

Basic PLC Programming 1

Chapter 3

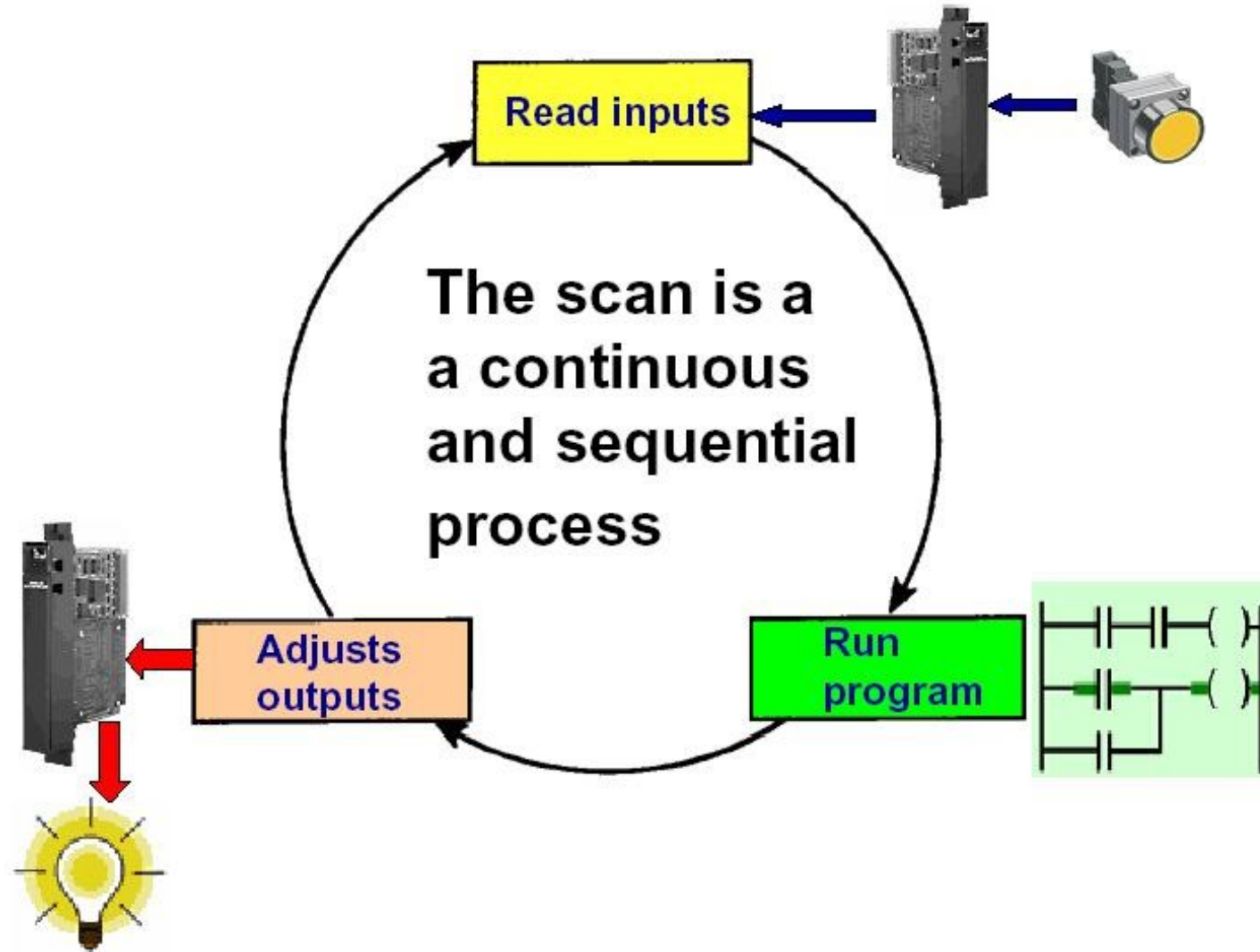
1

Table of Content

- PLC Addressing
- Basic Input Instructions
- Basic Output Instructions

Review

Scan Process



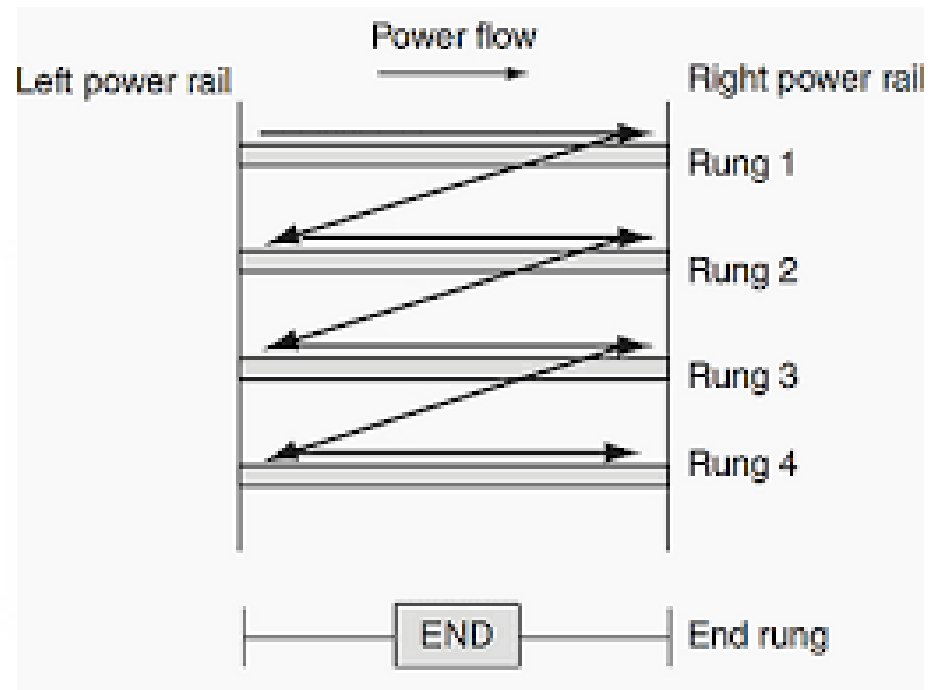
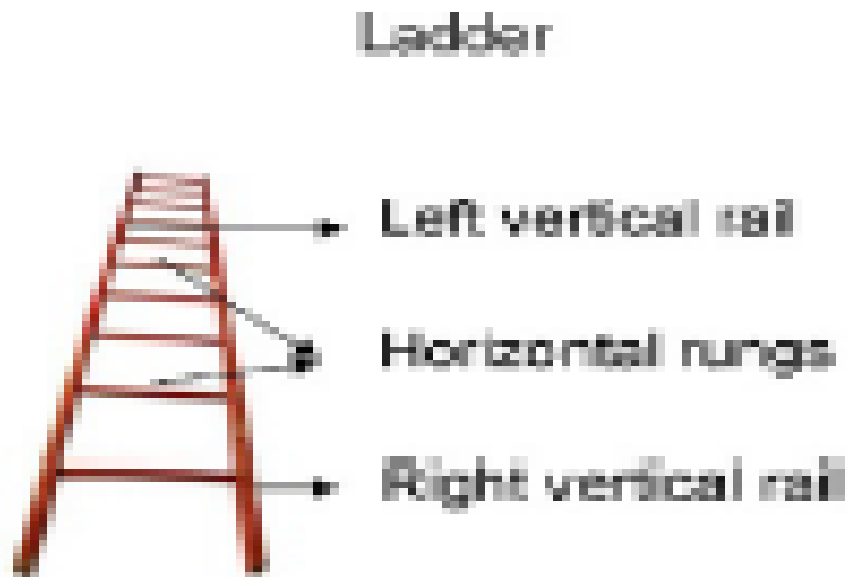
Review

