2 Basic Ladder Logic Programming

Chapter Topics:

- Basic ladder logic symbols
- Ladder logic diagram
- Ladder logic evaluation
- Converting relay logic to ladder logic

OBJECTIVES

Upon completion of this chapter, you will be able to:

- Understand basic ladder logic symbols
- Write ladder logic for simple applications
- Translate relay ladder logic into PLC ladder logic

Scenario: A program with a long scan time may not detect short-duration events.

A manufacturer of small gasoline engines had an intermittent problem on the final assembly line. Sometimes, a defective engine would not be automatically removed from the line for repair at a "kick-out" station. If an operator noticed a problem with an engine, he/she inserted a bolt into a certain hole in the engine carrier. A proximity sensor before the kick-out station sensed the presence of the bolt, and the PLC activated a hydraulic solenoid to push the carrier (and engine) off the main conveyor and into the repair area. A view of this station is shown in Figure 2.1. Further investigation revealed that the duration of the **on** pulse of the proximity sensor was approximately 3/4 seconds. One PLC controlled all of the stations on the assembly line and its ladder logic program was quite large. As indicated in the PLC status, the time to scan the ladder logic program was slightly less than 1 second. Hence, it was very likely that a pulse from the proximity sensor could be undetected by the PLC processor. The proximity sensor could be **off** at the start of the ladder scan, generate an **on** pulse from a passing bolt in the carrier, and be **off** at the start of the next ladder scan.

Solution: Logic to examine the proximity sensor is placed in a ladder logic routine that is executed every ½ second. If the proximity sensor is detected to be on, an internal coil is turned on for at least 1.5 seconds. The main PLC program is changed to examine this internal coil to determine when to activate the hydraulic solenoid and push a carrier off the main conveyor.

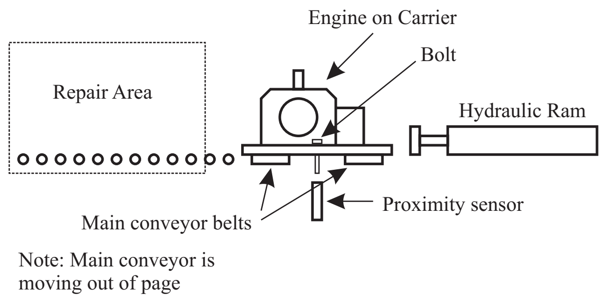


Figure 2.1. Kick-out station.

2.1 INTRODUCTION

Now that the PLC has been introduced, let us move on to programming the PLC. The first, and still most popular programming language, is ladder logic. Using examples, the language is developed from the electromechanical relay system-wiring diagram. After describing the basic symbols for the various processors covered by this text, they are combined into a ladder diagram. The subsequent section details the process of scanning a program and accessing the physical inputs and outputs. Programming with the normally closed contact is given particular attention because it is often misapplied by novice programmers. To solidify these concepts, the start/stop of a physical device is considered. Start/stop is a very common PLC application and occurs in many other contexts. An optional section on relay to PLC ladder logic conversion concludes the chapter.

2.2 SIMPLE LADDER LOGIC

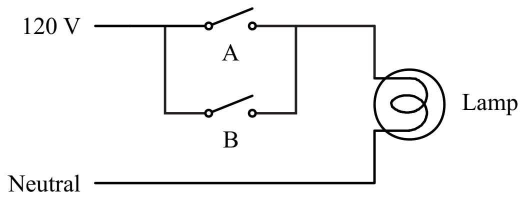
Ladder logic is the primary programming language of programmable logic controllers. Since the PLC was developed to replace relay logic control systems, it was only natural that the initial language closely resembles the diagrams used to document the relay logic. By using this approach, the engineers and technicians using the early PLCs did not need retraining to understand the program. To introduce ladder logic programming simple switch circuits are converted to relay logic and then to PLC ladder logic.

In all of the ladder logic examples used in this chapter, tags (symbols) are used for all inputs, outputs, and internal memory in the examples to avoid having to deal with input/output addressing. This addressing, treated in Chapter 3, is generally different for each PLC manufacturer.

Example 2.1. OR Circuit. Two switches labeled A and B are wired in parallel controlling a lamp as shown in Figure 2.2*a*. Implement this function as PLC ladder logic where the two switches are separate inputs.

Solution. The switch circuit action is described as, "The lamp is **on** when switch A is **on** (closed) or switch B is **on** (closed)." All possible combinations of the two switches and the consequent lamp action is shown as a truth table in Figure 2.2*b*.

To implement this function using relays, the switches A and B are not connected to the lamp directly, but are connected to relay coils labeled AR and BR whose normally-open



(a)

A	B	Lamp
off	off	off
off	on	on
on	off	on
on	on	on

(b)

Figure 2.2. Parallel switch circuit: *(a)* switch circuit; *(b)* truth table.

(NO) contacts control a relay coil, LR, whose contacts control the lamp, Figure 2.3*a*. The switches, A and B, are the inputs to the circuit. When either switch A or B is closed, the corresponding relay coil AR or BR is energized, closing a contact and supplying power to the LR relay coil. The LR coil is energized, closing its contact and supplying power to the lamp.

The output (lamp in this case) is driven by the LR relay to provide voltage isolation from the relays implementing the logic. The switches, A and B, control relay coils (AR and BR) to isolate the inputs from the logic. Also, with this arrangement, the one switch connection to an input relay can be used multiple times in the logic. A typical industrial control relay can have up to 12 poles, or sets of contacts, per coil. For example, if the AR relay has six poles (only one shown in Figure 2.3*a*), then the other five poles are available for use in the relay logic without requiring five other connections to switch A.

Before the PLC was developed, engineers had already developed a graphical electrical circuit shorthand notation for the relay circuit of Figure 2.3*a*. This notation was called a *relay ladder logic diagram*, shown in Figure 2.3*b*. The switches are shown as their usual symbol, the circles indicate the relay coils, and the NO relay contacts are shown as the vertical parallel bars.

