLSY	FNC 57	PLSY 应用指令
	M8000	PC 运行常闭触点

说明：

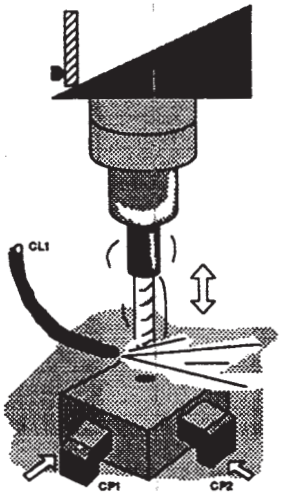
主灯 LP2 的亮度由开关 LS1 控制。工作时，开关使数据寄存器 D010 的数据值增大。

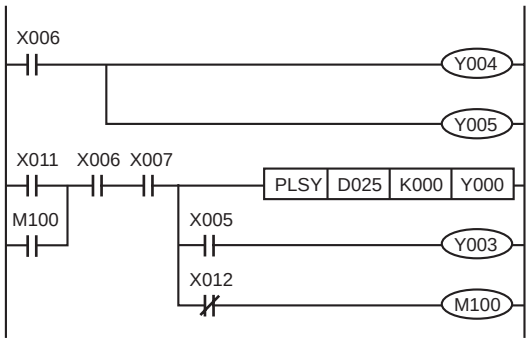
这个变化的数据寄存器用来作为 PLSY 指令中输出频率的源数据。数据寄存器 D010 能在 -32768 到 32767 的范围内递增： 当到达 32767 时，寄存器增值得到 - 32768，所以，整个过程是一个连续循环。

当数据寄存器 D010 等于“0”时，比较状态使 M1 接通，接着输出 Y002 起作用，这个输出与指示灯 LP1 相连，用来指示主灯 LP2 关闭，即没有脉冲输出给灯。

4.5 启动/停止

所有紧急情况电路的配线和操作一般应独立于可编程控制器电路。不过，可编程控制器可以方便地对启/停进行控制。





器件	PC 软元件	说明
CP1	Y004	夹具
CP2	Y005	
CL1	Y003	驱动冷却泵
	Y000	驱动主轴电机
	X005	冷却泵启动
	X006	夹具夹紧
	X007	安全防护
	X011	启动按钮
	X012	停止按钮
	D025	钻头速度
PLSY	FNC 57	PLSY 应用指令

说明：

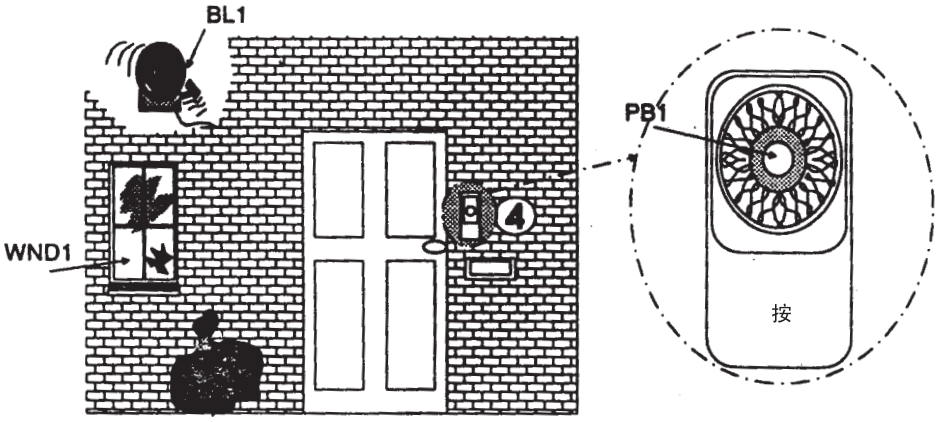
可编程控制器可以控制启/停时序。这意味着，附加的软件安全控制对硬件安全线路是一个补充。本节的例子中，夹具系统和用户防护装置可直接被检测。如果它们在运行过程中发生差错或失效，则主轴会自动停止。

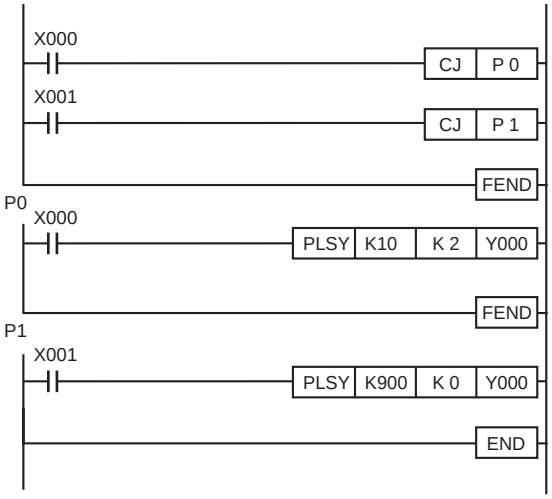
当所有条件都满足时，按“启动按钮”，X011 接通，这个“启动按钮”只提供一个瞬时输入。一收到输入信号，“锁定标志”M100就起作用，使钻头处于连续工作状态。而一收到从“停止按钮”来的输入，M100建立的“锁定”回路被切断，从而使得钻头停止工作。

本例中，钻头转速由 PLSY 输出间接控制，PLSY 通过输出 Y000 驱动主轴电机，电机驱动信号的频率由数据寄存器 D025确定。如果要直接改变数据必须注意要确保电机速率调整不是瞬时完成的—这种做法对机器和使用者是极其危险的。

4.6 使用多重 PLSY 指令

其功能是为不同场合选择相应的铃声。





器件	PC 软元件	说明
PB1	X000	门铃按钮
WND1	X001	窗检测器
BL1	Y000	门/警报铃
PLSY	FNC 57	PLSY 应用指令
CJ	FNC 00	CJ 应用指令
FEND	FNC 06	FEND 应用指令
	P0	门铃子程序
	P1	警报铃子程序

说明：

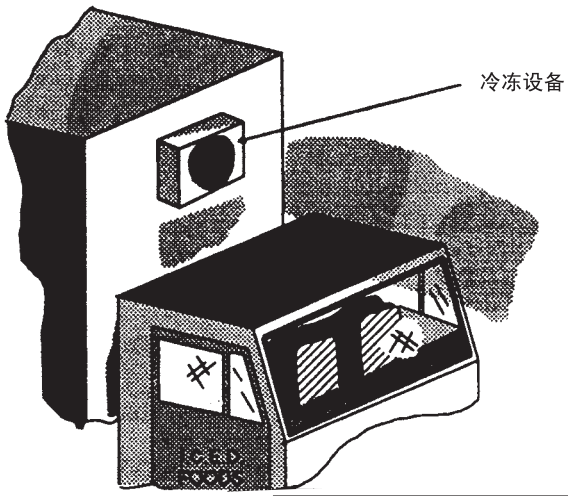
房屋主任有一个具有两项功能的响铃，一项是当作门铃使用，另一项是当作警铃。在两种情况下用同一个输出驱动响铃，使用 PLSY 指令改变铃响的次数和频率，从而控制每种操作要求的不同声音。一个高音量连续的铃声用作警报，而短促两鸣的铃声用来报告有客来访。

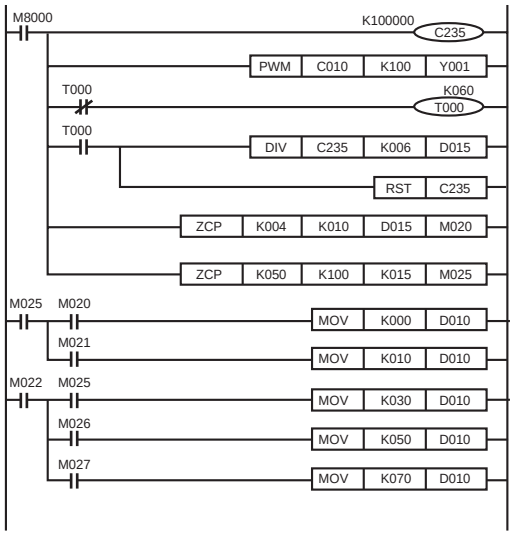
一个主程序中不允许使用两条 PLSY 指令，但可把每种情况插入到由条件跳转指令调用的子程序中，当两种情况中的任一条件满足时，调用相应的子程序。在本例中，使用门铃时，程序跳转至 P0 处，即表示指针 P0 的程序被激活；当要求使用警铃时，P1 被调用，因此跳转到程序指针 P1 处。

一旦条件跳转被激活，只有当遇到一个 FEND 或 END 指令时，它才被复位。

4.7 正反馈回路

系统有“静态的”和“动态的”两种。“静态”系统只是按指定顺序处理程序步，每年每月每小时做同样的事；“动态”系统也这么做，不过它使用输入数据不断地修正自己的操作。





器件	PC 软元件	说明
	X000	计数器 C235 的高速输入
	Y001	用 PWM 的高速输出
	C235	高速计数器-32 位 U/D
	T000	定时器，用以计算高速输入 X000 的频率
	M020-22 M025-27	范围标志
	D010	产生脉冲输出频率的源数据
	D015	计算的输出频率
PWM	FNC 58	PWM 应用指令
DIV	FNC 23	DIV 应用指令
ZCP	FNC 11	ZCP 应用指令
MOV	FNC 12	MOV 应用指令

说明：

所有程序都从同一地点（指令）开始，运行和建立更新更复杂的控制系统取决于编程者。这里将讲述编写一个好程序的方法，运行中程序是动态的还是静态的，完全取决于编程者。

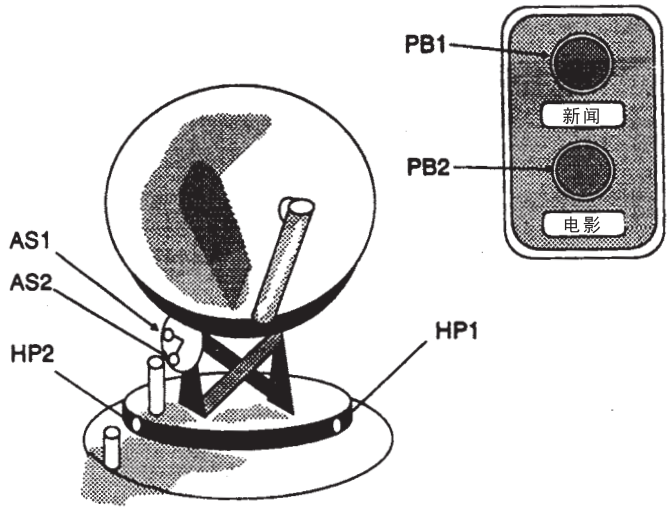
本节给出的例子说明一个制冷设备（可能安装在一辆卡车上）的温度可以用反馈回路的办法来控制。冷冻设备的持续运行是容易的，但最好对设备温度变化加以控制，如门打开了，冷藏箱内温度上升，制冷设备就要加大负荷，这可以通过改变给驱动制冷设备的压缩机的脉冲序列来实现。

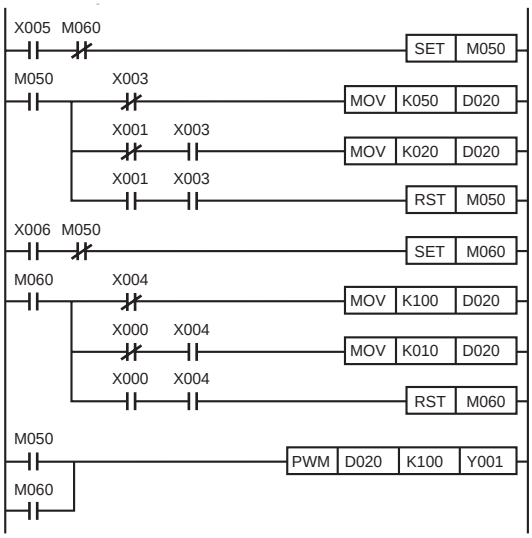
所以，最终系统用增加或降低冷冻设备工作负载的办法来动态地调节温度。

输入X000 用以读取来自温度检测元件的数据，而输出Y001 用来驱动冷冻压缩机。基本温度的校准，即什么时候加多少负载到压缩机，是由ZCP 来处理的，它“选择”适当数值送入数据寄存器D010。

4.8 多速度定位

下面程序阐述了一种多速度/定位技术。





器件	PC 软元件	说明
PB1	X005	卫星天线移向新闻卫星
PB2	X006	卫星天线移向电影卫星
HP1	X001	新闻卫星在卫星天线水平位置
HP2	X000	电影卫星在卫星天线水平位置
AS1	X003	新闻卫星在卫星天线垂直平面位置
AS2	X004	电影卫星在卫星天线垂直平面位置
	D020	输出信号长度
	Y001	驱动卫星天线定位电机的输出信号
MOV	FNC12	MOV 应用指令
PWM	FNC58	PWM 应用指令

说明：

该系统包括一个由两个伺服电机定位的卫星天线。系统有硬件和软件互锁，使得每个电机能独立地工作，软件互锁在程序别的地方编制。

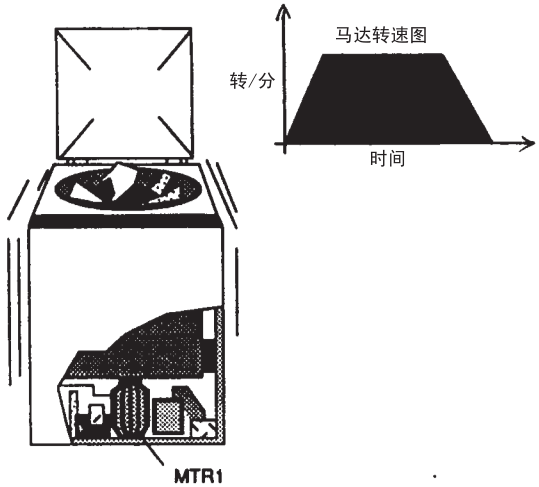
选择所要求的按钮PB1 或PB2，程序开始运行，此选择设定了“运行”顺序。

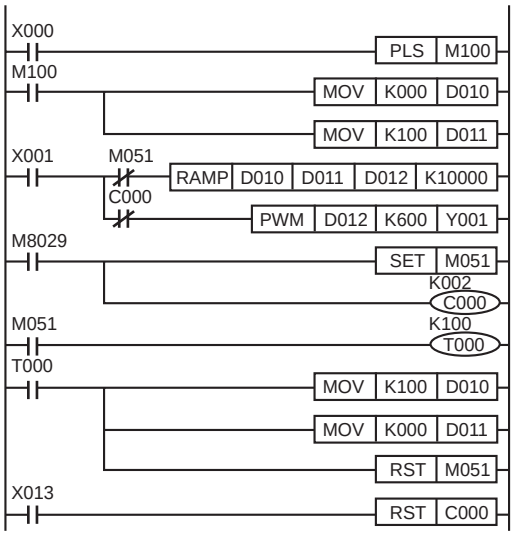
卫星天线先在一个平面内移动，直到位置传感器捕捉到目标。此时，第二平面运动开始，当卫星天线到达它的最终位置，系统复位。

因为每个平面运动都被独立地控制，可以用不同的速度，本节的例子程序使用4 种不同速度。按照要求，这些速度被送入数据寄存器D020。应当特别注意当K100 值送入数据寄存器D020 作为速度设定时，PWM 指令将提供一个连续的输出信号。因为输出信号时间段（D020）等于信号频率。

4.9 电机的软起动/软停止

电机控制常是用户关心的问题。下面例子提出一种软起动/停止的方法。





器件	PC 软元件	说明
MTR1	Y001	洗衣机电机的转动输出
	D010	RAMP 限值 1
	D011	RAMP 限值 2
	D012	实际跃升寄存器
	T000	定时常速过程
	C000	记录已做的 RAMP 指令次数
	M8029	RAMP 结束识别的特殊继电器
MOV	FNC 12	MOV 应用指令
RAMP	FNC 67	RAMP 应用指令
PWM	FNC 58	PWM应用指令

说明：

本节例子说明了一台洗衣机的电机速度是如何“提升”至额定转速，在所要求时间里保持这一速度，接着再返回至停止状态的。

程序通过组合两个应用指令来实现：RAMP 控制当前数据元件在规定扫描次数期间在两预定值之间线性变化，PWM指令使用“跃升”寄存器改变输出信号Y001 的频率。

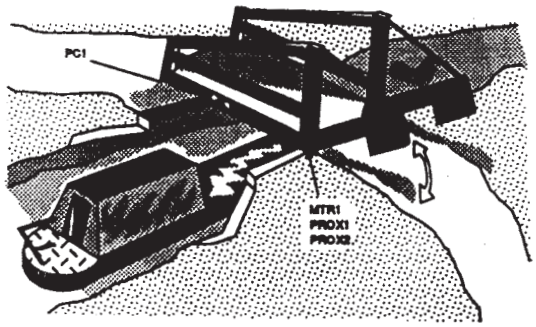
这两个指令都使用同一个特殊继电器来识别当前操作的结束与否。这个特殊继电器M8029 用以触发“跃升”过程结束与常速运行开始之间的转化。同样地，“下降”模式和停止模式间的转化也用M8029 来触发。在这两种情况中，用RAMP指令的结束触发M8029 标志。本例中C000 对M8029 计数，C000 两次被激活时，程序就停止。

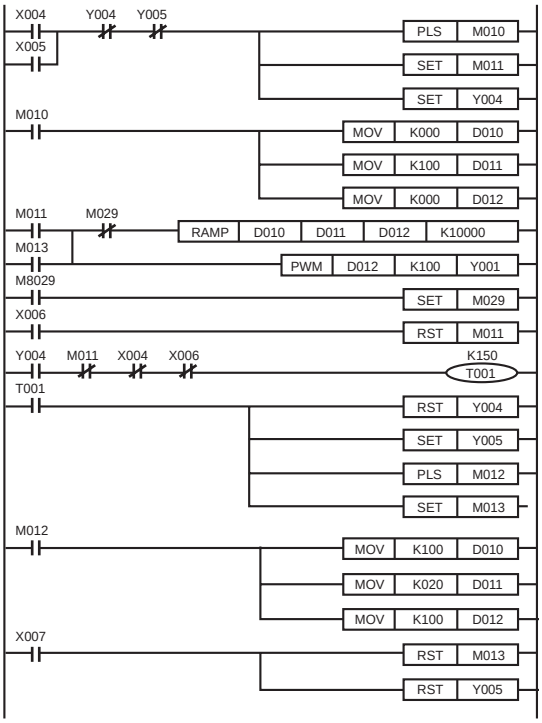
程序开始运行时，应给出一个瞬时输入X013，以保证C000 复位。接着应该是输入X000 动作，提供RAMP 指令的初始参数，再接下来，输入X001 起作用，程序就完全被起动。

附注：这里RAMP“模式标志”M8026 不需置位。

4.10 电机的软控制

一个重负载，比如一座桥需要软起动控制，以使电机能够工作。并且，使用摆桥时，当桥臂下放，电机也需要执行一个软结束过程，因为要防止桥板出现自由下落的状态。





器件	PC 软元件	说明
MTR1	Y001	驱动桥电机
	Y004	电机旋转-升起桥
	Y005	电机旋转-降下桥
PC1	X004	来自于下流的船
PC2	X005	来自于上流的船
PROX1	X006	桥升起位置
PROX2	X007	桥降下位置
MOV	FNC 12	MOV 应用指令
RAMP	FNC 67	RAMP 应用指令
PWM	FNC 58	PWM 应用指令

说明：

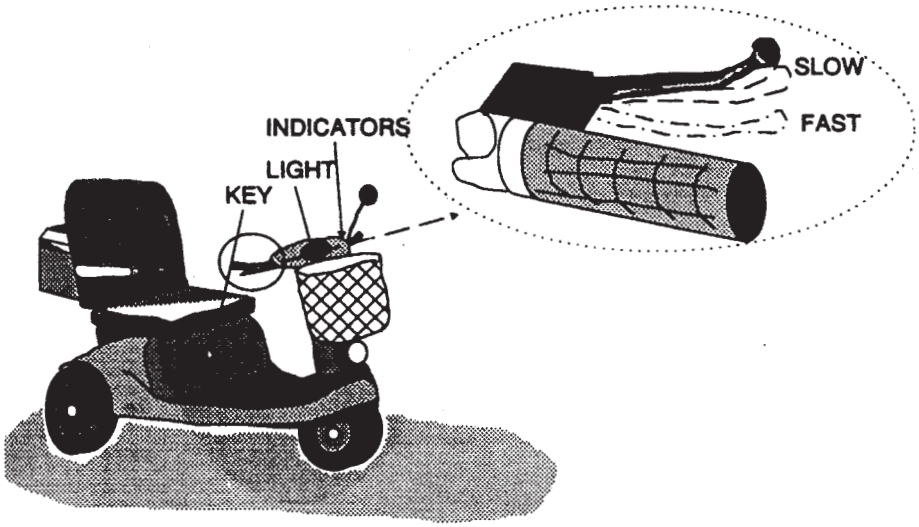
本例程序说明了一种可实现电机“软控制”的方法，它可以作为一个大型控制程序的一小部分程序。这个小程序检测任何到达的船（PC1 和PC2），并且升起桥，直到船离开，传感器不再有信号，经过一小段时间延迟后，桥才降落。

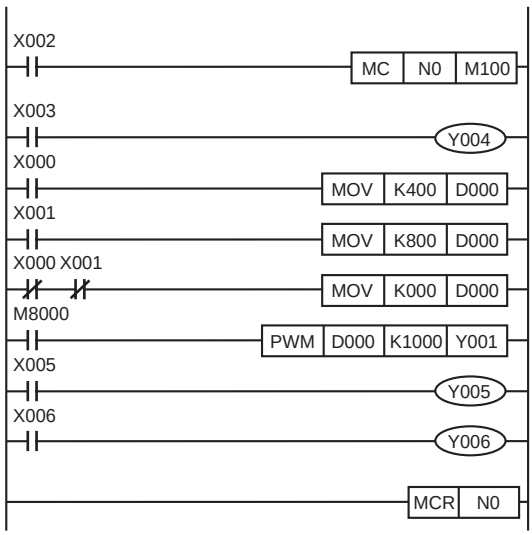
控制这么巨大的负载时，所关心的是电机的功率、效率和寿命。上升时，电机必须克服桥的重量，但下降时会碰到桥的自然运动趋势（在重力影响下），所以要经过计算以保持桥在被控状态，即不自由下落。

发驱动信号给电机在一定程序上要满足上面两个要求。上升时，电机缓慢上升，同时在下降时，电机缓慢下降，要求的最终结果是增加力矩以保持对电机的控制。在上升/下降过程中，这个系统最好有一个反馈回路来控制桥移动速度。

4.11 电动车

面对现有的可编程控制器的许多变化，为什么控制要局限于工厂的一个静止不动的小柜子里呢？使用带 DC 电源的微型 PC 就能做一项很实际的工作，比如使这种电动车在路上行驶。





器件	PC 软元件	说明
KEY	X002	允许电动车操作
LIGHT	X003	打开摩托车的起动灯
INDIC-ATORS	X004 X005	起动指示灯， 表示将运行的方向
FAST	X001	高速开动电动车
SLOW	X000	低速开动电动车
	Y001	电机驱动输出
	Y004	电动车起动灯
MOV	FNC 12	MOV 应用指令
PWM	FNC 57	PWM 应用指令

说明：

通常当想到一个可编程控制器时，工程师总想像到一个灰色的铁柜，里面有断路器、螺线管、保险丝、灰尘和油污。本节所讲到的电动车几乎包括所有这些物件，除了那个大铁柜。今天电池技术前进了一大步，在使用电气设备方面，给工程师提供了更大的余地，比如，便携式计算机等。所以，为什么不能在一个运动情形下使用可编程控制器呢？

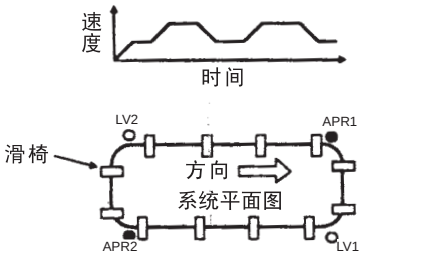
本例给出的程序包括了电动车的所有一般技术要求，即：指示、灯光等。它甚至有两种速率，这通过按压程序来选定。半按压给出一个向前的“低”速，而完全按压则给出一个“高”速。

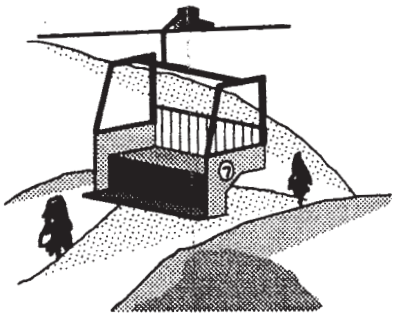
本例中使用到的一个有意思的技术是输入X002（钥匙开关），实际上是“使程序运行和断开”的方法。因为由输入来起动主控指令，主控中包含的程序步将只在X002 接通时被处理；而且由于主控中还写有其它程序，其结果是可靠地接通或断开程序。

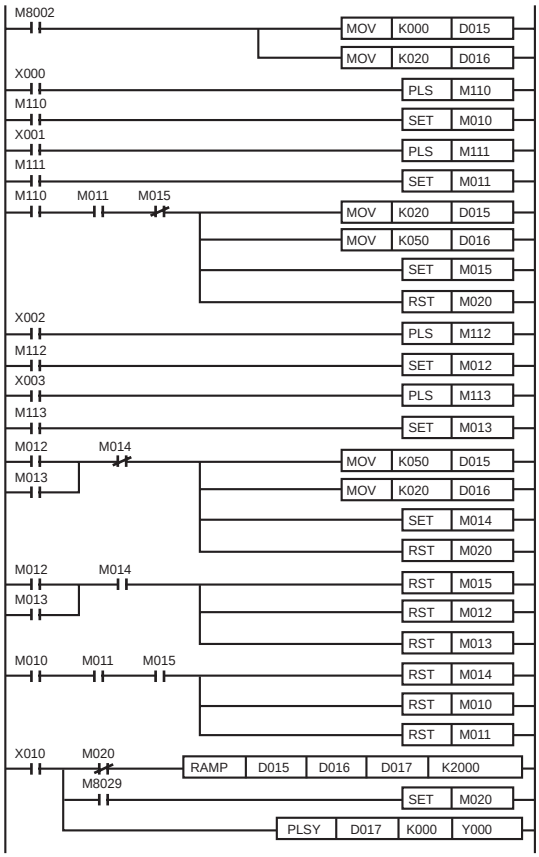
4.12 空中滑车

对于传送带、点动机器、空中缆车，它们都有某个共同点：适用于 PC 控制。如果我们观察空中滑车的运行，除了在减速的搭载点和卸载点，它一直以恒速移动，下面提出了一种传统双速控制系统的解决方法。

