Table of Contents

Introduction

Welcome to another course in the STEP 2000 series, **S**iemens **T**echnical **E**ducation **P**rogram, designed to prepare our distributors to sell Siemens Energy & Automation products more effectively. This course covers **Basics of PLCs** and related products.

Upon completion of **Basics of PLCs** you should be able to:

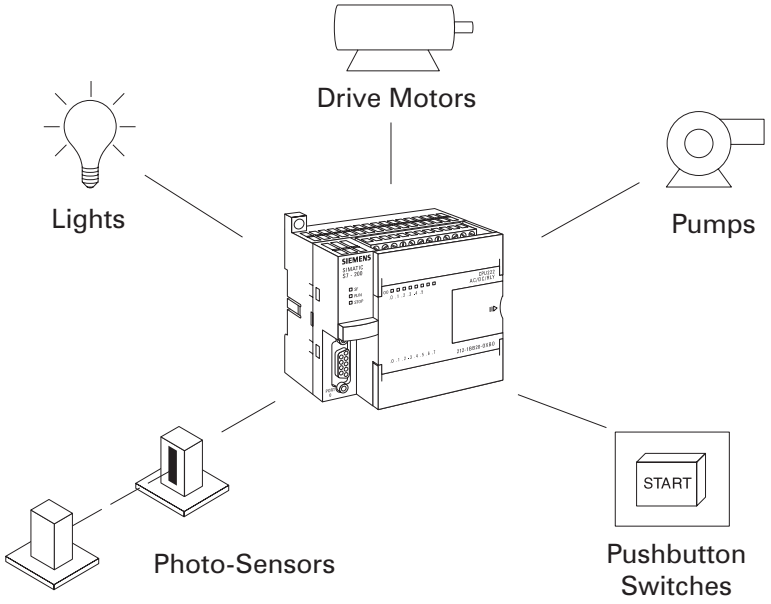
- Identify the major components of a PLC and describe their functions
- Convert numbers from decimal to binary, BCD, and hexadecimal
- Identify typical discrete and analog inputs and outputs
- Read a basic ladder logic diagram and statement list
- Identify operational differences between different S7-200 models
- Identify the proper manual to refer to for programming or installation of an S7-200 PLC
- Connect a simple discrete input and output to an S7-200
- Select the proper expansion module for analog inputs and outputs
- Describe the operation of timers and counters

This knowledge will help you better understand customer applications. In addition, you will be better able to describe products to customers and determine important differences between products. You should complete **Basics of Electricity** before attempting **Basics of PLCs**. An understanding of many of the concepts covered in **Basics of Electricity** is required for **Basics of PLCs**. In addition you may wish to complete **Basics of Control Components**. Devices covered in **Basics of Control Components** are used with programmable logic controllers.

If you are an employee of a Siemens Energy & Automation authorized distributor, fill out the final exam tear-out card and mail in the card. We will mail you a certificate of completion if you score a passing grade. Good luck with your efforts.

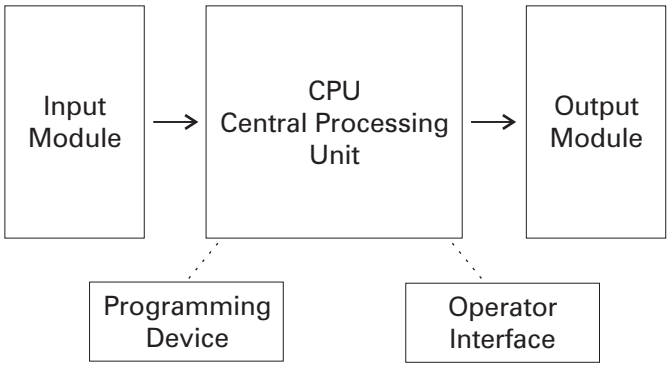
PLCs

Programmable Logic Controllers (PLCs), also referred to as programmable controllers, are in the computer family. They are used in commercial and industrial applications. A PLC monitors inputs, makes decisions based on its program, and controls outputs to automate a process or machine. This course is meant to supply you with basic information on the functions and configurations of PLCs.

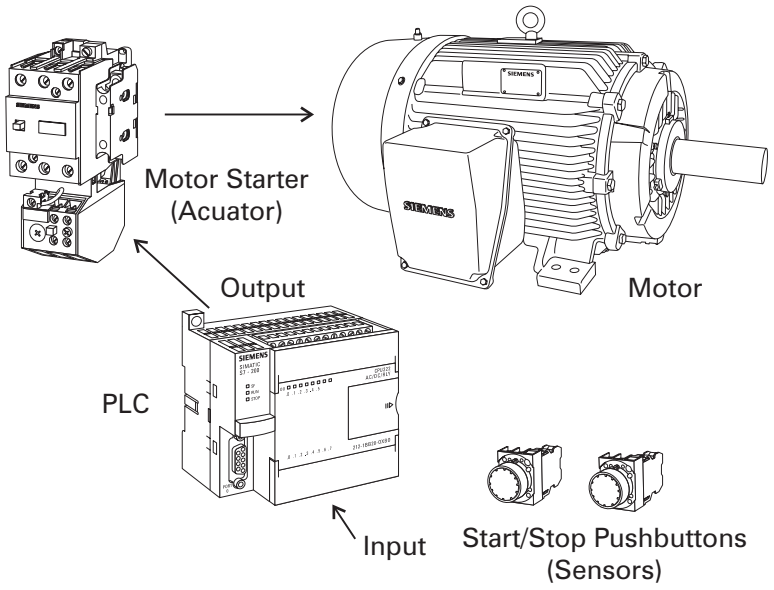


Basic PLC operation

PLCs consist of input modules or points, a Central Processing Unit (CPU), and output modules or points. An input accepts a variety of digital or analog signals from various field devices (sensors) and converts them into a logic signal that can be used by the CPU. The CPU makes decisions and executes control instructions based on program instructions in memory. Output modules convert control instructions from the CPU into a digital or analog signal that can be used to control various field devices (actuators). A programming device is used to input the desired instructions. These instructions determine what the PLC will do for a specific input. An operator interface device allows process information to be displayed and new control parameters to be entered.

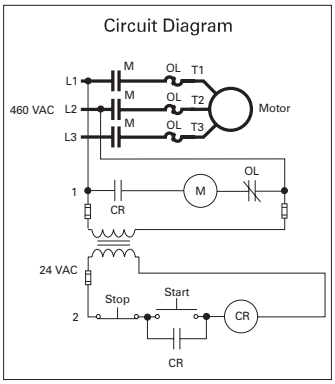


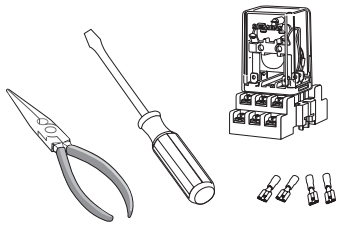
Pushbuttons (sensors), in this simple example, connected to PLC inputs, can be used to start and stop a motor connected to a PLC through a motor starter (actuator).



Hard-wired control

Prior to PLCs, many of these control tasks were solved with contactor or relay controls. This is often referred to as hard-wired control. Circuit diagrams had to be designed, electrical components specified and installed, and wiring lists created. Electricians would then wire the components necessary to perform a specific task. If an error was made the wires had to be reconnected correctly. A change in function or system expansion required extensive component changes and rewiring.





PLCs

The same, as well as more complex tasks, can be done with a PLC. Wiring between devices and relay contacts is done in the PLC program. Hard-wiring, though still required to connect field devices, is less intensive. Modifying the application and correcting errors are easier to handle. It is easier to create and change a program in a PLC than it is to wire and rewire a circuit.

Advantages

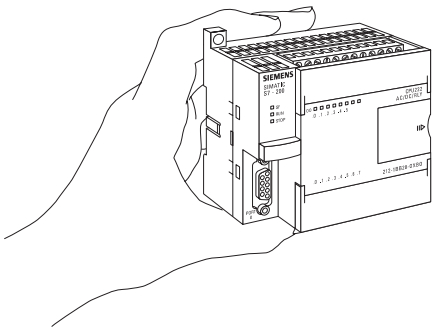
- Smaller physical size than hard-wire solutions
- Easier and faster to make changes
- PLCs have integrated diagnostics and override functions
- Diagnostics are centrally available
- Applications can be immediately documented
- Applications can be duplicated faster and less expensively

Siemens PLCs

Siemens makes several PLC product lines in the SIMATIC® S7 family. They are: S7-200, S7-300, and S7-400.

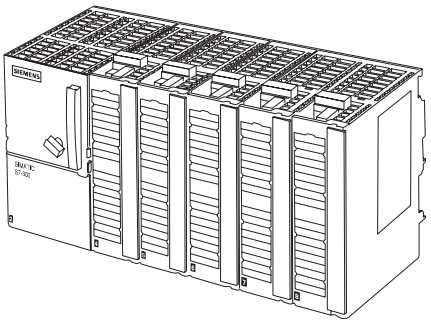
S7-200

The S7-200 is referred to as a micro PLC because of its small size. The S7-200 has a brick design which means that the power supply and I/O are on-board. The S7-200 can be used on smaller, stand-alone applications such as elevators, car washes, or mixing machines. It can also be used on more complex industrial applications such as bottling and packaging machines.

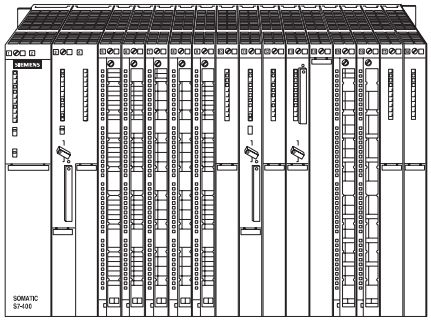


S7-300 and S7-400

The S7-300 and S7-400 PLCs are used in more complex applications that support a greater number of I/O points. Both PLCs are modular and expandable. The power supply and I/O consist of seperate modules connected to the CPU. Choosing either the S7-300 or S7-400 depends on the complexity of the task and possible future expansion. Your Siemens sales representative can provide you with additional information on any of the Siemens PLCs.



S7-300



S7-400

Number Systems

Since a PLC is a computer, it stores information in the form of On or Off conditions (1 or 0), referred to as binary digits (bits). Sometimes binary digits are used individually and sometimes they are used to represent numerical values.

Decimal system

Various number systems are used by PLCs. All number systems have the same three characteristics: digits, base, weight. The decimal system, which is commonly used in everyday life, has the following characteristics:

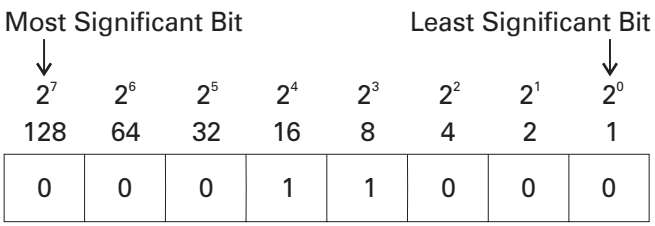
Ten digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Base	10
Weights	1, 10, 100, 1000, ...

Binary system

The binary system is used by programmable controllers. The binary system has the following characteristics:

Two digits	0, 1
Base	2
Weights	Powers of base 2 (1, 2, 4, 8, 16, ...)

In the binary system 1s and 0s are arranged into columns. Each column is weighted. The first column has a binary weight of 2^0. This is equivalent to a decimal 1. This is referred to as the least significant bit. The binary weight is doubled with each succeeding column. The next column, for example, has a weight of 2^1, which is equivalent to a decimal 2. The decimal value is doubled in each successive column. The number in the far left hand column is referred to as the most significant bit. In this example, the most significant bit has a binary weight of 2^7. This is equivalent to a decimal 128.

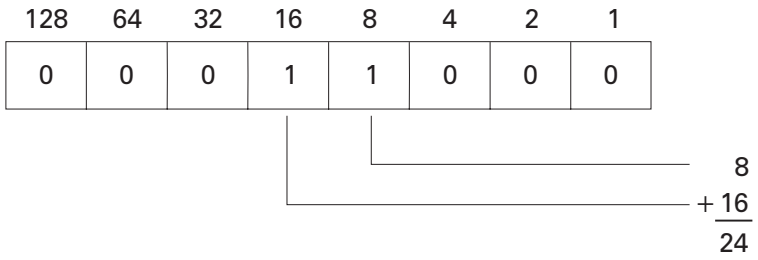


Converting binary to decimal

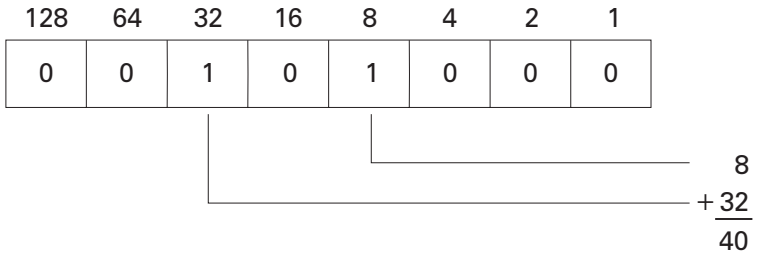
The following steps can be used to interpret a decimal number from a binary value.

1) Search from left to right (least significant to most significant bit) for 1s.
2) Write down the decimal representation of each column containing a 1.
3) Add the column values.

In the following example, the fourth and fifth columns from the right contain a 1. The decimal value of the fourth column from the right is 8, and the decimal value of the fifth column from the right is 16. The decimal equivalent of this binary number is 24. The sum of all the weighted columns that contain a 1 is the decimal number that the PLC has stored.

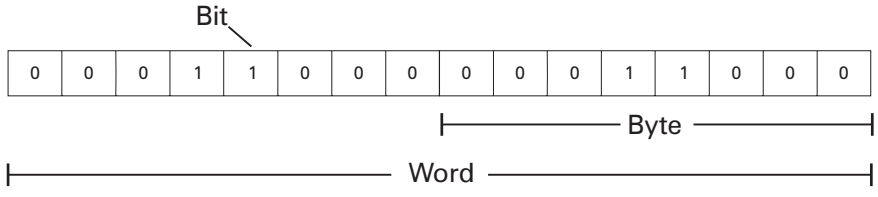


In the following example the fourth and sixth columns from the right contain a 1. The decimal value of the fourth column from the right is 8, and the decimal value of the sixth column from the right is 32. The decimal equivalent of this binary number is 40.



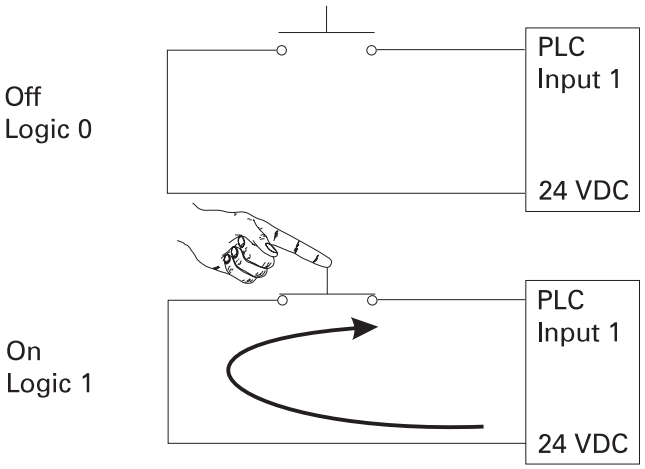
Bits, bytes, and words

Each binary piece of data is a bit. Eight bits make up one byte. Two bytes, or 16 bits, make up one word.



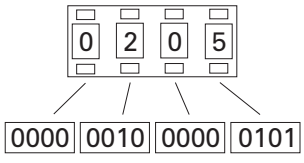
Logic 0, logic 1

Programmable controllers can only understand a signal that is On or Off (present or not present). The binary system is a system in which there are only two numbers, 1 and 0. Binary 1 indicates that a signal is present, or the switch is On. Binary 0 indicates that the signal is not present, or the switch is Off.



BCD

Binary-Coded Decimal (BCD) are decimal numbers where each digit is represented by a four-bit binary number. BCD is commonly used with input and output devices. A thumbwheel switch is one example of an input device that uses BCD. The binary numbers are broken into groups of four bits, each group representing a decimal equivalent. A four-digit thumbwheel switch, like the one shown here, would control 16 (4 x 4) PLC inputs.



Decimal Numbers	BCD Numbers
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Hexadecimal

Hexadecimal is another system used in PLCs. The hexadecimal system has the following characteristics:

16 digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
Base	16
Weights	Powers of base 16 (1, 16, 256, 4096 ...)

The ten digits of the decimal system are used for the first ten digits of the hexadecimal system. The first six letters of the alphabet are used for the remaining six digits.

A = 10	D = 13
B = 11	E = 14
C = 12	F = 15

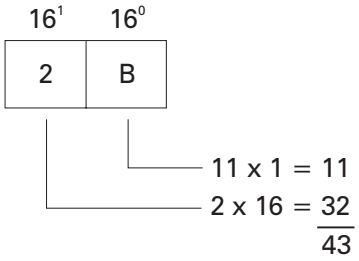
The hexadecimal system is used in PLCs because it allows the status of a large number of binary bits to be represented in a small space such as on a computer screen or programming device display. Each hexadecimal digit represents the exact status of four binary bits. To convert a decimal number to a hexadecimal number the decimal number is divided by the base of 16. To convert decimal 28, for example, to hexadecimal:

$$16\overline{)\,28}\quad 1 \text{ r } 12$$

Decimal 28 divided by 16 is 1 with a remainder of 12. Twelve is equivalent to C in hexadecimal. The hexadecimal equivalent of decimal 28 is 1C.

The decimal value of a hexadecimal number is obtained by multiplying the individual hexadecimal digits by the base 16 weight and then adding the results. In the following example the hexadecimal number 2B is converted to its decimal equivalent of 43.

$16^0 = 1$
$16^1 = 16$
B = 11



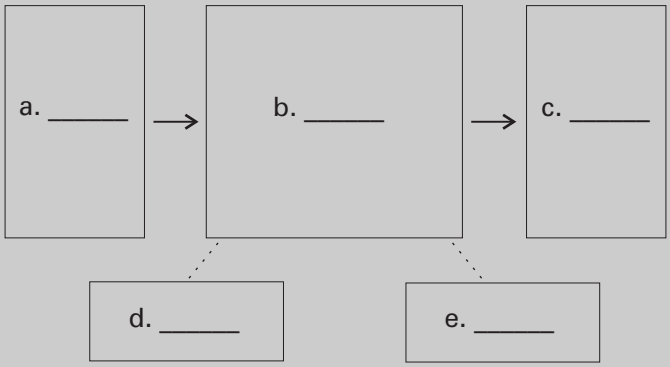
Conversion of numbers

The following chart shows a few numeric values in decimal, binary, BCD, and hexadecimal representation.

Decimal	Binary	BCD	Hexadecimal
0	0	0000	0
1	1	0001	1
2	1	0010	2
3	11	0011	3
4	100	0100	4
5	101	0101	5
6	110	0110	6
7	111	0111	7
8	1000	1000	8
9	1001	1001	9
10	1010	0001 0000	A
11	1011	0001 0001	B
12	1100	0001 0010	C
13	1101	0001 0011	D
14	1110	0001 0100	E
15	1111	0001 0101	F
16	1 0000	0001 0110	10
17	1 0001	0001 0111	11
18	1 0010	0001 1000	12
19	1 0011	0001 1001	13
20	1 0100	0010 0000	14
.	.	.	.
.	.	.	.
126	111 1110	0001 0010 0110	7E
127	111 1111	0001 0010 0111	7F
128	1000 1000	0001 0010 1000	80
.	.	.	.
.	.	.	.
510	1 1111 1110	0101 0001 000	1FE
511	1 1111 1111	0101 0001 0001	2FF
512	10 0000 0000	0101 0001 0010	200

Review 1

1. Identify the following:



2. The binary number system has a base _____ .

3. The hexadecimal number system has a base _____ .

4. Convert a decimal 10 to the following:

   Binary _____

   BCD _____

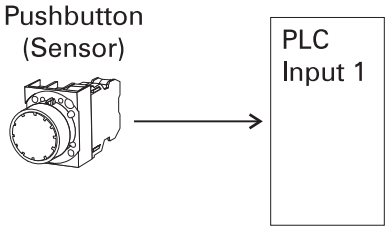
   Hexadecimal _____

Terminology

The language of PLCs consists of a commonly used set of terms; many of which are unique to PLCs. In order to understand the ideas and concepts of PLCs, an understanding of these terms is necessary.