The *PLC ladder logic* notation (Figure 2.3*c*) is shortened from the relay wiring diagram to show only the third line, the relay contacts and the coil of the output relay. The PLC ladder logic notation assumes that the inputs (switches in this example) are connected to discrete input channels (equivalent to the relay coils AR and BR in Figure 2.3*b*). Also, the actual output (lamp) is connected to a discrete output channel (equivalent to the normally open contacts of LR in Figure 2.3*b*) controlled by the coil. The label shown above a contact symbol is not the contact label, but the control for the coil that drives the contact. Also, the output for the rung occurs on the extreme right side of the rung and power is assumed to flow from left to right. The PLC ladder logic rung is interpreted as: "When input (switch) A is **on** OR input (switch) B is **on** then the lamp is **on**," which is the same as the statement describing the switch circuit in Figure 2.2*a*.

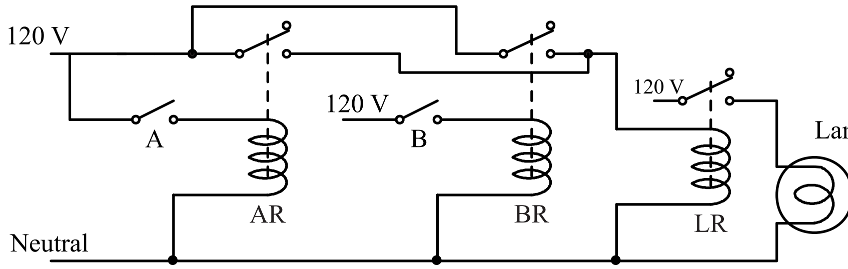
Notice that the original description of the switch circuit in Figure 2.2*a*,

The lamp is **on** when switch A is **on** or switch B is **on**.

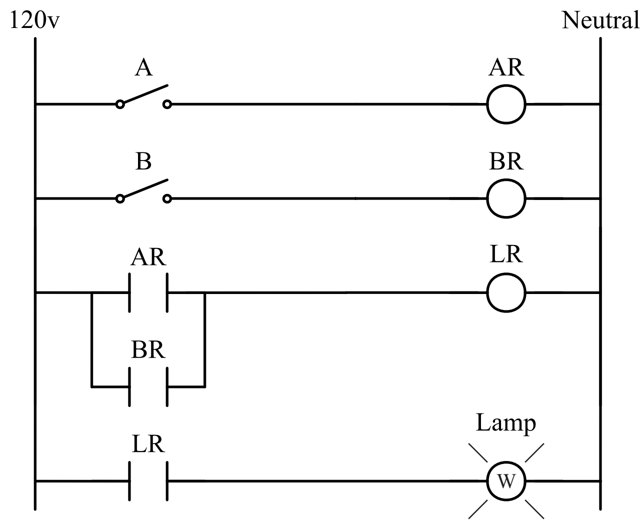
translates into a relay circuit described as

A parallel connection of **normally-open contacts**,

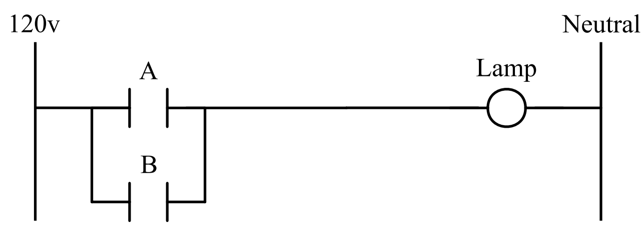
which describes the PLC ladder logic in Figure 2.3*c*.



(a)



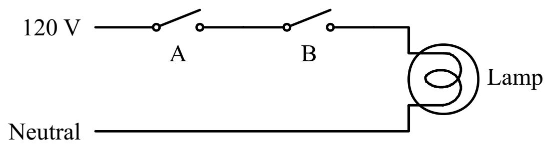
(b)



(c)

Figure 2.3. Parallel switch relay and ladder logic circuits: *(a)* equivalent relay circuit; *(b)* equivalent relay ladder logic circuit; *(c)* equivalent PLC ladder logic.

Example 2.2. AND Circuit. Two switches labeled A and B are wired in series controlling a lamp as shown in Figure 2.4*a*. Implement this function as PLC ladder logic where the two switches are separate inputs.



(a)

A	B	Lamp
off	off	off
off	on	off
on	off	off
on	on	on

(b)

Figure 2.4. Series switch circuit: *(a)* switch circuit; *(b)* truth table.

Solution. The switch circuit action is described as, “The lamp is **on** when switch A is **on** (closed) and switch B is **on** (closed).” All possible combinations of the two switches and the consequent lamp action is shown as a truth table in Figure 2.4*b*. To implement this function using relays, the only change from Example 2.1 is to wire the normally-open contacts of control relays AR and BR in series to control the light, Figure 2.5*a*. The wiring of switches A and B and the wiring of the lamp do not change. The relay circuit diagram, shown in Figure 2.5*b* is different from Figure 2.3*b* only in the third line. As for example 2.1, the PLC ladder logic notation (Figure 2.5*c*) is shortened from the relay wiring diagram to show only the third line, the relay contacts and the coil of the output relay. The PLC ladder logic rung is interpreted as: “When input (switch) A is **on** AND input (switch) B is **on** then the lamp is **on**.”

Notice that the original description of the switch circuit in Figure 2.4*a*,

The lamp is **on** when switch A is **on** and switch B is **on**.

translates into a relay circuit described as

A series connection of **normally-open contacts**,

which describes the PLC ladder logic in Figure 2.5*c*.

Example 2.3. As a third example, consider the implementation of a logical NOT function. Suppose a lamp needs to be turned **on** when switch A is **on** (closed) and switch B is **off** (open). Implement this function as PLC ladder logic where the two switches are separate inputs.

Solution. Figure 2.6 shows the truth table, relay implementation and ladder logic for this example. The only difference between the relay implementation in Figure 2.6*b* and Figure 2.5*a* is the wiring of the relay BR contacts. The logical NOT for switch B is accomplished with the normally closed (NC) contact of relay BR. The PLC ladder logic rung in Figure 2.6*c* is different from Figure 2.5*c* only in the second contact symbol. The PLC ladder logic is interpreted as: “When input (switch) A is **on** (closed) and input (switch) B is **off** (open) then the lamp is **on**.” This particular example is impossible to implement with a combination of only two normally open switches and no relays.

Notice that the original description of the Example 2.3,

The lamp is **on** when switch A is **on** and switch B is **off**.

translates into a relay circuit described as

A series connection of a **normally-open contact** and a **normally-closed contact**,

which describes the PLC ladder logic in Figure 2.6*c*.