Name	Equation	Truth table	log. symbol	pneumatic realisation	electr. realisation	electron. realisation															
Antivalence (exclusive OR)	$I1 \wedge \overline{I2}$ $\overline{I1 \wedge I2} = O$	<table><tr><th>I1</th><th>I2</th><th>O</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	I1	I2	O	0	0	0	0	1	1	1	0	1	1	1	0				
I1	I2	O																			
0	0	0																			
0	1	1																			
1	0	1																			
1	1	0																			
Equivalence	$I1 \wedge I2$ $\overline{I1 \wedge I2} = O$	<table><tr><th>I1</th><th>I2</th><th>O</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	I1	I2	O	0	0	1	0	1	0	1	0	0	1	1	1				
I1	I2	O																			
0	0	1																			
0	1	0																			
1	0	0																			
1	1	1																			
NAND	$\overline{I1 \wedge I2} = O$	<table><tr><th>I1</th><th>I2</th><th>O</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	I1	I2	O	0	0	1	0	1	1	1	0	1	1	1	0				
I1	I2	O																			
0	0	1																			
0	1	1																			
1	0	1																			
1	1	0																			
NOR	$\overline{I1 \vee I2} = O$	<table><tr><th>I1</th><th>I2</th><th>O</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	I1	I2	O	0	0	1	0	1	0	1	0	0	1	1	0				
I1	I2	O																			
0	0	1																			
0	1	0																			
1	0	0																			
1	1	0																			

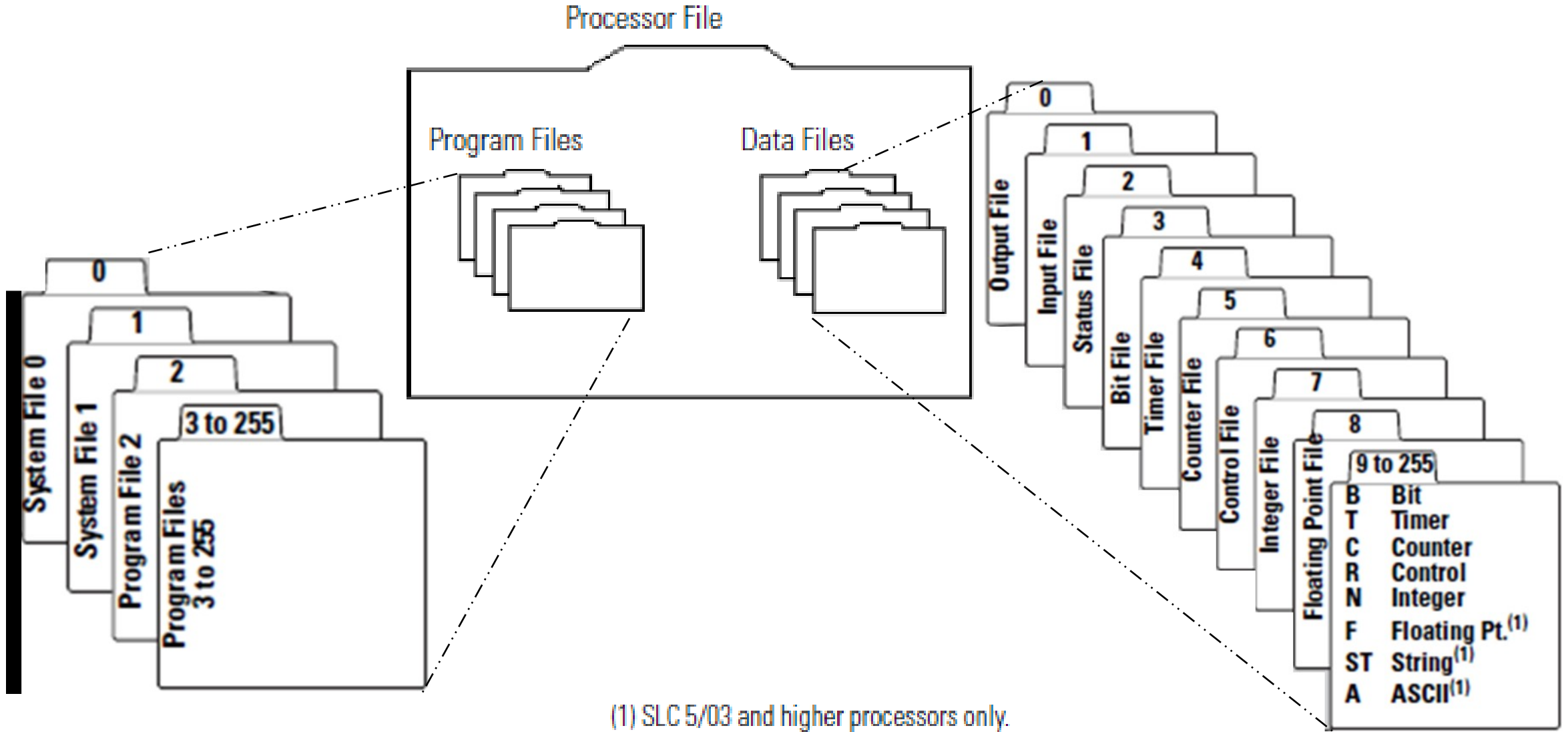
Name	Equation	Truth table	log. symbols	pneumatic realisation	electr. realisation	electron. realisation															
Identity	$I = A$	<table><tr><td>I</td><td>O</td></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	I	O	0	0	1	1													
I	O																				
0	0																				
1	1																				
Negation	$\bar{I} = O$	<table><tr><td>I</td><td>O</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	I	O	0	1	1	0													
I	O																				
0	1																				
1	0																				
electron. realisation	$2 = O$	<table><tr><td>I1</td><td>I2</td><td>O</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	I1	I2	O	0	0	0	0	1	0	1	0	0	1	1	1				
	I1	I2	O																		
0	0	0																			
0	1	0																			
1	0	0																			
1	1	1																			
$2 = O$	<table><tr><td>I1</td><td>I2</td><td>O</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	I1	I2	O	0	0	0	0	1	1	1	0	1	1	1	1					
I1	I2	O																			
0	0	0																			
0	1	1																			
1	0	1																			
1	1	1																			

The ladder diagram (LD)