器件	PC 软元件	说明
LV1	X000	椅子离开-检测器 1
LV2	X001	椅子离开-检测器 2
APR1	X002	椅子到达-检测器 1
APR2	X003	椅子到达-检测器 2
	M8002	PC 运行初始脉冲
MOV	FNC 12	MOV 应用指令
RAMP	FNC 67	RAMP 应用指令
PLSY	FNC 57	PLSY 应用指令







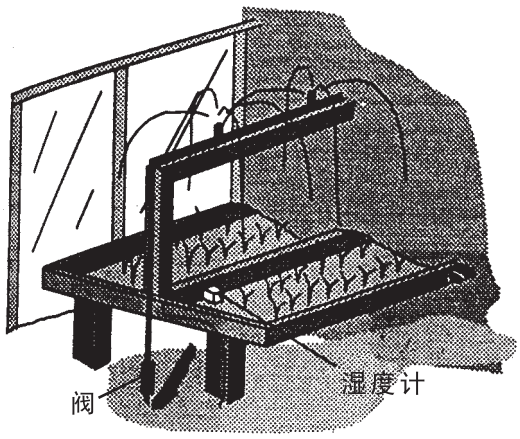
说明：

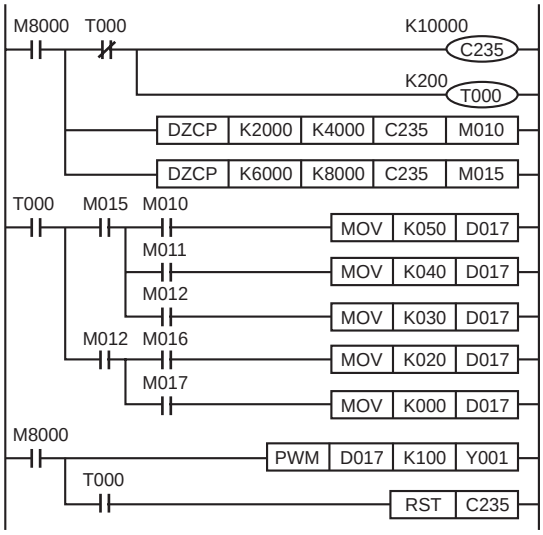
这个程序通过提供两种不同频率的脉冲序列给空中滑车的驱动电机。因为大多数空中滑车是很长的，而且带着许多车和人，所以要仔细考虑系统惯性。为了尽量使人满意，在电缆和电机上的负载减小时，这个程序保证所有的速度变化根据其方向上升或下降。

其中用到的一个有意思的安全互锁是：当其中一辆车到达搭载点或卸载点（输入X002 或X003 接通）时，系统转换到较低的速度。但系统如要转回高速，两车必须已经离开搭载/卸载区（输入X000 和X001 必须都接通）。

4.13 浇水控制

给花园里的植物浇水是一次肮脏的、湿漉漉的体验。如果每天都不得不做的话，不用说，肯定会让人厌烦的。看起来，这需要某种形式的自动化，特别对于带有商业性质的大规模区域。





器件	PC 软元件	说明
湿度计	X000	计数器 C235 脉冲湿度传感器输入
阀	Y001	用于设置阀门位置的输出
	C235	土壤湿度
	T000	采样和浇水的时间-根据当地条件设定
	D017	阀门比例设置
	M8000	PC 运行常闭触点
(D) ZCP	FNC 11	ZCP 应用指令
MOV	FNC 12	MOV 应用指令
PWM	FNC 58	PWM 应用指令

说明：

大多数工程师想到要给花园浇水时，他们拖出水管，把它接到厨房水龙头上，水溅得到处都是！在一个商业情况中，这是一场灾难。必须当心不要浇过多或过少的水。浇水时存在许多问题，当太阳光正强时，它会使植物“燃烧”，此时浇水会毁坏娇嫩的花朵，等等。当设计一个自动浇水系统时，所有这些因素及其它因素都需要考虑到。

本节给出的例子检测植物根部周围土壤的湿度，这个数据通过高速计数器输入X000传回PC，根据土壤条件决定要喷洒多少水量。PWM指令用来设定阀门打开的程序，及此时的水速及水压要多大。这个浇灌过程在定时器T000的定时时间内进行。

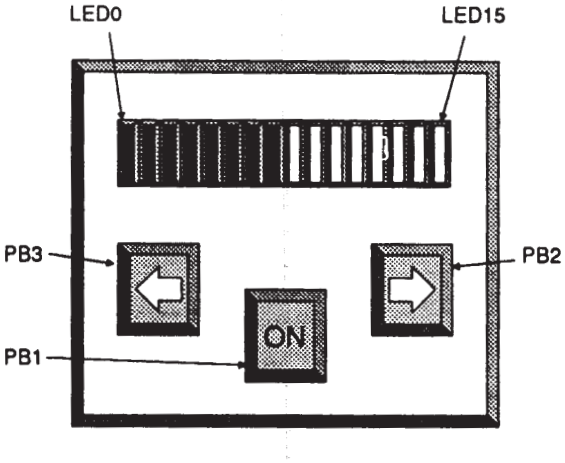
阀门复位到起始位置可以通过别的输出来做，或者此时阀门弹回。因此，不用驱动输出来使阀门复位（即关闭）。

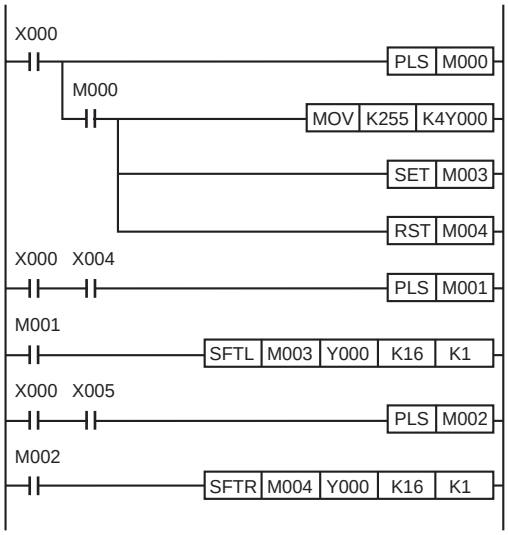
数据输入-数据输出：位元件

数据表现形多不止一种。工程师一听到字数据，便想到十进制或十六进制的数字。然而，有时需要考虑的是一种 ON 或 OFF 的位模式。

5.1 数值移位的保持控制

不是所有的编辑值或显示值都应该是数字形式的。有时，一个值表示为一种图形形式会更容易理解其含意。本例给出了一个数据的图形编辑和显示的方法。





器件	PC 软元件	说明
PB1	X000	激活设置单元
BP2	X004	增加条码图
PB3	X005	减少条码图
LED0-15	Y000-Y015	条码图发光二极管
MOV	FNC 12	MOV 应用指令
SFTL	FNC 35	SFTL 应用指令
SFTR	FNC 34	SFTR 应用指令

说明：

这个程序提供一个整齐的、易于操作的调整系统。当前数据显示为一个16位条码图，它在后面的程序中被读入一个数据寄存器。

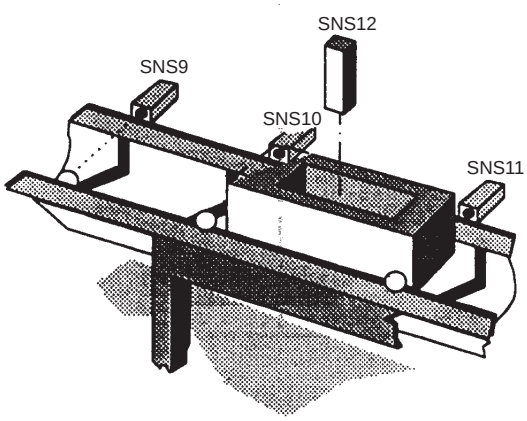
在按ON按钮前，整个单元是不运行的（此按钮应是一个自锁按钮，按一下为ON，再按一下为OFF）。运行后，设置条码图和结果数据到一个预定值：在本节例子中，是16位数据的中间值，即十进制数255。同时，两个作为移位寄存器源数据的辅助线圈初始化。

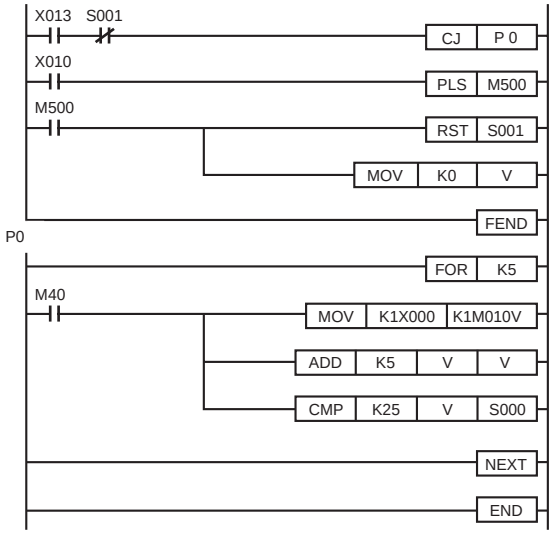
标着左右向箭头的两个按钮被用来调整当前值/条码图的设置。按下左箭头按钮，条码图的长度减小，当前数据值减小。这通过起动一个左移指令实现，简单地从位栈中删去最高有效位。同样地，如果按下右箭头按钮，条码图/数据值增大，这一次，用到一个右移指令，往位栈中加入一个ON位。

本节例子中，用到了一个16位条码图，不过使用两个移位寄存器可对更长的位进行同样的设置。这个程序可用来改变寄存器的数据值、定时器的时间值和计数器的计数值等。

5.2 使用一个回路“抓住”一串输入

本例说明怎样在短时间内用一个 FOR-NEXT 回路采集一个采样数据“集”。本例（工作时）在一个程序扫描期间对 4 个数据进行 5 次采集和记录。





器件	PC 软元件	说明
SNS9-SNS12	X000-X003	传感器输入
	X010	复位变址寄存器
	M040	采样序列控制
	X013	激活条件跳转
	S001	“V”=25 时有效
	V	变址寄存器
	P0	条件跳转调用的子程序
CJ	FNC 00	CJ 应用指令
MOV	FNC 12	MOV 应用指令
FEND	FNC 6	FEND 应用指令
FOR	FNC 8	FOR 应用指令
ADD	FNC 20	ADD 应用指令
CMP	FNC 10	CMP 应用指令
NEXT	FNC 9	NEXT 应用指令

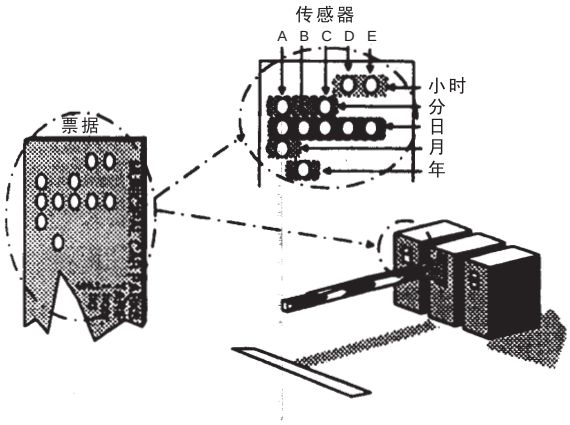
说明：

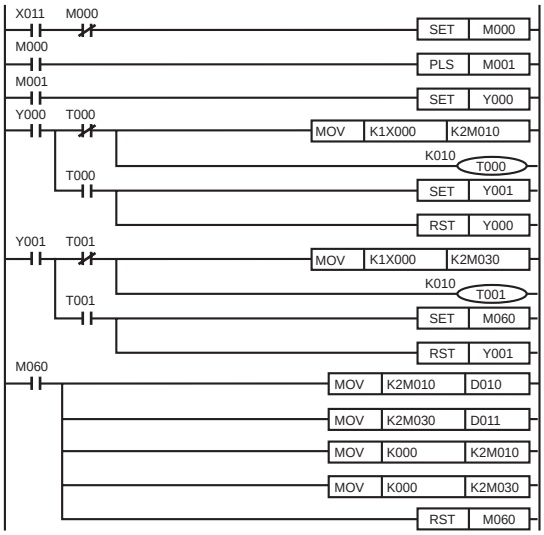
FOR-NEXT 回路是在完整的一次程序扫描中以规定次数运行某一程序段的一种方法。本例中，它用来控制某个采样输入数据的操作和存储。当输入 X013 有效，而且状态标志 S001 为 OFF 时，条件跳转到程序 P0。此时，如果辅助继电器 M40 为 ON，则对输入 X000-X003 进行 5 次“扫描”或“处理”，并且 5 组数据存入别的辅助线圈中。这 4 个 I/O 点的结果值被成批的连续存储，它们反映初始 4 个输入 X000-X003 的状态。每批结果之间用一个辅助线圈作为间隔。结果被存储后，标志 S001 被置位，防止别的数据采集，直到输入 X010 控制运行一段复位程序。

本例中，输入被用来“绘制”经过部件的形状，即：它是长的还是短的？它中间有洞吗？每个问题有相应的输入，这些输入将揭示答案，并因此展现扫描形状的轮廓。

5.3 读卡器

数据能以多种形式出现，从打孔带或打孔卡到磁带或可编程的 EEPROM。本例是输入数据到可编程控制器的有效方法。





器件	PC 软元件	说明
A	X000	读存在卡片上的数据
B	X002	
C	X004	
D	X001	
E	X003	
	X011	插入读卡器的票据
小时	Y000	起动系统读票据第一行
分	Y001	起动系统读票据第二行
	D010	存储“小时”值（从第一行卡数据上读出）
	D011	存储“分”值（从第二行卡数据上读出）
	T000	读“小时”数据的数据延迟
	T001	读“分”数据的时间延迟
	M000	操作标志-在程序别处复位（未标出）
	M060	数据移入存储器-复位位模式
MOV	FNC 12	MOV 应用指令

说明：

本例所示的停车场在你离开时作一次付费操作。离开车辆的司机把打孔卡放入出口处的机器中，机器计算停留的时间长度。打孔卡上有多种数据，但是为了储存在I/O 上，输入多次被使用。每条新数据线只在它对应的输出被激活时才被读入，即每条有洞的线依次被检测。洞的位置和数量决定存储的数据值是多少。所用到的检测技术可以是光学的，即：把检测器暴露在通过洞孔的光线中或检测金属通过洞孔，这只是许多可用技术中的两种。

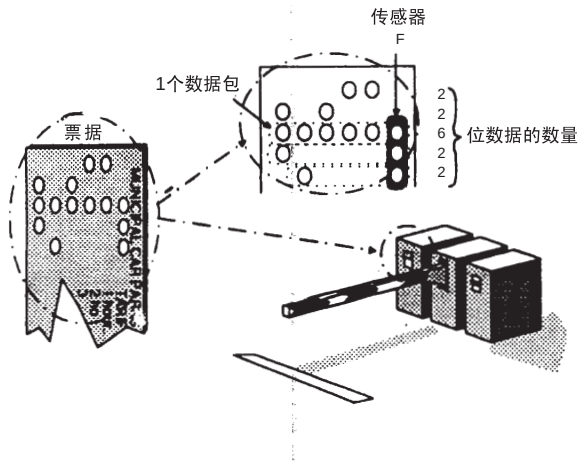
当存储数据作为一个位模式读入时，它被重组为一个数字值，从而计算停留时间、费用等。

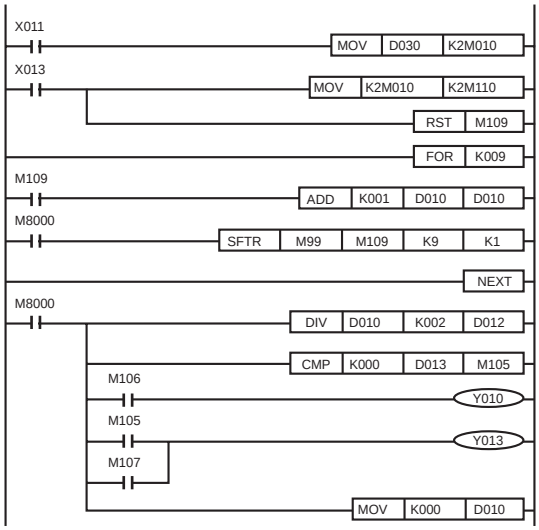
检测器/输入的位置混合起来，目的是实现一定程度的安全性。当检查打孔卡片时，使用的位模式不容易辨认，对于顾客来说，增加自己的打孔卡的洞孔，即改变进入时间，从而缩短停车时间和降低停车费用，是非常困难的。

注意：M000 是一个控制标志，用以初始化单个读操作。M000 在程序别的地方被复位。

5.4 偶数的奇偶校验

检测数据正确性的一种被认可的方法是观察这一批数据的数量，这叫做检测奇偶性。奇偶性可以是奇性的或偶性的，这意味着数据位的数量总和是一个奇数或偶数。





器件	PC 软元件	说明
F	X005	附加数据偶性标记（只作参考，不直接用在程序这部分）
	D030	测试数据寄存器
	D010	对测试数据中的有效位计数
	D012，D013	偶数校验
	M010-17	从打孔卡读出的原始数据
	M110-117	控制数据的复制
	Y010	数据是正确的
	Y013	数据是错误的（奇数数据元素）
MOV	FNC 12	MOV 应用指令
FOR	FNC 8	FOR 应用指令
ADD	FNC 20	ADD 应用指令
SFTR	FNC 34	SFTR 应用指令
NEXT	FNC 9	NEXT 应用指令
DIV	FNMC 23	DIV 应用指令
CMP	FNC 10	CMP 应用指令

说明：

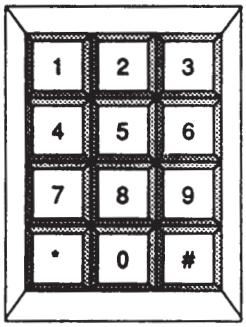
本节给出的例子说明了一种检测偶性的方法。一个数据包是对构成一个完整的数据值/事件的数据位的一次采集。数据包中的数据元素的数量加在一起，再除以2。如果结果没有余数，则表示数据包有偶数个数据位。输出Y010 接通来表示这一点。如果存在一个余数，则很明显地，数据包中的数据位数目是奇数，这意味着被测试数存在问题或错误。输出Y013 接通，报告这个错误事件。

例程中有两个重要的编程点，一个是处理一次程序扫描内的每个数据包的FOR-NEXT 回路的使用，另一个是对数据批内的数据位计数的SFTR 指令和ADD 指令的组合使用。

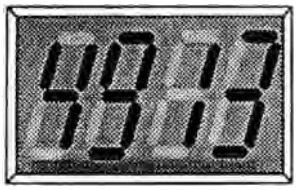
此技术常用于信息处理，但是正如本例所示，在别的数据采集场合它也有有益之处。

5.5 用户定义键盘

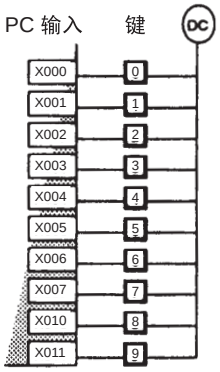
使用位处理技术，可开发出一个用户定义的键盘。

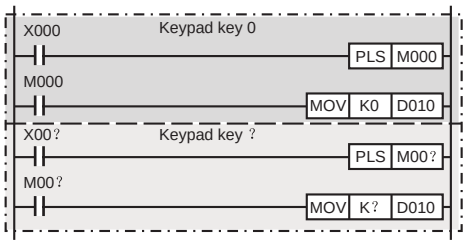


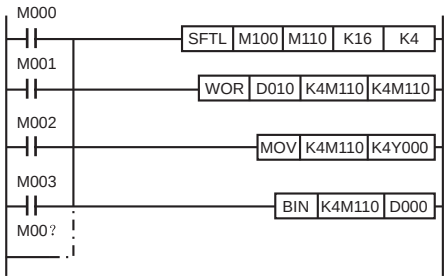
NK1



DSP1







器件	PC 软元件	说明
NK1	X000 到 X011	面板上的每个数字直接对应自己的输入
DSP1	K4Y000	显示输入的数据
	D000	后面程序段可能使用的实际数据存储
	D010	对应于激活键输入的数据值
	M100 到 M103	移位寄存器操作的空数据
SFTL	FNC 35	SFTL 应用指令
WOR	FNC 27	WOR 应用指令
MOV	FNC 12	MOV 应用指令
BIN	FNC 19	BIN 应用指令

说明：

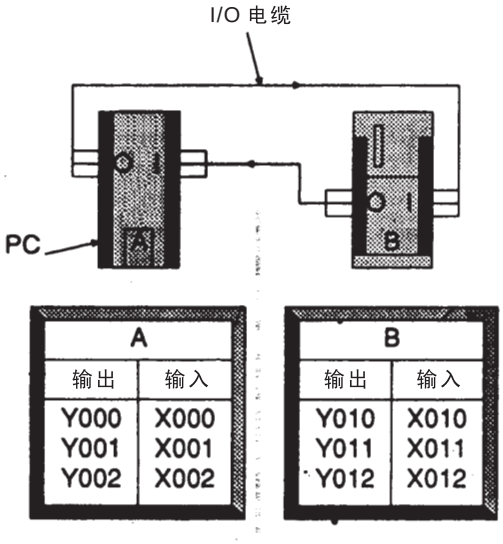
当按下一个数字键时，其值被加入存储在单个数据字中的数据串中。这个程序可以对0 到9999 的数字进行操作。如果超过最大限值，则最高位溢出、丢失。每个新近输入位放置在数据串的“单元”位置。经处理，输入的数字输出，给一个7 段显示DSP1，表示当前输入数据串是什么。

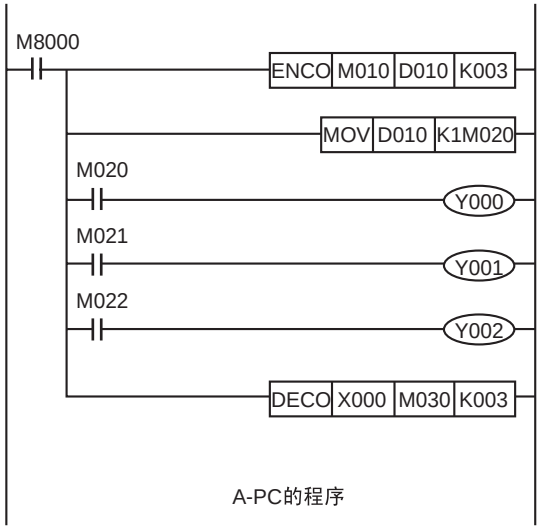
程序通过对位数据栈（首地址M110）左移4 位（SFTL 指令），把输入数字加到当前串。为实现这个目的，“空”数据值被移入到位元件M110、11、12 和13。当键入的数据值移入到寄存器D010 后，D010 内容与位数据栈（首地址M110）通过WOR 相连结。因为D010 的内容总是1 个数字（一个按钮输入），即一个4 位模式，可以说，D010 的前4 位被复制到位数据栈的预先“置空”区域中，此区域也为4 位。

接着位数据栈的内容被直接移出到一个7 段显示的输出。同时使用BIN 指令处理同一个位栈，其结果存在D000 中。这是一个直接读取当前数字串的过程。

5.6 多台 PC 间传送数据标志

在多台 PC 控制同一个工程时，这些 PC 之间往往有某种互锁。如果每台 PC 间传送的信息量有限，安装一个网络就不大有意义了。





A-PC的程序

器件	PC 软元件	说明
PC-A	X000-002	接收编码数据
PC-B	X010-12	
PC-A	Y000-002	发送编码数据
PC-B	Y010-012	
	D010	发送的编码数据
	D011	接收的编码数据
	M010-017	编码的数据标志
	M020-023	将要发送的编码数据
	M030-37	接收数据-解码
	M8000	PC 运行常闭触点
ENCO	FNC 42	ENCO 应用指令
MOV	FNC 12	MOV 应用指令
DECO	FNC 41	DECO 应用指令

说明：

本例说明了使用6 个I/O（3个输入，3个输出）是怎样使用户定义的状态标志在两台PC 间传递的。程序相当简单，只包含硬线连接，如果系统没有网络信息传送，这种方法是一个理想选择。

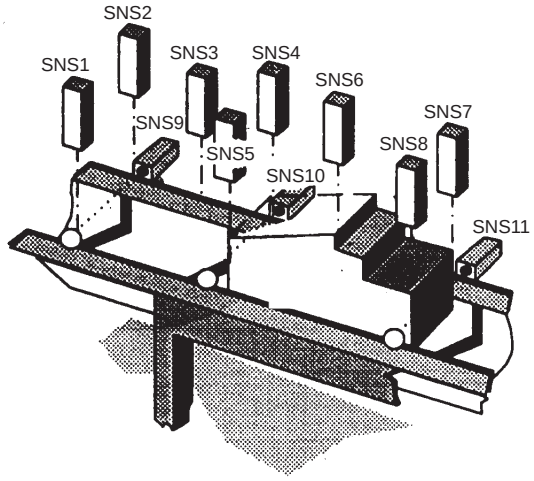
A-PC 的程序已在上边列写了。为了使之适于在B-PC 工作，只要求I/O 值变化，即X000 变为X010，Y000 变为Y010。

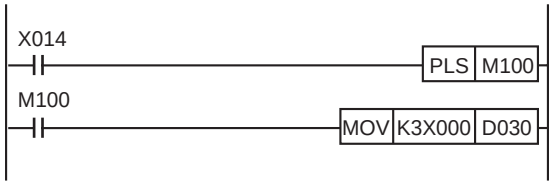
通过使用ENCO 和DECO 指令，本方法使用12 个I/O 点（每台6 个），能处理16 个不同标志。

