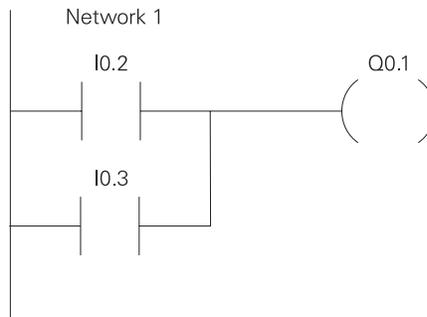


An OR Operation

In this example an OR operation is used in network 1. It can be seen that if either input I0.2 (input 3) or (O in the statement list) input I0.3 (input 4), or both are true, then output Q0.1 (output 2) will be true.

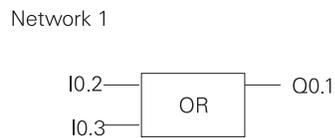
Ladder Diagram Representation



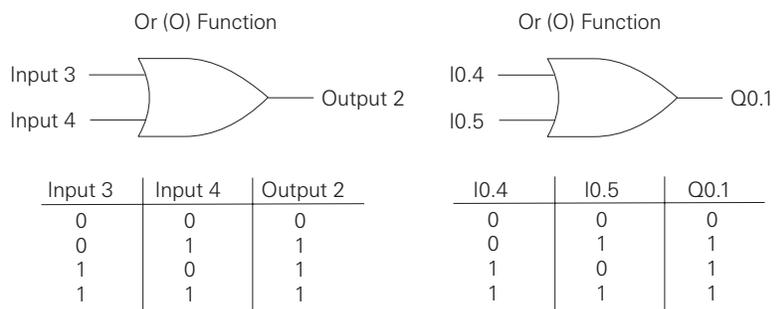
Statement List Representation

```
Network 1
LD  I0.2
O   I0.3
=   Q0.1
```

Function Block Diagram Representation

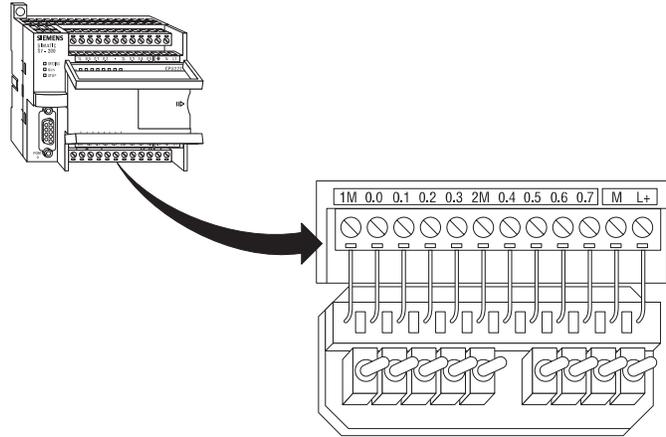


Another way to see how an OR function works is with a Boolean logic diagram. The symbol differs slightly from an AND function. The OR function is represented by a number of inputs on the left side. In this case there are two inputs. The output is represented on the right side. It can be seen from the table that any input can be a logic 1 in order for the output to be a logic 1.



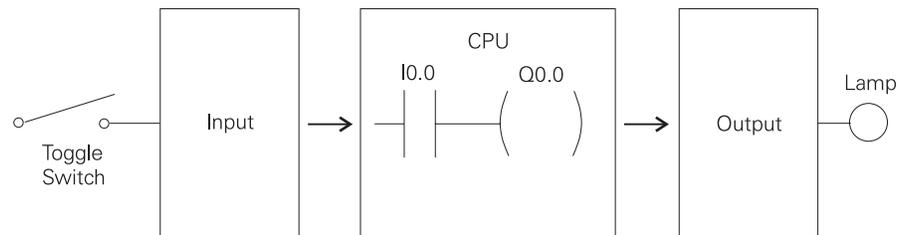
Testing a Program

Once a program has been written it needs to be tested and debugged. One way this can be done is to simulate the field inputs with an input simulator, such as the one made for the S7-200. The program is first downloaded from the programming device to the CPU. The selector switch is placed in the RUN position. The simulator switches are operated and the resulting indication is observed on the output status indicator lamps.

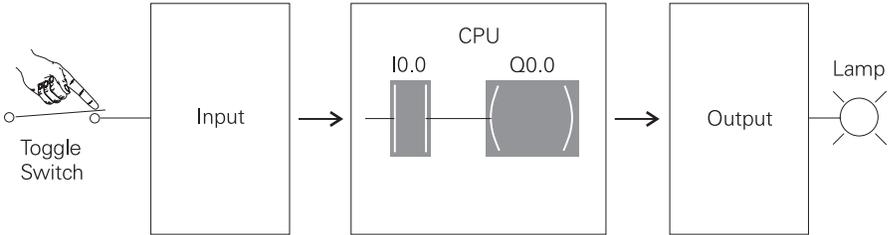


Status Functions

After a program has been loaded and is running in the PLC, the actual status of ladder elements can be monitored using STEP 7 Micro/WIN32 software. The standard method of showing a ladder element is by indicating the circuit condition it produces when the device is in the deenergized or non operated state. In the following illustration input 1 (I0.0) is programmed as a normally open (NO) contact. In this condition, power will not flow through the contacts to the output (Q0.0).

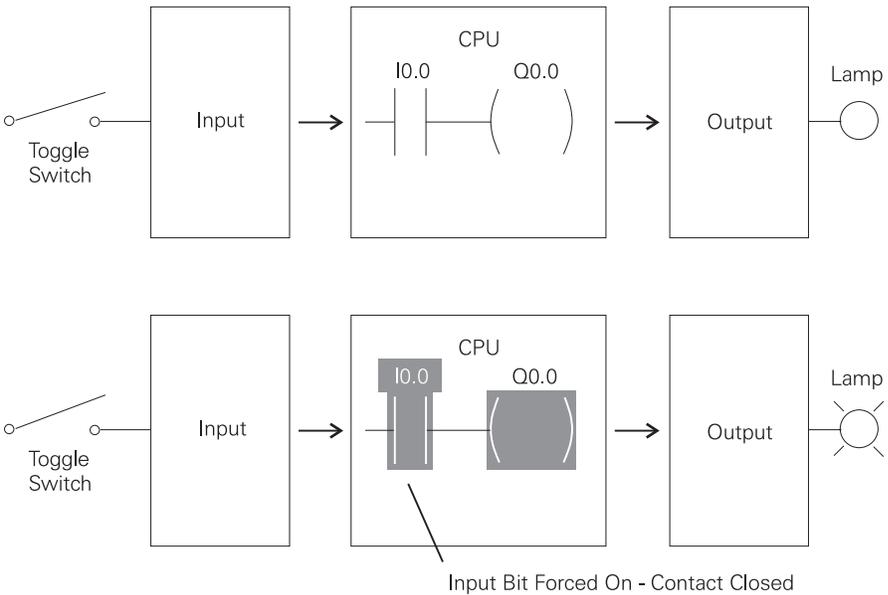


When viewing the ladder diagram in the status mode, control elements that are active, or true (logic 1), are highlighted. In the example shown the toggle switch connected to input 1 has been closed. Power can now flow through the control element associated with input 1 (I0.0) and activate the output (Q0.0). The lamp will illuminate.