Summarizing these three examples, one should notice that key words in the description of the operation translate into certain aspects of the solution:

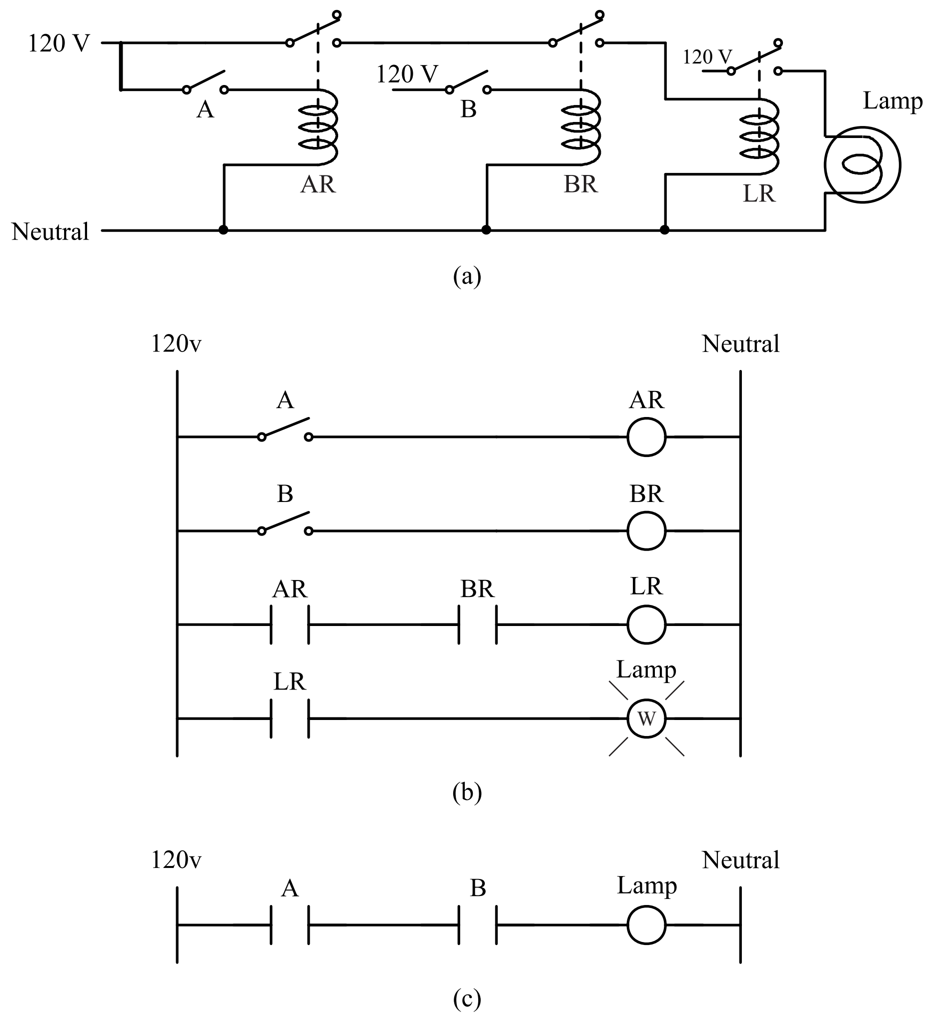


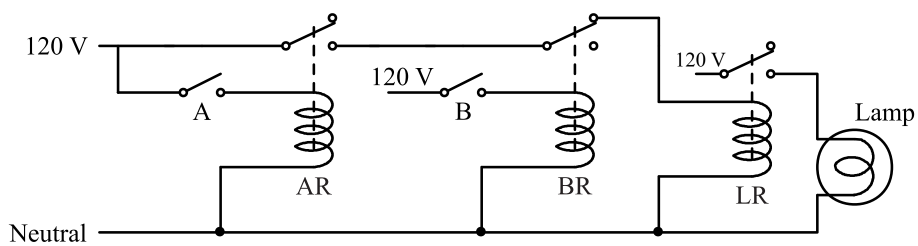
Figure 2.5. Series switch relay and ladder logic circuits: *(a)* equivalent relay circuit; *(b)* equivalent relay ladder logic circuit; *(c)* equivalent PLC ladder logic.

and → series connection of contacts
or → parallel connection of contacts
on → **normally-open** contact
off → **normally-closed** contact

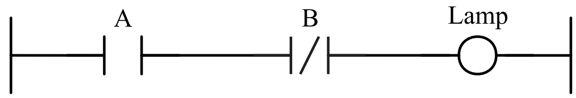
These concepts are key to being able to understand and write ladder logic. To many people these concepts appear strange and foreign at first. However, they will become more natural as one works problems. Ladder logic is a very visual and graphical language. It is very different from textual languages like C++, Fortran, Basic, and Java. In contrast, one can become proficient at ladder logic much quicker than with textual languages.

A	B	Lamp
off	off	off
off	on	off
on	off	on
on	on	off

(a)



(b)



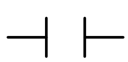
(c)

Figure 2.6. NOT function ladder logic circuits; *(a)* truth table; *(b)* equivalent relay circuit; *(c)* equivalent PLC ladder logic.

NAND and NOR logic functions are left as exercises for the interested reader. More information about the conversion between relay ladder logic and PLC ladder logic appears in section 2.8.

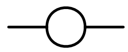
2.3 BASIC LADDER LOGIC SYMBOLS

At this point, one should start interpreting ladder logic directly and not think of its implementation with relays. As introduced by the examples in the previous section, the basic ladder logic symbols are



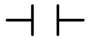
Normally open (NO) contact. Passes power (**on**) if coil driving the contact is **on** (closed).

Normally closed (NC) contact. Passes power (**on**) if coil driving the contact is **off** (open).



Output or *coil*. If any left-to-right path of contacts passes power, the output is energized. If there is no continuous left-to-right path of contacts passing power, the output is de-energized.

These symbols are ladder logic instructions that are scanned (executed) by the PLC. In order to avoid confusion, the contact symbols should be equated with certain concepts as follows:

 = **on** = Closed = True = 1

 = **off** = Open = False = 0

This crucial point will be repeated later when the use of the NC contact is clarified. Figure 2.7 is an example ladder logic diagram with the basic instructions. The first line (also called a *rung*) that determines output labeled Out1 is interpreted as follows: Out1 is **on** if inputs A, B, and C are all **on**, or if inputs A and C are **on** and input D is **off**. Notice that for Out1 to be **on** there must be a continuous electrical path through the contacts.