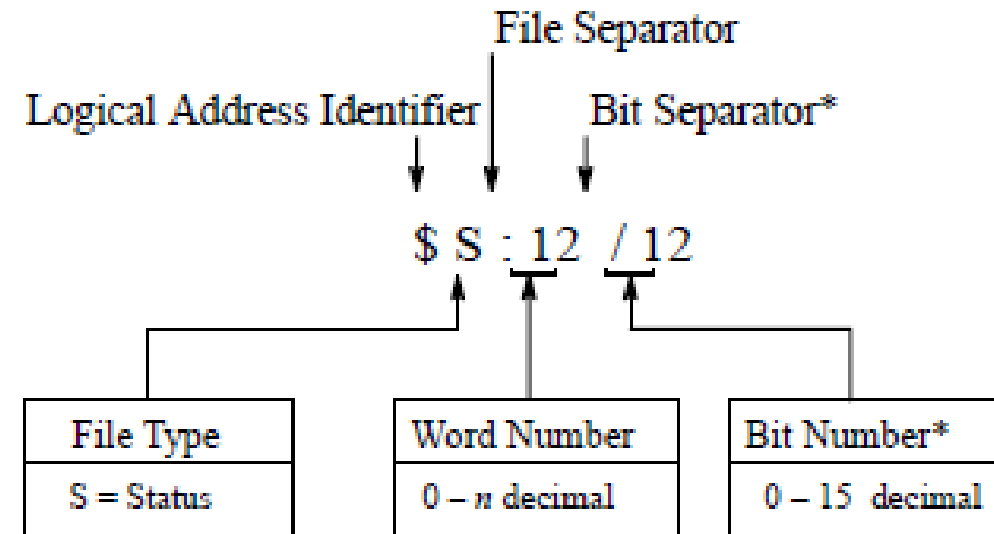
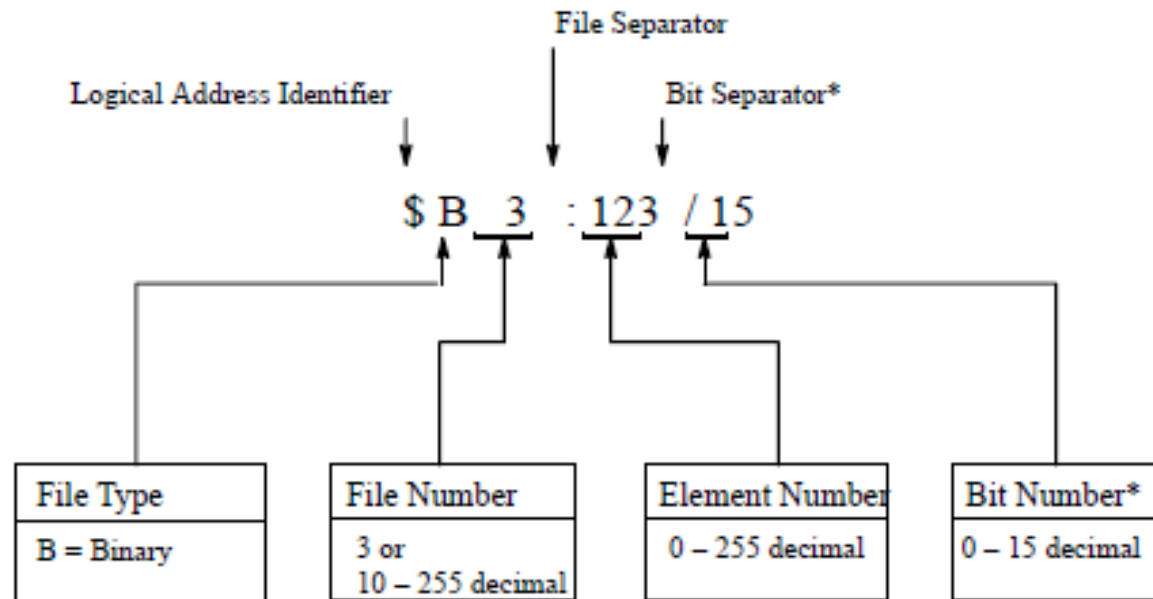
- The ladder diagram (LD) originated in the graphical representation of electrical control systems using relays (Hard Wired). It is mostly used for automation and is ideal for sequential control with interlocks.
- The name ladder diagram is derived from the program's similarity to a ladder with two vertical rails and a series of horizontal rungs between them.



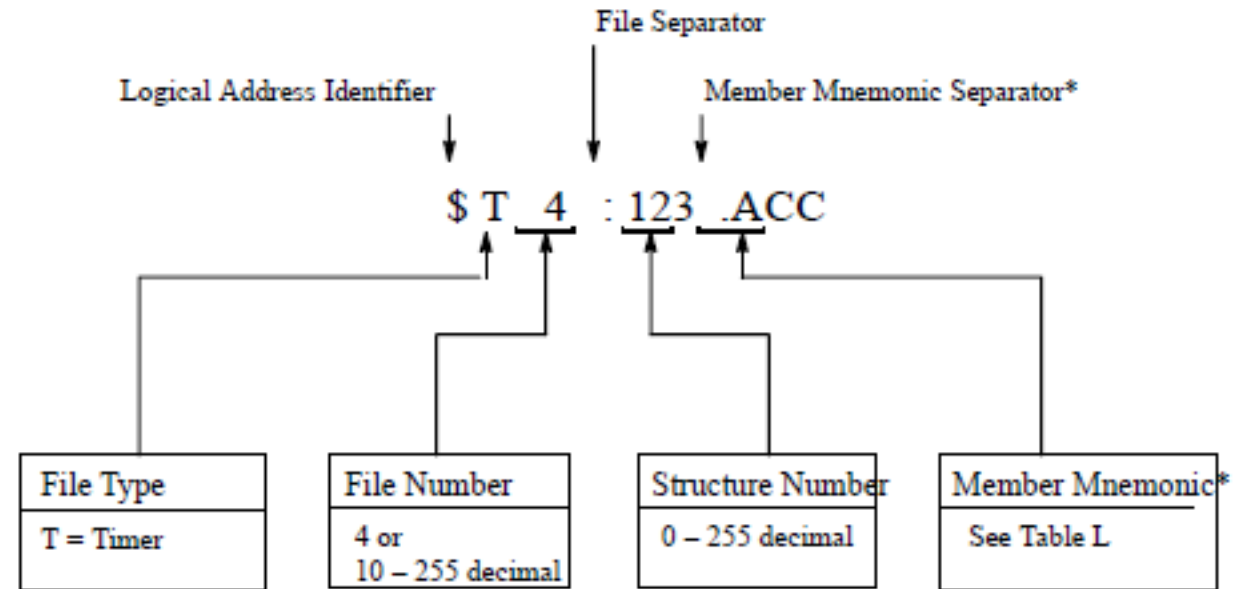
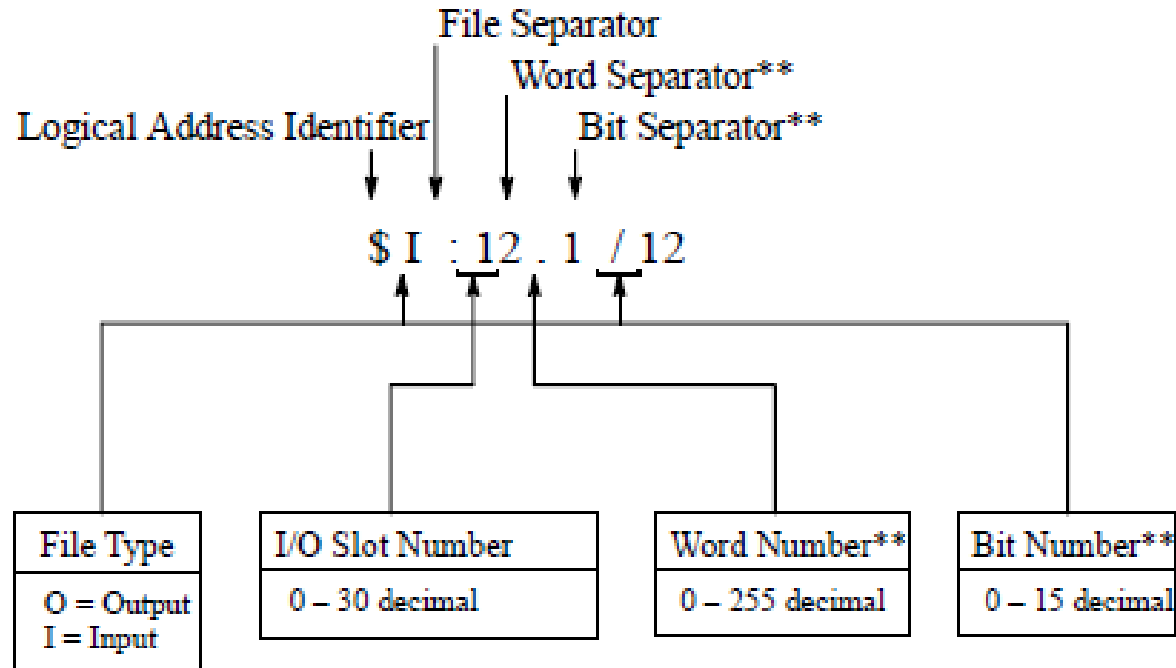
Allen Bradley Memory Map