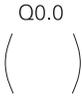


Forcing

Forcing is another useful tool in the commissioning of an application. It can be used to temporarily override the input or output status of the application in order to test and debug the program. The force function can also be used to override discrete output points. The force function can be used to skip portions of a program by enabling a jump instruction with a forced memory bit. Under normal circumstances the toggle switch, shown in the illustration below, would have to be closed to enable input 1 (I0.0) and turn on the output light. Forcing enables input 1 even though the input toggle switch is open. With input 1 forced high the output light will illuminate. When a function is forced the control bit identifier is highlighted. The element is also highlighted because it is on.

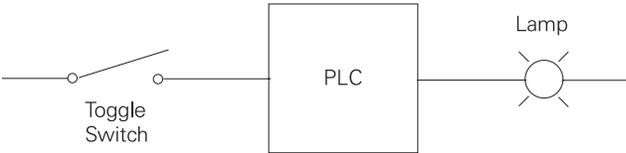


The following table shows the appearance of ladder elements in the Off, forced, and On condition.

	Status Bit On Contacts Closed	Status Bit Off Contacts Open	Status Bit Forced On Contacts Closed	Status Bit Forced Off Contacts Open
Normally Open Contacts				
	Status Bit On Contacts Open	Status Bit Off Contacts Closed	Status Bit Forced On Contacts Open	Status Bit Forced Off Contacts Closed
Normally Closed Contacts				
	Status Bit On Coil Energized	Status Bit Off Coil Deenergized	Status Bit Forced On Coil Energized	Status Bit Forced Off Coil Deenergized
Output Coils				

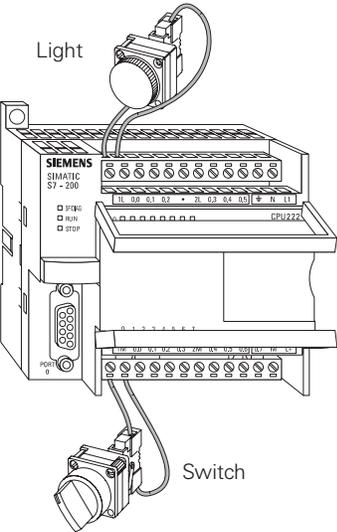
Discrete Inputs/Outputs

To understand discrete control of a programmable controller the same simple lamp circuit illustrated with forcing will be used. This is only for instructional purposes as a circuit this simple would not require a programmable controller. In this example the lamp is off when the switch is open and on when the switch is closed.

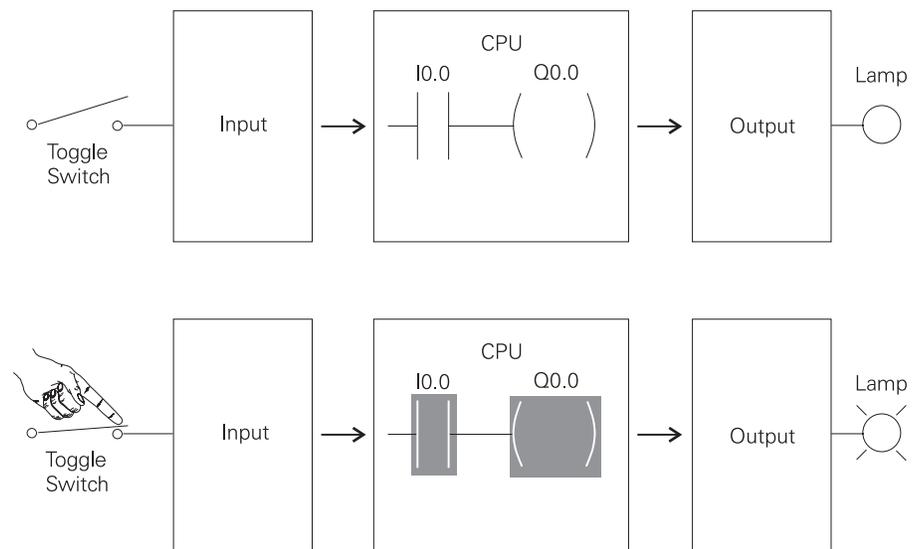


Wiring

To accomplish this task, a switch is wired to the input of the PLC and an indicator light is wired to output terminal.

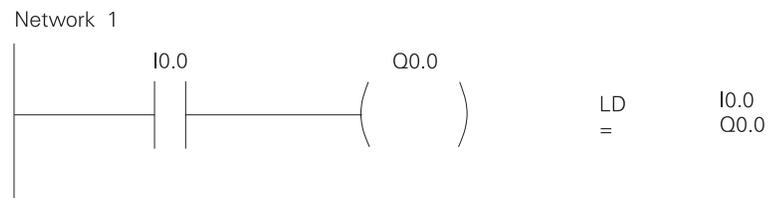


The following drawing illustrates the sequence of events. A switch is wired to the input module of the PLC. A lamp is wired to the output module. The program is in the CPU. The CPU scans the inputs. When it finds the switch open I0.0 receives a binary 0. This instructs Q0.0 to send a binary 0 to the output module. The lamp is off. When it finds the switch closed I0.0 receives a binary 1. This instructs Q0.0 to send a binary 1 to the output module, turning on the lamp.

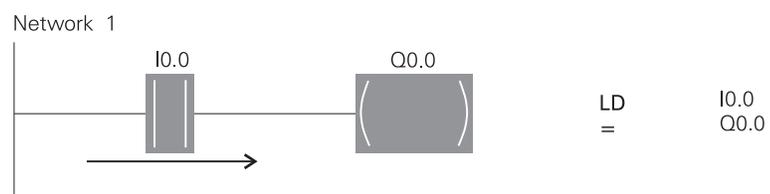


Program Instruction

When the switch is open the CPU receives a logic 0 from input I0.0. The CPU sends a logic 0 to output Q0.0 and the light is off.