Every PLC manufacturer uses the instruction symbols shown in the previous paragraph. There are other contact and coil instruction symbols, but there is no universal graphic representation for these other instructions among PLC vendors. The IEC 61131-3 standard has the most contact and coil instructions and many manufacturers do not implement the full set of instructions.

The industry trend is toward using the IEC 61131-3 (formerly IEC 1131-3) standard, and so it will be the primary language of this text. Since IEC 61131-3 is only a voluntary standard, individual manufacturers have some freedom in the implementation. Therefore, the Allen-Bradley ControlLogix, Modicon, and Siemens S7 implementations of the 61131-3 standard are covered. Because of their widespread use, Allen-Bradley PLC-5/SLC-500 and GE Fanuc PLC languages are also covered.

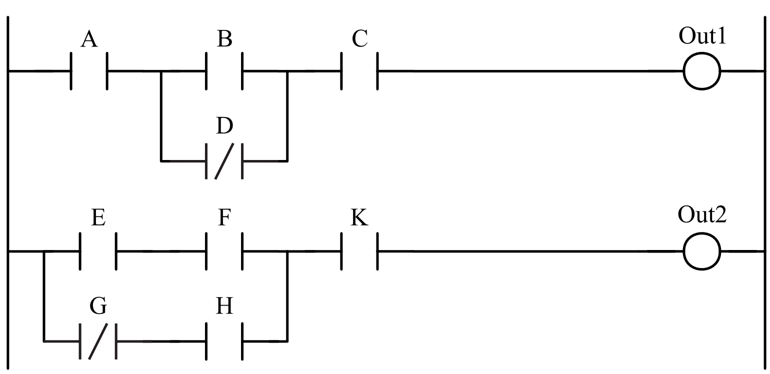


Figure 2.7. Ladder logic diagram with basic instructions.

For the remainder of the book, the languages will be presented in the following order:

IEC 61131-3 standard
Modicon Concept (IEC compliant)
Allen-Bradley ControlLogix (IEC compliant)
Allen-Bradley PLC-5/SLC-500 (not IEC compliant)
Siemens S7 (IEC compliant)
GE Fanuc (not IEC compliant)

The Modicon Concept ladder logic is presented first because it is closest to the IEC 61131-3 standard. The Allen-Bradley processors are presented next because of their widespread use in North America.

2.3.1 IEC 61131-3

The basic ladder logic contact symbols are

—| |— *Normally open (NO) contact*. Passes power (**on**) if coil driving the contact is **on** (closed).

—|/|— *Normally closed (NC) contact*. Passes power (**on**) if coil driving the contact is **off** (open).

—|P|— *Positive transition sensing contact*. If conditions before this instruction change from **off** to **on**, this instruction passes power for only one scan (until rung is scanned again).

—|N|— *Negative transition sensing contact*. If conditions before this instruction change from **on** to **off**, this instruction passes power for only one scan (until rung is scanned again).

The basic ladder logic coil (output) symbols are

—( )— Output or *coil*. If any left-to-right path of instructions passes power, the output is energized. If there is no continuous left-to-right path of instructions passing power, the output is de-energized.

—(/)— *Negated coil*. If any left-to-right path of inputs passes power, the output is de-energized. If there is no continuous left-to-right path of instructions passing power, the output is energized.

—(S)— *Set coil*. If any rung path passes power, output is energized and remains energized, even when no rung path passes power.

—(R)— *Reset coil*. If any rung path passes power, output is de-energized and remains de-energized, even when no rung path passes power.

—(P)— *Positive transition sensing coil*. If conditions before this instruction change from **off** to **on**, coil is turned **on** for one scan.

—(N)— *Negative transition sensing coil*. If conditions before this instruction change from **on** to **off**, coil is turned **on** for one scan.

—(M)— *Retentive memory coil*. Like the ordinary coil, except the value of the output is retained even when the PLC is stopped or power fails.

—(SM)— *Set retentive memory coil*. Like the set coil, except the value of the output is retained even when the PLC is stopped or power fails.

—(RM)— *Reset retentive memory coil*. Like the reset coil, except the value of the output is retained even when the PLC is stopped or power fails.

Comments about the basic instructions

1. The transition sensing contacts and coils are useful for initialization and detecting input transitions, for example, a push button press.
2. The set and reset coils are used in conjunction with each other. Figure 2.8 is a short example using these two coils in conjunction to control a lamp.
3. The retentive memory coil instructions are used in a situation where the state of the output must be retained when the PLC is stopped or power fails. Normally, PLC outputs are turned **off** when the PLC is stopped or power fails. Depending on the system, it may be important that the state of an output be retained in order for the system to operate safely through a power failure of the PLC processor or when the PLC is stopped. For certain PLC manufacturers, this function is provided as part of the discrete output module.
4. The author discourages use of the negated coil for the following reason. In most systems the safe position is one in which the output from the PLC is **off**. Generally, contacts (often called permissives) are placed in series with the coil, indicating multiple conditions must be satisfied before the output is allowed to be energized. With the negated coil the rung conditions must be satisfied to turn **off** the output which is opposite to most safety concepts.

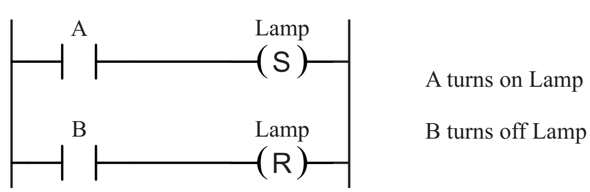


Figure 2.8. Set and reset coil example.

2.3.2 Modicon Quantum/Momentum

Using the Concept programming software, the Modicon Quantum and Momentum PLC processors may be programmed in ladder logic compatible with the older 984-series processors, or with IEC 61131-3 compliant ladder logic. The IEC 61131-3 compliant ladder logic instructions are described here. The Modicon IEC basic ladder logic contact symbols are the same as described in section 2.3.1.

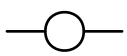
The Modicon IEC basic ladder logic coil symbols are similar to those described in section 2.3.1, except that Modicon does not support the following:

Retentive memory coil

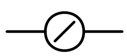
Set retentive memory coil

Reset retentive memory coil

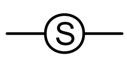
The instructions are:



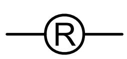
Output or *coil*. If any left-to-right path of instructions passes power, the output is energized. If there is no continuous left-to-right path of instructions passing power, the output is de-energized.