Element Addressing



Structure Addressing

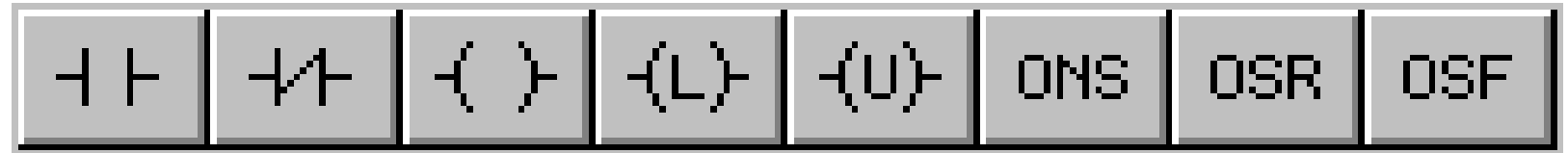


Bit Instructions Overview

- ◉ Bit instructions operate on a single bit of data. During operation, the processor may set or reset the bit, based on logical continuity of ladder rungs.
- ◉ You can address a bit as many times as your program requires.
- ◉ Bit instructions are used with the following data files.
 - Output/Input Files
 - Status File
 - Bit File
 - Timer File
 - Counter File



Ladder Diagram



Definition

- ◉ **XIC - Examine if Closed**
- ◉ **XIO - Examine if Open**
- ◉ **OTE - Output Energize**
- ◉ **OTL - Output Latch**
- ◉ **OTU- Output Unlatch**
- ◉ **OSR - One-Shot Rising**

XIC Examine if Closed

- **Symbol**



- **Definition:** Examines a bit for an On condition

- Use the XIC instruction in your ladder logic to determine if a bit is ON.

- 0 = False
- 1 = True

- **Devices**

- Start push buttons
- Selectors
- Limit switch
- Proximity switch
- Light
- Internal bit

	The status of the instruction is:		
	XIC Examine If Closed	XIO Examine If Open	OTE Output Energize
If the data table bit is:			
Logic 0	False	True	False
Logic 1	True	False	True

XIO Examine if Open

▪ **Symbol** 




▪ **Definition:** Examines a bit for an off condition.

▪ Use an XIO instruction in your ladder logic to determine if a bit is off.

- 0 = True
- 1 = False

▪ **Devices**

- Stop push buttons
- motor overload
- Limit switch
- Proximity switch
- Internal bit

	The status of the instruction is:		
	XIC Examine If Closed	XIO Examine If Open	OTE Output Energize
If the data table bit is:			
Logic 0	False	True	False
Logic 1	True	False	True

OTE Output Energize

■ Symbol



■ Definition: Turns a bit on or off .

- Use OTE instruction in your ladder logic to turn on a bit when rung condition is evaluated as true.
- OTE instructions are reset when:
 - the SLC enters or returns to the REM Run or REM Test mode or power is restored.
 - the OTE is programmed within an inactive or false Master Control Reset (MCR) zone.

■ Devices

- Light
- Motor run signal
- Internal bits

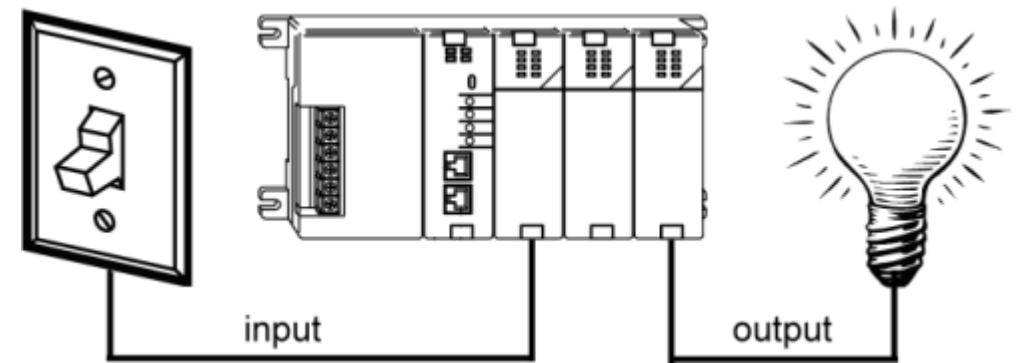
	The status of the instruction is:		
	XIC Examine If Closed	XIO Examine If Open	OTE Output Energize
If the data table bit is:			
Logic 0	False	True	False
Logic 1	True	False	True

Example 1

- Construct a simple circuit with one switch as an input and one output as a light. When the switch is on the light will go on and when the switch is off the light will turn off.