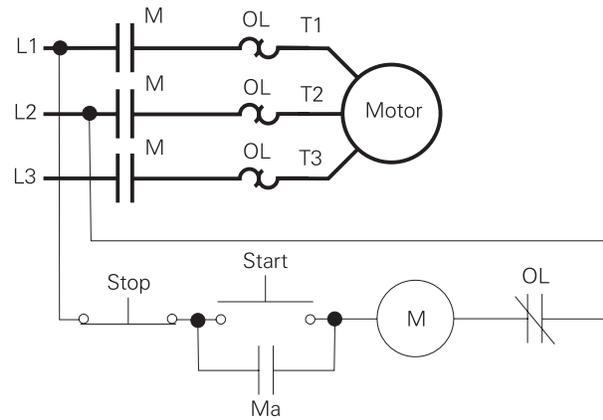


When the switch is closed the CPU receives a logic 1 from input I0.0. The CPU sends a logic 1 to output Q0.0, thus activating Q0.0. The light turns on.

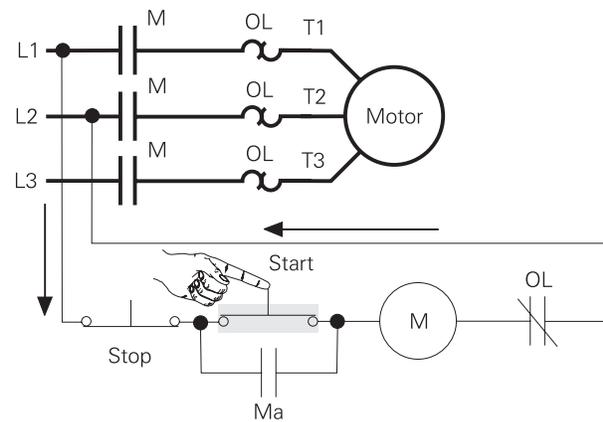


Motor Starter Example

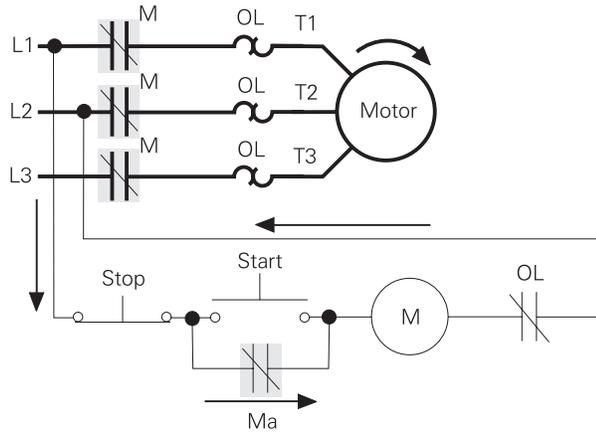
The following example involves a motor start and stop circuit. The line diagram illustrates how a normally open and a normally closed pushbutton might be used in a control circuit. In this example a motor started (M) is wired in series with a normally open momentary pushbutton (Start), a normally closed momentary pushbutton (Stop), and the normally closed contacts of an overload relay (OL).



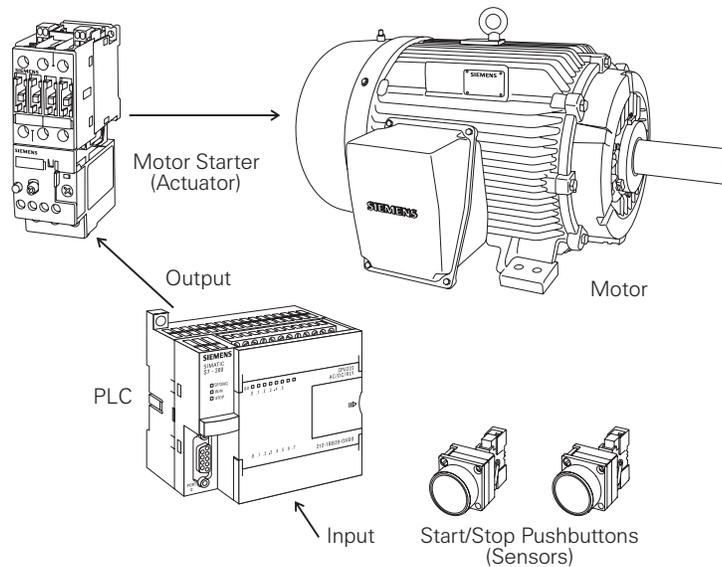
Momentarily depressing the Start pushbutton completes the path of current flow and energizes the motor starter (M).



This closes the associated M and Ma (auxiliary contact located in the motor starter) contacts. When the Start button is released a holding circuit exists to the M contactor through the auxiliary contacts Ma. The motor will run until the normally closed Stop button is depressed, or the overload relay opens the OL contacts, breaking the path of current flow to the motor starter and opening the associated M and Ma contacts.

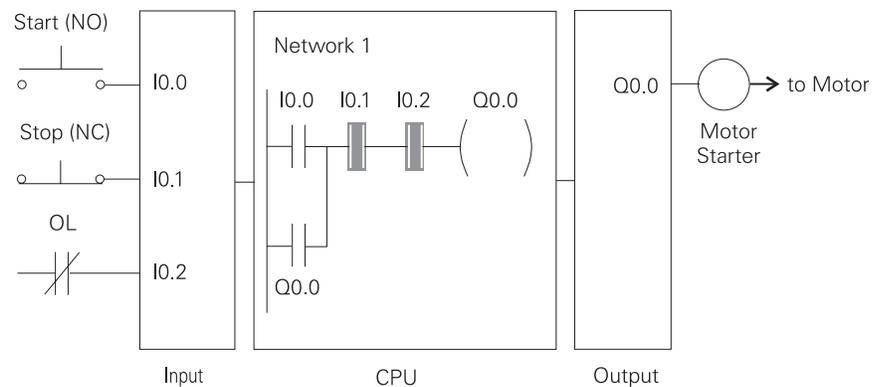


This control task can also be accomplished with a PLC.