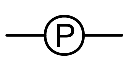
Negated coil. If any left-to-right path of inputs passes power, the output is de-energized. If there is no continuous left-to-right path of instructions passing power, the output is energized.



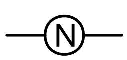
Set coil. If any rung path passes power, output is energized and remains energized, even when no rung path passes power.



Reset coil. If any rung path passes power, output is de-energized and remains de-energized, even when no rung path passes power.



Positive transition sensing coil. If conditions before this instruction change from **off** to **on**, coil is turned **on** for one scan.



Negative transition sensing coil. If conditions before this instruction change from **on** to **off**, coil is turned **on** for one scan.

2.3.3 Allen-Bradley ControlLogix and PLC-5/SLC-500

The Allen-Bradley PLC basic instructions are not as numerous as for the IEC 61131-3 basic instructions. In addition, for many of the instructions, a different symbol is used, though the function is the same as an IEC 61131-3 instruction. The Allen-Bradley basic ladder logic contact symbols are

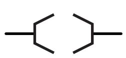
Normally open (NO) contact. Passes power (**on**) if coil driving the contact is **on** (closed). Allen-Bradley calls it XIC (eXamine If Closed).

Normally closed (NC) contact. Passes power (**on**) if coil driving the contact is **off** (open). Allen-Bradley calls it XIO (eXamine If Open).

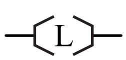
One-shot contact. If conditions before this instruction change from **off** to **on**, this instruction passes power for only one scan (ControlLogix and PLC-5 only). It is analogous to the IEC positive transition sensing contact except that this instruction <u>follows</u> the contact(s) whose transition is being sensed.

One-shot rising contact. If conditions before this instruction change from **off** to **on**, this instruction passes power for only one scan (SLC-500 only). Must immediately precede an output coil. It is analogous to the IEC positive transition sensing contact except that this instruction <u>follows</u> the contact(s) whose transition is being sensed.

For the Allen-Bradley PLCs, the basic ladder logic coil (output) symbols are

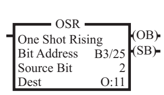


Output or *coil*. If any left-to-right path of instructions passes power, the output is energized. If there is no continuous left-to-right path of instructions passing power, the output is de-energized. Allen-Bradley calls it OTE (OuTput Energize).

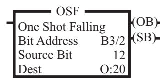


Latch coil. If any rung path passes power, output is energized and remains energized, even when no rung path passes power. It is analogous to the IEC set coil instruction. Allen-Bradley calls it OTL (OuTput Latch).

Unlatch coil. If any rung path passes power, output is de-energized and remains de-energized, even when no rung path passes power. It is analogous to the IEC reset coil instruction. Allen-Bradley calls it OTU (OuTput Unlatch).



One shot rising output. If conditions before this instruction change from **off** to **on**, the specified bit is turned **on** for one scan (ControlLogix and enhanced PLC-5 only). This is more appropriately a function block instruction because of its appearance. It is analogous to the IEC positive transition sensing coil instruction.

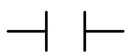


One shot falling output. If conditions before this instruction change from **on** to **off**, the specified bit is turned **on** for one scan (ControlLogix and enhanced PLC-5 only). This is more appropriately a function block instruction because of its appearance. It is analogous to the IEC negative transition sensing coil instruction.

There are no retentive memory coil instructions. The retentive function is handled in the discrete output modules.

2.3.4 Siemens S7

The three types of S7 processors (S7-200, S7-300, and S7-400) have the same basic instructions. The only exception is the midline output coil that is not valid for the S7-200 processors. The basic ladder logic contact symbols are



Normally open (NO) contact. Passes power (**on**) if coil driving the contact is **on** (closed).

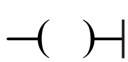
Normally closed (NC) contact. Passes power (**on**) if coil driving the contact is **off** (open).

Positive transition sensing contact. If conditions before this instruction change from **off** to **on**, this instruction passes power for only one scan (until rung is scanned again). For the S7-200 processors, this contact uses vertical bars, rather than parentheses.

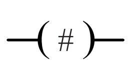
Negative transition sensing contact. If conditions before this instruction change from **on** to **off**, this instruction passes power for only one scan (until rung is scanned again). For the S7-200 processors, this contact uses vertical bars, rather than parentheses.

Invert power flow. If any left-to-right path of inputs before this contact passes power, the power flow to succeeding elements is interrupted (turned **off**). If no left-to-right path of inputs before this contact passes power, the power flow to succeeding elements is turned **on**.

The basic ladder logic coil (output) symbols are



Output or *coil*. If any left-to-right path of instructions passes power, the output is energized. If there is no continuous left-to-right path of instructions passing power, the output is de-energized.



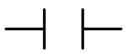
Midline output coil. Output coil in middle of rung. Other logic can occur to the right of this coil. Not valid for the S7-200 processors.

Set coil. If any rung path passes power, output is energized and remains energized, even when no rung path passes power.

Reset coil. If any rung path passes power, output is de-energized and remains de-energized, even when no rung path passes power.

2.3.5 GE Fanuc

For the GE Fanuc PLCs, the basic ladder logic contact symbols are



Normally open (NO) contact. Passes power (**on**) if coil driving the contact is **on** (closed).

Normally closed (NC) contact. Passes power (**on**) if coil driving the contact is **off** (open).

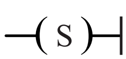
Positive transition sensing contact. If conditions before this contact change from **off** to **on**, power is passed for one scan (until rung is scanned again). Valid for 90-70 processors only.

Negative transition sensing contact. If conditions before this contact change from **on** to **off**, power is passed for one scan (until rung is scanned again). Valid for 90-70 processors only.

The basic ladder logic coil (output) symbols are

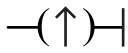
Output or *coil*. If any left-to-right path of instructions passes power, the output is energized. If there is no continuous left-to-right path of instructions passing power, the output is de-energized.

Negated coil. If any left-to-right path of inputs passes power, the output is de-energized. If there is no continuous left-to-right path of instructions passing power, the output is energized.

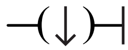


Set coil. If any rung path passes power, output is energized and remains energized, even when no rung path passes power.