- Input / Output table**

- Switch I: 1.0/0
- Light O: 2.0/0

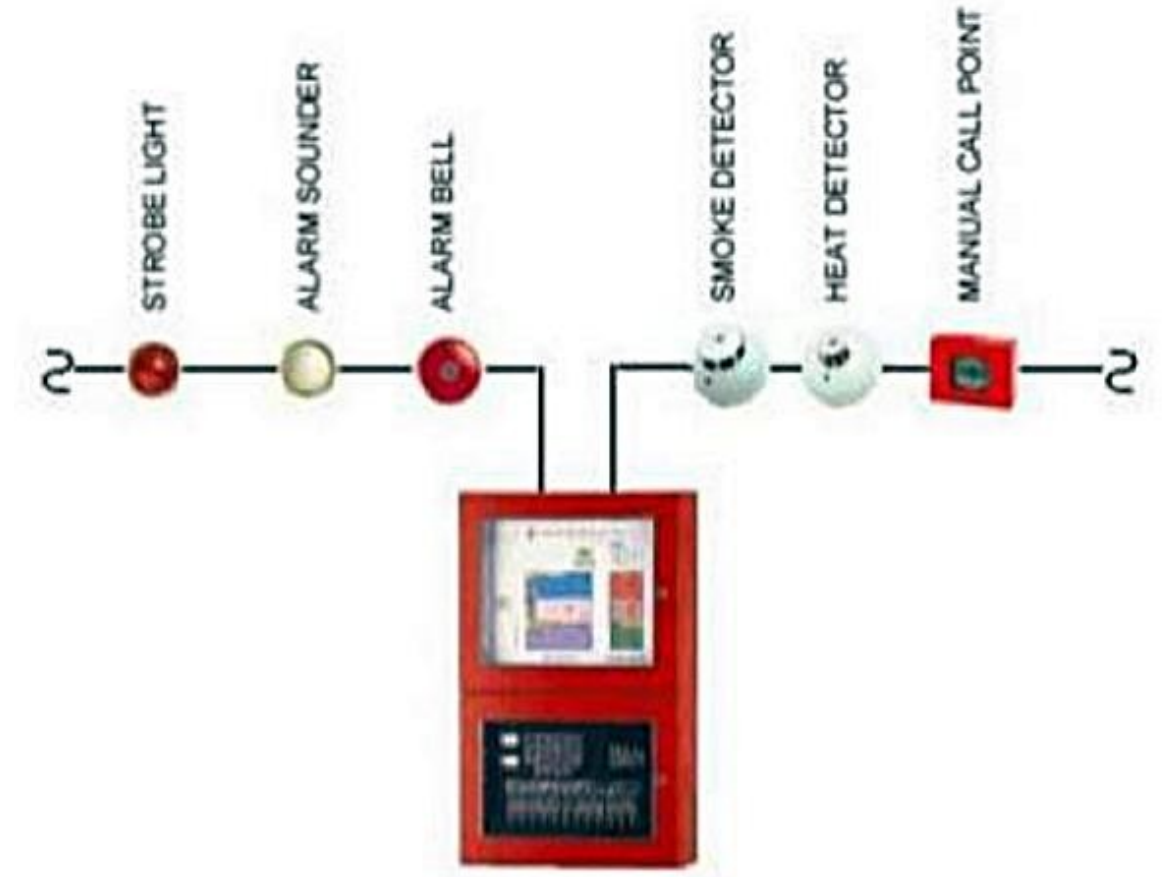


- Write a Ladder Logic Solution**

- When the switch is on the input Address: I:1.0/0 will be on and hence the rung will be energized hence the outputs on address O:2.0/0 will be energize and the light will go on.
- When the switch is off the rung will be de-energized and the light will go off.

Example 2

- Building an alarm system. Let's assume that we have 3 signals (Inputs) that we get from the field that warn us about some kind of fire danger.
 - Any 1 signal, turn the warning light on;
 - Any 2 signals will sound the Alarm.
 - All 3 signals, will start the water spray system
- **Input / Output table**
 - Alarm 1 I:1.0/7
 - Alarm 2 I:1.0/8
 - Alarm 3 I:1.0/9
 - Alarm Sound O:2.0/7
 - Alarm Light O:2.0/8
 - Spray System O:2.0/9
- **Ladder Logic Solution:**

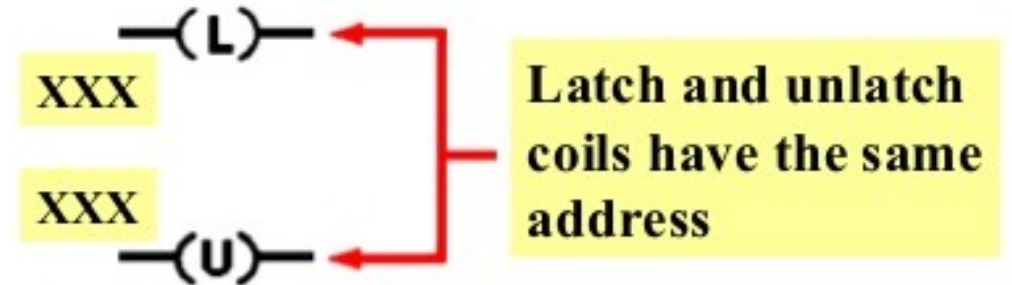


OTL Output Latch

- Usually it latch a signal with a condition and then unlatch it when a different condition becomes true. Most of time the Latch / Unlatch go together.



- Symbol**

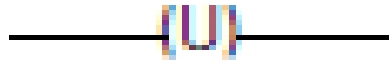


- Definition:**

- Turns a bit on when the rung is executed, and this bit retains its state when the rung is not executed or a power cycle power occurs.
- OTL is a retentive output instruction. OTL can only turn on a bit. This instruction is usually used with OUT, both OTL and OTU addressing the same bit.
- Ladder logic can examine a bit controlled by OTL as often as necessary.
- When you assign an address to the OTL instruction that corresponds to the address of a physical output, the output device wired to the screw terminal is energized when the bit is set. When rung conditions become false, the bit remains set and the corresponding output device remains energized.

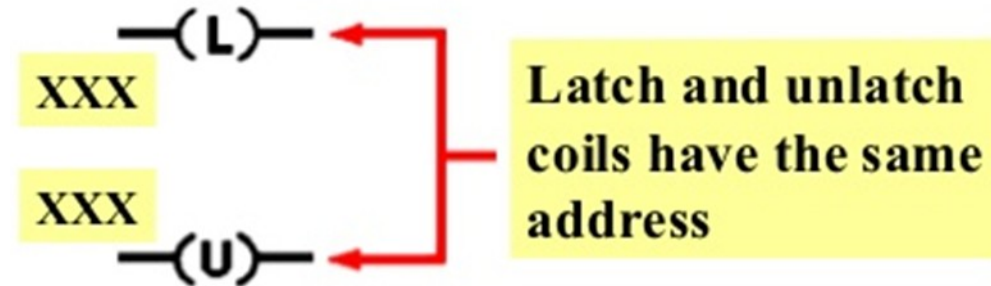
OTU Output Unlatch

■ Symbol



■ Definition:

- Turns a bit off when the rung is executed, and this bit retains its state when the rung is not executed or when power cycle occurs.
- OTU is a retentive output instruction. OTU can only turn off a bit. This instruction is usually used with OTL, with both OTL and OTU addressing the same bit.
- Ladder logic can examine a bit controlled by OTU as often as necessary.
- When you assign an address to the OTU instruction that corresponds to the address of a physical output, the output device wired to the screw terminal is de-energized when the bit is cleared.
- The unlatch instruction tells the controller to turn off the addressed bit. Thereafter, the bit remains off, regardless of the rung condition, until it is turned on.



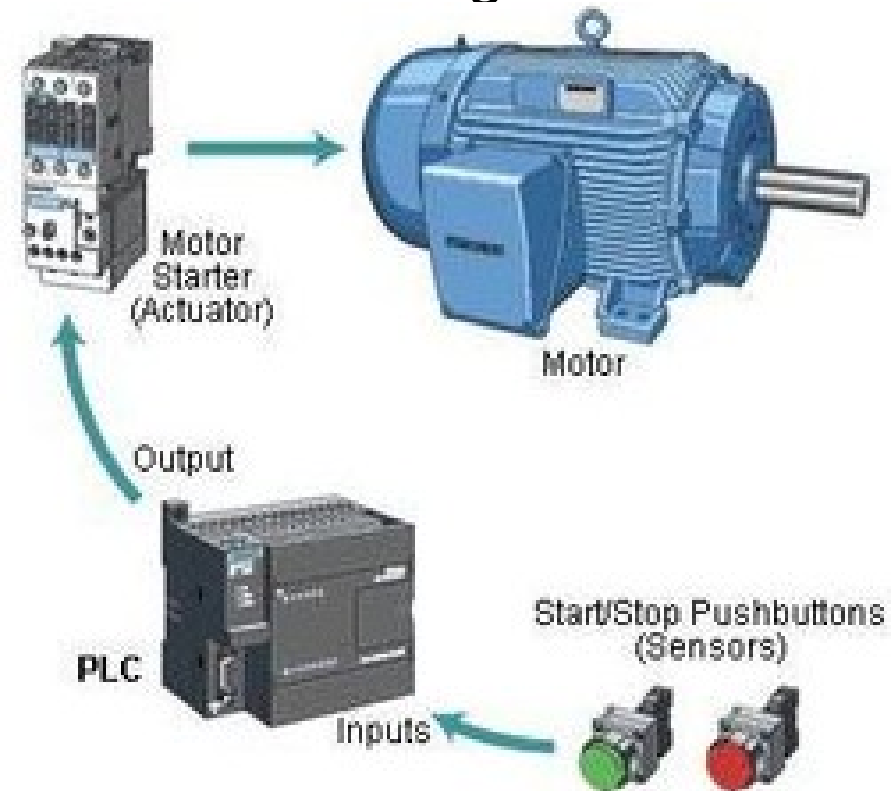
Example 3

- You need to start/stop the motor. When the start button is pushed we want to start the motor, and when the stop button is pushed we want to stop the motor. Mind you when you the push buttons is pushed the input will go on as long as the button is pushed otherwise it will go off.