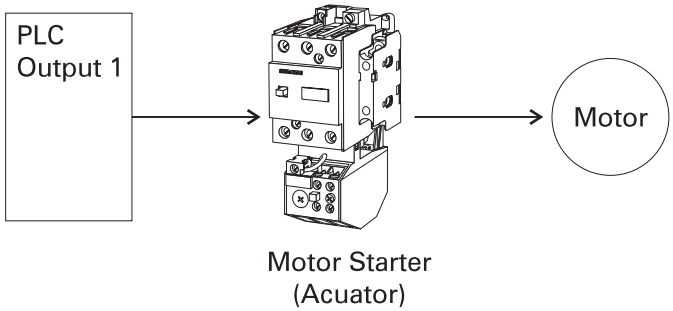
Sensor

A sensor is a device that converts a physical condition into an electrical signal for use by the PLC. Sensors are connected to the input of a PLC. A pushbutton is one example of a sensor that is connected to the PLC input. An electrical signal is sent from the pushbutton to the PLC indicating the conditon (open/ closed) of the pushbutton contacts.



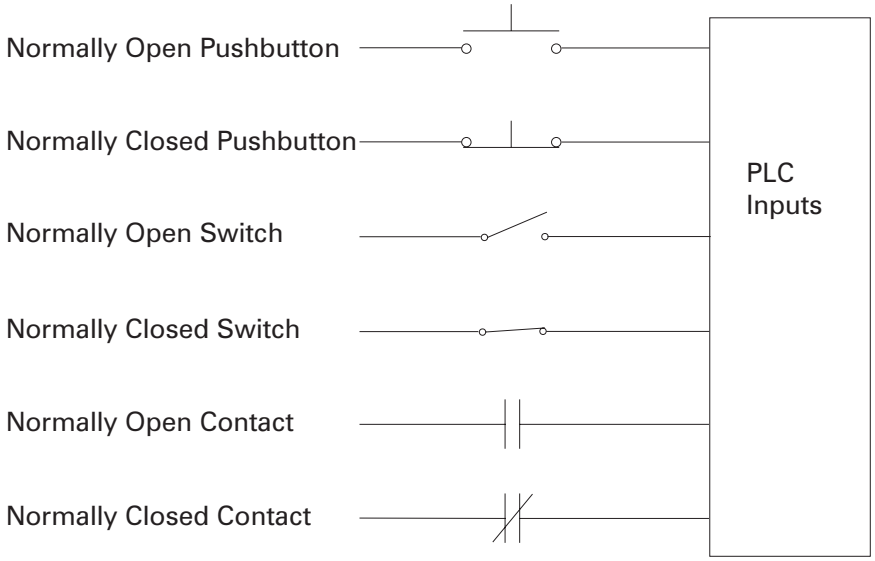
Actuators

Actuators convert an electrical signal from the PLC into a physical condition. Actuators are connected to the PLC output. A motor starter is one example of an actuator that is connected to the PLC output. Depending on the output PLC signal the motor starter will either start or stop the motor.

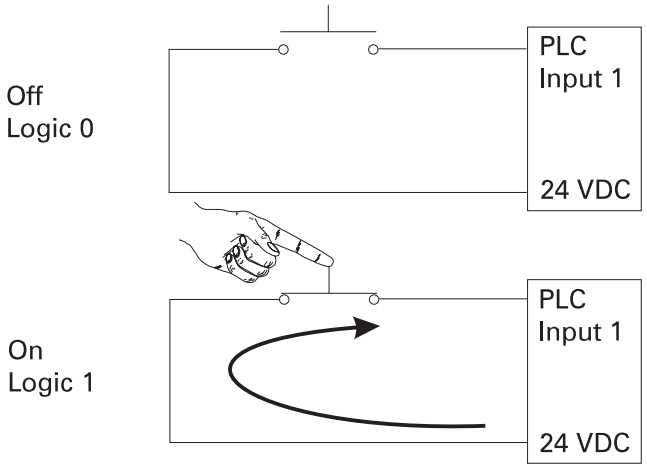


Discrete input

A discrete input, also referred to as a digital input, is an input that is either in an ON or OFF condition. Pushbuttons, toggle switches, limit switches, proximity switches, and contact closures are examples of discrete sensors which are connected to the PLCs discrete or digital inputs. In the ON condition a discrete input may be referred to as a logic 1 or a logic high. In the OFF condition a discrete input may be referred to as a logic 0 or a logic low.

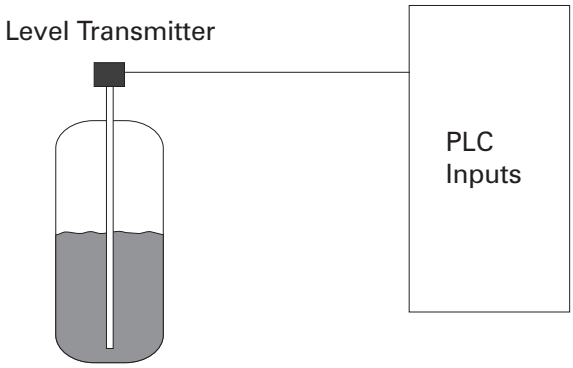


A Normally Open (NO) pushbutton is used in the following example. One side of the pushbutton is connected to the first PLC input. The other side of the pushbutton is connected to an internal 24 VDC power supply. Many PLCs require a separate power supply to power the inputs. In the open state, no voltage is present at the PLC input. This is the OFF condition. When the pushbutton is depressed, 24 VDC is applied to the PLC input. This is the ON condition.



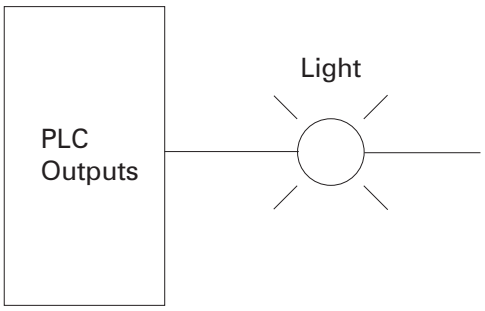
Analog inputs

An analog input is an input signal that has a continuous signal. Typical analog inputs may vary from 0 to 20 milliamps, 4 to 20 milliamps, or 0 to 10 volts. In the following example, a level transmitter monitors the level of liquid in a tank. Depending on the level transmitter, the signal to the PLC can either increase or decrease as the level increases or decreases.



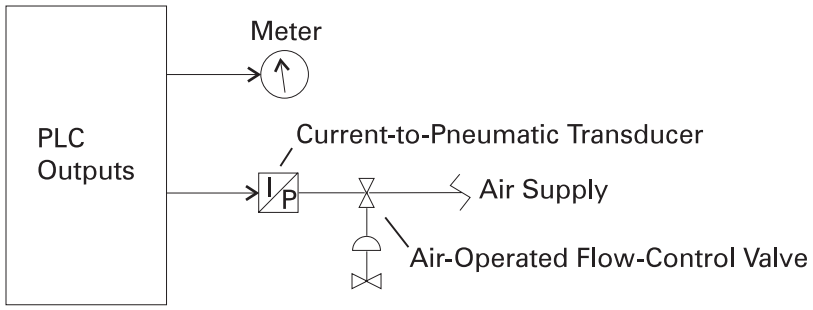
Discrete outputs

A discrete output is an output that is either in an ON or OFF condition. Solenoids, contactor coils, and lamps are examples of actuator devices connected to discrete outputs. Discrete outputs may also be referred to as digital outputs. In the following example, a lamp can be turned on or off by the PLC output it is connected to.



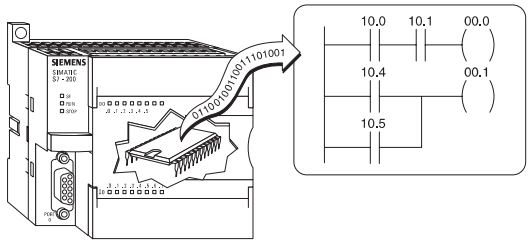
Analog outputs

An analog output is an output signal that has a continuous signal. The output may be as simple as a 0-10 VDC level that drives an analog meter. Examples of analog meter outputs are speed, weight, and temperature. The output signal may also be used on more complex applications such as a current-to-pneumatic transducer that controls an air-operated flow-control valve.



CPU

The central processor unit (CPU) is a microprocessor system that contains the system memory and is the PLC decision-making unit. The CPU monitors the inputs and makes decisions based on instructions held in the program memory. The CPU performs relay, counting, timing, data comparison, and sequential operations.

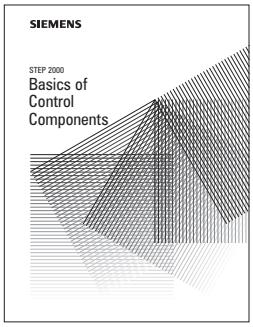


Programming

A program consists of one or more instructions that accomplish a task. Programming a PLC is simply constructing a set of instructions. There are several ways to look at a program such as ladder logic, statement lists, or function block diagrams.

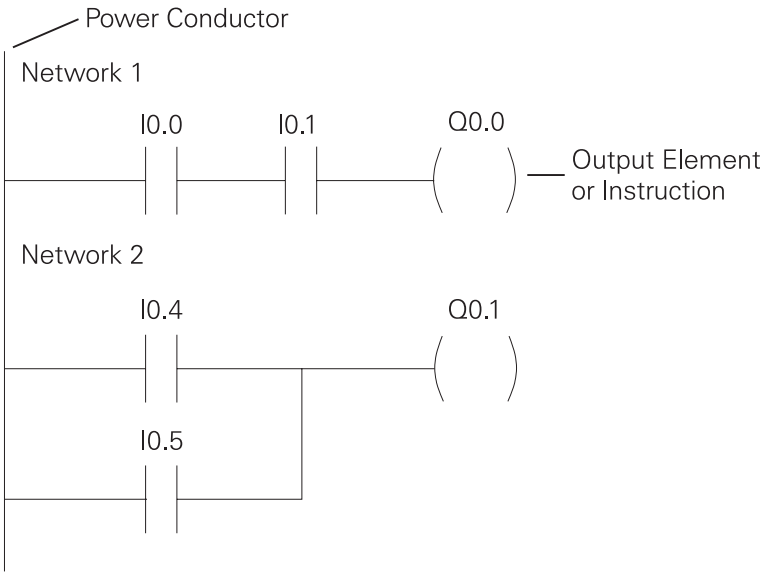
Ladder logic

Ladder logic (LAD) is one programming language used with PLCs. Ladder logic uses components that resemble elements used in a line diagram format to describe hard-wired control. Refer to the STEP 2000 course **Basics of Control Components** for more information on line diagrams.



Ladder logic diagram

The left vertical line of a ladder logic diagram represents the power or energized conductor. The output element or instruction represents the neutral or return path of the circuit. The right vertical line, which represents the return path on a hard-wired control line diagram, is omitted. Ladder logic diagrams are read from left-to-right, top-to-bottom. Rungs are sometimes referred to as networks. A network may have several control elements, but only one output coil.



In the example program shown example I0.0, I0.1 and Q0.0 represent the first instruction combination. If inputs I0.0 and I0.1 are energized, output relay Q0.0 energizes. The inputs could be switches, pushbuttons, or contact closures. I0.4, I0.5, and Q1.1 represent the second instruction combination. If either input I0.4 or I0.5 are energized, output relay Q0.1 energizes.

Statement list

A statement list (STL) provides another view of a set of instructions. The operation, what is to be done, is shown on the left. The operand, the item to be operated on by the operation, is shown on the right. A comparison between the statement list shown below, and the ladder logic shown on the previous page, reveals a similar structure. The set of instructions in this statement list perform the same task as the ladder diagram.

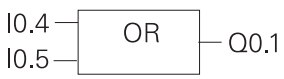
```
NETWORK 1
          LD      I0.0
          A       I0.1
          =       Q0.0
NETWORK 2
          LD      I0.4
          O       I0.5
          =       Q0.1
```

Function Block Diagrams

Function Block Diagrams (FBD) provide another view of a set of instructions. Each function has a name to designate its specific task. Functions are indicated by a rectangle. Inputs are shown on the left-hand side of the rectangle and outputs are shown on the right-hand side. The function block diagram shown below performs the same function as shown by the ladder diagram and statement list.

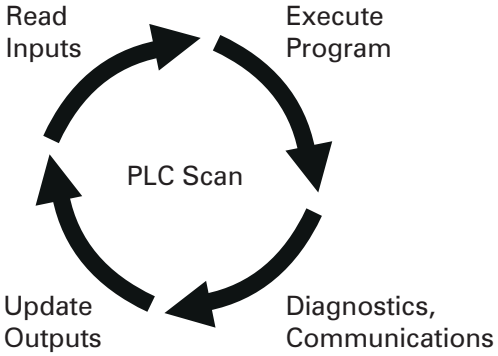
NETWORK 1

NETWORK 2



PLC scan

The PLC program is executed as part of a repetitive process referred to as a scan. A PLC scan starts with the CPU reading the status of inputs. The application program is executed using the status of the inputs. Once the program is completed, the CPU performs internal diagnostics and communication tasks. The scan cycle ends by updating the outputs, then starts over. The cycle time depends on the size of the program, the number of I/Os, and the amount of communication required.

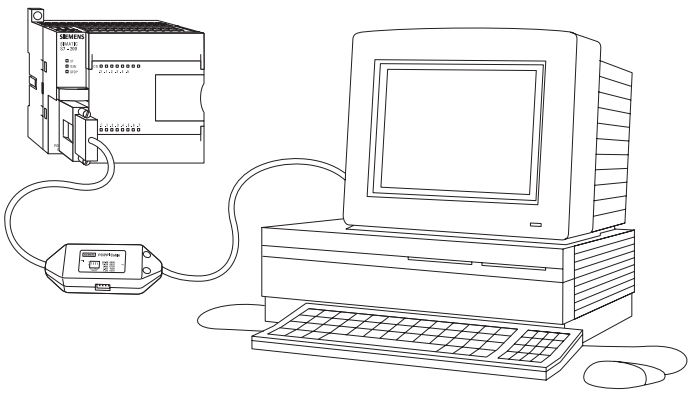


Software

Software is any information in a form that a computer or PLC can use. Software includes the instructions or programs that direct hardware.

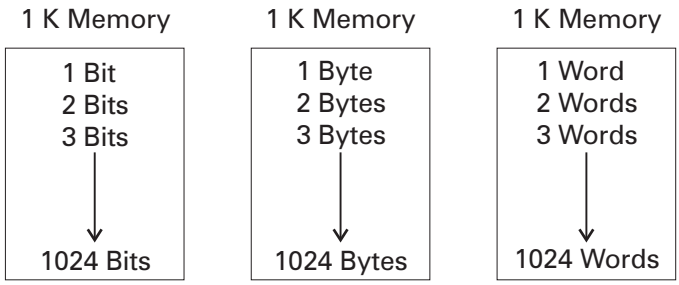
Hardware

Hardware is the actual equipment. The PLC, the programming device, and the connecting cable are examples of hardware.



Memory size

Kilo, abbreviated K, normally refers to 1000 units. When talking about computer or PLC memory, however, 1K means 1024. This is because of the binary number system (2^{10}=1024). This can be 1024 bits, 1024 bytes, or 1024 words, depending on memory type.



RAM

Random Access Memory (RAM) is memory where data can be directly accessed at any address. Data can be written to and read from RAM. RAM is used as a temporary storage area. RAM is volatile, meaning that the data stored in RAM will be lost if power is lost. A battery backup is required to avoid losing data in the event of a power loss.

ROM

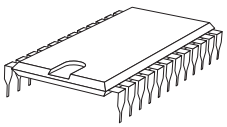
Read Only Memory (ROM) is a type of memory that data can be read from but not written to. This type of memory is used to protect data or programs from accidental erasure. ROM memory is nonvolatile. This means a user program will not lose data during a loss of electrical power. ROM is normally used to store the programs that define the capabilities of the PLC.

EPROM

Erasable Programmable Read Only Memory (EPROM) provides some level of security against unauthorized or unwanted changes in a program. EPROMs are designed so that data stored in them can be read, but not easily altered. Changing EPROM data requires a special effort. UVEPROMs (ultraviolet erasable programmable read only memory) can only be erased with an ultraviolet light. EEPROM (electronically erasable programmable read only memory), can only be erased electronically.

Firmware

Firmware is user or application specific software burned into EPROM and delivered as part of the hardware. Firmware gives the PLC its basic functionality.



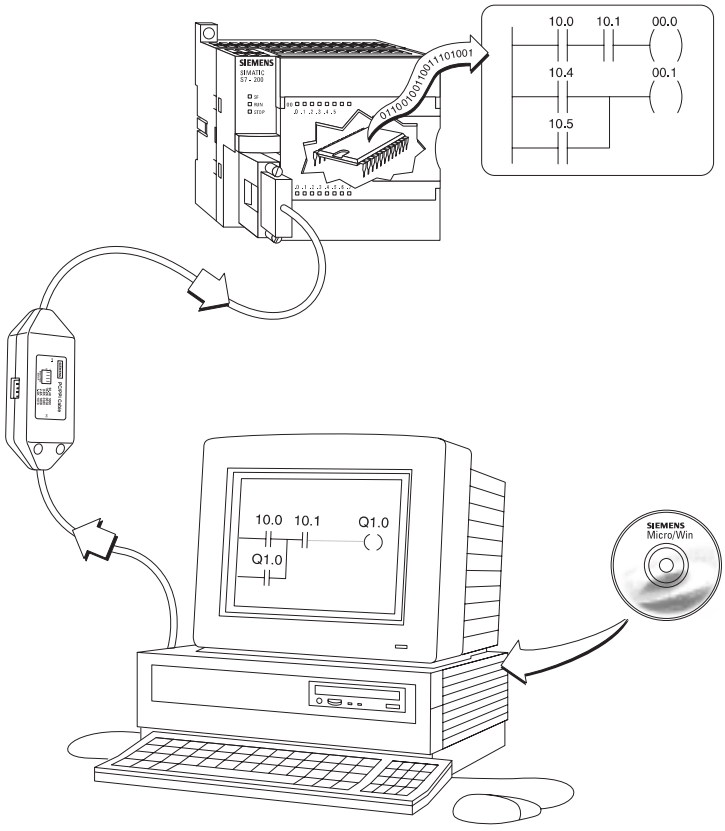
Putting it together

The memory of the S7-200 is divided into three areas: program space, data space, and configurable parameter space.

- Program space stores the ladder logic (LAD) or statement list (STL) program instructions. This area of memory controls the way data space and I/O points are used. LAD or STL instructions are written using a programming device such as a PC, then loaded into program memory of the PLC.

- Data space is used as a working area, and includes memory locations for calculations, temporary storage of intermediate results and constants. Data space includes memory locations for devices such as timers, counters, high-speed counters, and analog inputs and outputs. Data space can be accessed under program control.

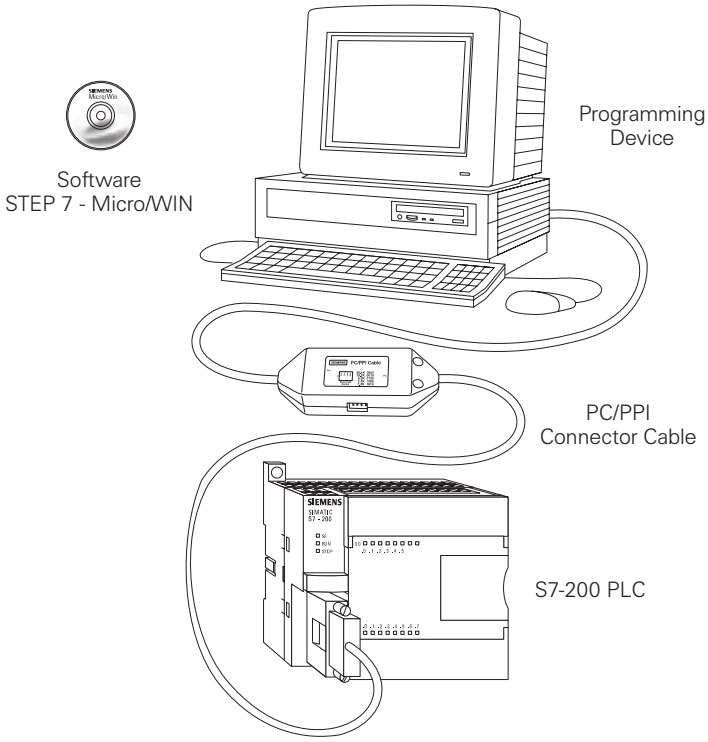
- Configurable parameter space, or memory, stores either the default or modified configuration parameters.