标志可以用来表示任何事（从报告错误状态，到指示机器或生产线的控制）在相连PC 间传递。其他可能用途包括两台PC 顺序排列，或简单的说是“准备接收新的一批产品X”。

5.7 处理多输入情况

检测器阵列同时检测多个特性，为了便于处理所有数据，可以把他们组成一个数据字。





器件	PC 软元件	说明
SNS1-11	X000-13	组成一个阵列的不同检测器
	D030	保存采集到的检测器数据的数据区
	X014	控制信号
MOV	FNC 12	MOV 应用指令

说明：

如果使用基本编程技术分析来源于11 个检测器的数据，程序会是非常复杂冗长的。通过移动检测器状态，即把输入X000至X012 的状态移入一个数据字（例：D030），处理程序就变得容易得多了。

把位模式移入一个数据字时，应该注意，要确保源数据以4 位为单元的方式提供。K3X000的“K”规定多少个作为源数据的4 位单元组被读取或涉及。X000 表示源数据的首地址，因此，在例K3X000 中，使用从X000 开始的12 个位元件（到X013结束）。

移到数据字中的每个位元件直接映射到数据字的相同位置。X005 与X000 相距6 位，所以数据字中的位5（b5）将被用到，位5 也距离首位0（b0）6位。

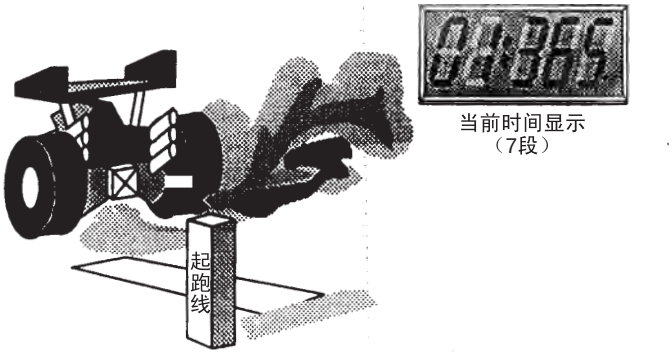
数据输入、数据输出：字元件

字数据可能是可编程控制器最常用的部分，不过也常是不容易理解的部分。

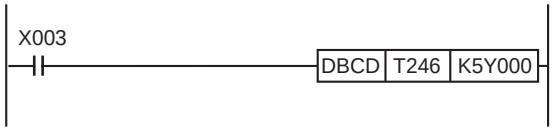
6.1 定时器数据的 7 段 BCD 输出

赛车完成比赛所用的时间显示在一个 7 段模板上。

输出	Y20-23	Y14-17	Y10-13	Y4-7	Y0-3
BCD 码	0	1	3	6	5



器件	PC 软元件	说明
	X003	使存在 T246 中的当前时间数据输出到 一个 7 段显示的触发
	T246	对短程赛车计时的定时器
	K5Y000	输出 Y000 至 Y023 用来输出 BCD 码到 7 段显示
(D)BCD	FNC 18	BCD 应用指令



说明：

保存在定时器T246 中的当前时间能被“复制”到输出，这里为K5Y000。这实际上表明5 个BCD 码数字将从输出点送出，从地址Y000 开始。

每个BCD 码要求4 个输出，即一个BCD 数字要求4 个输出，比如输出K1Y000，将用到输出Y000 至Y003。本节例子中，使用Y000 至Y003。最低BCD 数一般分配输出区的首地址。如果39876 要被输出，则6 就被分到首地址区，为了查看输出数据，选择的输出必须接线至一个接受BCD 码的显示单元。这里使用一个典型的7 段显示。

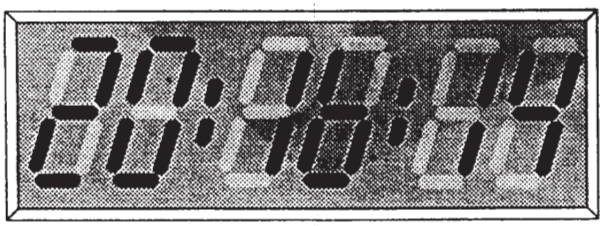
本例中，当X003 接通时，整个操作被激活。在X003 保持为ON 期间，将连续进行BCD 的转换和输出。

附注：因为有5 个数字用BCD 指令编码，指令必须设置使用双字。将D 加于BCD 指令前来实现这一功能。

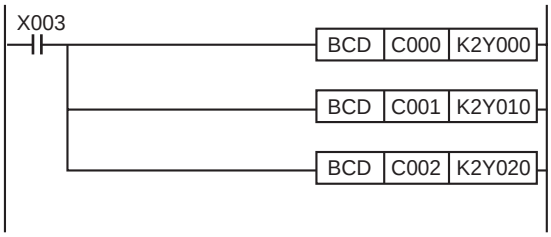
6.2 显示计数器内容

输出三个计数器的内容到一个 7 段显示。

输出	Y24-7	Y20-3	Y14-7	Y10-3	Y4-7	Y0-3
BCD 码	2	0	1	6	1	4



当前时间显示（7 段显示）



器件	PC 软元件	说明
	C000	计数器-秒
	C001	计数器-分
	C002	计数器-时
	X003	激活显示
	K2Y000	秒输出
	K2Y010	分输出
	K2Y020	时输出
BCD	FNC 18	BCD 应用指令

说明：

每个计数器的内容以BCD 的格式输出。这些输出与一个7 段显示的连接使用户可以看到数据。因为本节例子中的每个计数器只用2 个数字，BCD 码输出写成K2Y**。K2可以当作是要显示的数字个数。

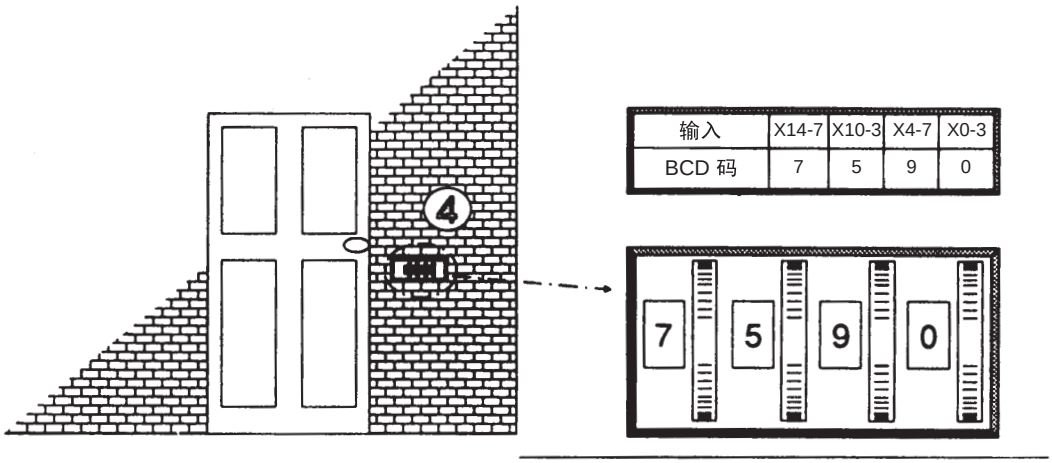
每个BCD 码数字要求4 个输出，因此要求使用Y**0 到Y**7。本节例子中，C000、001 和002 分别使用Y000-007、010-017、020-027（注：每个计数器有2 个数字）。K2Y**中表示的Y**可以当作是输出区的首地址。

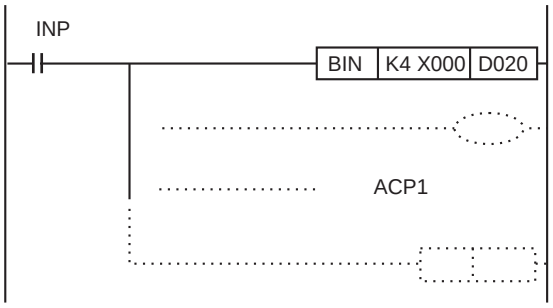
附注：当BCD 码数据输出时，最低位数字一般从首地址的第一组4 个输出读取，如对于数字12345，会发现从输出Y000 至Y003 读出的值是5。

本例中，接通输入X003 时，数据转换为BCD 码并显示。

6.3 输入一个通行码

进入房子需要输入正确的通行码。





器件	PC 软元件	说明
INP		激活输入
ACP1		比较输入码与当前存储码的程序-若码相同，门开
BIN	FNC 19	BIN 应用指令
	K4X000	用以接受输入码的输入元件
	D020	输入码的数据存储元件

说明：

为了进入房子，必须在指轮上输入正确的通行码。来访者在指轮上拨入他们的通行码，可编程控制器（PC）将输入与一已知码相比较。

接着程序结构就能根据比较结果拒绝或允许通行。如果研究数据输出的“机械结构”。就可看到数据以BCD 格式从指轮传送到PC，即每4 个输入信号表示一个指轮上选择的数字。本节例子中，K4X000 意味着使用输入X000 至X017。X000 作为输入数据区的首地址。

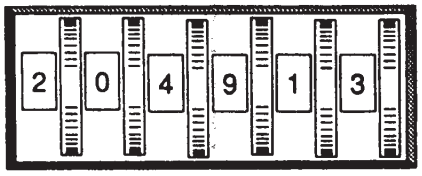
注：输入BCD 码数据时，最低位数字一般从首地址的第一组4 个输入读取，即对于数字56789 会发现从输入X000 至X003 读出的值是9。

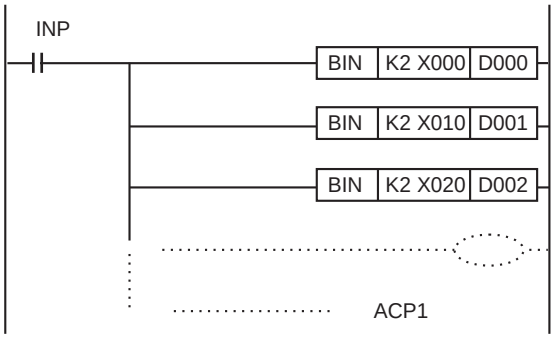
本例中，当用户程序接收到以INP 为标记的某种起动信号时，BIN 指令被激活。

6.4 输入多组数据

在指轮上输入正确的时间设置。

输入	X24-7	X20-3	X14-7	X10-3	X4-7	X0-3
BCD 码	2	0	4	9	1	3





器件	PC 软元件	说明
INP		触发输入
ACP1		激活 7 段显示的附加程序
	K2X000 K2X010 K2X020	输入器件-分别输入秒、 分、时的数据
	D000 D001 D002	数据存储区-分别存储秒、 分、时的数据
BIN	FNC 19	BIN 应用指令

说明：

时钟需要正确设置。在指轮上拨入一个所要求的时间，接着使用BIN 应用指轮从指轮读出拨入数据，紧跟着，用户程序（记为ACP1）允许将输入的数据设定为当前时间。

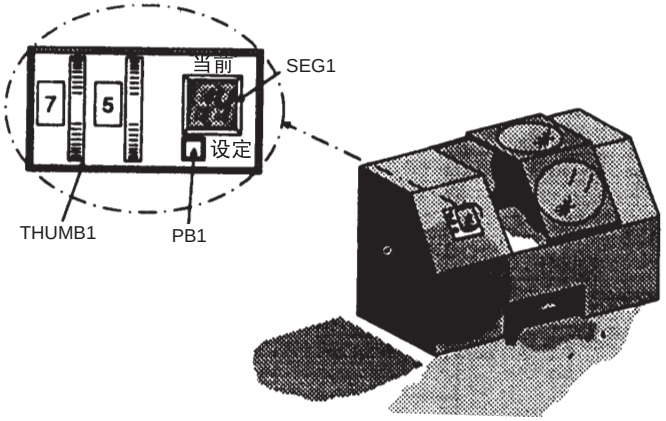
本节例子中，K2X000意味着使用输入X000 至X007。X000 作为输入时间区的首地址，接着读入的数据存入一个数据地址中。这是一种好做法，保证每个BIN 指轮把“读入”数据移入不同的数据位置。如K2X000 的数据读入到D000 中。

注：输入BCD 码数据时，最低位数字一般从首地址的第一组4 个输入中读出，即对于数字65734，可以发现在输入X000 至X003 读出的值是4。

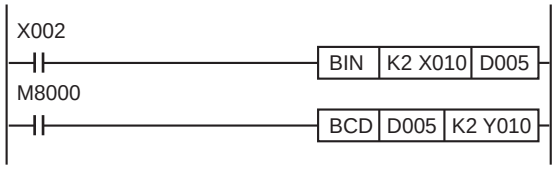
本例中，当用户程序接收到以INP 为标记的某种起动信号时，BIN 指令被激活。

6.5 输入一个数据并检测结果

用指轮输入数据时，最重要的是检测当前设置，其次是检测新数据是否已装入。



器件	PC 软元件	说明
SEG1	K2Y010	对 7 段显示的 BCD 输出-当前使用工具号
PB1	X002	选择新工具号
THUMB1	K2X010	将使用的新工具号
	M8000	PC 运行常闭触点
BIN	19	BIN 应用指令
BCD	18	BCD 应用指令



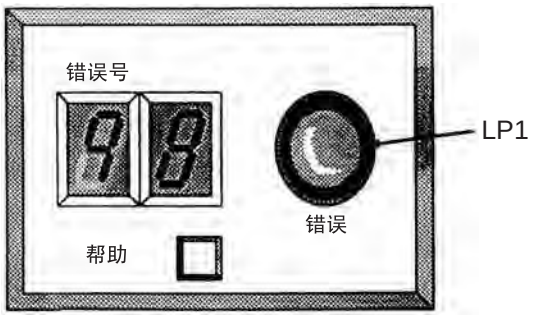
说明：

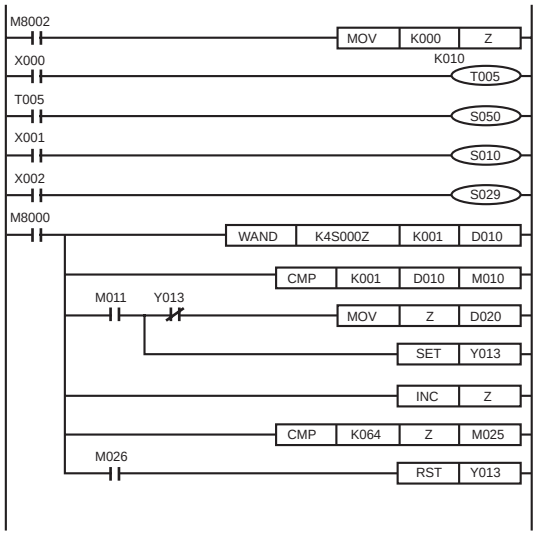
数据寄存器D005 的当前内容持续输出到一个7 段显示，使操作者能察看寄存器内容。如本例情况，数据反映某个参数变化，这种方式能提供给操作者有关操作过程中机器所到达位置的有用信息。

新数据通过指轮输入。只有在“设定”按钮PB1 被按下时此数据被读取。这使得新数据在输入到机器运行过程之前被显示检测。一旦新数据输入，设置键按下后，7 段显示会改成这个新的数据。

6.6 用户设定的错误处理

设定一个系统时，可能需要一种自诊断或保持源错误码的方法。本节例子给出了一个达到此目的的方法。





器件	PC 软元件	说明
LP1	Y013	错误指示灯
	M8000	PC 运行常闭触点
	M8002	PC 运行初始脉冲
	S010	用户定义错误号 10
	S029	用户定义错误号 29
	S050	用户定义错误号 50
	D010	被扫描的单个错误状态
	D020	最低的错误号
MOV	FNC12	MOV 应用指令
WAND	FNC26	WAND 应用指令
CMP	FNC10	CMP 应用指令
INC	FNC24	INC 应用指令

说明：

这里所示的程序能够处理多达63 种不同的错误。每个错误分配给一个位元件，S000 到S063。本例说明了在一种错误情况下三个错误码是如何被定位和处理的。

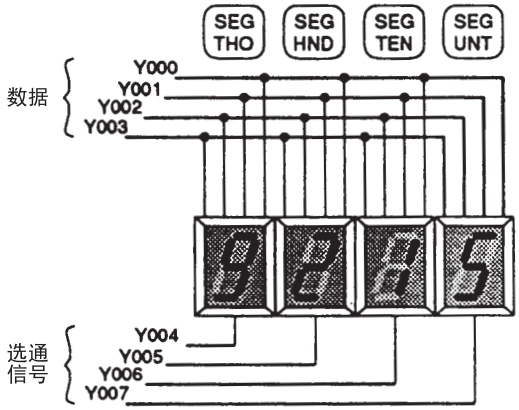
输入X000 至X002 激活三个错误。输入接通时，表示错误存在，当一个错误出现，如X001 为ON 时，它的错误号移入数据寄存器D020。这时，D020 会等于数字10（因为X000 使标志S010 接通）。为了确认当前有错误，输出Y013 被设置为ON。这个输出表现为脉冲或闪烁输出，这是因为每64 次程序扫描，它被置位和复位一次。

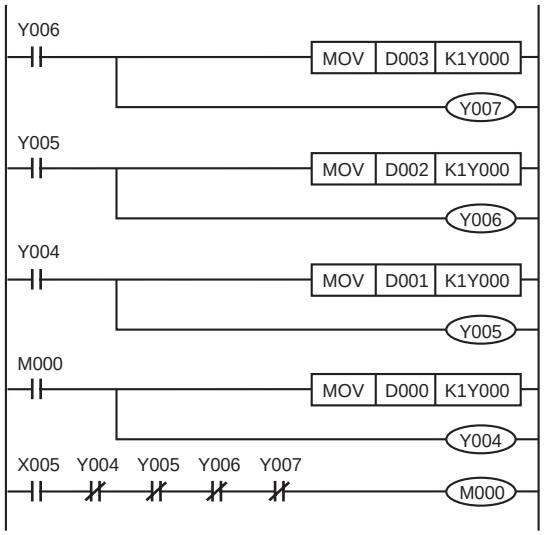
如果存在不只一种错误标志，如S029 和S050 有效，最低错误号即29 会存入D020。当引起错误29 的条件修正时，下一个错误号即50 会存入D020。

注意：高序号的错误将比那些较低号错误能引起输出Y013 更快地变化。这是因为在一个扫描期间，每个错误标志被单独地处理，64 个错误标志需要64 次扫描。一个较低错误标志会比较高错误号更长时间地使输出Y013 为ON。这可以作为一种确定错误紧急程度的方法，即决定较低错误号是重要的，因为在扫描出现的错误时，它们有优先权，并且它们使输出Y013 近乎持久地保持为ON。错误号可输出到一个7 段显示。

6.7 多路传送 7 段显示

本例说明多路传送显示设备是如何更有效地使用有限资源。为使本例能真正被应用，锁定 7 段显示要求使用晶体管输出。如果使用继电器输出，控制器的寿命会受到严重影响。在控制信号消失后，锁定显示用以保持当前显示数据。





器件	PC 软元件	说明
	X005	运行系统
SEG-THO DATA	D000	通过 Y000-003 输出-代表千位数据
SEG-THO STROBE	Y004	使数据读出-千位显示
SEG-HND DATA	D001	通过 Y000-003 输出-代表百位数据
SEG-HND STOBE	Y005	使数据读出-百位显示
SEG-TEN DATA	D002	通过 Y000-003 输出-代表十位数据
SEG-TEN STOBE	Y006	使数据读出-十位显示
SEG-UNT DATA	D003	通过 Y000-003 输出-代表个位数据
SEG-UNT STOBE	Y007	使数据读出-个位显示
MOV	FNC12	MOV 应用指令

说明：

本节例子说明了如何用8 个输出发送正常要求16 个输出的4 位数据。秘密在于多路传送，使用的8 个输出分为一个“数据”组和一个“控制”组。

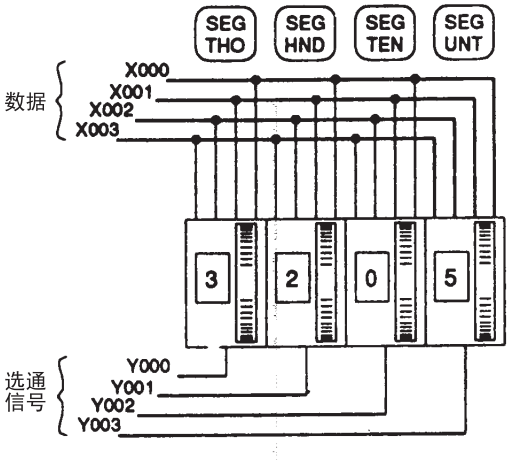
数据的每一位要求有4 个输出发送信息到显示上，对于每一位数字，有一个显示，因此4 个数字有4 个显示。保存的4 个输出轮番激活每个显示，最后结果是选通这些显示。因为这发生得非常快，显示需要锁定数据，使操作者能看到这些值。

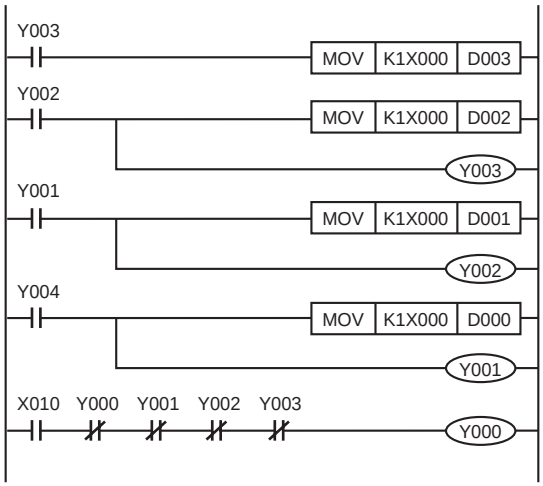
要输出的数据源在本例中是数据寄存器D000-003。每个数据寄存器只保存一位，根据输出/显示顺序，该位可能是千位、百位、十位或个位。

因为每个数据寄存器中的数据只有1 个数字，即0-9，使用MOV 指令来代替BCD 指令，这会使程序操作速度稍微快一点。

6.8 指轮输入的多路传送

本例说明多路传送显示设备是如何更有效地使用有限资源。为了本例能真正被应用，要求使用晶体管输出。如果使用继电器输出，可编程控制器的寿命当然会受影响。同时也建议在输入上使用分压电阻，并配有可调整的 PC 输入滤波器。





器件	PC 软元件	说明
	X010	运行系统
SEG-THO DATA	D000	X000-003输入-代表千位数据
SEG-THO STROBE	Y000	使数据读入-千位指轮
SEG-HND DATA	D001	X000-003输入-代表百位数据
SEG-HND STOBE	Y001	使数据读入-百位指轮
SEG-TEN DATA	D002	X000-003输入-代表十位数据
SEG-TEN STOBE	Y002	使数据读入-十位指轮
SEG-UNT DATA	D003	X000-003输入-代表个位数据
SEG-UNT STOBE	Y003	使数据读入-个位指轮
MOV	FNC12	MOV 应用指令

说明：

本节例子说明了如何用4 个输出和4 个输入发送正常要求16 个输入的4 位数据。秘密在于多路传送，4 个输入作为一个“数据”组，而4 个输出作为一个“控制”组。

数据的每一位要求有4 个输入从指轮读信息。对于每一位数据，有一个指轮，因此4 位对应有4 个指轮。保存的4 个输出轮番激活每个指轮开关，最后结果是选通这些指轮。因为这发生得非常快，需要有预防措施，如使用分压电阻、和/或调整输入滤波器。

输入数据的目标元件，在本例中是数据寄存器D000-003。每个数据寄存器只保持一个数字，根据使用顺序，该位可以是千位、百位、十位或个位。

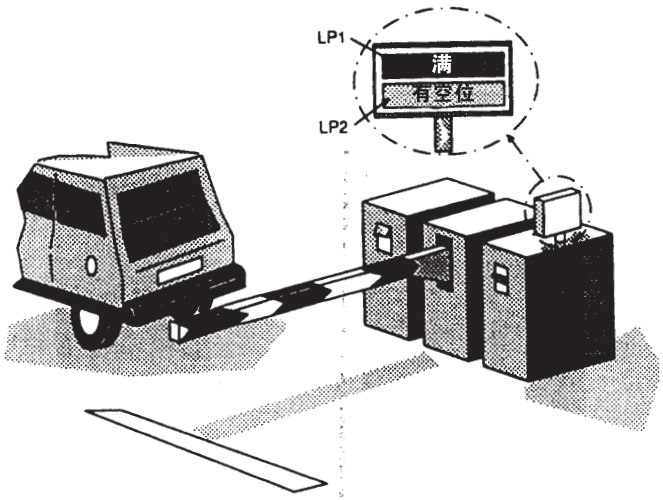
因为每个数据寄存器中的数据只有1 个数字，即0-9，使用MOV 指令来代替BCD 指令，这会使程序操作速度稍微快一点。

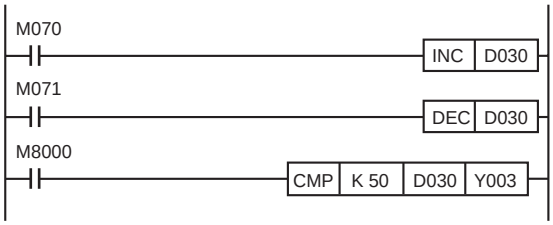
比较和基本事件跟踪

一个想法的价值只为那些欣赏其价值的人所知。这里提出的完整的陈述能同等地适用于位元件状态或数据寄存器内容。

7.1 一个数值的保持控制

数据控制和处理可报告一个动作的实际状态，比如，停车场现有多少辆车？





器件	PC 软元件	说明
LP1	Y004Y005	停车场已满
LP2	Y003	停车场有空位
	M070	有车进入停车场
	M071	有车离开停车场
	D030	停车场车辆数（最大数=50）
	M8000	PC 运行常闭触点
INC	FNC24	INC 应用指令
DEC	FNC25	DEC 应用指令
CMP	FNC10	CMP 应用指令

说明：

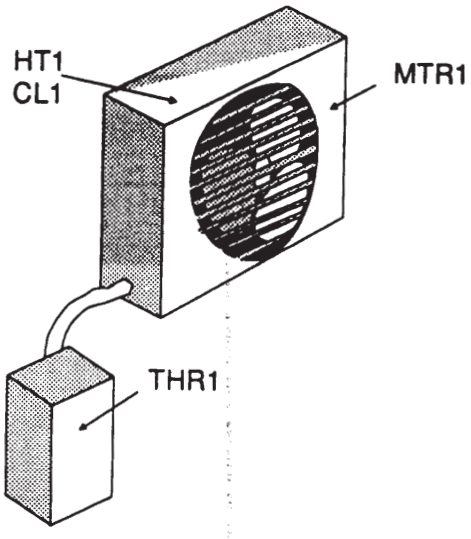
用户程序的前面部分可以检测车辆进入或离开停车场，这个信息可以通过使用内部线圈和标志传送到程序的其他部分。本例中，标志M070 和M071 代表车进入或离开停车场。当有一辆车进入停车场时，当前停车数量的记录加1， 即对数据寄存器D030 的内容执行一个INC 指令。

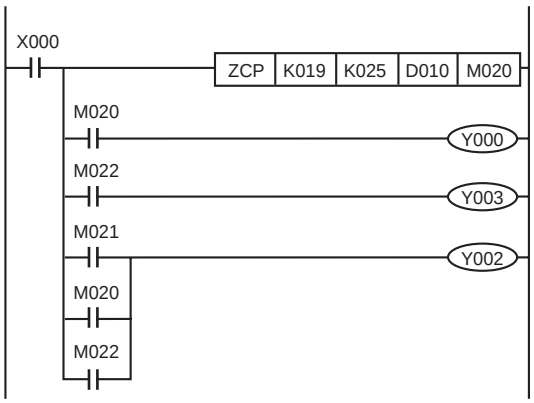
CMP 指令由特殊M 线圈M8000 驱动，使寄存器D030 不断地与已知能容纳的最大车辆数作比较，当两值相等时，停车场满位。因此比较指令使灯LP1（Y004）亮，表示停车场满位。

反之，如果一辆车离开，DEC 指令（车离开由标志M071 驱动）对数据寄存器减1。当停车场的车辆小于最大数时，灯LP2（X003）亮，允许司机进入。

7.2 比较数据范围

某些应用要求输入数据与预置数据值相比较。有时是单个数字与别的数字相比较，有时是一个数字与一个预置范围值相比较。





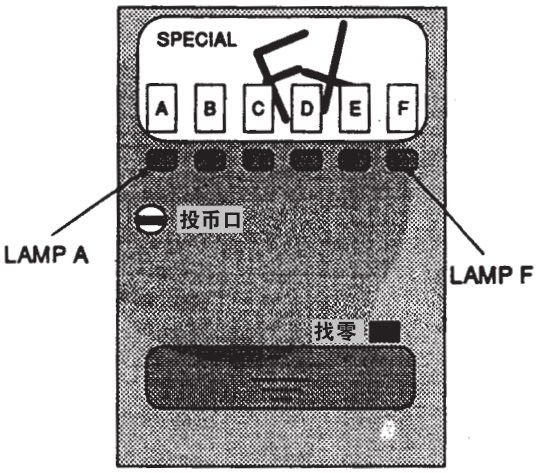
器件	PC 软元件	说明
	X000	空气调节器起动
HT1	Y000	控制加热设备
MTR1	Y002	控制风扇运行
CL1	Y003	控制冷却器
THR1	D010	来自自动调温器的数据输入
ZCP	FNC11	ZCP 应用指令

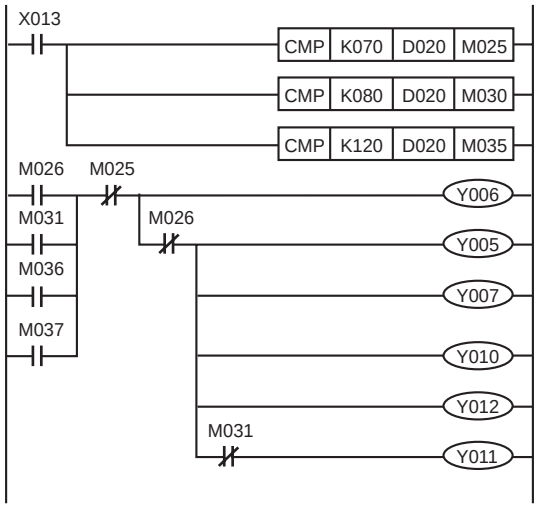
说明：

一个加热控制系统要求当前温度对照一个温度范围来校核，此范围规定了一个周围工作环境。本例将存储在 D010 中的温度值，由调温器 THR1 读出，与 K19 和 K25 作比较，这温度范围表示一个舒适的办工环境。做完比较，必须有一个结果，这个结果比较得到 3 个主要状态，即＞、=、＜。这 3 个状态可用位标志表示，本例中使用了 M20 到 M22。所有情况中，当使用空气调节器时（X000 接通），电扇会运行，即输出 Y002 为 ON。这是因为所有的指示标志都起动这个输出。当房间温度低于规定范围时，标志 M20 接通，起动输出 Y000，使加热设备 HT1 运行。如果房间太热，即实际房间温度超过规定范围，标志 M22 接通输出 Y003，冷却器工作。

7.3 从比较中得到更多

每个可编程控制器有许多独有的特征。但是常常要问，指令真地最大限度地使用了吗？很常见地，当编写程序时，一种“习惯”开始出现，就很难改掉。基本上我们每个人总坚持我们所最了解的。





器件	PC 软元件	说明
LAMPA-F	Y005-Y012	对应每种产品的指示灯
	M025-M037	3次比较的结果标志
CMP	FNC10	CMP 应用指令
	D020	包含有关投入的硬币的数据，即多少钱

说明：

比较指令是最容易使用的指令之一。然而，当数据范围被要求检测时，好象是要引起恐慌，或者好象过于复杂的系统是当今的要求。区间比较很容易地用多个单值比较状态来完成，正如使用复杂的区间比较状态。这里给出的例子是一台售货机。

投入机器的钱数多少决定选择的是哪种饮料。钱的总数必须等于或超过所买商品的规定价格，一种简单的单值比较就足以解决这个问题。因此这里只关心当钱不等于或超过物品价时的情况。

本例中货品：

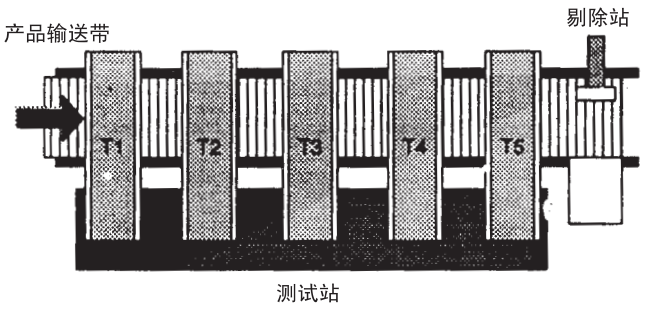
A，C，D和 F 价格为 80 美分；

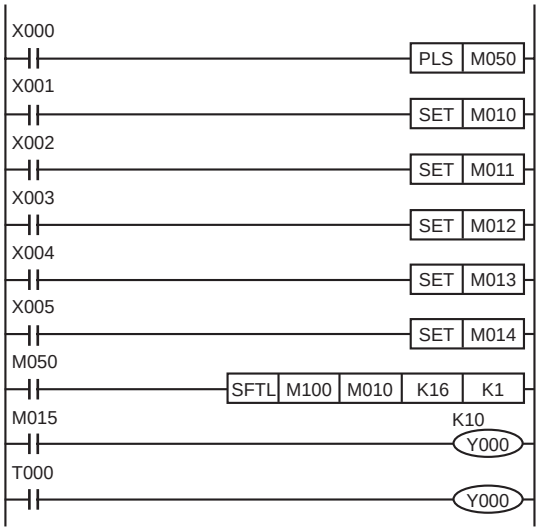
E 价格 120 美分；

B 价格 70美分。

7.4 跟踪一个事件

一个产品在 5 个测试站被检测，任何一个测试站上的失败会导致该产品被剔除。本编程技术可用于一个简单跟踪系统。





器件	PC 软元件	说明
	X000	指示传送带已移过一个站的距离
T1	X001/M010	测试站 1-接通=失败
T2	X002/M011	测试站 2-接通=失败
T3	X003/M012	测试站 3-接通=失败
T4	X004/M013	测试站 4-接通=失败
T5	X005/M014	测试站 5-接通=失败
	M015/Y000	控制剔除塞
	M100	移入 SFTL 的空数据位
	T000	剔除塞操作前的时间延迟
SFTL	FNC35	SFTL 应用指令

说明：

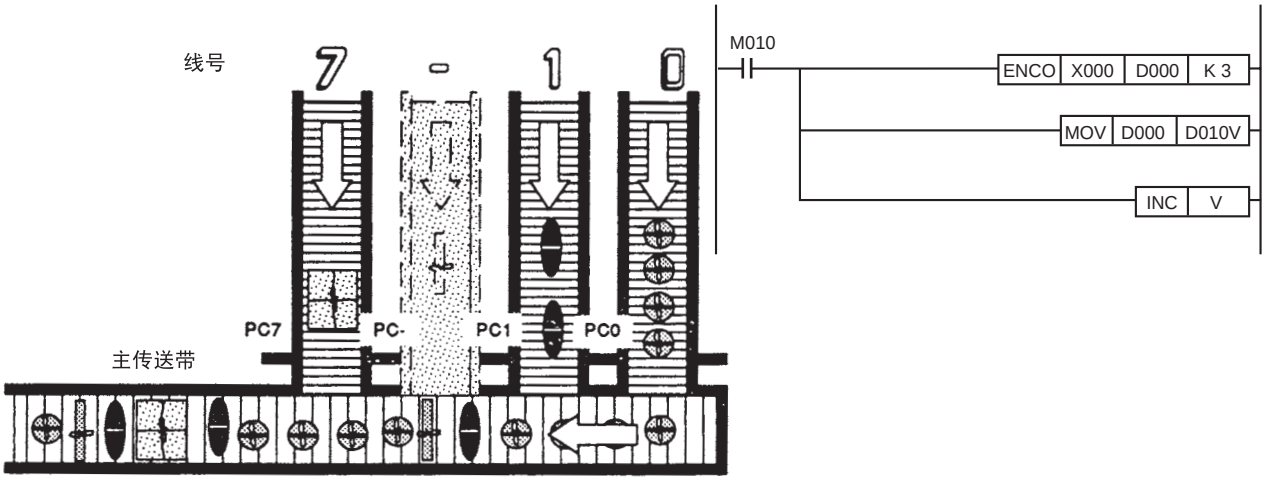
产品经过每个测试站时，进行一个不同的测试。如果产品没通过测试，一个标志设置为 ON。当产品移到别的测试站时，它的当前状态即失败或通过，会跟着它。当产品到达传送线末端时，它可能从流水线上剔除，或被允许继续行进。这决定于测试的结果。本例中，如果任何一站测试失败，则该产品就被剔除。

程序依赖于一个位移进程来保持数据对与它相关的产品的跟踪。每个测试站分配给一个位，从而形成“数据栈”。本例中，如果产品没通过一个测试，该站相应的数据位设置为 ON。当产品移至下一个站时，状态也移到新站分配的位。

经过最后一个测试站后，最后的位置是一个剔除站。本例中，如果任何一个测试出现失败，产品就自动从传送带上剔除。如果产品的测试没有失败，则继续行进。

7.5 编码一个响应为一个数据值

下面的邮包/包裹流水线在每个包裹离开时给它分配一个号码，这个号码表示包裹来源于哪条线，由此表示产品是什么。



器件	PC 软元件	说明
PC0 - PC7	X000 - X007	生产线检测器
	M010	内部激活标志
	D000	编码数据区
	D010	数据栈
	V	变址寄存器
ENCO	FNC42	ENCO 应用指令
MOV	FNC12	MOV 应用指令
INC	FNC24	INC 应用指令

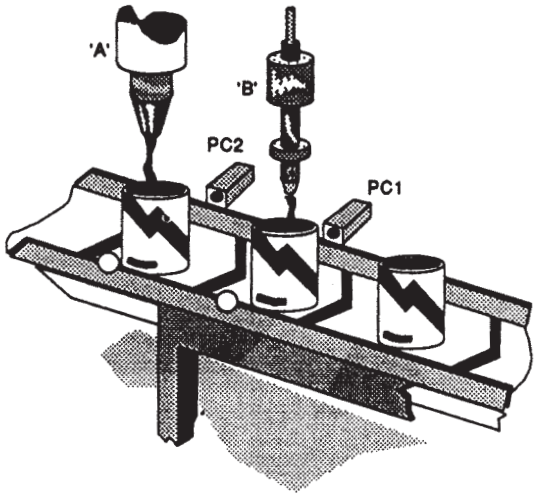
说明：

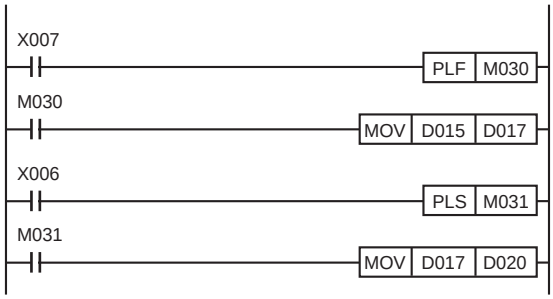
本例显示的一小段程序给产品分配一个数字码，此码表示起初的存贮线。当每个产品送到主传送带时，在存贮线末端它切断一个光电管（PC0 到 PC7）的光线。这提供了 X000 到 X007 范围的一个输入。这个输入被编码以表明它源于哪条存贮线，即如果 PC5 接通输入 X005，那么线号全是 5。

这个线是被输入到一个数据栈（本例中，数据栈首地址是 D010）。数据栈反映主传送带上每个包裹的位置，这提供了一种产品跟踪的基本模式。当包裹离开主传送带，“跟踪数据”从数据栈中删除，而栈数据的其余部分就往上“移”。本例中，数据栈装有每个新数据。新近装入数据的位置决定于变址寄存器，这里为 V。使用类似方式“移”数据，删除刚离开主传送带的包裹的相关数据。

7.6 移动相关信息

可编程控制器通过给生产线上的每个罐子加一个数字标识来跟踪它。





器件	PC 软元件	说明
PC1	X006	光电管（检测涂料站 B 上的罐子）
PC2	X007	光电管（检测涂料站 A 上的罐子）
	D015	涂料站 A 上的罐子的数据
	D017	涂料站 B 上的罐子的数据
	D020	离开涂料站的罐子的数据
MOV	FNC12	MOV 应用指令

说明：

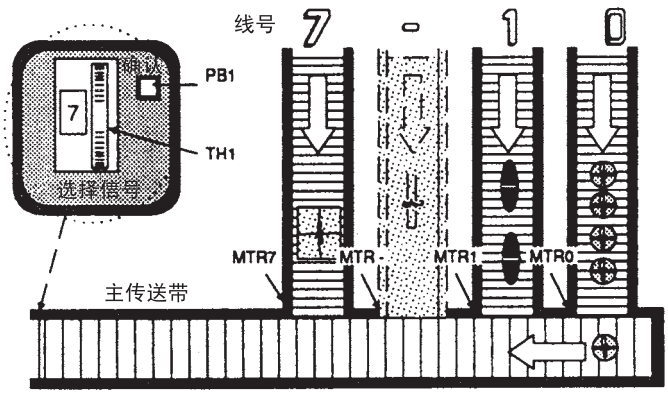
本例是一个较大的数据跟踪系统的一小部分。在此前的用户程序中，初始数据已经输入。

在工厂中罐子移送期间，它进入涂料车间，当罐子定位在涂料站 A，其相关数据放入数据寄存器 D015。当罐子移到涂料站 B，它的相关数据跟着移动，即数据寄存器 D015 内容被移到数据寄存器 D017 中。

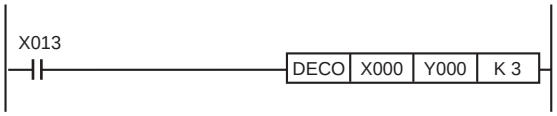
离开涂料站 B 时，相关数据被送到寄存器 D020，全厂使用类似的处理。在某些时候，数据可用来决定相关罐子的下一个操作或目的地，也可以代表罐子的颜色、货主或尺寸。这个系统可扩展为多类数据移动，例如同时跟踪三个数据。

7.7 输入解码为相应的输出

本例说明用指轮完成的输入/选择是如何直接接通一个相应的输出的。



器件	PC 软元件	说明
PB1	X013	“确认”按钮 - 起动线选择
TH1	X002 - X002	选择“呼叫线”的指轮输入
MTR0 - MTR7	Y000 - Y007	接通选择的“呼叫线”
DECO	FNC41	DECO 应用指令



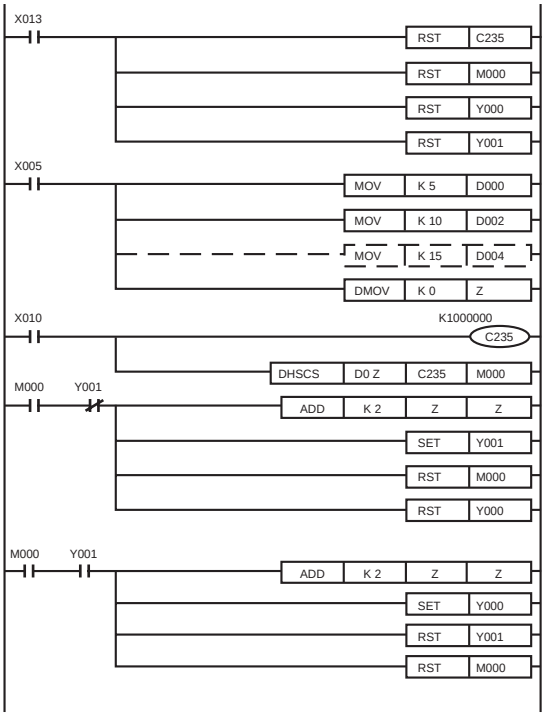
说明：

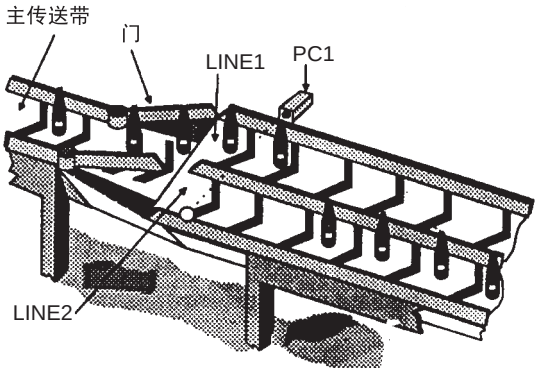
指轮（TH1）用来选择取下包裹的流水线。当指轮显示所要求的线时，按下“确认”按钮（PB1），从而激活解码指令。解码指令从指轮上读 BCD 输入，并且把它解码为一个相应的输出，即如果指轮上设置 7，则 Y007 接通，因此第 7 条流水线作为“呼叫线”被选择。

例子能接受来自指轮的 0 到 7 数字的 BCD 输入（即所有数字可以用 3 位表示），因此程序只使用输入 X000 到 X002。没有接收到指轮的输入时，0 被读入，将选择 0 号线作为“呼叫线”，接着就接通 Y000。

7.8 高速分离/分类

两种类型的产品从一条主传送带移过来，两种产品最后分到它们自己的产品传送带上。





器件	PC 软元件	说明
PC1	X000	计数器 C235 的输入
LINE1	Y001	1 线选择
LINE2	Y000	2线选择
	C235	瓶子计数器
	M000	一定数量的瓶子经过 PC1（小于计数器预定值）
	D000-	中间瓶数的数据存储
	Z	变址寄存器
（D）MOV-	FNC12	MOV 应用指令
（D）HSCS	FNC53	HSCS 应用指令
ADD	FNC20	ADD 应用指令

说明：

以计数为基础的分类系统—对本例来说，当 X005 接通时，数据移入从 D000 起始的双字寄存器。用一个变址寄存器“Z”对数据编页码。根据计数处理的某种判断（可能是 C235 计数结束），给出的数据（数据栈）和变址寄存器都需要复位。源数据可以来自于许多地方：数据可能是从制造产品并把产品放置在主传送带上的机器上传送过来的。输入 X013 使计数/分离系统复位。

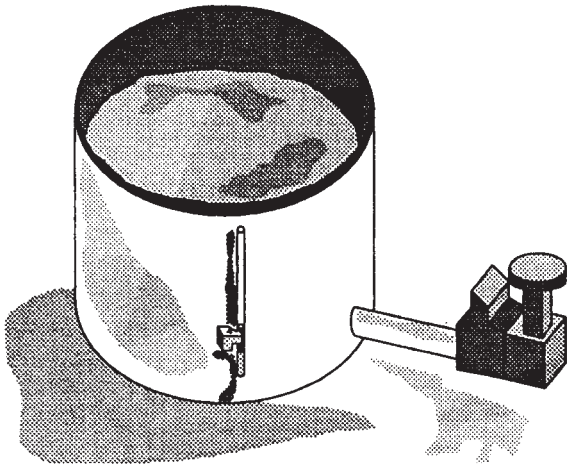
程序工作，使一定数量的产品传送到特定产品线。光电管 PC1（X000）提供计数输入给 C235，当达到预置值时，程序就知道当前一批产品“X”已传送到专门的产品线上，此时，门切换到第二条产品线。因为只有两种产品，而且其中一种不在主传送带上，则现有的产品一定是第二种。当前一批产品分离完，处理过程就继续在两条产品线间切换。

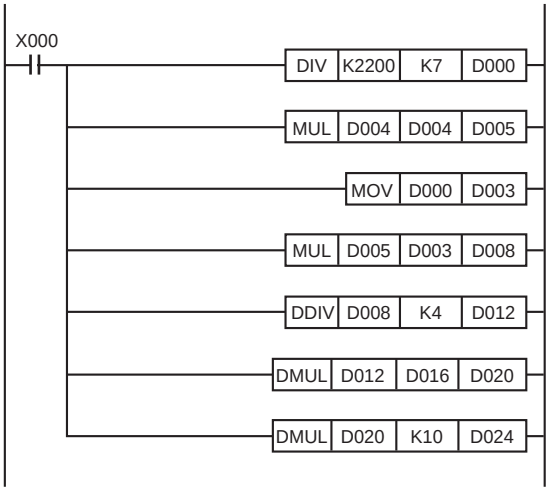
工程运算

经常地，一个过程或者最后归于一个计算值，或者要求一个值去得到某个内部设定参数。使用简单的加、减、乘和除数学工具，能完成大多数计算。有人会说：缺少小数或浮点数会严重妨碍用户工作，因为用整数处理的情况是很少见的。但没问题，它能得到许多数学操作的精确答案。不想对此说些什么，请继续读，自己去看……

8.1 最佳角度

运算时，很少看到结果值、中间的或最终的值等于完整的数（整数）。起初看起来可编程控制器不能用来执行精确运算。事实不是这样，采用比例和单位操作可以得到高精度的答案。





器件	PC 软元件	说明
	X000	计算角度
	D010	经纬仪的基准高度
水平线	D012	以毫米为单位的测试标尺的水平线
	D015	基准和实际标尺间的高度差
距离	D020	测试标尺和经纬仪间的距离，以米为单位
	D025	所求角度的正切值
SUB	FNC21	SUB 应用指令
DIV	FNC23	DIV 应用指令

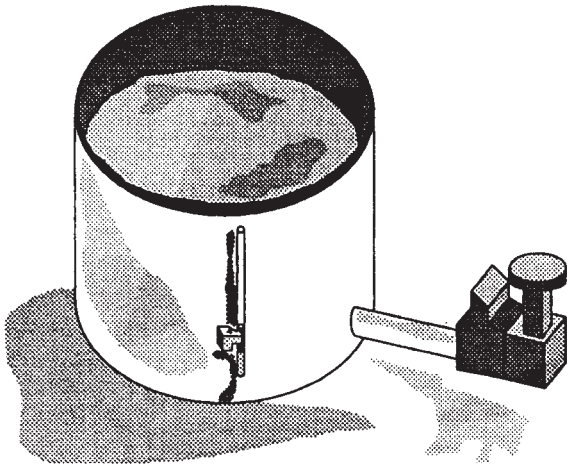
说明：

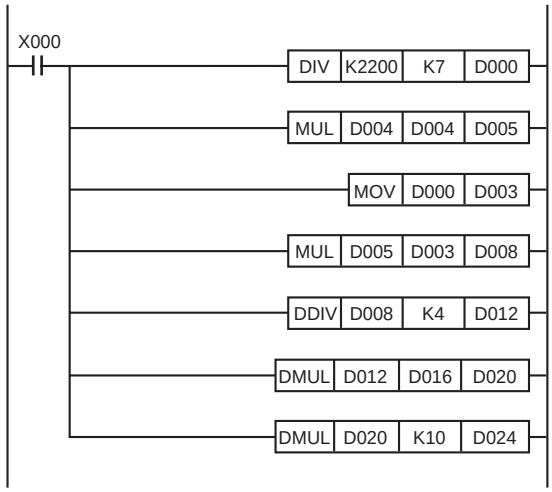
本节给出的例子显示了一个用以测量两点间地平面变化的经纬仪。计算实际上是已知三角形的两个直角边，求得斜边与相邻角度。

经纬仪和标尺的距离已知（邻边），基准位和当前位也已知（所以对边也能计算出）。这意味着所要求的角度的正切可计算出。为了得到一个合理的精确度，输入参数按比例给出，基准位和实际位以毫米为单位输入，经纬仪和标尺的距离以米为单位输入。在进行“反正切”运算前，所计算的正切需要除以100，这种方法的结果误差限制在近似0.5度。

8.2 使用基本指令的高级运算

下面程序计算经过管子的流量。管子直径必须以 mm 为单位，流速必须以 m/sec 为单位给出，答案以 mm^3/sec为单位，精确到小数后第 2 位。





器件	PC 软元件	说明
DIV	FNC23	DIV 应用指令
MUL	FNC22	MUL 应用指令
MOV	FNC12	MOV 应用指令
D（MUL/DIV）		双字操作的前缀
	D000	计算 π 的操作结果 - 16 位数据
	D003	π 值的复制 - 16 位数据
	D004	管子直径 - 以 mm 为单位 - 16位数据
	D005，006	管子直径平方的结果 - 32 位数据
	D008，009	管子直径平方和 π 相乘的结果 - 32 位数据
	D012，013，014，015	管子截面积的最后结果 - 64 位数据
	D016，017	流速 - 以 m/sec 为单位 - 32 位数据
	D020，021	流速×时间 - 32 位数据
	D024，025，026，027	最后答案 - 流量 以mm^3/sec 为单位 - 64位数据

说明：

本例说明了不用浮点数是如何做精确运算的。程序计算通过管子的流量。前面提到的两个参数必须输入到确定寄存器中。为了得到流量，管子的截面积与流速相乘。计算圆面积要求使用 π，因为没有现成的参数来表示它，使用古老的数学技巧，22 除以 7 得到一个相当接近的分数。注意本例中 22 已被 2200 代替，其原因下面会解释。

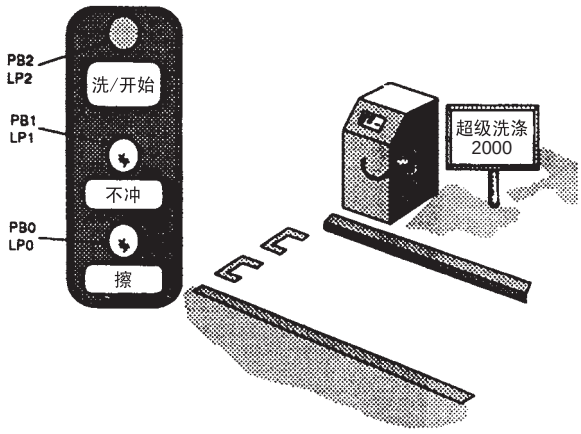
管子的直径进行平方，即自己与自己相乘，接着乘以前面得到的 π 值，再除以 4 就得到管子的面积。

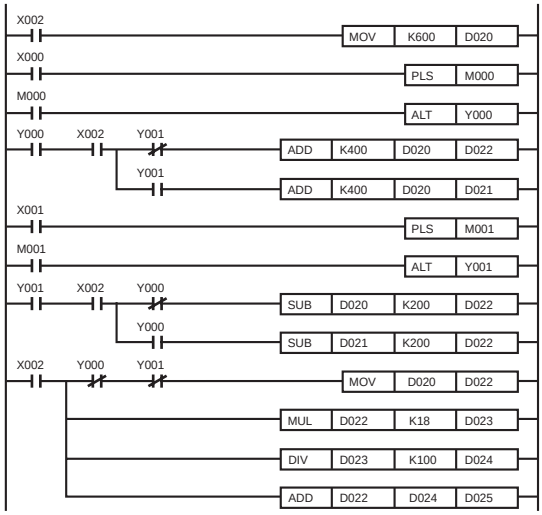
接着对所得面积乘以流速，最后一步是对答案乘以 10，从而给出以 mm^3/sec 为单位的流量。

对 22乘以 100 和对最终结果乘以 10 的原因是为了校正单位，即计算中 mm 和 m 都使用到，但从计算目的来说，要求转换到同一单位。不对流速乘 1000，而是增大 π 的计算值，这样可得到一个更精确的结果，达到小数点后 2 位的最终精度。

8.3 服务收费

这个例子说明一个收费系统是如何与选择的服务相联系的，最后总数可以以所要求的任意形式给出，如本例甚至还加入了税费。





器件	PC 软元件	说明
PB0	X000	要求擦服务
PB1	X001	不要求冲
PB2	X002	选择有适当服务更改的洗程序
LP0	Y000	指示灯-擦
LP1	Y001	指示灯-不冲
LP2		在程序别的部分中
MOV	FNC12	MOV 应用指令
ADD	FNC20	ADD 应用指令
SUB	FNC21	SUB 应用指令
MUL	FNC22	MUL 应用指令
DIV	FNC23	DIV 应用指令
ALT	FNC66	ALT 应用指令
	D020	基本冲和洗价格
	D022	更改的服务价格
	D024	计算税费
	D025	总费用

说明：

现有三种服务，用户费用决定于哪一个服务被选择。洗服务常被选择，但也可改为另两项服务。基本费用对应于洗程序，再根据选择的其它服务，增加或减少费用。

使用按钮 PB0 和 PB1 选择不同的服务。选择状态由按钮里的灯的 ON 或 OFF 表示，即如果 PB1 按下，LP1 使灯亮（ON），接着基本程序将进行，不带冲功能。当已经选择了所要的擦/冲时，按下洗/开始按钮，接着计算所选服务的费用。实际程序的最后 4 行使用一个简单的乘、除和加顺序对最终费用加 18% 的税费。费用单见下：

472-不带洗服务的清洗

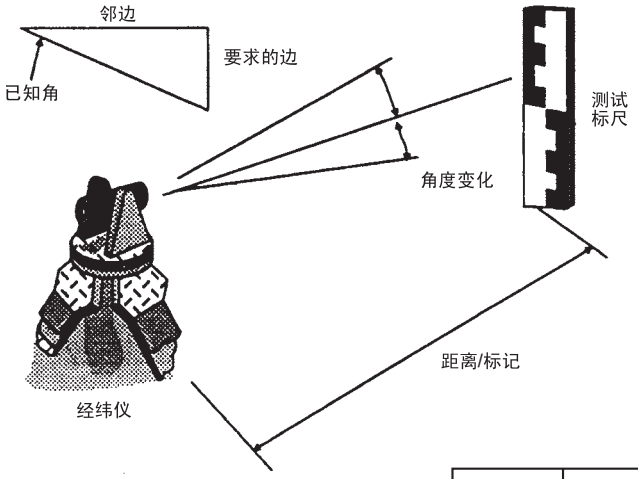
708-基本清洗

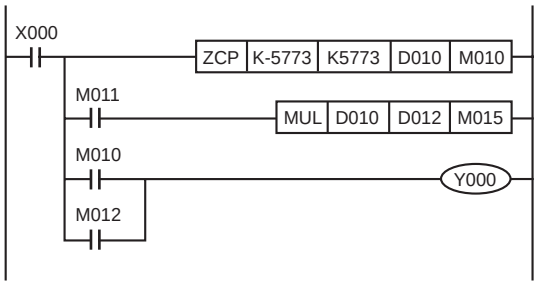
944-不带冲服务，但带洗服务的清洗

1180-全项服务，即冲、洗、擦。

8.4 保持检测中的数据值

计算常是通用的，即它们能处理所有数据输入情况。然后常发现只对很小范围的数据有实际应用。本例从一个预定操作范围中接收数据，在此期间，删除不想要的数据并标记错误。





器件	PC 软元件	说明
角度	D010	围住的角度 - 作为正切值输入 （在输入前需要乘以 10^4）
距离	D012	经纬仪与测试标尺间的距离， 以米为单位
	D015	地平面变化的 32 位的答案， （答案需要除以 10^4）
	M010 - M012	超过范围的正切值输入
	M011	可接收的正切值
ZCP	FNC11	ZCP 应用指令
MUL	FNC22	MUL 应用指令

说明：

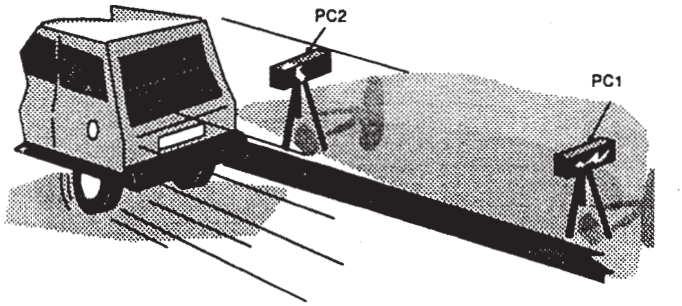
经纬仪用来测量从基准位置到实际测量位置的角度差。经纬仪和测试标尺间的距离已知，因此能计算出地平面的变化。计算是相当简单的，因为它只是一个简单的三角运算。测量角的正切输入到数据寄存器 D010。输入值作 10000 倍的比例放大，因此 20 度的正切值 0.3640，但 3640 被存入数据寄存器 D010。D010的值不应超过-30 到+30 度的范围。用这些角度的正切作比较：30 度的正切为 0.5774。如果输入值在限度范围内，则标志 M011 有效，做对边计算。如果输入值在范围外，则输出 Y000 被标志 M010 和 M012 激活，表示有一个错误。

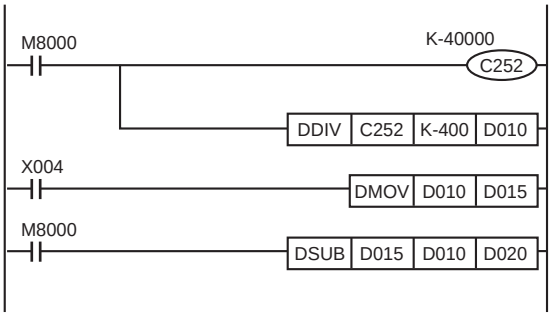
接下来，三角形的邻边距离（从经纬仪到标尺）以米为单位被输入（数据寄存器 D012）。

完成一个计算时，答案存入一个 32 位数据寄存器（D015 和 D016）中。显示的数据需要除以 10000，从而除去原来的放大比例。给出的答案将精确到对边真实长度的近 1 mm 范围内。

8.5 一个测速器

并不总是要求单单检测动作本身，在有的时候，从多个检测器来的输入信息会改变数据的整个意思；在有的时候，输入数据被解释或处理，给人一个全新的感觉，使它看起来象是一段不重要的信息。





器件	PC 软元件	说明
PC1	X000	开始计时
PC2	X001	停止计时
	X004	计算速度
	TXXX	10 微秒定时器
	D010	在 PC1 和 PC2 之间运动所用的时间
	D012（D013）	PC1 和 PC2 之间的实际距离，以米为单位
	D020，021	PC1 和 PC2 之间的运动速度 （除以 100，得到 Km/h 单位）

说明：

本节例子说明一个简单的时间测量是如何转变为一个有更多含义的速度/速率指示的。

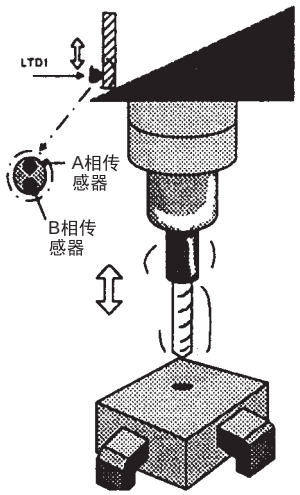
当一辆车经过光电管 PC1 和 PC2 时，从它挡住第一条 PC1 射线到挡住第二条 PC2 射线的时间段被计时。设备操作者知道两固定点间的距离，此数据输入到数据寄存器 D012（以米为单位）。

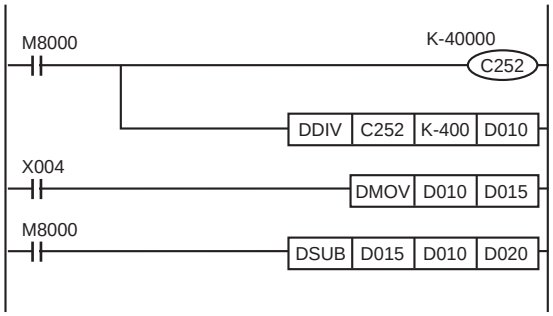
操作者有检测车速或不检测的选择权，这通过按下按钮，接通输入 X004 来实现。

当车经过检测，它的速度被计算，以千米每小时为单位。计算期间，对其值做一定的比例增大，这是为了提高结果的精确度：应该记住，这里没有使用浮点值或浮点指令。最后计算结果存放在数据寄存器 D020，D021（32位寄存器）中。此答案为了以 Km/h 为单位，需要除以 100。

8.6 相位计数器

常有一种错误观点，认为所有编码器应在轴向面上工作。尽管有很多线性编码器时常被用来确认定位运动，而在所有定位运动中，有正向的和反向的运动。处理此问题的最简单方法是应该使用二相计数器。





器件	PC 软元件	说明
A - 相	X000	计数器 C252 的 A 相输入
B - 相	X001	计数器 C252 的 B 相输入
	X003	复位计数器 C252（自动分配）
	X004	当前位置作为数据存储
	M8000	PC 运行常闭触点
	C252	2 相计数器（32 位）
	D010，011	当前位置 - 32 位数据
	D015，016	数据位置 - 32 位
DDIV	FNC23	DDIV 应用指令
DMOV	FNC12	DMOV 应用指令
DSUB	FNC21	DSUB 应用指令

说明：

二相计数器可以描述为是一对计数器，每一个都独立检测，但只有在被检测信号同时出现时，才真正计数。这听起来相当复杂，但实际上很简单，下面说明这一点：

A 相通道检测输入 - A 相现在起作用。

B 相通道检测输入 - B 相现在起作用。

A 相通道检测消失 - A 相现在不起作用。

B 相通道检测消失 - B 相现在不起作用。

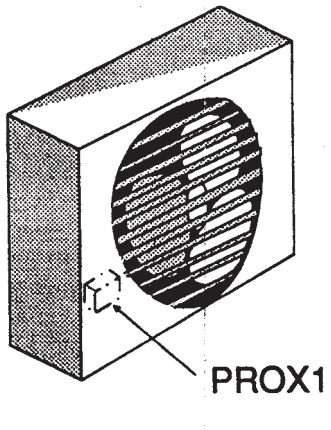
上面序列将引起增计数，事件序列不断重复计数器连续增计数。如果 A 相和 B 相相序交换，则引起减计数。使用一个如上图所示的检测器，很容易检测发生的运动方向。

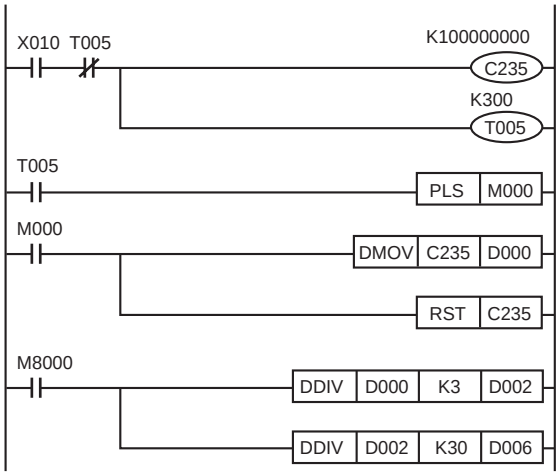
程序得到计数值，并用 -400 除，得到被测量单元的距离。（这是一个孔径计算，依赖于所用的线性编码器）

如果，在这种钻机的例子中要求钻孔的准确深度。接着输入 X004 建立一个数据点。此操作可在任何时刻执行，不过经常地，此操作在钻头刚接触被钻孔的表面时开始进行。保存在数据寄存器 D020，D021 中的距离与此数据点有关，并确定已做的运动有多远，什么方向。由此，钻深就能知道了。

8.7 高速计数器检测速度

本程序计算风扇每秒旋转的圈数。





器件	PC 软元件	说明
PROX1	X000	检测旋转叶片的接近开关
	T005	计时检测时间
	C235	高速计数器，在 T005 定时期间， 对叶片经过 PROX1 的次数计数
（32 位数据 寄存器）	D000 D002 D004 D006 D008	被检测信号的数量 总共旋转次数 （除法操作，余数） 每秒旋转圈数 （除法操作，余数）
（D）MOV	FNC12	MOV 应用指令
（D）DIV	FNC23	DIV 应用指令

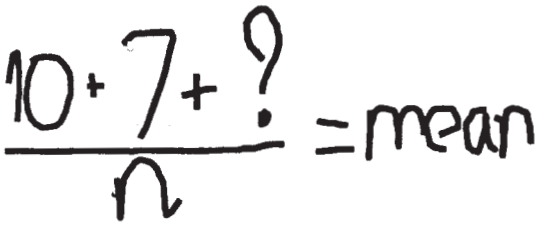
说明：

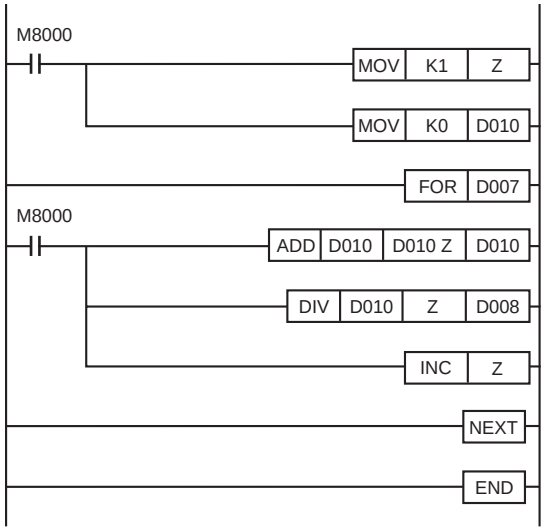
风扇有三个叶片，当它们各自经过接近开关 PROX1 时被检测到。此开关与驱动高速计数器 C235 的输入 X000 相连。因为要计算速度，计数过程必须在一个定时段内实现，因此，自关断定时器 T005 用来触发 C235 中的当前数据转移到数据寄存器 D001，D000（C235 是一个 32 位寄存器）。当数据转移结束，计数器复位，并且定时计数过程再次开始。其间，存于 D000 的新数据被 3 除。这是因为有 3 个叶片，而速度是要求整圈的圈数。除法运算后的答案存在D003，D002 中，除法的余数部分保存在寄存器 D005，D004 中。为了得到每秒旋转的速度，存在 D003，D002 中的答案必须要再除以 30：这是因为 T005 是 300 个 100 毫秒，即 30 秒。接着答案存在寄存器 D007，D006中，余数部分存在 D009，D008 中。

注：要求使用双字操作，因为来自计数器的初始数据是 32 位格式。应该注意有余数寄存器的除法操作。

8.8 求平均数

有时为了把最简单的数字问题转化成一种使用基本指令如 ADD，MUL，SUB 和 DIV 的形式，需要一些想像力。





器件	PC 软元件	说明
	M8000	PC 运行常闭触点
	Z	求平均数中所包括的元素个数的当前计数
	D010	求平均数中当前包括的所有元素的累积和
	D007	平均数计算中所包括元素的最大个数
	D008，D009	当前处理元素所计算的平均值
MOV	FNC12	MOV 应用指令
FOR	FNC8	FOR 应用指令
ADD	FNC20	ADD 应用指令
DIV	FNC23	DIV 应用指令
INC	FNC24	INC 应用指令
NEXT	FNC9	NEXT 应用指令

说明：

本节叙述的平均数计算程序例子是非常复杂的。要进行平均数计算的元素个数直接等于 FOR-NEXT 回路将运行的次数。本例中，这通过把一个值直接输入到数据寄存器 D007 来实现。

本例中，平均数处理的源数据放在首地址为 D011 的连续数据寄存器中，每次扫描将重作一次平均数值计算。这是因为 FOR-NEXT 回路照顾到源数据寄存器的递加。

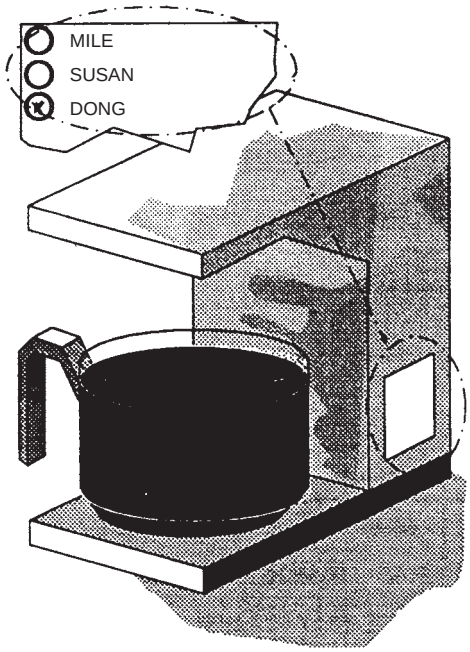
应注意的是，所算得的平均数（DIV 指令之后）会占用 2 个 16 位数据寄存器。一个包含所计算的整数部分，而另一个则包含余数的数据。

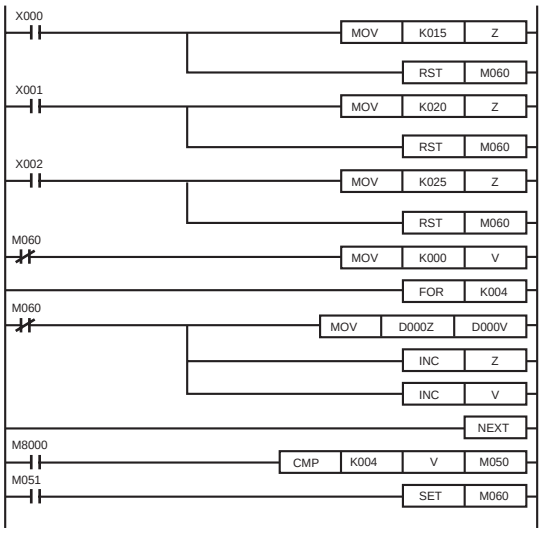
数据运算

可编程控制器有数据操作，但它能用来做什么呢？一些更高级的技术将在这里论述，范围从食谱操作到排时间表工作，及存贮控制。

9.1 使用食谱

为了冲出一杯咖啡需要一个食谱，即多少糖、牛奶、水、咖啡等。如果有几种咖啡，则必须有几种相应的食谱。





器件	PC 软元件	说明
MILE	X000	选择制作 MILE 喜爱的饮料的食谱
SUSAN	X001	选择制作 SUSAN 喜爱的饮料的食谱
DONG	X002	选择制作 DONG 喜爱的饮料的食谱
	D000 - 003	通用食谱工作区
	D015 - 018	MILE 的食谱
	D020 - D023	SUSAN 的食谱
	D025 - D028	DOUG 的食谱
	V，Z	变址寄存器
MOV	FNC12	MOV 应用指令
FOR	FNC8	FOR 应用指令
INC	FNC24	INC 应用指令
NEXT	FNC9	NEXT 应用指令
CMP	FNC10	CMP 应用指令

说明：

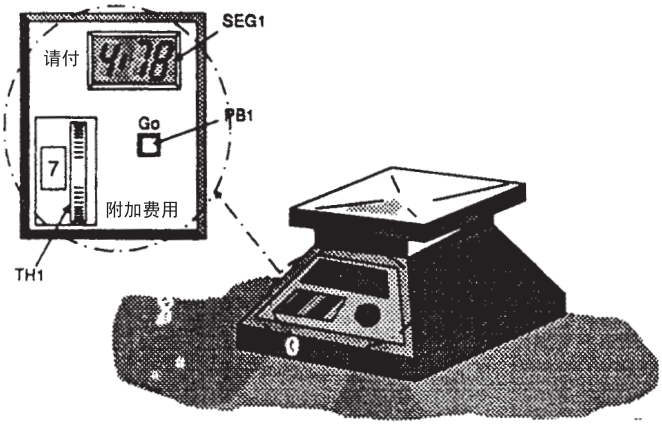
本节例子说明一个特殊的咖啡机，这种咖啡机能制作三种类型的咖啡，每一种类型可通过按相应按钮来选择。每种情况中所要求的制作咖啡动作是一样的，唯一的变化是：多少量和多长时间。简单地说，是所要求的食谱是什么？

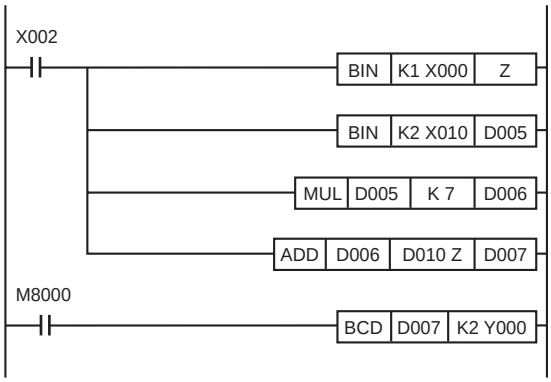
这个数据是纯数字的。选定所要的咖啡类型，相关食谱数据就载入一个工作区，即 D000 到 D003。

食谱数据可以在控制咖啡机程序的前面部分预先载入或输入。食谱数量只限于现有设备的数量。本节例子中，使用了3个有 4 个参数的食谱，但也很容易得到 10 个有 2 个参数的食谱或 20 个 12 个参数的食谱。

9.2 使用变址寄存器查找数据

这些自动台秤对邮包称重并计算费用。费用分为 10 个等级区，从而能计算远程的附加费用。





器件	PC 软元件	说明
SEG1	K2Y000	7 段显示的 BCD 输出 - 需付的钱
TH1	K1X000	来自指轮的 BCD 输入 - 区间附加费用
PB1	X002	启动按钮，激活称重和计算程序
	K2X010	来自称量装置的重量输入
	D005	读重量
BIN	FNC19	BIN 应用指令
BCD	FNC18	BCD 应用指令
MUL	FNC22	MUL 应用指令
ADD	FNC20	ADD 应用指令

说明：

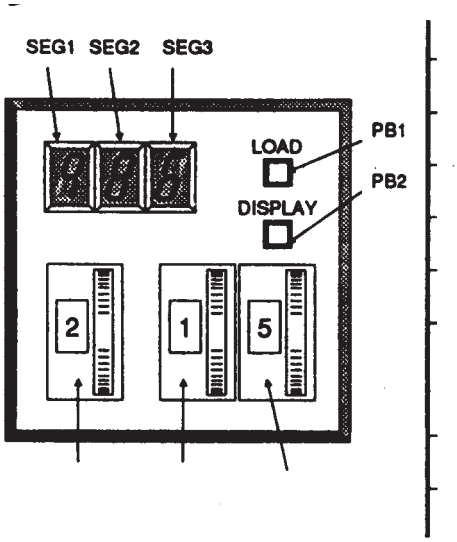
天平秤一般显示称重物的最终费用，这意味着一个邮包能很快地被称重并移走。当用户把要送出/称重的邮包放在秤上，并按下“GO”按钮 PB1，系统开始工作。这时，重量被读入到可编程控制器中并用以计算邮包的总费用，这通过用一个数值乘以重量来实现，这里数值为 7。