Reset coil. If any rung path passes power, output is de-energized and remains de-energized, even when no rung path passes power.



Positive transition sensing coil. If conditions before this instruction change from **off** to **on**, coil is turned **on** for one scan.



Negative transition sensing coil. If conditions before this instruction change from **on** to **off**, coil is turned **on** for one scan.

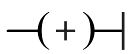
Retentive memory coil. Like the ordinary coil, except the value of the output is retained even when the PLC is stopped or power fails.

Negated retentive memory coil. Like the negated coil, except the value of the output is retained even when the PLC is stopped or power fails.

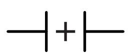
Set retentive memory coil. Like the set coil, except the value of the output is retained even when the PLC is stopped or power fails.

Reset retentive memory coil. Like the reset coil, except the value of the output is retained even when the PLC is stopped or power fails.

A continuation coil and contact are used to handle ladder rungs with more than 10 columns:



Continuation coil. If any left-to-right path of instructions passes power, the next continuation contact is turned **on**. If there is no continuous left-to-right path of instructions passing power, the next continuation contact is turned **off**.



Continuation contact. Passes power (**on**) if preceding continuation coil is **on**.

2.4 LADDER LOGIC DIAGRAM

An example PLC ladder logic diagram appears in Figure 2.9. The vertical lines on the left and right are called the power rails. The contacts are arranged horizontally between the power rails, hence the term *rung*. The ladder diagram in Figure 2.9 has three rungs. The arrangement is similar to a ladder one uses to climb onto a roof. In addition, Figure 2.9 shows an example diagram like one would see if monitoring the running program in the PLC. The thick lines indicate continuity and the state (**on**/**off**) of the inputs and outputs is shown next to the tag. Regardless of the contact symbol, if the contact is closed (continuity through it), it is shown as thick lines. If the contact is open, it is shown as thin lines. In a relay ladder diagram, power flows from left to right. In PLC ladder logic, there is no real power flow, but there still must be a continuous path through closed contacts in order to energize an output. In Figure 2.9 the output on the first rung is **off** because the contact for C is open, blocking continuity through the D and E contacts. Also notice that the E input is **off**,

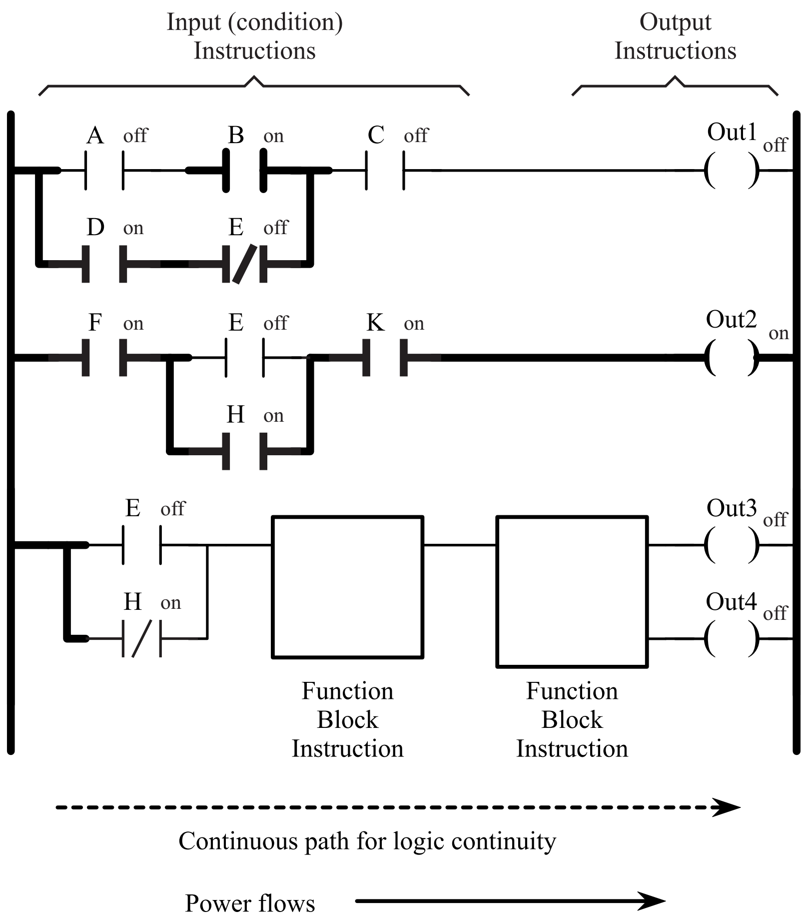


Figure 2.9. Sample ladder logic diagram.

which means the NC contact in the first rung is closed and the NO contact in the second rung is open.

Figure 2.9 also introduces the concept of *function block instructions*. Any instruction that is not a contact or a coil is called a function block instruction because of its appearance in the ladder diagram. The most common function block instructions are timer, counter, comparison, and computation operations. More advanced function block instructions include sequencer, shift register, and first-in first-out operations.

Some manufacturers group the instructions into two classes: input instructions and output instructions. This distinction was made because in relay ladder logic, outputs were never connected in series and always occurred on the extreme right hand side of the rung. Contacts always appeared on the left side of coils and never on the right side. To turn on multiple outputs simultaneously, coils are connected in parallel. This restriction was relaxed in IEC 61131-3 and outputs may be connected in series. Also, contacts can occur on the right side of a coil as long as a coil is the last element in the rung. Of the ladder logic languages covered by this text, only the IEC 61131-3, Modicon Concept, and Allen-Bradley ControlLogix allow coil instructions to be connected in series.

This text avoids using a series connection of coils for two reasons:

1. most PLCs do not allow it, and
2. it is counterintuitive to maintenance personnel who often interpret ladder logic in the context of an electrical diagram.

Also, in IEC 61131-3, all function block instructions are input instructions because the only output instructions are the coils. The Allen-Bradley PLC-5 and SLC-500 have function block output instructions (e.g., timer, counter, and computation) which must be remembered when constructing ladder logic programs for these PLCs.

Example 2.4. Draw a ladder diagram that will cause the output, pilot light PL2, to be **on** when selector switch SS2 is **closed**, push-button PB4 is **closed** and limit switch LS3 is **open**. (Note: no I/O addresses yet.)