- **Input / Output table**

- Start Button I:1.0/1
- Stop Button I:1.0/2
- Motor O:2.0/1

- **Ladder Logic Solution**



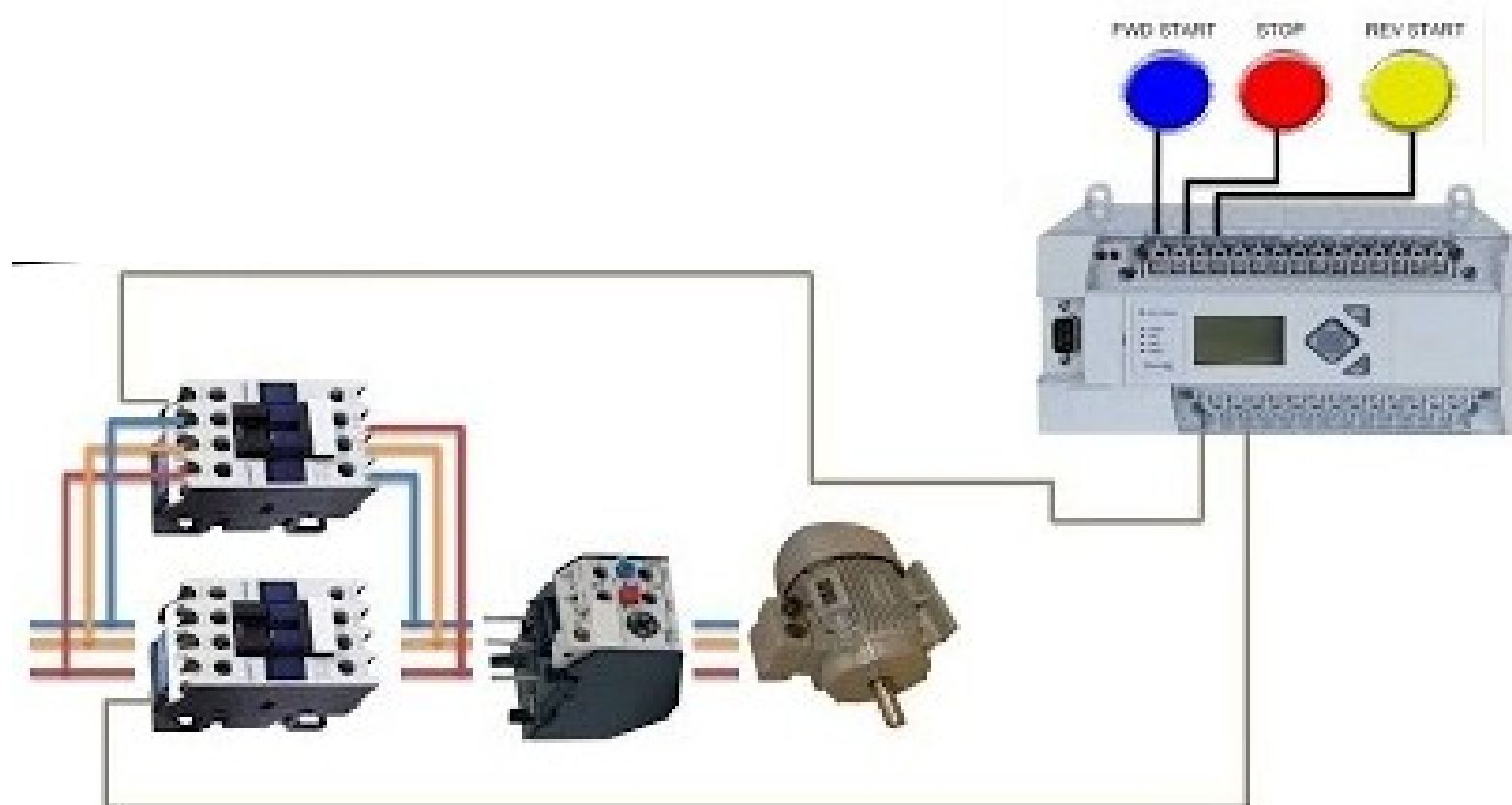
Example 4

- This example is about a bi-directional motor (Forward/ Reverse). Each direction has its own button and only one button to stop the motor. We also need to stop the motor before we change its direction.

- **Input / Output table**

- Forward Button I:1.0/3
- Reverse Button I:1.0/4
- Stop Button I:1.0/5
- Motor Forward O:2.0/3
- Motor Reverse O:2.0/4

- **Ladder Logic Solution**

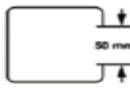



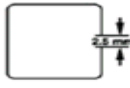

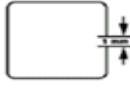

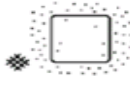

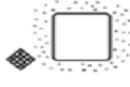





The IP Code

The IP Code, International Protection Marking, IEC standard 60529, sometimes interpreted as Ingress Protection Marking, classifies and rates the degree of protection provided against intrusion (body parts such as hands and fingers), dust, accidental contact, and water by mechanical casings and electrical enclosures. It is published by the International Electrotechnical Commission (IEC).

For example a particular cellular phone rated at IP58 means:

Reference Table of IP Rating Code

1 st Digit	Symbol	Solid Object Protection	2 nd Digit	Symbol	Water Protection
0		Not protected	0		Not protected
1		Protected against solid objects greater than 50mm	1		Protected against vertically dripping water
2		Protected against solid objects greater than 12.5mm	2		Protected against dripping water when tilted up to 15°
3		Protected against solid objects greater than 2.5mm	3		Protected against spraying water
4		Protected against solid objects greater than 1.0mm	4		Protected against splashing water
5		Protected from the amount of dust	5		Protected against jetting water
6		Dust tight	6		Protected against powerfully jetting water
<div style="text-align: center;"> IP 6 6 </div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> Code Letters ——— IP 1st Digit ——— 6 2nd Digit ——— 6 </div>			7		Protected against temporary immersion in water
			8		Protected against continuous immersion in water