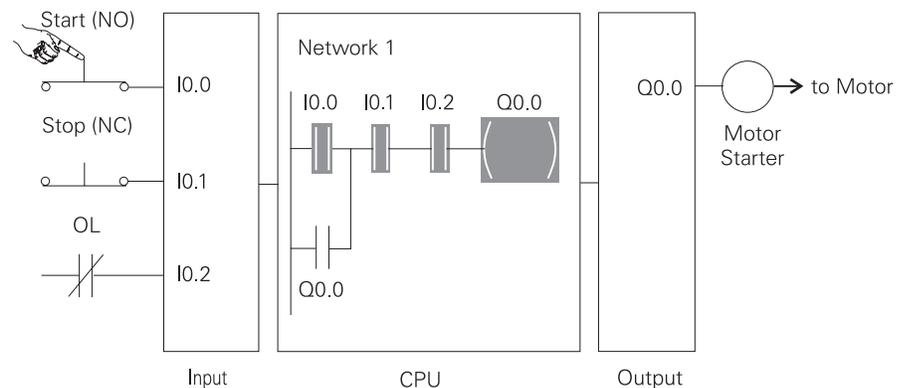


Program Instruction

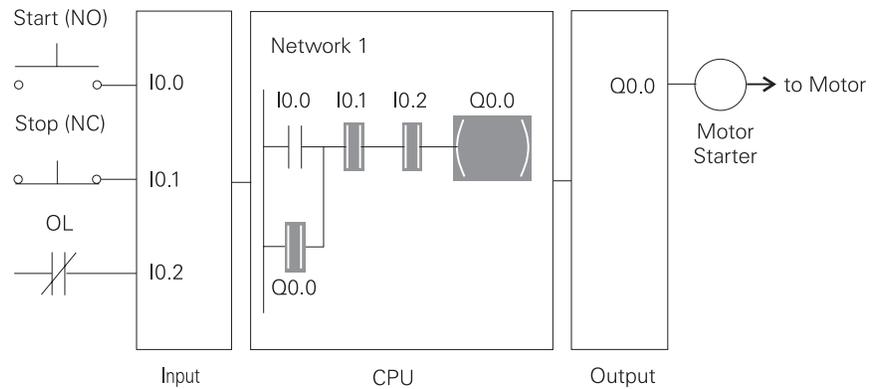
A normally open Start pushbutton is wired to the first input (I0.0), a normally closed Stop pushbutton is wired to the second input (I0.1), and normally closed overload relay contacts (part of the motor starter) are connected to the third input (I0.2). The first input (I0.0), second input (I0.1), and third input (I0.2) form an AND circuit and are used to control normally open programming function contacts on Network 1. I0.1 status bit is a logic 1 because the normally closed (NC) Stop Pushbutton is closed. I0.2 status bit is a logic 1 because the normally closed (NC) overload relay (OL) contacts are closed. Output Q0.0 is also programmed on Network 1. In addition, a normally open set of contacts associated with Q0.0 is programmed on Network 1 to form an OR circuit. A motor starter is connected to output Q0.0.



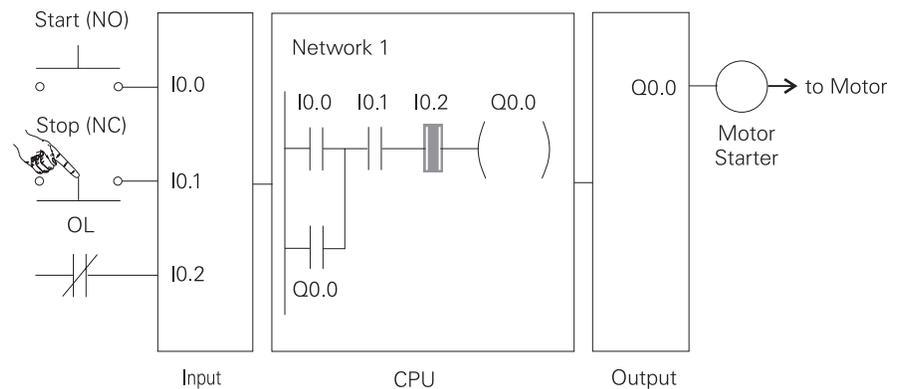
When the Start pushbutton is depressed the CPU receives a logic 1 from input I0.0. This causes the I0.0 contact to close. All three inputs are now a logic 1. The CPU sends a logic 1 to output Q0.0. The motor starter is energized and the motor starts.



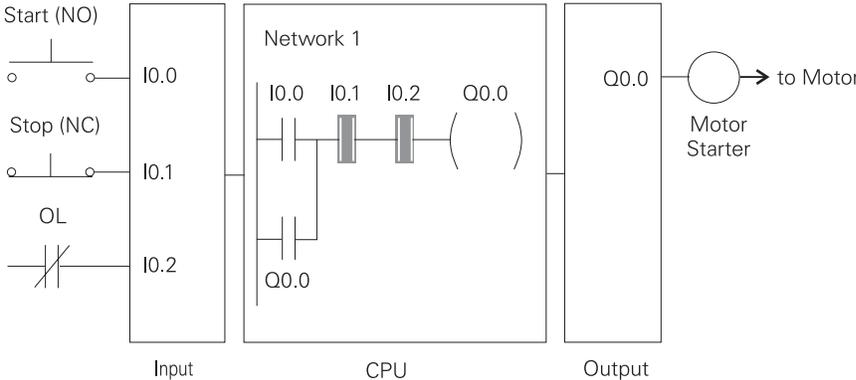
When the Start pushbutton is pressed, output Q0.0 is now true and on the next scan, when normally open contact Q0.0 is solved, the contact will close and output Q0.0 will stay on even if the Start pushbutton has been released.



The motor will continue to run until the Stop pushbutton is depressed. Input I0.1 will now be a logic 0 (false). The CPU will send a binary 0 to output Q0.0. The motor will turn off.

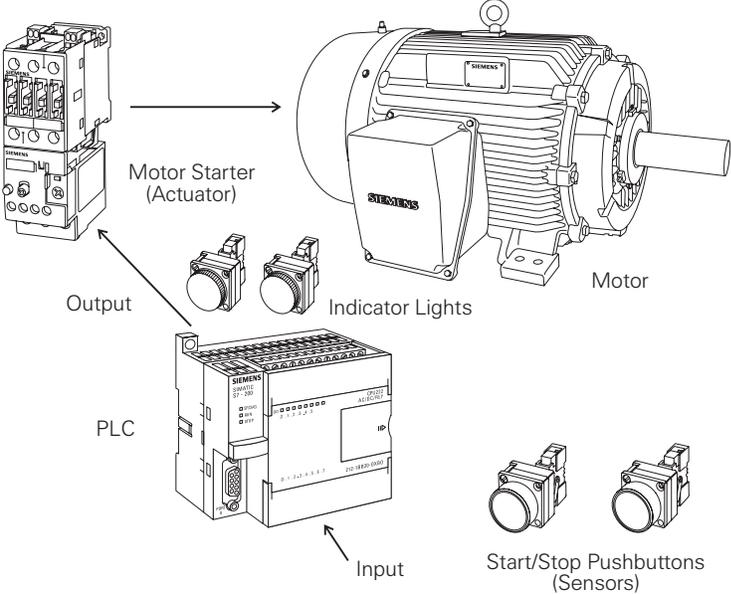


When the Stop pushbutton is released IO.1 logic function will again be true and the program ready for the next time the Start pushbutton is pressed.