Basic Requirements

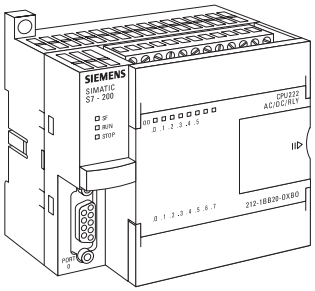
In order to create or change a program, the following items are needed:

- PLC
- Programming Device
- Programming Software
- Connector Cable



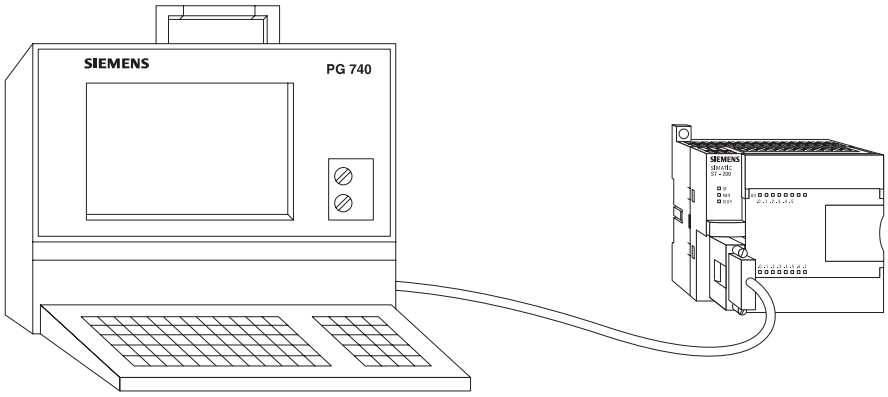
PLC

Throughout this course we will be using the S7-200 because of its ease of use.

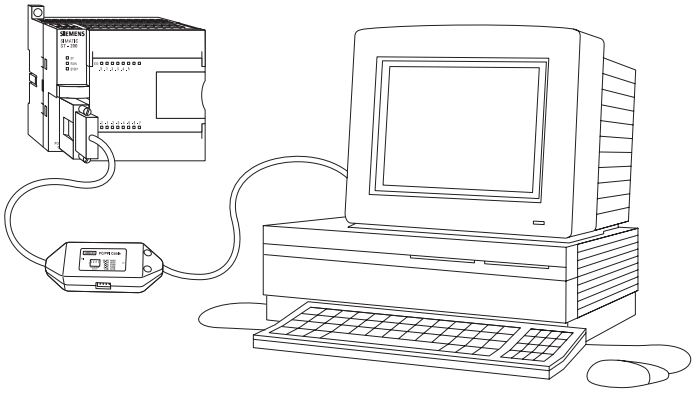


Programming devices

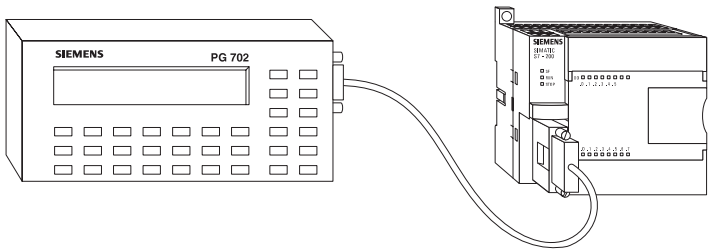
The program is created in a programming device (PG) and then transferred to the PLC. The program for the S7-200 can be created using a dedicated Siemens SIMATIC S7 programming device, such as a PG 720 (not shown) or PG 740, if STEP 7 Micro/WIN software is installed.



A personal computer (PC), with STEP 7 Micro/WIN installed, can also be used as a programming device with the S7-200.



A Siemens PG 702 hand-held programming device can also be used. This device uses a Boolean instruction set built into the S7-200. A Boolean function is a logic function in which there are two possible values, ON or OFF. The PG 702 is useful when making modifications or troubleshooting the PLC. It should be noted, however, that the PG 702 will not match all the functions of Micro/WIN.

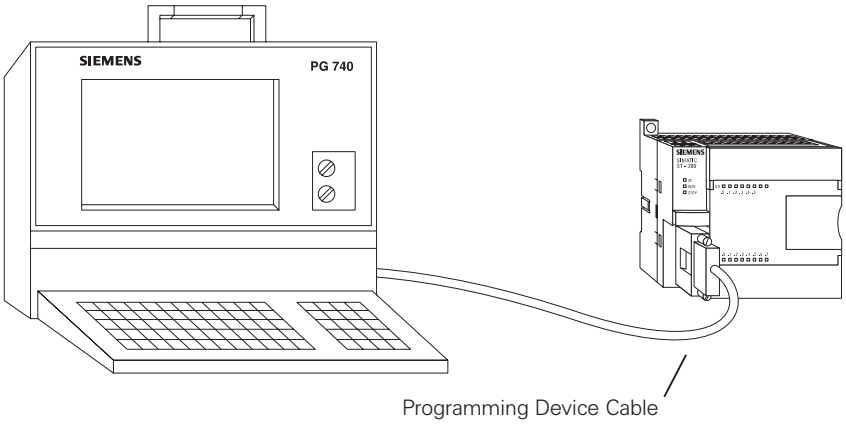


Software

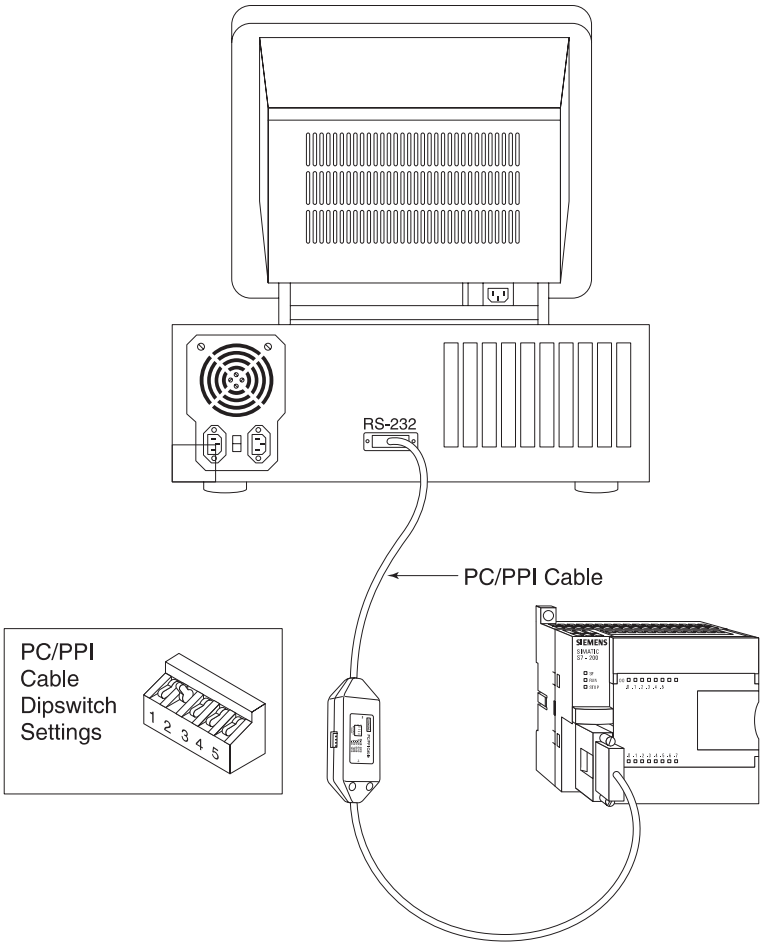
A software program is required in order to tell the PLC what instructions it must follow. Programming software is typically PLC specific. A software package for one PLC, or one family of PLCs, such as the S7 family, would not be useful on other PLCs. The S7-200 uses a Windows based software program called STEP 7-Micro/WIN. The PG 720 and PG 740 have STEP 7 software pre-installed. The PG 702 programming device uses a Boolean instruction set. It is important to note that the PG 702 does not match all the functions of Micro/WIN. Micro/WIN is installed on a personal computer in a similar manner to any other computer software.

Connector cables PPI (point-to-point interface)

Connector cables are required to transfer data from the programming device to the PLC. Communication can only take place when the two devices speak the same language or protocol. Communication between a Siemens programming device and the S7-200 is referred to as PPI protocol (point-to-point interface). An appropriate cable is required for a programming device such as a PG 720, PG 740, or PG 702. The S7-200 uses a 9-pin, D-connector. This is a straight-through serial device that is compatible with Siemens programming devices (MPI port) and is a standard connector for other serial interfaces.



A special cable, referred to as a PC/PPI cable, is needed when a personal computer is used as a programming device. This cable allows the serial interface of the PLC to communicate with the RS-232 serial interface of a personal computer. DIP switches on the PC/PPI cable are used to select an appropriate speed (baud rate) at which information is passed between the PLC and the computer.



Review 2

1. A switch or a pushbutton is a _____ input.

2. A lamp or a solenoid is an example of a _____ output.

3. The _____ makes decisions and executes control instructions based on the input signals.

4. _____ _____ is a PLC programming language that uses components resembling elements used in a line diagram.

5. A _____ consists of one or more instructions that accomplish a task.

6. Memory is divided into three areas: _____ , _____ , and _____ _____ space.

7. When talking about computer or PLC memory, 1K refers to _____ bits, bytes, or words.

8. Software that is placed in hardware is called _____ .

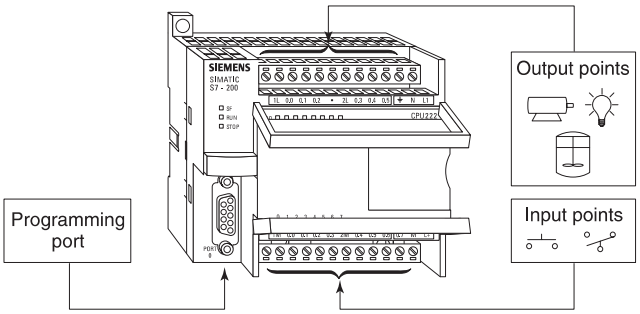
9. Which of the following is <u>not</u> required when creating or changing a PLC program?

   a. PLC
   b. Programming Device
   c. Programming Software
   d. Connector Cable
   e. Printer

10. A special cable, referred to as a _____ cable, is needed when a personal computer is used as a programming device.

S7-200 Micro PLCs

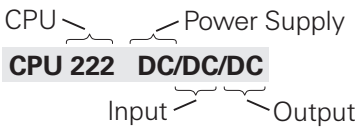
The S7-200 Micro PLC is the smallest member of the SIMATIC S7 family of programmable controllers. The central processing unit (CPU) is internal to the PLC. Inputs and outputs (I/O) are the system control points. Inputs monitor field devices, such as switches and sensors. Outputs control other devices, such as motors and pumps. The programming port is the connection to the programming device.



S7-200 models

There are three S7-200 CPU types: S7-221, S7-222, and S7-224. There are three power supply configurations for each CPU type. The model description indicates the type of CPU, the power supply, the type of input, and the type of output.

Model Description	Power Supply	Input Types	Output Types
221 DC/DC/DC 221 AC/DC/Relay	20.4-28.8 VDC 85-264 VAC 47-63 Hz	6 DC Inputs 6 DC Inputs	4 DC Outputs 4 Relay Outputs
222 DC/DC/DC 222 AC/DC/Relay	20.4-28.8 VDC 85-264 VAC 47-63 Hz	8 DC Inputs 8 DC Inputs	6 DC Outputs 6 Relay Outputs
224 DC/DC/DC 224 AC/DC/Relay	20.4-28.8 VDC 85-264 VAC 47-63 Hz	14 DC Inputs 14 DC Inputs	10 DC Outputs 10 Relay Outputs



S7-200 Features

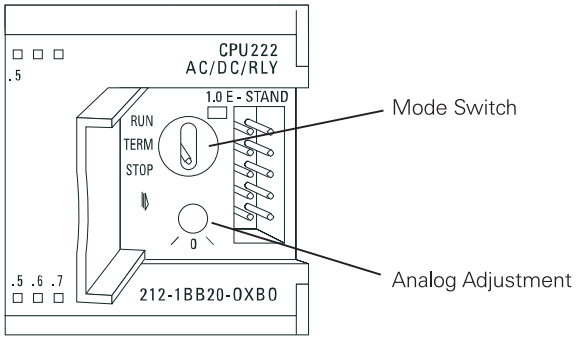
The S7-200 family includes a wide variety of CPUs and features. This variety provides a range of features to aid in designing a cost-effective automation solution. The following table provides a summary of the major features, many of which will be covered in this course.

Feature	CPU 221	CPU 222	CPU 224
Memory			
Program	2048 Words	2048 Words	4096 Words
User Data	1024 Words	1024 Words	2560 Words
Memory Type	EEPROM	EEPROM	EEPROM
Memory Cartridge	EEPROM	EEPROM	EEPROM
Data Backup	50 Hours	50 Hours	190 Hours
Local I/O			
Local I/O	6 In/4 Out	8 In/6 Out	14 In 10 Out
Expansion	None	2 Modules	7 Modules
Total I/O			
Digital I/O Image Size	256 (128 In/128 Out)	256 (128 In/128 Out)	256 (128 In/128 Out)
Digital I/O Physical Size	10	62	128
Analog I/O Image Size	None	16 In/16 Out	16 In/16 Out
Analog I/O Physical Size	None	12 In/12 Out	12 In/12 Out
Instructions			
Boolean Execution Speed	0.37 μs/Inst.	0.37 μs/Inst.	0.37 μs/Inst.
Internal Relays	256	256	256
Counters/Timers	256/256	256/256	256/256
Sequential Control Relays	256	256	256
For/Next Loops	Yes	Yes	Yes
Integer Math (+-*/)	Yes	Yes	Yes
Real Math (+-*/)	Yes	Yes	Yes
Enhanced Features			
Built-In High-Speed Counter	4 (20 KHz)	4 (20 KHz)	6 (20 KHz)
Analog Adjustments	1	1	2
Pulse Outputs	2 (20 KHz, DC)	2 (20 KHz, DC)	2 (20 KHz, DC)
Communication Interrupts	1 Transmit/2 Receive	1 Transmit/2 Receive	1 Transmit/2 Receive
Timed Interrupts	2 (1ms - 255ms)	2 (1ms - 255ms)	2 (1ms - 255ms)
Hardware Input Interrupts	4	4	4
Real-Time Clock	Yes (Cartridge)	Yes (Cartridge)	Yes (Built-In)
Password Protection	Yes	Yes	Yes
Communications			
Number of Ports	1 (RS-485)	1 (RS-485)	1 (RS-485)
Protocols Supported Port 0	PPI, MPI Slave, Freeport	PPI, MPI Slave, Freeport	PPI, MPI Slave, Freeport
Profibus Peer-to-Peer	(NETR/NETW)	(NETR/NETW)	(NETR/NETW)

Mode switch and Analog adjustment

When the mode switch is in the RUN position the CPU is in the run mode and executing the program. When the mode switch is in the STOP position the CPU is stopped. When the mode switch is in the TERM position the programming device can select the operating mode.

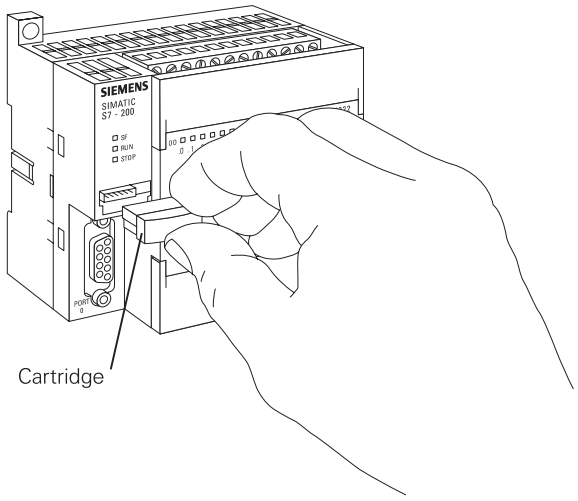
The analog adjustment is used to increase or decrease values stored in special memory. These values can be used to update the value of a timer or counter, or can be used to set limits.



Optional Cartridge

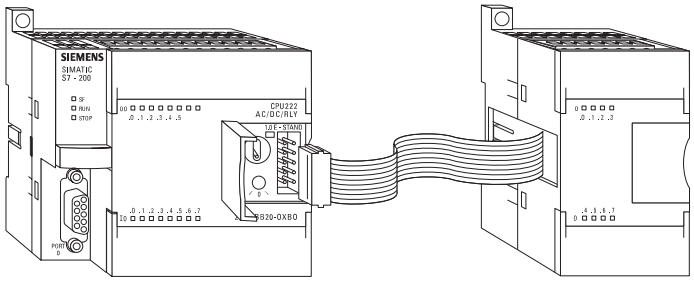
The S7-200 supports an optional memory cartridge that provides a portable EEPROM storage for your program. The cartridge can be used to copy a program from one S7-200 PLC to a like S7-200 PLC.

In addition, two other cartridges are available. A real-time clock with battery is available for use on the S7-221 and S7-222. The battery provides up to 200 days of data retention time in the event of a power loss. The S7-224 has a real-time clock built in. Another cartridge is available with a battery only.

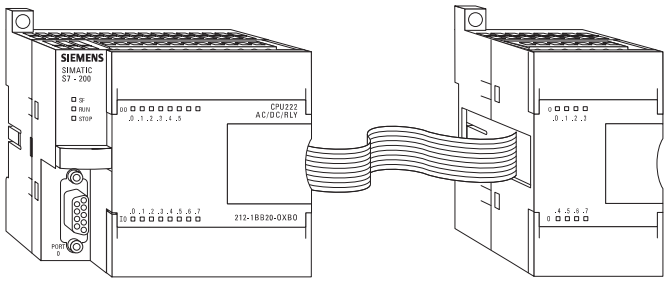


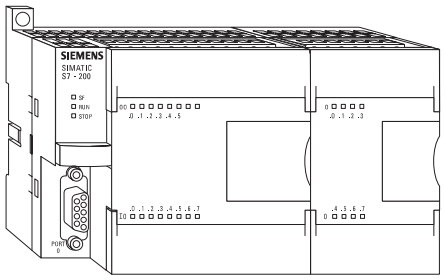
Expansion modules

The S7-200 PLCs are expandable. Expansion modules contain additional inputs and outputs. These are connected to the base unit using a ribbon connector.



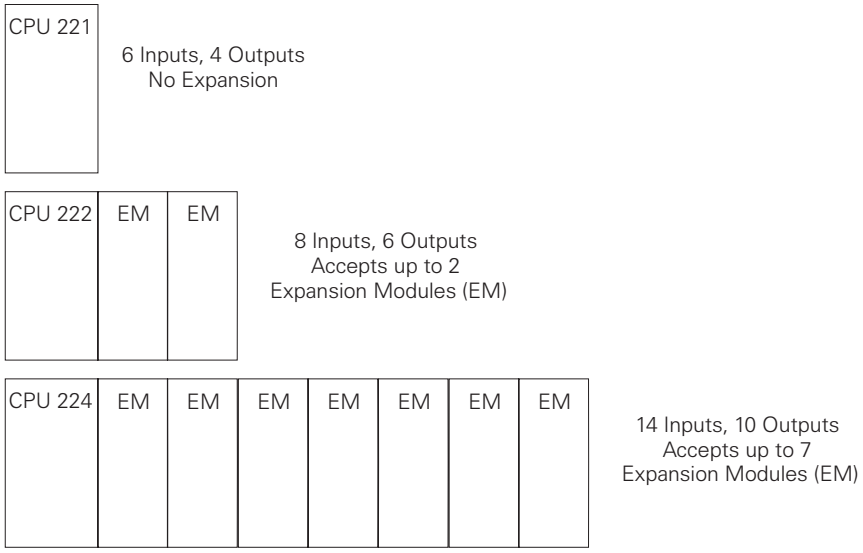
The ribbon connector is protected by a cover on the base unit. Side-by-side mounting completely encloses and protects the ribbon connector.