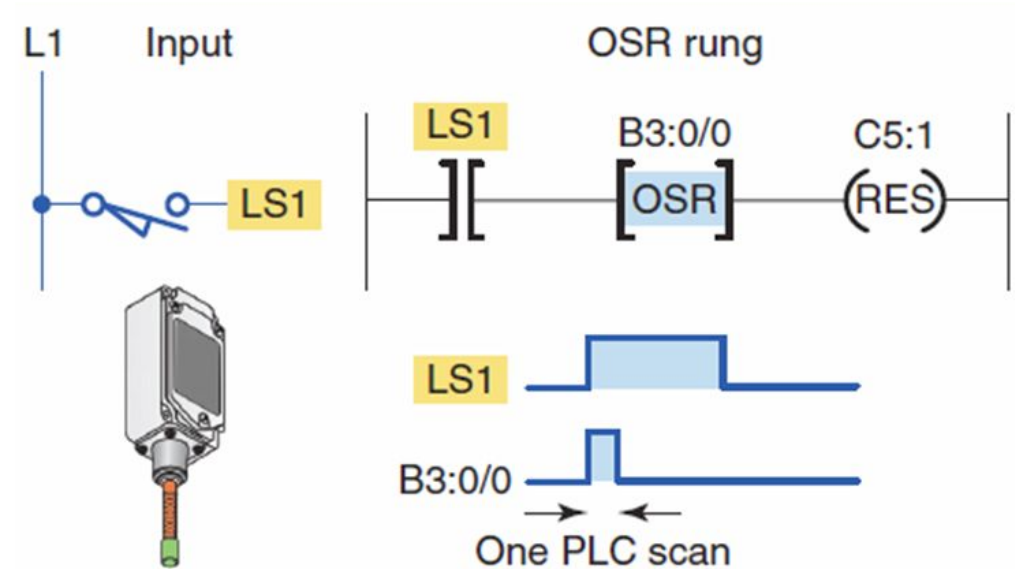
OSR One-Shot Rising

■ Symbol



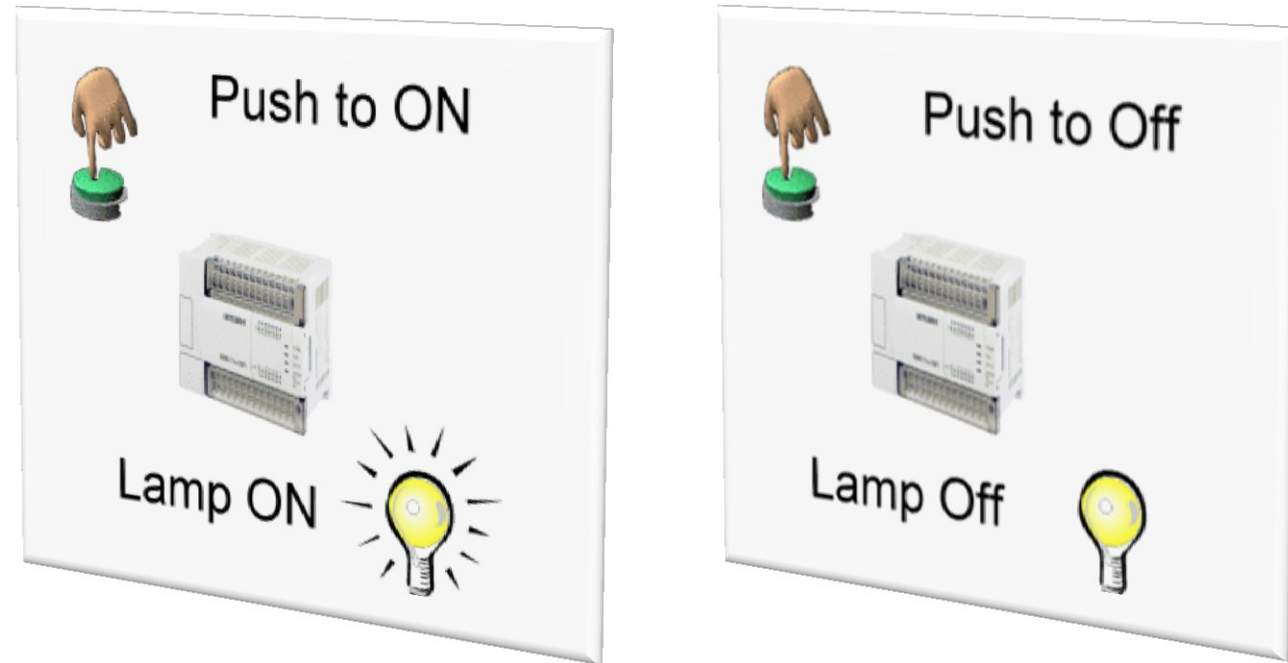
■ Definition:

- Triggers a one-time event.
- The OSR instruction is a retentive input instruction that triggers an event to occur only one time. Use the OSR instruction when an event must start based on change of state of the rung from false to true.
- When the input instruction goes from false to true, the OSR instruction conditions the rung so that the output goes true for one scan. The output goes false and remains false for successive scans until the input makes another false to true transition.



Example 5

- In this example we have one button and a light. First time we press on the button the light will go on and the second time we press the button, In other word we want this button to work exactly like a toggle button.
- **Input / Output table**
 - Switch I:1.0/6
 - Light O:2.0/6
- **Ladder Logic Solution**

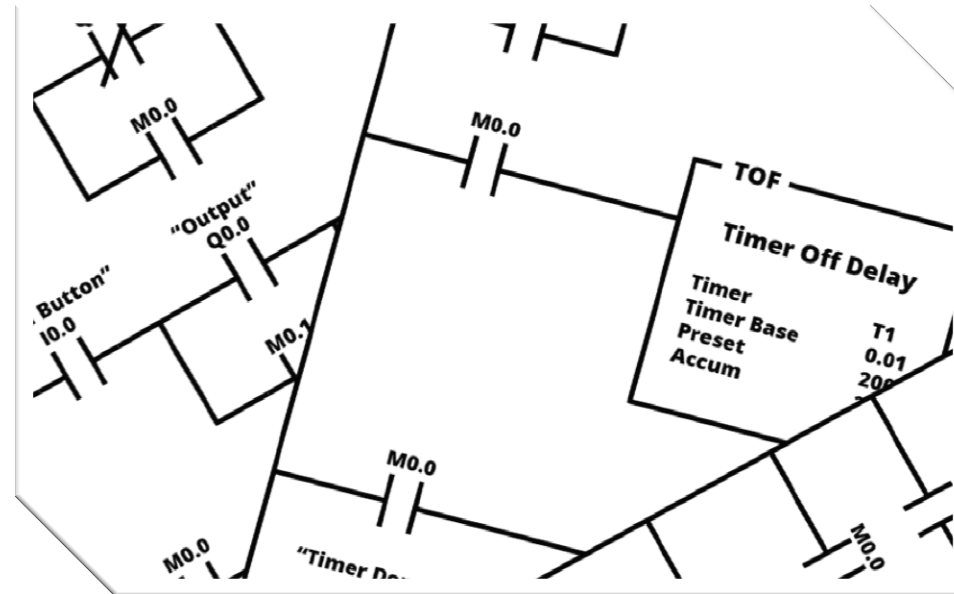


Patterns of Ladder Logic Programming

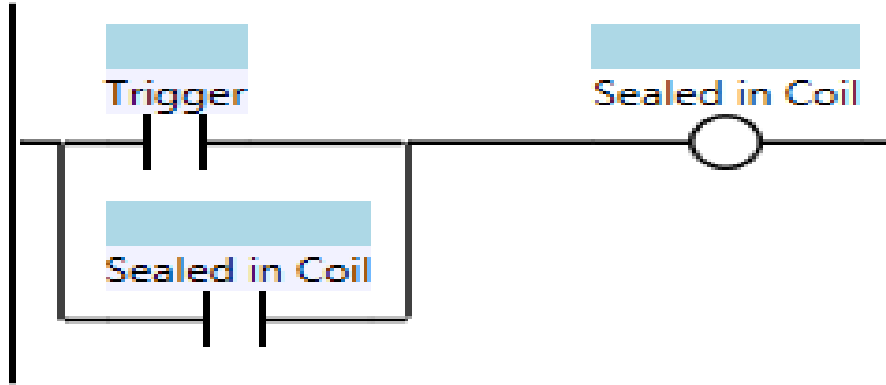
- Below a list of common patterns used in ladder logic programming. This list of ladder logic programming patterns serves two purposes:
 - First, each pattern is a tool for solving a common problem using ladder logic, and having these patterns in your toolbox will allow you to program faster and spend more time focusing on the higher level structure of your program.
 - Second, since these are common patterns, you'll start to find it easier to read other people's ladder logic, and other experienced programmers will find it easier to follow your logic.