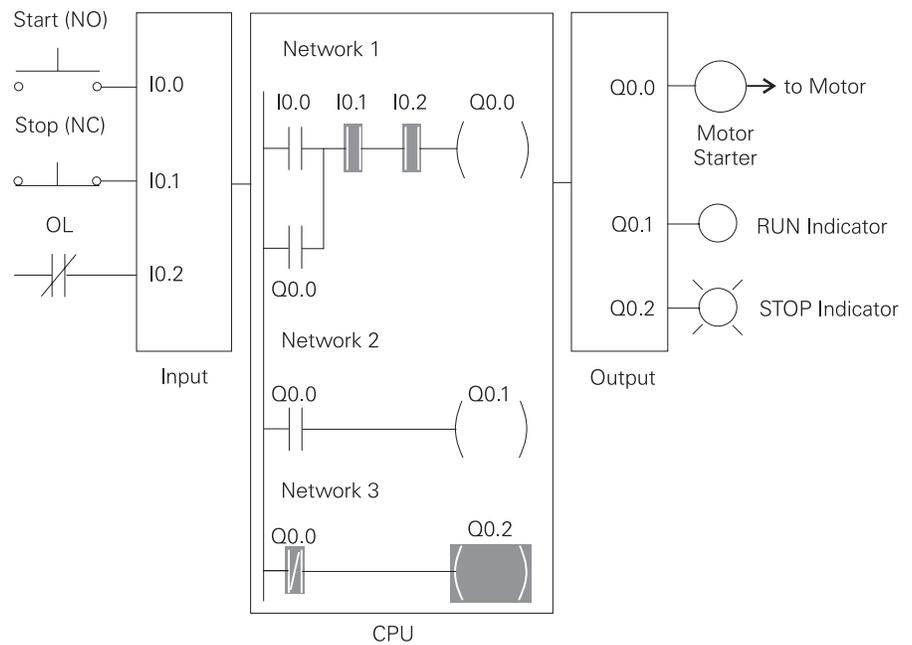


Expanding the Application

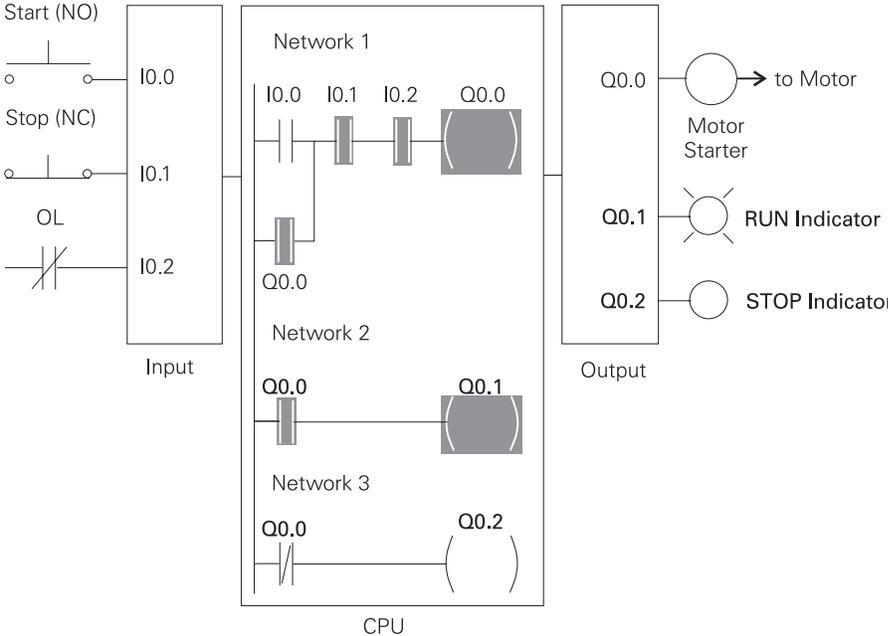
The application can be easily expanded to include indicator lights for RUN and STOP conditions. In this example a RUN indicator light is connected to output Q0.1 and a STOP indicator light is connected to output Q0.2.



It can be seen from the ladder logic that a normally open output Q0.0 is connected on Network 2 to output Q0.1 and a normally closed Q0.0 contact is connected to output Q0.2 on network 3. In a stopped condition output Q0.0 is off. The normally open Q0.0 contacts on Network 2 are open and the RUN indicator, connected to output Q0.1 light is off. The normally closed Q0.1 on Network 3 lights are closed and the STOP indicator light, connected to output Q0.2 is on.

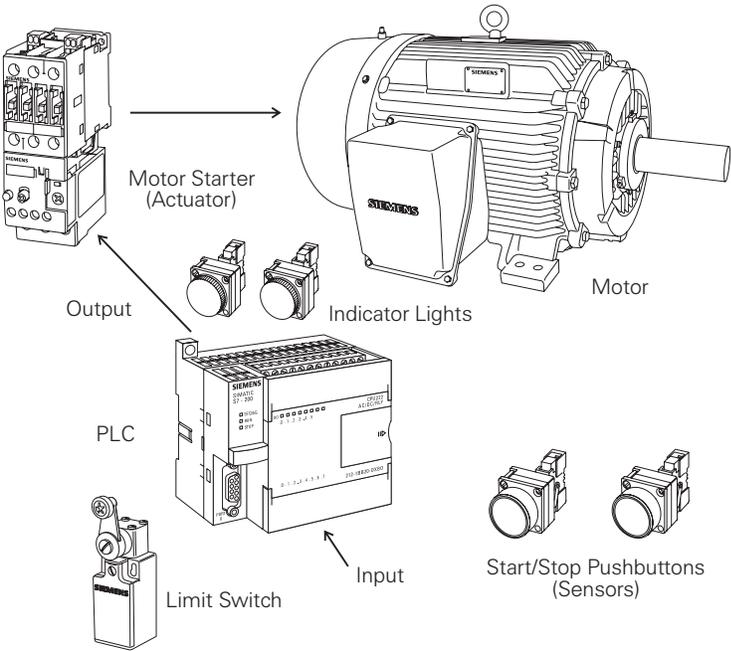


When the PLC starts the motor output Q0.0 is now a logic high (On). The normally open Q0.0 contacts on Network 2 now switch to a logic 1 (closed) and output Q0.1 turns the RUN indicator on. The normally closed Q0.0 contacts on Network 3 switch to a logic 0 (open) and the STOP indicator light connected to output Q0.2 is now off.