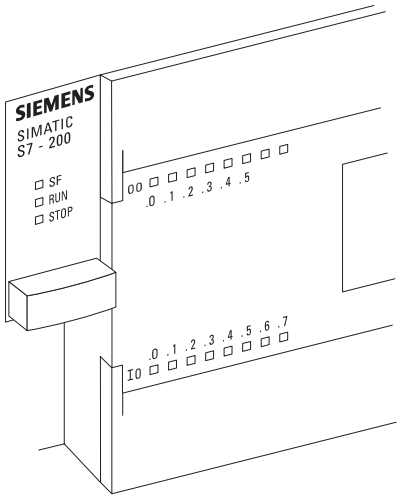
Available expansion

The S7-221 comes with 6 digital inputs and 4 digital outputs. These are not expandable. The S7-222 comes with 8 digital inputs and 6 digital outputs. The 222 will accept up to 2 expansion modules. The S7-224 comes with 14 digital inputs and 10 digital outputs. The 224 will accept up to 7 expansion modules.



Status indicators

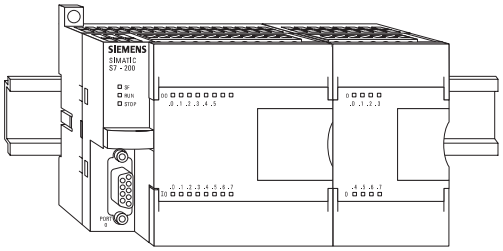
The CPU status indicators reflect the current mode of CPU operation. If, for example, the mode switch is set to the RUN position, the green RUN indicator is lit. When the mode switch is set to the STOP position, the yellow STOP indicator is lit.



The I/O status indicators represent the On or Off status of corresponding inputs and outputs. When the CPU senses an input is on, the corresponding green indicator is lit.

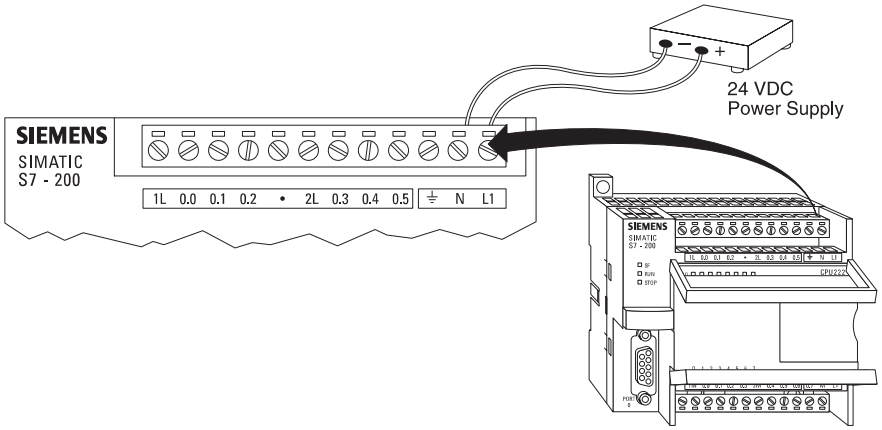
Installing

The S7-200 can be installed in one of two ways. A DIN clip allows installation on a standard DIN rail. The DIN clip snaps open to allow installation and snaps closed to secure the unit on the rail. The S7-200 can also be panel mounted using installation holes located behind the access covers.

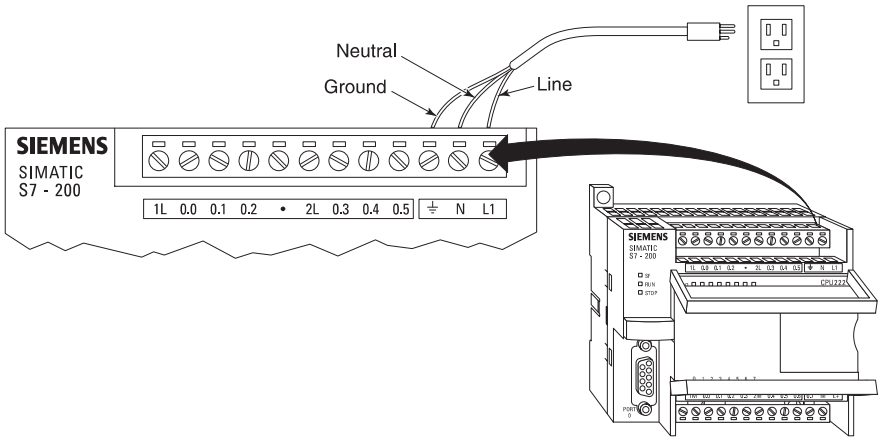


External power supply sources

An S7-200 can be connected to either a 24 VDC or a 120/230 VAC power supply depending on the CPU. An S7-200 DC/DC/DC would be connected to a 24 VDC power supply. The power supply terminals are located on the far right side of the top terminal strip.



An S7-200 AC/DC/Relay would be connected to a 120 or 230 VAC power supply.



I/O numbering

S7-200 inputs and outputs are labeled at the wiring terminations and next to the status indicators. These alphanumeric symbols identify the I/O address to which a device is connected. This address is used by the CPU to determine which input is present and which output needs to be turned on or off. **I** designates a discrete input and **Q** designates a discrete output. The first number identifies the byte, the second number identifies the bit. Input I0.0, for example, is byte 0, bit 0.

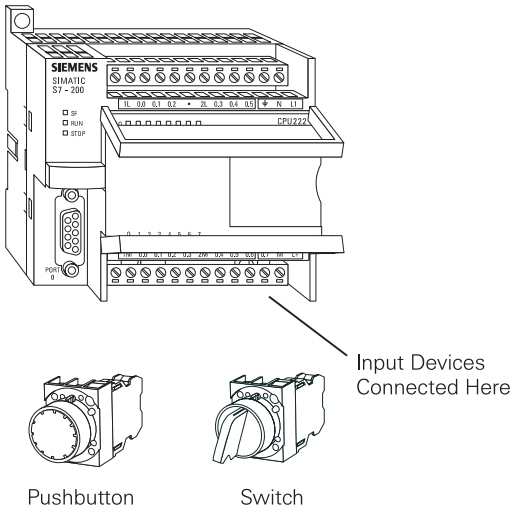
I0.0 = Byte 0, Bit 0
I0.1 = Byte 0, Bit 1
I1.0 = Byte 1, Bit 0
I1.1 = Byte 1, Bit 1

The following table identifies the input and output designations.

I0.0	1st Input	I1.0	9th Input	Q0.0	1st Output	Q1.0	9th Output
I0.1	2nd Input	I1.1	10th Input	Q0.1	2nd Output	Q1.1	10th Output
I0.2	3rd Input	I1.2	11th Input	Q0.2	3rd Output		
I0.3	4th Input	I1.3	12th Input	Q0.3	4th Output		
I0.4	5th Input	I1.4	13th Input	Q0.4	5th Output		
I0.5	6th Input	I1.5	14th Input	Q0.5	6th Output		
I0.6	7th Input			Q0.6	7th Output		
I0.7	8th Input			Q0.7	8th Output		

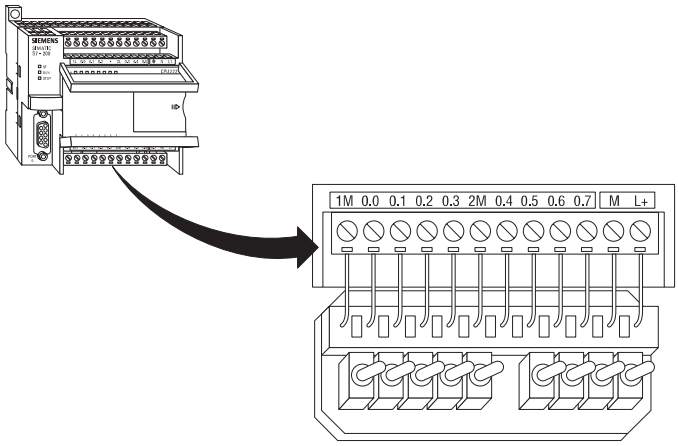
Inputs

Input devices, such as switches, pushbuttons, and other sensor devices are connected to the terminal strip under the bottom cover of the PLC.



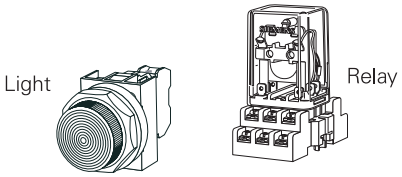
Input simulator

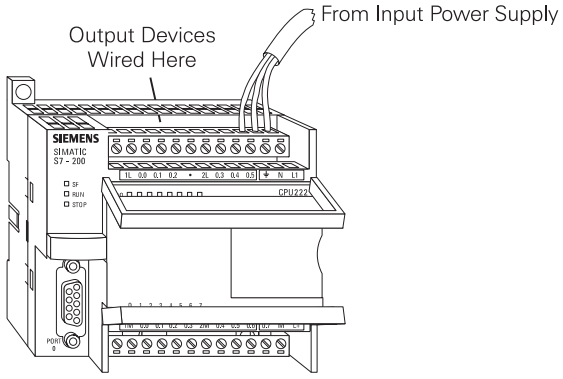
A convenient method of testing a program is to wire toggle switches to the inputs. Input simulators with prewired toggle switches are available for the S7-200s. Switches are wired between the 24 VDC power supply (L+) and the inputs. For example, the switch on the far left is wired between the first input (0.0) and L+. When the switch is closed, 24 VDC is applied to the input. This is referred to as a logic 1. When the switch is open, 0 VDC is applied to the input. This is referred to as a logic 0.



Outputs

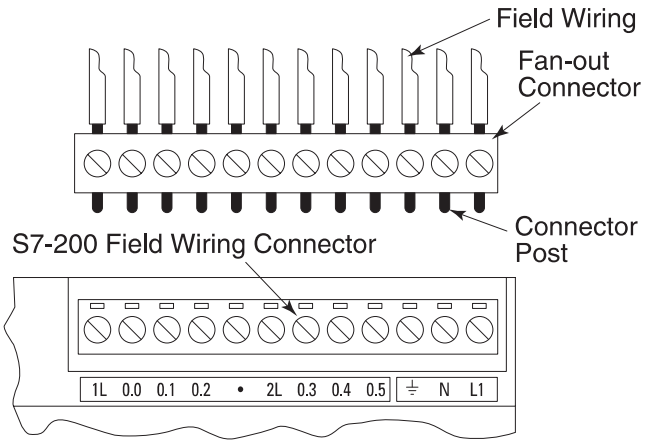
Output devices, such as relays, are connected to the terminal strip under the top cover of the PLC. When testing a program, it is not necessary to connect output devices. The LED status indicators signal if an output is active.





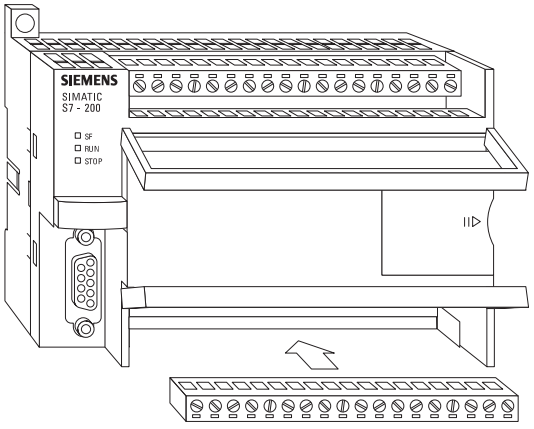
Optional connector

An optional fan-out connector allows for field wiring connections to remain fixed when removing or replacing an S7-221 or 222. The appropriate connector slides into either the input, output, or expansion module terminals.



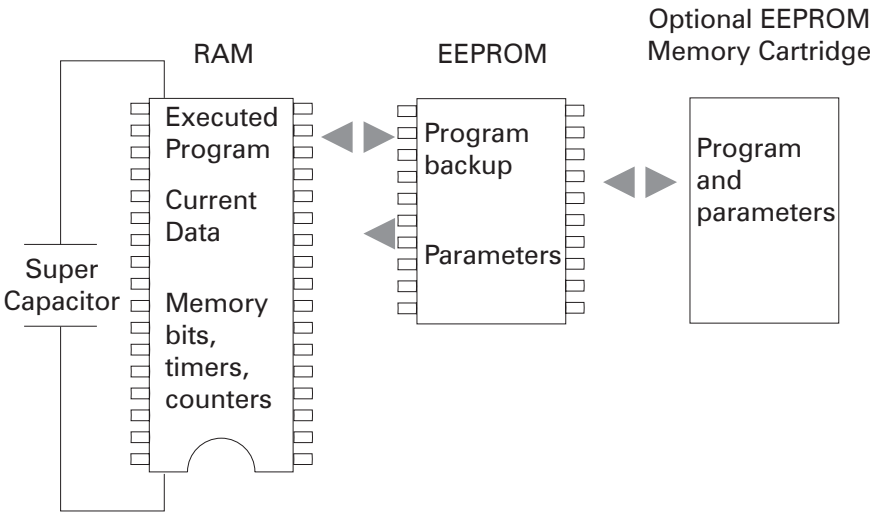
Removable Terminal Strip

The S7-224 does not have an optional fan-out connector. Instead, the terminal strips are removable. This allows the field wiring connections to remain fixed when removing or replacing the S7-224.



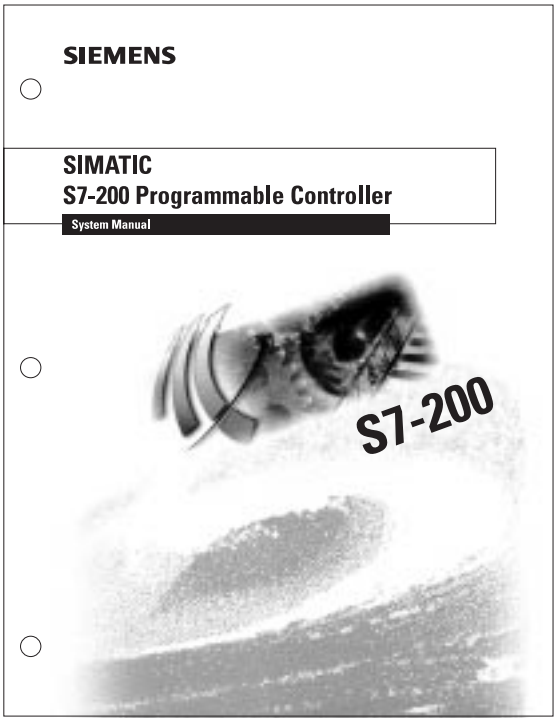
Super capacitor

A super capacitor, so named because of its ability to maintain a charge for a long period of time, protects data stored in RAM in the event of a power loss. The RAM memory is typically backed up on the S7-221 and 222 for 50 hours, and on the S7-224 for 72 hours.



Reference manual

The **SIMATIC S7-200 Programmable Controller System Manual** provides complete information on installing and programming the S7-200 PLCs.



Review 3

1. The three models of S7-200 are _____ , _____ and _____ .

2. Which of the following is not available on an S7-221?

   a. Mode Switch
   b. Expansion Port
   c. Programming Port
   d. Status Indicators

3. An S7-222 can have a maximum of _____ expansion modules and an S7-224 can have a maximum of _____ expansion modules.

4. An S7-222 has _____ DC inputs and _____ DC outputs.

5. An S7-224 has _____ DC inputs and _____ DC outputs.

6. The fourth output of an S7-200 would be labeled _____ .

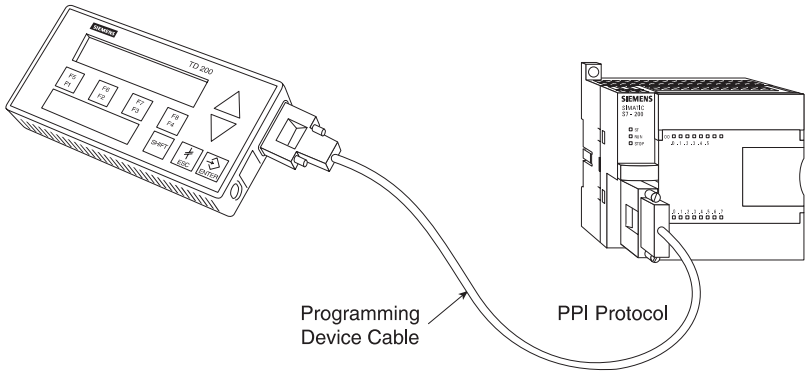
7. S7-200 can be panel mounted or installed on a _____ rail.

8. A super capacitor will maintained data stored in RAM for up _____ hours on an S7-222 and up to _____ hours on an S7-224.

Connecting External Devices

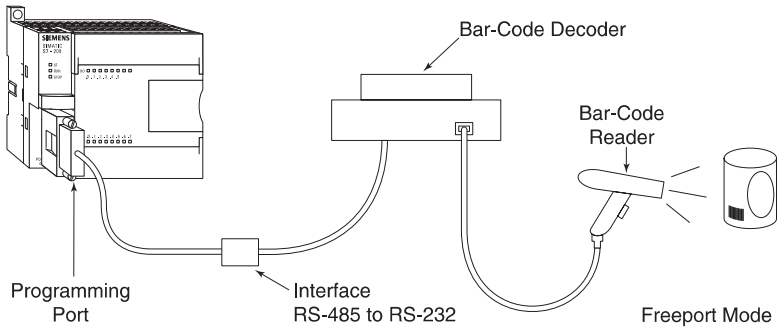
TD200

The S7-200 programming port can be used to communicate with a variety of external devices. One such device is the TD200 text display unit. The TD200 displays messages read from the S7-200, allows adjustment of designated program variables, provides the ability to force, and permits setting of the time and date. The TD200 can be connected to an external power supply or receive its power from the S7-200.



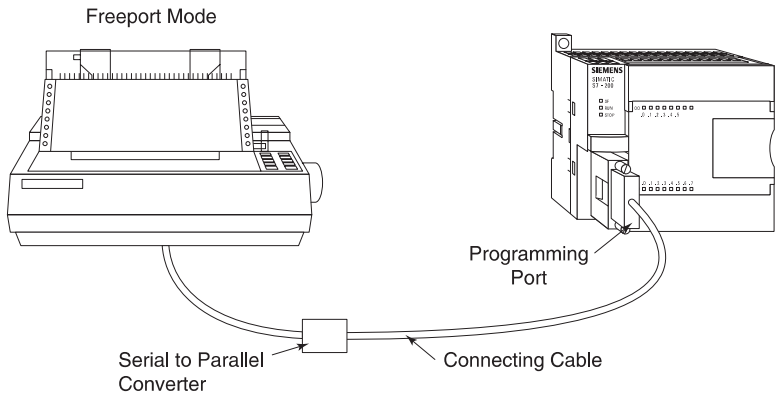
Freeport mode

The programming port has a mode called freeport mode. Freeport mode allows connectivity to various intelligent sensing devices such as a bar code reader.



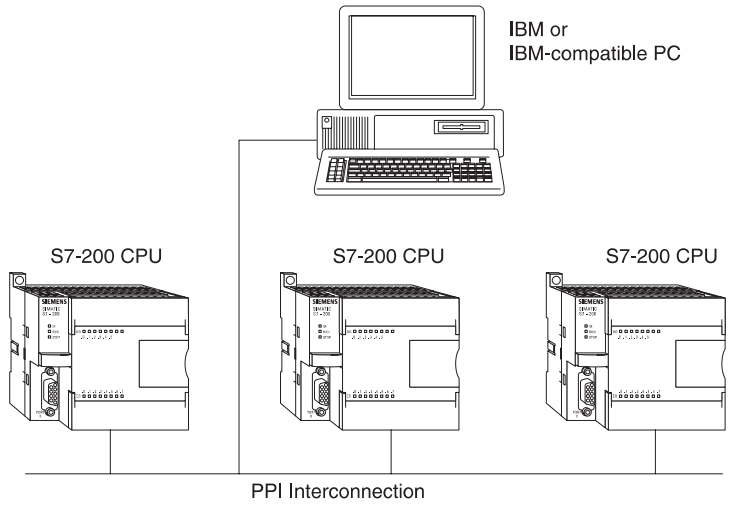
Printer

Freeport mode can also be used to connect to a non-SIMATIC printer.