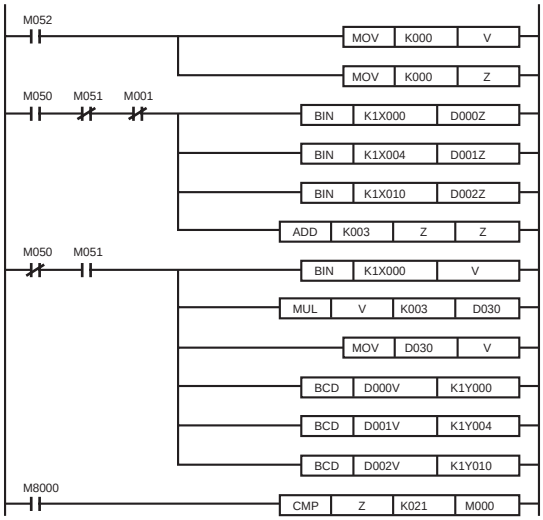
然而，这个费用有由指轮输入 TH1 提供的第二个变量。通过指轮得到的 9 个附加费用会改变总费用。在本程序中，实际上指轮提供了一个补偿量，它表示哪个数据寄存器所包含的附加费用将加到总邮包费用中。

补偿过程的结构包括把指轮输入移入变址寄存器 Z，这在 ADD 指令中的 D010Z 项中确定。Z 提供一个首地址的偏移量（本例首地址为 D010）。

9.3 数据保存和取出

可以被访问的数据才是有用的。可访问性意味着数据能被操作者使用或指能在程序中使用数据。





器件	PC 软元件	说明
SEG1	Y000 - 003	显示保存在第一寄存器的数据
SEG2	Y004 - 007	显示保存在第二寄存器的数据
SEG3	Y010 - 013	显示保存在第三寄存器的数据
PB1	间接 M050	新数据载入数据栈
PB2	间接 M051	取出数据的显示
指轮1	X000 - 003	双重功能 - 载入到第一寄存器的源数据 要取出数据的位置
指轮2	X004 - 007	载入到第二寄存器的源数据
指轮3	X010 - 013	载入到第三寄存器的源数据
	M001	数据栈满（用户可定义的）
	M052	复位变址寄存器
	M8000	PC 运行常闭触点
MOV	FNC12	MOV 应用指令
BIN	FNC19	BIN 应用指令
ADD	FNC20	ADD 应用指令
MUL	FNC22	MUL 应用指令
BCD	FNC18	BCD 应用指令
CMP	FNC10	CMP 应用指令

说明：

本例阐述了使用变址寄存器作为控制/定位一个数据栈中数据的一种方法。

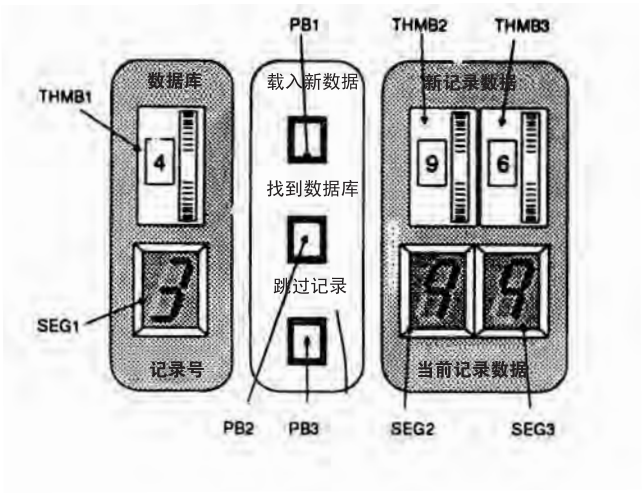
数据在三个指轮上输入。当按下“LOAD”按钮 PB1 时，指轮数据被读取并存储在三个数据位置上（每个数据对应一个位置）。本例看起来总把数据（使用 BIN 指令）存储在相同的三个数据寄存器中，即 D000、D001 和 D002。不过，如果仔细地观察该指令，可看到每个确定的数据位置都带着一个“Z”。Z 实际上是一个代表偏量的变址寄存器，系统中更为人所知的是“首地址和指针”。在每次存储数据的操作后，Z 的值被修正，以使它准备在下一个可用空白区中存储别的数据。本例中，数据栈的大小限定为 21 个“数据库”，可以这么实现；用最后的比较指令设置位 M001 为 ON，并使数据存贮程序不运行。

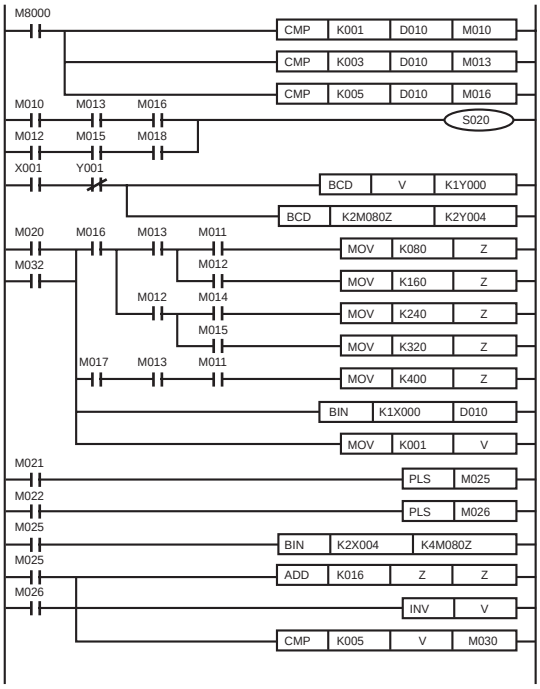
数据的取出以类似方式进行。这一次，使用了变址寄存器 V。当 V 读取 3 个数据寄存器的每个新数据块时，它被乘以 3，从而得到正确间隔。V 和 Z 都以相同方式使用。

注：V 和 Z 可以作为 16 位寄存器单独使用，或组合在一起作为一个 32 位寄存器。对于乘或除操作，V 不能作为目标操作数（本例把 V 乘积放入 D030，再把它移回 V）。

9.4 使用位元件来保持字数据

在某些情况下，程序所要求的字存储元件可能会超过可编程控制器所能提供的。如果所要求的数目不是太大，大概 25 个 16 位字数据，则可以使用辅助位元件。





器件	PC 软元件	说明
PB1	间接 M021	按钮 - 载入新数据
PB2	间接 M020	按钮 - 找到数据库
PB3	间接 M022	按钮 - 跳过这个记录
THMB1	K1X000	选择数据库（只能是 1 - 5）
THMB2，TNMB3	K2X004	将载入数据库记录的新数据
SEG1	K1Y000	将载入的记录号
SEG2，SEG3	K2Y004	当前选择的记录数据
	M8000	PC 运行常闭触点
	S020	无效的数据库号
	D010	选择的数据库号
CMP	FNC10	CMP 应用指令
BCD	FNC18	BCD 应用指令
BIN	FNC19	BIN 应用指令
ADD	FNC20	ADD 应用指令
MOV	FNC12	MOV 应用指令
INC	FNC24	INC 应用指令

说明：

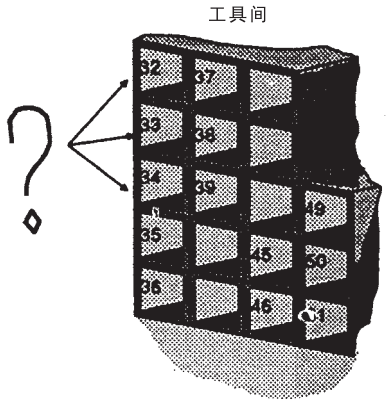
本例使用位元件存储器调用 16 位字数据。在 THB1 上拨入所需要的数据库号，可选择 5 个数据库中的一个。 当期望值出现时，按下按钮 PB2，显示第一个记录，SEG1 显示记录号，SEG2 和 SEG3 则显示记录内容，本例允许每个数据库有 5 个记录。

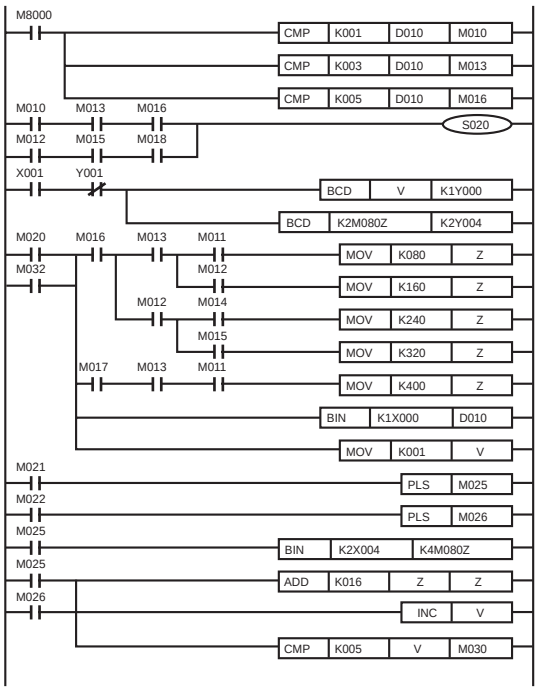
如果要输入数据，则数据值通过指轮 2 和 3 来设定。当按下按钮 PB1 时，数据被载入当前记录中，程序进入下一个记录。如果当前记录不是所期望的记录，则按下按钮 PB3，使程序指向下一个可用记录。当到达记录 5 后，程序下一次自动复位至该数据库的记录 1。

要注意的编程要点是获得 7 个不同结果的 3 个比较指令的使用，另一个是 Z 和 V 变址用以隔离和追踪当前/已存储数据的方法。

9.5 先入先出堆栈控制

FIFO，即先入先出，这是一个有很多实际应用的简单思想。其中常用到的一种应用是堆栈控制系统。本例说明了 FIFO 是如何应用于一个大型工厂的工具间的。





器件	PC 软元件	说明
	X000	载入新记录值
	X001	读取第一个记录值
	X004（-X023）	新记录值输入 - 可能 4 有个指轮
	Z	变址寄存器 - 保持总记录数在限定范围内
	V	变址寄存器 - 用以计算从首地址起的第一组记录的偏离量
	M020	FIFO 栈的首地址
	D010	从第一个记录位置中读取的数据
	M8002	PC 运行初始脉冲
	M8000	PC 运行常闭触点
MOV	FNC12	MOV 应用指令
SFTL	FNC35	SFTL 应用指令
INC	FNC24	INC 应用指令
SUB	FNC21	SUB 应用指令
MUL	FNC22	MUL 应用指令
DEC	FNC25	DEC 应用指令
CMP	FNC10	CMP 应用指令

说明：

本例说明了可编程控制器是如何应用在不同于自动设备的开和关的场合。背景是一个标准工具间，其作用是为一个自动化工厂打磨工具。当一个工具送到工具间时，它被增列到计算机清单上，这份清单保证所有工具都被打磨，并且按接收的次序对它们一一处理。

工具数据通过输入 X004 至 X023 被输入。这些 16 位数据可由工具间的操作者通过指轮开关输入，或者直接从 CNC 加工中心读取。当数据输入时，输入 X000 进行初始化，对来自于上面所提到的源数据进行读操作 FIFO 栈自动更新。

当工具间的操作者决定对一个新工具进行打磨时，按下输入 X001，读取 FIFO 第一个记录中存储的数据，并把它移至数据寄存器 D010，当数据存入寄存器 D010 后，该数据被处理，向用户显示下一个要处理的工具，与工具打磨相关的数据也可能被送到打磨机器中：如工具长度、半径、材料等。当一个记录从 FIFO 中读出后其位置被删除。

本例程序使用 M 线圈元件来存储 20 个 16 位的记录，这是为什么 CMP 指令判断到超过 K20（20 个记录）或小于 K0 时引起出错的原因。

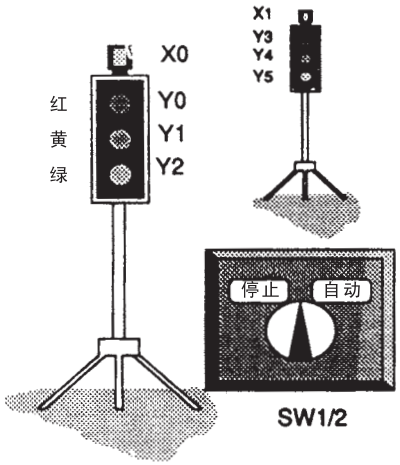
数据读取看似很复杂，实际上相当简单，难的是指针/变址 V 寻找数据的计算，因此，使用了 MOV，SUB，MUL 指令。

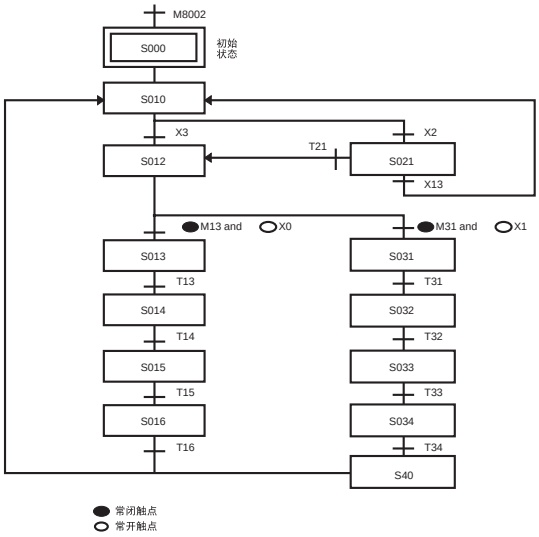
较复杂问题

下面 12 个应用较为复杂，程序也较长。可能的话，你可以提出一个问题的解法，再与我们的解法进行比较。

10.1 交通灯

一般而言当提出一个问题时，解法看似相当简单。然而，当开始建立必要的安全性和操作特性时，会马上使那个简单问题成为一个“噩梦”，比如轻便交通信号灯……





器件	PC 软元件	说明
SW1	X002	停止模式选择
SW2	X003	自动模式选择
	X000	汽车检测 1
	Y000	红灯设置 1
	Y001	黄灯设置 1
	Y002	绿灯设置 1
	X001	汽车检测 2
	Y003	红灯设置 2
	Y004	黄灯设置 2
	Y005	绿灯设置 2
ZRST	FNC40	ZRST 应用指令

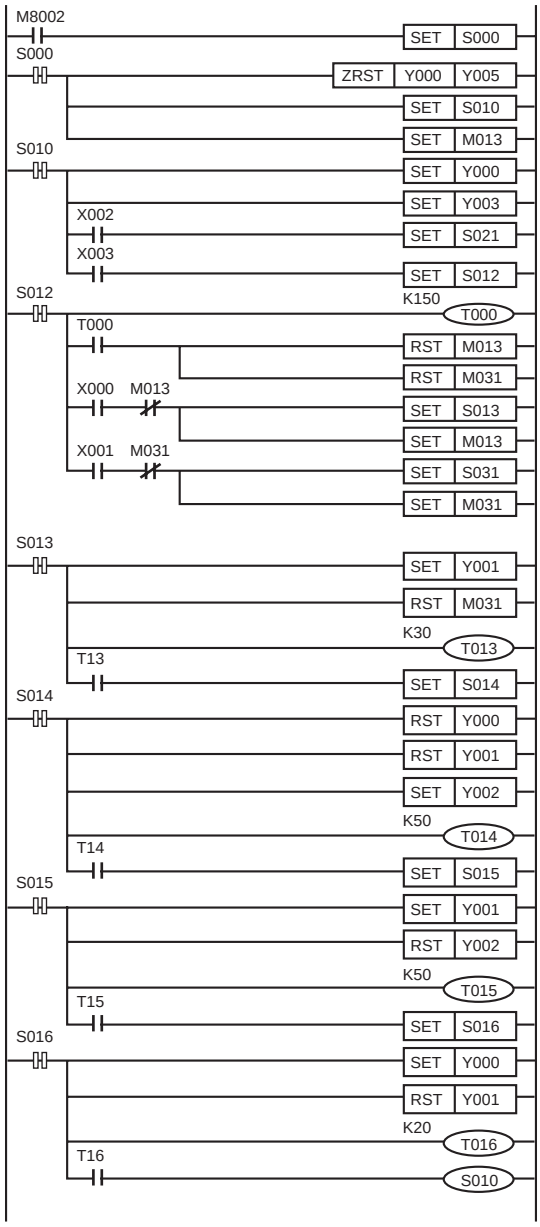
说明：

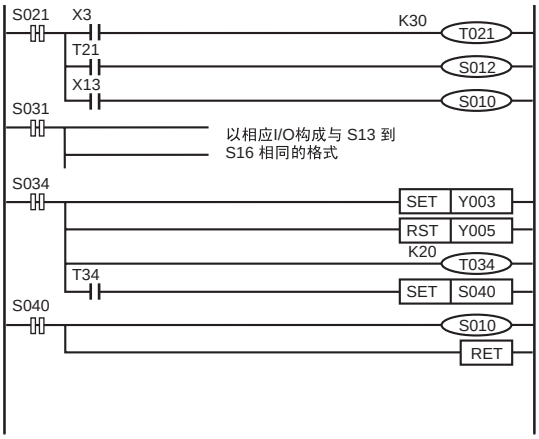
交通信号灯的工作在世界各地略有不同，不过其基本原理是一样的：

- 司机原地不动等待，直到一条安全路线出现。
- 对正在穿越马路的车辆，为了避开车辆相撞或障碍物，允许短暂停顿。
- 当所有可能危险路线暂停时，允许原地等待的司机前行。

对一盏交通灯来说，这不是个复杂的过程。但是，它们常常是多路一起出现的！所以要求安全互锁和控制。本例中，情况更为复杂化，正在使用这组灯的工作人员在执行某项工作或某个操作时，任何车辆经过都会导致灾难，所以此时允许工作人员停止所有方向的交通，此时最简单的方法是用步进梯形图或 STL 编程。要考虑每个事件并把它转换为一个 STL 步，确保没有遗漏任何步，因为每一个 STL 步都可以认为是一个“微型的”完整程序，编程任务就变得简单多了。

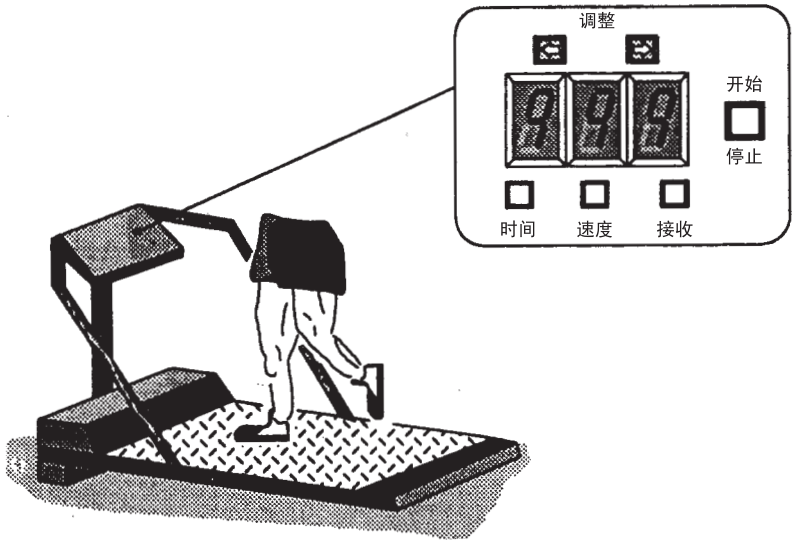
程序：


M8002
SET S000
S000
ZRST Y000 Y005
SET S010
SET M013
S010
SET Y000
SET Y003
X002
SET S021
X003
SET S012
S012
K150
T000
T000
RST M013
RST M031
X000 M013
SET S013
SET M013
X001 M031
SET S031
SET M031
S013
SET Y001
RST M031
K30
T013
T13
SET S014
S014
RST Y000
RST Y001
SET Y002
K50
T014
T14
SET S015
S015
SET Y001
RST Y002
K50
T015
T15
SET S016
S016
SET Y000
RST Y001
K20
T016
T16
S010


S021 X3
K30
T021
T21
S012
X13
S010
S031
以相应I/O构成与 S13 到
S16 相同的格式
S034
SET Y003
RST Y005
K20
T034
T34
SET S040
S040
S010
RET

10.2 匹配

可编程控制器的使用范围只受下面思想的限制：对要求一台复杂控制器的任何工程，使用 PC 更便宜、更可靠。



器件	PC 软元件	说明
时间	X001	时间设置按钮
速度	X002	速度设置按钮
接收	X003	接收所有按钮
开始/停止	X004	跑步机开始/停止按钮
⇐	X005	加设置按钮
⇒	X006	减设置按钮
显示	K3Y000	显示当前设置数据
	D010	时间设置寄存器
	D011	速度设置寄存器
	D020	控制寄存器-时间
	D021	控制寄存器-速度
MOV	FNC12	MOV 应用指令
ADD	FNC20	ADD 应用指令
SUB	FNC21	SUB 应用指令
ZCP	FNC11	ZCP 应用指令
BCD	FNC18	BCD 应用指令
ALT	FNC66	ALT 应用指令

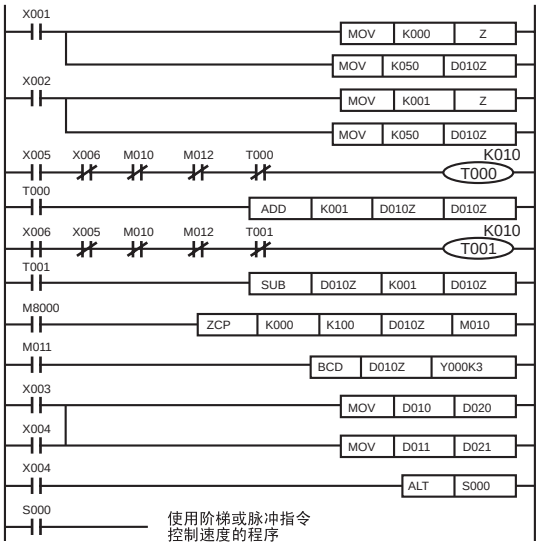
说明：

可编程控制器对执行重复性任务是理想的，那么有什么比在跑步机上跑步更有重复性的呢？通过使用数据处理指令，本例允许在跑步机的小控制台上设定时间和速度。当输入数据被接收，按下“开始/停止”按钮起动运行过程，再按一下“开始/停止”按钮则停止此过程。

速度和时间的细节已被省略，以便能更详细地观察数据输入方法，为了设定时间或速度，跑步机选择参数，接着使用两个箭头键增加或减小数值，直到所期望值，按“接受”钮确认这个数据，下一个参数以相似方式被选择和处理。当跑步机对这些设置值满意了，剩下要做的是按“开始停止”按钮。

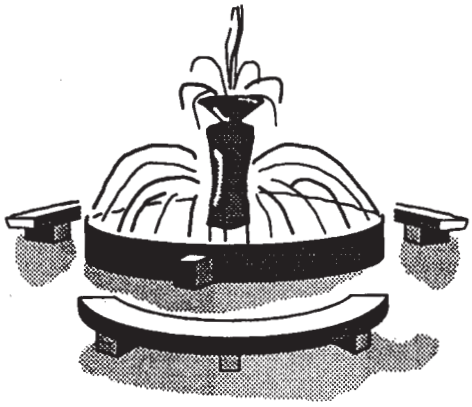
程序的改进可包括使速度从静止上升到希望的运行速度，并在任务结束后再下降到零，也可加入一个滚动屏幕。可能的改进内容是无穷的……。

程序：



10.3 有趣的喷泉

为了得到一个解决方法，有时需要反向地思考问题。通过每个喷嘴使用一个独立泵，可能很容易地解决下面的喷泉问题，泵压变化会使喷泉高低变化。但，这真的是最有效的方法吗？。



器件	PC 软元件	说明
	X010	起动系统
	Y000	脉冲输出至选定位置的阀门喷嘴
	Y005	选择喷嘴阀门“A”
	Y006	选择喷嘴阀门“B”
	Y007	选择喷嘴阀门“C”
	Y010	选择喷嘴阀门“D”
	D10 - 13	第一个半小时的喷嘴设置
	D15 - 18	第二个半小时的喷嘴设置
	M030 - M033	“Z”内容的位模式表示
	M020 - M022 M025 - M027	比较指令的标志位
	C000 - C001	用以对 M8014 计数的计数器– 30 次计数=1/2 小时
	M8014	1分时钟脉冲
	M8029	特殊 M 线圈 - 指令完成标志
MOV	FNC12	MOV 应用指令
PLSY	FNC57	PLSY 应用指令
ADD	FNC20	ADD 应用指令
CMP	FNC10	CMP 应用指令

说明：

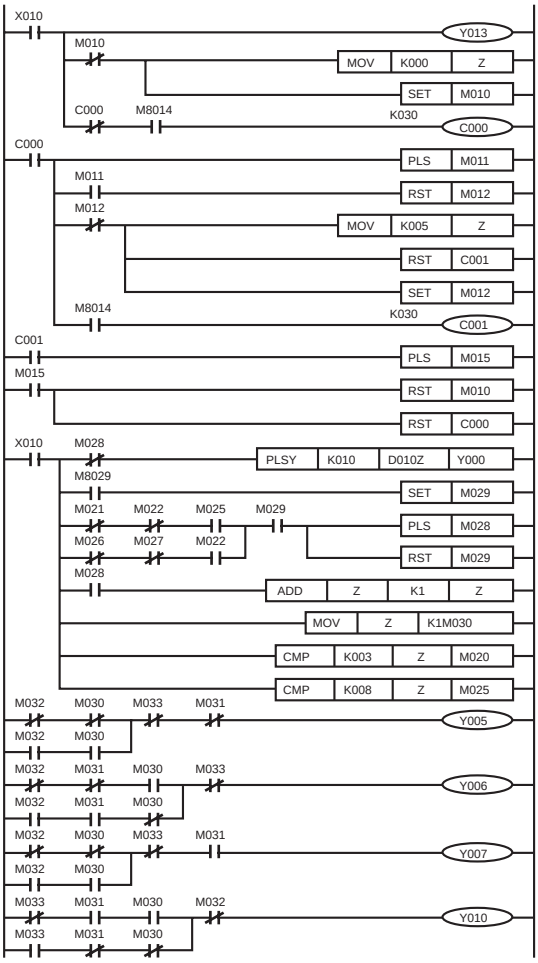
本例目的是控制和改变喷泉的高低。为了做到这一点，从不同喷嘴中喷出的水需要变化以创造一种预期的模式。一个简单方法是使用多路泵并且控制泵运行的压力。然而，压力是力/面积的函数。如果压力保持为常数，而流过的水面积变化，力和由此导致的水“喷射”距离会变化，其方式与改变泵压力引起的变化相同。