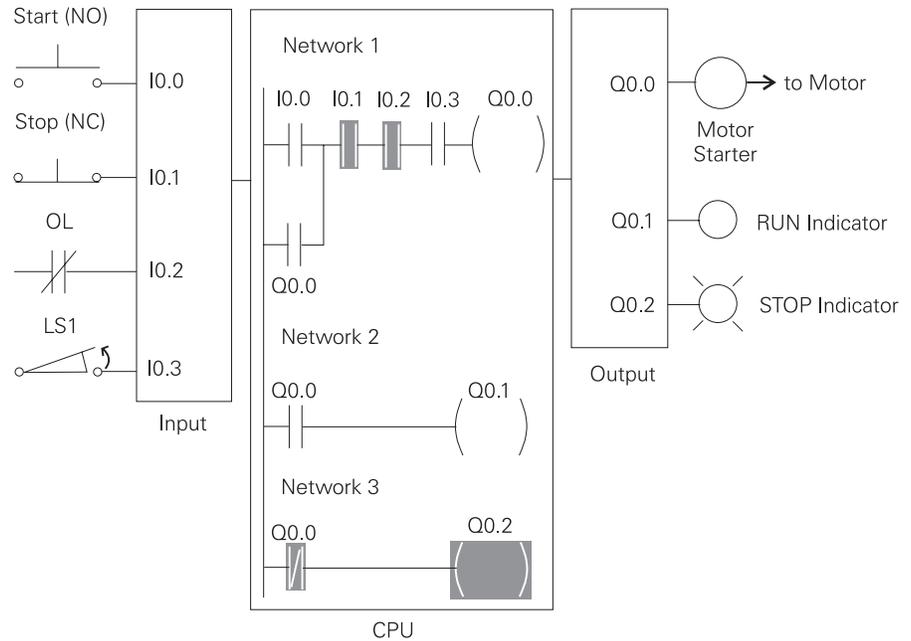


Adding a Limit Switch

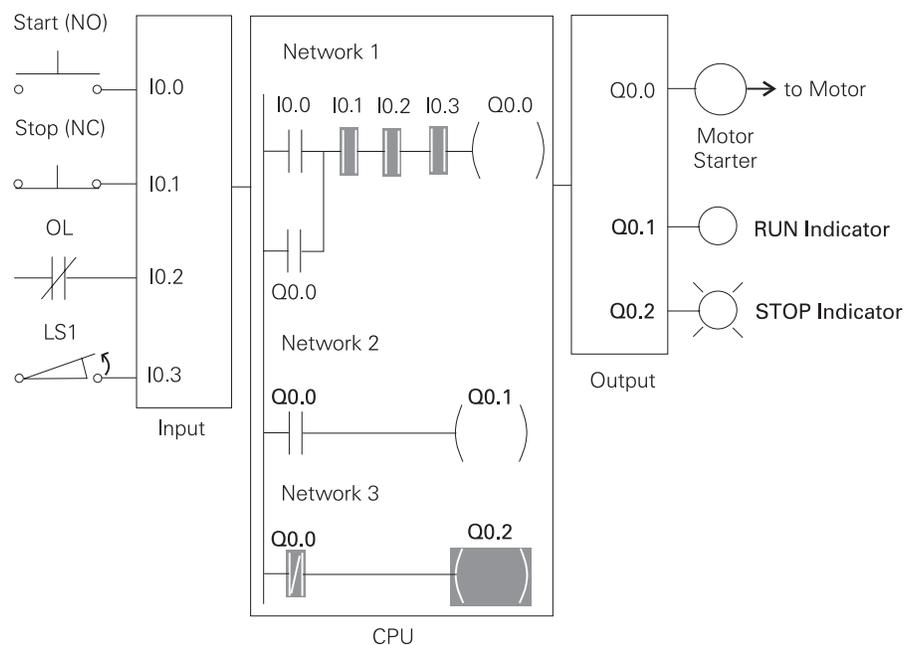
The application can be further expanded by adding a limit switch with normally open contacts to input I0.3.



A limit switch could be used to stop the motor or prevent the motor from being started. An access door to the motor, or its associated equipment, is one example of a limit switch's use. If the access door is open, the normally open contacts of LS1 connected to input I0.3 are open and the motor will not start.

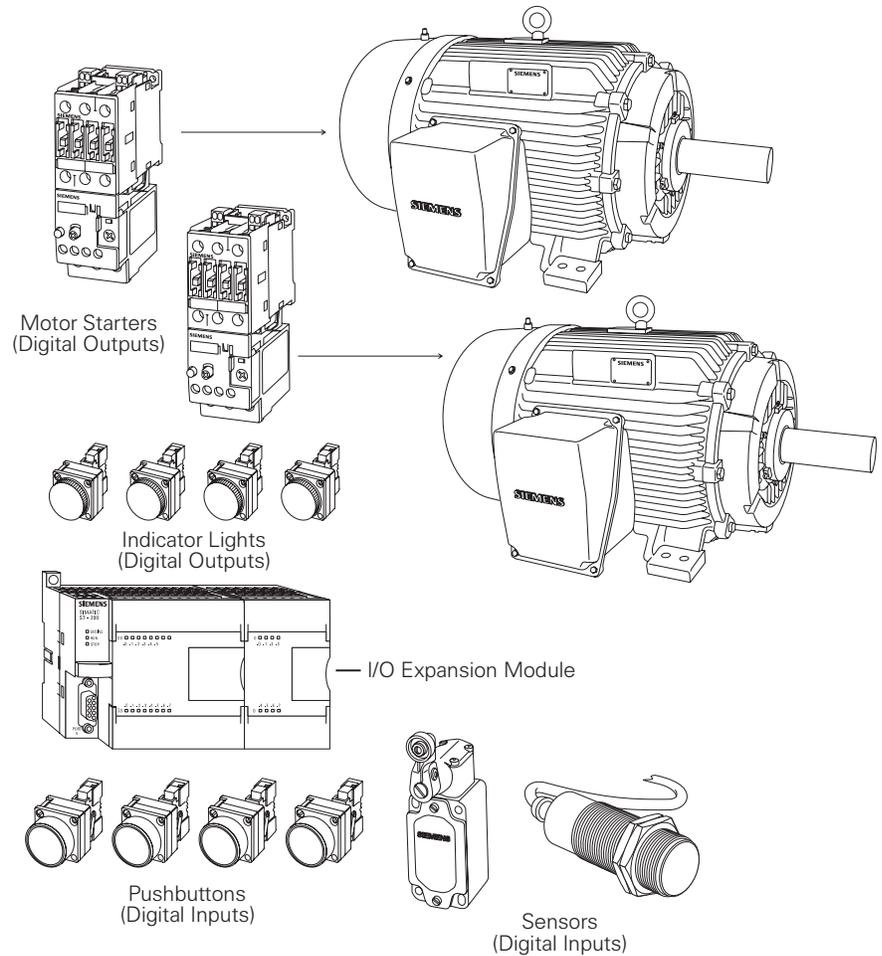


When the access door is closed, the normally open contacts on the limit switch (LS1) are closed. Input I0.3 is now on (logic 1), and the motor will start when the Start pushbutton is pressed.



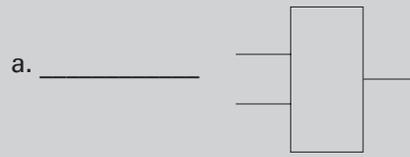
Expansion

The PLC program can be expanded to accommodate many commercial and industrial applications. Additional Start/Stop pushbuttons and indicator lights can be added for remote operation, or control of a second motor starter and motor. Overtravel limit switches can be added along with proximity switches for sensing object position. In addition, expansion modules can be added to further increase the I/O capability. The applications are only limited by the number of I/Os and amount of memory available on the PLC.