Interconnection

It is possible to use one programming device to address multiple S7-200 devices on the same communication cable. A total of 31 units can be interconnected without a repeater.



Programming A PLC

STEP 7-Micro/WIN is the program software used with the S7-200 PLC to create the PLC operating program. STEP 7 consists of a number of instructions that must be arranged in a logical order to obtain the desired PLC operation. These instructions are divided into three groups: standard instructions, special instructions, and high-speed instructions.

Standard instructions

Standard instructions consists of instructions that are found in most programs. Standard instructions include; timer, counter, math, logical, increment/decrement/invert, move, and block instructions.

Special instructions

Special instructions are used to manipulate data. Special instructions include shift, table, find, conversion, for/next, and real-time instructions.

High-speed instructions

High-speed instructions allow for events and interrupts to occur independent of the PLC scan time. These include high-speed counters, interrupts, output, and transmit instructions.

It is not the purpose of this text to explain all of the instructions and capabilities. A few of the more common instructions necessary for a basic understanding of PLC operation will be discussed. PLC operation is limited only by the hardware capabilities and the ingenuity of the person programming it. Refer to the **SIMATIC S7-200 Programmable Controller System Manual** for detailed information concerning these instructions.

Micro/WIN

The programming software can be run Off-line or On-line. Off-line programming allows the user to edit the ladder diagram and perform a number of maintenance tasks. The PLC does not need to be connected to the programming device in this mode. On-line programming requires the PLC to be connected to the programming device. In this mode program changes are downloaded to the PLC. In addition, status of the input/output elements can be monitored. The CPU can be started, stopped, or reset.

Symbols

In order to understand the instructions a PLC is to carry out, an understanding of the language is necessary. The language of PLC ladder logic consists of a commonly used set of symbols that represent control components and instructions.

Contacts

One of the most confusing aspects of PLC programming for first-time users is the relationship between the device that controls a status bit and the programming function that uses a status bit. Two of the most common programming functions are the normally open (NO) contact and the normally closed (NC) contact. Symbolically, power flows through these contacts when they are closed. The normally open contact (NO) is true (closed) when the input or output status bit controlling the contact is 1. The normally closed contact (NC) is true (closed) when the input or output status bit controlling the contact is 0.

Normally Open (NO)

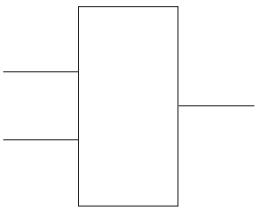
Normally Closed (NC)

Coils

Coils represent relays that are energized when power flows to them. When a coil is energized, it causes a corresponding output to turn on by changing the state of the status bit controlling that output to 1. That same output status bit may be used to control normally open and normally closed contacts elsewhere in the program.

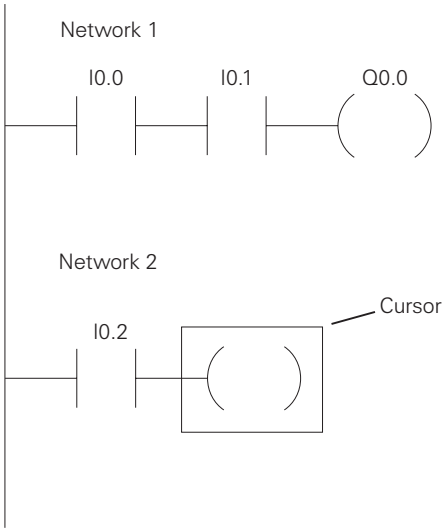
Boxes

Boxes represent various instructions or functions that are executed when power flows to the box. Typical box functions are timers, counters, and math operations.



Entering elements

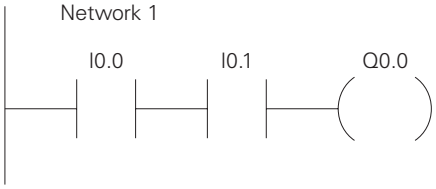
Control elements are entered in the ladder diagram by positioning the cursor and selecting the element from a lists. In the following example the cursor has been placed in the position to the right of I0.2. A coil was selected from a pull-down list and inserted in this position.



An AND operation

Each rung or network on a ladder represents a logic operation. The following programming example demonstrates an AND operation. Two contact closures and one output coil are placed on network 1. They were assigned addresses I0.0, I0.1, and Q0.0. Note that in the statement list a new logic operation always begins with a load instruction (LD). In this example I0.0 (input 1) and (A in the statement list) I0.1 (input 2) must be true in order for output Q0.0 (output 1) to be true. It can also be seen That I0.0 and I0.1 must be true for Q0.0 to be true by looking at the function block diagram representation.

Ladder Diagram Representation



Statement List Representation

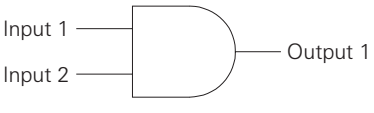
```
Network 1
LD      I0.0
A       I0.1
=       Q0.0
```

Function Block Diagram Representation

Network 1

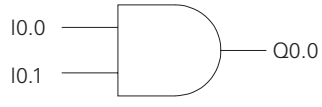
Another way to see how an AND function works is with a Boolean logic diagram. In Boolean logic an AND gate is represented by a number of inputs on the left side. In this case there are two inputs. The output is represented on the right side. It can be seen from the table that both inputs must be a logic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

And (A) Function



Input 1	Input 2	Output 1
0	0	0
0	1	0
1	0	0
1	1	1

And (A) Function

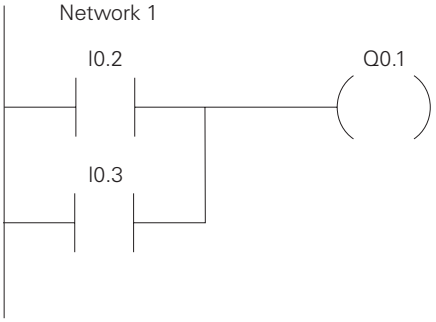


I0.0	I0.1	Q0.0
0	0	0
0	1	0
1	0	0
1	1	1

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

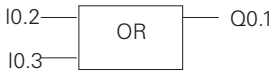


Statement List Representation

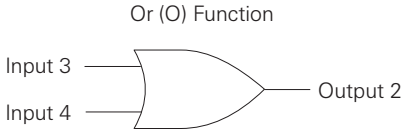
```
Network 1
LD      I0.2
O       I0.3
=       Q0.1
```

Function Block Diagram Representation

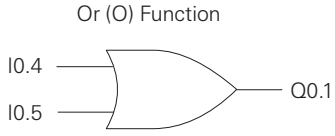
Network 1



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.



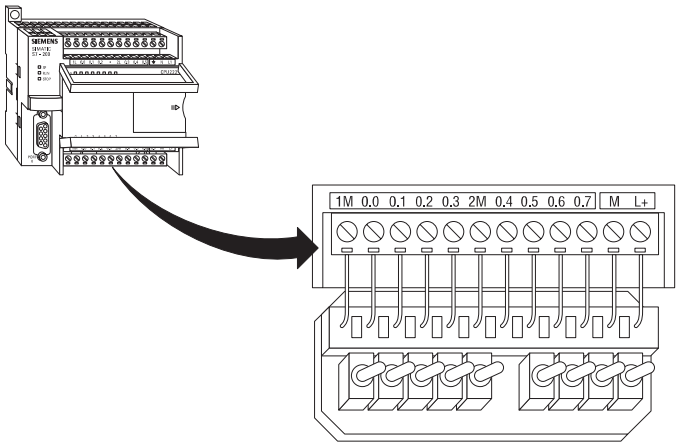
Input 3	Input 4	Output 2
0	0	0
0	1	1
1	0	1
1	1	1



I0.4	I0.5	Q0.1
0	0	0
0	1	1
1	0	1
1	1	1

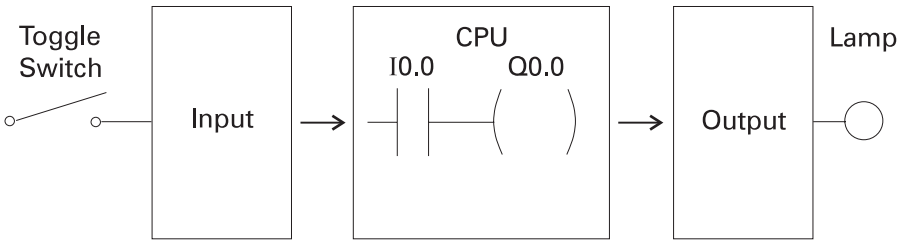
Testing a program

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.

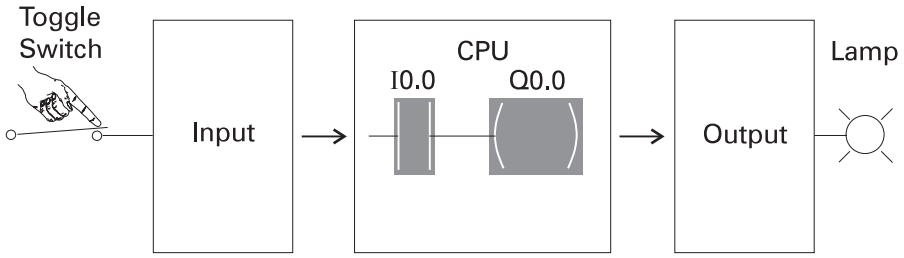


Status functions

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/WIN 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 nonoperated 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).

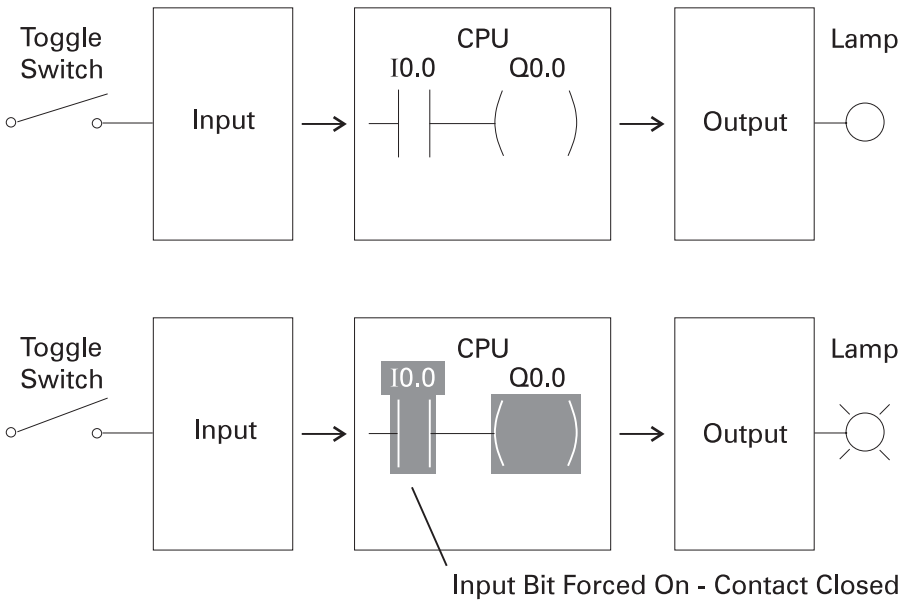


When viewing the ladder diagram in the status mode, control elements that are active, or true (logic 1), are highlighted. In the example shown the toggle switch connected to input 1 has been closed. Power can now flow through the control element associated with input 1 (I0.0) and activate the output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

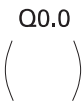
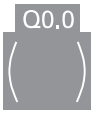


Forcing

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.

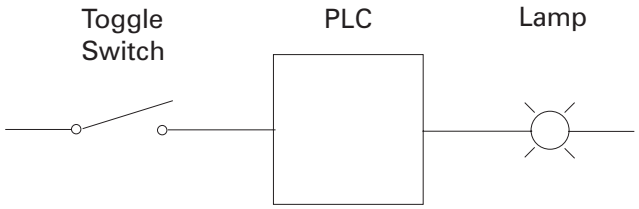


The following table shows the appearance of ladder elements in the Off, forced, and On condition.

	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				

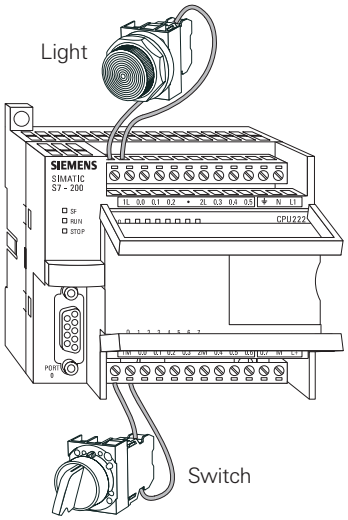
Discrete Inputs/Outputs

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.

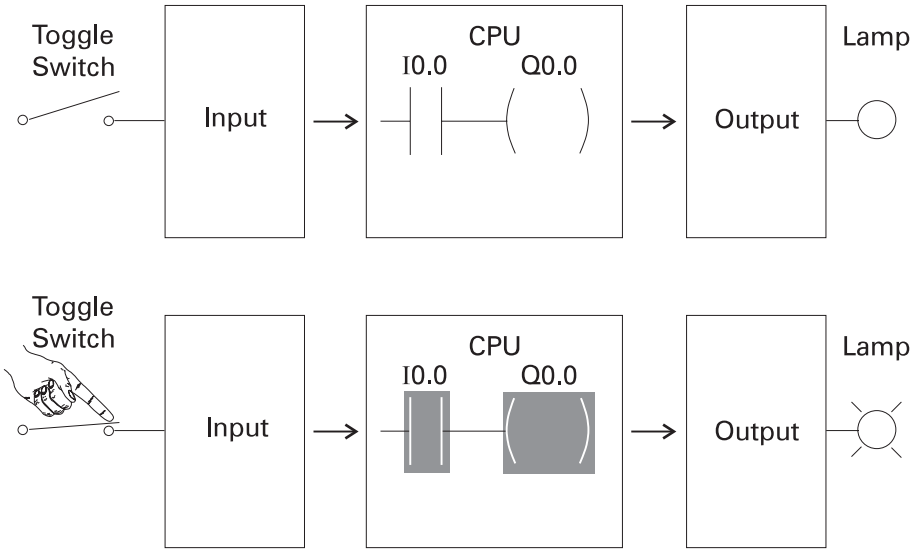


Wiring

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

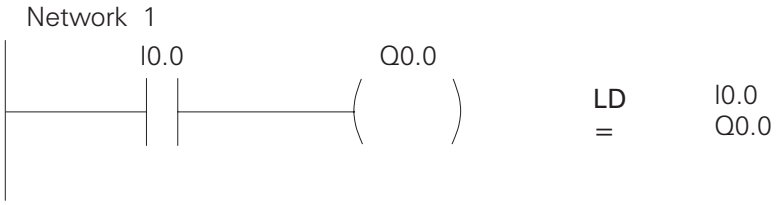


The following drawing illustrates the sequence of events. A switch is wired to the input module of the PLC. A lamp is wired to the output module. The program is in the CPU. The CPU scans the inputs. When it finds the switch open I0.0 receives a binary 0. This instructs Q0.0 to send a binary 0 to the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

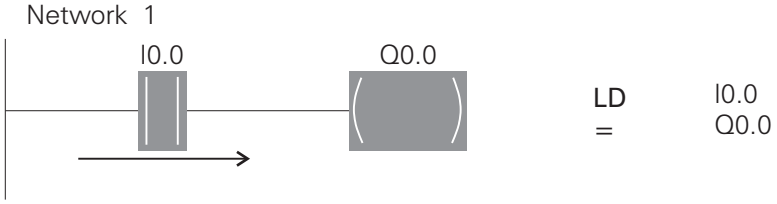


Program instruction

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.

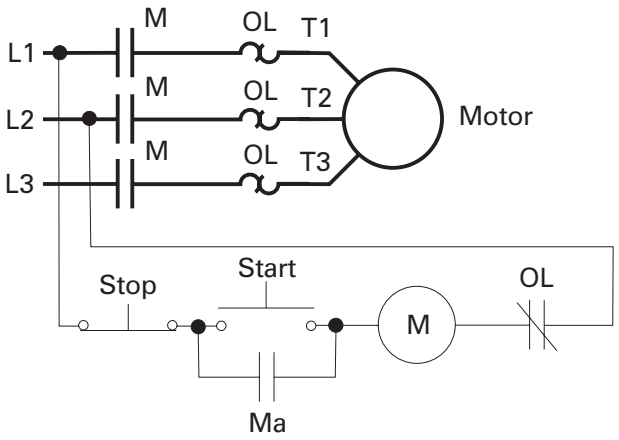


When the switch is closed the CPU receives a logic 1 from input I0.0. The CPU sends a logic 1 to output Q0.0, thus activating Q0.0. The light turns on.

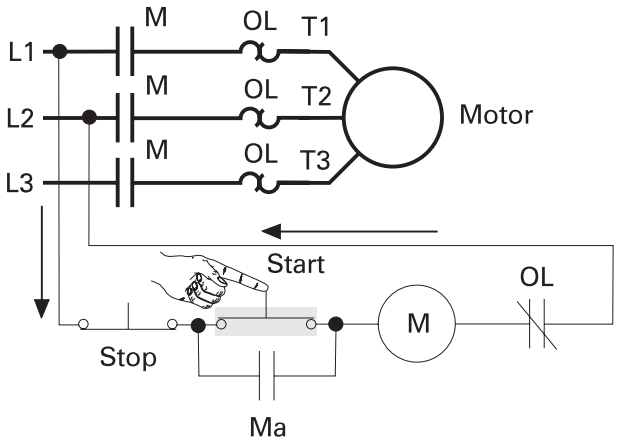


Motor starter example

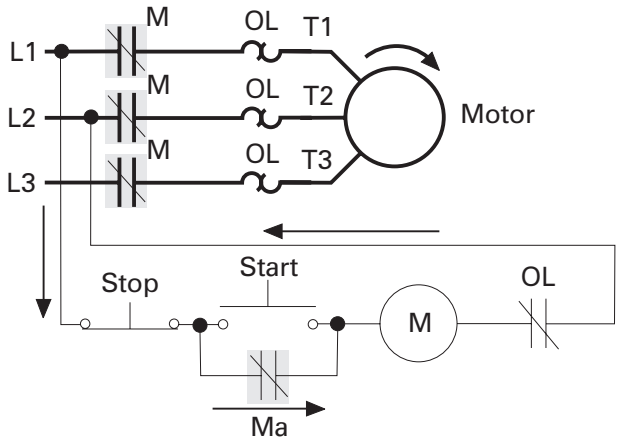
The following example involves a motor start and stop circuit. The line diagram illustrates how a normally open and a normally closed pushbutton might be used in a control circuit. In this example a motor started (M) is wired in series with a normally open momentary pushbutton (Start), a normally closed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



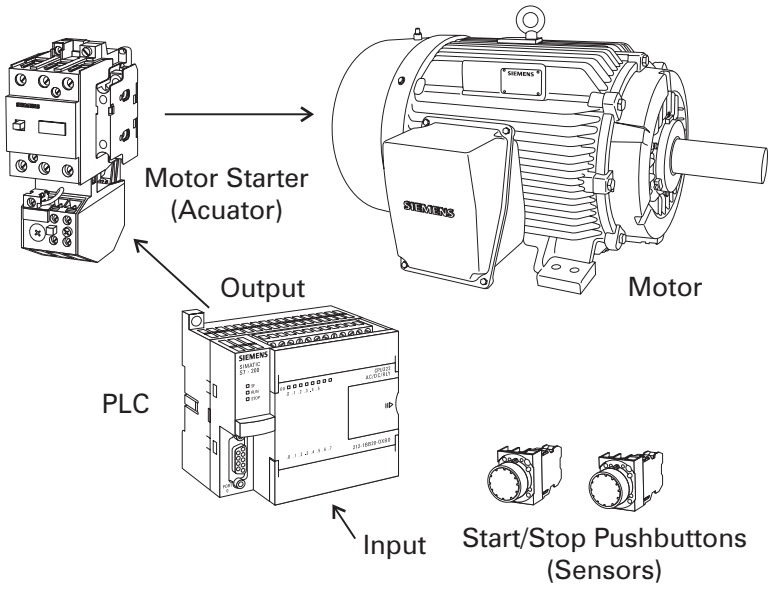
Momentarily depressing the Start pushbutton completes the path of current flow and energizes the motor starter (M).



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.

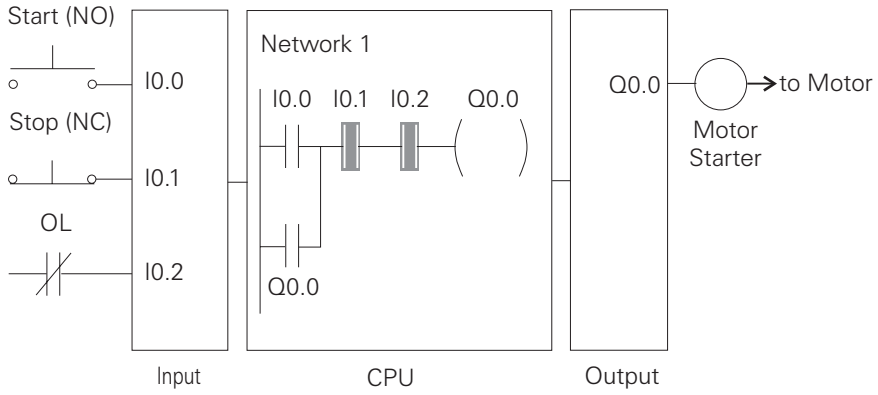


This control task can also be accomplished with a PLC.

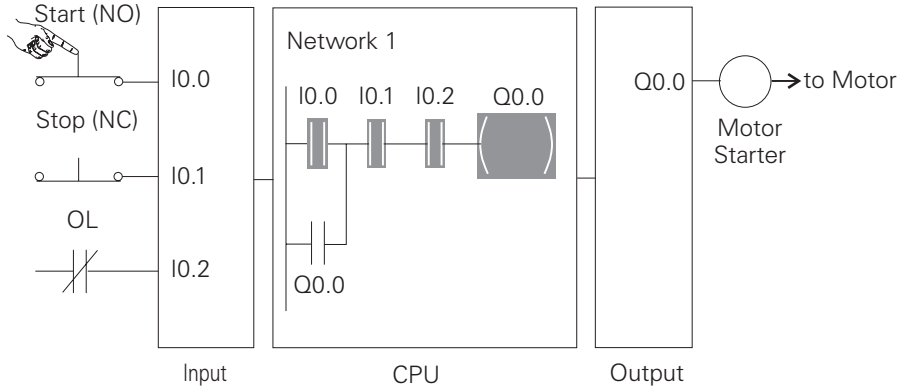


Program instruction

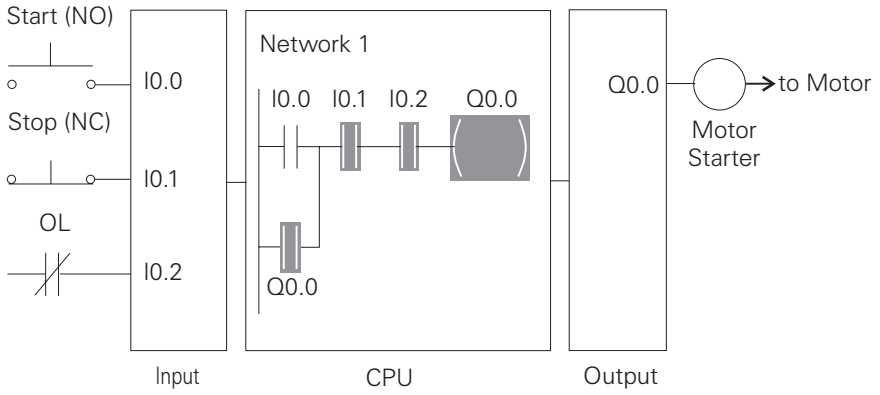
A normally open Start Pushbutton is wired to the first input (I0.0), a normally closed Stop Pushbutton is wired to the second input (I0.1), and normally closed overload relay contacts (part of the motor starter) are connected to the third input (I0.2). The first input (I0.0), second input (I0.1), and 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.



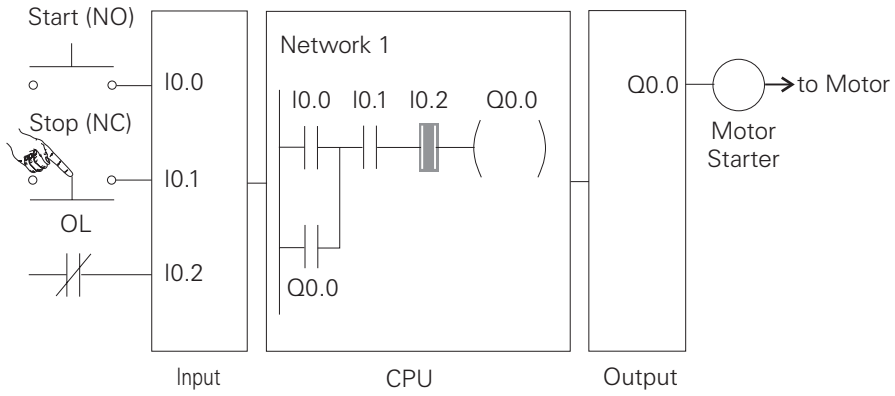
When the Start Pushbutton is depressed the CPU receives a logic 1 from input I0.0. This causes the I0.0 contact to close. All three inputs are now a logic 1. The CPU sends a logic 1 to output Q0.0. The motor starter is energized and the motor starts.



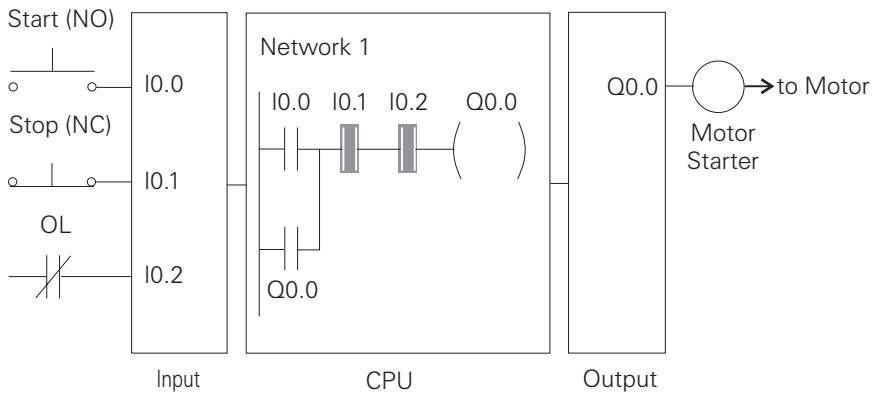
When the Start pushbutton is pressed, output Q0.0 is now true and on the next scan, when normally open contact Q0.0 is solved, the contact will close and output Q0.0 will stay on even if the Start Pushbutton has been released.



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.

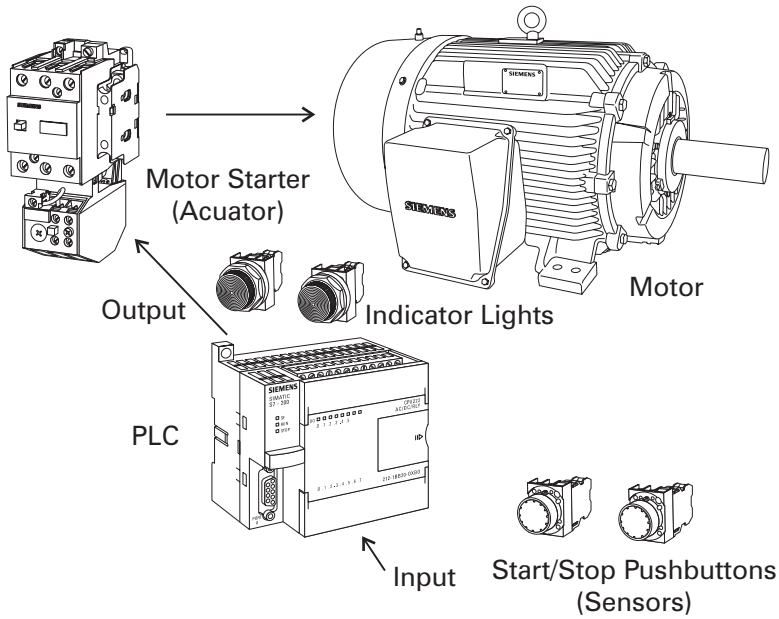


When the Stop Pushbutton is released I0.1 logic function will again be true and the program ready for the next time the Start Pushbutton is pressed.

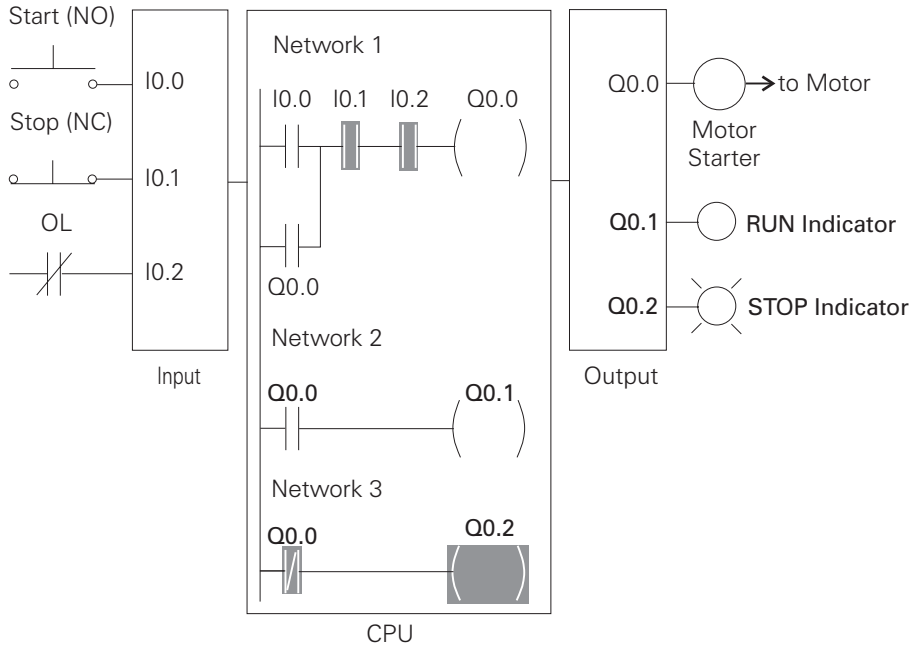


Expanding the application

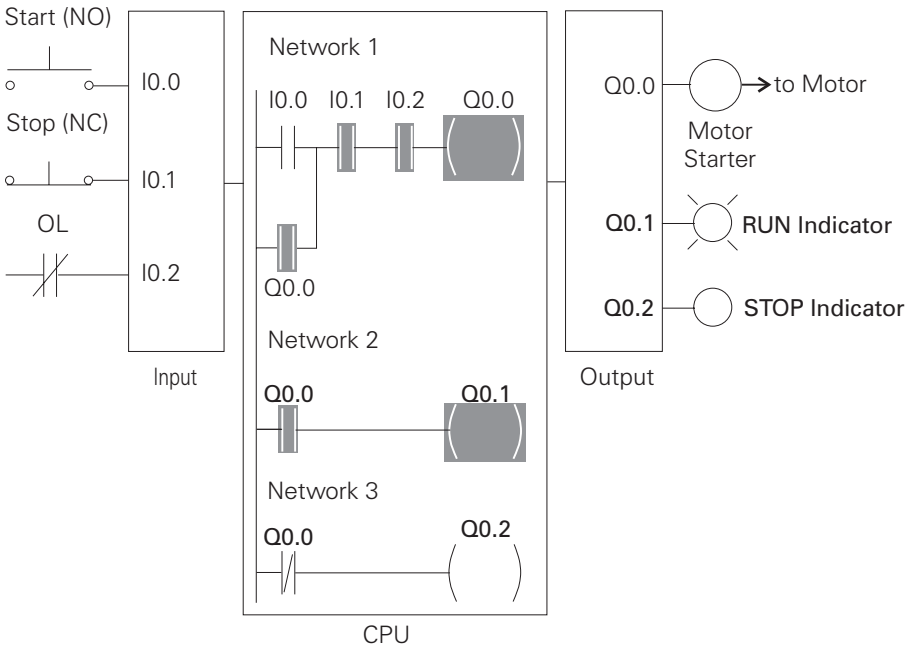
The application can be easily expanded to include indicator lights for RUN and STOP conditions. In this example a RUN indicator light is connected to output Q0.1 and a STOP indicator light is connected to output Q0.2.



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.

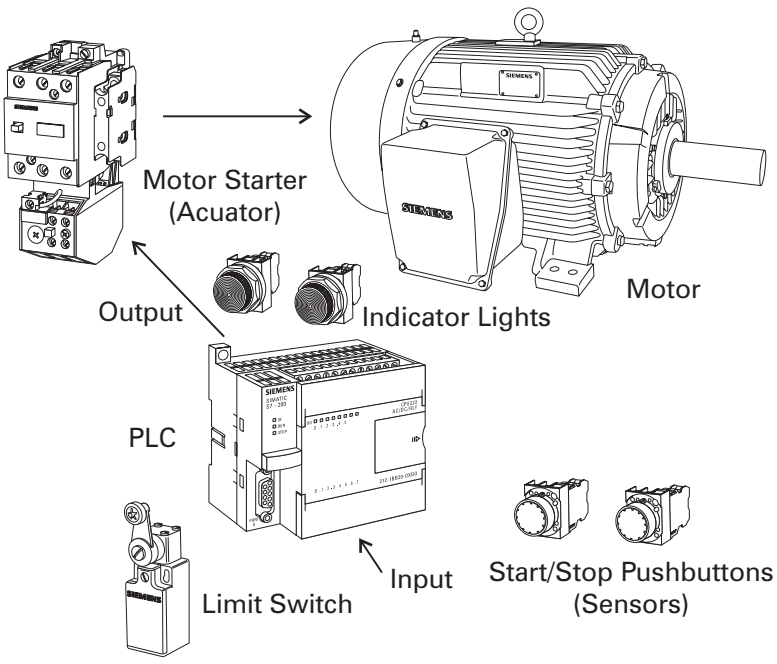


When the PLC starts the motor output Q0.0 is now a logic high (On). The normally open Q0.0 contacts on Network 2 now switch to a logic 1 (closed) and output Q0.1 turns the RUN indicator on. The normally closed Q0.0 contacts on Network 3 switch to a logic 0 (open) and the STOP indicator light connected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

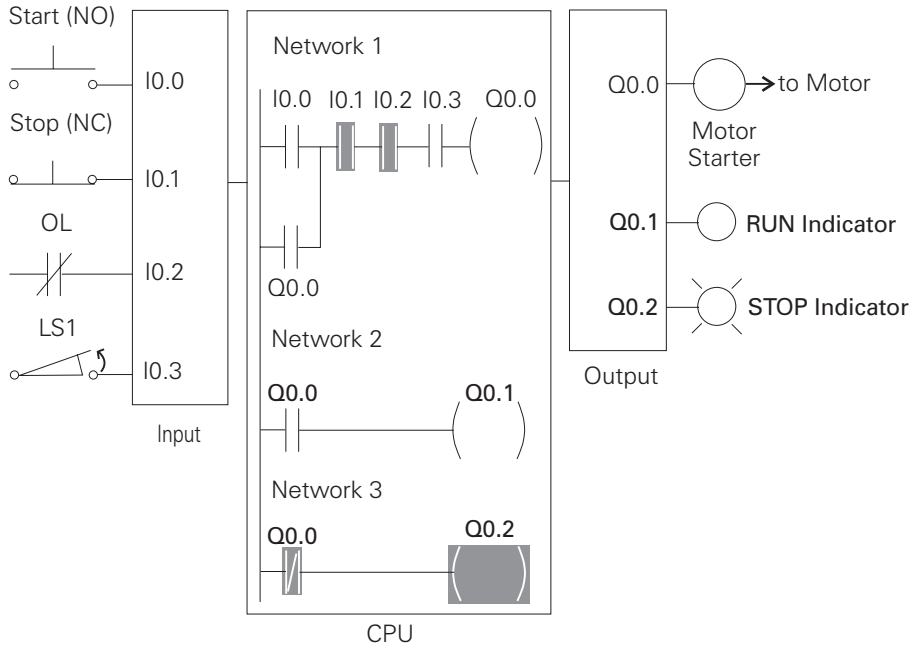


Adding a limit switch

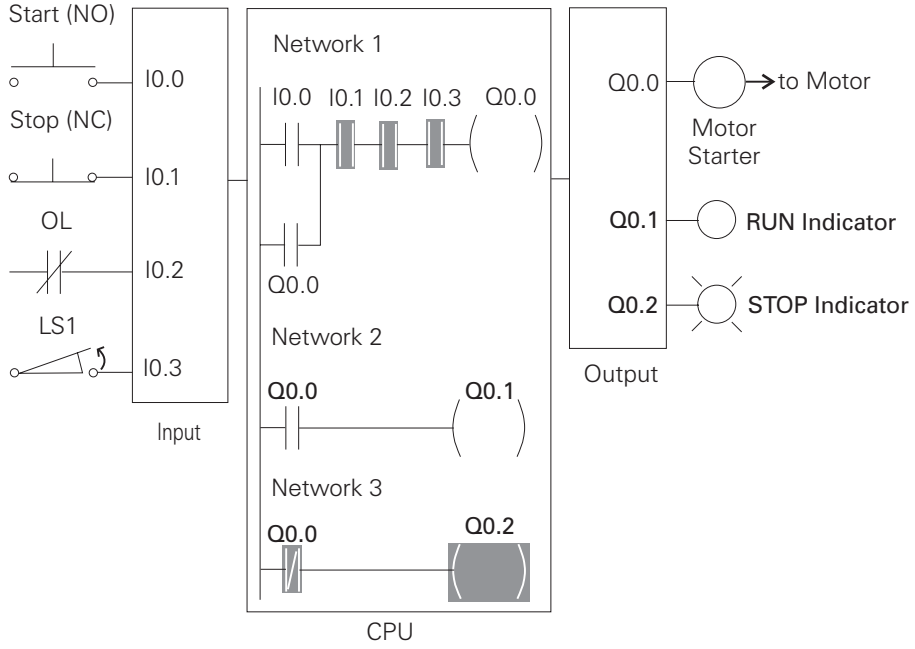
The application can be futher expanded by adding a limit switch with normally open contacts to input I0.3.