Solution. The first question to answer is "What is the output?" The output is PL2, so the coil labeled as PL2 is put on the right side of the rung. Secondly, consider the type of connection of contacts to use. Since **all** three switches must be in a certain position to turn on the pilot light, a <u>series</u> connection is needed. Thirdly, the type of contact is determined by the switch position to turn on the pilot light:

SS2 closed	→	─┤ ├─
PB4 closed	→	─┤ ├─
LS3 open	→	─┤/├─

Putting all the pieces together, only one rung of ladder logic is needed, as shown in Figure 2.10.

Design Tip

The concept of placing the output on the rung first and then "looking back" to determine the input conditions is very important. Because of the way the diagram is configured, one has a tendency to consider the input conditions first and then position the output coil as the last step. As will be shown later, the coil or negated coil instruction referring to a particular output must only occur <u>**once**</u> in a ladder program. Considering the output coil first and the conditions for which it is active (on) will avoid repeating coils.

Example 2.5. Draw a ladder diagram that is equivalent to the digital logic diagram in Figure 2.11, which is the same as the following descriptions.

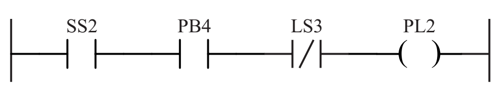


Figure 2.10. Solution to Example 2.4.

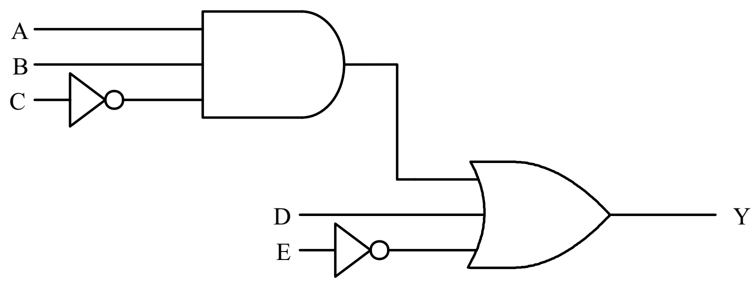


Figure 2.11. Digital logic for Example 2.5.

In words:

Y is **on** when (A is **on** and B is **on** and C is **off**) or D is **on** or E is **off**.

Boolean logic equation:

$Y = AB\overline{C} + D + \overline{E}$

Solution. First, answer, "What is the output?" The output is Y, so the coil labeled as Y is put on the right side of the rung. Secondly, consider the type of connection of contacts to use. For this problem, there is more than one type of connection. The three inputs within the parentheses (the AND gate in Figure 2.11) are connected with "and," so a <u>series</u> connection is required for these three contacts. The other two inputs (D and E) are connected with the three series contacts by "or" (the OR gate inputs), so a <u>parallel</u> connection is required. Thirdly, the type of contact is determined by the input state that turns **on** the output, Y:

A **on**	→	┤ ├	D **on**	→	┤ ├
B **on**	→	┤ ├	E **off**	→	┤/├
C **off**	→	┤/├			

Putting all the parts together, only one rung of ladder logic is needed, as shown in Figure 2.12.

Suppose one changes the D contact in Figure 2.12 to refer to Y, the output (shown as Figure 2.13). Is this legitimate? Yes, it is legitimate, though probably not something one

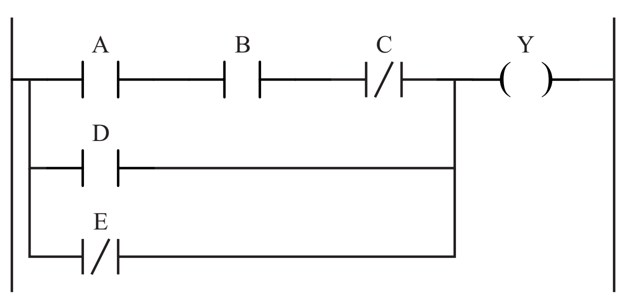


Figure 2.12. Solution to Example 2.5.

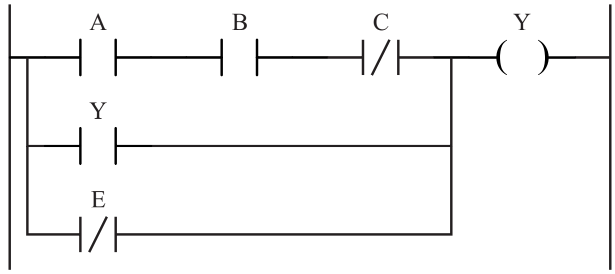


Figure 2.13. Output that appears as an input.

would want to do for this example. Even in relay ladder logic, it is legal and there is no wiring short because the coil for relay Y and its NO contact are not connected. This concept is called *sealing* or latching an output without using the set (or latch) coil instruction. In this example, it is not a good idea because once Y is sealed **on**, there is no provision to turn it off. Why?

There are some precautions to observe when programming in ladder logic:

1. **DO NOT** repeat normal output coils or negated coils that refer to the same tag. To illustrate what happens when this is done, consider the ladder logic diagram in Figure 2.14. This is the ladder of Figure 2.9, modified for this illustration. Note that the coils for both the first and second rung refer to Out1. When the first rung of the ladder is scanned, Out1 is turned **on**. However, when the second rung is scanned, Out1 is turned **off**, overriding the logic in the first rung. If all of these conditions are needed to turn on Out1, then they all should be placed in parallel, as in Figure 2.15. In this illustration, it was obvious there is a problem. Normally,

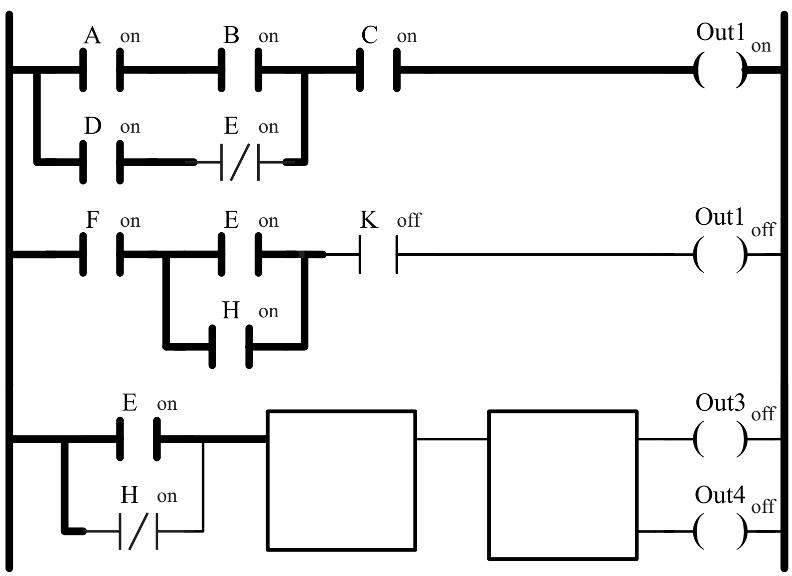


Figure 2.14. Ladder with repeated output.