从一个不同的角度来考虑这个问题，就能找到一个完全不同的想法。下面给出的程序以一个工作程序形式说明这个想法。喷嘴每半个小时改变它的模式，一个通过程序扫描执行 PLSY 指令的回路控制每组阀门。

两个数据寄存器库被处理，从而得到两个不同的喷泉模式。每个数据寄存器库包含所要求的用 PLSY 指令输出的脉冲数，目的是设定选择的阀门在正确位置上。

因为数据寄存器通过变址寄存器 Z 的递增来选择，这种方法也用于独立地选择每个阀门。是这么实现的：移动 Z 中的数据进入位模式，接着有效位元件用来驱动所选择的阀门，也可使用多路比较指令来完成。通过接通阀门选择输出 Y5 到 Y10中的一个，PLSY指令（Y000）的单脉冲输出依次重定向至每个阀门。

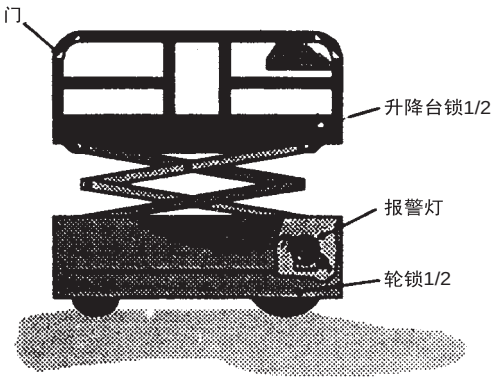
程序：

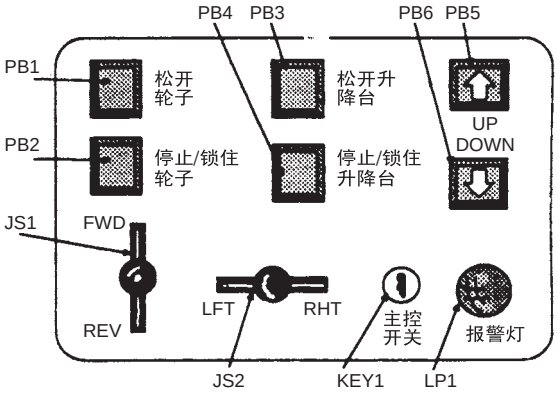


10.4 维护电梯

理想的编程技术是建立许多标准程序，这些程序能转换使用到别的程序中。这就象把微型程序压缩在一起，并把它们放入一个更大的程序中，这也能使每个较小的程序在不同时候被多次调用。这是 STL 编程的最大优点之一。

器件	PC 软元件	说明
KEY1	X006	主控制钥匙开关
PB1	X016	松开轮锁
PB2	X007	扣住轮锁/停止
PB3	X017	松开升降锁
PB4	X005	扣住升降锁/停止
PB5	X010	升高升降台
PB6	X011	降低升降台
LP1	Y005	错误/危险灯
JS1 - FWD	X012	向前移动升降车
JS1 - REV	X015	向后移动升降车
JS2 - LFT	X013	升降车转向左边
JS2 - RHT	X014	升降车转向右边
报警灯	Y010，011	双重目的的警示灯
门	X000	检查平台上的安全门
	Y001，002	锁住升降平台
	Y003，004	锁住车轮
	Y012	正向起动车子电机
	Y015	反向起动车子电机
	Y013	使车子转向左
	Y014	使车子转向右
	Y010	起动电机 - 升高平台
	Y011	起动电机 - 降低平台
	X001，002	检查升降机锁住
	X003，004	检查车轮锁住
	M050	操作锁定



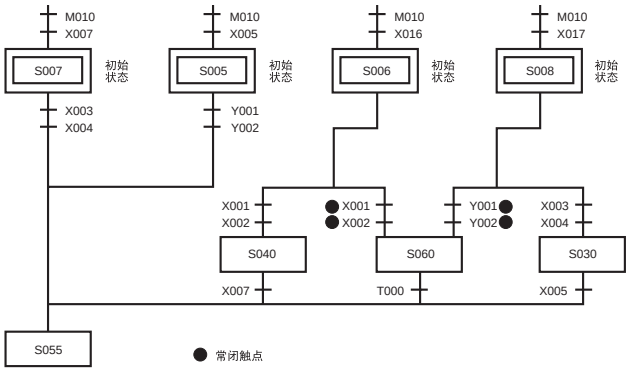


说明：

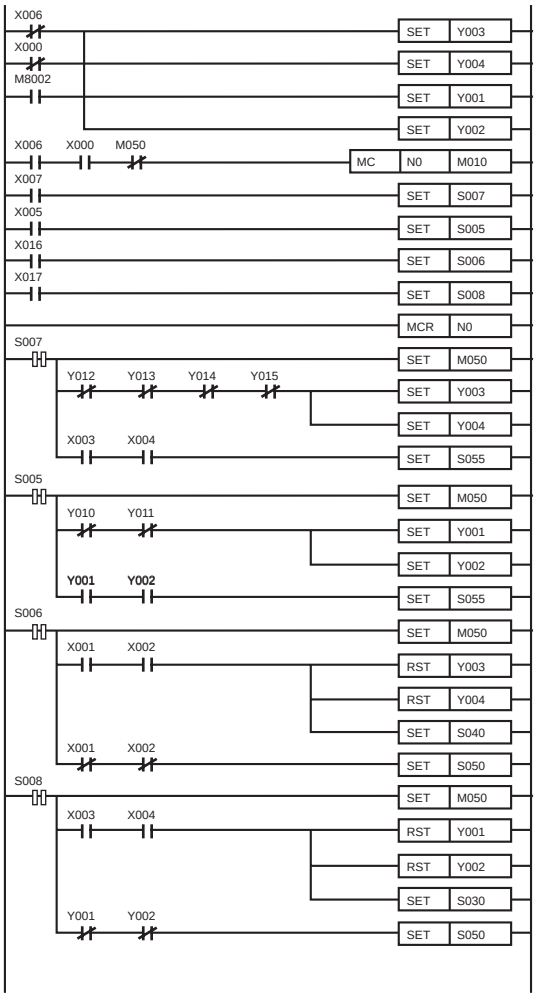
本例围绕一台移动平台编写。每个操作或每方面操作写入到它自己的 STL 步中。这确实给编程者提供了一个模板程序，此程序能传送入别的程序中。而且这种做法提供了安全性，并鼓励了通用程序的重复合用。下页所示程序说明了这点。对于平台上移，其基本条件之一是当平台升起，它就不能在任何其他平面上移动。可以从下面方式看到这一点：升高和降低平台（STL 步 S8）的程序完全孤立于在 XY 平面上移动平台（STL 步 S6）的程序，当其中一种模式有效，另一模式就不能工作。

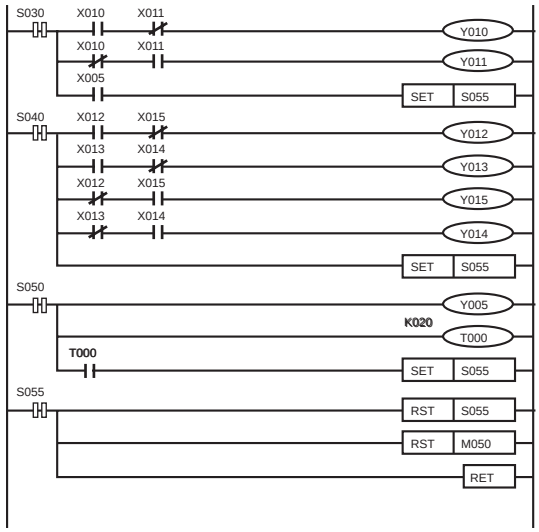
为了说明共享程序的使用方法和好处，看一下 STL 步 S50 会明白，当存在一个错误状态时，STL 步 S6 和 S8 调用这个程序。STL 步 S55 也被前面的多路 STL 所调用，主要是 S5，7，30，40 和 50，在这个特殊情况中，S55 只用来复位和返回程序控制。另外一个要点是使用主控指令选择要被接通的模式或功能，当选择确定，标志 M50 用以锁定所有别的选择，直到当前选择完成。

STL 描述：


M010
X007
S007
初始状态
M010
X005
S005
初始状态
M010
X016
S006
初始状态
M010
X017
S008
初始状态
X003
X004
Y001
Y002
X001
X002
X001
X002
Y001
Y002
X003
X004
S040
S060
S030
X007
T000
X005
S055
常闭触点

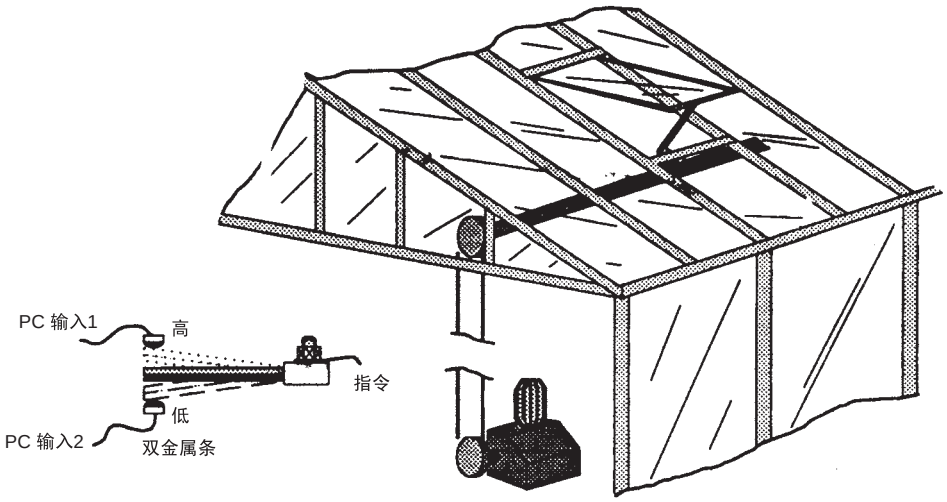
程序：


X006
SET Y003
X000
SET Y004
M8002
SET Y001
SET Y002
X006 X000 M050
MC N0 M010
X007
SET S007
X005
SET S005
X016
SET S006
X017
SET S008
MCR N0
S007
SET M050
Y012 Y013 Y014 Y015
SET Y003
SET Y004
X003 X004
SET S055
S005
SET M050
Y010 Y011
SET Y001
SET Y002
Y001 Y002
SET S055
S006
SET M050
X001 X002
RST Y003
RST Y004
SET S040
X001 X002
SET S050
S008
SET M050
X003 X004
RST Y001
RST Y002
SET S030
Y001 Y002
SET S050


S030
X010 X011
Y010
X010 X011
Y011
X005
SET S055
S040
X012 X015
Y012
X013 X014
Y013
X012 X015
Y015
X013 X014
Y014
SET S055
S050
Y005
K020
T000
T000
SET S055
S055
RST S055
RST M050
RET

10.5 保温

可编程控制器不仅可以控制复杂的高速生产线，同样也能在家里控制一些简单的应用。作为一个很有代表性的例子，下面给出了大棚温度控制器。



器件	软元件	说明
	X000 到 X003	来自双金属条的开窗信号，即大棚太热：（分别从窗 1 到 4）
	X010 到 X013	来自双金属条的关窗信号，即大棚太冷：（分别从窗 1 到 4）
	X004 到 X007	检测窗户已全开：分别从窗 1 到 4
	X014 到 X017	检测窗户已全关：分别从窗 1 到 4
	Y004 到 Y007	选择电机，并起动开窗操作
	Y010 到 Y017	选择电机，并起动关窗操作
	Y000	稳定的驱动脉冲
	D010	输出脉冲数 - 决定于玻璃房
PLSY	FNC57	PLSY 应用指令

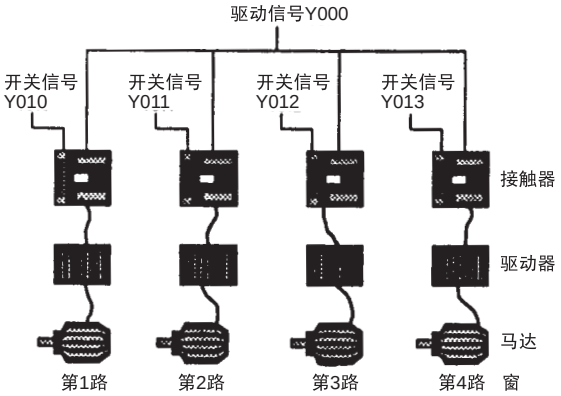
说明：

编程者经常违背的规则中有一条是“简单化原则”。如果程序简单，则是容易写出，容易检查的，而且是很少会出错的。不是所有情况都要求 7 位精度或好的控制，本页给出的大棚说明了一个在学校中学到的基本原理是如何用来有效而廉价地解决问题的。本例中，双金属条用来检测周围温度，这可以是一个控制所有窗户的大型商业绿棚，或是有一个窗户的花园绿棚。本例是控制四扇窗户，每扇窗户有一个检测温度的双金属条，对每个双金属条，存在两个输入，一个对应金属条冷的情况，这可以认为是常闭的，另一个对应金属条热的情况（即它变形时），可认为是一个常开接点。这些输入与窗户状态有直接联系，如果过热，双金属条弯曲，触点接通，于是窗户打开。当大棚部分变冷时，双金属条伸直，窗户关上。

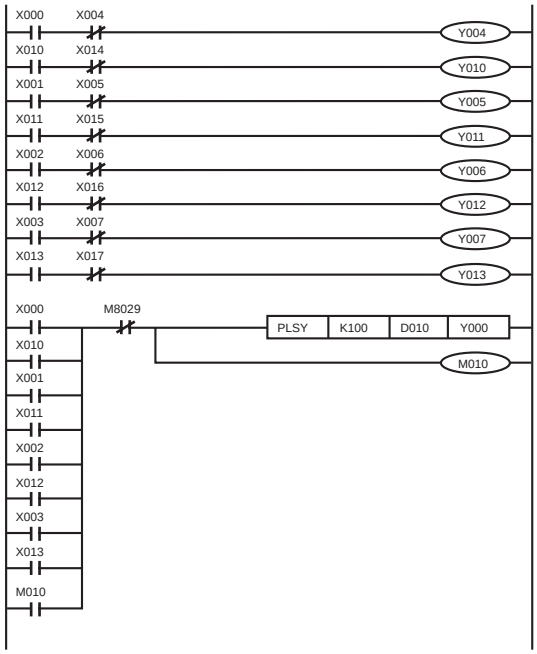
要注意的是只用到一个 PLSY 指令，这是因为它的输出如所要求的那样，直接与四个电机中的每个相连（一个电机对应于一个窗户）。还需检测窗户位置为全开或全关，如果其中一个条件满足，与它对应的控制电机就关断。

PLSY 指令被开或关窗户的请求接通。这部分程序可做改进：请求检查窗户状态 - 可以检测全开全关的检测器。

接线示意图（关窗）：

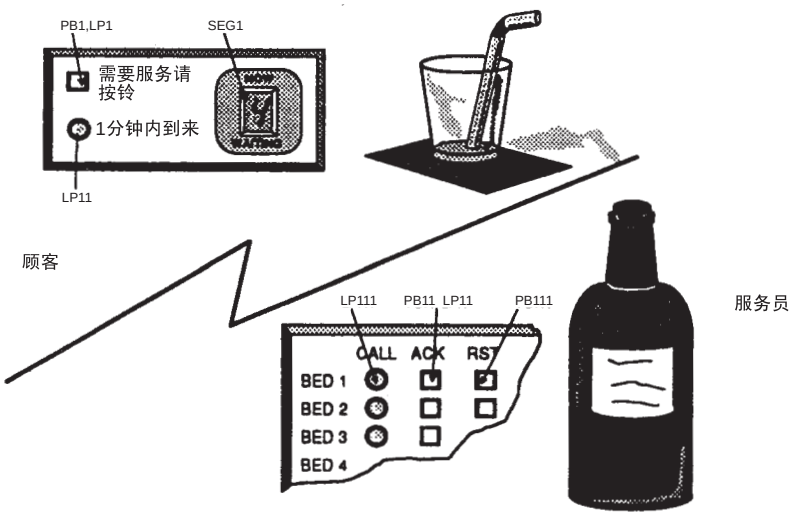

驱动信号Y000
开关信号
Y010
开关信号
Y011
开关信号
Y012
开关信号
Y013
接触器
驱动器
马达
第1路
第2路
第3路
第4路
窗

程序：


X000
X004
Y004
X010
X014
Y010
X001
X005
Y005
X011
X015
Y011
X002
X006
Y006
X012
X016
Y012
X003
X007
Y007
X013
X017
Y013
X000
M8029
PLSY
K100
D010
Y000
X010
M010
X001
X011
X002
X012
X003
X013
M010

10.6 有呼必应的侍者

在很多娱乐场所，比如旅馆、体育馆等，按钮和铃将很快成为常见的东西了。当提供的设备被按、敲或拉时，很难知道是否有人听到这次呼叫，下面使用控制器技术方案能解决这个问题。



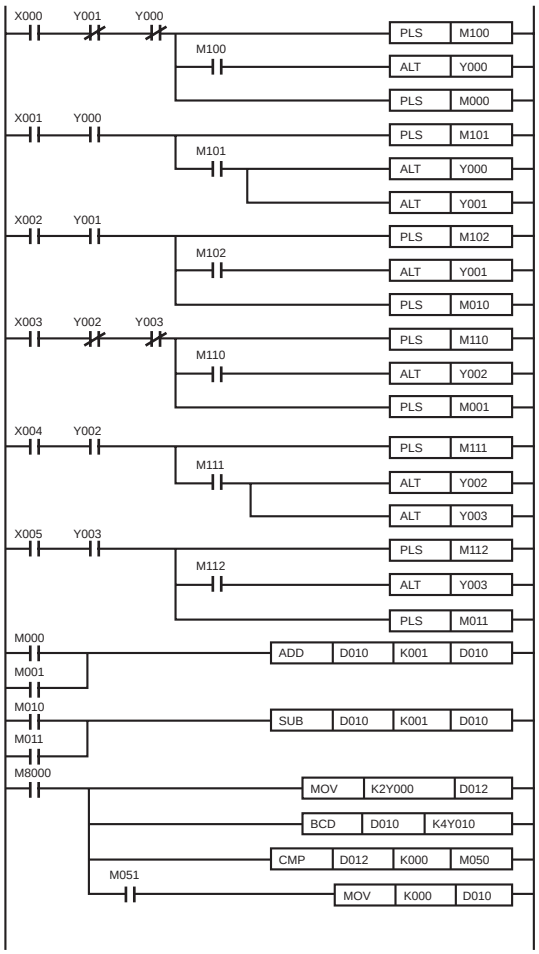
器件	PC 软元件	说明
PB1	X000	BED1 - 请求服务
LP1	Y000	BED1 - 已请求服务
PB11	X001	BED1 - 应答请求
LP11	Y001	BED1 - 确认应答
PB111	X002	BED1 - 复位
	X003	BED2 - 请求服务
	Y002	BED2 - 已请求服务
	X004	BED2 - 应答请求
	Y003	BED2 - 确认应答
	X005	BED2 - 复位
	D010	请求队列中的人数
	D012	错误检测，即未复位请求，对照队列总数如果队列数=0 且仍有请求 - 复位它们 - 有错误
	M8000	PC 运行常闭触点
SEG1	Y010	队列信息显示
ALT	FNC66	ALT 应用指令
ADD	FNC20	ADD 应用指令
SUB	FNC21	SUB 应用指令
MOV	FNC12	MOV 应用指令
BCD	FNC18	BCD 应用指令
CMP	FNC10	CMP 应用指令

说明：

本例可以称为远程顾客和服务员间的信息传送系统，PC 作为调度和传送器。远程顾客请求某种服务，通过一个输入

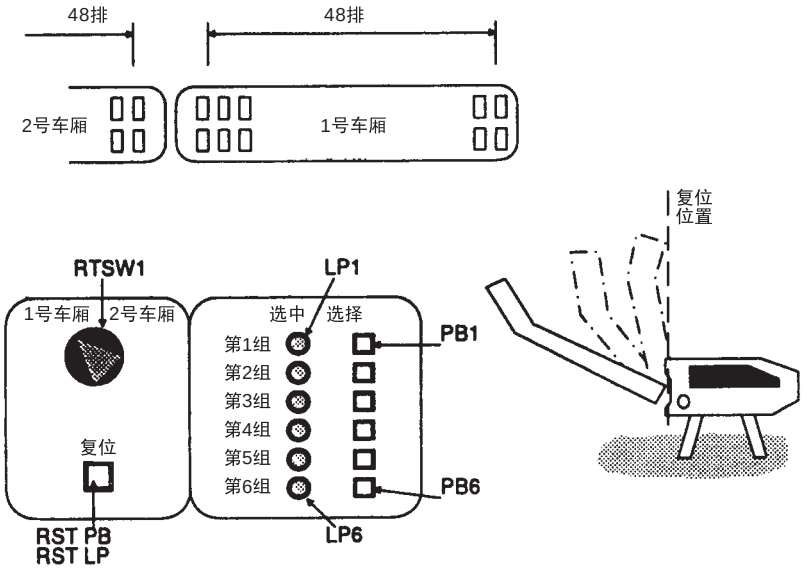
（如：X000）给服务员发信号，这个输入驱动一个信号灯，灯告诉顾客和服务员已经发出一个请求。服务员知道请求来于哪里，因为用以接通信号灯的输出对应于一个位置（如：Y000）。当服务员注意到这个请求，一个应答信号送回至顾客（如 X001，Y001），顾客就能确信服务员已收到请求，服务员也会知道哪位顾客正在请求。当服务员满足了顾客要求时，复位应答信号（如 X002）。此过程可为下一个顾客所重复（如：X003-005，Y002-003）。所以，当前等待顾客能知道轮到他们得花多少时间，一个 7 段显示表示现在排队的顾客数。本例只列出两个“呼叫”台，当然还可以增加。

程序：



10.7 复位 - 回到水平位置

在顾客利益日益增强的当今现代世界里，可以看到很多长途汽车、公共汽车、电车公司正提供给顾客更好的座椅设备，其中的一种是可调整靠背的座椅。

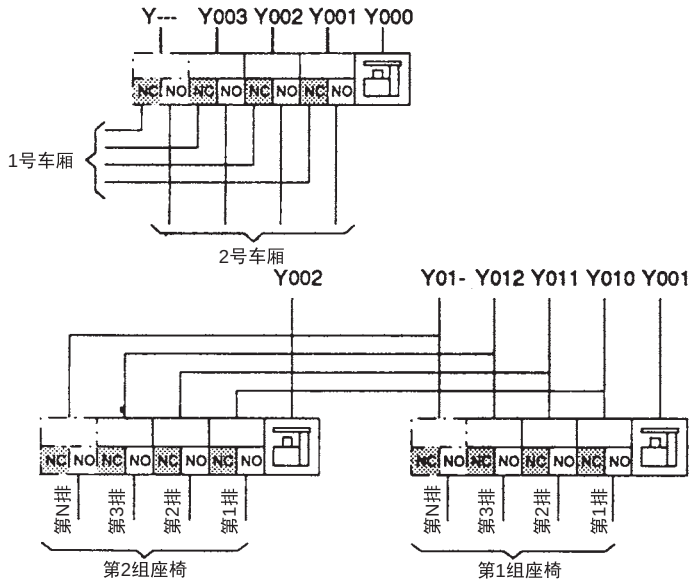


器件	PC 软元件	说明
RTSW1	X011	选择 1 号车厢
	X012	选择 2 号车厢
RSTPB	X000	复位选择的座位组
RSTLP	Y007	复位有效
PB1 - 6	X001 - 6	选择所期望复位的座位组（1 到 6）
LP1 - 6	Y001 - 6	确认所选择复位的座位组（1 到 6）
	Y000	车厢选择的输出（Y000—OFF=1号车厢）
	Y010 - Y017	复位所选择的第 1 到 8 排
	M019	复位过程标志
	T000	复位 1 排座位的时间
ZRST	FNC40	ZRST 应用指令
SFTL	FNC35	SFTL 应用指令

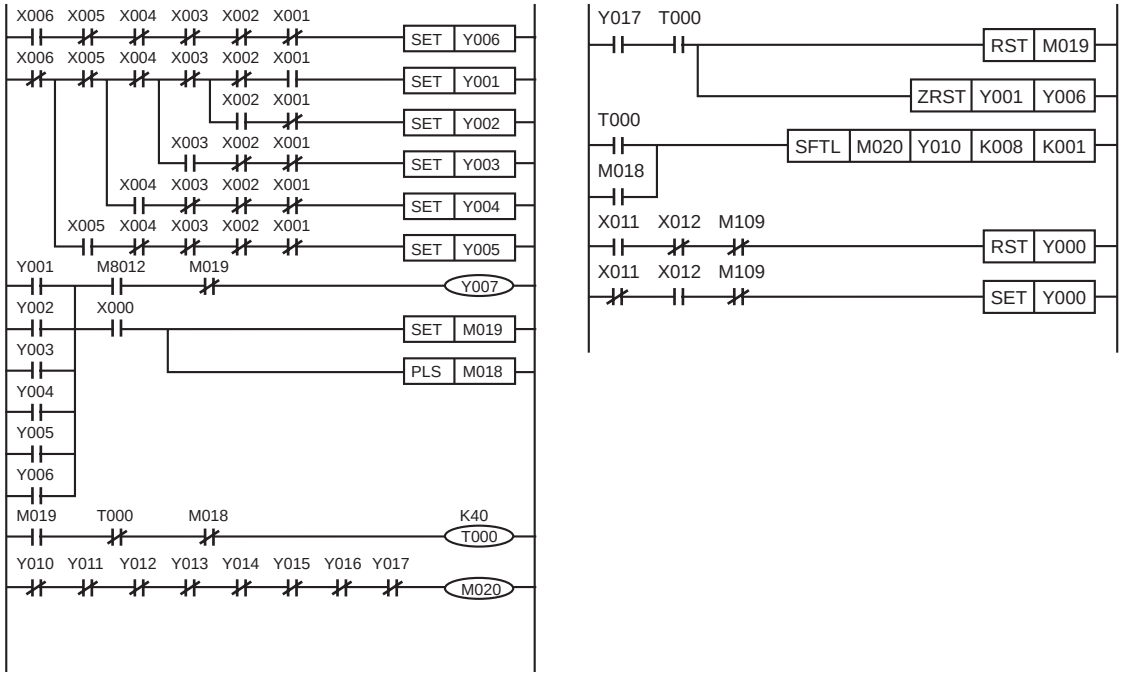
说明：

正如上面所提到的，大量公司正在提供这些座位的附加优点，其中许多公司，在每次旅行开始时，将座位复位到垂直位置，这会使外观整齐有序。用手复位所有的椅子，需费很多的时间，本节给出的例子用以复位两节相邻车厢的座位。操作人员使用两位旋转开关（RTSW1）选择要复位座位的车厢，选择要复位的座位组，且任何一个时间只能选中一个组，选择的座位组用接通相应的灯表示，接着复位钮闪烁，等待被按下。接下来开始座位复位操作，这个操作持续一段预定时间，而不检查是否都已复位。本程序背后的思想是：这不是一个有源复位，只是解开一个锁，使座位在弹簧的拉力下弹回，从而使座位复位。如果旅客们都仍在座位上（可能在睡觉）而进行该操作，这样做会更安全些。旅客不会从位子上弹出，而是没觉察有什么事发生，这是因为弹簧力一般不会大到足以弹起椅背和旅客体重的总重量。

接线示意图：

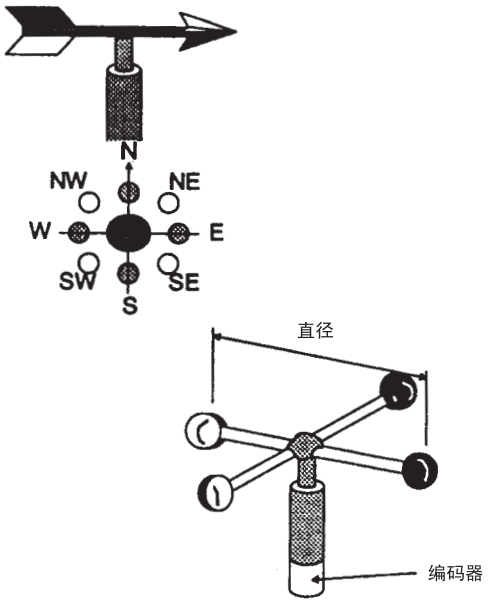

Y--- Y003 Y002 Y001 Y000
NC NO NC NO NC NO NC NO
1号车厢
2号车厢
Y002
Y01- Y012 Y011 Y010 Y001
NC NO NC NO NC NO NC NO
第N排
第3排
第2排
第1排
第2组座椅
第1组座椅

程序：


X006 X005 X004 X003 X002 X001
SET Y006
SET Y001
SET Y002
SET Y003
SET Y004
SET Y005
Y001 M8012 M019
Y007
Y002 X000
SET M019
PLS M018
Y003
Y004
Y005
Y006
M019 T000 M018
K40
T000
Y010 Y011 Y012 Y013 Y014 Y015 Y016 Y017
M020
Y017 T000
RST M019
ZRST Y001 Y006
T000
M018
SFTL M020 Y010 K008 K001
X011 X012 M109
RST Y000
X011 X012 M109
SET Y000

10.8 监测风

用可编程控制器很容易做基本的检测和计算，这类应用中的一例是小型气象站，这对于高层办公室的窗户清洁是极其重要的，它可以用来判断风是否大到有危险的程度。



器件	PC 软元件	说明
N	X010	风向输入 - 北
NE	X010，11	风向输入 - 东北
E	X011	风向输入 - 东
SE	X012，11	风向输入 - 东南
S	X012	风向输入 - 南
SW	X012，13	风向输入 - 西南
W	X013	风向输入 - 西
NW	X010，13	风向输入 - 西北
	X000，C235	计算风速的编码器输入
DIA	D012	旋轴的直径，以 mm 为单位
	D010	计算的 π 值
	D014	5分钟内的运动距离
	D018	1 分钟内旋转次数
	D027	所计算的风速，以 m/min 为单位
	Y010-13	表示风向的输出，分别 E、S、W、N
（D）DIV	FNC23	DIV 应用指令
MOV	FNC12	MOV 应用指令
（D）MUL	FNC22	MUL 应用指令

说明：

本例程序包括两个功能，它计算相对风速，并指出风向。风向标是老式的机械结构，但其数据被电子存储和控制。首先看测风速功能，风速每 5 分钟计算一次，这由定时器 T000 控制，这种计算方法能给出一个较好的平均风速，但是它不能报告阵风的速度。

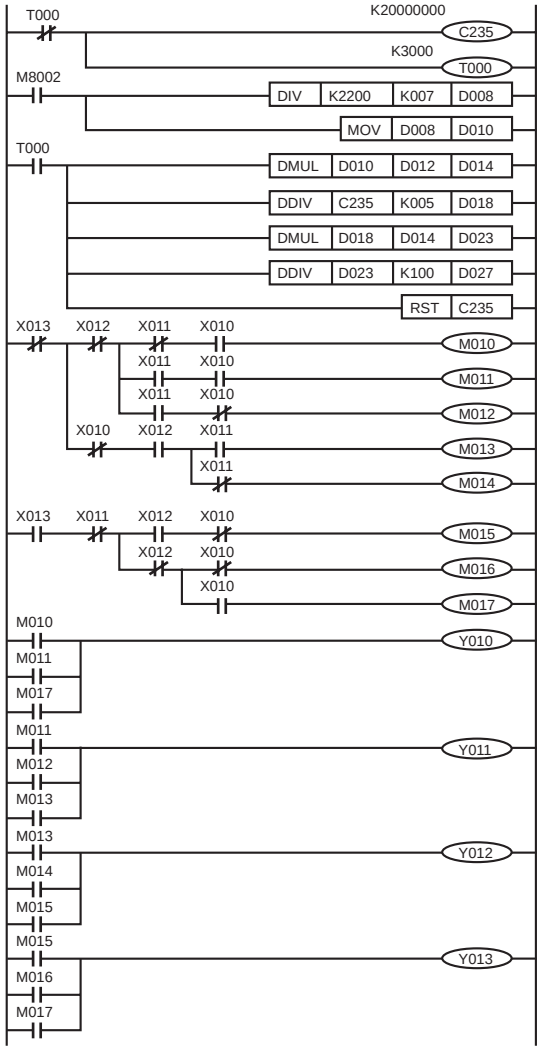
由风驱动的旋轴直径必须输入到数据寄存器 D12 中，它应以 mm 为单位输入，即0.1m=100mm: 输入100至D12。旋轴的旋转次数由高速计数器 C235 来计数，数据输入由 X000 输入。

先计算旋轴运动轨迹圆的圆周长，再乘以旋转次数，就计算出经过的距离。因为运行时间固定为 5 分钟，答案除以5，就得到一个每分钟运动距离的答案。

为了得到更高精确度，本例的计算中加入了比例缩放项。最后答案应该除以 10^3，从而得到米每分钟为单位的风速。对于一个 200mm 直径的旋轴，可以测量计算 0.628 到 10，000m/min 的风速。

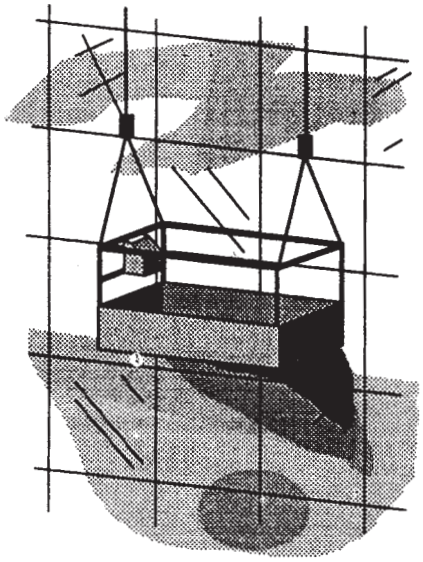
程序第二部分包括风向的检测。使用 8 个检测器，而只有 4 个输入，检测器在正向指向之间的位置时与 2 个输入连接。一些编程者会说，用以指示风向的四个输出灯的逻辑并不需要。但其实这是一个好主意：可用它们进行安全全检测、程序互锁，从而防止意外事件的发生。在某些情况下，对象不是生物或危险的机器，则可以自行处理，但是编程者不应该与安全进行赌博。本例中，程序进行检测以预防风刮起的垃圾和想在风向标上栖息的鸟。

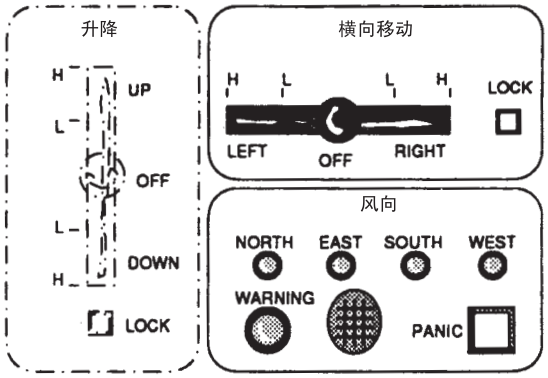
程序：


T000
K20000000
C235
K3000
T000
M8002
DIV K2200 K007 D008
MOV D008 D010
T000
DMUL D010 D012 D014
DDIV C235 K005 D018
DMUL D018 D014 D023
DDIV D023 K100 D027
RST C235
X013 X012 X011 X010 M010
X011 X010 M011
X011 X010 M012
X010 X012 X011 M013
X011 M014
X013 X011 X012 X010 M015
X012 X010 M016
X010 M017
M010 Y010
M011
M017
M011 Y011
M012
M013
M013 Y012
M014
M015
M015 Y013
M016
M017

10.9 左右移动

下面的程序是控制一个擦窗器的大型程序的一小部分，本程序控制在一个建筑面上的左右转向。





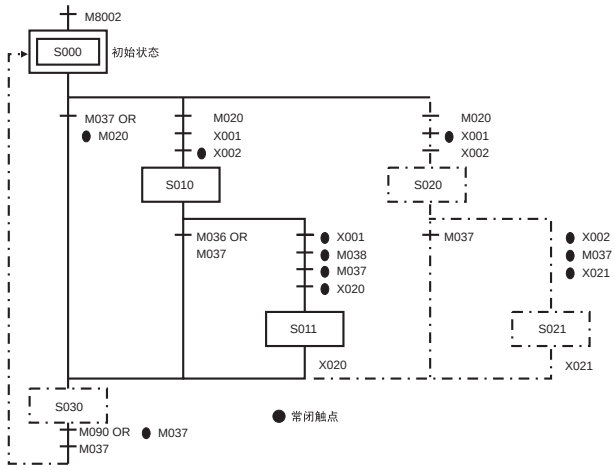
器件	PC 软元件	说明
NORTH	Y010	风向指示灯
EAST	Y011	
SOUTH	Y012	
WEST	Y013	
	Y014	报警灯和蜂鸣器
LOCK - LAMP/BRAK	Y007	横向操作的刹车及指示
LOCK PB	X001	横向操作的锁定
LFTH	X004	选择左向高速
LFTL	X005	选择左向低速
RHTH	X007	选择右向高速
RHTL	X006	选择右向低速
	Y005	驱动吊车向左
	Y006	驱动吊车向右
	Y001	电机速度控制
PANIC	X000	紧急按钮
	Y000	紧急报警
	D027	当前风速
	X020	检测移动锁定
（D）ZCP	FNC11	ZCP 应用指令
MOV	FNC12	MOV 应用指令
PWM	FNC58	PWM 应用指令

说明：

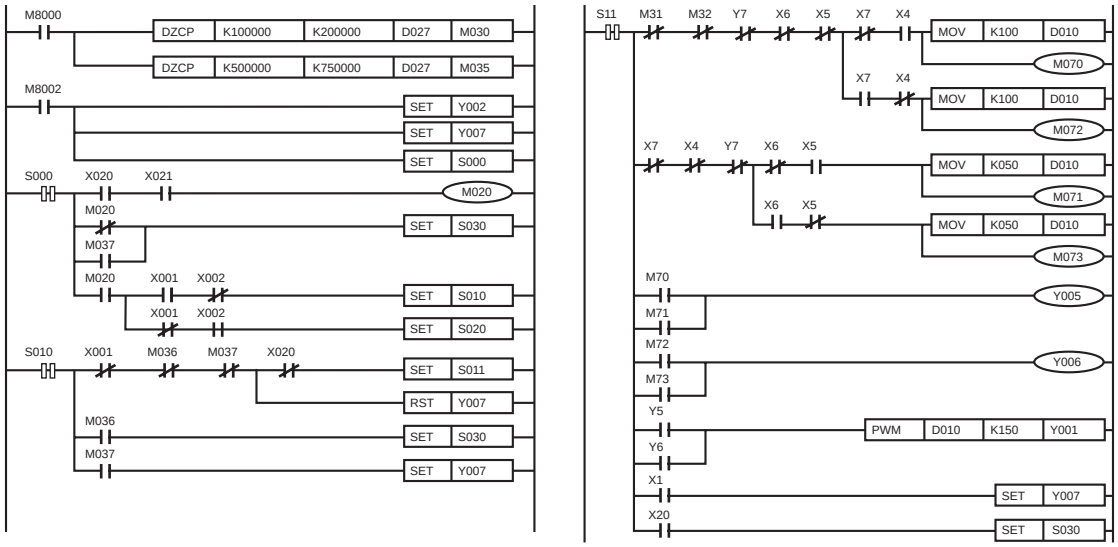
本程序只是一个大程序的一小部分，不过它仍有一些值得注意的特别之处。对一个擦窗车的控制着来说，最大的危险之一是风，这个影响在本例程序中已考虑到。最初的步检测存在数据寄存器 D027 中的值，事实上是检测风速是否在安全范围内。要注意的是这里只有 2 个 ZCP 指令，然而它们包含了 5 个风速范围，这通过检测 2 个 ZCP 指令之间的范围来实现。其结果是，如果风力稍强，则水平速度限定为一个较低的值；如果风力过强，则所有水平运动都停止，并给出警报信号。另一注意点是在 STL 步 S10 的第一行，可看到 S11 是在 X20 断开时才被置位，这是很重要的，因为 X20 用于检测输出X7 所控制的刹车的位置。

将要发生的是程序将允许对已完全完成的输出 Y7 做复位操作，即在程序继续之前 X20 被删除。对一个编程者来说，很容易犯简单地说“我已做了”的错误，因为所期望的动作被控制了一段时间，而这段时间对于完全完成操作来说可能太短了 。

STL 描述：

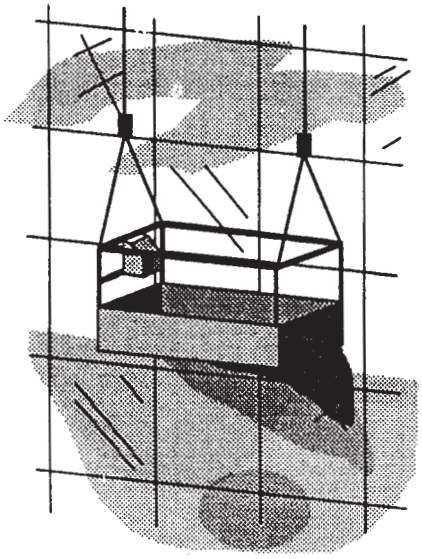


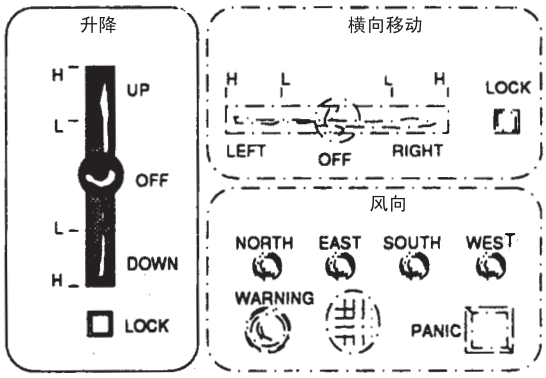
程序：



10.10 上下移动

从擦窗吊车程序选摘的这部分程序控制吊车在建筑面上垂直运动，尽管这只是一个大型程序的一小部分，但它仍有许多有意义的编程要点。





器件	PC 软元件	说明
	X000	紧急按钮
LOCK - LAMP/BRAKE	Y002	垂直操作的刹车及指示
LOCK PB	X002	垂直操作的锁定
UP H	X014	高速上升
UPL	X015	低速上升
DOWNH	X017	高速下降
DOWNL	X016	低速下降
	Y000	紧急报警
	Y004	驱动吊车向上
	Y003	驱动吊车向下
	Y001	电机速度控制
	X021	检测升降锁定
	M090	降到地面标志 - 用于在极强风情况时的自动下降
MOV	FNC12	MOV 应用指令
PWM	FNC58	PWM 应用指令

说明：

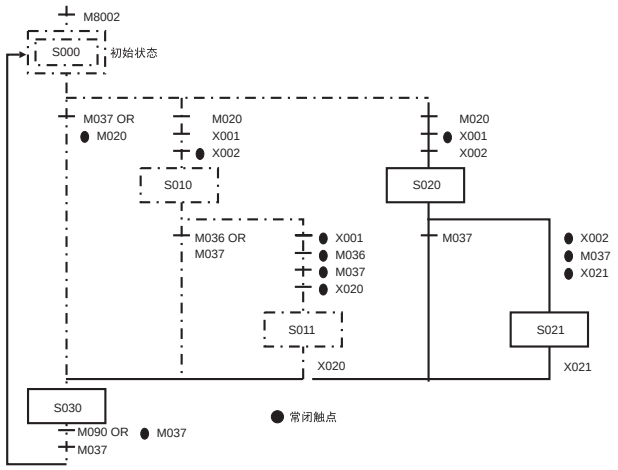
可以注意到本程序是以梯形图或 STL 形式编写的，这对于下面要讲的两点来说是非常重要的。

如果研究 STL 步 S20 和 S30 的内容，可看到它们都包含 Y3 输出，在标准编程技术中，这叫做双线圈输出，是绝对不允许的。在正常编程中，如果第一次出现的输出 Y3 接通，而在同一次扫描中第二次出现时复位此输出，从而导致一种很危险的情况。不过，因为 STL 编程每一个 STL 步的程序相对独立，输出线圈在许多 STL 块中的使用是很安全的，从而使编程更简单。

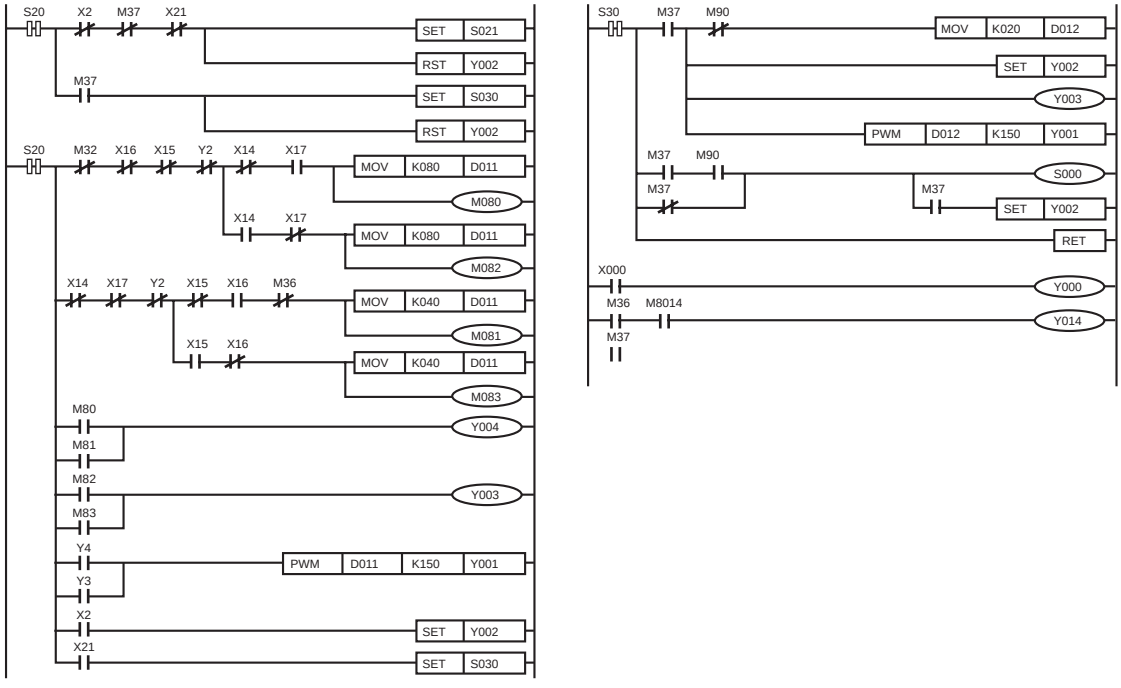
值得注意的第二点实际上也是出于与第一点同样的原因。检查 STL 步 S21 和 S30，可注意到实际上存在 2 个 PWM 指令，如果查看编程手册，可知每一个程序只能使用一次 PWM 指令。STL 步根据其自身特性实际上是一个程序，所以“每程序一个 PWM”规则实际上仍没被破坏。

本程序允许吊车在正常时以两种不同速度升高或降低，如果风速超过一定限值，吊车会被限制在低速运行，如果风力过强，吊车会自动以“爬行”速度降到地面。用以表示风速变化的标志是：M32，M36 和 M37。

STL 描述：

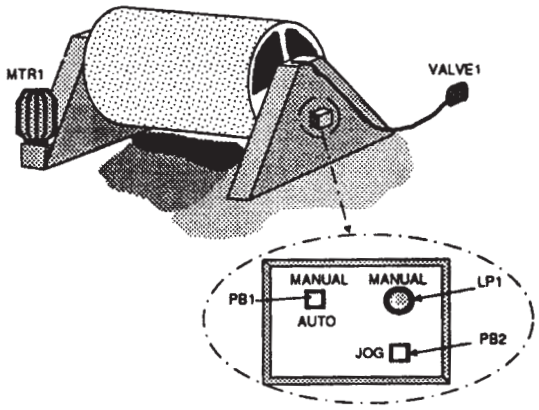


程序：



10.11 蘑菇种植

下面的应用是使用有限空间来种植更多庄稼，在本例中为蘑菇。所以，如果你没有很多的地方，请看下面：



器件	PC 软元件	说明
VALVE1	Y003	水供应阀门
LP1	Y005	人工模式指示灯
PB1	X000	模式选择按钮
PB2	X002	点动旋转 - 人工模式
MTR1	Y001	电机驱动
	X001	检测太阳光的光电管
	Y004	旋转方向的控制信号
ALT	FNC66	ALT 应用指令
CJ	FNC00	CJ 应用指令
PWM	FNC58	PWM 应用指令

说明：

整个例子基于一个简单的物理规则：一个圆的直径小于圆周长。因此开发一种机器，它能把蘑菇种植在一个圆柱体的圆周上，这样种植的庄稼大约 3 倍于机器所覆盖的面积所产生的量。

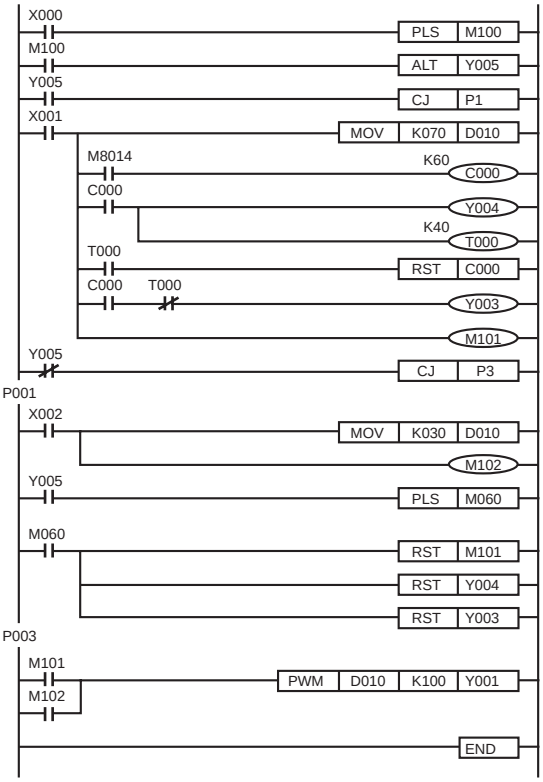
这里有一个问题需要克服，如果植物沿一个圆柱体种植，那些不在圆柱体顶部的，不在太阳直射下的植物，它们的生长速度会慢一些，同时会力图接近光源。所以，蘑菇可能会“弯曲”或“绞结”。另一种可能是：圆柱体下部的蘑菇会完全不生长。

解决方法是旋转这个圆柱体。因此，开发了下面的程序，用以每小时改变旋转方向，这样就能生产出直而健康的蘑菇了。另外，每小时内一个规定时间段中，对旋转圆柱体里面浇水。

最后一个主要困难是机器必须能切换到人工模式，使耕种者检查和收割生长的庄稼。一个按钮交替切换自动/人工模式（PB1），在人工模式下，灯 LP1 亮，操作者可以按点动控制钮，使圆柱体旋转。

从编程角度来看，使用条件跳转对每个相关部分作跳转控制的方法是非常实用的。最后部分的以 P3 指针标记的程序用于人工和自动模式，自动模式跳到指针 P3，而人工模式时，把它作为一般顺序的程序处理。

程序：


X000
PLS M100
M100
ALT Y005
Y005
CJ P1
X001
MOV K070 D010
M8014
K60
C000
C000
Y004
K40
T000
T000
RST C000
C000 T000
Y003
M101
Y005
CJ P3
P001
X002
MOV K030 D010
M102
Y005
PLS M060
M060
RST M101
RST Y004
RST Y003
P003
M101
PWM D010 K100 Y001
M102
END