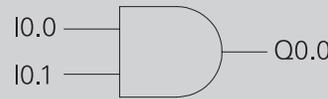
Review 4

1. Identify the following symbols:



2. Complete the following tables:

And (A) Function



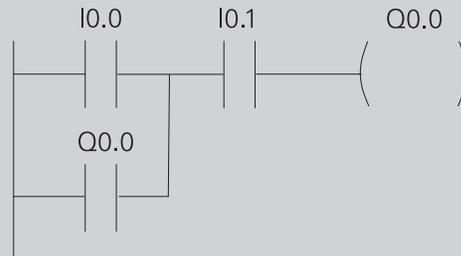
I0.0	I0.1	Q0.0
0	0	a. _____
0	1	b. _____
1	0	c. _____
1	1	d. _____

Or (O) Function



I0.4	I0.5	Q0.1
0	0	e. _____
0	1	f. _____
1	0	g. _____
1	1	h. _____

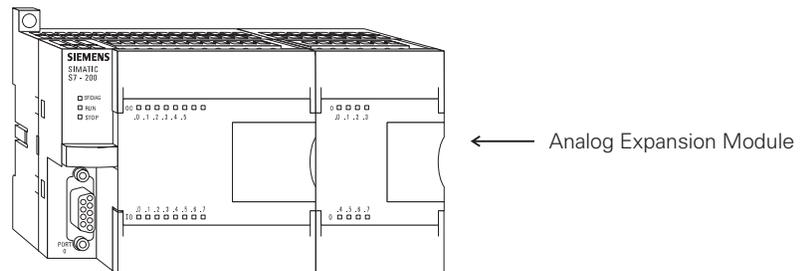
3. In the following instruction Q0.0 will be true (logic 1) when _____ or _____ is true, and when _____ is true.



Analog Inputs and Outputs

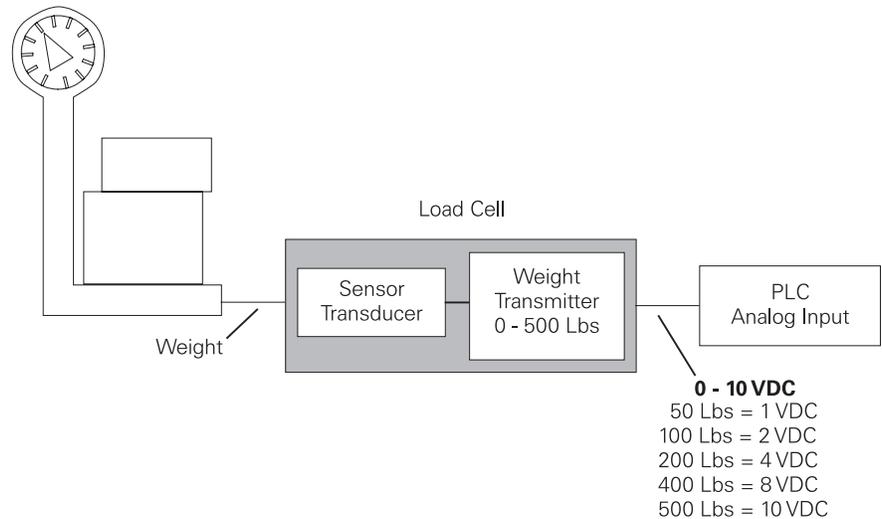
PLCs must also work with continuous or analog signals. Typical analog signals are 0 - 10 VDC or 4 - 20 mA. Analog signals are used to represent changing values such as speed, temperature, weight, and level. A PLC cannot process these signals in an analog form. The PLC must convert the analog signal into a digital representation. An expansion module, capable of converting the analog signal, must be used. The S7-200 analog modules convert standard voltage and current analog values into a 12-bit digital representation. The digital values are transferred to the PLC for use in register or word locations.

In addition, analog modules are available for use with thermocouple and RTD type sensors used in to achieve a high level of accuracy in temperature measurement.

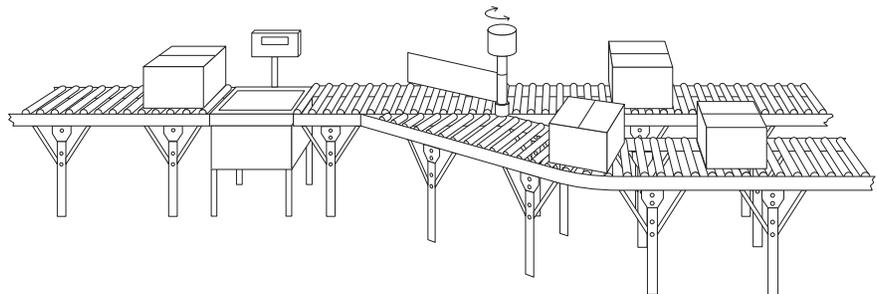


Application Example

A field device that measures a varying value is typically connected to a transducer. In the following example a scale is connected to a load cell. A load cell is a device that takes a varying value and converts it to a variable voltage or current output. In this example the load cell is converting a value of weight into a 0 - 10 VDC output. The output value depends entirely on the manufactured specifications for the device. This load cell outputs 0 - 10 VDC for a 0 - 500 Lbs input. The 0 - 10 VDC load cell output is connected to the input of an analog expansion module.

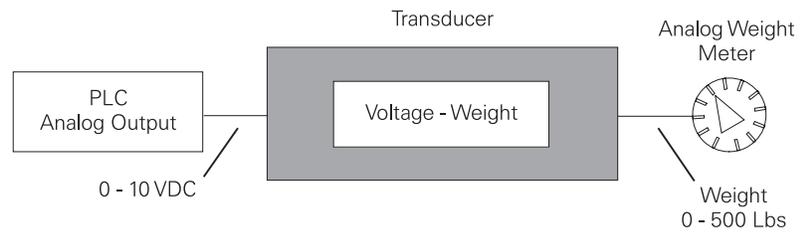


The example application can be expanded to include a conveyor system with a gate to direct packages of varying weight. As packages move along the conveyor they are weighed. A package that weighs at or greater than a specified value is routed along one conveyor path. A package that weighs less than a specified value is routed along another conveyor path, where it will later be inspected for missing contents.