- Bit Patterns**

- The Sealed in Coil pattern
- The State Coil pattern
- The Start/Stop Circuit pattern
- The Set/Reset pattern

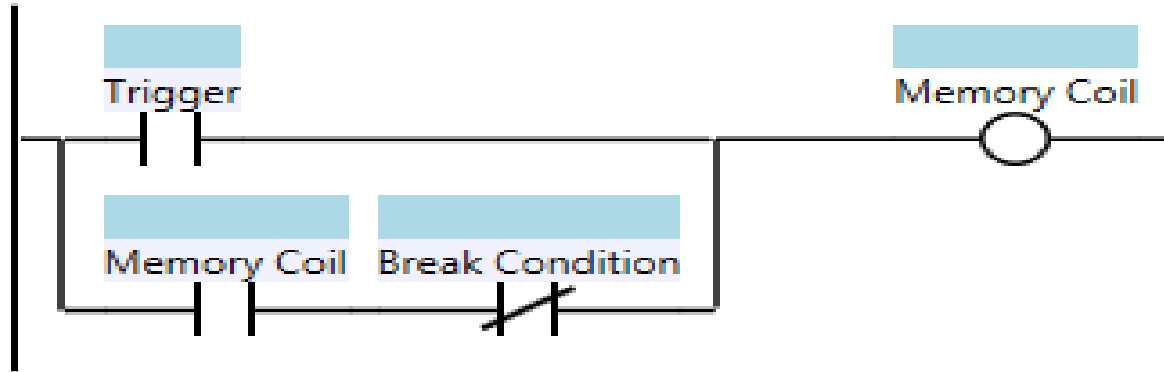


The Sealed in Coil pattern



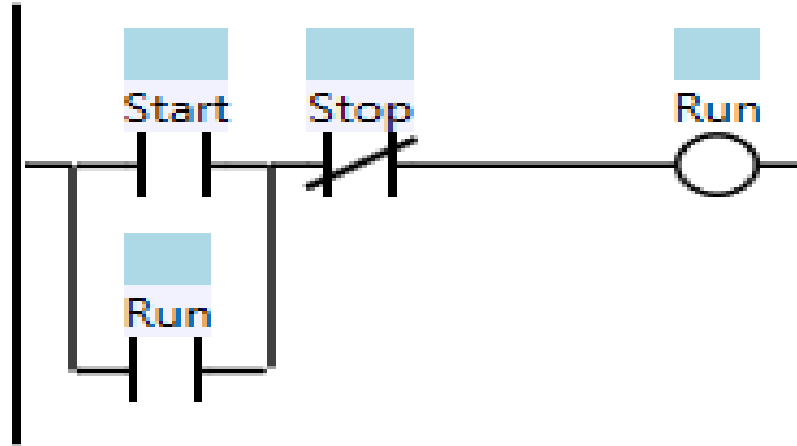
- The most basic Ladder Logic Programming Pattern is the Sealed in Coil. This pattern is the basis for remembering state in a PLC. The pattern consists of a trigger condition and a coil, where a contact from the coil branches around the trigger condition to “seal itself in” once the coil is energized:
- The “Trigger” contact can be replaced with any logic you want. It’s simply the condition that will cause the coil to energize.
- You may be questioning the utility of this pattern because after the coil is energized, it never turns off! What use is this? Well, it’s not actually true that this coil will never turn off. The coil will revert to a de-energized (off) state if the PLC loses power, or you do a hard reset on the PLC. That means the Sealed in Coil pattern is useful for remembering that something has happened *at least once* since the program has been running.

The State Coil pattern



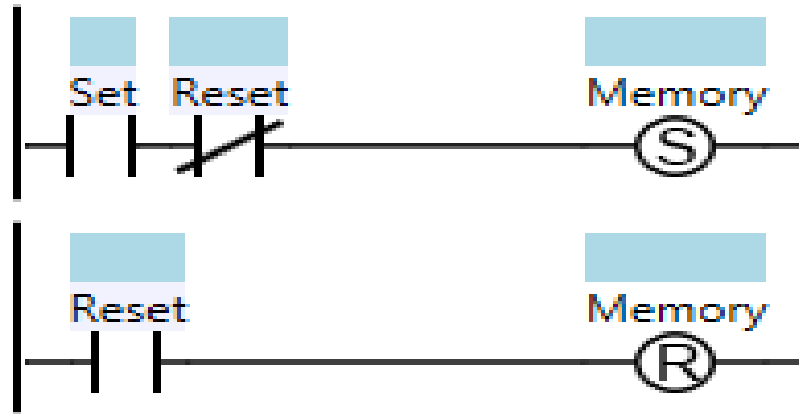
- One of the most common examples of a Ladder Logic Programming Pattern is the State Coil (sometimes called a Memory Coil). This pattern is an extension to the Sealed in Coil pattern. The pattern consists of a trigger condition, a coil, with a contact from the coil branching around the trigger condition to “seal itself in”, and then some other condition to “break the seal”:
- Like the Sealed in Coil, the State Coil will always revert to a de-energized state if the PLC loses power or the program is reset. However, unlike the Sealed in Coil, the State Coil turns on and off during normal operation of the program. The purpose of this pattern is to remember a single true/false condition representing internal state of the machine.
- You will commonly see the State Coil pattern to represent a “Fault” condition.

The Start/Stop Circuit pattern



- A very useful Ladder Logic Programming Pattern is the Start/Stop Circuit. This pattern is an extension to the Sealed in Coil pattern and is similar to the State Coil. However, where the State Coil is “trigger dominant” (i.e. the Trigger condition takes priority over the Break Condition), the Start/Stop Circuit is “stop dominant”.
- Like the Sealed in Coil, the Run coil will always revert to a de-energized (off) state if the PLC is turned off, or if the ladder logic program is reset. That’s a useful property because when starting up the machine we likely want motors, etc., to be in the off state until the logic decides to start them.

The Set/Reset pattern



- The Set/Reset pattern, also known as the Latch/Unlatch pattern or simply “Latch Bit” is for remembering some on/off state of the machine that *has to survive a power outage*.
- Note that in an Allen-Bradley PLC, you will see the terms Latch (L) and Unlatch (U) used instead of Set and Reset.
- In the logic shown above, this is a “reset dominant” Set/Reset. That’s because if both the Set and Reset conditions are active at the same time, the Memory will be reset.