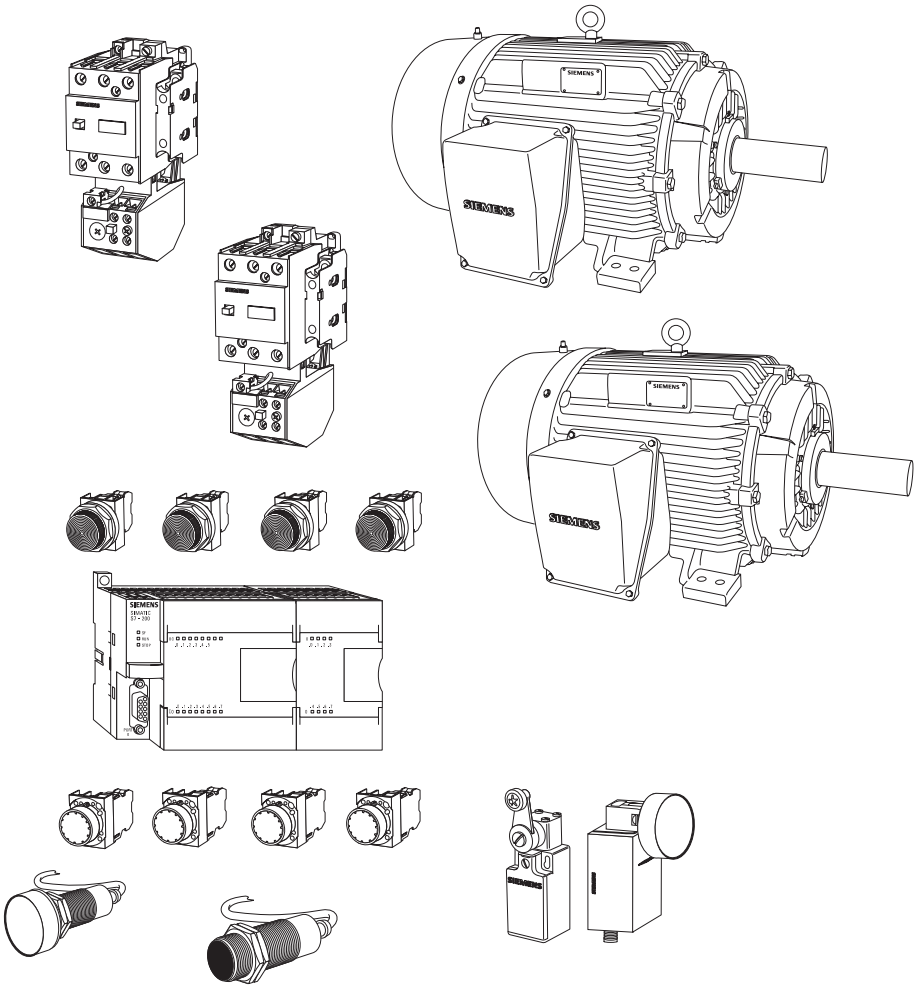
A limit switch could be used to stop the motor or prevent the motor from being started. An access door to the motor, or its associated equipment, is one example of a limit switch's use. If the access door is open, the normally open contacts of LS1 connected to input I0.3 are open and the motor will not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

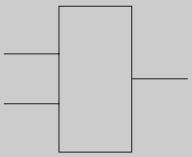


The PLC program can be expanded to accomodate many commercial and industrial applications. Additional Start/Stop Pushbuttons and indicator lights can be added for remote operation, or control of a second motor starter and motor. Overtravel limit switches can be added along with proximity switches for sensing object position. 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:

a. _____ 

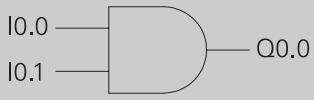
b. _____ 

c. _____ 

2. In a statement list each new logic operation begins with a _____ instruction.

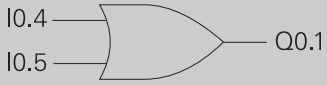
3. Complete the following tables:

And (A) Function



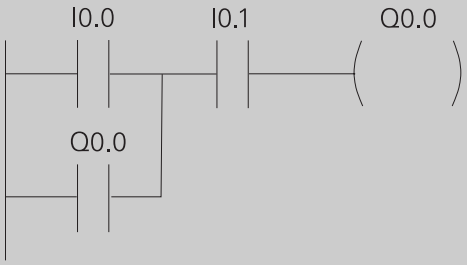
I0.0	I0.1	Q0.0
0	0	a. _____
0	1	b. _____
1	0	c. _____
1	1	d. _____

Or (O) Function



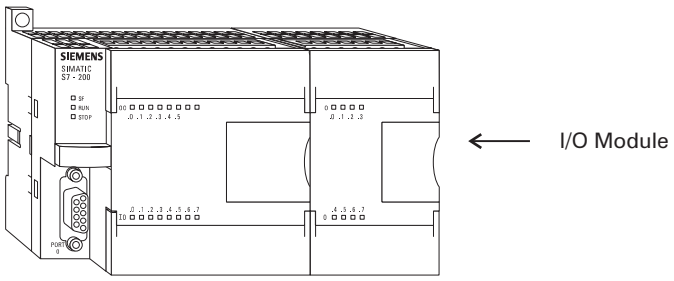
I0.4	I0.5	Q0.1
0	0	e. _____
0	1	f. _____
1	0	g. _____
1	1	h. _____

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



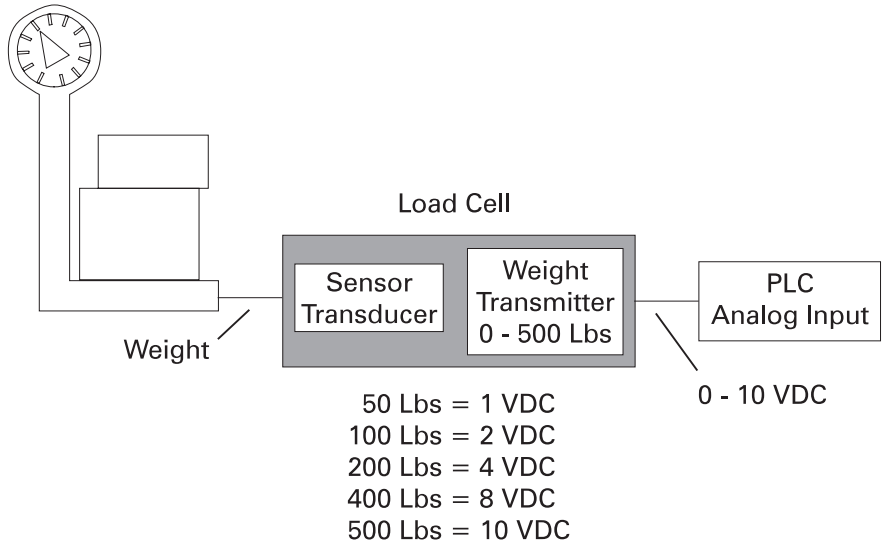
Analog Inputs and Outputs

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 converts analog values into a 12-bit digital representation. The digital values are transferred to the PLC for use in register or word locations.



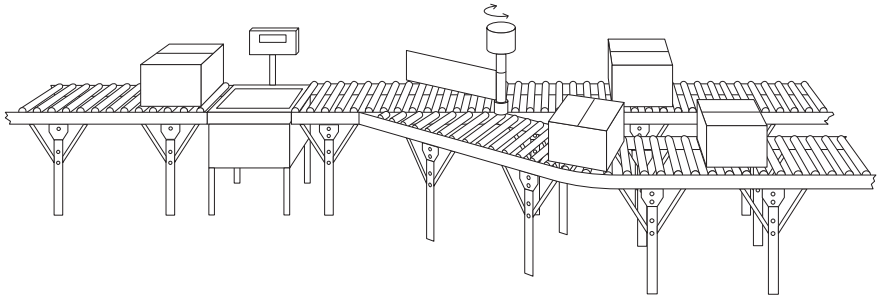
Analog inputs

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.



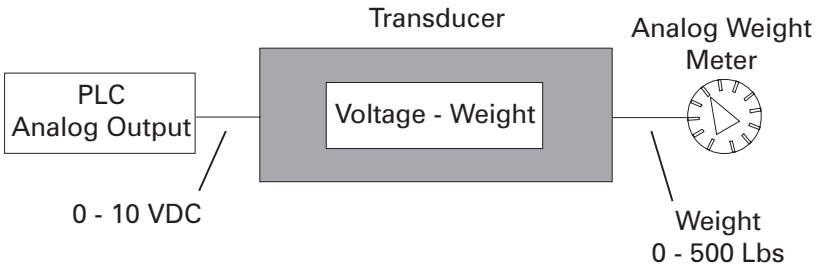
Application example

An example of an analog input can be seen in the following illustration. As packages move along a 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.



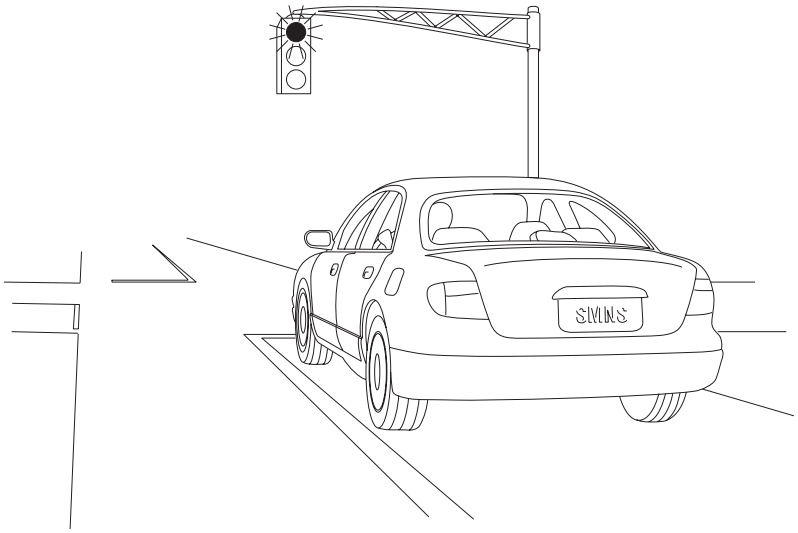
Analog outputs

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.

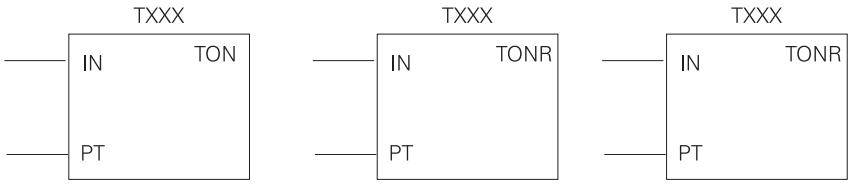


Timers

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).



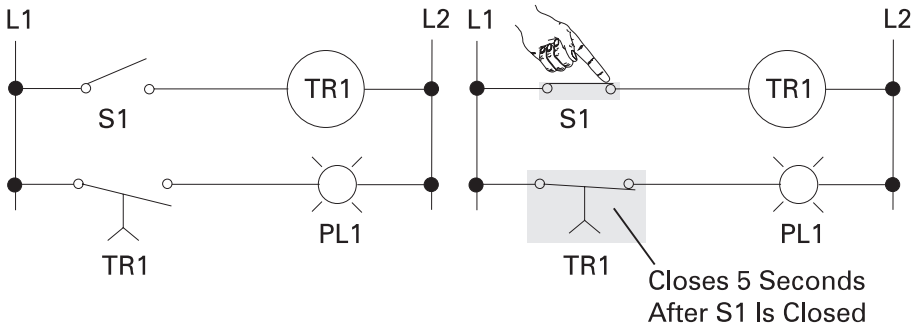
On-Delay Retentive On-Delay Off-Delay

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.

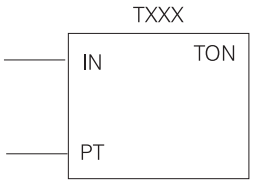
Hard-wired timing circuit

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.

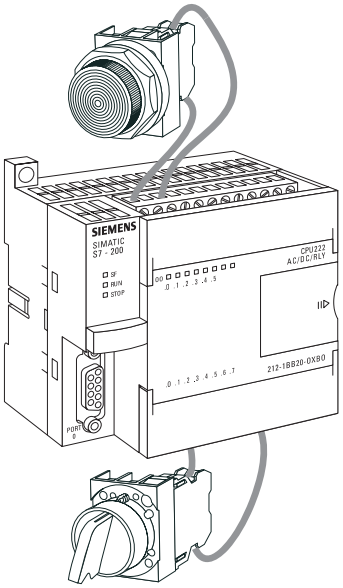


On-Delay (TON)

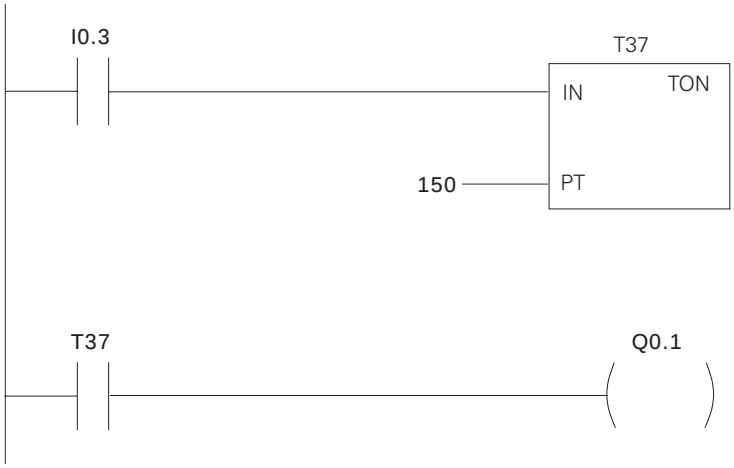
When the On-Delay timer (TON) receives an enable (logic 1) at its input (IN), a predetermined amount of time (preset time - PT) passes before the timer bit (T-bit) turns on. The T-bit is a logic function internal to the timer and is not shown on the symbol. The timer resets to the starting time when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



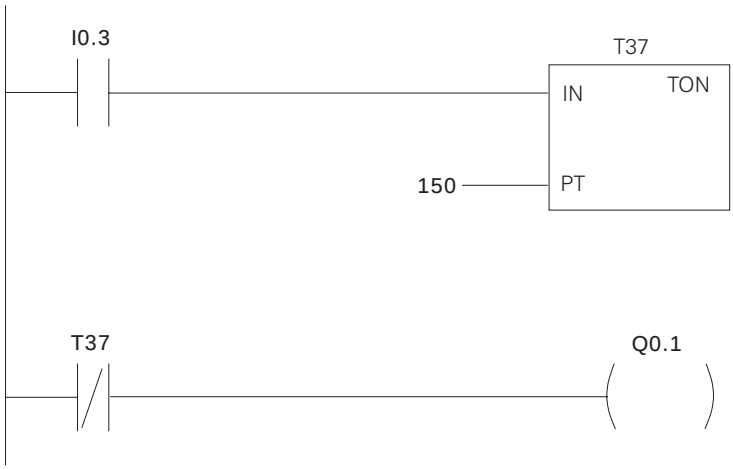
In the following simple timer example, a switch is connected to input I0.3, and a light is connected to output Q0.1.



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.

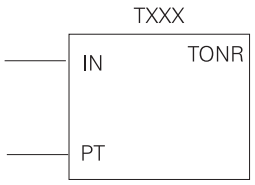


A small sample of the flexibility of PLCs is shown in the following program logic. By reprogramming the T37 contact as a normally closed contact, the function of the circuit is changed to cause the indicator light to turn off only when the timer times out. This function change was accomplished without changing or rewiring I/O devices.

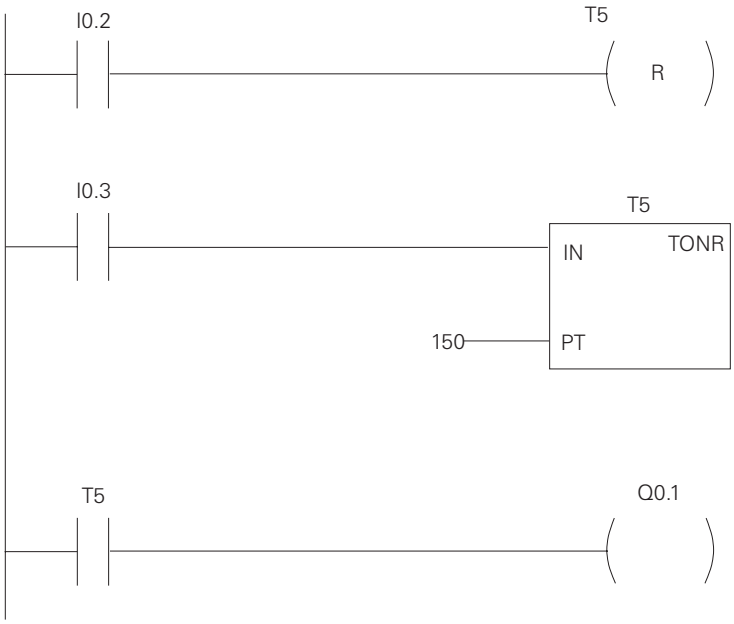


Retentive On-Delay (TONR)

The Retentive On-Delay timer (TONR) functions in a similar manner to the On-Delay timer (TON). There is one difference. The Retentive On-Delay timer times as long as the enabling input is on, but does not reset when the input goes off. The timer must be reset with a RESET (R) instruction.

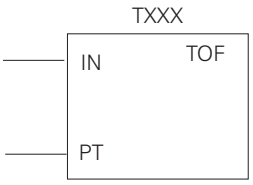


The same example used with the On-Delay timer will be used with the Retentive On-Delay timer. When the switch is closed at input I0.3, timer T5 (Retentive timer) begins timing. If, for example, after 10 seconds input I0.3 is opened the timer stops. When input I0.3 is closed the timer will begin timing at 10 seconds. The light will turn on 5 seconds after input I0.3 has been closed the second time. A RESET (R) instruction can be added. Here a pushbutton is connected to input I0.2. If after 10 seconds input I0.3 were opened, T5 can be reset by momentarily closing input I0.2. T5 will be reset to 0 and begin timing from 0 when input I0.3 is closed again.



Off-Delay (TOF)

The Off-Delay timer is used to delay an output off for a fixed period of time after the input turns off. When the enabling bit turns on the timer bit turns on immediately and the value is set to 0. When the input turns off, the timer counts until the preset time has elapsed before the timer bit turns off.



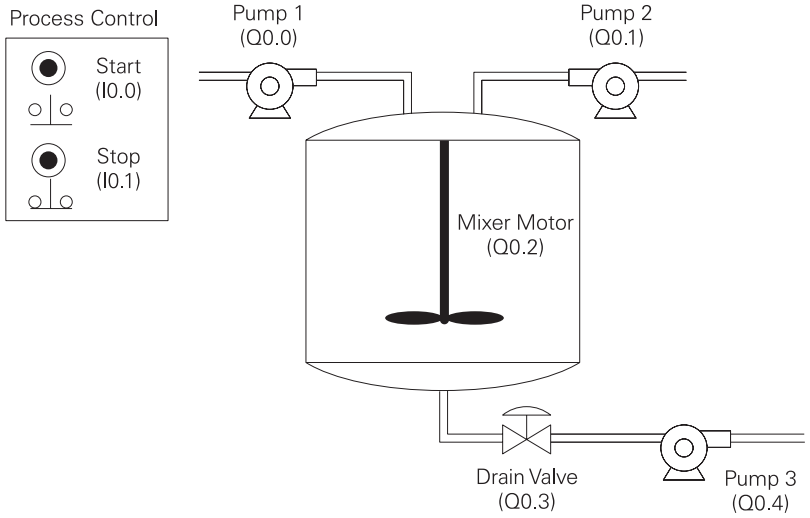
S7-200 timers

The S7-200s have 256 timers. The specific T number chosen for the timer determines its time base and whether it is TON, TONR, or TOF.

Timer Type	Resolution	Maximum Value	Timer Number
TONR	1 ms	32.767 seconds	T0, T64
	10 ms	327.67 seconds	T1-T4, T65-T68
	100 ms	3276.7 seconds	T5-T31, T69-T95
TON, TOF	1 ms	32.767 seconds	T32, T96
	10 ms	327.67 seconds	T33-T36, T97-T100
	100 ms	3276.7 seconds	T37-T63, T101-T255

Timer example

In the following example a tank will be filled with two chemicals, mixed, and then drained. When the Start Button is pressed at input I0.0, the program starts pump 1 controlled by output Q0.0. Pump 1 runs for 5 seconds, filling the tank with the first chemical, then shuts off. The program then starts pump 2, controlled by output Q0.1. Pump 2 runs for 3 seconds filling the tank with the second chemical. After 3 seconds pump 2 shuts off. The program starts the mixer motor, connected to output Q0.2 and mixes the two chemicals for 60 seconds. The program then opens the drain valve controlled by output Q0.3, and starts pump 3 controlled by output Q0.4. Pump 3 shuts off after 8 seconds and the process stops. A manual Stop switch is also provided at input I0.1.



Review 5

1. Analog signals are converted into a _____ format by the PLC.

2. Three types of timers available in the S7-200 are On-Delay, _____ On-Delay, and _____-Delay.

3. The maximum time available on a 100 millisecond time base timer is _____ seconds.

4. A count of 25 on a 10 millisecond time base timer represents a time of _____ milliseconds.

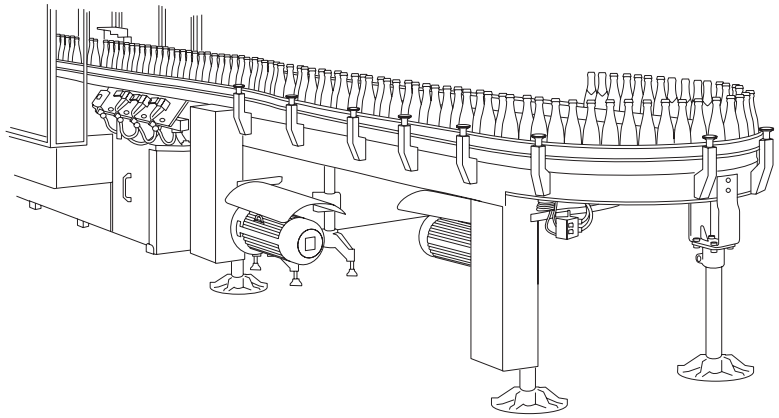
5. There are _____ timers in the S7-200.

Counters

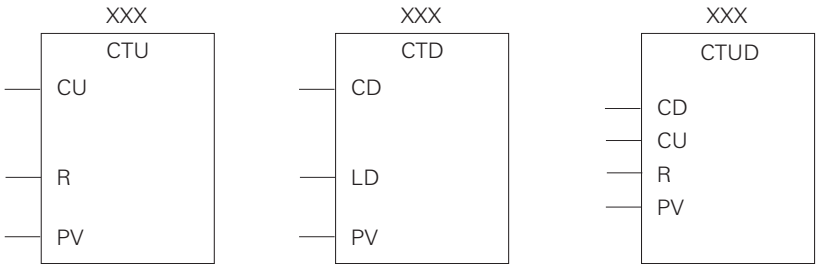
Counters used in PLCs serve the same function as mechanical counters. Counters compare an accumulated value to a preset value to control circuit functions. Control applications that commonly use counters include the following:

- Count to a preset value and cause an event to occur
- Cause an event to occur until the count reaches a preset value

A bottling machine, for example, may use a counter to count bottles into groups of six for packaging.



Counters are represented by boxes in ladder logic. Counters increment/decrement one count each time the input transitions from off (logic 0) to on (logic 1). The counters are reset when a RESET instruction is executed. S7-200 uses three types of counters: up counter (CTU), down counter (CTD), and up/down counter (CTUD).

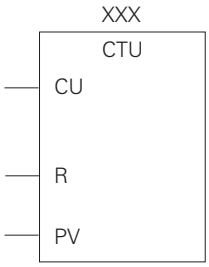


S7-200 counters

There are 256 counters in the S7-200, numbered 0 through 255. The same number cannot be assigned to more than one counter. For example, if an up counter is assigned number 45, a down counter cannot also be assigned number 45. The maximum count value of a counter is ±32,767.

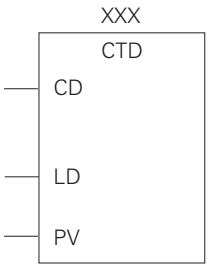
Up counter

The up counter counts up from a current value to a preset value (PV). Input CU is the count input. Each time CU transitions from a logic 0 to a logic 1 the counter increments by a count of 1. Input R is the reset. A preset count value is stored in PV input. If the current count is equal to or greater than the preset value stored in PV, the output bit (Q) turns on (not shown).



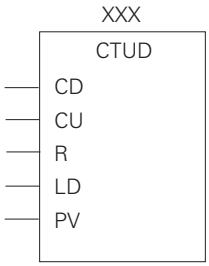
Down counter

The down counter counts down from the preset value (PV) each time CD transitions from a logic 0 to a logic 1. When the current value is equal to zero the counter output bit (Q) turns on (not shown). The counter resets and loads the current value with the preset value (PV) when the load input (LD) is enabled.



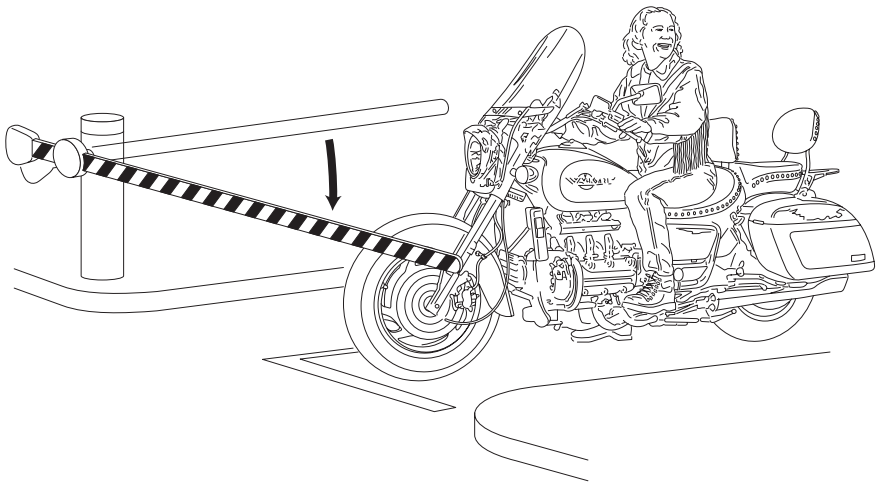
Up/Down counter

The up/down counter counts up or down from the preset value each time either CD or CU transitions from a logic 0 to a logic 1. When the current value is equal to the preset value, the output QU turns on. When the current value (CV) is equal to zero, the output QD turns on. The counter loads the current value (CV) with the preset value (PV) when the load input (LD) is enabled. Similarly, the counter resets and loads the current value (CV) with zero when the reset (R) is enabled. The counter stops counting when it reaches preset or zero.

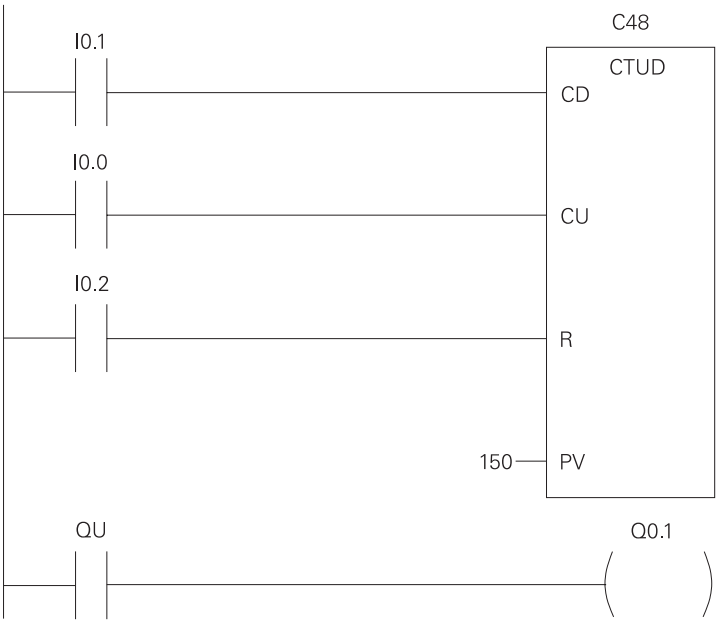


Counter example

A counter might be used to keep track of the number of vehicles in a parking lot. As vehicles enter the lot through an entrance gate, the counter counts up. As vehicles exit the lot through an exit gate, the counter counts down. When the lot is full a sign at the entrance gate turns on indicating the lot is full.

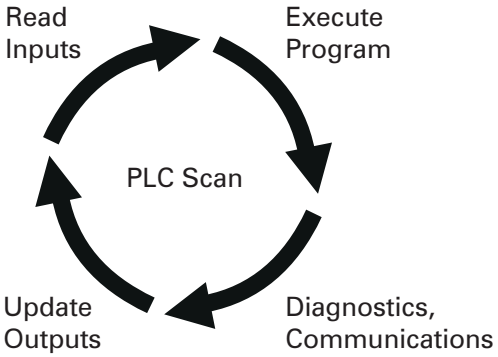


Up/down counter C48 is used in this example. A switch, connected to the entrance gate, has been wired to input I0.0. A switch, connected to the exit gate, has been wired to input I0.1. A reset switch, located at the collection booth, has been wired to input I0.2. The parking lot has 150 parking spaces. This value has been stored in the preset value (PV). The counter output has been directed to output Q0.1. Output 2 is connected to a "Parking Lot Full" sign. As cars enter the lot the entrance gate opens. Input I0.0 transitions from a logic 0 to a logic 1, incrementing the count by one. As cars leave the lot the exit gate opens. Input I0.1 transitions from a logic 0 to a logic 1, decrementing the count by 1. When the count has reached 150 output Q0.1 transitions from a logic 0 to a logic 1. The "Parking Lot Full" sign illuminates. When a car exits, decrementing the count to 149, the sign turns off.



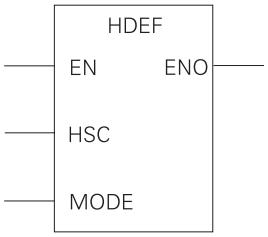
High-Speed Instructions

As discussed earlier, PLCs have a scan time. The scan time depends on the size of the program, the number of I/Os, and the amount of communication required. Events may occur in an application that require a response from the PLC before the scan cycle is complete. For these applications high-speed instructions can be used.

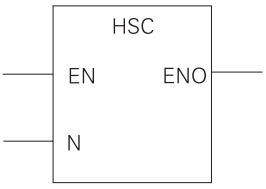


High speed counters

High-speed counters are represented by boxes in ladder logic. The S7-221 and S7-222 supports four high-speed counters (HSC0, HSC3, HSC4, HSC5). The CPU 224 supports six high-speed counters (HSC0, HSC1, HSC2, HSC3, HSC4, HSC5).



High-Speed Counter Definition



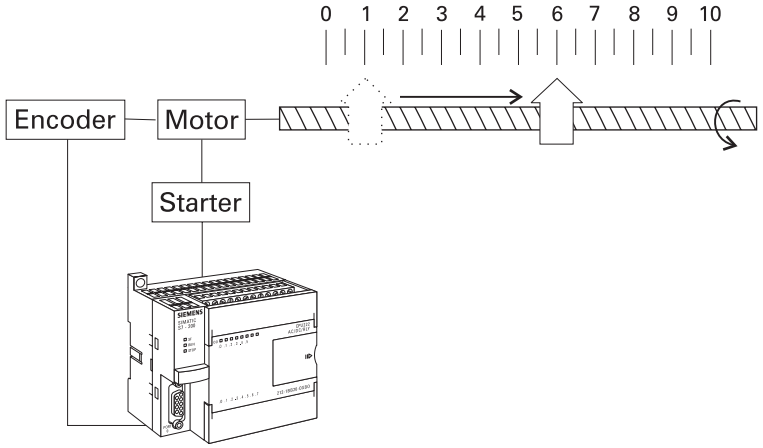
High-Speed Counter

Definition boxes and high-speed counters

The high-speed counter definition boxes are used to assign a mode to the counter. High-speed counters can be defined by the definition box to operate in any of the12 available modes. It should be noted that not all counters can operate in all of the available modes. Refer to the *S7-Programmable Controller System Manual* for definitions available for each counter. Each counter has dedicated inputs for clocks, direction control, reset, and start where these functions are supported. The maximum clock input frequency is 20 KHz. For the two-phase counters, both clocks may be run at 20 KHz. In quadrature mode, 1x or 4x counting rates can be selected. At 1x rate the maximum counting frequency is 20 KHz. At 4x rate the maximum counting frequency is 80 KHz.

Positioning

Positioning is one example of an application that can use high-speed counters. In the following illustration a motor is connected through a starter to a PLC output. The motor shaft is connected to an encoder and a positioning actuator. The encoder emits a series of pulses as the motor turns. In this example the program will move an object from position 1 to position 6. Assume the encoder generates 600 pulses per revolution, and it takes 1000 motor revolutions to move the object from one position to another. To move the object from position 1 to position 6 (5 positions) would take 5000 motor revolutions. The counter would count up 30,000 counts (5000 revolutions x 600 pulses per revolution) and stop the motor.



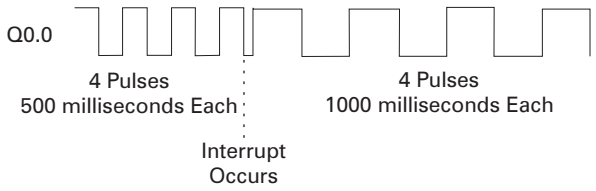
Interrupts

Interrupts are another example of an instruction that must be executed before the PLC has completed the scan cycle. Interrupts in the S7-200 are prioritized in the following order:

1. Communications
2. I/O Interrupts
3. Time-Based Interrupts

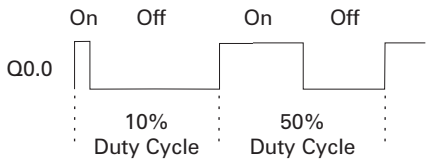
PTO

Pulse train output (PTO) is used to provide a series of pulses to an output device, such as a stepper motor driver. The PTO provides a square wave output for a specified number of pulses and a specified cycle time. The number of pulses can be from 1 to 4,294,967,295 pulses. The cycle time can either be from 250 to 65,535 microseconds or 2 to 65,535 milliseconds. PTOs have a 50% duty cycle. This means the pulse is off for the same amount of time it is on. The number of pulses and the cycle time can be changed with an interrupt. In the following example each pulse is on for 500 ms, and off for 500 ms. After four pulses an interrupt occurs which changes the cycle time to 1000 ms.



PWM

The Pulse Width Modulation (PWM) function provides a fixed cycle time with a variable duty cycle time. The cycle time and pulse width can be specified from 250 to 65,535 microseconds or 2 to 65,535 milliseconds. The pulse width time has a range of 0 to 65,535 microseconds or 0 to 65,535 milliseconds. When the pulse width is equal to the cycle time, the duty cycle is 100% and the output is turned on continuously. In the following example the output has a 10% duty cycle (on 10% off 90%). After an interrupt the cycle switches to a 50% duty cycle (on 50%, off 50%).



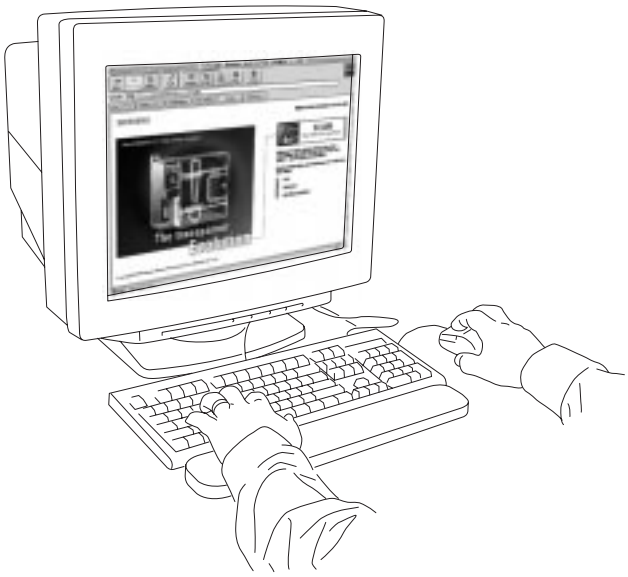
The PWM function can be used to provide a programmable or adjustable control of machine timing. This allows machine operation to be varied to compensate for product variations or mechanical wear.

Transmit

Transmit allows communication with external devices, such as modems, printers, computers, via the serial interface. See the section titled "Connecting External Devices" for examples.

Web site

For more information and sales support on the S7-200 visit our web site at http://www.siemens.com/s7-200.



Review 6

1. The S7-200 supports _____ counters.
2. Three types of counters used in S7-200 are _____ , _____ , and _____ .
3. Counters can count to a maximum of _____ .
4. Events that require an action from the PLC before the scan cycle is complete are controlled by _____ _____ instructions.
5. Depending on the counter, there are up to _____ modes available on high-speed counters.

Review Answers

Review 1

1) a: input module, b: CPU, c: output module, d: programming device, e: operator interface; 2) 2; 3) 16; 4) 1010, 0001 000, A.

Review 2

1) discrete; 2) discrete; 3) CPU; 4) Ladder logic; 5) program; 6) program, data, configuarable parameter; 7) 1024; 8) firmware; 9) e; 10) PC/PPI.

Review 3

1) 221, 222, 224; 2) b; 3) 2, 7; 4) 8, 6; 5) 14, 10; 6) Q0.3; 7) DIN; 8) 50, 72.

Review 4

1) a: box, b: contact, c: coil; 2) load; 3) AND Function - a: 0, b: 0, c: 0, d: 1, Or Function - e: 0, f: 1, g: 1, h: 1; 4) I0.0 or Q0.0, and I0.1.

Review 5

1) digital; 2) retentive, off; 3) 3276.7 seconds; 4) 250; 5) 256.

Review 6

1) 256; 2) CTU, CTD, CTUD; 3) ±32,767; 4) high-speed; 5) 12.

Final Exam

The final exam is intended to be a learning tool. The book may be used during the exam. A tear-out answer sheet is provided. After completing the test, mail the answer sheet in for grading. A grade of 70% or better is passing. Upon successful completion of the test a certificate will be issued.

1. The component of a PLC that makes decisions and executes control instructions based on the input signals is the _____ .

   a. CPU
   b. Input module
   c. Programming device
   d. Operator interface

2. One byte is made up of _____ .

   a. 2 bits
   b. 8 bits
   c. 16 bits
   d. 32 bits

3. The binary equivalent of a decimal 5 is _____ .

   a. 11
   b. 100
   c. 101
   d. 111

4. An input that is either On or Off is a/an _____ input.

   a. analog
   b. discrete
   c. high-speed
   d. normally open

5. A programming language that uses symbols resembling elements used in hard-wired control line diagrams is referred to as a _____ .

   a. ladder logic diagram
   b. statement list
   c. network
   d. PLC scan

6. A type of memory that can be read from but not written to is _____ .

   a. RAM b. ROM
   c. firmware d. K memory

7. Which type of interface cable is needed when a personal computer is used as a programming device for an S7-200 PLC?

   a. PC/PPI b. parallel
   c. serial d. MPI

8. The CPU 224 AC/DC/RELAY has _____ .

   a. 8 DC inputs and 10 relay outputs
   b. 8 AC inputs and 6 relay outputs
   c. 14 DC inputs and 14 relay outputs
   d. 14 DC inputs and 10 relay outputs

9. _____ expansion modules can be used on the S7-224.

   a. none b. 7
   c. 10 d. 30

10. The S7-222 has the ability to store _____ words in user data.

    a. 4096 b. 2,048
    c. 1,024 d. 8,192

11. A new logic operation begins with a _____ instruction.

    a. A b. NOP
    c. LD d. =

12. The address designation for output four of an S7-200 is _____ .

    a. I0.4 b. I0.3
    c. Q0.3 d. Q0.4

13. The maximum clock input frequency for high-speed counters is _____ .

    a. 0.37 µs b. 10 KHz
    c. 20 KHz d. 1 ms

14. The maximum value of an S7-200 timer with a resolution of 1 millisecond is _____ seconds.

a. 3.2767 b. 32.767
c. 327.67 d. 3276.7

15. An S7-200 timer with a time base of 100 ms can count to a maximum value of _____ seconds.

a. 3.2767 b. 32.767
c. 327.67 d. 3276.7

16. The time base of TON 32 of is _____ ms.

a. .1 b. 10
c. 1 d. 100

17. The maximum count of an S7-200 up counter is _____ .

a. 32,767 b. 65,534
c. 98,301 d. 1,000,000

18. A/An _____ is used to assign a mode to a high-speed counter.

a. toggle switch b. interrupt
c. PLC scan d. definition box

19. _____ instructions allows communication with external devices, such as modems, printers, and computers.

a. Transmit b. Interrupt
c. High-speed counters d. High-speed outputs

20. _____ is used to temporarily override the input or output status in order to test and debug the program.

a. Transmit b. Forcing
c. Interrupt d. PLC scan