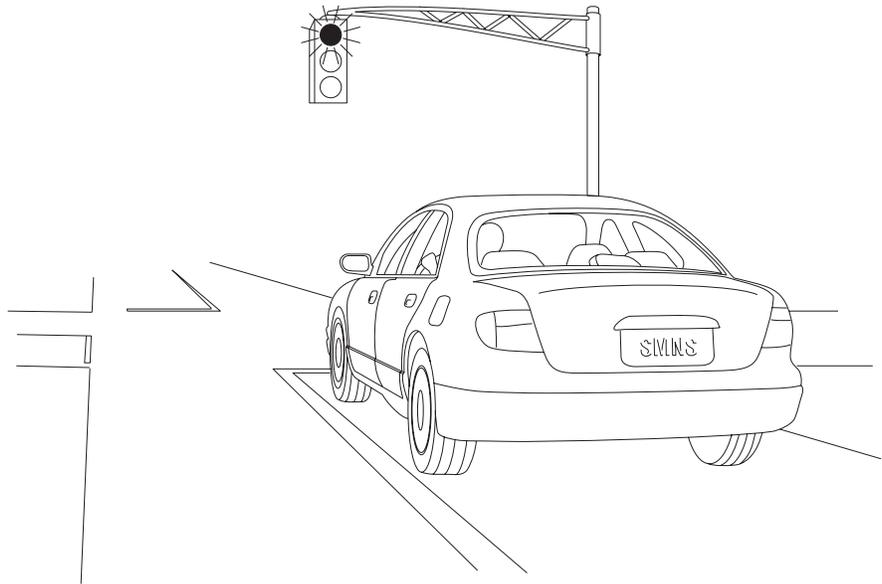
Analog Outputs

Analog outputs are used in applications requiring control capability of field devices which respond to continuous voltage or current levels. Analog outputs may be used as a variable reference for control valves, chart recorders, electric motor drives, analog meters, and pressure transducers. Like analog inputs, analog outputs are generally connected to a controlling device through a transducer. The transducer takes the voltage signal and, depending on the requirement, amplifies, reduces, or changes it into another signal which controls the device. In the following example a 0 - 10 VDC signal controls a 0 - 500 Lbs. scale analog meter.

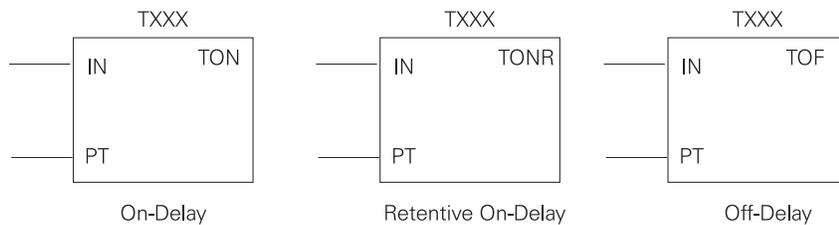


Timers

Timers are devices that count increments of time. Traffic lights are one example where timers are used. In this example timers are used to control the length of time between signal changes.



Timers are represented by boxes in ladder logic. When a timer receives an enable, the timer starts to time. The timer compares its current time with the preset time. The output of the timer is a logic 0 as long as the current time is less than the preset time. When the current time is greater than the preset time the timer output is a logic 1. S7-200 uses three types of timers: On-Delay (TON), Retentive On-Delay (TONR), and Off-Delay (TOF).

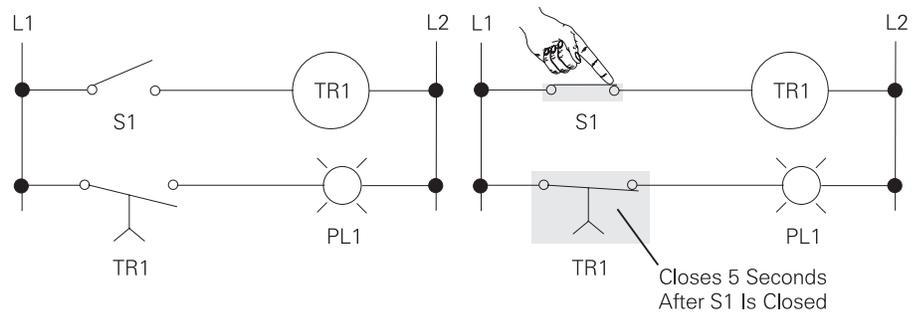


S7-200 Timers

S7-200 timers are provided with resolutions of 1 millisecond, 10 milliseconds, and 100 milliseconds. The maximum value of these timers is 32.767 seconds, 327.67 seconds, and 3276.7 seconds, respectively. By adding program elements, logic can be programmed for much greater time intervals.

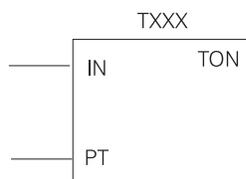
Hard-Wired Timing Circuit

Timers used with PLCs can be compared to timing circuits used in hard-wired control line diagrams. In the following example, a normally open (NO) switch (S1) is used with a timer (TR1). For this example the timer has been set for 5 seconds. When S1 is closed, TR1 begins timing. When 5 seconds have elapsed, TR1 will close its associated normally open TR1 contacts, illuminating pilot light PL1. When S1 is open, deenergizing TR1, the TR1 contacts open, immediately extinguishing PL1. This type of timer is referred to as ON delay. ON delay indicates that once a timer receives an enable signal, a predetermined amount of time (set by the timer) must pass before the timer's contacts change state.

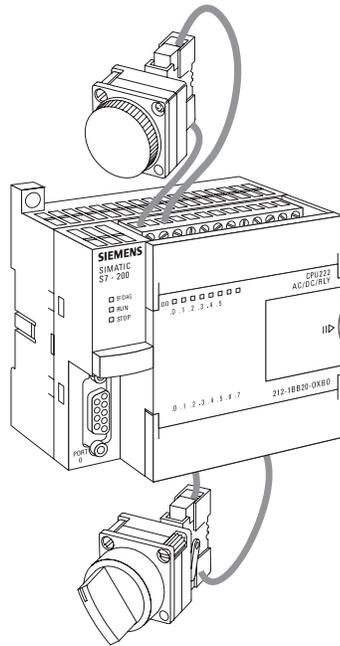


On-Delay (TON)

When the On-Delay timer (TON) receives an enable (logic 1) at its input (IN), a predetermined amount of time (preset time - PT) passes before the timer bit (T-bit) turns on. The T-bit is a logic function internal to the timer and is not shown on the symbol. The timer resets to the starting time when the enabling input goes to a logic 0.



In the following simple timer example, a switch is connected to input I0.3, and a light is connected to output Q0.1.



When the switch is closed input 4 becomes a logic 1, which is loaded into timer T37. T37 has a time base of 100 ms (.100 seconds). The preset time (PT) value has been set to 150. This is equivalent to 15 seconds (.100 x 150). The light will turn on 15 seconds after the input switch is closed. If the switch were opened before 15 seconds had passed, then reclosed, the timer would again begin timing at 0.