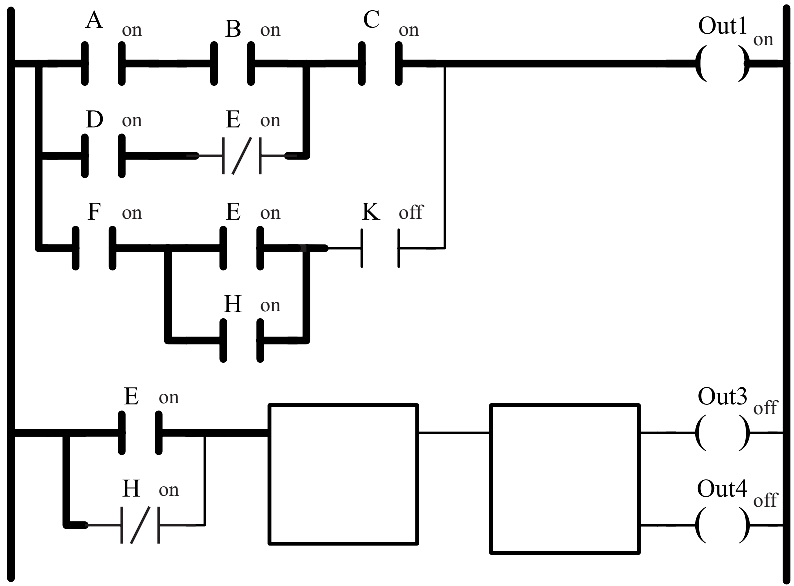


Figure 2.15. Repeated output corrected.

when this problem occurs, the rungs are not adjacent, and it is not so obvious. Compounding the problem, not all PLC programming software checks for this situation. Therefore, the best way to prevent this problem is to consider the output coil **first** and then consider all of the conditions that drive that output.

2. Use the set (latch) coil and reset (unlatch) coil instructions together. If a set coil refers to an output, there should also be a reset coil for that output. Also, for the same reason that output coil and negated coil instructions should not be repeated, do not mix the set/reset coils with the output coil and negated coil instructions that refer to the same output.
3. Be careful when using the set/reset coil instructions to reference PLC physical outputs. If the system involves safety and a set coil is used for a PLC physical output, simply interrupting the condition on the set coil rung **will not** turn off the physical output. All of the conditions that prevent the device from being turned on must also appear on a rung with a reset coil output. For this reason, some companies forbid the use of the set/reset coil instructions.
4. Reverse power flow in the contact matrix is **not** allowed. When electromechanical relays are used to implement ladder logic, power can flow either way through the contacts. For example, consider the ladder logic in Figure 2.16. If implemented with electromechanical relays, power is allowed to flow right-to-left through the contact for SS2. When solid state relays replaced electromechanical relays for ladder logic, power can flow only one way (left-to-right) through the contacts. This restriction was carried to PLC ladder logic. If the reverse power flow path is truly needed, then insert it as a separate path, where the power flows from left to right. The reverse power flow path in Figure 2.16 is added as a separate path in Figure 2.17.

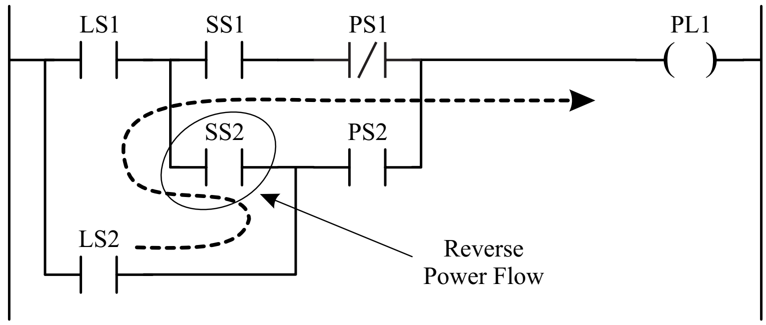


Figure 2.16. Reverse power flow in ladder logic.

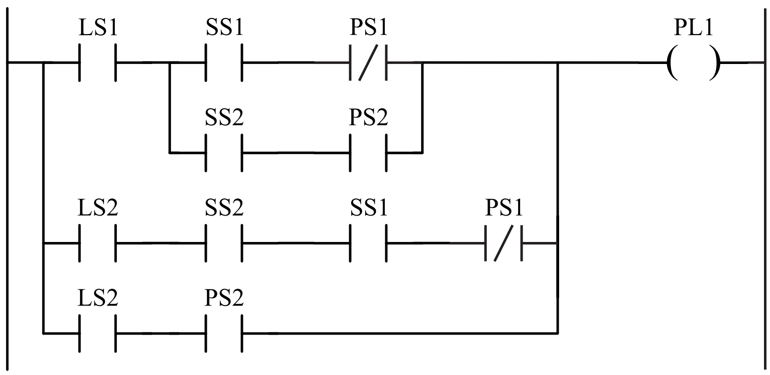


Figure 2.17. Reverse power flow in ladder logic corrected.

2.5 PLC PROCCESSOR SCAN

Previously, the process that the PLC uses to scan the ladder logic has only been implied. Now it will be discussed in detail. In addition to scanning the ladder logic, the PLC processor must also read the state of its physical inputs and set the state of the physical outputs. These three major tasks in a PLC processor scan are executed in the following order:

Read the physical inputs

Scan the ladder logic program

Write the physical outputs

The processor repeats these tasks as long as it is running, as shown pictorially in Figure 2.18. The time required to complete these three tasks is defined as the *scan time* and is typically 1 - 200 milliseconds, depending on the length of the ladder logic program. For very large ladder logic programs, the scan time can be more than one second. When this happens, the PLC program may miss transient events, especially if they are shorter than one second. In this situation, the possible solutions are: