reluctance of the end user to learn a new language and the advent of the microprocessor gave the industry what is now known as the **programmable logic controller (PLC)**. The first programmable logic controller was invented in 1969 by Richard (Dick) E. Morley, who was the founder of the Modicon Corporation.

Internally there are still AND gates, OR gates, and so forth in the processor, but the design engineers have preprogrammed the PLC so that programs can be entered using RELAY LADDER LOGIC. While RELAY LADDER LOGIC may not have the mystique of other computer languages such as FORTRAN and COBOL, it is a high-level, real-world, graphic language that is understood by most electricians.

The National Electrical Manufacturing Association (NEMA) defines a programmable controller as follows:

> A programmable controller is a digital electronic apparatus with a programmable memory for storing instructions to implement specific functions, such as logic, sequencing, timing, counting, and arithmetic to control machines and processes.

What does a PLC consist of, and how is it different from a computer control system? The PLC consists of a programming device (keyboard), processor unit, power supply, and an input/output **interface** such as the computer system illustrated in Figure 1–1. And while there are similarities, there are also some major differences.

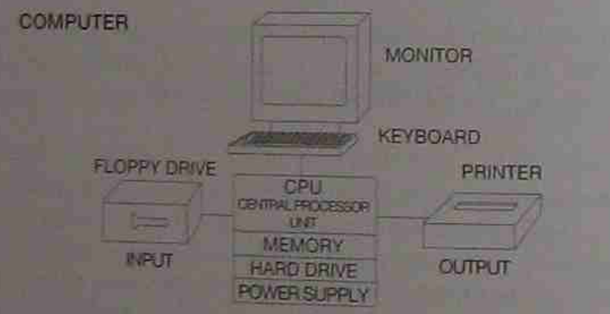


Figure 1–1 Comparison of a Computer System and a PLC

Note: *An interface occurs when two systems come together and interact, or communicate. In the case of the PLC, the communication or interaction is between the inputs (limit switches, push buttons, sensors, and the like), outputs (coils, solenoids, lights, and so forth), and the processor. This interface happens when any input voltage (AC or DC) or current signal is changed to a low-voltage DC signal that the processor uses internally for the decision-making process.*

PLCs are designed to be operated by plant engineers and maintenance personnel with limited knowledge of computers. Like the computer, which has an internal memory for its operation and storage of a program, the PLC also has memory for storing the user program, or LOGIC, as well as a memory for controlling the operation of a process machine or driven equipment. But unlike the computer, the PLC is programmed in RELAY LADDER LOGIC, *not* one of the computer languages. It should be stated, however, that some PLCs will use a form of Boolean Algebra to enter the RELAY LADDER LOGIC. A brief description of Boolean Algebra will be covered in Chapter 18.

The PLC is also designed to operate in the industrial environment with wide ranges of ambient temperature, vibration, and humidity, and is not usually affected by the electrical noise that is inherent in most industrial locations.

Note: *Electrical noise is discussed in Chapter 2.*

Maybe one of the biggest, or at least most significant, differences between the PLC and a computer is that PLCs have been designed for installation and maintenance by plant electricians who are not required to be highly-skilled electronics technicians. Troubleshooting is simplified in most PLCs because they include fault indicators, blown-fuse indicators, input and output status indicators, and written fault information that can be displayed on the programmer.

Although the PLC and the personal computer are different in many ways, the personal computer is often used for programming and monitoring the PLC. Using personal computers in conjunction with PLCs will be discussed in later chapters.

A typical PLC can be divided into four components. These components consist of the **processor unit**, **power supply**, the **input/output section** (interface), and the **programming device**.

The processor unit houses the processor which is the decision-maker, or "brain" of the system. The brain is a microprocessor-based system that replaces control relays, counters, timers, sequencers, and so forth, and is designed so that the user can enter the desired program in RELAY LADDER LOGIC. The processor then makes all the decisions necessary to carry out the user program, based on the status of the inputs and outputs for control of a machine or process. It can also perform arithmetic functions, data manipulation, and communications between the local input/output section, remotely located I/O sections, and/or other networked PLC systems. Figure 1–2 shows an Allen-Bradley PLC-5/40 processor unit.

Note: *Some manufacturers refer to the processor as a* **CPU** *or central processing unit.*

Figure 1–2 Allen-Bradley PLC-5/40 Processor Unit
(Courtesy of Allen-Bradley)

The power supply is necessary to convert 120 or 240 volts AC voltages to the low voltage DC required for the logic circuits of the processor, and for the internal power required for the I/O modules. The power supply can be a separate unit as shown in Figure 1–3 or one of modular design that plugs into the processor rack as shown in Figure 1–4, or one, depending on the manufacturer, that is an integral part of the processor.

Note: *The power supply does not supply power for the actual input or output devices themselves; it only provides the power needed for the internal circuitry of the input and output modules. DC power for the input and output devices, if required, must be provided from a separate source.*

Figure 1–3 Allen-Bradley Separate Power Supply
(Courtesy of Allen-Bradley)

Figure 1–4 Modular (Plug-in) Power Supply
(Courtesy of GE Fanuc Automation)

The power supply can be broken down into four basic parts as shown in Figure 1–5. The first block, or section, of the power supply consists of a step-down transformer. The step-down transformer reduces the voltage level of the incoming AC power. Many power supplies use a step-down transformer that is also a constant voltage transformer. A constant voltage transformer maintains a constant output voltage, even if the incoming power is fluctuating. The second portion of the

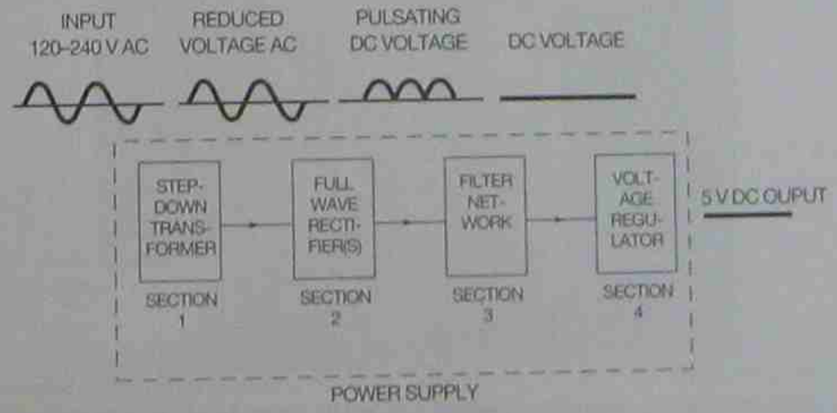


Figure 1–5 Block Diagram of a Typical Power Supply

power supply is the rectifier section, and contains the full wave bridge rectifier(s) to convert the AC sine wave from the secondary of the transformer to a pulsating DC voltage (shown by the wave form in Figure 1–5). The pulsating DC voltage must be further conditioned before it can be used by the processor and I/O modules. The third section of the power supply, the filter section, uses filter devices and/or networks to filter and smooth the DC voltage coming from the rectifier section. The final section of the power supply consists of a voltage regulator. The regulator's function is to maintain a constant DC output voltage, even if the incoming AC voltage fluctuates or varies due to load changes or line disturbances.

The size or amperage rating of the power supply is based on the size, number, and type of I/O modules that are to be used. Power supplies are normally available with output current ratings of 3–20 amps.

Note: *Consider future needs and the possibility of expansion when initially sizing the power supply. It is cheaper in the longrun to install a larger power supply initially than to try and add additional capacity at a later date.*

The input/output section consists of input modules and output modules. The number of input and output modules necessary is dictated by the requirements of the equipment that is to be controlled by a PLC. Figure 1–6 shows an input/output section. Modules are "plugged-in" or added as required.

Input and output modules, referred to as the I/O (I for input and O for output) are where the real-world devices are connected. The real-world input (I) devices can be push buttons, limit switches,

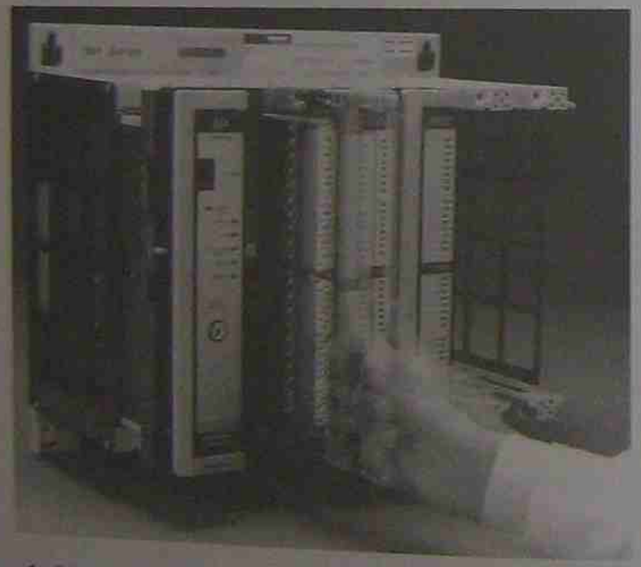


Figure 1–6 Inserting a 32-Point Input Module into a Modicon I/O Rack
(Courtesy of Modicon Inc.)

analog sensors, thumbwheels, selector switches, etc., while the real-world output devices (O) can be hard-wired to motor starter coils, solenoid valves, indicator lights, positioning valves, and the like. The term *real world* is used to separate actual devices that exist and must be physically wired as compared to the internal functions of the PLC system that duplicate the function of relays, timers, counters, and so on, even though none physically exists. This may seem a bit strange and hard to understand at this point, but the distinction between what the processor can do internally—which eliminates the need for all the previously-used control relays, timers, counters, and so forth—will be graphically shown and readily understandable later in the text.

Real-world input and output devices are of two types: discrete and analog. *Discrete* I/O devices are either ON or OFF, *open* or *closed*, while *analog* devices have an infinite number of possible values. Examples of analog input devices are temperature probes, and pressure indicators. The input from an analog input device (varying voltage or current) is converted by way of an Analog-to-Digital Converter (ADC). The conversion value is proportional to the analog input signal and is stored in memory for later use or comparison. Limit switches, push buttons, and the like, are examples of *discrete input devices*.

Examples of *discrete output devices* are motor starter coils, solenoids, and indicator lamps. Analog output devices require varying voltage or current levels to control the analog output. The varying value will be accomplished using a Digital-to-Analog Converter (DAC). The analog output normally is isolated from the output logic circuitry by means of optical coupling.

Note: *Optical coupling will be discussed in detail in Chapter 2.*

A reference was made earlier in this chapter to the I/O section as an interface. Although not a common reference, it is an accurate one. The I/O section contains the circuitry necessary to convert input voltages of 120–240 V AC or 0–24 V DC, etc., from discrete input devices to low-level DC voltages (typically 5 V) that the processor uses internally to represent the status or condition (*ON* or *OFF*). The I/O section can also convert 4–20 milliampere (mA) input signals to low-level DC voltages for the processor. Similarly, the output module changes low-level DC signals from the processor to 120–240 V AC or DC voltages required to operate the discrete output devices. This is a brief overview of the I/O section and its function. How input and output devices are wired to I/O modules and more information about the module circuitry itself is covered in Chapter 2.

The programming device is used to enter the desired program or sequence of operation into the PLC memory. The program is entered using RELAY LADDER LOGIC, and it is this program that determines the sequence of operation and ultimate control of the process equipment or driven machinery. The programming device can be any one of three types: hand-held; dedicated; or personal computer. The personal computer, or PC, is the most common programming device.

The hand-held programmer uses either **LED** (light emitting diode) or **LCD** (liquid crystal display). The hand-held programmers are small, lightweight, and convenient to use in the field (Figure 1–7). The small size, however, limits the display capabilities, which in turn limits its effectiveness for reviewing the program or using the programmer for troubleshooting.

Figure 1–7 Hand-Held Programmer
(Courtesy of Modicon Inc.)

Some PLC manufacturers offer a dedicated programmer that, like a standard personal computer, uses a standard video display terminal (VDT) and keyboard for entering and displaying the program. The use of a larger viewing screen allows more of the program to be viewed at one time and makes troubleshooting and memory access much easier. Dedicated programming devices are typically more expensive than a personal computer because they are designed to operate in the industrial environment. Figure 1–8 shows an Allen-Bradley T-60 programming terminal.

A personal computer (PC) can be used to program most of the PLCs on the market today and many operate in the Windows® environment. Some PLCs require only software to communicate with a personal computer, while others require special hardware keys and/or communication cards for them to work successfully as programming devices. Once communications between the personal computer and the PLC have been established, the PC provides all the benefits of the dedicated programmer, plus it provides program storage as well as runs all the various software packages we have come to depend on today such as spreadsheets, word processing, and graphics. Due to the versatility and lower cost—when compared to dedicated programmers—PCs have become the most popular programming device. Figure 1–9 shows a laptop personal computer that, with the appropriate software, is used to control a programmable controller.

Figure 1–8 Programming Terminal
(Courtesy of Allen-Bradley)

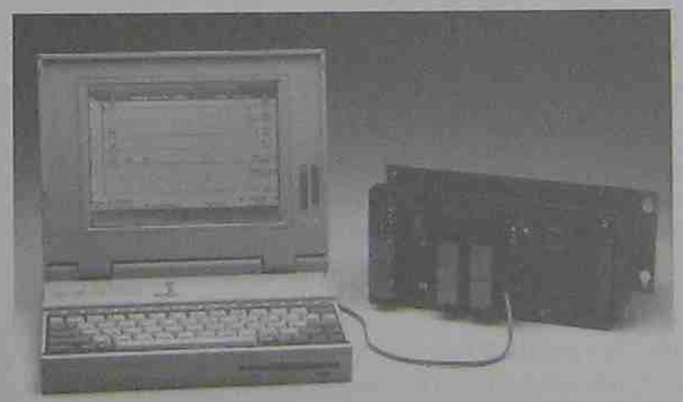


Figure 1–9 Laptop Computer Connected to a Siemans SIMATIC T1305
(Courtesy of Siemens Industrial Automation)

Chapter Summary

Programmable logic controllers (PLCs) have made it possible to precisely control large process machines and driven equipment with less physical wiring and lower installation costs than is required with standard electromechanical relays, pneumatic timers, drum switches, and so on. The programmability allows for fast and easy changes in the RELAY LADDER LOGIC to meet the changing needs of the process or driven equipment without the need for expensive and time-consuming rewiring. By designing the modern programmable logic controller (PLC) to be "technician friendly," the PLC is easier to program and be used by plant engineers and maintenance technicians who have little or no electronic background.

Review Questions

1. List the four main components of a programmable logic controller.
2. Define the term *interface*.
3. Define the term *real world*.
4. Define the term *discrete*.
5. Define the following initials or acronyms:
   LED PLC NEMA
   CPU ADC LCD
   VDT DAC PC
6. Define the term *analog*.
7. List the three types or styles of programming devices.
8. T F RELAY LADDER LOGIC is a high-level graphic computer language.
9. What is the major advantage of a PLC system over the traditional hard-wired control system?
10. Draw a block diagram and label the main components of a typical DC power supply.

CHAPTER

2 Understanding the Input/Output (I/O) Section

Objectives

After completing this chapter, you should have the knowledge to:

- Describe the I/O section of a programmable controller.
- Identify DIP switches.
- Describe how basic AC and DC input and output modules work.
- Define *optical isolation* and describe why it is used.
- Describe the proper wiring connections for input and output devices and their corresponding modules.
- Explain why a hardwired emergency-stop function is desirable.
- Define the term *interposing*.
- Describe what I/O shielding does.
- List environmental concerns when installing PLCs.

I/O SECTION

The input/output section, or I/O section, is the major reason that PLCs are so versatile when used with process machines or driven equipment. The I/O section has the ability to change virtually any type of voltage or current signal into a logic-level signal (typically 5 V DC) that is compatible with the processor. The I/O section automatically makes the conversions necessary for the processor to interpret input signals and to activate output devices, even when the input and output devices are of various voltage and current levels.

A DC input module, for example, can be used with a 12 V DC proximity switch to turn on a 240 V AC motor starter coil that is connected to an AC output module. The conversion and interfacing is all accomplished automatically in the I/O section of the PLC, and it is the ease with which the interfacing is accomplished that has made the PLC such a viable tool in industrial and process control.

The input modules of the I/O section provide the status (*ON* or *OFF*) of push buttons, limit switches, proximity switches, and the like, to the processor so decisions can be made to control the machine or process in the proper sequence. Outputs, such as motor-starter coils, indicator lights, and solenoids are interfaced to the processor through the output section of the I/O. Once a decision has been made by the processor, a signal is sent to the output section to control the flow of current to the output device. In general, the status of the inputs are relayed to the processor and based on the logic of the program that has been written, a decision is made to turn the outputs to *ON* or

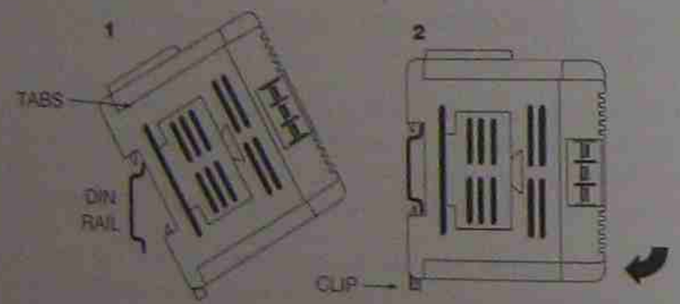
OFF. All of the different types and levels of signals (voltages and currents) used in the control process are interfaced in the I/O section.

The I/O section generally can be divided into two categories: fixed I/O and modular I/O.

Fixed I/O

PLCs with fixed I/O typically come in a complete unit that contains the processor, I/O section, and power supply. The I/O section contains a fixed number of inputs and outputs, all of which have the same voltage level (120 V AC, 24 V DC or 230 V AC). For example, the Modicon Micro shown in Figure 2–1 has 16 inputs and 12 outputs in the self-contained base unit. The base unit measures 10 inches long by 5 inches high and is only 3 inches deep. Like most small PLCs, the Modicon Micro can be DIN-rail or panel mounted. Figure 2–1a shows the DIN–rail mounting instructions for the GE Fanuc Micro PLC.

Figure 2–1 Modicon Micro PLC
(Courtesy of Modicon Inc.)



Position the upper edge of the unit over the DIN rail, so that the rail is behind the tabs as shown above.

Pivot the unit downward (for a unit being mounted right side up) until the spring-loaded clip in the bottom of the unit clicks firmly into place.

Figure 2–1a DIN Rail Mounting
(Courtesy of GE Fanuc Automation)

If more I/O capability is required or different voltages are needed, expansion units with various I/O configuration can be added. Figure 2–2 shows an Allen-Bradley SLC 500 20 fixed I/O controller with an optional two-slot expansion chassis.

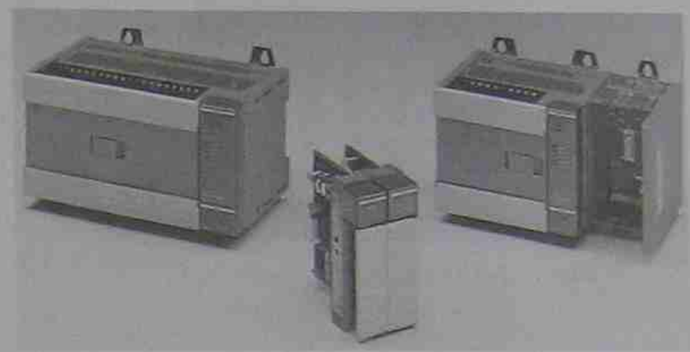


Figure 2–2 SLC 500 Fixed I/O Chassis with Optional Two-Slot Expansion
(Courtesy of Allen-Bradley)

Small PLCs with fixed I/O typically have a discrete input and output section. As discussed in Chapter 1, discrete-type I/O signals are *ON* or *OFF* and do not vary in level. When a 120 V limit switch closes or is *ON*, the signal to the input section will be 120 V and 0 V when the switch is open (*OFF*). Discrete I/O is sometimes referred to as digital I/O, whereas digital signals are either *ON* or *OFF*, and do not vary in magnitude once the signal is established. An analog-type signal, on the other hand, will vary in magnitude and be constantly changing based on control variables in the process. A pressure transducer that sends a 20-milliampere signal at 500 psi or a 10-milliampere signal at 250 psi is an example. Many manufacturers offer analog expansion units that can be added to their fixed I/O PLCs.

While these PLCs are small in size, they are big on features. Most include full-feature instruction sets that include timers, counters, sequencers, shift registers, word moves, data compare, and much more. One should consult the specific dealer for a full list of features.

As the cost of these compact units have decreased, their use has increased. The costs are so competitive that any control processes that use only a small number of relays and/or timers can now be accomplished using a small PLC. The use of a small PLC not only saves money, but also gives added reliability and flexibility.

Because of their shape and size, the term "shoebox" or "brick" is often used by manufacturers and users alike when referring to PLCs with fixed I/O. Figure 2–3 shows the Modicon Micro and the expandable Modicon A120. Note the relative size compared to the ball-point pen.

Figure 2–3 Modicon Micro and A 120 PLCs
(Courtesy of Modicon Inc.)

Modular I/O

Modular I/O, as the name implies, is modular in nature, more flexible than fixed I/O units, and provides added versatility when it comes to the type and number of input and output devices that can be connected to the system. The various types of input and output modules that make up the I/O section are housed, or installed, in an I/O rack or chassis.

The I/O rack or chassis is a framework or housing into which modules are inserted. Figure 2–4a shows three different size racks. Figure 2–4b shows the rack with the I/O modules installed and the processor ready for installation.

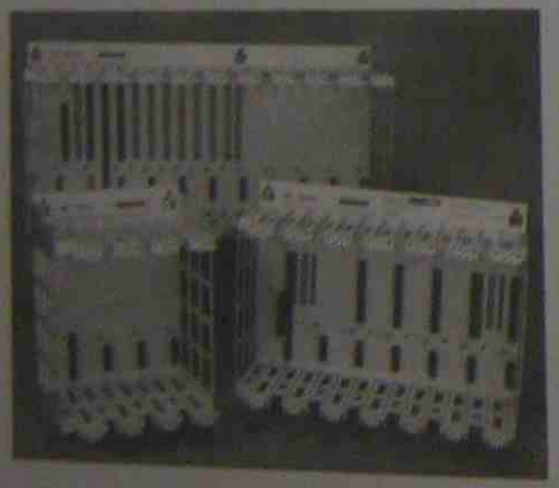


Figure 2–4a I/O Racks
(Courtesy of Modicon Inc.)

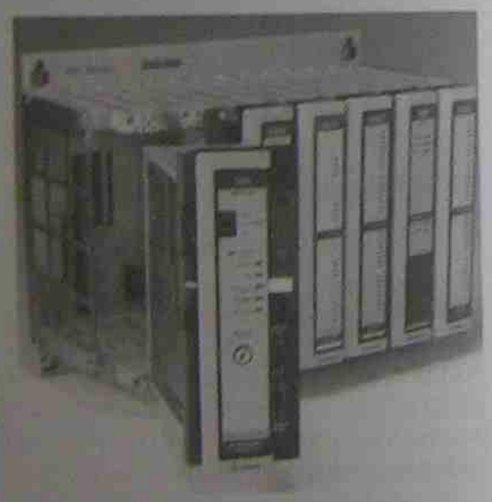


Figure 2–4b I/O Rack with Processor and I/O Modules Installed
(Courtesy of Modicon Inc.)

Racks or chassis come in many shapes and sizes, and typically allow 4, 8, 12 or 16 modules to be inserted. Racks that contain I/O modules and the processor are referred to as *local* I/O. Racks that contain I/O modules, remote I/O communication cards, power supplies, and are mounted separately or away from the processor are referred to as *remote* I/O. An advantage of remote I/O racks is that they can be mounted up to 10,000 feet away from the processor. The number of remote I/O racks that a processor can control varies with each manufacturer. The communication between the remote rack and the processor is accomplished using several different types of communication methods. These methods include coaxial cable, twin axial cable, shielded-twisted pair, or fiber optics. If distance or electrical noise are considerations, the fiber optic communication method may be the best option. Figure 2–5 shows a local rack and three remote I/O racks.

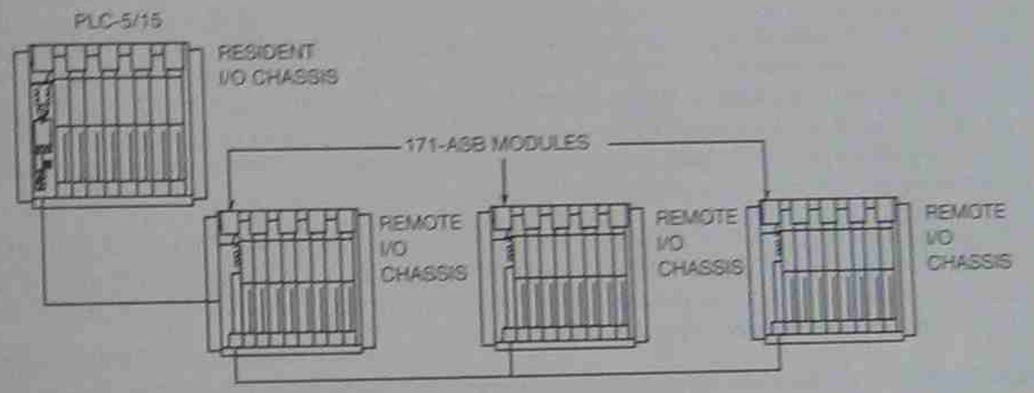


Figure 2–5 Local Rack with Processor and Three Remote I/O Racks
(Courtesy of Allen-Bradley)

Whether local or remote, racks normally have jumpers or switches that have to be set or configured in order for the racks to communicate with the processor. A common switch used for rack configuration is referred to as a DIP switch. DIP is short for Dual-In-Line Package, a common electronic package design, or style, for use on printed circuit boards. These DIP switches are either *ON* or *OFF*, and when set in the proper sequence, are used to assign an address to a rack, such as Rack 1, Rack 2, Rack 3, etc. DIP switches are also used to set fault parameters as well as other processor functions. DIP-switch settings will be specified by the PLC manufacturers and are found in the installation manual.

Note: *Under no circumstance should a pencil be used to change a DIP-switch position. The graphite in the pencil tip can break off, causing the switch to short. Instead, use the tip of a ballpoint pen or other nonconducting pointed object to change switch positions.*

DIP switches are generally mounted on a printed circuit board located in the back of the I/O rack or chassis. This printed circuit board is often referred to as the *backplane*. Figure 2–6 shows a backplane printed circuit board and a DIP-switch group assembly.

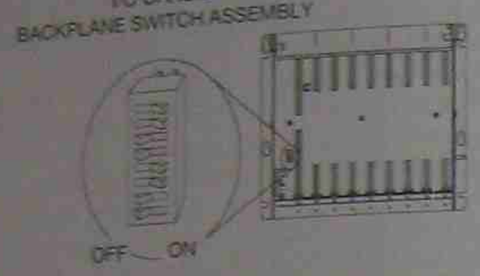


Figure 2–6 Backplane and I/O DIP Switches
(Courtesy of Allen-Bradley)

Within each rack, individual input and output device connections must have a distinct **address** so the processor knows where the device is located, and in return, can send and receive signals, enabling the processor to monitor and/or control the device. Allen-Bradley, for example, uses the rack number, location of a module within a rack, and the terminal number of a module to which an input or output device is connected to determine the device's address. Addresses and addressing of input and output devices will be covered in Chapter 5.

Also mounted on the backplane of the I/O rack are prewired slots or connectors into which the individual I/O modules are inserted. When inserting the modules, proper alignment is assured by card guides (also referred to as printed circuit board guides) which are mounted on the top and bottom of the I/O rack as seen in Figures 2–4a and 2–4b.

Input and output modules can be separated into three basic groups: discrete or digital input/output modules; analog input/output modules; and specialized modules.

DISCRETE I/O MODULES

Discrete I/O modules are types of modules that only accept digital or *ON*- and *OFF*-type signals. These modules only recognize these two states or conditions and that again is *ON* or *OFF*. If a discrete device, such as a limit switch, is connected to this type of module, the module determines the state, or position, of the limit switch, and communicates the state, or status, to the processor. If the limit switch is open (*OFF*), the module indicates to the processor that the limit switch is *OFF*. This *OFF* condition is stored in the processor memory as a zero (0). Had the limit switch been in a closed position, the module would have sent a signal to the processor indicating that the limit switch was *ON*, or closed. The *ON* condition would have been stored in the processor memory as a one (1). All information stored in the processor memory about the status or condition of discrete I/O devices are always in ones and zeros.

Discrete modules are the most common type used in a majority of PLC applications and can be divided into two groups: input and output.

Discrete Input Module

The discrete input module communicates the status of the various real-world input devices connected to the module (*ON* or *OFF*) to the processor.

Note: *The term* real world *is used to indicate that an actual device is involved. As you will learn later in the text, the PLC has the ability to provide timing and counting functions for a machine, even though the timers and counters exist only within the processor, and are not wired into the circuit as with real-world, or actual devices.*

Once the real-world input device is connected, an open or closed electrical circuit exists, depending on the position (open or closed) of the device. The status of the real-world input device is then converted to a logic-level DC electrical signal by the input module and sent to the processor.

Discrete input modules come in a wide range of voltages for various applications. Some of the more common voltage modules are 120 V AC, 240 V AC, 24 V DC, and 12–24 V DC. Some manufacturers give their modules on AC/DC rating to increase their flexibility and reduce required inventory. It is important to note, however, that while the module may be used with either AC or DC input voltages, the voltages *cannot* be intermixed on the same module.

Input modules can be purchased with a wide range of input terminals or points, that determine the number of individual field devices that can be connected to the module. Common sizes, depending on the manufacturer, are 8, 16, and 32 points. Sixteen- and 32-point modules are often referred to as high-density modules since they are physically the same size as an 8-point module. High-density modules usually provide lower cost per point, or input device, but are also more difficult to wire. The increased difficulty in wiring is caused by the closer proximity of the wiring terminals and the increased number of wires in the wiring harness.

AC Discrete Input Modules

Figure 2–7 shows a simplified diagram of one of the input circuits of a typical AC discrete input module. Resistors are used to drop the incoming voltage; then a bridge rectifier is used to convert the AC input voltage to DC. Next, a filtering circuit is used to condition the DC and guard against electrical noise. Electrical noise can cause a short-duration DC pulse that is sometimes interpreted by the processor as a closed signal. This false, or erroneous, signal could be interpreted as a valid signal, and a 1 would be placed in memory to indicate the device was *ON*, even though it was not. To eliminate the possibility of faulty operation due to electrical noise, the filter section of the module delays an actual input signal from being sent to the processor for 15 to 25 milliseconds (msec).

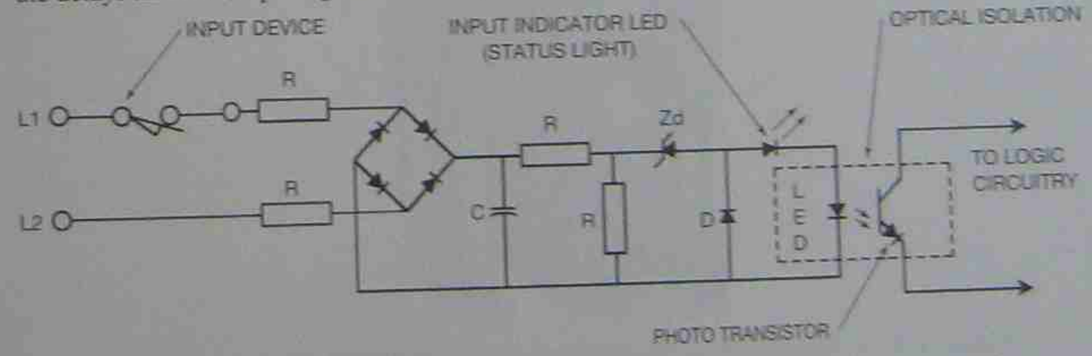


Figure 2–7 Simplified AC Input Module Circuit with Indicator Light

The filter requires that the AC signal be not only of a specific value, but also be present for a specific amount of time before the module views it as a real signal and communicates the results to the processor. A valid signal is relayed through an **optically-coupled** circuit, across the backplane of the I/O rack to the processor.

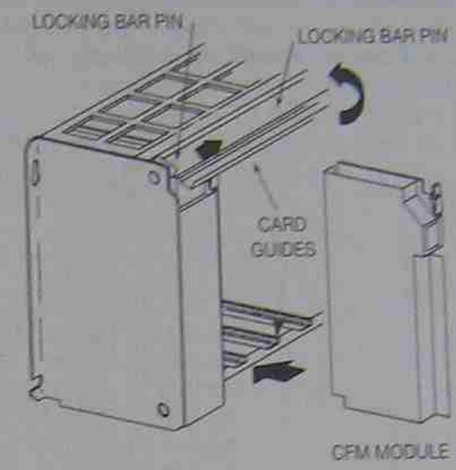
The optically-coupled circuit uses a LED (light emitting diode) to turn *ON*, or forward bias, a photo transistor to complete the electrical circuit to the processor. When the LED is turned *ON* to indicate that the actual input device has closed, the light from the LED is picked up by the photo transistor and that makes the transistor conduct, completing a 5 V DC logic circuit, and the status of the input is communicated to the processor. This form of optical coupling is also referred to as **optical isolation**. By employing optical coupling, or isolation, there is no actual electrical connection between the input device and the processor. This eliminates any possibility of the input line voltage, i.e. 120 or 240 V AC, from coming in contact with and damaging the low-voltage DC section of the processor. Optical isolation also protects the processor from electrical noise, voltage transients, or spikes. In summary, optical isolation prevents any unwanted voltage from the I/O section from reaching the logic section of the processor.

Individual status lights are provided for each device that is connected to an input terminal (Figure 2–7). The status light is lit when the input device is closed and is *OFF* when the input device is *OFF* (open). With the status lights showing the actual position of the various input devices connected to the input module, they make a valuable troubleshooting aid. The electrician or technician need only look at the status lights on the input module to determine the position, or status, of any input device.

A typical I/O module consists of two parts: a printed circuit board and a terminal assembly. The printed circuit board plugs into a slot, or connector, in the I/O rack and contains the solid-state electronic circuits that interface the I/O devices with the processor. The terminal assembly then attaches to the front edge of the printed circuit board, which may or may not have a protective cover, depending on the manufacturer. Figure 2–8, from left to right, shows a typical 8-point 120V input module, a 16-point 120 volt input module, and a 10–30 volt DC output module.

Figure 2–8 AC 8 and 16-Point Input Modules and a 10–30 Volt DC Output Module
(Courtesy of Allen-Bradley)

Figure 2–9 shows how the input module (Figure 2–8b) is installed in the I/O rack.



Swing the chassis locking bar down into place to secure the modules. Make sure the locking pins engage.

Figure 2–9 Installing a Module in an I/O Rack
(Courtesy of Allen-Bradley)

After the input modules have been installed in the I/O rack, they are ready to have one side of each input device connected to their terminals or wiring arms (Figure 2–10).

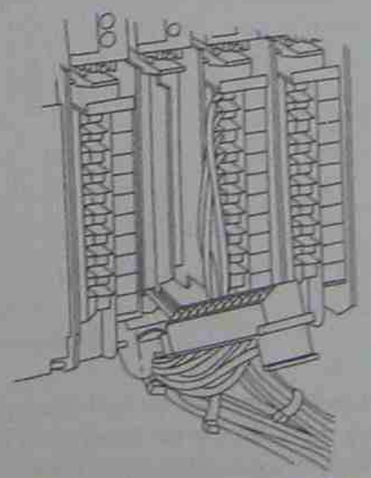


Figure 2–10 I/O Module Field Wiring Arm
(Courtesy of Allen-Bradley)

While each input device has two wires connected, only one wire is connected directly to the input module. The other wire of each input device is connected common to Line 1 (Figure 2–11a). The same connection scheme is used for 8-, 16-, or 32-point input modules. Figure 2–11b shows the wiring connections for an Allen-Bradley 16-point input module.

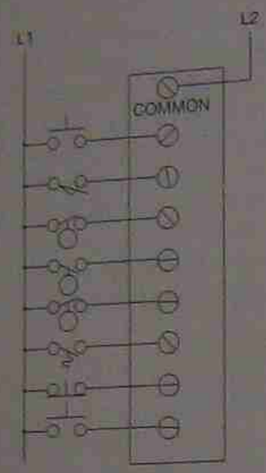


Figure 2–11a Field Wiring for AC Module

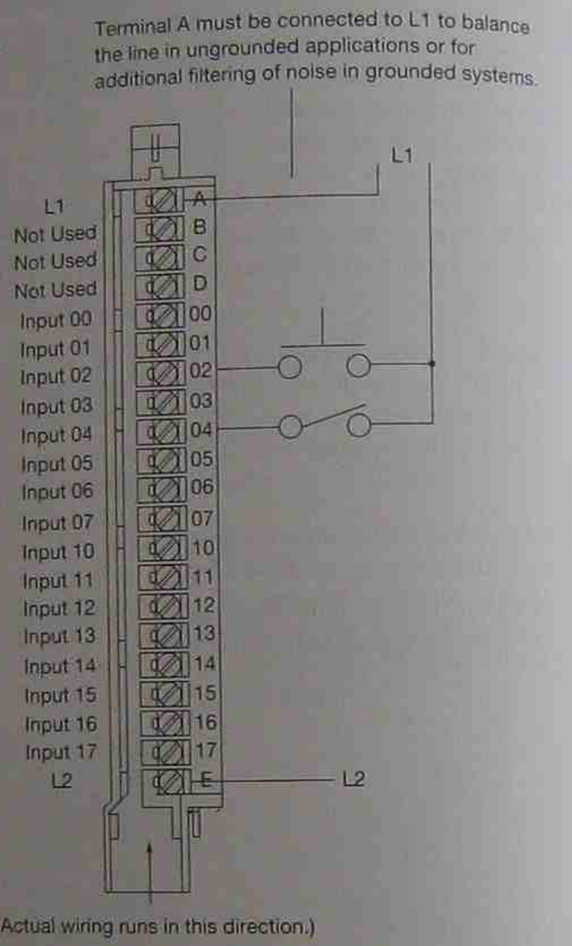


Figure 2–11b Connections for an Allen-Bradley 16-Point Input Module
(Courtesy of Allen-Bradley)

The wires from the individual devices are referred to as field wiring, since the wires are external to the PLC and are connected in the field rather than at the factory. On larger process machines, the field wiring that is brought into the I/O rack consists of hundreds of wires. The basic rule is that one side of each input device is wired to a hot conductor (L1 for AC or + for DC), and the other side of the device is wired to an input terminal on the input module. The input module has a common connection for the

neutral, or grounded potential (L2), for AC modules and the negative (–) for DC modules. Consult the literature that comes with each input module to ensure that the correct wiring connections are made.

Figure 2–12 shows two simplified circuits. In the first, or traditional circuit, the input device (single-pole switch) is connected to, and controls, the light. In the PLC circuit, the input device is connected to the input module instead of the light. The module converts the 120 V AC input signal to 5 V DC, and communicates the status of the single-pole switch to the processor which, in turn, controls a light that is connected to an output module.

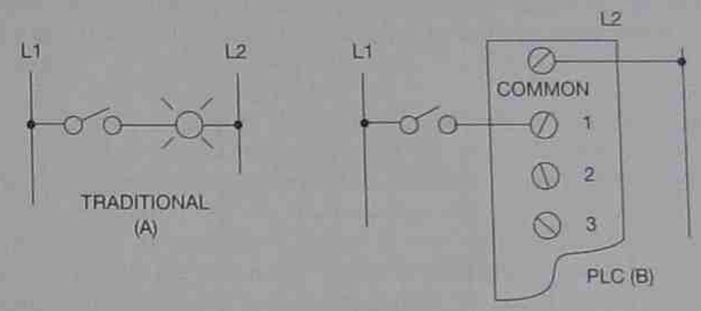


Figure 2–12 Traditional Wiring for Single Pole Switch (A) Compared to PLC Wring (B)

DC Discrete Input Module

Figure 2–13 shows a simplified diagram for a typical DC input module. With the exception of the bridge rectifier used in the AC module, the principles are the same. Resistors are used to lower, or drop, the incoming voltage. Filtering circuits condition the low-voltage signal and add an additional delay in the response time. This period of delay is slightly less than the delay used in the AC input module, but it is also used to verify that the signal received is a valid signal of a proper duration, and not a signal caused by electrical noise, voltage transients, or the like.

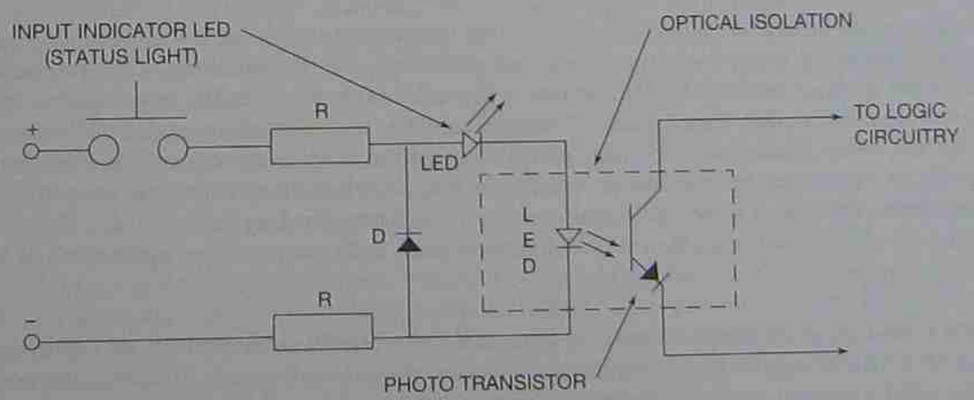


Figure 2–13 Simplified Circuit for DC Input Module with Indicator Light

Optical isolation is also used on the DC input module to isolate the processor from the higher voltage of the input devices. When the LED is turned *ON* by the closing of the input device, the photo transistor conducts and the status of the input device is communicated across the backplane of the I/O rack to the processor. Status lights, shown in the diagram, are also provided for each input device to indicate whether the input device is open or closed. Figure 2–14 shows how a typical DC input module is wired.

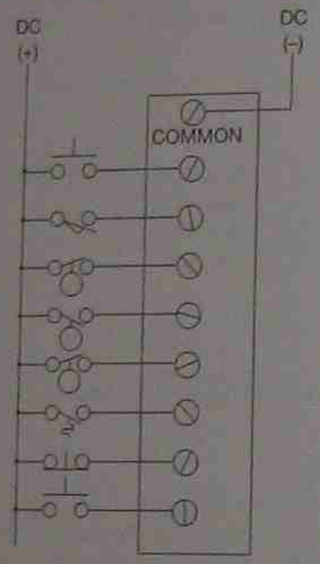


Figure 2–14 Field Wiring for DC Input Module

Fast-Responding DC Input Modules

Fast-responding DC input modules are used when the process requires fast-acting sensors to respond to high-speed and/or high-volume applications. Encoders, other types of sensors that respond with many pulses for each rotation of a shaft, or proximity switches that are counting parts or products produced by high-speed machines are all examples of the benefits of a fast-responding module. The internal operation of the module is the same as the discrete DC input module, with the exception of the delay created in the filtering circuit. The fast-responding module has only a 1-msec or less delay. With this short time, normal mechanical contact devices may not work correctly due to the contact bounce that is inherent in some mechanical switches and contacts. The contact bounce is counted as an additional input signal and processed by the processor. This extra count has an obvious adverse affect on machine operation and is not a proper application of this type of input module.

Discrete input modules come in a variety of types and styles that fit most applications requiring an *ON*- or *OFF*-type of signal. Allen-Bradley, for example, manufactures nearly 20 different types of discrete input modules. Selecting the proper module for any given application is relatively easy when the voltage level, voltage type, current, and response time are known.

Discrete Output Modules

The purpose of a discrete output module is to control the current flow to real-world devices such as motor starter coils, pilot lights, control relays, and solenoid valves. As was true for the discrete input module, the discrete output module also works on a digital, or *ON* and *OFF*, basis. When the processor has made a decision to turn a specific device *ON*, the processor places a 1 in the memory location assigned to that output device, and later in the process, the information is communicated by way of the backplane of the I/O rack to the output module, and the required real-world device will be turned *ON*. Similarly, when the processor determines that a device needs to be turned *OFF*, a 0 is placed in the device's memory location and the device will be turned *OFF*. The output module acts like a remote control switch that is controlled by the processor for turning output devices *ON* and *OFF*.

Output modules are generally classified as AC and DC. Both types offer a wide range of voltages, typical for the input modules discussed earlier. Read the information sheets carefully to determine that the module selected is appropriate for the load (output device) that is to be controlled.

Output modules are sized by the number of output devices that can be connected to them. The number of terminals, or points, for connecting output devices are typically 8, 16, and 32. Output modules have a current rating for each terminal or connection point, as well as an overall rating for the module. The individual terminal rating indicates the maximum current, or load, that can be controlled. This rating is a continuous duty rating. A surge rating is usually provided, indicating the maximum current that can be controlled and the length of time. The time rating may be given in milliseconds or in cycles. A typical current rating for a 120 V AC output module would be: 1.5 amperes maximum continuous duty, with a surge current rating of 4 amperes for 8.3 msec (1/2 cycle).

The surge current rating is necessary for the inrush, or "pull-in," current that is present when motor starter coils, solenoids, and other inductive loads are initially energized. Once the load has been energized, the "hold-in," or "seal-in," current is much less, and the continuous duty rating of the module is sufficient.

Each module also has a total current rating. The total current rating must be determined from the manufacturer's literature, not by simply adding the ampere rating per point. To further clarify this point, consider the rating of one manufacturer's 16-point 120 V AC output module. Each point is rated for 2 amperes continuous duty, yet the maximum current rating for the module is only 8 amperes. Why is the total rating not 2 amperes times 16 points, or 32 amperes? The answer has to do with the way the module dissipates the heat generated by the current flow in the module. Normally, no fans or other external methods of cooling are used, and the heat that is generated within the module is dissipated using heat sinks. Heat sinks work on convection alone, and this limits the amount of heat that they can effectively dissipate. The total current rating that a module is given, therefore, is determined by the ability of the module to dissipate heat. Thus, the lower current rating of the 16-point 120 V AC module shows that the module can only satisfactorily dissipate the heat from 8 amperes of continuous current flow, not the full 32 amperes that would be possible if all loads were operating at the maximum 2 amperes rating. In reality, however, how likely is it that all 16 loads connected to the module would be on and operating at their full 2-amperes capacity?

Subjecting the module to higher-than-rated current loads creates excess heat in the module. This excess heat has a detrimental affect on the electronic components and leads to shortened operating life and/or component failure. The ambient temperature that the I/O operates in is another factor that must be considered. PLCs are normally designed to operate in a temperature range of 32°–140°F or 0°–60°C. Operating in temperatures above these limits affects the module's ability to dissipate heat and can lead to early component failure.

Another PLC manufacturer rates their 16-point, 120 V AC output module at .5 amperes per point, but starts to derate the current level when more that 50% of the points are *ON*, and/or when the temperature exceeds 40°C. With each manufacturer rating their output modules differently, it is necessary to carefully review the literature when specifying modules.

AC Output Module

The internal circuitry for 1 point of a typical AC output module is shown in Figure 2–15. The AC output module usually consists of a **Triac** (shown in the figure). However, some manufacturers use a **Silicon-Controlled Rectifier (SCR)** instead of a Triac. When the processor determines that the output is to be turned *ON*, a signal is sent across the backplane of the I/O rack and the LED (light emitting diode) is turned *ON*. The light from the LED causes the photo transistor to conduct, which provides current for the gate of the Triac. This portion of the output module optically isolates the logic section of the processor from the line voltage of the output devices.

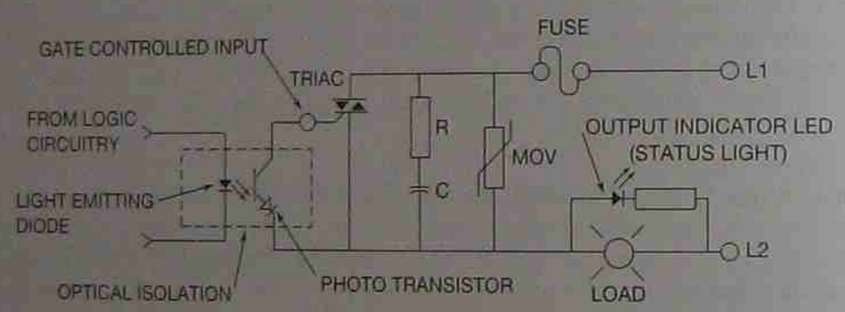


Figure 2–15 Typical AC Output Module Circuit

The Triac is used as an electronic switch to turn output devices *ON* and *OFF*. The Triac itself is the equivalent of two silicon-controlled rectifiers in inverse parallel connection with a common gate. The gate controls the switching state (*ON* or *OFF*) of the device. Once a signal is applied and the break-over voltage point is reached on the gate (normally 1 to 3 V), the Triac freely conducts in either direction, completing the path for current flow to the output device.

A Triac is a solid block of crystalline material and is more sensitive to applied voltages and currents than standard relay contacts. Triacs are also limited to a maximum peak applied voltage, and, if this value is exceeded, a "dielectric-type" breakdown can result, causing a permanent short-circuit condition. A snubber circuit that consists of a resistor in series with a capacitor and a metal oxide varistor is used to protect the Triac from damage from electrical noise and voltage spikes. The resistor and capacitor form an RC circuit that is used to control the rate at which voltage

builds up in the circuit. If the voltage rises too fast, the capacitor absorbs, and the resistor dissipates, excess voltage. The metal oxide varistor (MOV) is designed to break down, or conduct, at certain voltage levels. In a 120 V AC circuit, the peak voltage is approximately 170 V. The MOV would typically be set to break down at 190 V. When the MOV conducts, it clips the excess voltage and thus prevents damage to the module.

Triacs constructed of a solid "block of material" have some characteristics that are not found with standard relay contacts. The Triac, rather than having *ON* and *OFF* states, actually has low- and high-resistance levels, respectively. In its *OFF* state (high resistance), a small leakage current still flows through the Triac. This leakage current, which is usually only a few milliamperes or less, normally causes no problem. When low-resistive pilot lights are connected to AC output modules, a faint glow of the filament may be detectable, even when the module is *OFF*. Similarly, the coils of some small control relays or solenoids may produce a detectable hum due to the Triac leakage current, even though the Triac is technically *OFF*. This small leakage current also causes false readings in some digital and analog meters. Troubleshooting techniques for triacs are covered in Chapter 20.

While Triacs are capable of carrying surge currents higher than their continuous current ratings, such surges must be of short duration (1/2 to 1 cycle) and not repetitive. Exceeding the manufacturer's listed surge values or the maximum continuous current rating, usually referred to as maximum RMS on-state current, results in a permanent short circuit.

After an output module is installed in the I/O rack, the actual real-world output devices are connected. Figure 2–16 shows the proper termination for an 8-circuit AC output module. Each output device has two wires; one wire from each output device is wired to the L2 potential. The other wire from each device is wired to one terminal of the output module, as shown in the figure. L1 is then connected to a common terminal on the module to supply the other potential for the output devices.

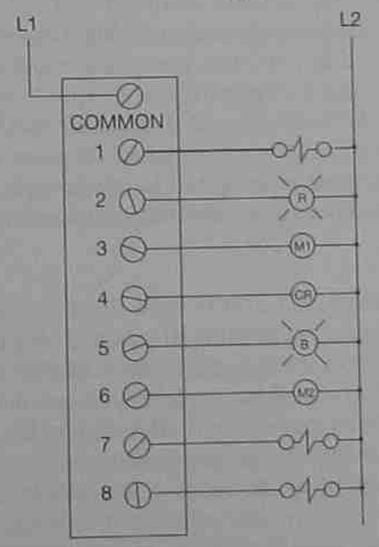


Figure 2–16 Field Wiring for AC Output Module

Figure 2–17 shows two simplified circuits, each including an output device. In the first circuit, the output is wired to the L2 potential and the single-pole switch is used to control the L1 potential to complete the circuit. In the PLC circuit, the output is again wired to L2, but the switch is replaced by the output module which is wired to the L1 potential. Simply stated, the output module may be viewed as an electronic switch that is controlled by the PLC's processor.

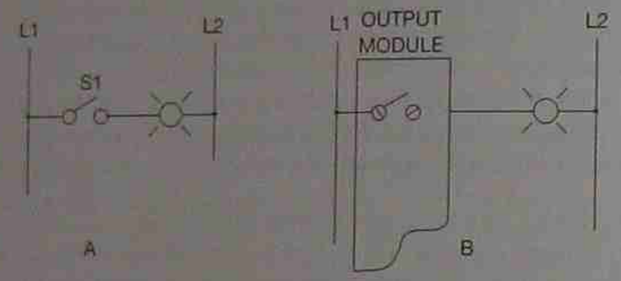


Figure 2–17 Traditional Wiring for an Output Device (A), Compared to PLC Wiring (B)

Output Fuses

In order to prevent damage to output modules, it is important not to exceed the current rating or to exceed its inrush capability. It is also important that output modules be protected from short circuits and ground faults. To provide protection for overcurrent, short circuits and ground faults, output modules are always fused. The number of fuses used vary with each manufacturer. Some modules have an individual fuse for each output terminal, or point, others have one fuse for each 8 outputs, while still others use one fuse to protect all 16 points on their output module. Some PLCs, like the Allen-Bradley SLC 500, do not come with internal fuses, and fuses must be added to protect the outputs.

Most PLCs come with "blown-fuse" indicators to show that a fuse has blown. Some modules have a "blown-fuse" indicator for each output terminal. If a fuse is blown on output terminal 6, an indicator lamp at terminal 6 will be lit to indicate the location of the blown fuse. Other modules have only one "blown-fuse" indicator for the whole module. This one indicator lamp only indicates that a fuse has blown; it does not indicate its location, and it is up to the electrician or technician to determine which fuse (or fuses) is faulty. When individual indicators are used for each output terminal, troubleshooting is greatly simplified. When only one indicator is used for the entire module, however, a blown-fuse indicator merely tells you that a fuse has blown, but does not indicate which one. Figure 2–18 shows an Allen-Bradley AC output module with the blown-fuse indicator light identified.

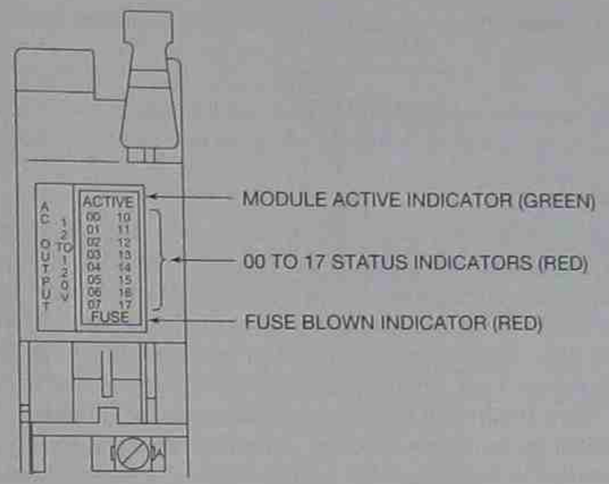


Figure 2–18 AC Output Module with Blown-Fuse Indicator Light
(Courtesy of Allen-Bradley)

As you may expect, access to the fuses for changing blown fuses varies with the manufacturer. Some modules must be removed from the I/O rack and a cover removed before the fuses can be changed, while other modules provide direct access to the fuses on the front edge of the module. For modules that must be removed to change fuses, electricians or technicians must be sure to turn the power *OFF* before removing or reinserting the module.

It is important that when a blown fuse is removed, its replacement fuse be of the same voltage rating, amperage rating, interrupting rating, response time, and physical size, as recommended by the manufacturer.

To speed troubleshooting and fuse replacement, many plants now add additional fuses to each output circuit. Terminal blocks are available that have built-in fuse holders and individual blown-fuse indicators. A separate fuse can then be wired in series with each output device, external to the output module. The fused terminal blocks are then mounted in a convenient location for easy access and troubleshooting.

Status Lights

Status lights are provided for each output point to indicate when that point has been turned *ON*. For troubleshooting purposes, it is important to understand how these status lights are wired. If the power for the lights comes from the processor side of the module, an illuminated light indicates that the processor has sent a signal to turn *ON* the output attached to *that* particular point. It is *not* an indication that current is flowing to the output device. If the status light is powered from the actual output power, then it is an indication that current is flowing to the output device.

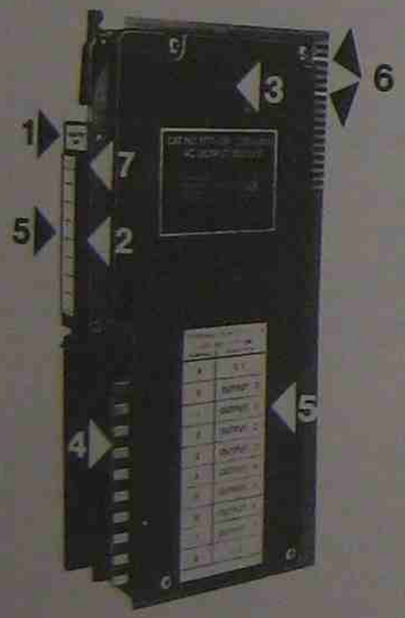
Some modules have two status lights. One is referred to as the logic light, and the other as the output light. When the logic lamp is lit, it indicates that the logic to turn on the output device has

been sent from the processor. When the output light is lit, it indicates that there is a path for current flow to the output device.

The status lights are a powerful troubleshooting tool, but it is important to understand exactly what the status lights indicate, and not to read more information into them than they are able to provide.

Module Keying

Figure 2–19 shows an Allen-Bradley 240 V AC output module. This AC output module looks just like the AC or DC input module discussed earlier (Figure 2–8). This is not only true for Allen-Bradley modules, but also for other manufacturer's products as well. In all instances, the manufacturer's label and/or color codes the fronts of all modules to distinguish between the different types (AC input, DC output, TTL, Analog, and so on). Most manufacturers have designed each module so it can be **keyed**. In Figure 2–19, item 6, the module has been notched in two places. Installing keying bands on the Allen-Bradley I/O rack backplane connector (Figure 2–20) where a specific module is to be installed prevents any module, other than the type for which the connector is keyed, from being installed in that connector or slot. Figure 2–21 shows a close-up view of what the keying band looks like and how it is installed. Each type of module has a unique combination of notches. This feature prevents inadvertent or accidental replacement of the wrong type of module, say an input module, into a slot that is already wired to output devices. To prevent damage and downtime, it is important that the keying system be used.



1. Orange identification label
2. Status indicators
3. Protective cover
4. Field Wiring Arm connects here
5. Labels identify user outputs
6. Slotted for I/O slot insertion only
7. Blown fuse indicator

Figure 2–19 Notched AC Output Module
(Courtesy of Allen-Bradley)

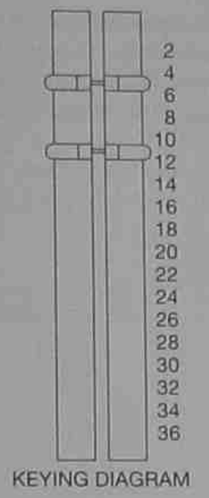


Figure 2–20 Backplane Connector Keying Diagram
(Courtesy of Allen-Bradley)

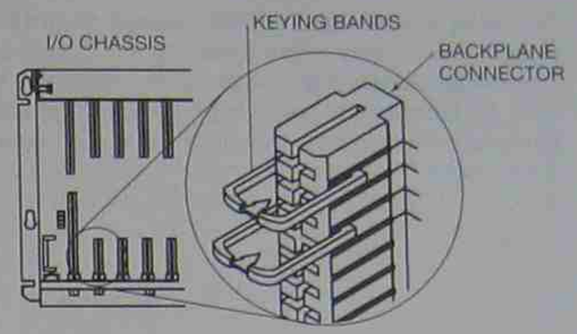


Figure 2–21 Connector Keying Pin
(Courtesy of Allen-Bradley)

There are some manufacturers that do not use the physical keying system, but rely on software configuration to prevent inadvertent or accidental replacement of a wrong module. Each module is given a unique "signature" that is relayed to the processor. If the signature does not match the software configuration, a fault will occur and prevent the PLC from operating.

DC Output Modules

DC output modules are basically the same in operation as AC output modules. The difference is the use of a power transistor instead of a Triac for the control of output current. The power transistor has a quicker switching capability than the Triac; therefore, the response time for DC modules is faster than for AC modules. The RC circuit and MOV used in the AC output module is replaced with a diode to provide protection from electrical noise and spikes. A typical DC output circuit is shown in Figure 2–22.

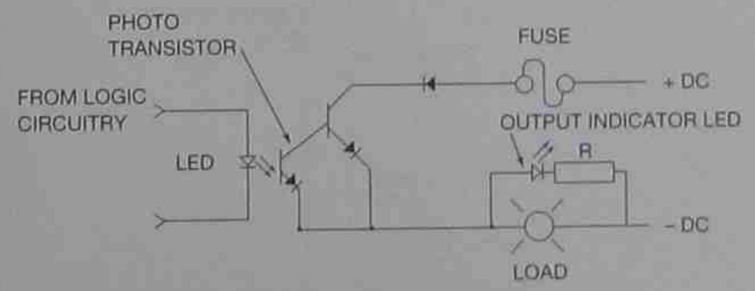


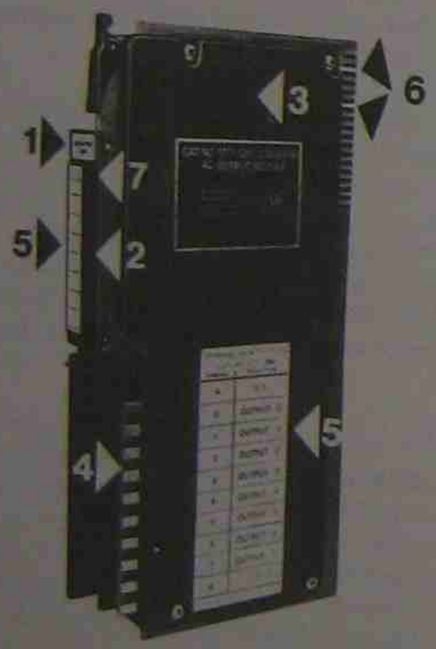
Figure 2–22 Simplified DC Output Module Circuit

been sent from the processor. When the output light is lit, it indicates that there is a path for current flow to the output device.

The status lights are a powerful troubleshooting tool, but it is important to understand exactly what the status lights indicate, and not to read more information into them than they are able to provide.

Module Keying

Figure 2–19 shows an Allen-Bradley 240 V AC output module. This AC output module looks just like the AC or DC input module discussed earlier (Figure 2–8). This is not only true for Allen-Bradley modules, but also for other manufacturer's products as well. In all instances, the manufacturer's label and/or color codes the fronts of all modules to distinguish between the different types (AC input, DC output, TTL, Analog, and so on). Most manufacturers have designed each module so it can be **keyed**. In Figure 2–19, item 6, the module has been notched in two places. Installing keying bands on the Allen-Bradley I/O rack backplane connector (Figure 2–20) where a specific module is to be installed prevents any module, other than the type for which the connector is keyed, from being installed in that connector or slot. Figure 2–21 shows a close-up view of what the keying band looks like and how it is installed. Each type of module has a unique combination of notches. This feature prevents inadvertent or accidental replacement of the wrong type of module, say an input module, into a slot that is already wired to output devices. To prevent damage and downtime, it is important that the keying system be used.



1. Orange identification label
2. Status indicators
3. Protective cover
4. Field Wiring Arm connects here
5. Labels identify user outputs
6. Slotted for I/O slot insertion only
7. Blown fuse indicator

Figure 2–19 Notched AC Output Module
(Courtesy of Allen-Bradley)

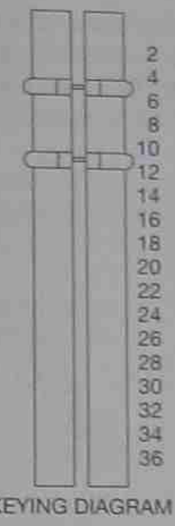


Figure 2–20 Backplane Connector Keying Diagram
(Courtesy of Allen-Bradley)

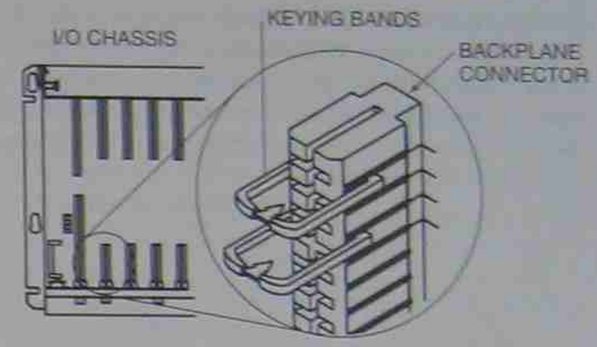


Figure 2–21 Connector Keying Pin
(Courtesy of Allen-Bradley)

There are some manufacturers that do not use the physical keying system, but rely on software configuration to prevent inadvertent or accidental replacement of a wrong module. Each module is given a unique "signature" that is relayed to the processor. If the signature does not match the software configuration, a fault will occur and prevent the PLC from operating.

DC Output Modules

DC output modules are basically the same in operation as AC output modules. The difference is the use of a power transistor instead of a Triac for the control of output current. The power transistor has a quicker switching capability than the Triac; therefore, the response time for DC modules is faster than for AC modules. The RC circuit and MOV used in the AC output module is replaced with a diode to provide protection from electrical noise and spikes. A typical DC output circuit is shown in Figure 2–22.

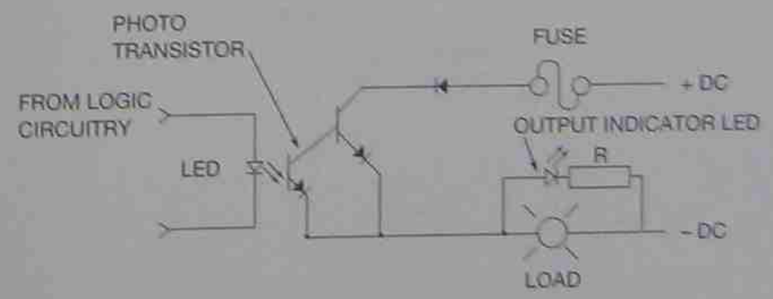


Figure 2–22 Simplified DC Output Module Circuit

DC output modules are available in ranges from 12–240 V DC depending on the manufacturer. It is important to check to make sure that the module is appropriate for the voltage and current level of the output device that is intended for connection to the module. DC modules will be fused like their AC counterparts and also have blown-fuse indicators, as well as status lights. Figure 2–23 shows how the output devices are wired to a DC output module.

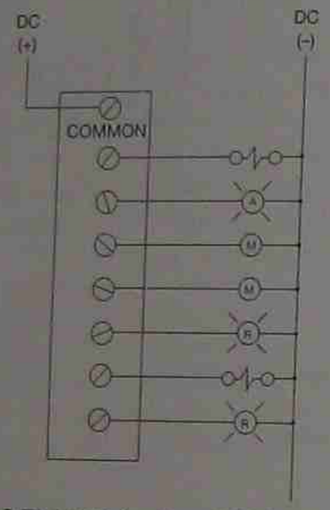


Figure 2–23 Field Wiring for a DC Output Module

Sourcing and Sinking

The terms sourcing and sinking refer to the manner in which DC devices are wired. To properly interface DC devices with the outside world, the difference between sourcing and sinking must be understood.

Figure 2–24 is an example of a sourcing application. The positive potential is connected to the input module and the negative potential is connected to the input device. Using conventional current flow (+ to –), it is said that the input module is the source of supply for the real-world input device. Stated simply, the input device receives current from the input module. In electronics, if a device (input module) provides current, or is the source of current, it is said to be sourcing.

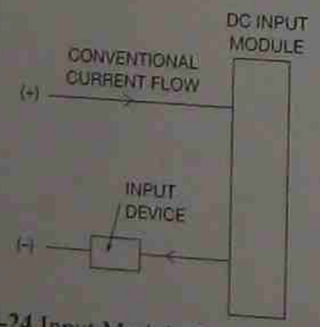


Figure 2–24 Input Module Sourcing Application

Figure 2–25 is an example of a sinking application. In this configuration, the positive potential is connected to the input device and the negative potential is connected to the input module. In this case, using a conventional current flow of + to –, the input device is said to be providing current to the input module. If the device (input module) is receiving current, it is said to be sinking.

When an output module is connected positive, as shown in Figure 2–26, and provides current to the outside world (output device), the module is referred to as sourcing.

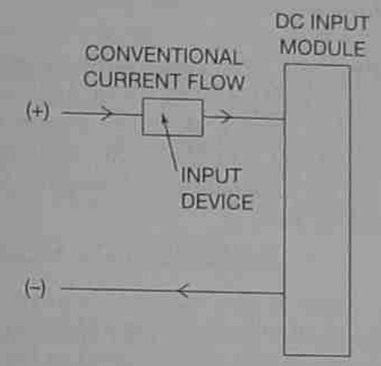


Figure 2–25 Input Module Sinking Application

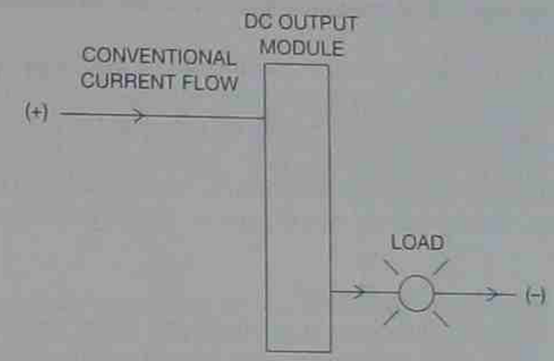


Figure 2–26 Output Module Source Capability

Figure 2–27 shows an output module wired negative, and the output device wired positive. When the output device (outside world) provides the current to the module, the module is referred to as sinking.

There is always a great deal of confusion surrounding sourcing and sinking as it applies to input and output modules and devices. As a general rule, sinking modules are used with output modules when interfacing with electronic equipment (TTL or CMOS compatible), while sourcing modules are used for such DC loads as solenoids. The safest way to make sure that connections to DC devices are correct is to ignore the terms sink and source, and select a module that works for the application, based on the manufacturer's specifications and wiring diagram.

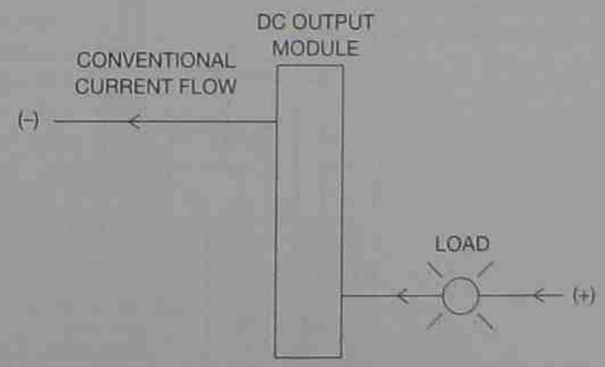


Figure 2–27 Output Module Sink Capability

Contact Output Modules

Contact modules have electromechanical relays mounted on a printed circuit board that is inserted or plugged into the I/O rack. A signal from the processor energizes the coil of the electromechanical relay which, in turn, opens or closes a set of contacts. Each set of contacts is isolated and can be ordered normally open (N.O.) or normally closed (N.C.). This type of module is used when extra current ratings are required or when it is desirable to isolate loads of different voltages to prevent them from the same source. A contact output module is also used when the leakage current of a standard AC output module would affect the control process.

For example, a contact module can be used with a Variable Speed Drive application. Assume that the drive has been programmed for multiple speed settings. To select a given speed, a circuit must be completed at two points on a terminal strip mounted on the drive. One set of contacts is used for each speed setting. As the drive supplies its own power, all that is needed is to switch the control power through the contacts on the contact module. As the contacts are isolated, there is no possibility that the power from the drive system can damage the PLC.

Interposing Relay

When it is necessary to control loads larger than the rating of an individual output circuit, a standard control relay, which has a small inrush and sealed current value, is connected to the output module. The contacts of the control relay, which are generally rated at 10 amps, can then be used to control a larger load. This method of control is a common practice for NEMA size 4 and large motor starters, depending on the rating of the output module. When a control relay is used in this manner, it is called an interposing relay (Figure 2–28).

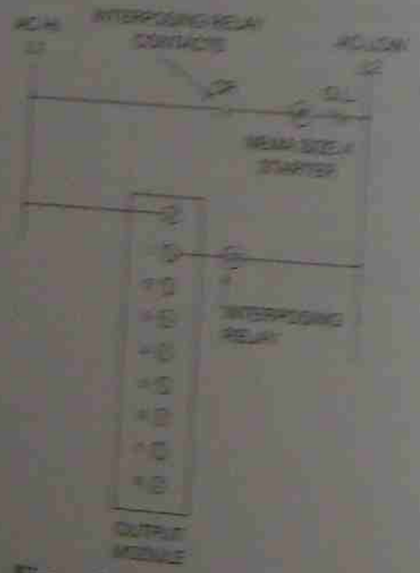


Figure 2–28 Interposing Relay

Reed Relay Output Module

The reed relay type output module is used when dry reed relays are desirable. They may be used for low-level switching (small current–low voltage), multiplexing analog signals, or for interfacing controls with different voltage levels. The voltage range of the reed relay contacts are normally in the range of 0–130 V AC or 0–125 V DC with a current rating of 1 ampere. Reed relay output modules are cheaper than normal solid-state AC/DC output modules and are usually recommended when indicator lamps are the output devices. Reed relay modules are available with normally open (N.O.) contacts, normally closed (N.C.) contacts, or a combination of both N.O. and N.C. contacts, again depending on the manufacturer.

Transistor-Transistor Logic (TTL) I/O Modules

TTL input modules are designed to be compatible with other solid-state controls, sensing instruments, many types of photoelectric sensors, and some 5 V DC level control devices. TTL output modules are used for interfacing with discrete or integrated circuit (IC) TTL devices, LED displays, and various other 5 V DC devices.

ANALOG I/O MODULES

Analog input modules are used to convert analog signals from analog devices that sense such variables as temperature, light intensity, speed, pressure, and position to 12-Bit Binary or to 3-digit Binary-Coded Decimal (BCD), depending on the manufacturer, for use by the processor. The conversion from analog to digital is accomplished with an Analog-to-Digital Converter, or ADC. The analog output module changes the 12-Bit Binary or 3-digit BCD value used by the processor into analog signals using a Digital-to-Analog Converter, or DAC. These analog signals can be used for speed controllers, signal amplifiers, or valve positioners. Binary and Binary-Coded Decimal (BCD) is covered in Chapter 6. Figure 2–29 shows an analog input module and how the analog signal source is wired.

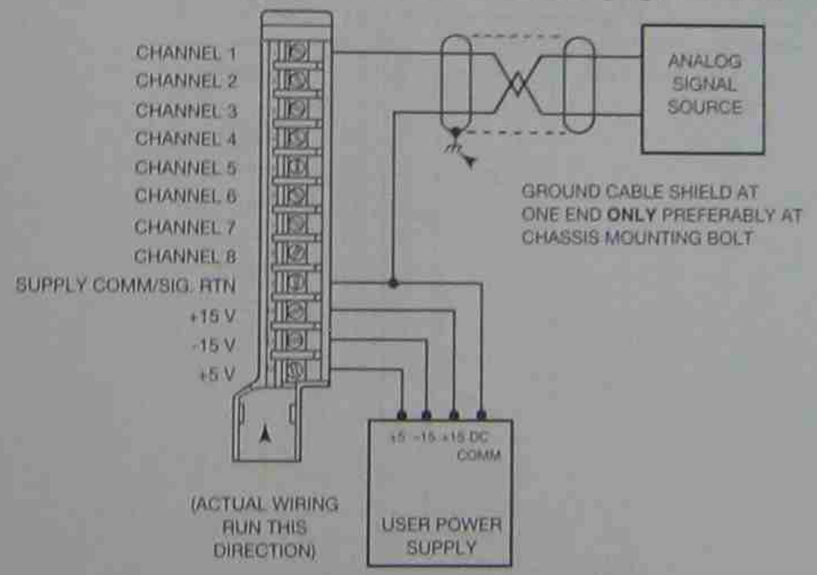


Figure 2–29 Analog Input Module *(Courtesy of Allen-Bradley)*

Another type of analog input module is the Resistance Temperature Detector (RTD) Input module. The module senses RTD signals at its inputs and converts them to corresponding temperature or ohmic values in 4-digit BCD or 12-bit binary format. Figure 2–30 shows typical wiring for an RTD device. Note that only one end of the shielded cable has been grounded.

Note: *PLC manufacturers are introducing new and special application modules almost daily. A few modules have been discussed in this chapter for a basic understanding only. The local PLC representative(s) should be contacted for a full and complete list(s) of modules that are available.*

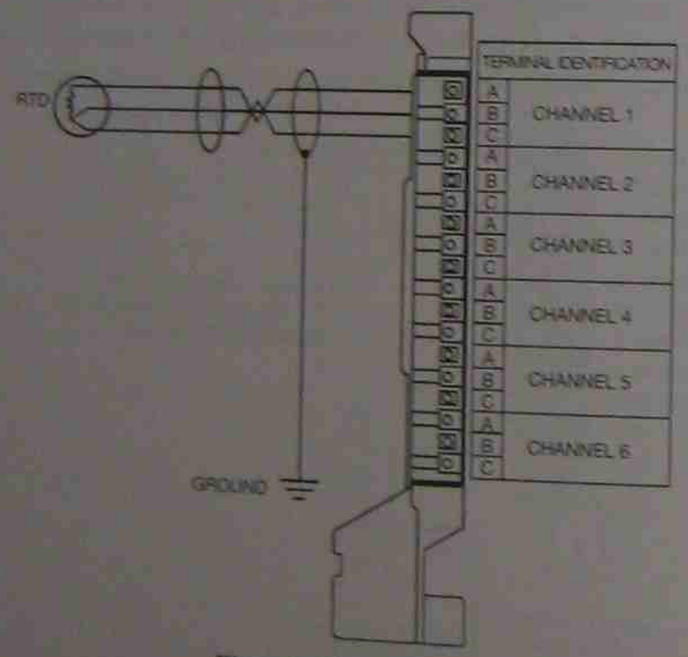


Figure 2–30 RTD Input Module
(Courtesy of Allen-Bradley)

Safety Circuit

The National Electrical Manufacturing Association (NEMA) standards for programmable controllers recommends that consideration be given to the use of emergency-stop functions that are independent of the programmable controller. The standard reads in part: "When the operator is exposed to the machinery, such as loading or unloading a machine tool, or where the machine cycles automatically, consideration should be given to the use of an electromechanical override or other redundant means, independent of the controller, for starting or interrupting the cycle."

While programmable logic controllers of today are rugged and dependable, where safety is concerned ***do not*** depend on the solid-state devices and circuitry of the PLC, or PLC program. The NEMA recommendation recognizes the importance of a hardwired emergency-stop circuit, or E-Stop, (shown in Figure 2–31) to remove power to the output devices. A second set of MCR contacts could be added in the X1 line to remove power to the input devices. It is common, however, to put the MCR contacts on the output side only so that the inputs can remain energized for troubleshooting.

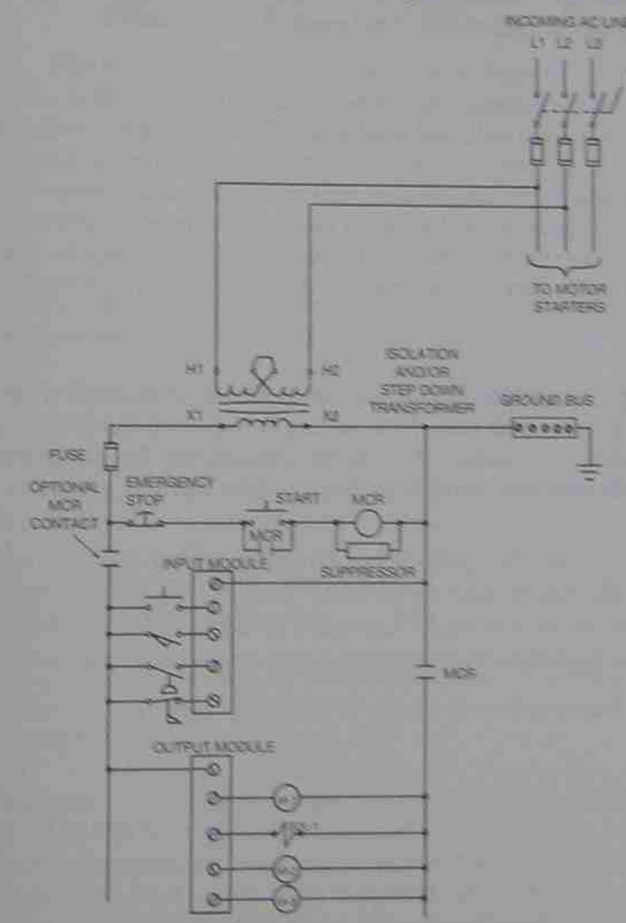


Figure 2–31 Power Distribution with Master Control Relay (MCR) for a Grounded AC System

It is also worth noting that solid-state output devices usually (though not always) fail shorted, rather than in an *open* condition. By failing in a shorted or *ON* condition, an added safety hazard is possible if a hardwire emergency stop (E-Stop) or master control is not included as part of the PLC installation.

Rack Installation

Before installing a rack or chassis, consideration must be given to the following:

- temperature
- dust
- vibration
- humidity
- field wiring distances
- troubleshooting accessibility

The ambient temperature of the proposed location should not be lower than 32˚F or higher than 140˚F (0˚C and 60˚C). Fans are normally not used with I/O racks, and all cooling of the electrical or electronic components is accomplished by convection. Convection cooling is accomplished when warm air caused by heat in the components rises and creates a movement of air. This movement of air draws cool air in through the bottom of the rack and expels warm air out through the top. To maintain efficient convection cooling, it is important that the rack be installed correctly and not used as a shelf for notebooks or other material that would impede or block the natural flow of heat up through the rack.

During initial installation it is common practice to cover the top of the rack to prevent any scrap wire, stripped insulation, screws, and nuts, from falling into the I/O modules, power supply, or processor which could cause a short circuit or other electrical failure. The protective cover must be removed after installation to assure that proper cooling can take place.

Under adverse conditions, when the ambient temperatures exceed the manufacturers' recommended maximums, rack or chassis fans can be used. The fans are mounted under the I/O rack (Figure 2–32) and force air up through the I/O assembly to aid with the movement of air. The addition of the fans reduce air temperatures within the rack and substantially increases reliability.

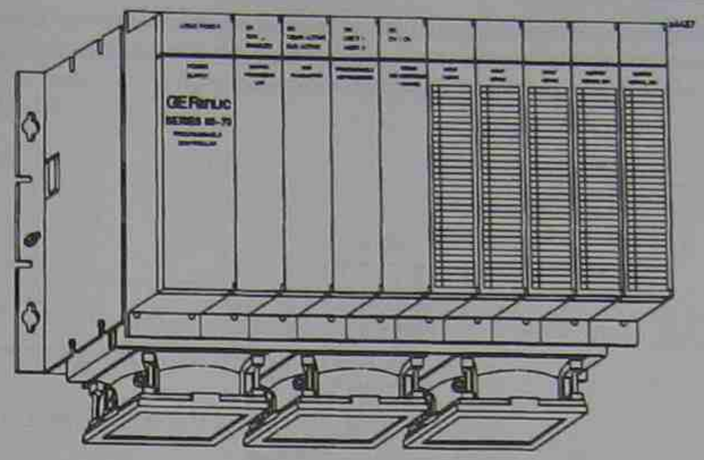


Figure 2–32 Rack Fans Mounted Under Rack to Aid Cooling

When the temperature is expected to go below 32˚F, a thermostatically-controlled heater is used inside the enclosure to prevent condensation.

Dust can also cause a problem in the I/O rack when it accumulates on the electronic components of the modules, power supply, or processor. Accumulated dust prevents the components from dissipating heat effectively. A dust-tight enclosure with a cover and gasket can be used to prevent problems that too much dust can create. It is important to remember that any enclosure used to house PLC components must be large enough to allow for proper air circulation and heat dissipation. If the enclosure is too small, the heat will build up inside the enclosure and have a detrimental effect on the electrical or electronic components. The installation manual usually specifies the minimum size of enclosure that can be used.

Excessive vibration can also lead to early component failure. It is important to mount PLC equipment on solid, nonvibrating surfaces. Vibration affects from equipment must be minimized to assure proper longevity for the equipment.

Humidity, while normally not a problem, must be considered when installing a PLC. Allen-Bradley, for example, rates their PLCs for operation in a humidity range of 5%–95% (without condensation). Some manufacturing processes, however, create high humidity (high moisture content) conditions. Exposing electronic equipment to extremely high humidity environments over an extended period of time can reduce component life and affect operation. Evaluate the environment carefully and mount equipment in an area that minimizes the exposure to high humidity and moisture.

While it is important that the controller and programming unit be installed or mounted in a control center or other central location, the use of remote I/O racks allows the input and output modules to be installed close to the actual operating equipment (Figure 2–33).

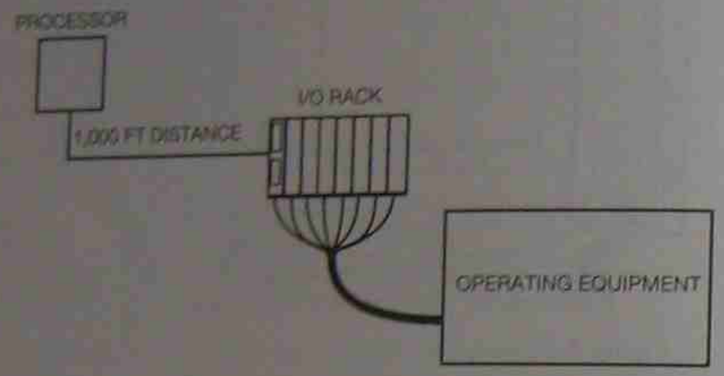


Figure 2–33 Remote I/O Rack Close to Operating Equipment

By mounting the I/O rack close to the actual equipment, the amount of conduit, cable, and other associated wiring and labor costs will be decreased. The only wiring needed for communication back to the processor will be a shielded-twisted pair, twin axial cable, fiber optic cable, and the like. By having the input and output modules located close to the process or driven equipment, troubleshooting is also easier and more efficient. As discussed earlier in this chapter, each input and output module has status lights that indicate whether an input or output device is *ON* or *OFF*. Having this capability close to the actual equipment shortens troubleshooting time and increases production.

Before mounting the racks and other associated equipment, give careful consideration to location and accessibility. If access is restricted or difficult to reach, troubleshooting, repair, and maintenance will be more difficult and time consuming.

Electrical Noise (Surge Suppression)

Electrical noise is generated whenever inductive loads such as relays, solenoids, motor starters, and motors are operated by "hard contacts" such as push buttons, selector switches, and relay contacts. The noise, or high transient voltages (spikes), are caused by the collapsing magnetic field when the inductive device is switched *OFF*. The level of the voltage spike can be very high and is capable of causing erratic operation of the processor and/or output module, or can cause permanent damage to the module. The interference caused by these voltage spikes and the accompanying electrical noise is often called Electromagnetic Interference (EMI). There are several steps that can be taken to reduce or eliminate the effects of EMI. Two of the most common are isolation and suppression.

Isolation of the electrical noise is accomplished by installing an isolation transformer for the PLC system (Figure 2–34) to supply the power for the controller and the input circuits. The figure shows a constant voltage transformer, but a standard step-down transformer of the proper size also effectively isolates electrical noise.

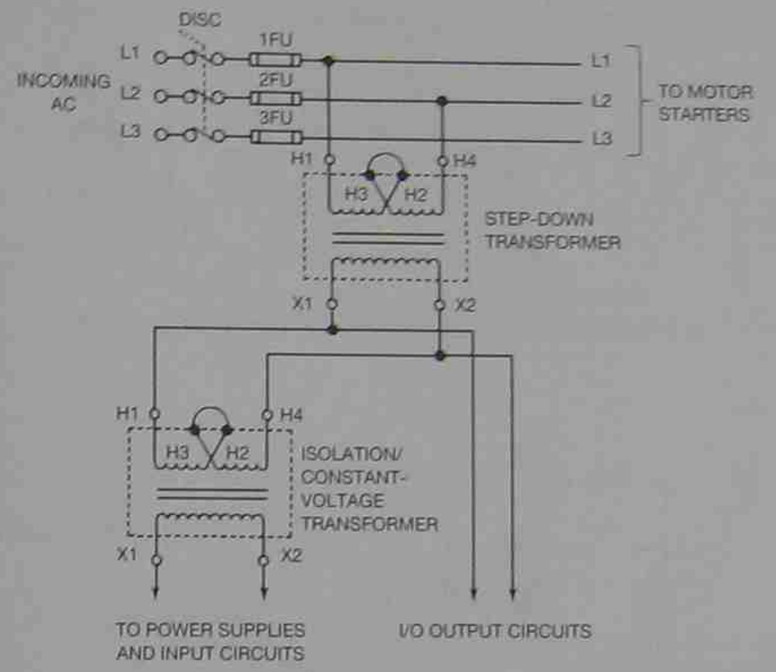
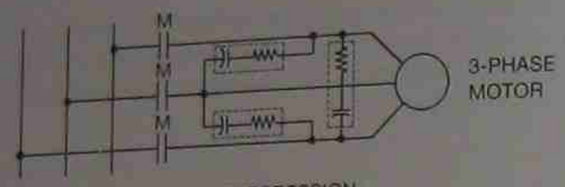


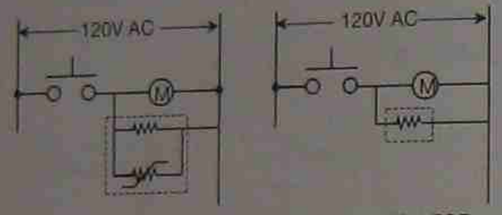
Figure 2–34 Reducing Electrical Noise with an Isolation Transformer

A second method in reducing EMI is to install surge suppression networks or devices on the individual motor starters, motors, and solenoids. These suppression devices can consist of an RC circuit (resistor/capacitor), a Metal Oxide Varistor (MOV) or a MOV and RC combination for AC loads and a diode for DC coils. The collapsing magnetic field of the inductive device is, in a sense, dissipated by the suppression network and reduces the effects of EMI. Figure 2–35 shows typical installations of the various types of suppression devices.

The type of surge suppressor to use depends on the size and type of load. An equipment representative or local electrical distributor should be consulted for help with selection and application.

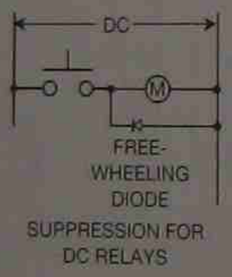


3-PHASE SUPPRESSION



SUPPRESSION FOR STARTERS UP TO SIZE 5

SUPPRESSION FOR STARTERS UP TO SIZE 1



SUPPRESSION FOR DC RELAYS

Figure 2–35 Typical Equipment Grounding Configuration

Grounding

With solid-state control systems, proper grounding helps eliminate the effects of electromagnetic induction. Figure 2–36 shows a typical installation using an equipment grounding conductor to connect several PLCs and/or I/O racks together. The equipment grounding conductor is attached to the metal frame of the PLC and/or I/O rack with a ground lug. A detail of the connection is shown in Figure 2–37.

Note: *Check local codes and manufacturers' specifications to ensure proper installation.*

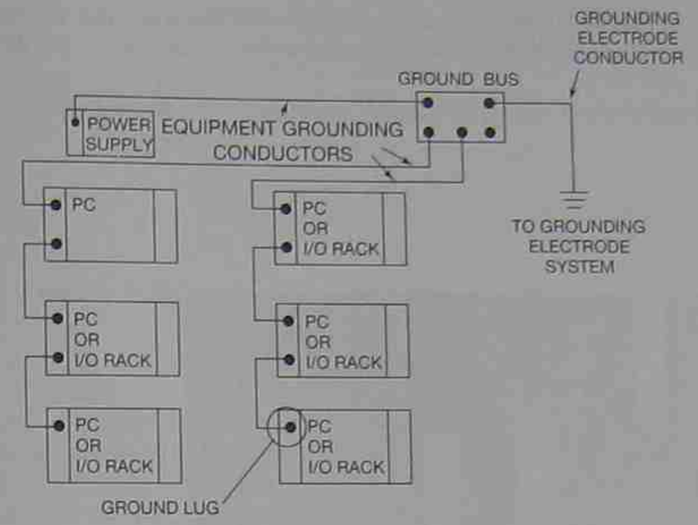


Figure 2–36 Typical Equipment Grounding Configuration

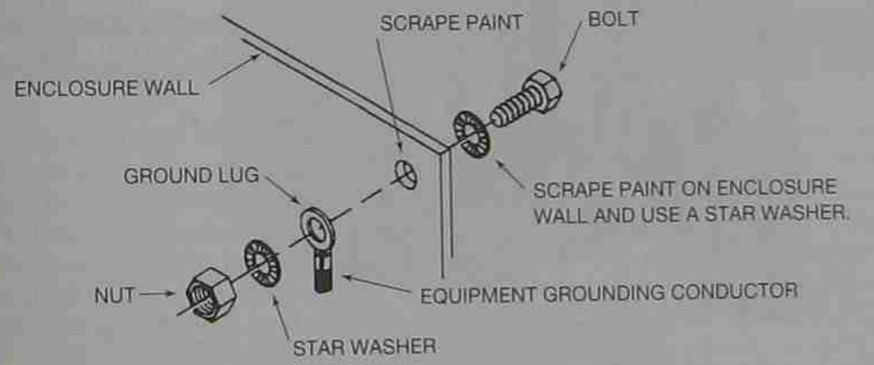


Figure 2–37 Detail of Grounding Lug Attachment to PLC

I/O Shielding

Certain I/O modules such as TTL, analog, and thermocouples require shielded cable to reduce the effects of electrical noise. The cable shield, which surrounds the cable conductors, shields the conductors from electrical noise.

When installing shielded cable, it is important that the shield only be grounded at one end. If the shield is grounded at both ends, a ground loop is created and can introduce ground currents that may result in faulty operation of the processor.

As a properly grounded I/O rack is already connected to earth ground through an equipment grounding conductor, the shield should be terminated at the I/O rack, *not* at the device end.

Figure 2–38 shows the shield of a shielded cable connected to the I/O rack frame.

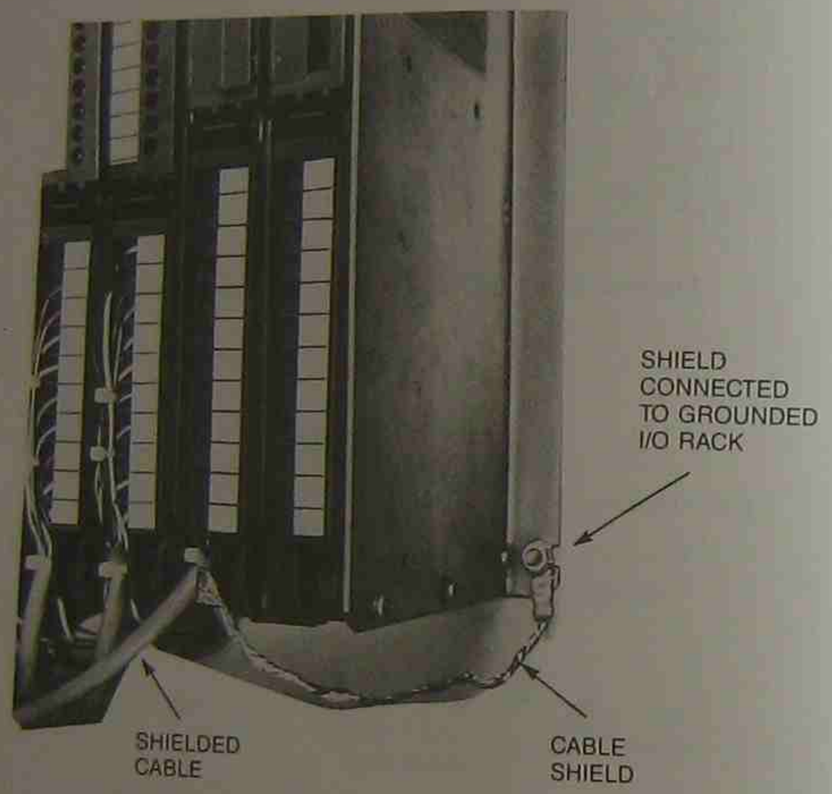


Figure 2–38 Cable Shield Connected to Grounded I/O Rack
(Courtesy of Allen-Bradley)

Figure 2–39 shows a shielded-twisted pair cable connected to a sensing device and I/O rack. Note that the shield is connected at one end only.

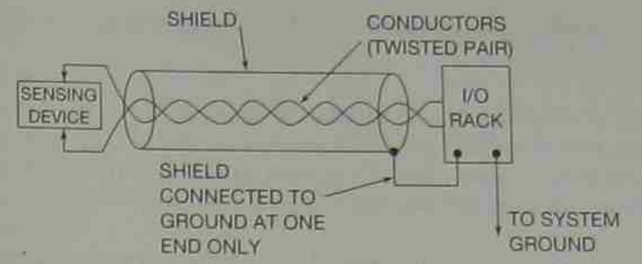


Figure 2–39 Shields Connected to Ground at One End Only

Additional methods of noise reduction are as follows:

- Mount equipment in steel metal enclosures, when possible, because metal helps protect against electromagnetic interference (EMI)
- Separate I/O and PLC wiring from the motor and other large loads to reduce the possibility of induction in the control circuits. This is usually accomplished by installing the control wiring in one raceway, or cable-tray, and the power circuits in another raceway or cable-tray with physical distance or separation between the two

Chapter Summary

The I/O rack houses the individual input and output modules that are connected to real-world devices. The input modules act as an interface between the actual input devices and the processor, while the output modules act as an interface between the actual output devices and the processor. The status (*ON* or *OFF*) of the input devices is communicated to the processor; the processor makes a decision, and in turn communicates to the output modules to turn *ON* or *OFF* the output devices that are connected to the output module. The processor may be connected to the I/O rack by way of an interconnecting cable(s), through a bus duct, or it may be mounted in the same rack as the I/O.

The I/O section is divided into two categories: fixed and modular. Discrete I/O modules operate on digital, or *ON* and *OFF* signals, whereas analog I/O operates on a variety of signal levels and types. Input and output modules are available in a variety of voltages and normally can control 8, 16, or 32 individual input or output devices. Optical coupling, or isolation, is used to protect the low-voltage (5 V DC) side of the processor from the line-voltage input and output signals that can be as high as 240 V.

AC output modules typically use Triacs for switching *ON* and *OFF* the actual output devices. Triacs when in the high-resistive, or *OFF* state, have a small leakage current that flows through the output device. When Triacs fail, they normally fail in the *ON* condition. Fuses used for protection of output modules are carefully selected by the manufacturer for current and time characteristics,

and only fuses recommended by the manufacturer should be used to prevent possible damage to the equipment.

PLC troubleshooting is simplified by the addition of status lights on the I/O modules. The lights indicate which inputs are *ON* or *OFF* and which outputs are *ON* or *OFF*. Indicator lights also indicate if an output module has a blown fuse. To prevent an incorrect module from being installed in a given rack slot, the modules are often keyed.

There is a wide variety of input and output modules available that fit almost any application. Care must be taken to ensure that the module has the correct voltage, current, and time characteristics. The various PLC manufacturers continue to introduce innovative new modules to meet the changing requirements of automated equipment and process control.

Proper installation of PLC equipment requires that the environment (dust, heat, humidity, and vibration) be considered, as well as the physical location for access and troubleshooting. Reduction and/or elimination of electrical noise, voltage spikes, voltage variation, and the like, is necessary to ensure proper operation of the system.

Review Questions

1. Describe briefly the purpose of the I/O section.
2. State two reasons for employing optical isolation.
3. Draw an input module with four input devices, show all necessary electrical connections, and identify potentials L1 and L2.
4. Draw an output module with four output devices, show all necessary electrical connections, and identify potentials L1 and L2.
5. T F Triacs are susceptible to "dielectric-type" breakdown if the maximum peak voltage level is exceeded.
6. Briefly describe why a hardwired emergency-stop circuit is recommended for PLC installations.
7. Briefly describe the function of an interposing relay.
8. T F I/O modules are keyed to prevent unauthorized personnel from removing them from the I/O rack.
9. Which of the following are *not* normally sources of electrical noise?
   a. solenoid
   b. relay
   c. indicator lamp
   d. motor starter
   e. motor
   f. overload heaters
10. T F To ensure maximum benefit of shielding, the shield of a shielded cable must be terminated and grounded at both ends.

11. E-Stop refers to
    a. extra stop
    b. emergency stop
    c. every stop
    d. elevator stop
    e. energy stop
12. T F Electromagnetic interference (EMI) can be reduced with the proper grounding of equipment.
13. Solid-state output devices tend to
    a. never fail
    b. fail in the *open* or *OFF* condition
    c. fail in the *shorted* or *ON* condition
    d. not be affected by overload
14. List three environmental considerations when installing PLC equipment.
15. What type of tool or object should be used to change the position of DIP switches?

CHAPTER

3 Processor Unit

Objectives

After completing this chapter, you should have the knowledge to:

- Describe the function of the processor.
- Describe a typical program scan.
- Identify the two distinct types of memory.
- Describe the function of the *watchdog timer*.
- Identify various memory designs.
- Identify and explain the two broad categories of memory use.
- Identify different peripheral devices that are used with a programmable controller.

The processor unit houses the microprocessor, memory module(s), and the communications circuitry necessary for the processor to operate and communicate with the I/O and other peripheral equipment. The DC power required for the processor is provided either by a power supply that is an integral part of the processor unit, or by a separate power supply unit, depending on the manufacturer. The processor, or "brain," of the programmable logic controller is the decision-maker that controls the operation of the equipment to which it is connected. The processor controls the operation of the output devices that are connected to the output modules based on the status of the input devices and the program that has been entered into memory (Figure 3–1). The processor is often referred to as the central processing unit or CPU.

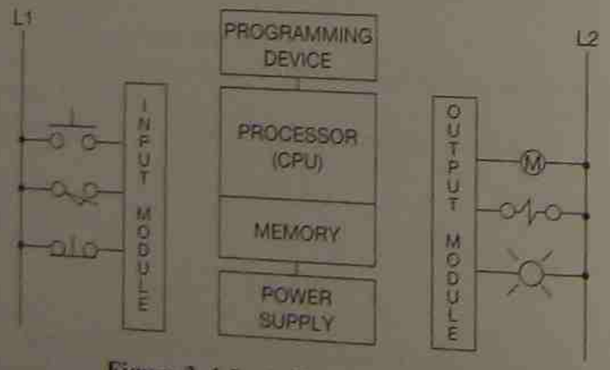


Figure 3–1 Basic PLC Configuration

Processors are available that control as few as 8, or as many as 40,000, real-world inputs and/or outputs. The size of the processor unit to be used is dependent on the size of the process(es) or driven equipment to be controlled. The larger the number of input and output devices that are required for the process, the more powerful the processor must be to properly control the number of I/O that will be connected. One PLC can control more than one machine or process line and is limited only by the I/O required, physical distance, and memory capacity of the PLC used.

Note: *It is difficult to discuss processor unit configuration due to the differences in PLC hardware from the various manufacturers. The discussion that follows is general and is not intended to cover all the PLCs on the market today. It should also be noted that when pictures are used to illustrate a given configuration or concept, one of a particular manufacturer's models is illustrated; however, the manufacturer also has other models larger and/or smaller, and in different configurations.*

THE PROCESSOR

The processor may be a self-contained unit, or may be modular in design, and plug directly into the I/O rack as shown in Figure 3–2. Whatever the configuration, the processor consists of a microprocessor, memory chips, circuits necessary to store and retrieve information from the memory, and communication circuits required for the processor to interface with the programmer, printer, and other peripheral devices. The memory and communication circuits can be modules separate from the microprocessor module. The actual hardware configuration will depend on the PLC.

Figure 3–2 Allen-Bradley SLC 500 with a SLC 5/01 Processor (CPU) Installed
(Courtesy of Allen-Bradley)

The microprocessor is the device that

1. Monitors the state or status (*ON* or *OFF*) of the input devices.
2. Systematically solves the logic of the user program.
3. Controls the state of the output devices (*ON* or *OFF*).
4. Communicates with other devices (hand-held programmers, personal computers, etc.).
5. Manages memory and updates timers, counters, and internal registers.

The execution or completion of these tasks is referred to as the processor **scan**.

When the PLC is powered up or turned *ON*, the processor runs an internal self-diagnostic, or self-check, prior to initiating its first scan. If any part of the processor system is not functioning, such as a faulty memory, improper communication with the I/O section, or failure in a remote rack, the processor fault light or other indicator light comes *ON*. With some systems, if a monitor is connected, a written explanation or fault code will appear on the VDT screen. Some systems use status words to indicate the hardware or software that has malfunctioned. Status words can be included in the program so that when a malfunction is detected, an alarm will sound to alert the operator that there is a problem.

Once the processor has passed the self-diagnostic check, it is ready to go to work. Figure 3–3 illustrates a typical four-step PLC scan. In the first step of the scan, the processor determines the status of the input devices. It does so by looking at the memory locations that have been designated for all the input devices. Remember, as stated earlier in the text, the actual status (*ON* or *OFF*) of any input device is stored in a memory location as either a 1 or a 0. A 1 indicates that a device is *ON* or closed, while a 0 indicates that the input device is *OFF* or open. Based on the 1s and 0s, the processor determines the actual condition of all the input devices.

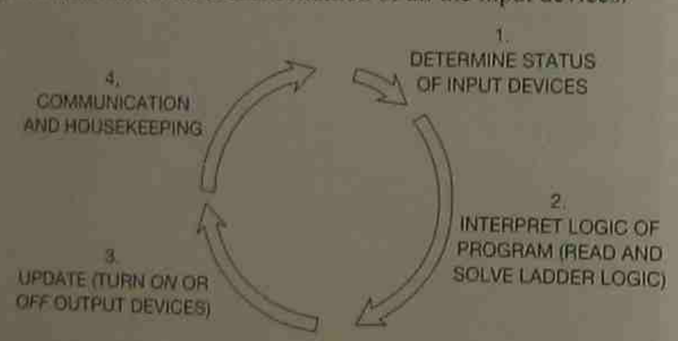


Figure 3–3 Typical Processor Scan

The second step in the processor scan is to interpret the logic of the program that has been written and stored in the processor memory. Based on the program requirement, the processor will turn the required output devices *ON* or *OFF*, which is the third step in the processor scan. This third step is referred to as **updating** the outputs. This updating process occurs once during each scan. The fourth step of the scan is often referred to as communications and housekeeping. During this part of the scan the processor will communicate with any connected devices (communications) and then perform any necessary memory management (housekeeping). Memory management will include updating timers and counters as well as internal registers, etc.

The scan is continuous and the four-step process is repeated over and over every few milliseconds. To summarize, the four steps of the scan are:

1. Determine the status of the input devices (*ON* or *OFF*).
2. Read and solve the logic of the program (ladder logic).
3. Update the output devices (turn *ON* or *OFF*).
4. Evaluate communications and housekeeping procedures.

The time it takes to complete one scan will vary from a fraction of a millisecond to 100+ milliseconds, depending on the size of the program. Each manufacturer lists the scan time based on the use of 1K of program memory. A typical scan time is 3–5 msec per 1K (1,024 words) of memory used. Scan time is also affected by factors other than just the amount of memory that was used in the program. If a program that is written to control a particular piece of equipment uses 1K of memory with a 5 msec scan time, the program is scanned approximately 200 times each second. Increasing the size or complexity of the circuit uses additional memory, and extra words (memory) increase the scan time. A program with 3K of memory takes approximately 15 msec for each scan.

The use of remote I/O racks also increases the scan time. The added time is required because the status of the input devices must be transmitted from the remote I/O location to the processor location.

Note: *As part of the processor's internal self-diagnostic system, a* ***watchdog timer*** *is used. The watchdog timer is preset to an amount of time that is slightly longer than the scan time would be under normal conditions. At the start of each scan, the watchdog timer is turned* ON *and starts to accumulate time. If the program is correct, the program scan will be completed prior to the time set on the watchdog timer, and at the end of each scan, the watchdog timer is reset to 0. If for some reason the program scan is not completed in the allotted time, indicating that there is a problem with the program, the watchdog timer will time out, which puts the processor into a faulted condition. The range of the timer is software selectable (adjustable) on many PLCs.*

Normally, before any output devices can be turned *ON* or *OFF*, the processor has to scan the entire program that is in user memory. The program may be only a few rungs long or it may be hundreds of pages in length, depending on the equipment that is being controlled. Some input devices operate so fast that by the time the user program can be read and solved and outputs updated, the input device may have changed positions more than once since the processor originally determined its status at the start of the scan. The same may be true for an output device that needs to be updated sooner than a regular scan will allow. To solve this problem, many PLCs have special program instructions that allow critical or high-speed input and output devices to be updated sooner than would be possible under normal scan conditions. The special instructions actually interrupt the scan when it is reading the program and allow I/O devices to be updated immediately.

Scan time of a given PLC and its ability to provide special function programming to accommodate high-speed or critical input and output devices must be a consideration when initially selecting a PLC.

Note: *Additional information, and a more detailed discussion on how the processor scans the user program, is covered in Chapter 10.*

The memory section of the processor consists of hundreds or thousands of locations where information is stored. In the broadest sense, the memory is divided into two classifications: user and storage. The **user memory** is for the storage of the user program which contains the relay logic, or instructions, that control the driven equipment or the process. The **storage memory** is used to store information such as input/output status, timer or counter preset and accumulated values, and internal control relays, etc. that are necessary for the processor to control the equipment or process. The actual memory structure of various PLC manufacturers will be covered in Chapter 5, while the purpose or use of each memory is covered later in this chapter.

Memory chips used in the processor can be separated into two distinct groups: **volatile** and **nonvolatile**. A volatile memory is one that loses its stored information when power is removed. Even momentary loss of power erases any information stored or programmed on a volatile memory chip.

A nonvolatile memory has the ability to retain stored information when power is removed, accidentally or intentionally.

To protect a volatile memory, backup batteries are included in the processor power supply. The batteries may be "D" size dry cells, rechargeable nickel cadmium, lead acid, or nonrechargeable alkaline or lithium types.

✲ **Caution:** Extra care must be exercised when disposing of batteries, since they are classified as hazardous waste. Special care must be taken with lithium batteries because they may explode when exposed to, or dropped into, water.

When batteries are included, they may be located in the processor, or in the power supply, depending on the PLC. Wherever they are located, there is a battery indicator light(s) to indicate the condition, or state of charge, of the batteries. Common indicator lights are *BAT OK* and *BAT LOW*. A simpler system uses one light to indicate that the battery condition is normal. When the light goes out, it is a warning that the batteries need to be replaced.

When the battery indicator light comes on, indicating that the batteries need to be replaced, the memory is still protected for a minimum of two weeks. Depending on the size of the memory and the type of batteries used, in many cases the memory remains protected for one year or more with fully-charged batteries. In reality, rarely is the power interrupted or off for more than a few hours.

The type of batteries used and the number required will vary with each manufacturer. Because alkaline and lithium batteries are not rechargeable and must be replaced periodically, care must be taken to always replace the batteries with the type specified, paying special attention to the orientation of each battery in the battery holder to ensure that proper polarity is maintained. Some batteries are available with leads that simply plug into a connector on the PLC. Figure 3–4 shows a drawing of a lithium battery with leads that is mounted on the back of the faceplate cover, and is used to backup the RAM of the GE Fanuc PLC.

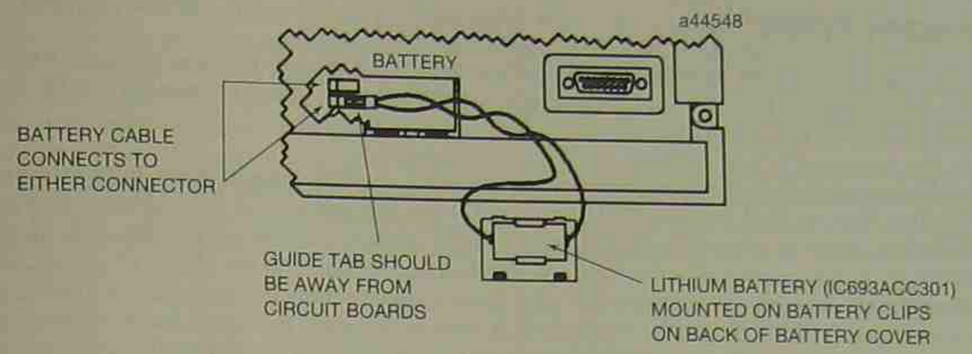


Figure 3–4 Lithium Battery with Leads
(Courtesy of GE-Fanuc Automation)

✲ **Caution:** As a general rule, a copy is made of the current program prior to changing the batteries. This copy is referred to as a "backup" copy, and is used to replace the original program if for some reason the program in memory is lost. The batteries in several PLCs can be changed without turning off the main power. The processor unit of the GE Fanuc PLC, shown in Figure 3–5, uses a single lithium battery that protects the volatile memory. Note that a second battery connector has been added and wired in parallel. The new battery is attached to the second connector in order to protect the memory before the old battery is removed. Changing batteries is one of the few maintenance requirements of a PLC. Failure to change the batteries in a timely manner may have serious consequences if a backup copy of the program is not made. Common sense dictates that a backup copy be made of every PLC program.

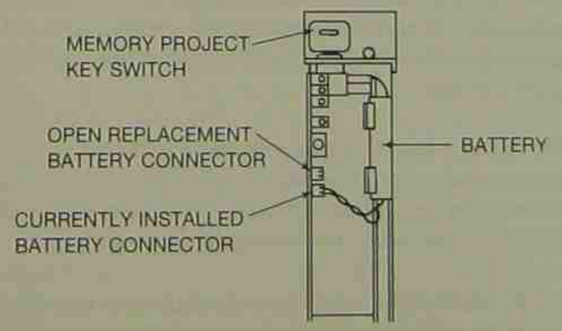


Figure 3–5 Parallel Battery Connection used with GE Fanuc 90-70 CPU
(Courtesy of GE Fanuc Automation)

MEMORY TYPES

No attempt will be made to explain solid-state memory types in more than a generalized way for basic understanding. Detailed explanations of solid-state memory types are available in the electronics section of most libraries.

The most common type of volatile memory is **Random Access Memory (RAM)**. Information can be written into, or read from, a RAM chip, and it is often referred to as read/write memory. Information stored in memory can be retrieved or read, while write indicates that the user can program or write information into the memory. Random access refers to the ability of any location (address) in the memory to be accessed or used. Ram is used for both the user memory and storage memory in many PLCs. Since RAM is volatile, it must have battery backup to retain or protect the stored program. Various forms of RAM include MOS, HMOS, and CMOS-RAM (Complimentary Metal Oxide Semiconductor), one of the most popular, to name just a few.

CMOS-RAM is popular because it has a very low current drain when not being accessed (15 μ amperes), and the information stored in memory can be retained by as little as 2 V DC. A typical fully charged lithium battery is rated 2.95 V at 1.75 amperes/hours and normally holds or protects a program for 60 days or longer.

Nonvolatile memories are memories that retain their information or program when power is lost, and do not require battery backup. A common type of nonvolatile memory is **Read Only Memory (ROM)**. Read only indicates that the information stored in memory can be read only, and cannot be changed. Information in ROM is placed there by the manufacturer for the internal use and operation of the PLC, and the manufacturer does not want the information changed or altered. PLCs, like other computer-based systems, undergo constant change. When changes are made in the way a system operates, or when new features are added, ROM chips can be replaced to upgrade the PLC.

Other types of nonvolatile memory are PROM, UVPROM, EPROM, EAROM, and EEPROM.

PROM Programmable Read Only Memory allows initial and/or additional information to be written into the chip. PROM may be written into only once after being received from the PLC manufacturer, and programming is accomplished by pulses of current. The current melts fusible links in the device, preventing it from being reprogrammed. This type of memory is used to prevent unauthorized program changes.

NOTE: *Regardless of the memory type, the memory can also be protected by a key switch located on the front of the processor, or on the programming device. With the programmer "locked-out," the program in the processor can be run but not changed. The key switch can also be used to lock the processor out completely and prevent it from running the program.*

Another popular method of restricting access to the program is to use "passwords." Passwords restrict access to the program to only those personnel who know the correct password and how to enter it using the programming device. Passwords are often referred to as "software" locks, whereas key switches are referred to as "hardware" locks.

UVPROM–EPROM Ultra Violet Programmable Read Only Memory is ideally suited when program storage is to be semipermanent, or additional security is needed to prevent unauthorized program changes. The UVPROM chip is also referred to as **EPROM** (Erasable Programmable Read Only Memory) (Figure 3–6). The EPROM chip has a quartz window over a silicon material that contains the electronic integrated circuits. This window is normally covered by an opaque material, but when the opaque material is removed and the circuitry exposed to ultraviolet light, the memory content can be erased. Once erased, the EPROM chip can be reprogrammed, using a special programmer. After programming, the chip window must once again be covered with an opaque material, such as electrician's tape, to avoid undesirable alteration of the memory.

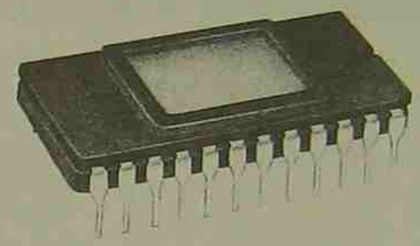


Figure 3–6 Typical UVPROM or EPROM Memory Chip
(Courtesy of Allen-Bradley)

Caution: Special care and handling of the UVPROM, or for that matter any integrated circuit (IC) chip, must be exercised to ensure that the pins do not become dirty, bent, or subjected to any static electric charges.

EAROM Electrically Alterable Read Only Memory chips can have the stored program erased electrically. This is accomplished by applying different values of positive (+) and negative (–) voltage values to specific circuit points. When erasing an EAROM chip, the voltage values and pin locations are supplied in the manufacturer's literature. Once erased, the EAROM chip can be reprogrammed.

EEPROM Electrically Erasable Programmable Read Only Memory is also referred to as Double EPROM and E2PROM. EEPROM is a chip that can be programmed using a standard programming device and can be erased by the proper signal being applied to the erase pin. EEPROM is used primarily as a nonvolatile backup for the user program RAM. If the user program in RAM is lost or erased, a copy of the program stored on an EEPROM chip can be downloaded into RAM. It is common on some PLCs for the processor to load the program from the E2PROM chip into RAM memory each time the processor is powered up or after a power failure. Figure 3–7 shows an EEPROM memory card used with the Modicon 984-120 Compact PLC to store the user program. This credit-card size device offers a convenient method for copying and/or loading user programs.

Figure 3–7 EEPROM Memory Card
(Courtesy of Modicon Inc.)

MEMORY SIZE

PLCs are available with memory sizes ranging from as little as 256 words for small systems up to 2 Meg (million) for the larger systems. Memory size is usually expressed in K values: 2K, 4K, 16K, and so on. K, or Kilo, which usually stands for 1,000, actually represents 1,024 in computerese. The difference between a standard K (1,000) and the 1,024K value used with processors and computers, is due to the way the words were counted. One of the counting, or numbering systems used with PLCs is the **binary system**. The binary system has a base 2, as contrasted to the decimal system we use every day that is base 10. Base 10 represents the numbers 0 through 9, which is 10 digits. The binary numbering system with a base 2 only has 2 digits. The digits are 1 and 0. As with the decimal system that has place values (tens, hundreds, thousands), the binary system also has place values. These are 1, 2, 4, 8, 16, and so on, and each place value is equal to twice the value of the previous number. Base 2^0 represents the number 1, base 2^1 represents 2, base 2^2 represents the number 4 ($2 \times 2 = 4$), base 2^3 represents the number 8 ($2 \times 2 \times 2 = 8$), and so on. Counting in this fashion, 2^{10} would equal 1,024. While 1,024 is actually larger than the 1,000 that K actually represents, K (with a value of 1,024) is used in PLCs, and personal computers as well. This also explains the reason for the odd memory sizes of individual memory chips: 256 (1×2^8); and 512 (1×2^9). A memory chip of 256 words would be ¼K and 512 words would be ½K. A PLC with a total memory of 64K would actually have 65,536 words of memory (64 times 1,024). Words, word structure, and numbering systems are covered in Chapter 6.

While it is common for PLCs to measure their memory capacity in words, it is important to know the number of bits in each word. A PLC that uses 8-bit words would have half the memory capacity of a PLC that uses 16-bit words. For example, the PLC that uses 8-bit words has 65,536 bits of storage with an 8K word capacity ($8 \times 8 \times 1024 = 65{,}536$), whereas a PLC using 16-bit words has 131,072 bits of storage with the same 8K memory ($16 \times 8 \times 1024 = 131{,}072$). It is important to know the word size of any given PLC before memory size can be accurately compared.

Note: *Personal computers and some PLC manufacturers, such as Simatic T.I., size memory in bytes, not words. A byte is 8 bits, or half of a 16-bit word. A 32-bit word would have 4 bytes.*

The actual size of the memory required depends on the application. In the event that future expansion is planned, there are two options: buy a PLC with more memory than is presently necessary to allow for future expansion; or buy a PLC that meets present needs and add memory (upgrade) when the need arises. Depending on the manufacturer, adding memory may be as simple as replacement of the memory module, or it may require that additional memory chips be added to the existing memory module. Some processors have no provisions for memory expansion and must be replaced if the memory needs to be increased.

GUARDING AGAINST ELECTROSTATIC DISCHARGE (ESD)

A major cause of failure of memory chips and other sensitive electronic components is electrostatic discharge. ESD is simply the discharge of static electricity. Static electricity can build up on the surface of a workstation, on clothing, the carpet on the floor of the workplace, on plastic cups, styrofoam, cellophane, and other such materials. To reduce the possibility of damage from ESD, the following precautions should be taken:

1. Use nonstatic floor coverings.
2. Handle chips correctly.
3. Ground the work surface.
4. Wear a wrist strap.

Grounding the work surface and wearing a wrist strap are the two most important precautions to take. Be sure to put on the wrist strap before starting to handle memory chips. Make sure it fits snugly, that the metal pad of the strap is touching your skin, and that the wrist strap ground wire is securely fastened to ground.

Although some manufacturers indicate that ESD is not a problem with their products, this is misleading; always follow the ESD precautions when handling memory chips and other sensitive electronic components.

For PLCs in which the memory can be expanded by adding volatile RAM chip(s) to the memory module, the following procedures should be used:

1. Record a copy of the current user program on disk or magnetic tape, depending on the system.
2. Remove main power from the PLC.
3. Remove the memory module and take to a clean area.
4. Carefully remove any screws necessary to gain access to the printed circuit board where the extra RAM sockets are located. If the backup battery is located on the module, disconnect the battery before removing or installing memory chip(s).

NOTE: *The RAM chip(s) will come packaged in a conductive plastic bag (often referred to as a "static" bag). Within the bag, each RAM chip will be inserted into a conductive sponge-like material. The conductive, yet highly resistive, material is used to keep all the pins of the chip at the same electrical potential.*

Caution: When working with the RAM chips, *do not* handle cellophane covered articles such as cigarette packages or candy wrappers, plastic, styrofoam, or other materials that can cause a static charge. *Do not* install the chip in carpeted or contaminated areas where pins may become fouled. And *never* slide the RAM chip across any surface, store a RAM chip in a conductive plastic bag, or insert the chip into conductive material.

The volatile RAM chips used today are not as susceptible to damage from static charges as they were a few years ago. But rather than just removing the chip from the conductive material and installing it into the proper socket, the following precautions still should be used:

1. Ground all tools before contacting the RAM chip.
2. Wear a conductive wrist strap which has a minimum 200K ohm resistance and is connected to earth ground as shown in Figure 3–8.
3. Control relative humidity at 40%–60%, if possible.

Figure 3–8 Wrist Strap Grounding Device
(Courtesy of GE-Fanuc Automation)

Remove the chip from the conductive foam. Be careful to touch only the chip base. *Do not* touch the pins. Inspect the pins for proper alignment. If any pins have been bent, gently straighten them using needle-nose pliers that have been grounded. A dot or notch on the case of the chip is used for proper orientation of the chip into the socket. Grasp the chip by both ends and gently set it in the socket. *Do not insert*. Be sure the chip is positioned so the dot or notch of the chip matches the dot or notch on the socket.

Before attempting to insert the chip into the socket, check each pin to make sure it lines up properly with the corresponding socket point. Make any necessary pin adjustments as outlined above.

When pin alignment is ensured, insert the chip into the socket. Insertion is accomplished by pressing *gently* on the case of the chip until the chip is fully seated into the socket.

Carefully reassemble the memory module, remembering to reconnect the backup battery if one was mounted on the module. Reinstall the module in the processor and reapply power to the system. The user program can now be reentered into the processor and any additional user program can be added using the new memory chip(s).

MEMORY STRUCTURE

As indicated earlier, the processor memory is divided into two general classifications: user memory and storage memory.

User memory contains the ladder diagram instructions programmed by the user. The instructions are entered either by a programming device, hand-held or desktop-type, a magnetic tape, or a system computer. Programming can also be accomplished by using a telephone interface from a remote computer, tape loader, or programming device. The user memory may also store user programmed messages that are recalled or activated by contacts in the ladder diagram. Upon activation, the message(s) is displayed on the VDT of the programming device or other remote VDT, or can be printed out on a compatible printer.

Message storage and recall can be used to alert an operator if a bearing is heating up. A typical programmed message might be: "Bearing hot on Motor #4." A thermocouple at the bearing of Motor #4 would be tied to one input circuit of a thermocouple input module. When the temperature of the bearing exceeds a predetermined value, the thermocouple causes that circuit of the thermocouple input module to close, or turn *ON*. The corresponding contact of the input device programmed in the ladder diagram closes, and the message is generated.

Storage memory is where the status (*ON* or *OFF*) of all input and output devices is stored. Numeric values of timers and counters (preset and accumulated), numeric values for arithmetic instructions, and the status of internal relays also are stored in this memory.

While the information presented in this section applies generally to all PLCs, more specific information and memory structure can only be obtained by reviewing the specifications and literature of individual manufacturers. In subsequent chapters, the memory structure of specific PLCs will be discussed and illustrated, but the text does not cover all the PLCs on the market today.

PERIPHERALS

Peripherals are defined as devices connected to a processor that provide an auxiliary or support function.

A printer (Figure 3–9) for producing hard copy printouts of the user program and/or other processor information is an example of a peripheral. The printer may be activated by a keyed sequence entered from the programming device and/or initiated from the processor itself. The mode of printer communication (serial or parallel) and the cables that are required vary, based on the hardware and/or software used. The connector for connecting the printer may be mounted on the back of the programming device, on the processor, or both. Figure 3–10 shows a connector on the back of a programmer. To avoid the frustration of having the printer and processor not communicate with each other, consult the installation manual(s) for the PLC that is being installed.

Using a compatible computer terminal and the appropriate software enables the user program and data files to be stored on floppy disk and/or the hard drive. Figure 3–11 shows an Allen-Bradley T60 industrial terminal.

Figure 3–9 Dot Matrix Printer
(Courtesy of OKIDATA)

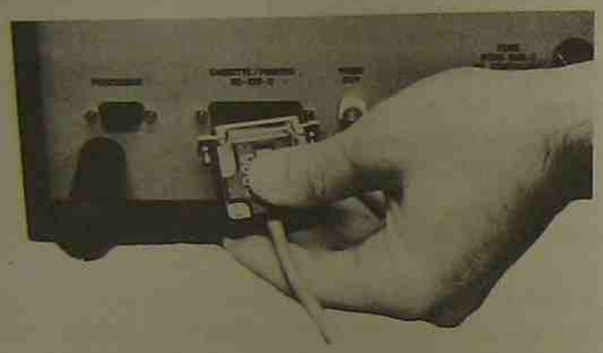


Figure 3–10 RS-232C Connection for Printer/Cassette
(Courtesy of Square D Company)

Figure 3–11 Allen-Bradley T-60 Industrial Terminal with 3 1/2" Floppy Disk and Hard Drive
(Courtesy of Allen-Bradley)

Another peripheral device is the magnetic tape loader. The tape loader is used to record and store the user program or to load preprogrammed instructions into the processor. Data quality cassette tapes or data cartridges are used by the tape loader, depending on the type, to record and store the user program. Recording the user program provides a backup program in the event the processor program is lost due to memory failure or accidental erasure.

A standard cassette recorder can be used with some smaller PLCs for recording and loading programs. Expense is greatly reduced when an ordinary cassette recorder or player can be used.

Dedicated tape loader and cassette recorders, while still in use in many plants, are not as popular as they were a few years ago. The increased use of E2PROM, UVPROM, and EAROM memory chips, as well as the increased use of the personal computer for programming PLCs, has eliminated the need for tape loaders, because the program can automatically be stored on the memory chip, or stored on the computer floppy disk or hard drive.

Another peripheral device is the **MODEM** (**MO**dulator and **DEM**odulator). The MODEM can be used to connect the processor via telephone lines to a programming device, computer, tape loader, printer, or to another PLC. Figure 3–12 shows a typical MODEM hookup from a PLC processor to a remote programming device.

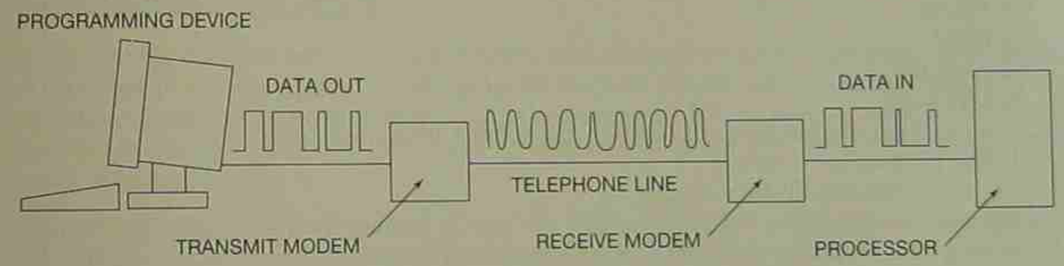


Figure 3–12 Typical MODEM Hookup

Chapter Summary

The processor contains the circuitry necessary to monitor the status (*ON* or *OFF*) of all inputs and control the condition (*ON* or *OFF*) of all outputs. It also has the ability to solve and execute the individual program steps in the user program, has a memory for storing the user program and other numeric information, and has the ability to retrieve and use any and all information stored in memory. The processor not only communicates with I/O racks and the programming device, but also communicates with any peripheral device(s) that may be connected to it. The memory used in PLCs is of two distinct groups: volatile and nonvolatile. Volatile memory requires a battery backup to prevent the program from being lost due to a power failure; nonvolatile memory holds the program when power is lost or turned off. Programs can be stored on various types of memory chips, as well as on floppy disks or on the hard drive of a computer programmer.

Peripherals such a printers, tape loaders, and MODEMs are devices that can be connected to the PLC to provide various support functions.

Review Questions

1. The processor is often referred to as the _____ of the programmable controller.
2. Briefly describe *volatile memory*.
3. Briefly describe *nonvolatile memory*.
4. 1K of memory is actually
   a. 1,000 words
   b. 1,010 words
   c. 1,024 words
   d. 1,042 words
5. Calculate the actual number of words in an 8K memory.
6. The most common type of volatile memory is
   a. PROM
   b. EAROM
   c. UVPROM
   d. RAM
7. Which of the following are types of nonvolatile memory?
   a. EEPROM
   b. PROM
   c. RAM
   d. EAROM
   e. UVPROM
8. List the two broad categories of memory (not volatile and nonvolatile).
9. Define the word *peripheral*.
10. List two common peripheral devices.
11. What is a *watchdog timer*?
12. What special precautions should be taken with lithium batteries?
13. T F When a PLC is first turned *ON*, it will run a self-diagnostic or self-check test.
14. Describe the four steps of a typical PLC processor scan.
15. T F The actual scan time, or time it takes the PLC to complete a four-step scan, decreases as the number of program words increases.

CHAPTER

4 Programming Devices (Programmers)

Objectives

After completing this chapter, you should have the knowledge to:

- Describe the function of a programming device.
- Describe the three types of programming devices.
- List advantages and disadvantages of the three types of programmers.
- Explain the term *on-line* and *off-line programming*.

PROGRAMMING DEVICES

A programming device is needed to enter, modify, and troubleshoot the PLC program, or to check the condition of the processor. Once the program has been entered and the PLC is running, the programming device may be disconnected. It is not necessary for the programming device to be connected for the PLC to operate, but it can be used to monitor the PLC program while the program is running.

Programming devices, or programmers as they are most often called, come in three types: hand-held, dedicated desktop, and computer (Figures 4–1, 4–2a, and 4–2b).

Figure 4–1 Hand-Held Programmer
(Courtesy of Modicon Inc.)

Figure 4–2a Dedicated Desktop Programmer
(Courtesy of Allen-Bradley)

Figure 4–2b IBM® Laptop Computer

Each type of programmer has advantages and disadvantages. After all three types have been described, the merits of each will be discussed.

Dedicated Desktop Programmers

Dedicated desktop programmers like the one pictured in Figure 4–2a are primarily designed for programming and monitoring the PLC. They are not capable of performing other computer functions like running software programs for word processing and spreadsheets, but are, however, designed to be portable and withstand the mechanical shock associated with moving the programmer from job site to job site. They are also designed to function in the industrial environment where there is electromagnetic noise, high temperatures, and humidity.

A typical dedicated programmer consists of a keyboard, VDT (video display terminal), and the necessary electronic circuitry and operating memory for developing, modifying, and loading a program into the processor memory.

Keyboard The keyboard of the dedicated programmer may have raised keys, like a standard computer terminal, or have a flush, sealed touchpad-type keyboard (Figure 4–3). The flush, sealed touchpad keyboard is ideally suited for environments that are dusty or have metallic particulates in the air.

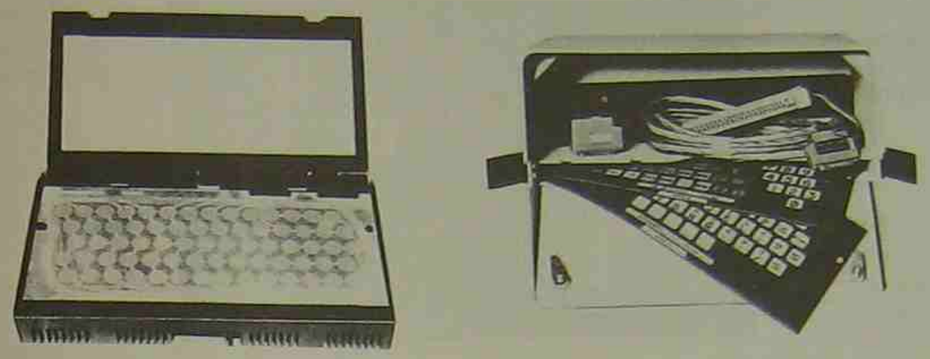


Figure 4–3 Sealed Touchpad Keyboard and Keyboard Overlays
(Courtesy of Allen-Bradley)

Keyboards on dedicated programmers vary quite a bit, depending on the manufacturer. Most dedicated programmer keyboards have electrical symbol keys for normally open contacts, normally closed contacts, timer contacts, and the like. The various symbols are also referred to as **instructions**. The normally open contact symbol key, when pressed, is an instruction to the processor that creates a contact in the user program. Other symbol keys could include: output symbol, timer, counter, etc. These dedicated keys make programming easier for those electricians or technicians who are intimidated by computers or computer-like devices. The keyboard also has special function keys that are used for program development, along with numeric (number) keys for addressing the various components used in the circuit. The address of a device tells the processor the type of device it is (input or output) and its location. Most keyboards also have alpha (letter) keys for writing program notations, labeling the devices in the program, report generation, and other special programming functions.

Video Display Terminals Video display terminals give the user a visual display of the program and range in screen or tube sizes from 5 inches to 12 inches, measured diagonally. The video screen allows the user to display multiple lines or rungs of the program. The ability to view several lines of the program at once is a powerful troubleshooting aid.

Video brightness and contrast adjustments are found either on the front or rear of the programmer. A video jack is available on the rear of most programmers for connecting an additional video monitor. VTDs are sometimes referred to as CRTs (cathode ray tubes).

Once the program has been entered into user memory, the PLC is ready to control the circuit. When the processor is placed in the *RUN* mode and the circuit activated, the programmer VDT gives a visual display of the circuit condition.

Actual circuit condition is shown on the VDT display in basically two ways: some PLCs intensify, or make brighter, all contacts, interconnecting lines, and coils that are passing current or have

power flow; others intensify or use reverse video to indicate which contacts and coils have power flow. Figure 4–4a illustrates how a circuit appears before the *START* button is pushed for a system that intensifies contacts, interconnecting lines, and coils; Figure 4–4b shows the VDT display after the *START* button is pushed and the holding contacts close.

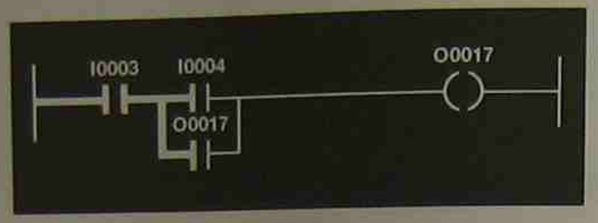


Figure 4–4a VDT Display Prior to *START* Button Being Depressed

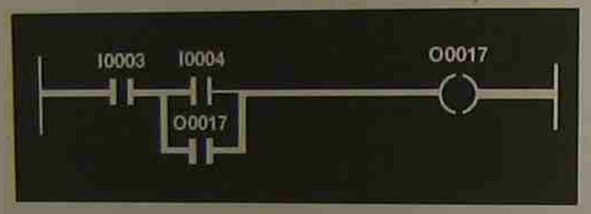


Figure 4–4b VDT Display After *START* Button is Depressed and Holding Contacts Close

The terms "Passing current" or "power flow" are hold-overs from hardwired circuits. In fact, there is no power flow, or current flow, as we normally think of it; rather, it is logic continuity or logically true statements. The logic of the circuit shown in Figure 4–4a is as follows; if input I0003 and input I0004 are closed or *true*, then output O0017 should be turned *ON* or go true. When output O0017 goes true, the instruction labeled O0017 (holding contacts) will become true, and an alternate logic statement is now true. When I0003 and O0017 are true, output O0017 will remain true or *ON*, even if input I0004 opens, or goes *false*.

Figure 4–5a illustrates how a display using reverse video looks before the *START* button is pushed, and Figure 4–5b shows the display after the *START* button is depressed and the holding contacts close.

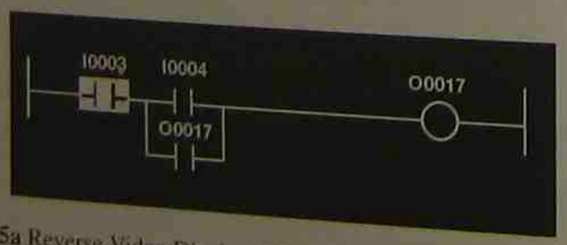


Figure 4–5a Reverse Video Display Prior to *START* Button Being Depressed

Figure 4–5b Reverse Video Display After *START* Button is Depressed and Holding Contacts Close

No matter which method is used, this feature of dedicated desktop programmers is a powerful troubleshooting aid. By viewing the display on the VDT, the electrician or technician can determine which contacts are closed and which outputs are turned on.

Note: *An output coil that is intensified only indicates that the output module circuit is* ON. *It does not guarantee that the actual output device is* ON. *However, if the output device and associated wiring are complete, the output device will be* ON *any time the output module circuit is* ON.

To provide a dependable backup of the program in case the memory fails or is inadvertently cleared or altered, an E^2PROM chip, floppy disk, or a cassette tape is used to record and/or load the user program. Data quality tapes should be used to ensure accurate transfer of data.

Hand-Held Programmers

Hand-held programmers are smaller, cheaper, and more portable than dedicated desktop programmers. While the portability is a real plus, the hand-held programmer has some limitations.

Unlike the desktop programmer that can display a complete circuit network on the VDT, hand-held programmers have limited display capabilities. Some hand-held programmers display a Rung of logic with up to four horizontal lines, while others only display one line or one element at a time. The display is either LEDs or liquid crystals. Figure 4–6 shows an Allen-Bradley hand-held programmer with liquid crystal display.

Figure 4–6 Hand-Held Programmer
(Courtesy of Allen-Bradley)

The hand-held programmer may not have the full programming features of the dedicated desktop programmer, and requires more "keystrokes" to actually enter a program. Hand-helds typically have restricted access to the processor memory.

On the plus side, hand-held programmers are well-suited for installations that require constant changes in circuit requirements because they are light-weight (normally less than 2 pounds), portable, and ruggedly constructed. It is much easier to connect the hand-held programmer to the processor for changing program parameters or for troubleshooting than it is to bring out the large, heavier desktop programmer.

While the relatively low cost of hand-helds make them affordable troubleshooting tools, it takes more time to go through the program one contact or rung at a time. The extra time is the trade-off for the lower initial cost of the programming device.

Computer Programmers

With software available for all major brands of PLCs, the personal computer in now the most common programming device. The personal computer is lower in cost than a dedicated desktop programmer, and has the added feature of being able to perform functions other than programming a PLC.

PLC programming software has been developed by the PLC manufacturers themselves and also by other companies. Figure 4–7 shows a personal computer being used to run PLC software.

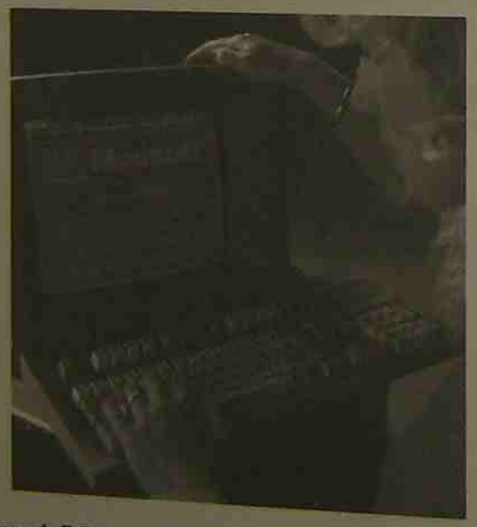


Figure 4–7 Personal Computer Running PLC Software
(Courtesy of Modicon Inc.)

When software for a specific brand of PLC is developed by a company other than the manufacturer of that PLC, the software is referred to as "third party software." These software programs can generally be used with most personal computers in either a DOS® or Windows® environment.

Using a personal computer for programming provides many of the advantages of the dedicated desktop programmer, but also provides features not available on most dedicated desktop programmers.

The personal computer usually has a color monitor (VDT), and the monitor shows multiple rungs of program logic, as well as highlighting the circuit elements to indicate status, just like the dedicated desktop programmer. The computer has the added ability to interface the PLC software program with other software programs for "cut and paste" program development and editing. PLC software usually provides more documentation capabilities than does the dedicated desktop processor. The documentation may be in the form of labeling each element, or writing rung comments. Added graphic capabilities are also normally a part of a PLC software program. The addition of graphics allows the electrician or technician to develop flow diagrams of the controlled equipment (Figure 4–8), provide operator alarms and messages, and utilize other useful process information.

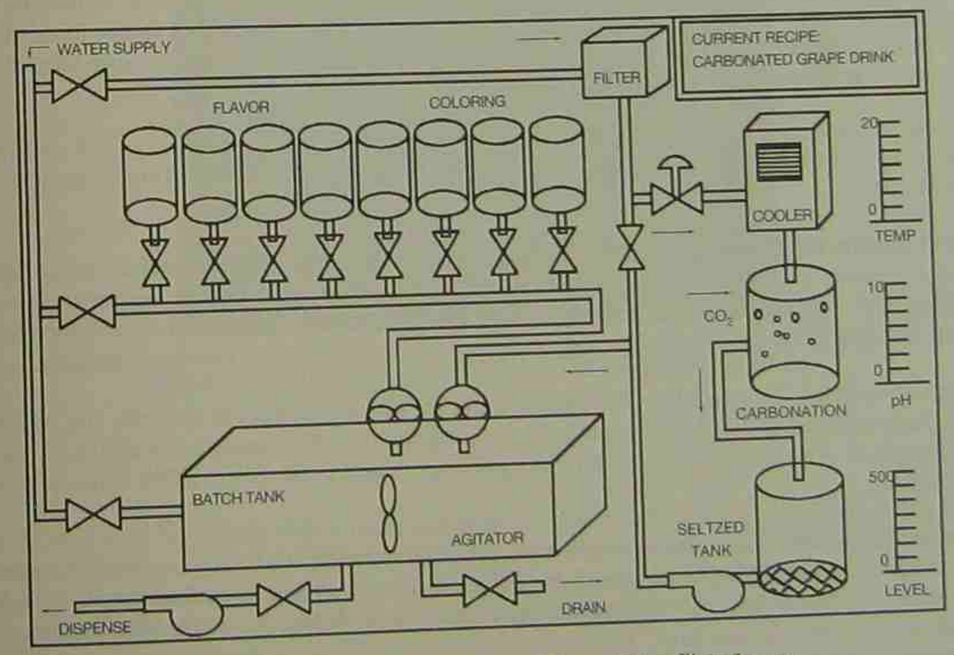


Figure 4–8 Flow Diagram Using ControlView™ Software
(Courtesy of Allen-Bradley)

A personal computer is relatively low in cost when compared to a dedicated desktop programmer. Some PLC manufacturers and software suppliers, however, require a communication card to run their software, and this requirement adds to the initial cost. The communication card must be installed in an empty slot in the personal computer, and without the card, the software will not communicate with the processor. If the software was copied and loaded into another computer, the computer could not communicate with the PLC without an additional communication card. Although the software could be used to develop a program on a computer without a communication card, the program would have to be transferred to the computer with the card to run the program. When a program is first developed and edited, it is done in the program mode or the **off line** mode. Off line indicates that the program has not yet been loaded into the processor. The program is not operational until it is loaded into processor memory, and the processor is placed in the *RUN* mode. Once the program has been checked, the program is loaded into the processor for testing and further verification and/or modification. Changes that are made after the program has been loaded into the processor and the I/O section has been activated, is called **on line** programming.

Caution: Making changes to the program while the program is running and the driven equipment is operational (on line programming) must only be done by trained personnel who not only understand the PLC program, but also thoroughly understand the driven equipment and/or process.

Other manufacturers use what is called a **hardware key** to prevent unauthorized use of their software programs. The hardware key is usually connected to one of the printer ports. When the software is first turned *ON*, the software looks for the hardware key. If the key is not installed, the software will not function. Still other manufacturers do not copy-protect their software, and, in fact, encourage copying and widespread use of the software.

Personal computers are also available in the small laptop variety (shown in Figure 4–2b) or the even smaller notebook style. These small computers add to the flexibility of using a computer as a programmer.

Using Personal Computers for Programming—Advantages A major advantage of using a personal computer for programming is the ability of the personal computer to store the program on floppy disk and/or on the hard disk. If for some reason the program is lost, the restoration of the program is simple. Merely copy the program from disk to the processor memory.

When the software is updated by the manufacturer to provide additional features, the update is easily accomplished by loading the new software program onto the computer hard disk by way of the floppy disk drive. A dedicated desktop programmer requires changes in ROM chips to accomplish updates.

When troubleshooting, the software that is used for programming a PLC from a personal computer has the added feature of being able to display Rungs of logic in any order that the electrician or technician may find helpful. On a large program, for example, Rungs 4, 45, 46, 67, and 75 are displayed on the screen at one time. This feature, while available on a few dedicated desktop programmers, is standard on most software programs that have been developed for the various PLCs on the market today.

While dedicated desktop programmers can be networked with multiple PLC processors, they typically do not have the flexibility and networking capabilities of a personal computer.

Using Personal Computers for Programming—Disadvantages One disadvantage of using a personal computer for programming is that the programming is not as user- or electrician-friendly as a dedicated desktop programmer with its symbol keys. It's not that the programming is hard using a computer; it just takes longer to learn how the programming is done. In other words, the learning curve is longer. Another, and probably the biggest disadvantage of using a personal computer as a programmer, is that the computer is not designed for the industrial environment. The personal computer is affected by electrical noise, and high and low temperatures as well as high humidity, conditions that dedicated desktop programmers are designed to handle.

The ideal programmer, then, appears to be a mix of the dedicated desktop programmer and the personal computer. That is exactly what a number of the PLC manufacturers thought, so they developed industrial-rated computer programmers. Allen-Bradley, Siemens T.I., Modicon Inc., and GE-Fanuc Automation are several of the companies that offer these highbred programmers. Figure 4–9 shows the Modicon Inc. P230 programming panel.

Figure 4–9 Modicon Inc. P230 Industrial Programming Panel
(Courtesy of Modicon Inc.)

The P230 is a 25 Megahertz (Mhz) computer with a 120MB self-parking hard drive. Self-parking means that the computer automatically parks the hard drive each time it is shut off. Parking the hard drive prevents damage during transportation of the computer. The P230 comes with two serial communication ports, one parallel port, and a VGA monitor port.

As might be expected, this type of programmer is more expensive than the dedicated desktop programmer or a personal computer. For the money, though, one gets the best features of both. Is the

added cost justifiable? That question can only be answered after all the variables have been considered. The variables could include, but not be limited to:

- Environment.
- Size of program.
- Graphic requirements.
- Requirements for other software, i.e., spreadsheets, word processing.
- Programmability.
- Networking ability.

A final review of some of the advantages and disadvantages of each type of programmer follows.

DEDICATED DESKTOP PROGRAMMERS

Advantages

- VDT to display multiple rungs of the program.
- Ease of programming, user-friendly.
- Designed for industrial use.
- Portability.
- Status of circuit highlighted (intensified or reverse video).

Disadvantages

- Relative high cost.
- Limited models of PLC that can be programmed.
- Limited documentation and graphics capabilities.
- Physical size.

HAND-HELD PROGRAMMERS

Advantages

- Low cost.
- Small size.
- Very portable.

Disadvantages

- Can only view limited Rungs or instructions.
- Fewer functions.
- Limited access to memory.
- Limited or no documentation capabilities.
- Works with only a specific PLC model.
- More difficult to program than a dedicated desktop programmer.
- Uses logic status lights instead of intensified or reverse video.

PERSONAL COMPUTER PROGRAMMERS

Advantages

- VDT to display multiple Rungs of logic.
- Can program any type of PLC (with appropriate software).

- Extensive documentation capabilities; Rung comments, labels, and so forth.
- Cut and paste editing capabilities.
- Low cost when compared to a dedicated desktop programmer.
- Small and portable when laptop or notebook type are used.
- Easy to make copies of program on floppy disk or hard drive.
- Software can be upgraded without changes in hardware.
- View multiple Rungs in any numeric order for troubleshooting.

Disadvantages

- Must operate in normal environment, not industrial rated.
- Initial software costs.
- Software updates or license renewal costs.
- More difficult to program, longer learning curve.
- May require special communication card.

In the final analysis, if money is not a consideration, or the specific application warrants the difference in cost, the industrial computer programmer is the most versatile and serviceable of all the types of programmers that are available.

Chapter Summary

The programming device, or programmer, is used to enter, modify, and monitor the user program. Which type of programming device to use will vary with each application, and what is appropriate for one application may not be the most appropriate for another. The program (ladder diagram) is entered by pushing keys on the keyboard in a subscribed sequence, with the resultant circuit(s) displayed on the VDT of a desktop programmer, and with a LED or liquid crystal display for the hand-held programmer. The visual display can be used as an aid to test the circuit prior to entry into user memory, or for troubleshooting after the circuit is entered into memory and is operational. Contacts and coils are either intensified or displayed in reverse video to indicate power flow or logic continuity. Programming the PLC is not difficult, but time must be spent to become familiar with the specific PLC and its programming techniques.

Review Questions

1. Briefly describe the function of the *programming device*.
2. What does the acronym VDT stand for?
3. Define the term *on line*.
4. List the three type of programming devices and give advantages and disadvantages for each.
5. Define the term *off line*.
6. What is a hardware key?
7. What is meant by the term "third party software"?
8. What is the advantage of a sealed touch pad-type keyboard over a standard raised key keyboard?
9. In the context of this chapter, what do the terms *passing current* and *power flow* indicate?
10. Hand-held programmers usually have what type(s) of displays?

CHAPTER

5 Memory Organization

Objectives

After completing this chapter, you should have the knowledge to:

- Identify the two broad categories of memory and describe the function of each.
- Identify the types of information stored in each category of memory.
- Define the term *byte*.
- Define the acronym *bits*.
- Define *holding registers*.
- Understand the term *default*.

MEMORY WORDS AND WORD LOCATIONS

For the programmable controller to function properly and control a process or driven equipment, it must be able to perform the user program repeatedly and accurately. The system must also be able to perform its control function with great speed, which is achieved by processing all information in binary signals. The key to the speed with which binary information can be processed is that there are only two states, each of which is distinctly different. Binary signals fall into one of two states, which are 1 and 0. The 1 and 0 can represent *ON* or *OFF*, true or false, voltage or no voltage, high or low, or any other two conditions depending on the system. There is no in-between state or condition, and when information is processed, the decision is either *yes* or *no*. There is no *maybe*, *almost*, or any other alternative.

As indicated in Chapter 3, the processor memory consists of hundreds or thousands of locations that are referred to as words. Each word is capable of storing binary data in the form of binary digits, or **bits** (**BI**nary digi**TS**). A binary digit, like a binary signal, can only be a 1 or a 0. The number of bits that a word can store will depend on the system or PLC. Words can be made up of 32 bits, 16 bits, or 8 bits. The 16-bit word is the most common (Figure 5–1).

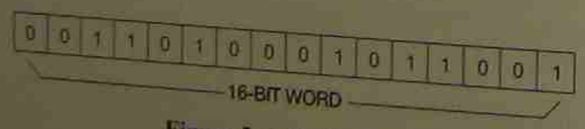


Figure 5–1 16-Bit Word

If a memory size is 256 words, then it can actually store 4,096 bits of information using 16-bit words (256 words × 16 bits per word) or 2,048 bits using an 8-bit word (256 words × 8 bits per word). When comparing memory sizes of different PLC systems, it is important to know the number of bits per word of memory. Bits can also be grouped within a word into **bytes**. A byte is a group of 8 bits.

So that information stored in each word can be located, each word is numbered or given an address. Addressing words in the memory serves the same function as the addresses used for homes or apartments. Word 100, for example, represents a specific word location in memory, just like N. 100 Lincoln represents the address of an apartment building. The bits in word 100 are found by referencing a given bit number, just like the occupant of the apartment complex is found by a given apartment number.

Since a bit of information can only be a 1 or a 0 (*ON* or *OFF*), how is the status of bits within a word determined? Words that store the status of individual bits for input devices are set to 1 (*ON*) or 0 (*OFF*), depending on the status (*ON* or *OFF*) of the input devices that the bit locations represent. Other bits are set to 1 or cleared to 0 by the processor in response to the logic of the user program, RELAY LADDER LOGIC, or special instructions, which, in turn, control the status (*ON* or *OFF*) of other bits that represent output devices.

A simple example of how this works is illustrated in Figure 5–2.

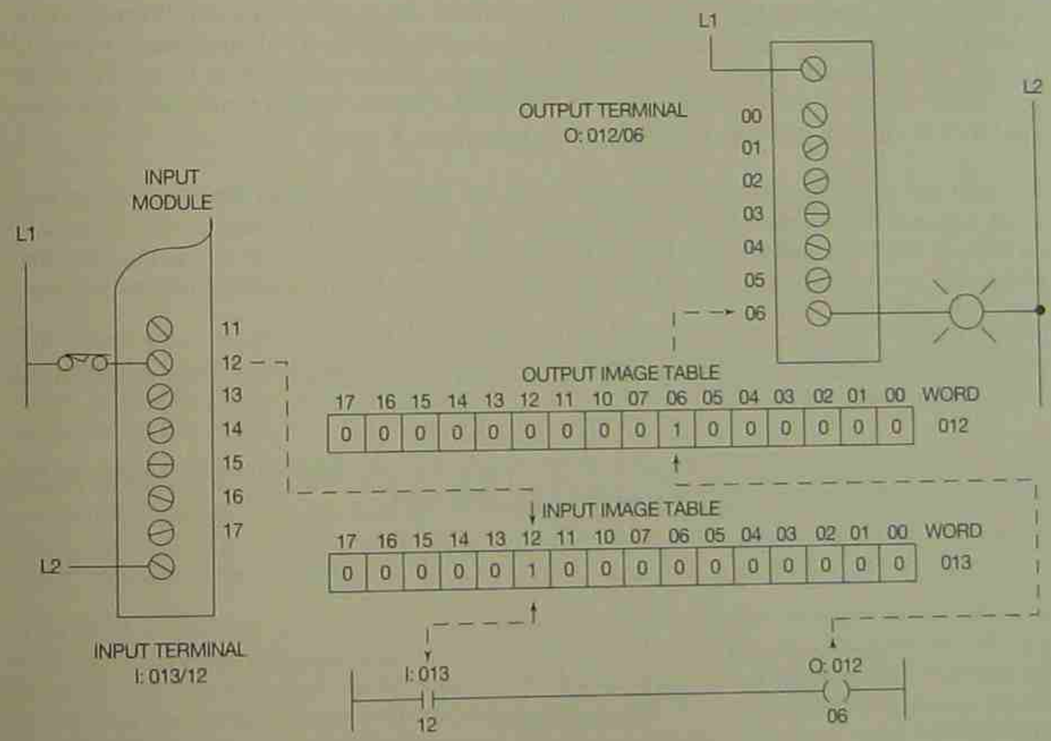


Figure 5–2 Relationship of Bit Address to Input and Output Devices

Note: *The example uses memory organization and addressing utilized by the Allen-Bradley PLC-5 family. While the example is specific to Allen-Bradley, the concepts illustrated are common to all PLCs. Allen-Bradley uses an octal numbering system to address bit locations. Notice that the 16 bits are numbered 00 through 07 and 10 through 17. In the octal numbering system, the numbers 8 and 9 are never used. The octal numbering system will be covered in detail in Chapter 6.*

Assume that when a given limit switch is closed, the closure will turn an indicator lamp *ON*. The limit switch is connected to an input module in the I/O rack, while the indicator lamp is connected to an output module. Chapter 2 discussed DIP switches that were set in a prescribed sequence to identify the I/O rack number for the processor, and that the location of each terminal point of each I/O module within the rack determined the address of a given device. In Figure 5–2, the limit switch is connected to terminal 12 on an input module, and is given an address of I:013/12. This indicates that bit 12 of input image table word 013 stores the status (*ON*-[1] or *OFF*-[0]) of the limit switch. The indicator lamp is connected to terminal 06 of an output module and is given an address of O:012/06. This address indicates that bit 06 of output image table word 012 controls the status (*ON*-1 or *OFF*-0) of the lamp.

By programming a simple circuit into the user memory of the processor as shown at the bottom of Figure 5–2, the processor controls the indicator lamp using the logic of the user program. The logic states that if contact I:013/12 closes, lamp O:012/06 should light, or go *ON*. When power is applied to the processor, the processor starts its scan and looks at bit 12 of input image word 013 to see if the bit is set to 1 or 0. If the limit switch is open, the bit will be set to 0, or *OFF*. If the limit switch is closed, as indicated in Figure 5–2, the input module sends a signal to the processor, and bit 12 of input image word 013 will be set to 1, or *ON*.

The next part of the scan solves the user program. The logic of the ladder diagram, or user program, indicates that when contact I:013/12 (bit 12 of input word 013) is closed, or *ON*, the indicator lamp O:012/06 should be turned *ON*. The processor reads the logic, and during the third step of the scan, sets bit 06 of output word 012 to 1, which turns the lamp connected to the output module terminal 06 *ON*.

The address I:013/12 also tells us that the limit switch is an input device, and is wired to terminal 12 of module group 3 of rack 1.

Figure 5–3 illustrates the significance of each letter/digit or group of digits used for addressing the Allen-Bradley PLC-5 family of programmable logic controllers.

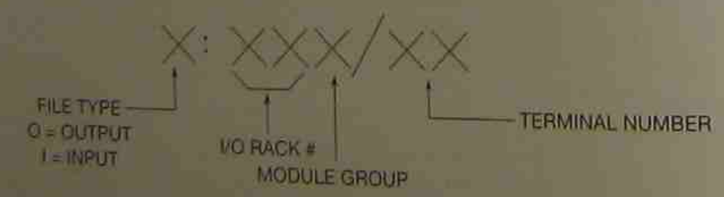


Figure 5–3 Allen-Bradley PLC-5 Address Format

The first letter is used to indicate the type of file (input, output, timer, counter, etc.) The letter I represents an input device, and the letter O represents an output.

The next two digits identify the rack number. One rack can control 128 I/O points. Rack numbers start at 00. A rack is different than a chassis. The chassis is the physical frame that actually holds the input and output modules that make up a rack. Depending on the density of the I/O modules used, a rack may require a 16-slot chassis or only 4 slots. If 8-point I/O are being used, it will take 16 slots of 8-point I/O modules to make 128 I/O points ($8 \times 16 = 128$), whereas if 32 point I/O are being used, it will only take a 4-slot chassis to make a rack ($32 \times 4 = 128$).

The next number identifies the module group within the rack. This is always a number from 0 through 7. The last two digits identify the actual terminal number to which the device is wired.

Figure 5–4 reviews the concept using the address of the limit switch I:013/12.

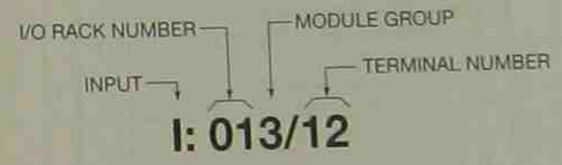


Figure 5–4 Limit Switch Address I:013/12

The letter I tells us that the address represents an input device. The next two digits, 01, tell us that the device is located in I/O rack number 01. The next digit, which is a 3, further identifies the location as module group number 3. The last two digits, 1 and 2, identify the actual terminal (12) on the input module to which the limit switch is connected.

Another example of this concept is shown in Figure 5–5. The limit switch address I:013/12 gives us a hardware location for an input device in rack 01, module group 3, terminal 12. This same address, I:013/12, tells us that the status (*ON* or *OFF*) or state of the limit switch is reflected by bit 12 of word 013 in the input image table.

This same addressing scheme gives us a hardware location for the indicator lamp addressed O:012/06. The letter O indicates an output device. The next two digits, 01, tell us that the I/O rack location is 01. The next digit identifies the module group as group 2. The last two digits locate terminal 06 as the terminal on the output module to which the indicator lamp is wired. Again, the address O:012/06 also locates the memory word and bit location that reflects the status (*ON* or *OFF*) of the indicator lamp as shown in Figure 5–6.

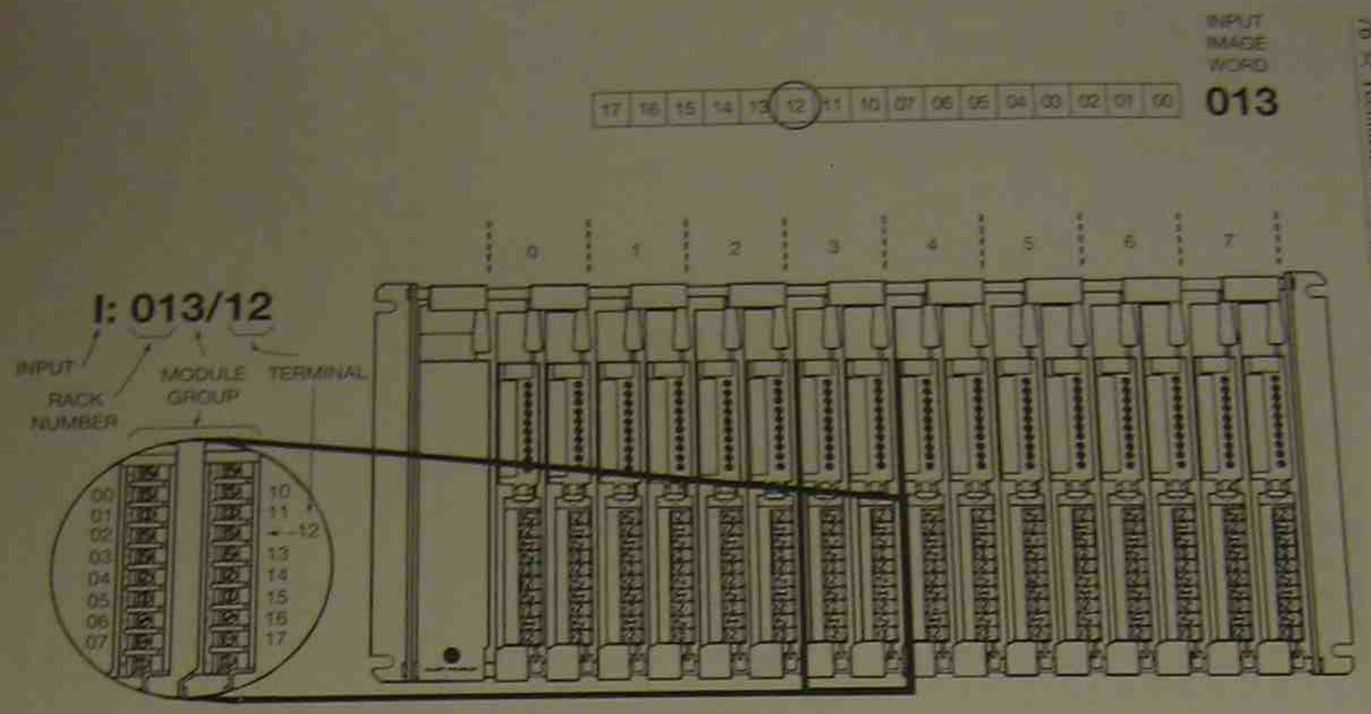


Figure 5–5 Relating Input Address I:013/12 to Actual Hardware Location
(Courtesy of Allen-Bradley)

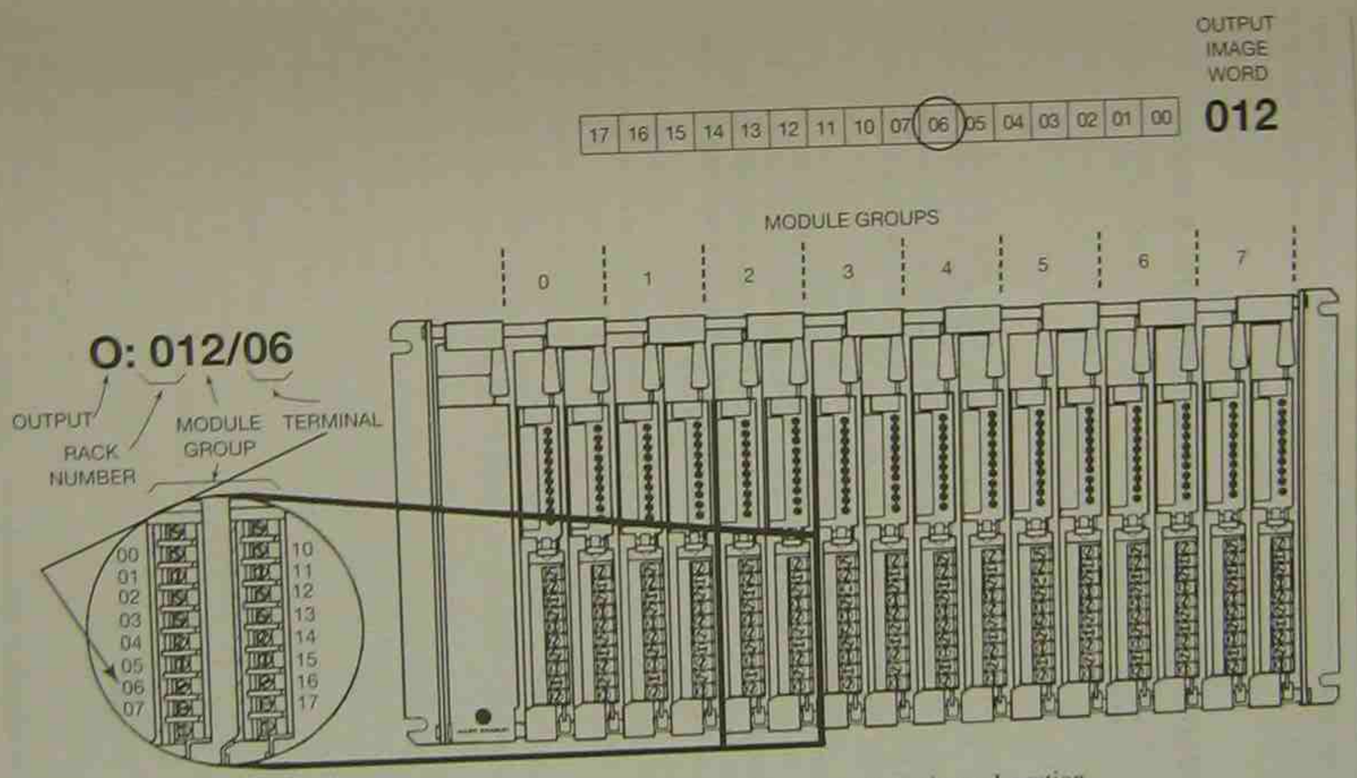


Figure 5–6 Relating Output Address O:012/06 to Actual Hardware Location
(Courtesy of Allen-Bradley)

While the address system discussed is specific to the Allen-Bradley PLC-5 family, most PLC manufacturers use an addressing scheme that identifies memory word locations, and may also give hardware locations.

SLC 500 AND MICROLOGIX 1000 AND 1500 ADDRESSING SCHEME

Like the PLC-5 family, the SLC 500 and MicroLogix family use the letter I for an input address and the letter O for an output address.

I = external input device
O = external output device

A typical address for an input device would be I:0.1/6. The I, of course, indicates that this is the address for an input device. The colon (:) is called an element delimiter. This means that the colon separates the input designator, I, from the rest of the address. The next character, the number 0, indicates the slot number that holds the actual input module. The slot number can range from slot 0, adjacent to the power supply in the first chassis, to a maximum of 30.

If the number of inputs is more than 16, a period (.), which is called a word delimiter, would be used after the slot number. The forward slash (/) is called the bit delimiter. The number that follows the forward slash is the bit number of the word as well as the terminal number of the I/O module. The digit 6 indicates the terminal number where the input device is wired. Figure 5–7 illustrates the addressing scheme for the SLC 500 and the MicroLogix family.

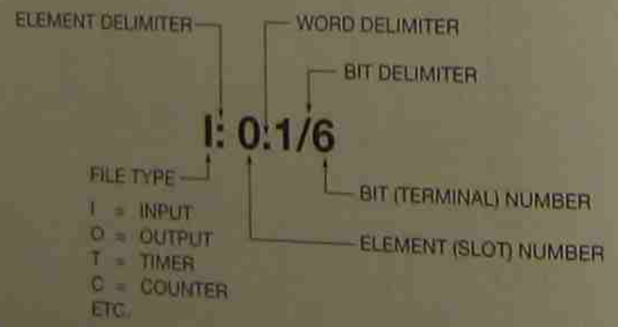


Figure 5–7 Allen-Bradley SLC 500 and MicroLogix Addressing Scheme

The first character of the address identifies the file type. The typical file types used are:

O	Output
I	Input
S	Status
B	Bit
T	Timer
C	Counter
R	Control
N*	Integer
F*	Float
St*	String
A*	ASCII

*Only available on selected processors.

In the address shown in Figure 5–7 the I indicates that the file type is an Input file and also indicates this is the address of an input device. The 0 after the :, which is the element delimiter, indicates that the input device is connected to slot 0. The period (.) after the slot number indicates that the inputs exceed 16 and require two words in the input image table. The number 1 indicates that this is word 1 in slot 0. The number 6 after the forward slash is the bit number.

Note: *The SLC 500 and MicroLogix do not use the octal numbering system like the PLC-5 family, but instead use the decimal numbering system.*

For 16 inputs, the bits are numbered 0–15. The address, by using the period word delimiter (.), indicates that there are more than 16 inputs installed in slot 0. Bit 6 is actually input device 22 (16 + 6 = 22). Because we started with bit 0, input device 22 is in reality the 23rd input device. Likewise, input address I:0.1/7 is the 24th input device.

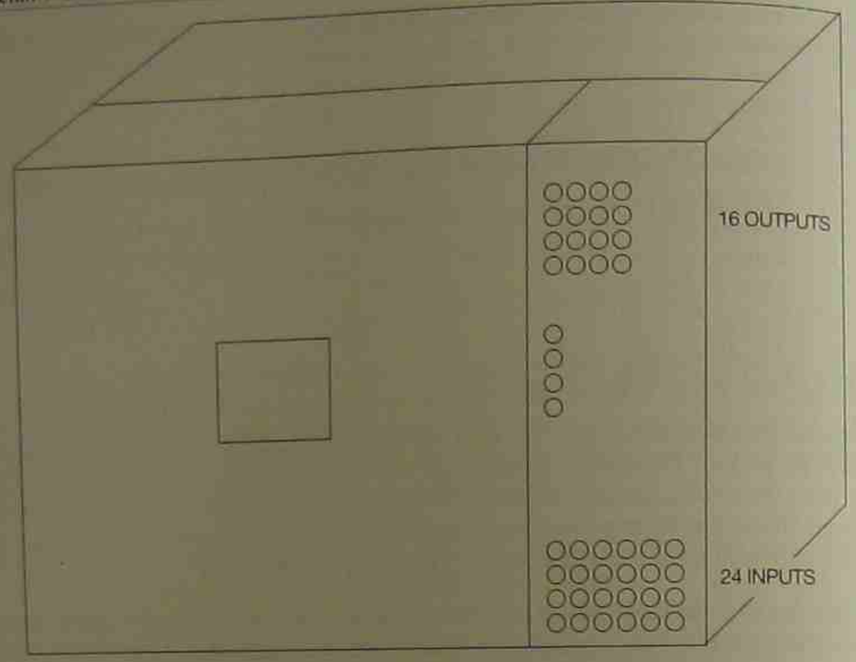
The SLC 500 controller is available in either fixed or modular I/O. The fixed I/O units have fixed I/O of 20 (12 inputs and 8 outputs), 30 (18 inputs and 12 outputs), and 40 (24 inputs and 16 outputs).

For fixed I/O controllers, all of the I/O are in slot 0. Figure 5–8 shows a fixed I/O controller that has 24 inputs and 16 outputs. Also shown are the input and output image tables for Data File 0 (output) and Data File 1 (input). Note that for 24 inputs, the Input Image table uses two words for slot 0. All 16 bits of word 0 (I:0) are used, whereas only the first 8 bits of word 1 (I:0.1) are used (bits 0 through 7). The unused bits of word 1—bits 8 through 15—are marked invalid and are not available for use.

An output address of O:0/4 indicates that this is the address of an output device in slot 0, terminal 4. This address also indicates that this is bit 4 of word O:0 in the Output Image table data file 0 as indicated by the *X* under bit 4 of output word O:0 (Figure 5–8).

An input address of I:0/15 indicates controller input 15 in slot 0. This address is indicated by the *X* placed under bit 15 of input word I:0 in the input image table (Figure 5–8).

An address of I:0.1/7 indicates an input device in slot 0, word 1, bit 7. This address is indicated by the *X* placed under bit 7 of input image Data File 1, word 1 in Figure 5–8. This address also indicates that this is input device 23. As we started with 0 as the first input device, the 23rd device is



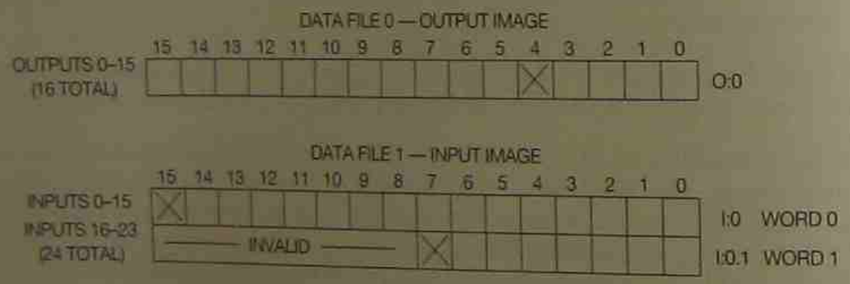


Figure 5–8 SLC 500 with 40 Fixed I/O

actually the 24th and last input device. The first word of the input image table, word I:0, held input device information for inputs 1 through 16, which are bits 0 through 15. The second word, word I:0.1, of the input image table holds the information for input devices 17 through 24, which are represented by bits 0 through 7. Bit 0 of word I:0.1 of Data File 1 represents input device 17. An alternate way to address input terminal 23 of slot 0 would be I:0/23.

Figure 5–9 shows an SLC 500 modular controller that consists of a seven-slot chassis interconnected to a ten-slot chassis.

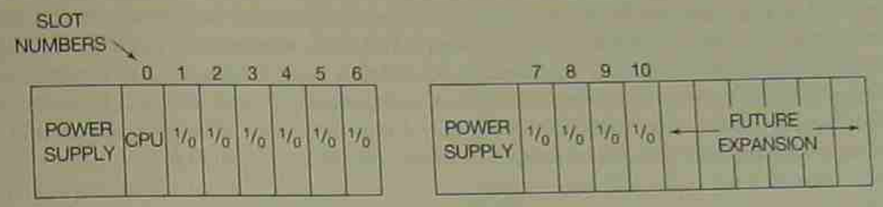


Figure 5–9 SLC 500 Modular Controller (Seven-Slot Chassis Connected to a Ten-Slot Chassis)

In this configuration, slot 0 contains the CPU, so slot 0 becomes an invalid I/O slot number. The controller is configured as follows:

SLOT	INPUTS	OUTPUTS
1	6	6
2	32	None
3	None	16
4	8	8
5	None	32
6	16	None
7	16	None
8	8	None
9	None	16
10	None	16

Figure 5–10 shows Data File 0 (the output image table) and Data File 1 (the input image table) as it would appear for the mix of I/O described above. Note that wherever 32-point I/O modules are used, the data table will require two 16-bit words. Slot 2 has a 32-point input module, and the input image table shows word I:2 (word 0 for inputs 0 through 15) and word I:2.1 (word 1 for inputs 0–15 which are actually inputs 17 through 32). Likewise, where 32 outputs are used in slot 5, the output image table for slot 5 shows that two words are used: O:5 and O:5.1.

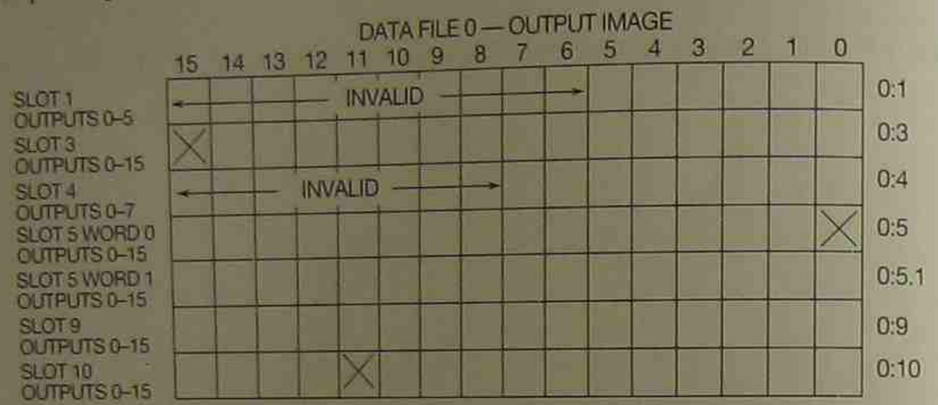


Figure 5–10 Output and Input Image Tables for the SLC 500 Configuration Shown in Figure 5–9

For the modular system shown in Figure 5–9, an address of O:3/15 indicates an output device 15 in slot 3. It also indicates bit 15 of word O:3 of the Output Image table, shown by an *X* under bit 15 of output word O:3 in Figure 5–10. An address of I:2.1/3 indicates input device 3 of word 1 in slot 2 (as indicated by an X under bit 3 of input word I:2.1). Slot 2 holds a 32-point input module which requires two words in the input image table: Word 0 (I:2) and Word 1 (I:2.1) shown in Figure 5–10. To further clarify the addressing scheme, address I:7/8 indicates input number 8 in slot 7 as indicated by the *X* under location 8 of slot 7 which is word I:7 of the input image table.

Note that slot 4 of both the input and output image tables has 8 inputs and 8 outputs. While only 8-bits of each word in the input and output image table are needed, the unused bits cannot be used for any other programming as indicated by the word *invalid* on the image tables.

Figure 5–11 shows an SLC 500 seven-slot chassis connected to a four-slot chassis using an Allen-Bradley 1746-C9 communications cable. The processor is installed in slot 0 of the first chassis which is adjacent to the power supply. Figure 5–11 shows an exploded view of the output/input module installed in slot 3. The output/input module is shown with the door that covers the terminals open. The inside of the door shows a pictorial view of the terminal layout. The first vertical row of terminal screws are for the six outputs plus an AC common connection. The second vertical row of terminals is for the six input devices and an AC common connection. The normally open pushbutton that is shown would have an address of I:3/0. I is for input, the 3 indicates that the input device is installed in slot 3, and the 0 indicates the input device is connected to input terminal 0.

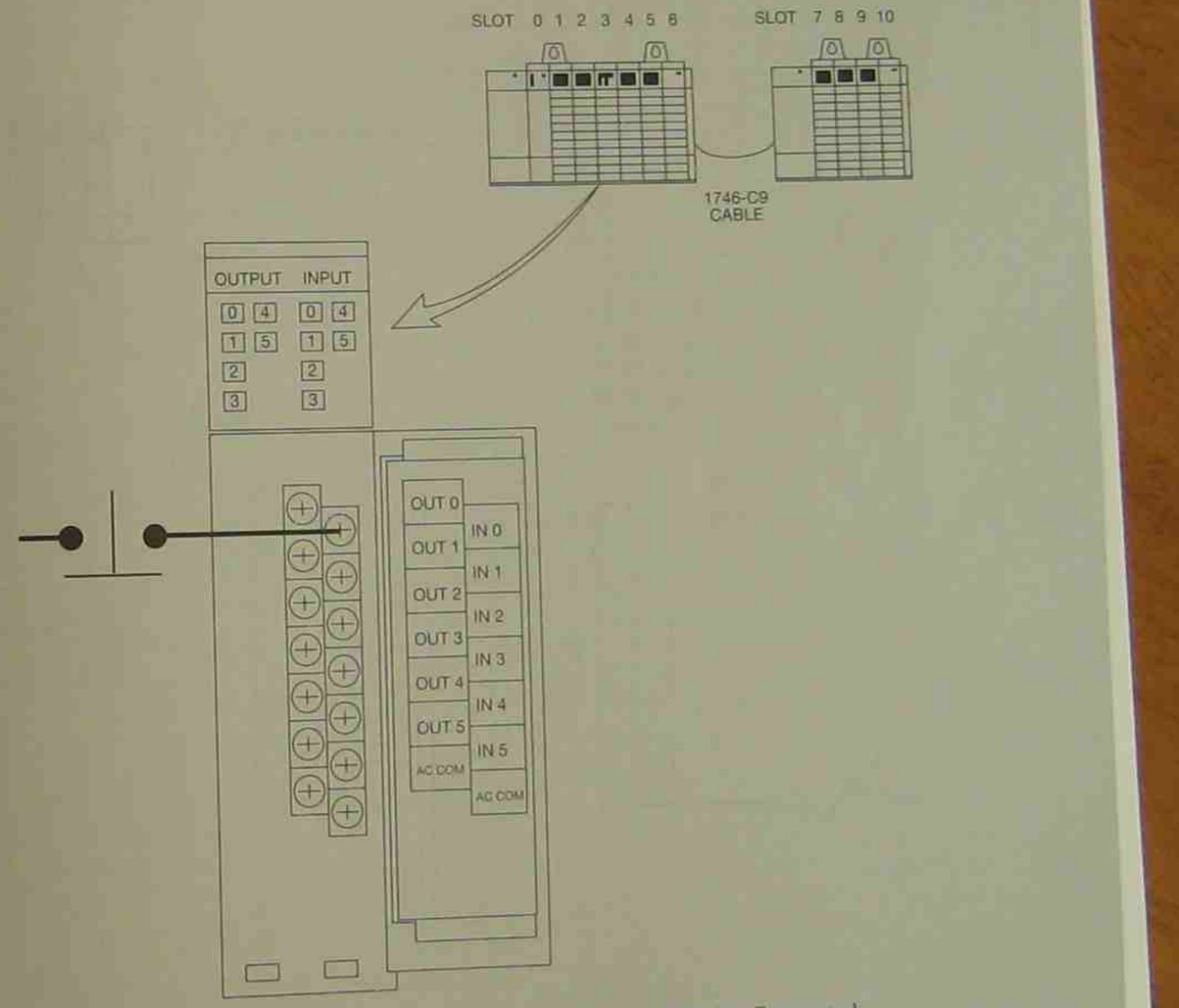


Figure 5–11 SLC 500 Modular Controller with Input Device Connected

Figure 5–12 shows the same seven-slot chassis connected to a four-slot chassis. In this figure, an output device (solenoid) is connected to one terminal of a 16-point output module that is installed in slot 8. As before, the terminal door is open and shows the layout of the terminals. The first terminal is for connecting the AC power. Notice that the output numbers alternate from Out 0, then Out 1, Out 2, and so on. The very last terminal is for the AC common, or neutral connection. The address for the solenoid connected as shown would be O:8/12. The O indicates an output device, the 8 indicates that the module is installed in slot 8, and the 12 indicates that the solenoid is connected to output terminal 12. This address also indicates that the status of the output device is found in bit 12 of Output Image Table word 8.

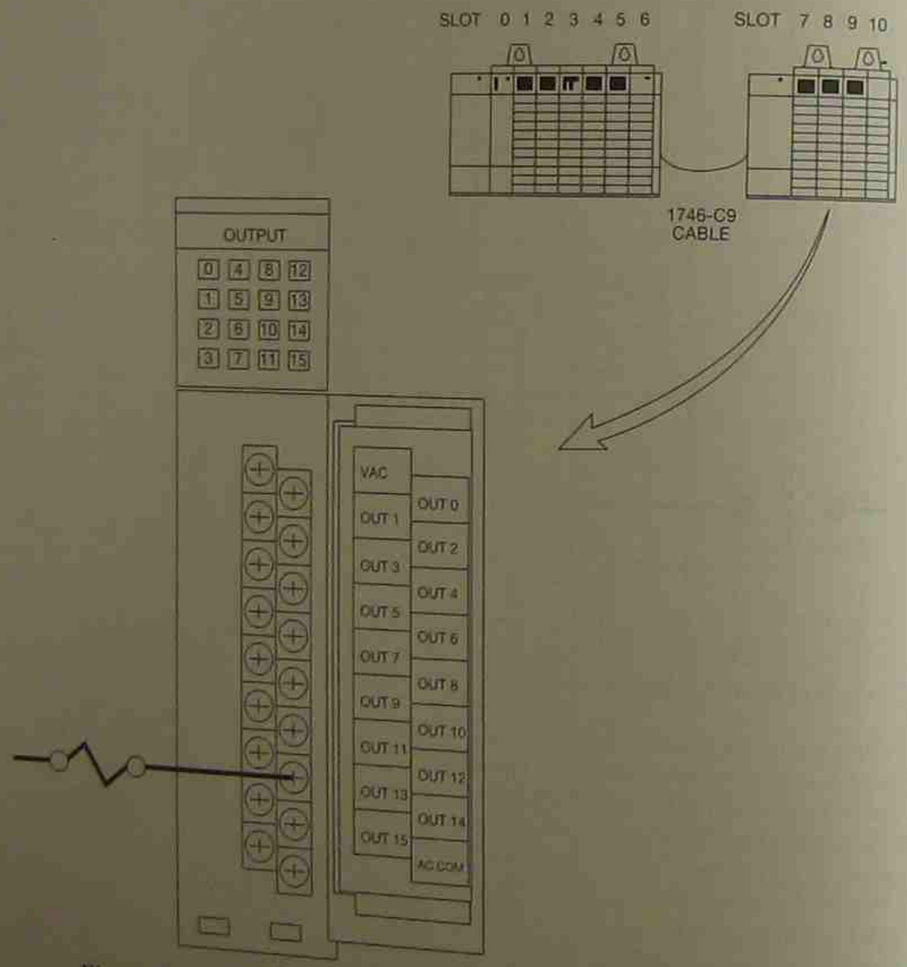


Figure 5–12 SLC 500 Modular Controller with Output Device Connected

Figure 5–13 shows an analog input module installed in slot 10. The exploded view of the module shows a sensor connected to the module at terminals 3 and 4. As connected, the address of the analog sensor is I:10/1. From the layout on the terminal door, we can see that terminal 3 and 4 are identified as Input 1+ and Input 1–.

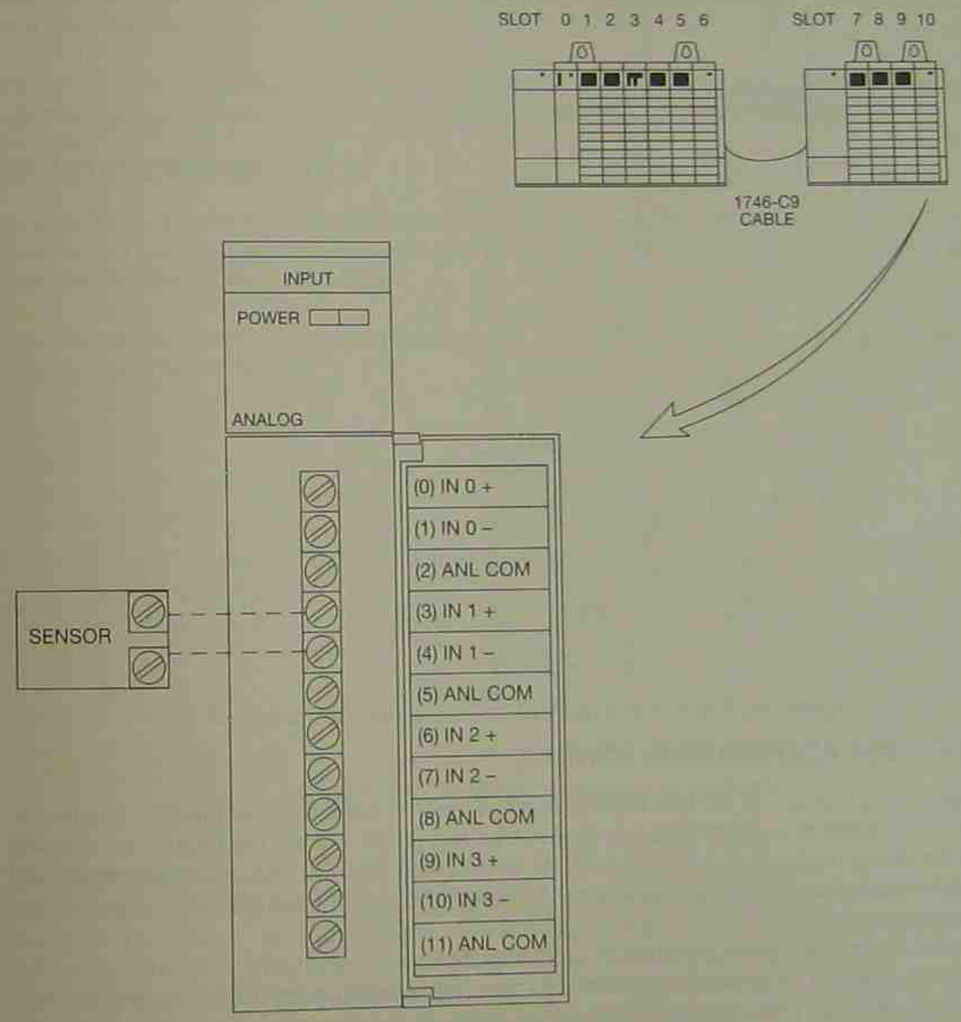


Figure 5–13 SLC 500 with Analog Input Device Connected

Figure 5–14 shows an analog output module installed in slot 6. The analog output in this illustration is an actuator connected to terminals 6 and 7. The address for the actuator is O:6/3.

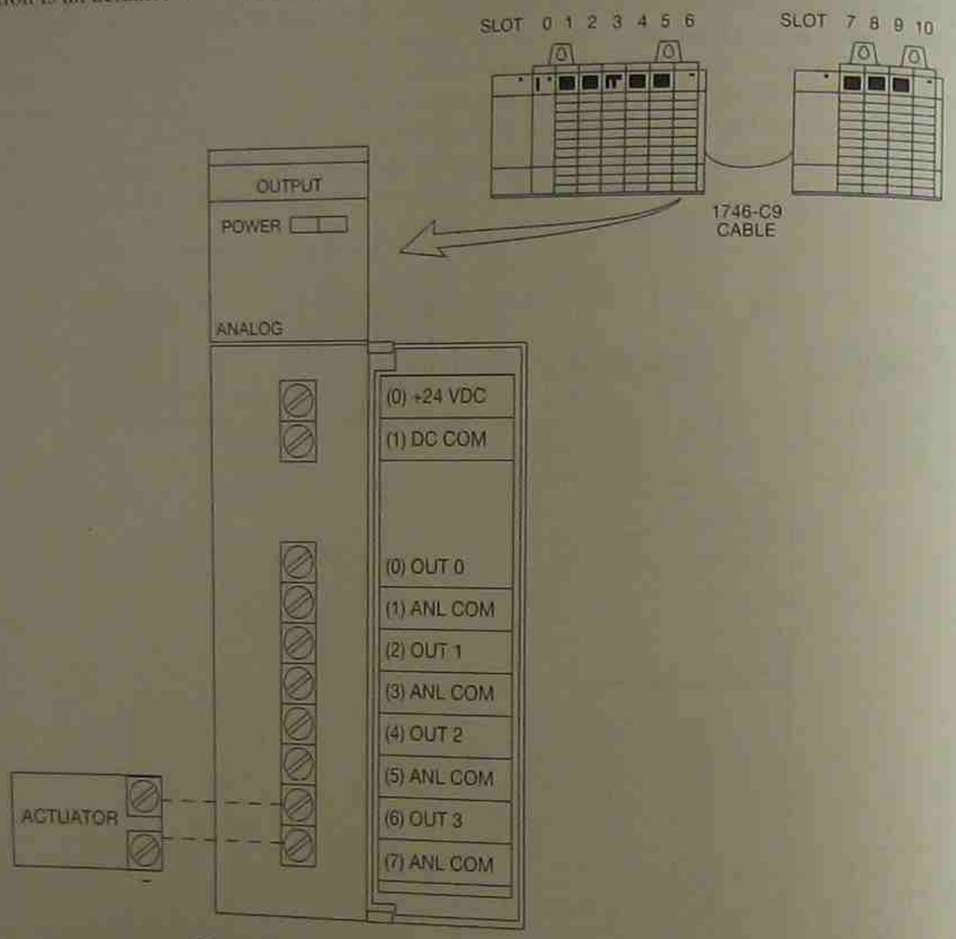


Figure 5–14 SLC 500 with Analog Output Device Connected

MODICON 984 ADDRESSING SCHEME

The Modicon 984 uses a 1 as the first number in an address for a discrete input device, and uses a 0 for the first number of a discrete output device. The rest of the address information is assigned by the programmer using a software utility called the I/O Map. The I/O Map is used to assign reference addresses to specific I/O modules, slots, and rack locations. By using the I/O Map software utility, the programmer defines:

- What slots in a given rack contain I/O modules.
- What type of I/O module is expected to be in a given slot, as opposed to what is actually there, as with the Allen-Bradley PLC-5 family.
- What addresses are to be assigned to which modules.

Instead of being hardware-driven like the Allen-Bradley addressing scheme, the 984 is software-driven and programmer-defined.

Discrete *input device* addresses use a five-digit number beginning with a 1 (1XXXX). The remaining four numbers that will make up the address are assigned by the programmer based on how the system is configured. Input registers are used for analog devices, absolute encoders, thermocouples, and the like, and these addresses will start with the number 3 (3XXXX).

Discrete *output devices* begin with the number 0 (0XXXX). For analog output devices such as meters, transducers, and variable speed drive motors, output registers are used and have addresses that start with the number 4 (4XXXX).

Once you gain experience using the I/O Map utilities portion of the Modicon programming software to assign addresses, the process becomes quite easy and very understandable.

MEMORY ORGANIZATION

At this point, it should be no surprise to find out that not all PLC manufacturers have organized their memories in the same way, or that they do not all use the same terminology for the configuration or make-up of their memories.

As discussed in Chapter 3, there are two general classifications of memory: storage memory and user memory (Figure 5–15).

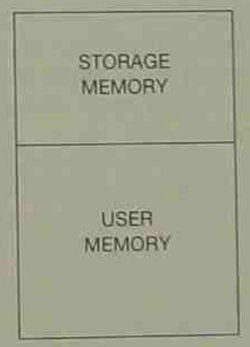


Figure 5–15 Two Broad Categories of Memory

Storage Memory

Storage memory is that portion of memory that will store information on the status of input and output devices, preset and accumulated values of timers and counters, internal relay equivalents, numerical values for arithmetic functions, and so on. The entire storage memory is called a data table, a register table, or other names, depending on the PLC manufacturer. A register is defined as an area for storing information (logic or numeric). Although the names or titles which are given to sections or subsections of the storage memory vary, the principles involved do not.

For example, the section of the memory that stores the status of the real-world input devices may be referred to as an input image table, input register, input status table, or external input section. No matter what name is used, the information is stored in the same way. The status (*ON* or *OFF*) of each input device is stored as either a 1 or a 0 (*ON* or *OFF*) in one bit of a memory word. When the processor is executing the user program (ladder diagram), it scans the input device status stored in the storage memory to determine which inputs are *ON* or *OFF*.

The section of storage memory set aside for output status may be referred to as the output image table, output register, output status table or external output section. Again, the name does not change the function of this section of the storage memory, or the method by which information is placed in memory for control of the actual output devices. As the processor executes the user program, it sends binary data (1s or 0s) to the output section of memory to control the output devices. Each output device is represented by one bit of a memory word.

Numeric information for timer or counter preset and accumulated values, arithmetic functions, sequencer functions, data manipulation, etc., uses a part of the storage memory that is called data registers or internal storage. Information is entered and stored in this part of memory using the binary, BCD, or hexadecimal numbering systems (the various numbering systems are covered in Chapter 6). The numbering system(s) used depend on the PLC hardware and system requirements. The storage of numeric information requires that several bits be used of one word to represent numbers. In a practical sense, any word used to store numerical information is not available for additional storage, even if all the bits of word are not used.

In general, any unused bits of a word that are not used for storing numeric values can be used as internal relays. Internal relays will replace the numerous control relays used in most hardwired control circuits. Many PLCs have a portion of memory set aside just for internal relays. Internal, or dummy relays, may also be programmed in the output section of memory when all the words or bits of words are not being used for real-world output devices. The concept and use of internal, or dummy relays, is covered later in the text.

Figure 5–16 shows the memory organization for an Allen-Bradley Mini PLC-2/15. The 2/15 is part of the PLC-2 family of Allen-Bradley processors. Allen-Bradley also has the newer PLC-5 family of processors.

Note: *While the PLC-2 family is the older version, there are literally thousands of this type still in use in industry today, and an understanding of their memory structure is meaningful.*

For the PLC-2 processors, Allen-Bradley uses 16-bit words. The word and bit addresses are numbered using the octal numbering system, as shown in Figure 5–11. The first eight words (000–007) of the data table are set aside for the processor. Words 010–017 and 020–026 are the output image table. An output device addressed 01000 is bit 00 of word 010; similarly, an address of 02617 is bit 17 of word 026. Note; word 027 is used by the processor for a *BAT LOW* condition, message generation, and data highway. The next 40-word section of memory, words 030–037, 040–047, 050–057, 060–067, and 070–077, is used for storing timer or counter accumulated values (numeric) or for internal storage.

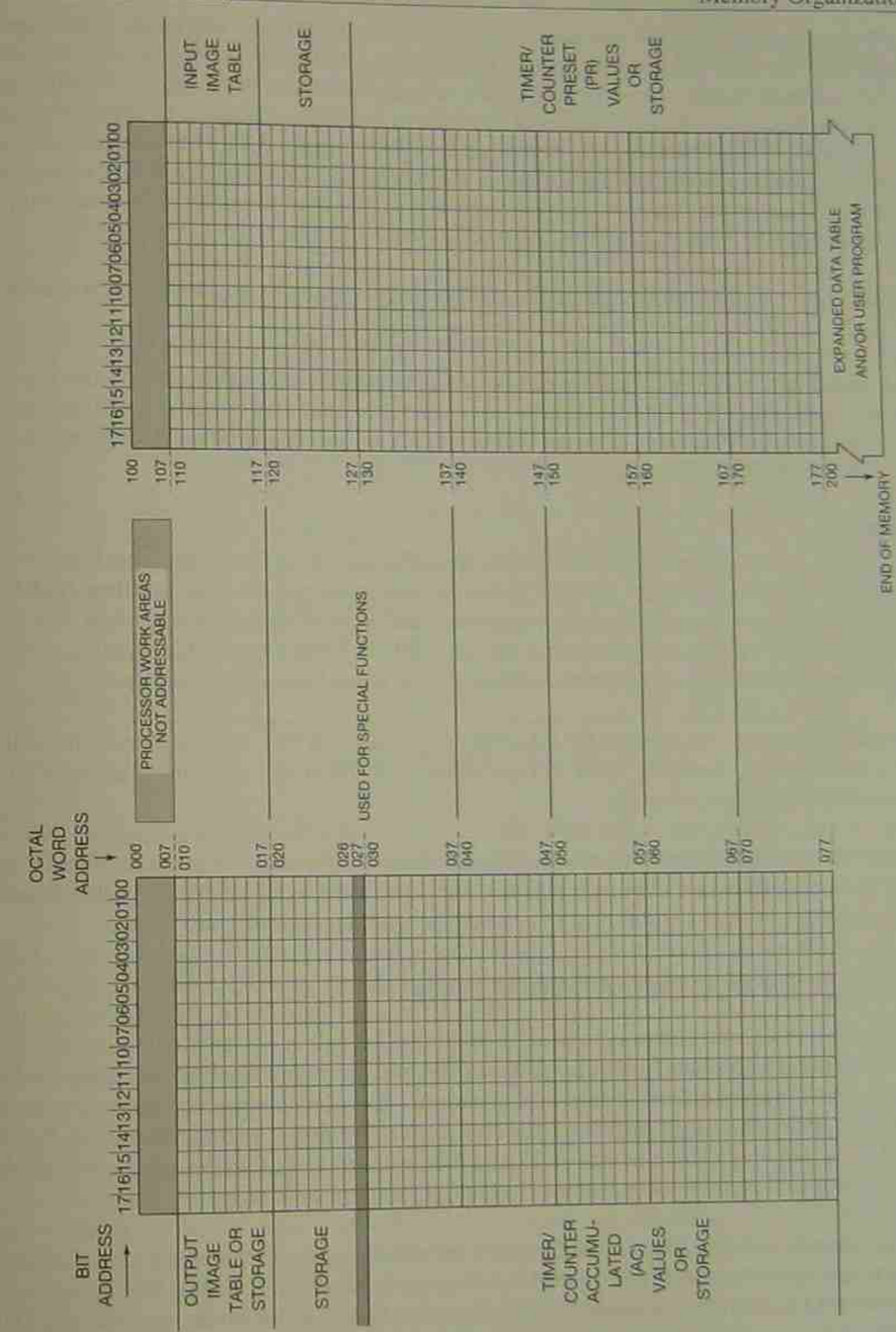


Figure 5–16 First 128 Words of an Allen-Bradley PLC-2/15 Memory

The next 8 words are the second processor work area. Words 110–127 (16 words) make up the input image table for discrete input addresses. Input address 12406 is bit 06 of word 124. Words 125 and 126 are used by the processor to indicate remote I/O faults. Words 130–177 (40 words) are used to store the preset values of timers or counters or for internal storage. If word 030 is used for a timer, word 130 automatically stores the preset value of timer 030. The accumulated value for timer 030 is stored in word 030. Any timer or counter in an Allen-Bradley PLC-2 system will store the preset value in the word that is numbered 100 higher than the timer or counter number. For example, counter 047 has its preset value stored in word 147.

Below the timer or counter preset values and internal storage section is the user program section of memory.

The actual configuration of the data table can be changed to meet user needs. Input/output image tables can be increased or expanded to handle more discrete inputs and outputs. Additional information on expanding the data table of Allen-Bradley PLCs is available from their local representative or from their technical publications.

User Memory

The user memory, or logic memory as it is sometimes called, is where the programmed ladder diagram is entered and stored. Within the user memory, words are set aside as **holding registers**. Holding registers typically store information generated and used by the processor when it is solving the user program. Holding registers that are set aside to store intermediate values or other short-term bits of information are sometimes referred to as *scratch areas* or *scratch pads*.

The user memory accounts for most of the total memory of a given PLC system. A system with an 8K memory (8192 words) typically has a storage memory of 2K or less, and the balance of memory (6K) is available for user memory.

Once the user program has been entered into the user memory, by either a programming device, tape loader, computer, or by means of a telephone interface, the programmable controller is ready to control the process or driven equipment in accordance with the user program logic.

ALLEN-BRADLEY PLC-5 FILE STRUCTURE

The Allen-Bradley PLC-5 processors are usually programmed with a personal computer and software specific for the PLC-5 family, and the areas of memory are often referred to as files, not tables, as is the case with the PLC-2 memory structure. Although there are still two memory sections (storage [data] and user [program]), the PLC-5 memory map, or structure, is very flexible in the way that the memory can be allocated. Figure 5–17 shows the PLC-5 **default** memory structure. Default refers to the initial value, setting, or configuration prior to any user changes.

In the data or storage memory section file 0 is the output image file. This file has 32 words of 16 bits each, and can hold the status of 512 real-world output devices (32 × 16). The status of the outputs (*ON* or *OFF*) is updated once each scan. On some PLC-5 models, like the PLC-5/25, the size of the file can be increased to accommodate more output devices.

	MAXIMUM NUMBER OF ELEMENTS		FILE NUMBER	DESIGNATION
DATA OR STORAGE MEMORY	32	OUTPUT IMAGE	0	O
	32	INPUT IMAGE	1	I
	32	STATUS (CLOCK FAULT TABLE, ETC.)	2	S
	1000	BIT (INTERNAL RELAYS)	3	B
	100	TIMER	4	T
	1000	COUNTER	5	C
	1000	CONTROL (MATH, SEQUENCER)	6	R
	100	INTEGER (WHOLE NUMBERS)	7	N
	1000	FLOATING POINT (DECIMAL NUMBER, I.E., 3.2)	8	F
USER OR PROGRAM MEMORY		ASSIGN FILE TYPE AS NEEDED	9-999	
		ASCII	0	A
		RESERVED	1	
		MAIN (LADDER LOGIC)	2	
		SUBROUTING	3	
		FAULT ROUTINE		
			999	

Figure 5–17 PLC-5 File Structure

File 1 is the input image file. This file, like file 0, has 32 words of memory and can store the status of 512 input devices. The status (*ON* or *OFF*) of the input devices, like the output image file, is updated once each scan and can be increased in size on many PLC-5 models.

Both files 0 and 1 use the octal numbering system, and the memory locations (bits) are also numbered using the octal numbering system (there are no 8s or 9s in the octal numbering system). The digits are 0–7, 10–17, 20–27, and so forth.

File 2 is the status, or S file. This file is used to store information on general processor status, fault codes, real time clock and calendar, major and minor fault bits, and program scan times in msecs. Information from this file is used or incorporated into the user program. The size of this file changes depending on the processor that is being used.

File 3 is the B, or bit file, and is used primarily for internal or dummy relays. The default size of this file is one word, but can be expanded to 1000 words if needed. All addresses from this file must start with B3. Another B or bit file may be created using the other areas of the memory. A B10 file could be created that would also have internal or dummy relays. The addressing B10 versus B3 is used for organization and ease of identification. The B3 file may be associated with one

piece of equipment, while the B10 file could be associated with another piece of equipment and/or operation. Up to 999 files can be created in a PLC 5/15 processor memory, as long as you do not try to allocate more memory than is physically available in the processor. The B3 file is typical of the remaining files in flexibility, as well as being addressed using the decimal numbering system.

File 4 is the T, or timer file. All timer addresses must start with T4 unless new timer files have been created (i.e., T 9, T 10, and T 11). When creating files, the same number cannot be used twice. If file 10 is used as a B file (B10), then file 10 cannot be used as a timer file. Each timer that is programmed uses three words of memory from its timer file.

File 5 is the C, or counter file. All counters that are programmed have C5 as the start of their addresses. Each counter, like timers, use three words of the counter file memory.

File 6 is the R, or control file. The words in this file are used with special functions like sequencer, file moves, word to file moves, and math functions. File 7 stores whole numbers (integers) and is called the N, or integer file. The integer file is used to store numeric values for data compare, arithmetic functions, and the like. For storing numbers with a decimal point, or floating point, file 8, the F file is used. Files 9–999 can be used or assigned as needed. They may be used to expand the size of input or output files, timer and counter files, etc.

The program portion, or user portion, of memory (Figure 5–12) is used to store information that relates to the user program, or for information that is needed for the processor to operate. File 0 is used to store ASCII information, while File 1 is reserved for internal use by the processor. File 2 is where the user program is stored in RELAY LADDER LOGIC. Files 3–999 are for storing subroutines, fault routines, and selectable timed interrupt (STI) as they are needed.

Figure 5–18 shows the data file structure used by the various PLC-5 models. Note that the I/O section varies from 32 words for the input image table and 32 words for the output image table for the PLC-5/10, 5/12, 5/15, 5/11, 5/20, and 5/20E, while there are 192 words for both the input and output image tables for the PLC-5/60, 5/60L, and 5/80.

The second column of the chart shows the file numbers for the default files. Files 0, 1, and 2 are fixed and cannot be changed. Files 3–8, however, can be changed from the default settings and used as required. For example, if one wanted to use file 3—the binary file—for a timer file, it would be necessary to delete the binary file. The binary file is deleted from the data table map screen and then used as a timer file. Because files 3–8 can be changed, files 3–999 can then be used for timer files, counter files, and the like. However, it is much easier to use files 9–999 when additional files are needed.

SLC 500 AND MICROLOGIX FILE STRUCTURE

When a PLC program for either the SLC 500, the MicroLogix 1000, or the MicroLogix 1500 is being developed, the information needed for the program to function properly is created and stored in processor files. These files are classified into two general types: Program Files (User) and Data Files (Storage).

*DATA TABLE FILES	FILE DESCRIPTION		NUMBER (Default file)	MAXIMUM SIZE OF FILE (16-BIT WORDS)				MEMORY USED	MEMORY USED
				PLC -5/10, -5/12, -5/15	PLC -5/11, -5/20, -5/20E	PLC -5/25	PLC -5/30 PLC -5/40, -5/40E, -5/40L	CLASSIC PLC-5 PROCESSORS	ENHANCED PLC-5 PROCESSORS
PLC-5 MEMORY data table program	OUTPUT	O	0	32	32	64	64 PLC -5/60, -5/60L, -5/80 128	2/file + 1/word	6/file + 1/word
	INPUT	I	1	32	32	64	64 128	2/file + 1/word	6/file + 1/word
	STATUS	S	2	32	128	32	128 128	2/file + 1/Word	6/file + 1/word
	BIT (BINARY)	B	3-999 (3)	1000				2/file + 1/word	6/file + 1/word
	TIMER	T	3-999 (4)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	COUNTER	C	3-999 (5)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	CONTROL	R	3-999 (6)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	INTEGER	N	3-999 (7)	1000				2/file + 3/word	6/file + 3/word
	FLOATING POINT	F	3-999 (8)	1000				2/file + 2/float word	6/file + 2/float word
	ASCII	A	3-999	1000				2/file + 1/2 per character	6/file + 1/2 per character
	BCD	D	3-999	1000				2/file + 1/word	6/file + 1/word
	BLOCK TRANSFER[1]	BT	3-999	1000 structures of 6					6/file + 6/structure
	MESSAGE[1]	MG	3-999	585 structures of 56					6/file + 56/structure
	PID[1]	PD	3-999	399 structures of 82					6/file + 82/structure
	SFC STATUS[1]	SC	3-999	1000 structures of 3					6/file + 3/structure
	ASCI STRING[1]	ST	3-999	780 structures of 42					6/file + 42/structure
	EXTRA STORAGE		3-999						

[1]enhanced PLC-5 processors only.

Figure 5–18 Data Table Map File Structure for the PLC-5 Family
(Courtesy of Allen-Bradley)

Program files typically contain controller information (type of processor, I/O configuration, etc.), the ladder logic program, subroutine programs, and interrupt subroutines. The specific program files for the SLC 500 are shown in the tables below.

SLC 500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
Subroutine ladder program file (file 3-255)	Stores any subroutines not created in the main ladder diagram.

SLC 500 Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.
Floating Point (file 8)* *This file applies to selected SLC 500 processors.	Stores numbers with a decimal point or floating point.
String (user defined file)*	*This file applies to selected SLC 500 processors.
ASCII (user defined file)*	*This file applies to selected SLC 500 processors.

MicroLogix processors have the same program files as the SLC 500, and three additional files.

MicroLogix 1000 and 1500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
User Error Fault Routine (file 3)	File is executed when a recoverable fault occurs.
High-Speed Counter Interrupt (file 4)	File is executed when a high-speed counter interrupt occurs. Can also be used for a subroutine ladder program.
Selectable Timed Interrupt (file 5)	Executed when a selectable time interrupt occurs. Can also be used for a subroutine ladder program.
Subroutine ladder program file (file 6-15)	Stores any subroutines that have been created in the main ladder diagram.

Note: *The MicroLogix 1000 and 1500 are set up to always have 15 files. These files cannot be deleted in the ladder program.*

MicroLogix Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.

added cost justifiable? That question can only be answered after all the variables have been considered. The variables could include, but not be limited to:

- Environment.
- Size of program.
- Graphic requirements.
- Requirements for other software, i.e., spreadsheets, word processing.
- Programmability.
- Networking ability.

A final review of some of the advantages and disadvantages of each type of programmer follows.

DEDICATED DESKTOP PROGRAMMERS

Advantages

- VDT to display multiple rungs of the program.
- Ease of programming, user-friendly.
- Designed for industrial use.
- Portability.
- Status of circuit highlighted (intensified or reverse video).

Disadvantages

- Relative high cost.
- Limited models of PLC that can be programmed.
- Limited documentation and graphics capabilities.
- Physical size.

HAND-HELD PROGRAMMERS

Advantages

- Low cost.
- Small size.
- Very portable.

Disadvantages

- Can only view limited Rungs or instructions.
- Fewer functions.
- Limited access to memory.
- Limited or no documentation capabilities.
- Works with only a specific PLC model.
- More difficult to program than a dedicated desktop programmer.
- Uses logic status lights instead of intensified or reverse video.

PERSONAL COMPUTER PROGRAMMERS

Advantages

- VDT to display multiple Rungs of logic.
- Can program any type of PLC (with appropriate software).

- Extensive documentation capabilities; Rung comments, labels, and so forth.
- Cut and paste editing capabilities.
- Low cost when compared to a dedicated desktop programmer.
- Small and portable when laptop or notebook type are used.
- Easy to make copies of program on floppy disk or hard drive.
- Software can be upgraded without changes in hardware.
- View multiple Rungs in any numeric order for troubleshooting.

Disadvantages

- Must operate in normal environment, not industrial rated.
- Initial software costs.
- Software updates or license renewal costs.
- More difficult to program, longer learning curve.
- May require special communication card.

In the final analysis, if money is not a consideration, or the specific application warrants the difference in cost, the industrial computer programmer is the most versatile and serviceable of all the types of programmers that are available.

Chapter Summary

The programming device, or programmer, is used to enter, modify, and monitor the user program. Which type of programming device to use will vary with each application, and what is appropriate for one application may not be the most appropriate for another. The program (ladder diagram) is entered by pushing keys on the keyboard in a subscribed sequence, with the resultant circuit(s) displayed on the VDT of a desktop programmer, and with a LED or liquid crystal display for the hand-held programmer. The visual display can be used as an aid to test the circuit prior to entry into user memory, or for troubleshooting after the circuit is entered into memory and is operational. Contacts and coils are either intensified or displayed in reverse video to indicate power flow or logic continuity. Programming the PLC is not difficult, but time must be spent to become familiar with the specific PLC and its programming techniques.

Review Questions

1. Briefly describe the function of the *programming device*.
2. What does the acronym VDT stand for?
3. Define the term *on line*.
4. List the three type of programming devices and give advantages and disadvantages for each.
5. Define the term *off line*.
6. What is a hardware key?
7. What is meant by the term "third party software"?
8. What is the advantage of a sealed touch pad-type keyboard over a standard raised key keyboard?
9. In the context of this chapter, what do the terms *passing current* and *power flow* indicate?
10. Hand-held programmers usually have what type(s) of displays?

CHAPTER

5 Memory Organization

Objectives

After completing this chapter, you should have the knowledge to:

- Identify the two broad categories of memory and describe the function of each.
- Identify the types of information stored in each category of memory.
- Define the term *byte*.
- Define the acronym *bits*.
- Define *holding registers*.
- Understand the term *default*.

MEMORY WORDS AND WORD LOCATIONS

For the programmable controller to function properly and control a process or driven equipment, it must be able to perform the user program repeatedly and accurately. The system must also be able to perform its control function with great speed, which is achieved by processing all information in binary signals. The key to the speed with which binary information can be processed is that there are only two states, each of which is distinctly different. Binary signals fall into one of two states, which are 1 and 0. The 1 and 0 can represent *ON* or *OFF*, true or false, voltage or no voltage, high or low, or any other two conditions depending on the system. There is no in-between state or condition, and when information is processed, the decision is either *yes* or *no*. There is no *maybe*, *almost*, or any other alternative.

As indicated in Chapter 3, the processor memory consists of hundreds or thousands of locations that are referred to as words. Each word is capable of storing binary data in the form of binary digits, or **bits** (**BI**nary digi**TS**). A binary digit, like a binary signal, can only be a 1 or a 0. The number of bits that a word can store will depend on the system or PLC. Words can be made up of 32 bits, 16 bits, or 8 bits. The 16-bit word is the most common (Figure 5–1).

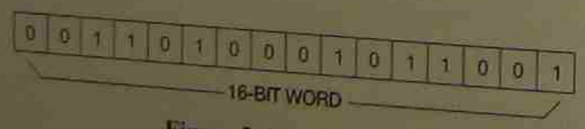


Figure 5–1 16-Bit Word

If a memory size is 256 words, then it can actually store 4,096 bits of information using 16-bit words (256 words × 16 bits per word) or 2,048 bits using an 8-bit word (256 words × 8 bits per word). When comparing memory sizes of different PLC systems, it is important to know the number of bits per word of memory. Bits can also be grouped within a word into **bytes**. A byte is a group of 8 bits.

So that information stored in each word can be located, each word is numbered or given an address. Addressing words in the memory serves the same function as the addresses used for homes or apartments. Word 100, for example, represents a specific word location in memory, just like N. 100 Lincoln represents the address of an apartment building. The bits in word 100 are found by referencing a given bit number, just like the occupant of the apartment complex is found by a given apartment number.

Since a bit of information can only be a 1 or a 0 (*ON* or *OFF*), how is the status of bits within a word determined? Words that store the status of individual bits for input devices are set to 1 (*ON*) or 0 (*OFF*), depending on the status (*ON* or *OFF*) of the input devices that the bit locations represent. Other bits are set to 1 or cleared to 0 by the processor in response to the logic of the user program, RELAY LADDER LOGIC, or special instructions, which, in turn, control the status (*ON* or *OFF*) of other bits that represent output devices.

A simple example of how this works is illustrated in Figure 5–2.

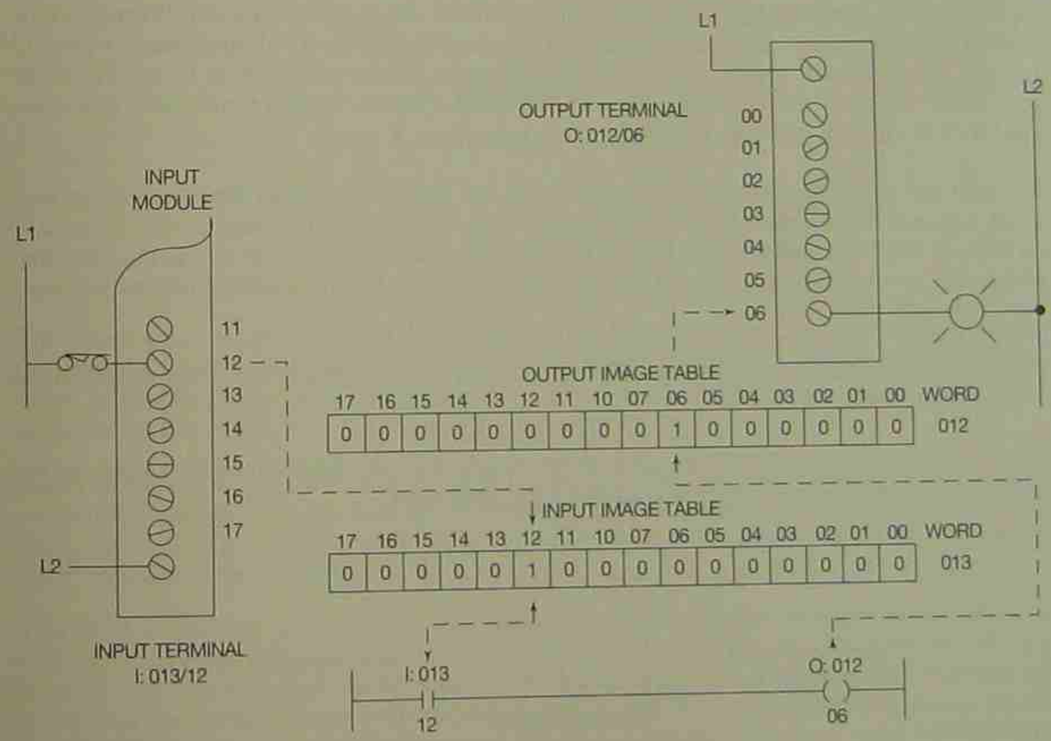


Figure 5–2 Relationship of Bit Address to Input and Output Devices

Note: *The example uses memory organization and addressing utilized by the Allen-Bradley PLC-5 family. While the example is specific to Allen-Bradley, the concepts illustrated are common to all PLCs. Allen-Bradley uses an octal numbering system to address bit locations. Notice that the 16 bits are numbered 00 through 07 and 10 through 17. In the octal numbering system, the numbers 8 and 9 are never used. The octal numbering system will be covered in detail in Chapter 6.*

Assume that when a given limit switch is closed, the closure will turn an indicator lamp *ON*. The limit switch is connected to an input module in the I/O rack, while the indicator lamp is connected to an output module. Chapter 2 discussed DIP switches that were set in a prescribed sequence to identify the I/O rack number for the processor, and that the location of each terminal point of each I/O module within the rack determined the address of a given device. In Figure 5–2, the limit switch is connected to terminal 12 on an input module, and is given an address of I:013/12. This indicates that bit 12 of input image table word 013 stores the status (*ON*-[1] or *OFF*-[0]) of the limit switch. The indicator lamp is connected to terminal 06 of an output module and is given an address of O:012/06. This address indicates that bit 06 of output image table word 012 controls the status (*ON*-1 or *OFF*-0) of the lamp.

By programming a simple circuit into the user memory of the processor as shown at the bottom of Figure 5–2, the processor controls the indicator lamp using the logic of the user program. The logic states that if contact I:013/12 closes, lamp O:012/06 should light, or go *ON*. When power is applied to the processor, the processor starts its scan and looks at bit 12 of input image word 013 to see if the bit is set to 1 or 0. If the limit switch is open, the bit will be set to 0, or *OFF*. If the limit switch is closed, as indicated in Figure 5–2, the input module sends a signal to the processor, and bit 12 of input image word 013 will be set to 1, or *ON*.

The next part of the scan solves the user program. The logic of the ladder diagram, or user program, indicates that when contact I:013/12 (bit 12 of input word 013) is closed, or *ON*, the indicator lamp O:012/06 should be turned *ON*. The processor reads the logic, and during the third step of the scan, sets bit 06 of output word 012 to 1, which turns the lamp connected to the output module terminal 06 *ON*.

The address I:013/12 also tells us that the limit switch is an input device, and is wired to terminal 12 of module group 3 of rack 1.

Figure 5–3 illustrates the significance of each letter/digit or group of digits used for addressing the Allen-Bradley PLC-5 family of programmable logic controllers.

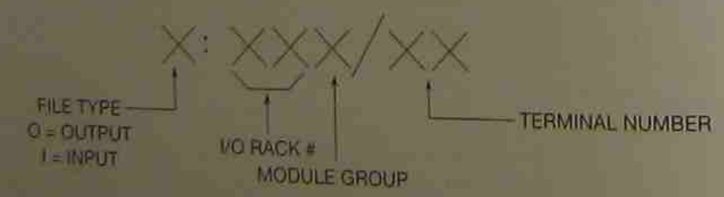


Figure 5–3 Allen-Bradley PLC-5 Address Format

The first letter is used to indicate the type of file (input, output, timer, counter, etc.) The letter I represents an input device, and the letter O represents an output.

The next two digits identify the rack number. One rack can control 128 I/O points. Rack numbers start at 00. A rack is different than a chassis. The chassis is the physical frame that actually holds the input and output modules that make up a rack. Depending on the density of the I/O modules used, a rack may require a 16-slot chassis or only 4 slots. If 8-point I/O are being used, it will take 16 slots of 8-point I/O modules to make 128 I/O points ($8 \times 16 = 128$), whereas if 32 point I/O are being used, it will only take a 4-slot chassis to make a rack ($32 \times 4 = 128$).

The next number identifies the module group within the rack. This is always a number from 0 through 7. The last two digits identify the actual terminal number to which the device is wired.

Figure 5–4 reviews the concept using the address of the limit switch I:013/12.

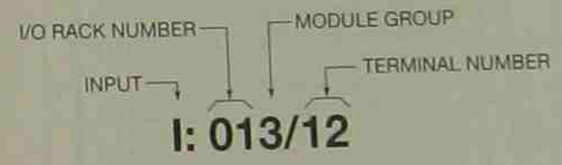


Figure 5–4 Limit Switch Address I:013/12

The letter I tells us that the address represents an input device. The next two digits, 01, tell us that the device is located in I/O rack number 01. The next digit, which is a 3, further identifies the location as module group number 3. The last two digits, 1 and 2, identify the actual terminal (12) on the input module to which the limit switch is connected.

Another example of this concept is shown in Figure 5–5. The limit switch address I:013/12 gives us a hardware location for an input device in rack 01, module group 3, terminal 12. This same address, I:013/12, tells us that the status (*ON* or *OFF*) or state of the limit switch is reflected by bit 12 of word 013 in the input image table.

This same addressing scheme gives us a hardware location for the indicator lamp addressed O:012/06. The letter O indicates an output device. The next two digits, 01, tell us that the I/O rack location is 01. The next digit identifies the module group as group 2. The last two digits locate terminal 06 as the terminal on the output module to which the indicator lamp is wired. Again, the address O:012/06 also locates the memory word and bit location that reflects the status (*ON* or *OFF*) of the indicator lamp as shown in Figure 5–6.

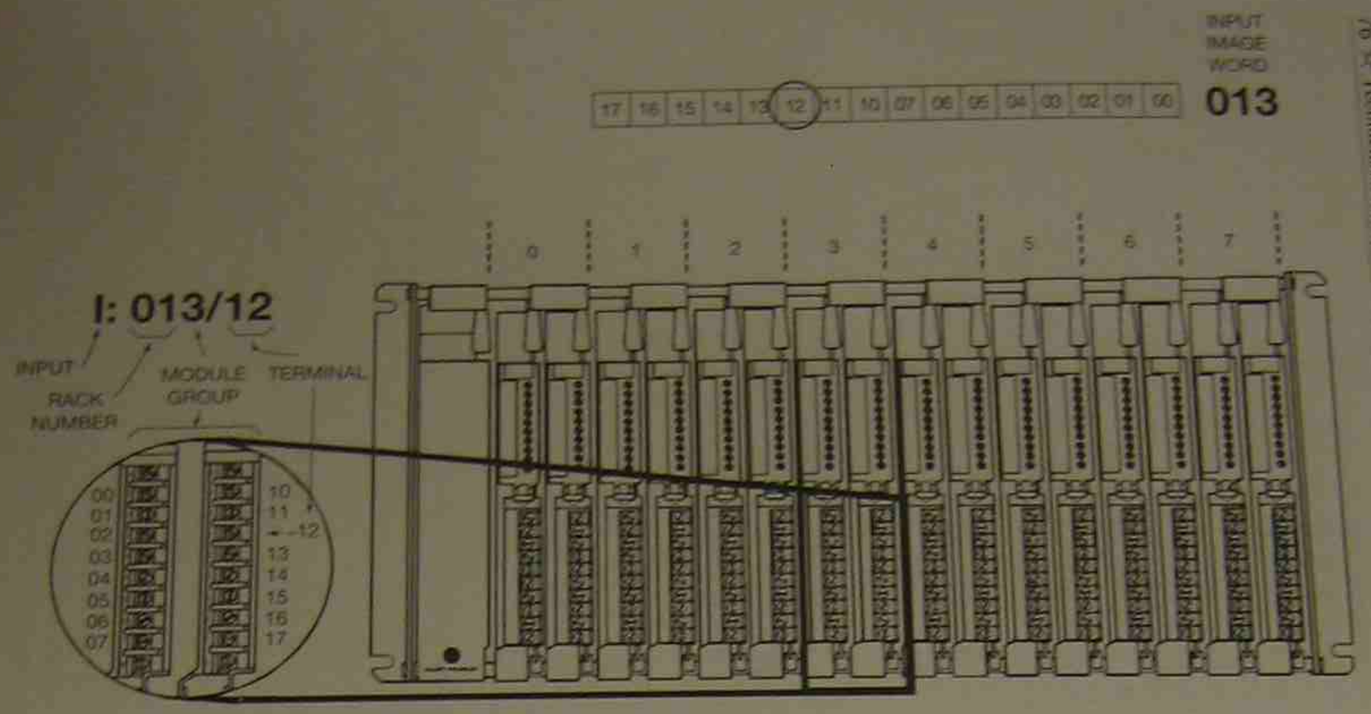


Figure 5–5 Relating Input Address I:013/12 to Actual Hardware Location
(Courtesy of Allen-Bradley)

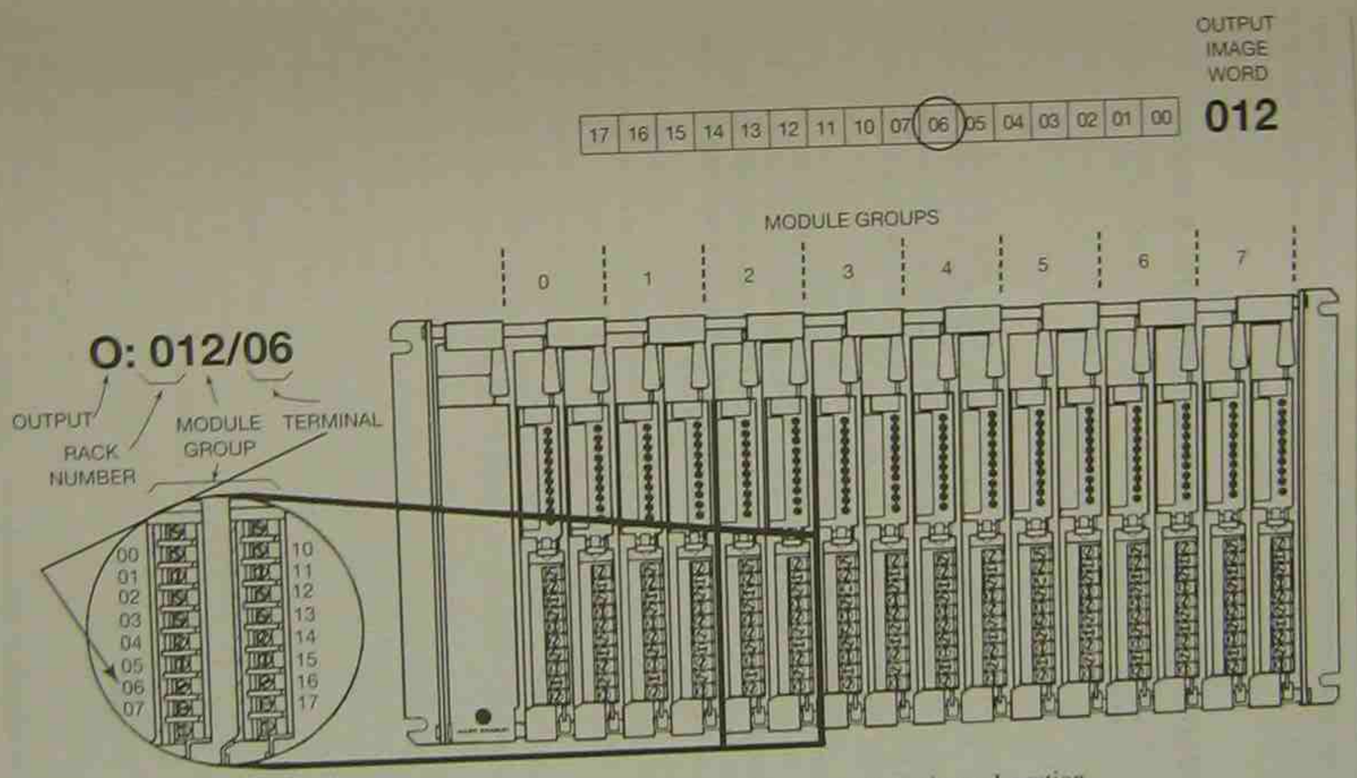


Figure 5–6 Relating Output Address O:012/06 to Actual Hardware Location
(Courtesy of Allen-Bradley)

While the address system discussed is specific to the Allen-Bradley PLC-5 family, most PLC manufacturers use an addressing scheme that identifies memory word locations, and may also give hardware locations.

SLC 500 AND MICROLOGIX 1000 AND 1500 ADDRESSING SCHEME

Like the PLC-5 family, the SLC 500 and MicroLogix family use the letter I for an input address and the letter O for an output address.

I = external input device
O = external output device

A typical address for an input device would be I:0.1/6. The I, of course, indicates that this is the address for an input device. The colon (:) is called an element delimiter. This means that the colon separates the input designator, I, from the rest of the address. The next character, the number 0, indicates the slot number that holds the actual input module. The slot number can range from slot 0, adjacent to the power supply in the first chassis, to a maximum of 30.

If the number of inputs is more than 16, a period (.), which is called a word delimiter, would be used after the slot number. The forward slash (/) is called the bit delimiter. The number that follows the forward slash is the bit number of the word as well as the terminal number of the I/O module. The digit 6 indicates the terminal number where the input device is wired. Figure 5–7 illustrates the addressing scheme for the SLC 500 and the MicroLogix family.

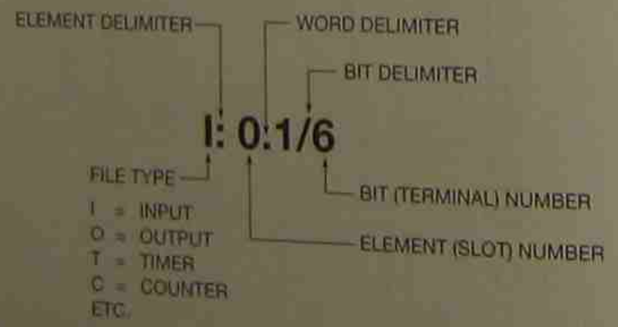


Figure 5–7 Allen-Bradley SLC 500 and MicroLogix Addressing Scheme

The first character of the address identifies the file type. The typical file types used are:

O	Output
I	Input
S	Status
B	Bit
T	Timer
C	Counter
R	Control
N*	Integer
F*	Float
St*	String
A*	ASCII

*Only available on selected processors.

In the address shown in Figure 5–7 the I indicates that the file type is an Input file and also indicates this is the address of an input device. The 0 after the :, which is the element delimiter, indicates that the input device is connected to slot 0. The period (.) after the slot number indicates that the inputs exceed 16 and require two words in the input image table. The number 1 indicates that this is word 1 in slot 0. The number 6 after the forward slash is the bit number.

Note: *The SLC 500 and MicroLogix do not use the octal numbering system like the PLC-5 family, but instead use the decimal numbering system.*

For 16 inputs, the bits are numbered 0–15. The address, by using the period word delimiter (.), indicates that there are more than 16 inputs installed in slot 0. Bit 6 is actually input device 22 (16 + 6 = 22). Because we started with bit 0, input device 22 is in reality the 23rd input device. Likewise, input address I:0.1/7 is the 24th input device.

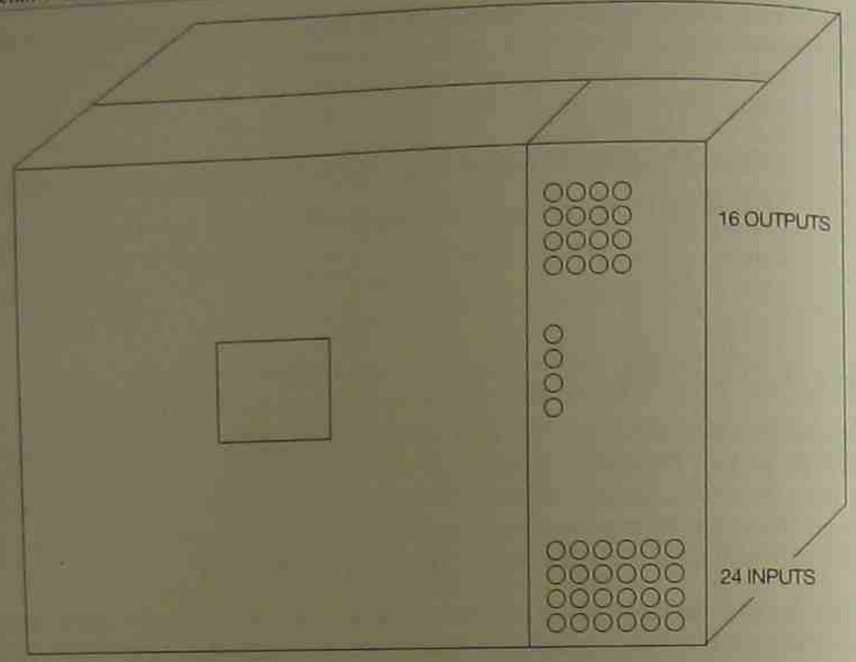
The SLC 500 controller is available in either fixed or modular I/O. The fixed I/O units have fixed I/O of 20 (12 inputs and 8 outputs), 30 (18 inputs and 12 outputs), and 40 (24 inputs and 16 outputs).

For fixed I/O controllers, all of the I/O are in slot 0. Figure 5–8 shows a fixed I/O controller that has 24 inputs and 16 outputs. Also shown are the input and output image tables for Data File 0 (output) and Data File 1 (input). Note that for 24 inputs, the Input Image table uses two words for slot 0. All 16 bits of word 0 (I:0) are used, whereas only the first 8 bits of word 1 (I:0.1) are used (bits 0 through 7). The unused bits of word 1—bits 8 through 15—are marked invalid and are not available for use.

An output address of O:0/4 indicates that this is the address of an output device in slot 0, terminal 4. This address also indicates that this is bit 4 of word O:0 in the Output Image table data file 0 as indicated by the *X* under bit 4 of output word O:0 (Figure 5–8).

An input address of I:0/15 indicates controller input 15 in slot 0. This address is indicated by the *X* placed under bit 15 of input word I:0 in the input image table (Figure 5–8).

An address of I:0.1/7 indicates an input device in slot 0, word 1, bit 7. This address is indicated by the *X* placed under bit 7 of input image Data File 1, word 1 in Figure 5–8. This address also indicates that this is input device 23. As we started with 0 as the first input device, the 23rd device is



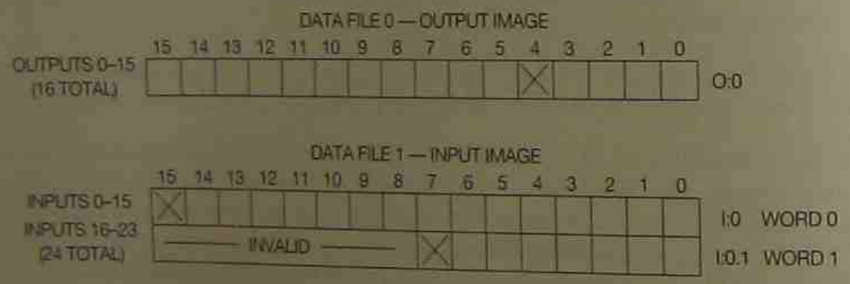


Figure 5–8 SLC 500 with 40 Fixed I/O

actually the 24th and last input device. The first word of the input image table, word I:0, held input device information for inputs 1 through 16, which are bits 0 through 15. The second word, word I:0.1, of the input image table holds the information for input devices 17 through 24, which are represented by bits 0 through 7. Bit 0 of word I:0.1 of Data File 1 represents input device 17. An alternate way to address input terminal 23 of slot 0 would be I:0/23.

Figure 5–9 shows an SLC 500 modular controller that consists of a seven-slot chassis interconnected to a ten-slot chassis.

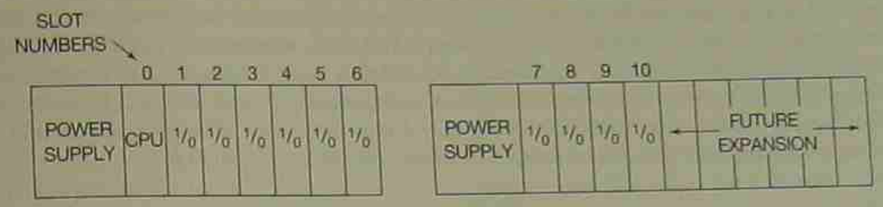


Figure 5–9 SLC 500 Modular Controller (Seven-Slot Chassis Connected to a Ten-Slot Chassis)

In this configuration, slot 0 contains the CPU, so slot 0 becomes an invalid I/O slot number. The controller is configured as follows:

SLOT	INPUTS	OUTPUTS
1	6	6
2	32	None
3	None	16
4	8	8
5	None	32
6	16	None
7	16	None
8	8	None
9	None	16
10	None	16

Figure 5–10 shows Data File 0 (the output image table) and Data File 1 (the input image table) as it would appear for the mix of I/O described above. Note that wherever 32-point I/O modules are used, the data table will require two 16-bit words. Slot 2 has a 32-point input module, and the input image table shows word I:2 (word 0 for inputs 0 through 15) and word I:2.1 (word 1 for inputs 0–15 which are actually inputs 17 through 32). Likewise, where 32 outputs are used in slot 5, the output image table for slot 5 shows that two words are used: O:5 and O:5.1.

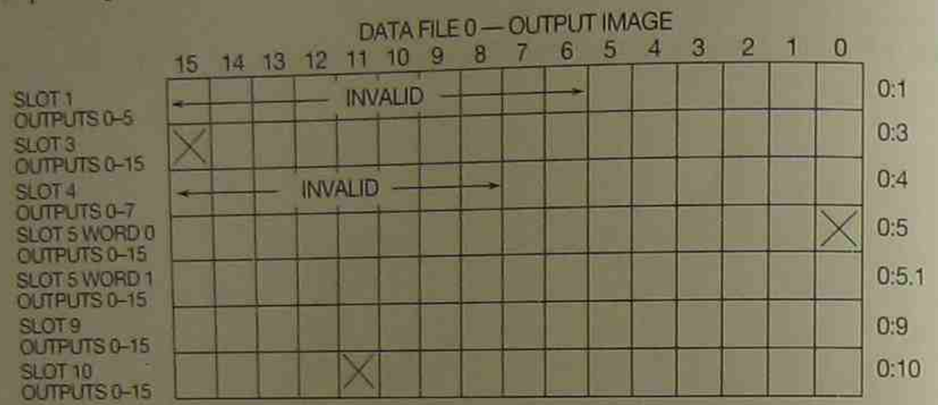


Figure 5–10 Output and Input Image Tables for the SLC 500 Configuration Shown in Figure 5–9

For the modular system shown in Figure 5–9, an address of O:3/15 indicates an output device 15 in slot 3. It also indicates bit 15 of word O:3 of the Output Image table, shown by an *X* under bit 15 of output word O:3 in Figure 5–10. An address of I:2.1/3 indicates input device 3 of word 1 in slot 2 (as indicated by an X under bit 3 of input word I:2.1). Slot 2 holds a 32-point input module which requires two words in the input image table: Word 0 (I:2) and Word 1 (I:2.1) shown in Figure 5–10. To further clarify the addressing scheme, address I:7/8 indicates input number 8 in slot 7 as indicated by the *X* under location 8 of slot 7 which is word I:7 of the input image table.

Note that slot 4 of both the input and output image tables has 8 inputs and 8 outputs. While only 8-bits of each word in the input and output image table are needed, the unused bits cannot be used for any other programming as indicated by the word *invalid* on the image tables.

Figure 5–11 shows an SLC 500 seven-slot chassis connected to a four-slot chassis using an Allen-Bradley 1746-C9 communications cable. The processor is installed in slot 0 of the first chassis which is adjacent to the power supply. Figure 5–11 shows an exploded view of the output/input module installed in slot 3. The output/input module is shown with the door that covers the terminals open. The inside of the door shows a pictorial view of the terminal layout. The first vertical row of terminal screws are for the six outputs plus an AC common connection. The second vertical row of terminals is for the six input devices and an AC common connection. The normally open pushbutton that is shown would have an address of I:3/0. I is for input, the 3 indicates that the input device is installed in slot 3, and the 0 indicates the input device is connected to input terminal 0.

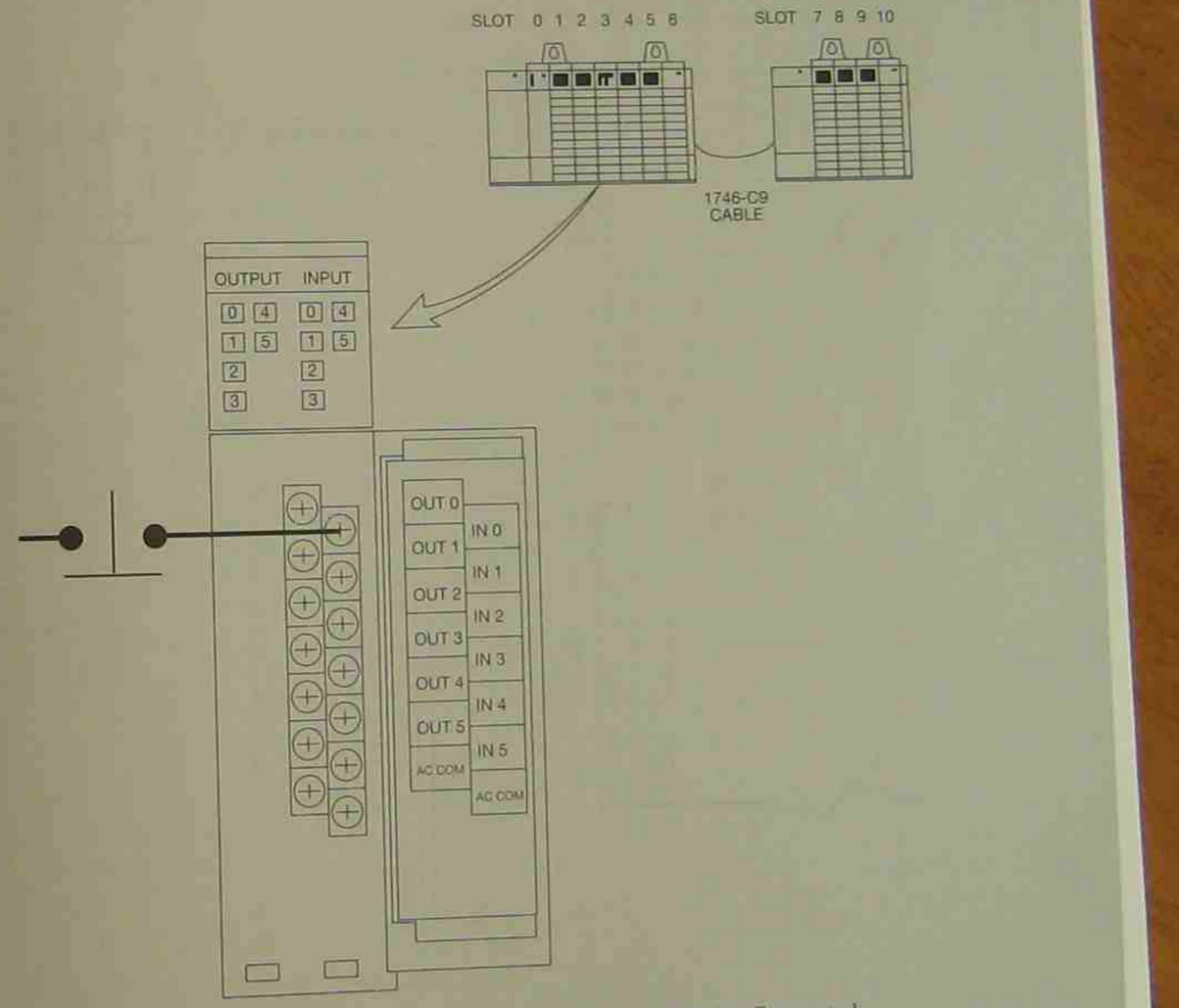


Figure 5–11 SLC 500 Modular Controller with Input Device Connected

Figure 5–12 shows the same seven-slot chassis connected to a four-slot chassis. In this figure, an output device (solenoid) is connected to one terminal of a 16-point output module that is installed in slot 8. As before, the terminal door is open and shows the layout of the terminals. The first terminal is for connecting the AC power. Notice that the output numbers alternate from Out 0, then Out 1, Out 2, and so on. The very last terminal is for the AC common, or neutral connection. The address for the solenoid connected as shown would be O:8/12. The O indicates an output device, the 8 indicates that the module is installed in slot 8, and the 12 indicates that the solenoid is connected to output terminal 12. This address also indicates that the status of the output device is found in bit 12 of Output Image Table word 8.

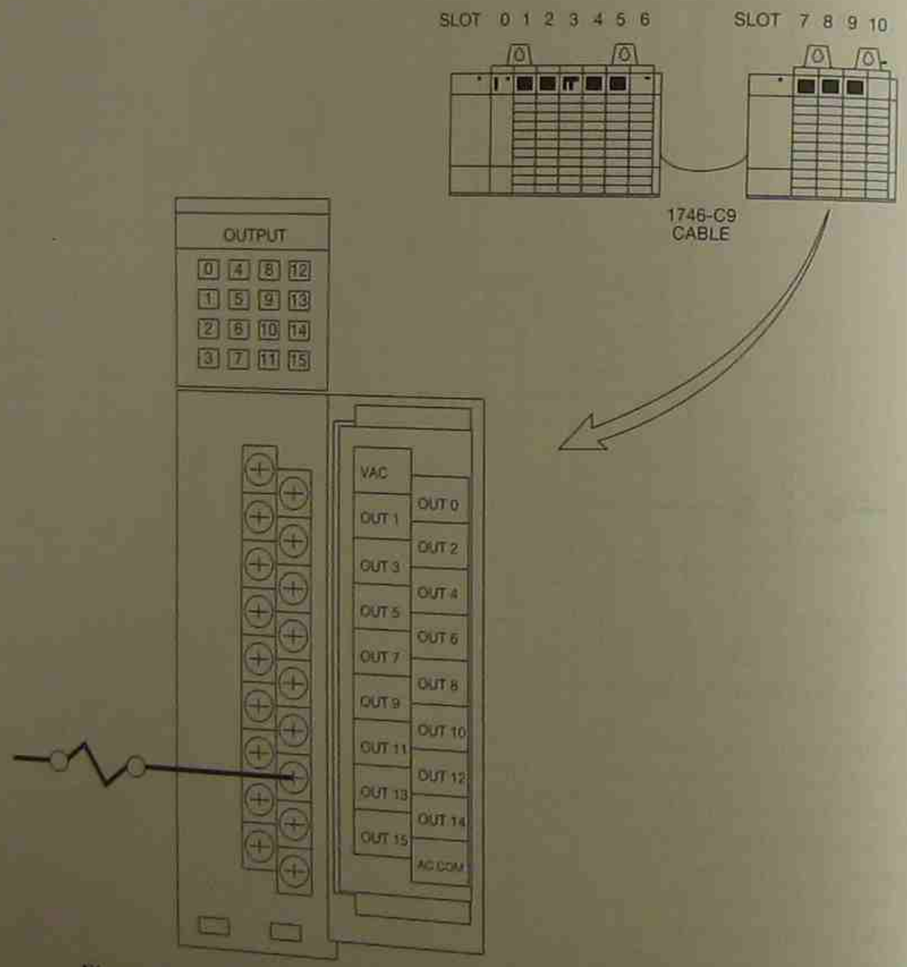


Figure 5–12 SLC 500 Modular Controller with Output Device Connected

Figure 5–13 shows an analog input module installed in slot 10. The exploded view of the module shows a sensor connected to the module at terminals 3 and 4. As connected, the address of the analog sensor is I:10/1. From the layout on the terminal door, we can see that terminal 3 and 4 are identified as Input 1+ and Input 1–.

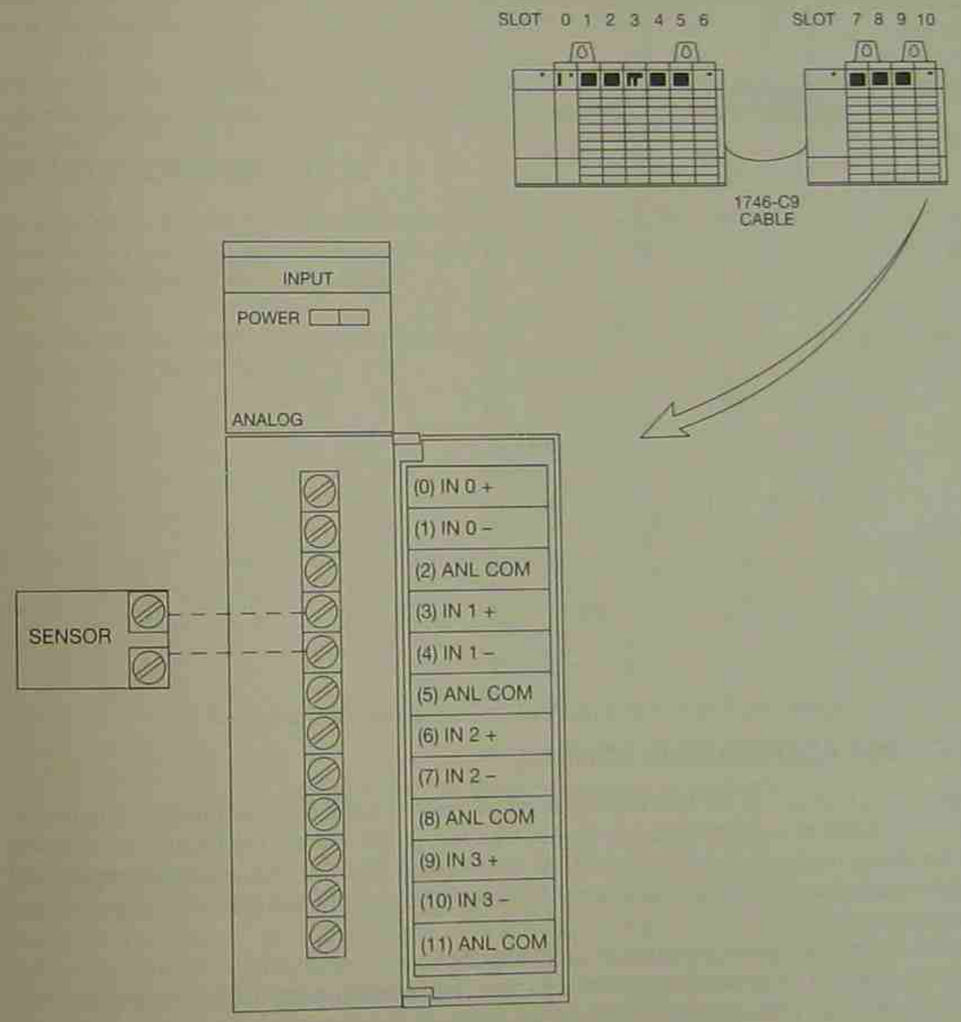


Figure 5–13 SLC 500 with Analog Input Device Connected

Figure 5–14 shows an analog output module installed in slot 6. The analog output in this illustration is an actuator connected to terminals 6 and 7. The address for the actuator is O:6/3.

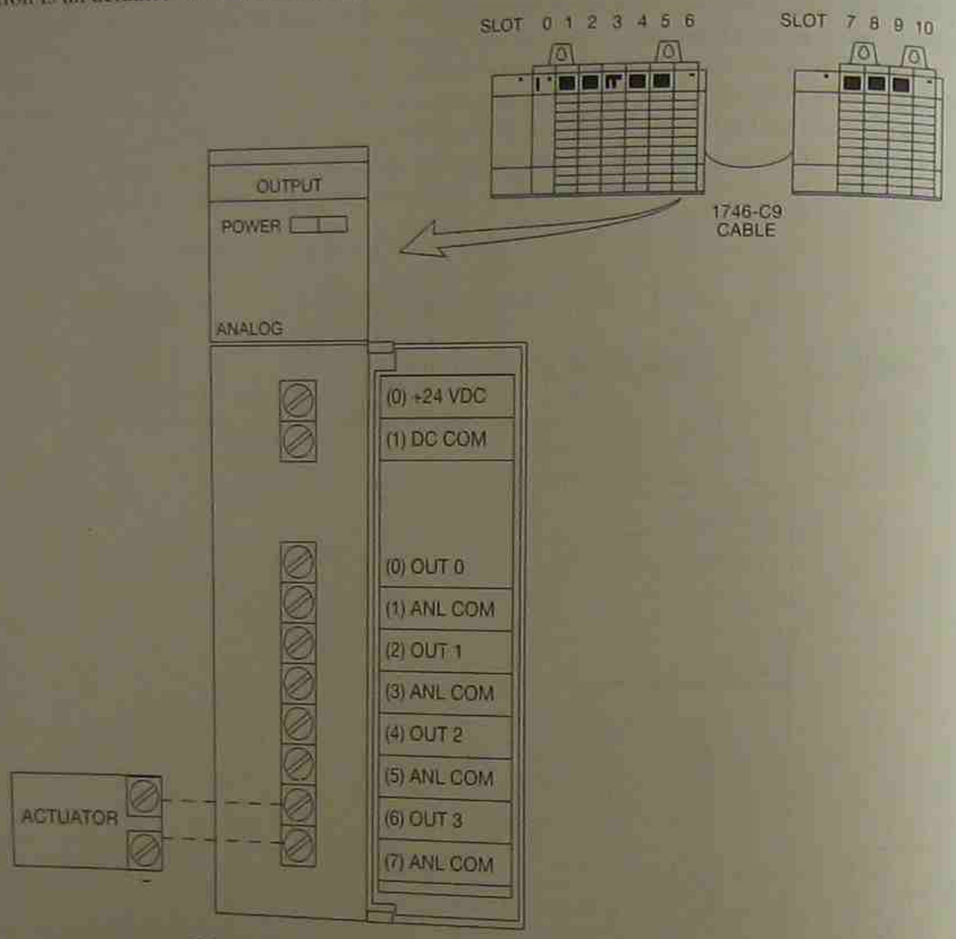


Figure 5–14 SLC 500 with Analog Output Device Connected

MODICON 984 ADDRESSING SCHEME

The Modicon 984 uses a 1 as the first number in an address for a discrete input device, and uses a 0 for the first number of a discrete output device. The rest of the address information is assigned by the programmer using a software utility called the I/O Map. The I/O Map is used to assign reference addresses to specific I/O modules, slots, and rack locations. By using the I/O Map software utility, the programmer defines:

- What slots in a given rack contain I/O modules.
- What type of I/O module is expected to be in a given slot, as opposed to what is actually there, as with the Allen-Bradley PLC-5 family.
- What addresses are to be assigned to which modules.

Instead of being hardware-driven like the Allen-Bradley addressing scheme, the 984 is software-driven and programmer-defined.

Discrete *input device* addresses use a five-digit number beginning with a 1 (1XXXX). The remaining four numbers that will make up the address are assigned by the programmer based on how the system is configured. Input registers are used for analog devices, absolute encoders, thermocouples, and the like, and these addresses will start with the number 3 (3XXXX).

Discrete *output devices* begin with the number 0 (0XXXX). For analog output devices such as meters, transducers, and variable speed drive motors, output registers are used and have addresses that start with the number 4 (4XXXX).

Once you gain experience using the I/O Map utilities portion of the Modicon programming software to assign addresses, the process becomes quite easy and very understandable.

MEMORY ORGANIZATION

At this point, it should be no surprise to find out that not all PLC manufacturers have organized their memories in the same way, or that they do not all use the same terminology for the configuration or make-up of their memories.

As discussed in Chapter 3, there are two general classifications of memory: storage memory and user memory (Figure 5–15).

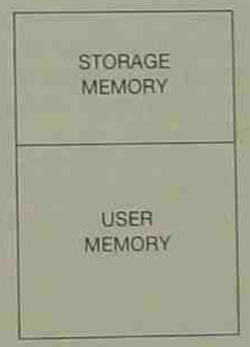


Figure 5–15 Two Broad Categories of Memory

Storage Memory

Storage memory is that portion of memory that will store information on the status of input and output devices, preset and accumulated values of timers and counters, internal relay equivalents, numerical values for arithmetic functions, and so on. The entire storage memory is called a data table, a register table, or other names, depending on the PLC manufacturer. A register is defined as an area for storing information (logic or numeric). Although the names or titles which are given to sections or subsections of the storage memory vary, the principles involved do not.

For example, the section of the memory that stores the status of the real-world input devices may be referred to as an input image table, input register, input status table, or external input section. No matter what name is used, the information is stored in the same way. The status (*ON* or *OFF*) of each input device is stored as either a 1 or a 0 (*ON* or *OFF*) in one bit of a memory word. When the processor is executing the user program (ladder diagram), it scans the input device status stored in the storage memory to determine which inputs are *ON* or *OFF*.

The section of storage memory set aside for output status may be referred to as the output image table, output register, output status table or external output section. Again, the name does not change the function of this section of the storage memory, or the method by which information is placed in memory for control of the actual output devices. As the processor executes the user program, it sends binary data (1s or 0s) to the output section of memory to control the output devices. Each output device is represented by one bit of a memory word.

Numeric information for timer or counter preset and accumulated values, arithmetic functions, sequencer functions, data manipulation, etc., uses a part of the storage memory that is called data registers or internal storage. Information is entered and stored in this part of memory using the binary, BCD, or hexadecimal numbering systems (the various numbering systems are covered in Chapter 6). The numbering system(s) used depend on the PLC hardware and system requirements. The storage of numeric information requires that several bits be used of one word to represent numbers. In a practical sense, any word used to store numerical information is not available for additional storage, even if all the bits of word are not used.

In general, any unused bits of a word that are not used for storing numeric values can be used as internal relays. Internal relays will replace the numerous control relays used in most hardwired control circuits. Many PLCs have a portion of memory set aside just for internal relays. Internal, or dummy relays, may also be programmed in the output section of memory when all the words or bits of words are not being used for real-world output devices. The concept and use of internal, or dummy relays, is covered later in the text.

Figure 5–16 shows the memory organization for an Allen-Bradley Mini PLC-2/15. The 2/15 is part of the PLC-2 family of Allen-Bradley processors. Allen-Bradley also has the newer PLC-5 family of processors.

Note: *While the PLC-2 family is the older version, there are literally thousands of this type still in use in industry today, and an understanding of their memory structure is meaningful.*

For the PLC-2 processors, Allen-Bradley uses 16-bit words. The word and bit addresses are numbered using the octal numbering system, as shown in Figure 5–11. The first eight words (000–007) of the data table are set aside for the processor. Words 010–017 and 020–026 are the output image table. An output device addressed 01000 is bit 00 of word 010; similarly, an address of 02617 is bit 17 of word 026. Note; word 027 is used by the processor for a *BAT LOW* condition, message generation, and data highway. The next 40-word section of memory, words 030–037, 040–047, 050–057, 060–067, and 070–077, is used for storing timer or counter accumulated values (numeric) or for internal storage.

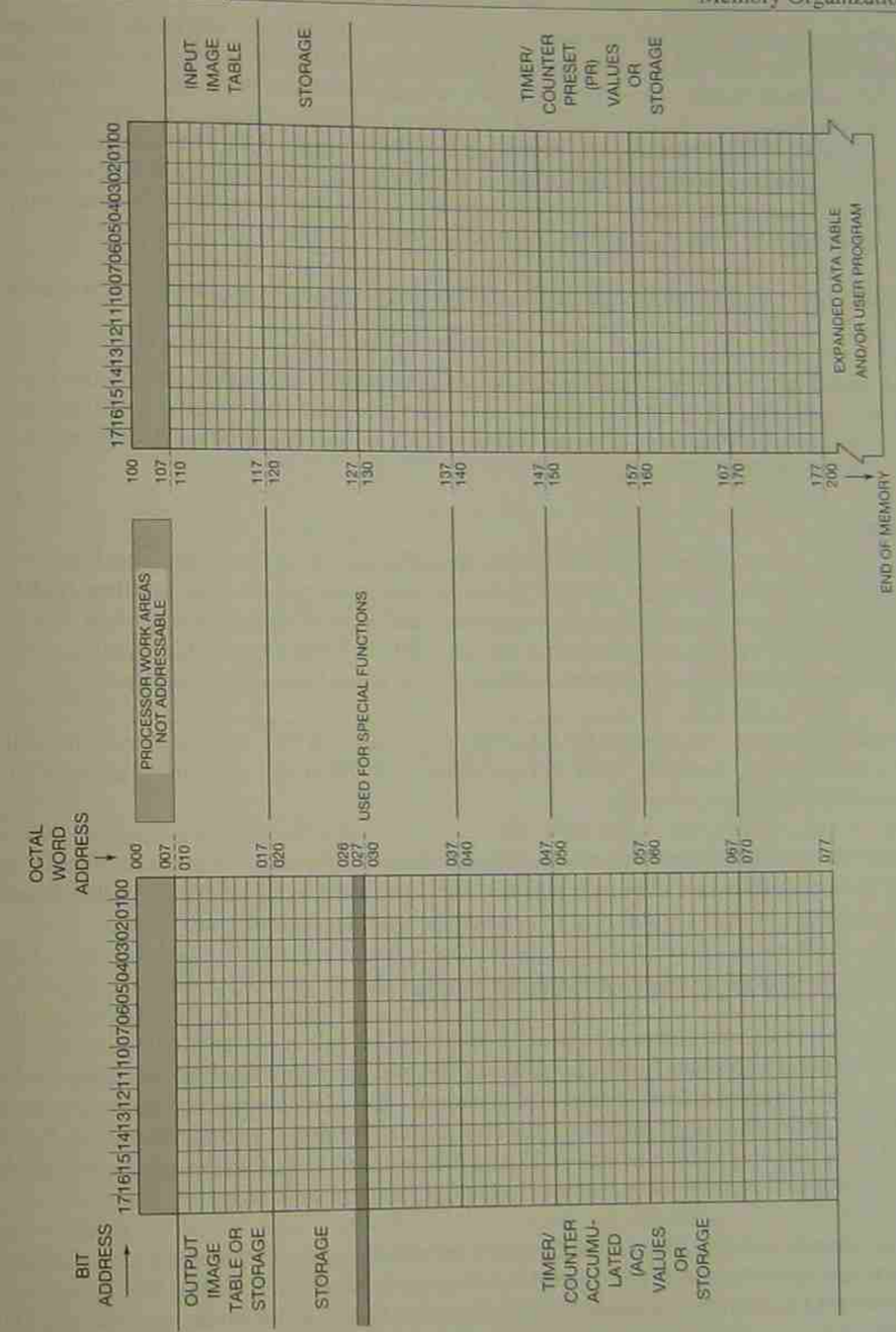


Figure 5–16 First 128 Words of an Allen-Bradley PLC-2/15 Memory

The next 8 words are the second processor work area. Words 110–127 (16 words) make up the input image table for discrete input addresses. Input address 12406 is bit 06 of word 124. Words 125 and 126 are used by the processor to indicate remote I/O faults. Words 130–177 (40 words) are used to store the preset values of timers or counters or for internal storage. If word 030 is used for a timer, word 130 automatically stores the preset value of timer 030. The accumulated value for timer 030 is stored in word 030. Any timer or counter in an Allen-Bradley PLC-2 system will store the preset value in the word that is numbered 100 higher than the timer or counter number. For example, counter 047 has its preset value stored in word 147.

Below the timer or counter preset values and internal storage section is the user program section of memory.

The actual configuration of the data table can be changed to meet user needs. Input/output image tables can be increased or expanded to handle more discrete inputs and outputs. Additional information on expanding the data table of Allen-Bradley PLCs is available from their local representative or from their technical publications.

User Memory

The user memory, or logic memory as it is sometimes called, is where the programmed ladder diagram is entered and stored. Within the user memory, words are set aside as **holding registers**. Holding registers typically store information generated and used by the processor when it is solving the user program. Holding registers that are set aside to store intermediate values or other short-term bits of information are sometimes referred to as *scratch areas* or *scratch pads*.

The user memory accounts for most of the total memory of a given PLC system. A system with an 8K memory (8192 words) typically has a storage memory of 2K or less, and the balance of memory (6K) is available for user memory.

Once the user program has been entered into the user memory, by either a programming device, tape loader, computer, or by means of a telephone interface, the programmable controller is ready to control the process or driven equipment in accordance with the user program logic.

ALLEN-BRADLEY PLC-5 FILE STRUCTURE

The Allen-Bradley PLC-5 processors are usually programmed with a personal computer and software specific for the PLC-5 family, and the areas of memory are often referred to as files, not tables, as is the case with the PLC-2 memory structure. Although there are still two memory sections (storage [data] and user [program]), the PLC-5 memory map, or structure, is very flexible in the way that the memory can be allocated. Figure 5–17 shows the PLC-5 **default** memory structure. Default refers to the initial value, setting, or configuration prior to any user changes.

In the data or storage memory section file 0 is the output image file. This file has 32 words of 16 bits each, and can hold the status of 512 real-world output devices (32 × 16). The status of the outputs (*ON* or *OFF*) is updated once each scan. On some PLC-5 models, like the PLC-5/25, the size of the file can be increased to accommodate more output devices.

	MAXIMUM NUMBER OF ELEMENTS		FILE NUMBER	DESIGNATION
DATA OR STORAGE MEMORY	32	OUTPUT IMAGE	0	O
	32	INPUT IMAGE	1	I
	32	STATUS (CLOCK FAULT TABLE, ETC.)	2	S
	1000	BIT (INTERNAL RELAYS)	3	B
	100	TIMER	4	T
	1000	COUNTER	5	C
	1000	CONTROL (MATH, SEQUENCER)	6	R
	100	INTEGER (WHOLE NUMBERS)	7	N
	1000	FLOATING POINT (DECIMAL NUMBER, I.E., 3.2)	8	F
USER OR PROGRAM MEMORY		ASSIGN FILE TYPE AS NEEDED	9-999	
		ASCII	0	A
		RESERVED	1	
		MAIN (LADDER LOGIC)	2	
		SUBROUTING	3	
		FAULT ROUTINE		
			999	

Figure 5–17 PLC-5 File Structure

File 1 is the input image file. This file, like file 0, has 32 words of memory and can store the status of 512 input devices. The status (*ON* or *OFF*) of the input devices, like the output image file, is updated once each scan and can be increased in size on many PLC-5 models.

Both files 0 and 1 use the octal numbering system, and the memory locations (bits) are also numbered using the octal numbering system (there are no 8s or 9s in the octal numbering system). The digits are 0–7, 10–17, 20–27, and so forth.

File 2 is the status, or S file. This file is used to store information on general processor status, fault codes, real time clock and calendar, major and minor fault bits, and program scan times in msecs. Information from this file is used or incorporated into the user program. The size of this file changes depending on the processor that is being used.

File 3 is the B, or bit file, and is used primarily for internal or dummy relays. The default size of this file is one word, but can be expanded to 1000 words if needed. All addresses from this file must start with B3. Another B or bit file may be created using the other areas of the memory. A B10 file could be created that would also have internal or dummy relays. The addressing B10 versus B3 is used for organization and ease of identification. The B3 file may be associated with one

piece of equipment, while the B10 file could be associated with another piece of equipment and/or operation. Up to 999 files can be created in a PLC 5/15 processor memory, as long as you do not try to allocate more memory than is physically available in the processor. The B3 file is typical of the remaining files in flexibility, as well as being addressed using the decimal numbering system.

File 4 is the T, or timer file. All timer addresses must start with T4 unless new timer files have been created (i.e., T 9, T 10, and T 11). When creating files, the same number cannot be used twice. If file 10 is used as a B file (B10), then file 10 cannot be used as a timer file. Each timer that is programmed uses three words of memory from its timer file.

File 5 is the C, or counter file. All counters that are programmed have C5 as the start of their addresses. Each counter, like timers, use three words of the counter file memory.

File 6 is the R, or control file. The words in this file are used with special functions like sequencer, file moves, word to file moves, and math functions. File 7 stores whole numbers (integers) and is called the N, or integer file. The integer file is used to store numeric values for data compare, arithmetic functions, and the like. For storing numbers with a decimal point, or floating point, file 8, the F file is used. Files 9–999 can be used or assigned as needed. They may be used to expand the size of input or output files, timer and counter files, etc.

The program portion, or user portion, of memory (Figure 5–12) is used to store information that relates to the user program, or for information that is needed for the processor to operate. File 0 is used to store ASCII information, while File 1 is reserved for internal use by the processor. File 2 is where the user program is stored in RELAY LADDER LOGIC. Files 3–999 are for storing subroutines, fault routines, and selectable timed interrupt (STI) as they are needed.

Figure 5–18 shows the data file structure used by the various PLC-5 models. Note that the I/O section varies from 32 words for the input image table and 32 words for the output image table for the PLC-5/10, 5/12, 5/15, 5/11, 5/20, and 5/20E, while there are 192 words for both the input and output image tables for the PLC-5/60, 5/60L, and 5/80.

The second column of the chart shows the file numbers for the default files. Files 0, 1, and 2 are fixed and cannot be changed. Files 3–8, however, can be changed from the default settings and used as required. For example, if one wanted to use file 3—the binary file—for a timer file, it would be necessary to delete the binary file. The binary file is deleted from the data table map screen and then used as a timer file. Because files 3–8 can be changed, files 3–999 can then be used for timer files, counter files, and the like. However, it is much easier to use files 9–999 when additional files are needed.

SLC 500 AND MICROLOGIX FILE STRUCTURE

When a PLC program for either the SLC 500, the MicroLogix 1000, or the MicroLogix 1500 is being developed, the information needed for the program to function properly is created and stored in processor files. These files are classified into two general types: Program Files (User) and Data Files (Storage).

*DATA TABLE FILES	FILE DESCRIPTION		NUMBER (Default file)	MAXIMUM SIZE OF FILE (16-BIT WORDS)				MEMORY USED	MEMORY USED
				PLC -5/10, -5/12, -5/15	PLC -5/11, -5/20, -5/20E	PLC -5/25	PLC -5/30 PLC -5/40, -5/40E, -5/40L	CLASSIC PLC-5 PROCESSORS	ENHANCED PLC-5 PROCESSORS
PLC-5 MEMORY data table program	OUTPUT	O	0	32	32	64	64 PLC -5/60, -5/60L, -5/80 128	2/file + 1/word	6/file + 1/word
	INPUT	I	1	32	32	64	64 128	2/file + 1/word	6/file + 1/word
	STATUS	S	2	32	128	32	128 128	2/file + 1/Word	6/file + 1/word
	BIT (BINARY)	B	3-999 (3)	1000				2/file + 1/word	6/file + 1/word
	TIMER	T	3-999 (4)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	COUNTER	C	3-999 (5)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	CONTROL	R	3-999 (6)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	INTEGER	N	3-999 (7)	1000				2/file + 3/word	6/file + 3/word
	FLOATING POINT	F	3-999 (8)	1000				2/file + 2/float word	6/file + 2/float word
	ASCII	A	3-999	1000				2/file + 1/2 per character	6/file + 1/2 per character
	BCD	D	3-999	1000				2/file + 1/word	6/file + 1/word
	BLOCK TRANSFER[1]	BT	3-999	1000 structures of 6					6/file + 6/structure
	MESSAGE[1]	MG	3-999	585 structures of 56					6/file + 56/structure
	PID[1]	PD	3-999	399 structures of 82					6/file + 82/structure
	SFC STATUS[1]	SC	3-999	1000 structures of 3					6/file + 3/structure
	ASCI STRING[1]	ST	3-999	780 structures of 42					6/file + 42/structure
	EXTRA STORAGE		3-999						

[1]enhanced PLC-5 processors only.

Figure 5–18 Data Table Map File Structure for the PLC-5 Family
(Courtesy of Allen-Bradley)

Program files typically contain controller information (type of processor, I/O configuration, etc.), the ladder logic program, subroutine programs, and interrupt subroutines. The specific program files for the SLC 500 are shown in the tables below.

SLC 500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
Subroutine ladder program file (file 3-255)	Stores any subroutines not created in the main ladder diagram.

SLC 500 Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.
Floating Point (file 8)* *This file applies to selected SLC 500 processors.	Stores numbers with a decimal point or floating point.
String (user defined file)*	*This file applies to selected SLC 500 processors.
ASCII (user defined file)*	*This file applies to selected SLC 500 processors.

MicroLogix processors have the same program files as the SLC 500, and three additional files.

MicroLogix 1000 and 1500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
User Error Fault Routine (file 3)	File is executed when a recoverable fault occurs.
High-Speed Counter Interrupt (file 4)	File is executed when a high-speed counter interrupt occurs. Can also be used for a subroutine ladder program.
Selectable Timed Interrupt (file 5)	Executed when a selectable time interrupt occurs. Can also be used for a subroutine ladder program.
Subroutine ladder program file (file 6-15)	Stores any subroutines that have been created in the main ladder diagram.

Note: *The MicroLogix 1000 and 1500 are set up to always have 15 files. These files cannot be deleted in the ladder program.*

MicroLogix Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.

added cost justifiable? That question can only be answered after all the variables have been considered. The variables could include, but not be limited to:

- Environment.
- Size of program.
- Graphic requirements.
- Requirements for other software, i.e., spreadsheets, word processing.
- Programmability.
- Networking ability.

A final review of some of the advantages and disadvantages of each type of programmer follows.

DEDICATED DESKTOP PROGRAMMERS

Advantages

- VDT to display multiple rungs of the program.
- Ease of programming, user-friendly.
- Designed for industrial use.
- Portability.
- Status of circuit highlighted (intensified or reverse video).

Disadvantages

- Relative high cost.
- Limited models of PLC that can be programmed.
- Limited documentation and graphics capabilities.
- Physical size.

HAND-HELD PROGRAMMERS

Advantages

- Low cost.
- Small size.
- Very portable.

Disadvantages

- Can only view limited Rungs or instructions.
- Fewer functions.
- Limited access to memory.
- Limited or no documentation capabilities.
- Works with only a specific PLC model.
- More difficult to program than a dedicated desktop programmer.
- Uses logic status lights instead of intensified or reverse video.

PERSONAL COMPUTER PROGRAMMERS

Advantages

- VDT to display multiple Rungs of logic.
- Can program any type of PLC (with appropriate software).

- Extensive documentation capabilities; Rung comments, labels, and so forth.
- Cut and paste editing capabilities.
- Low cost when compared to a dedicated desktop programmer.
- Small and portable when laptop or notebook type are used.
- Easy to make copies of program on floppy disk or hard drive.
- Software can be upgraded without changes in hardware.
- View multiple Rungs in any numeric order for troubleshooting.

Disadvantages

- Must operate in normal environment, not industrial rated.
- Initial software costs.
- Software updates or license renewal costs.
- More difficult to program, longer learning curve.
- May require special communication card.

In the final analysis, if money is not a consideration, or the specific application warrants the difference in cost, the industrial computer programmer is the most versatile and serviceable of all the types of programmers that are available.

Chapter Summary

The programming device, or programmer, is used to enter, modify, and monitor the user program. Which type of programming device to use will vary with each application, and what is appropriate for one application may not be the most appropriate for another. The program (ladder diagram) is entered by pushing keys on the keyboard in a subscribed sequence, with the resultant circuit(s) displayed on the VDT of a desktop programmer, and with a LED or liquid crystal display for the hand-held programmer. The visual display can be used as an aid to test the circuit prior to entry into user memory, or for troubleshooting after the circuit is entered into memory and is operational. Contacts and coils are either intensified or displayed in reverse video to indicate power flow or logic continuity. Programming the PLC is not difficult, but time must be spent to become familiar with the specific PLC and its programming techniques.

Review Questions

1. Briefly describe the function of the *programming device*.
2. What does the acronym VDT stand for?
3. Define the term *on line*.
4. List the three type of programming devices and give advantages and disadvantages for each.
5. Define the term *off line*.
6. What is a hardware key?
7. What is meant by the term "third party software"?
8. What is the advantage of a sealed touch pad-type keyboard over a standard raised key keyboard?
9. In the context of this chapter, what do the terms *passing current* and *power flow* indicate?
10. Hand-held programmers usually have what type(s) of displays?

CHAPTER

5 Memory Organization

Objectives

After completing this chapter, you should have the knowledge to:

- Identify the two broad categories of memory and describe the function of each.
- Identify the types of information stored in each category of memory.
- Define the term *byte*.
- Define the acronym *bits*.
- Define *holding registers*.
- Understand the term *default*.

MEMORY WORDS AND WORD LOCATIONS

For the programmable controller to function properly and control a process or driven equipment, it must be able to perform the user program repeatedly and accurately. The system must also be able to perform its control function with great speed, which is achieved by processing all information in binary signals. The key to the speed with which binary information can be processed is that there are only two states, each of which is distinctly different. Binary signals fall into one of two states, which are 1 and 0. The 1 and 0 can represent *ON* or *OFF*, true or false, voltage or no voltage, high or low, or any other two conditions depending on the system. There is no in-between state or condition, and when information is processed, the decision is either *yes* or *no*. There is no *maybe*, *almost*, or any other alternative.

As indicated in Chapter 3, the processor memory consists of hundreds or thousands of locations that are referred to as words. Each word is capable of storing binary data in the form of binary digits, or **bits** (**BI**nary digi**TS**). A binary digit, like a binary signal, can only be a 1 or a 0. The number of bits that a word can store will depend on the system or PLC. Words can be made up of 32 bits, 16 bits, or 8 bits. The 16-bit word is the most common (Figure 5–1).

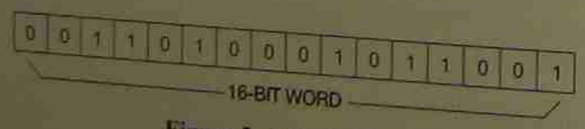


Figure 5–1 16-Bit Word

If a memory size is 256 words, then it can actually store 4,096 bits of information using 16-bit words (256 words × 16 bits per word) or 2,048 bits using an 8-bit word (256 words × 8 bits per word). When comparing memory sizes of different PLC systems, it is important to know the number of bits per word of memory. Bits can also be grouped within a word into **bytes**. A byte is a group of 8 bits.

So that information stored in each word can be located, each word is numbered or given an address. Addressing words in the memory serves the same function as the addresses used for homes or apartments. Word 100, for example, represents a specific word location in memory, just like N. 100 Lincoln represents the address of an apartment building. The bits in word 100 are found by referencing a given bit number, just like the occupant of the apartment complex is found by a given apartment number.

Since a bit of information can only be a 1 or a 0 (*ON* or *OFF*), how is the status of bits within a word determined? Words that store the status of individual bits for input devices are set to 1 (*ON*) or 0 (*OFF*), depending on the status (*ON* or *OFF*) of the input devices that the bit locations represent. Other bits are set to 1 or cleared to 0 by the processor in response to the logic of the user program, RELAY LADDER LOGIC, or special instructions, which, in turn, control the status (*ON* or *OFF*) of other bits that represent output devices.

A simple example of how this works is illustrated in Figure 5–2.

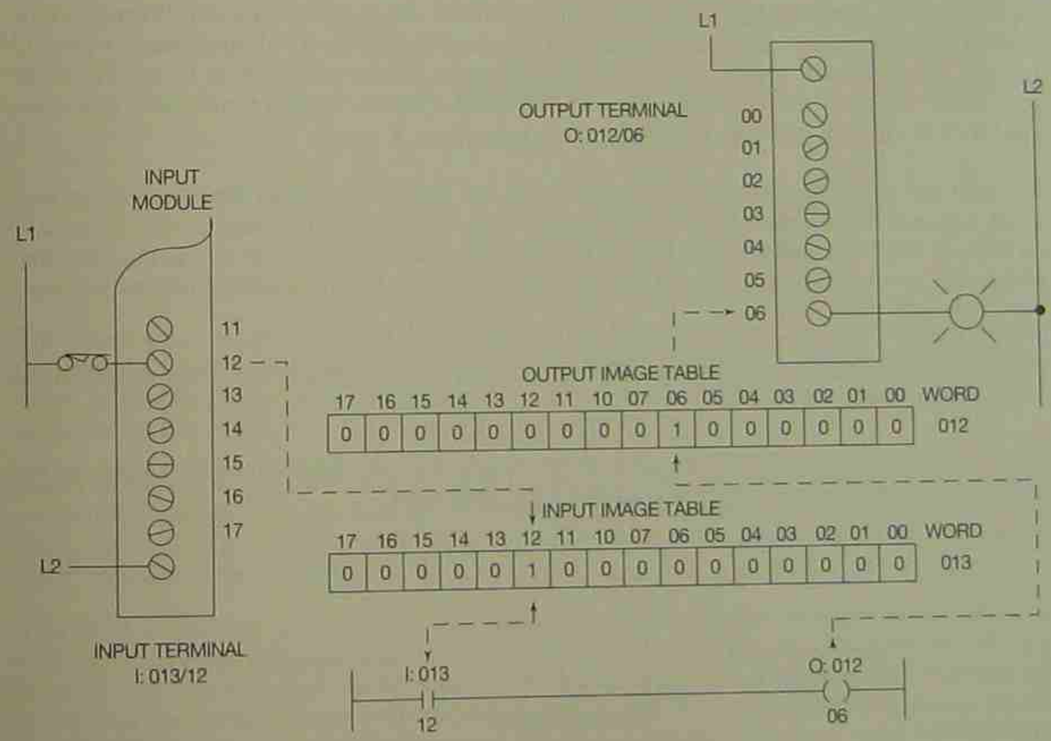


Figure 5–2 Relationship of Bit Address to Input and Output Devices

Note: *The example uses memory organization and addressing utilized by the Allen-Bradley PLC-5 family. While the example is specific to Allen-Bradley, the concepts illustrated are common to all PLCs. Allen-Bradley uses an octal numbering system to address bit locations. Notice that the 16 bits are numbered 00 through 07 and 10 through 17. In the octal numbering system, the numbers 8 and 9 are never used. The octal numbering system will be covered in detail in Chapter 6.*

Assume that when a given limit switch is closed, the closure will turn an indicator lamp *ON*. The limit switch is connected to an input module in the I/O rack, while the indicator lamp is connected to an output module. Chapter 2 discussed DIP switches that were set in a prescribed sequence to identify the I/O rack number for the processor, and that the location of each terminal point of each I/O module within the rack determined the address of a given device. In Figure 5–2, the limit switch is connected to terminal 12 on an input module, and is given an address of I:013/12. This indicates that bit 12 of input image table word 013 stores the status (*ON*-[1] or *OFF*-[0]) of the limit switch. The indicator lamp is connected to terminal 06 of an output module and is given an address of O:012/06. This address indicates that bit 06 of output image table word 012 controls the status (*ON*-1 or *OFF*-0) of the lamp.

By programming a simple circuit into the user memory of the processor as shown at the bottom of Figure 5–2, the processor controls the indicator lamp using the logic of the user program. The logic states that if contact I:013/12 closes, lamp O:012/06 should light, or go *ON*. When power is applied to the processor, the processor starts its scan and looks at bit 12 of input image word 013 to see if the bit is set to 1 or 0. If the limit switch is open, the bit will be set to 0, or *OFF*. If the limit switch is closed, as indicated in Figure 5–2, the input module sends a signal to the processor, and bit 12 of input image word 013 will be set to 1, or *ON*.

The next part of the scan solves the user program. The logic of the ladder diagram, or user program, indicates that when contact I:013/12 (bit 12 of input word 013) is closed, or *ON*, the indicator lamp O:012/06 should be turned *ON*. The processor reads the logic, and during the third step of the scan, sets bit 06 of output word 012 to 1, which turns the lamp connected to the output module terminal 06 *ON*.

The address I:013/12 also tells us that the limit switch is an input device, and is wired to terminal 12 of module group 3 of rack 1.

Figure 5–3 illustrates the significance of each letter/digit or group of digits used for addressing the Allen-Bradley PLC-5 family of programmable logic controllers.

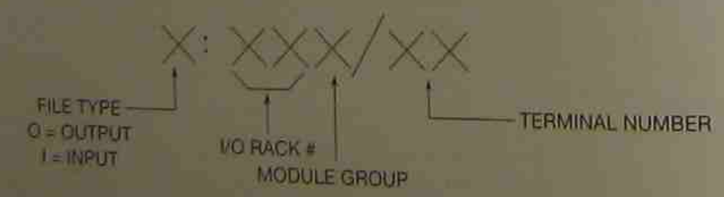


Figure 5–3 Allen-Bradley PLC-5 Address Format

The first letter is used to indicate the type of file (input, output, timer, counter, etc.) The letter I represents an input device, and the letter O represents an output.

The next two digits identify the rack number. One rack can control 128 I/O points. Rack numbers start at 00. A rack is different than a chassis. The chassis is the physical frame that actually holds the input and output modules that make up a rack. Depending on the density of the I/O modules used, a rack may require a 16-slot chassis or only 4 slots. If 8-point I/O are being used, it will take 16 slots of 8-point I/O modules to make 128 I/O points ($8 \times 16 = 128$), whereas if 32 point I/O are being used, it will only take a 4-slot chassis to make a rack ($32 \times 4 = 128$).

The next number identifies the module group within the rack. This is always a number from 0 through 7. The last two digits identify the actual terminal number to which the device is wired.

Figure 5–4 reviews the concept using the address of the limit switch I:013/12.

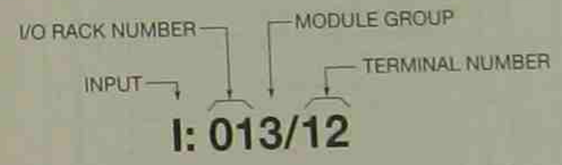


Figure 5–4 Limit Switch Address I:013/12

The letter I tells us that the address represents an input device. The next two digits, 01, tell us that the device is located in I/O rack number 01. The next digit, which is a 3, further identifies the location as module group number 3. The last two digits, 1 and 2, identify the actual terminal (12) on the input module to which the limit switch is connected.

Another example of this concept is shown in Figure 5–5. The limit switch address I:013/12 gives us a hardware location for an input device in rack 01, module group 3, terminal 12. This same address, I:013/12, tells us that the status (*ON* or *OFF*) or state of the limit switch is reflected by bit 12 of word 013 in the input image table.

This same addressing scheme gives us a hardware location for the indicator lamp addressed O:012/06. The letter O indicates an output device. The next two digits, 01, tell us that the I/O rack location is 01. The next digit identifies the module group as group 2. The last two digits locate terminal 06 as the terminal on the output module to which the indicator lamp is wired. Again, the address O:012/06 also locates the memory word and bit location that reflects the status (*ON* or *OFF*) of the indicator lamp as shown in Figure 5–6.

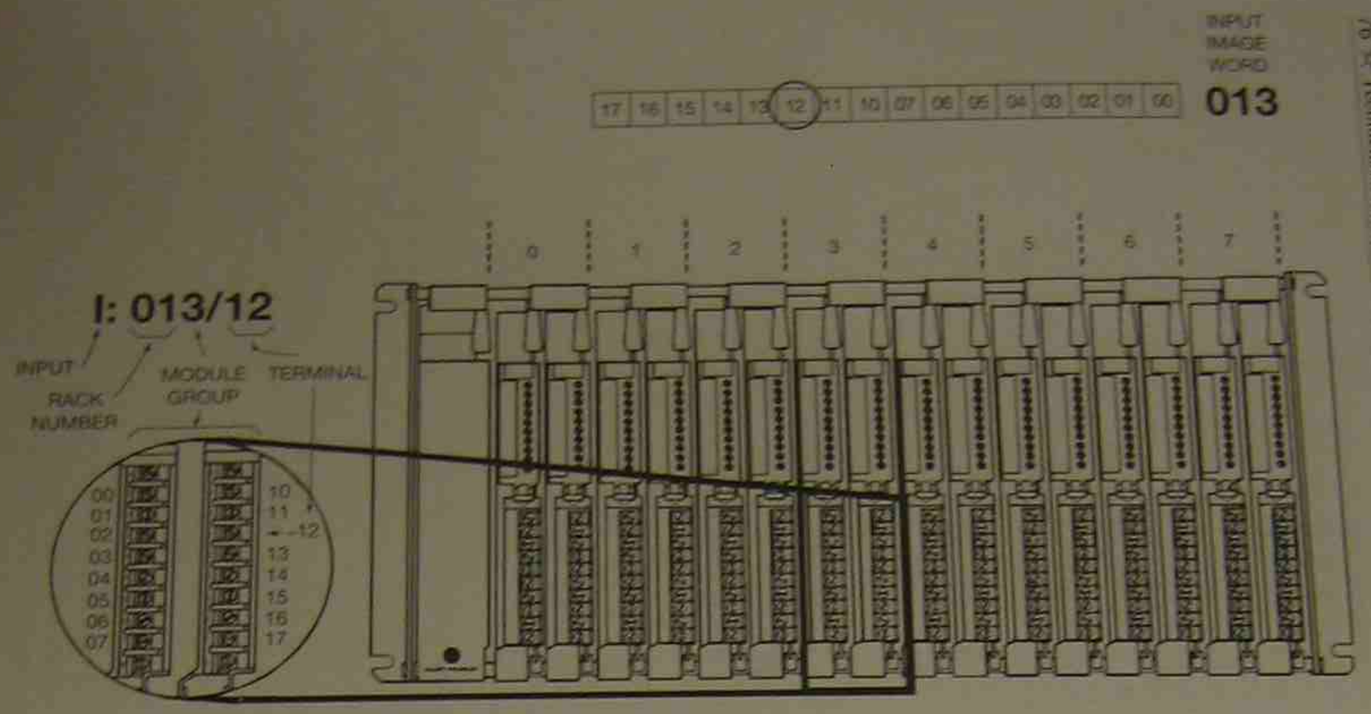


Figure 5–5 Relating Input Address I:013/12 to Actual Hardware Location
(Courtesy of Allen-Bradley)

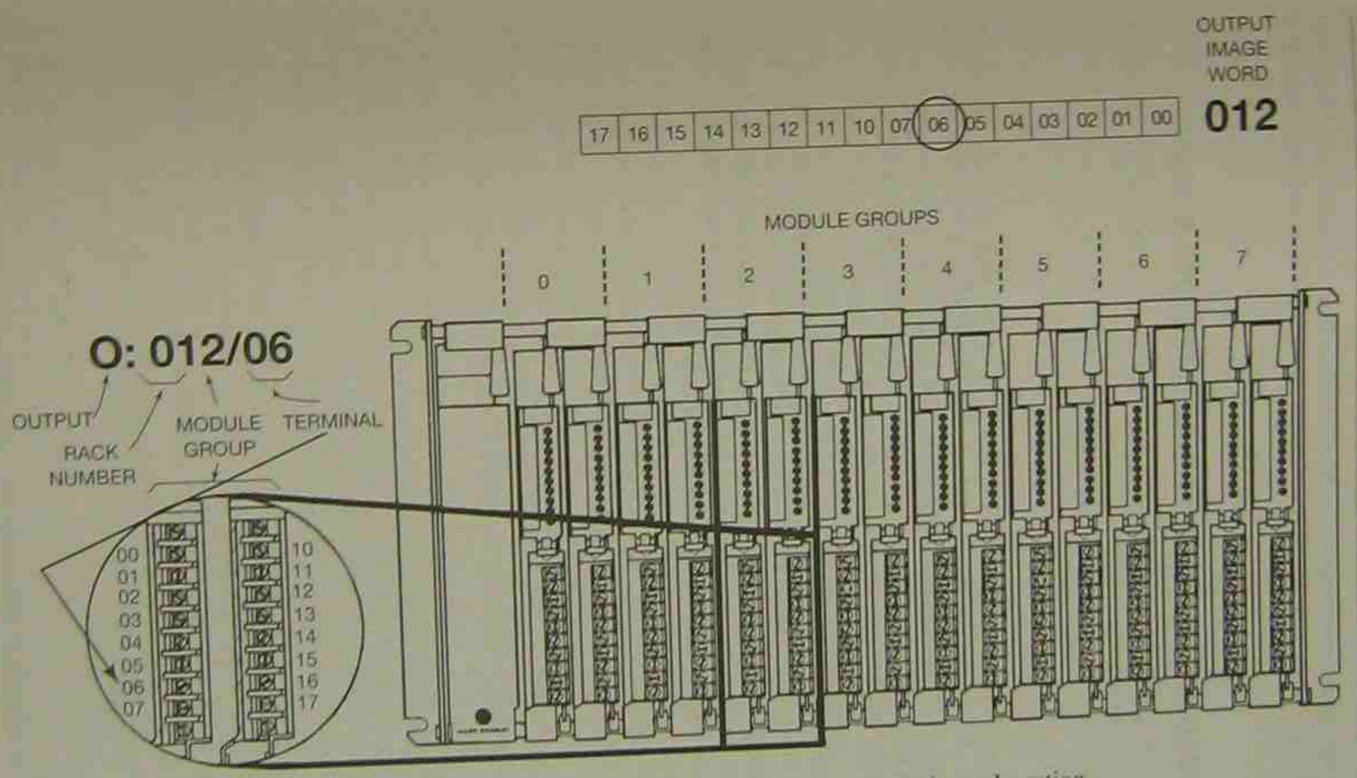


Figure 5–6 Relating Output Address O:012/06 to Actual Hardware Location
(Courtesy of Allen-Bradley)

While the address system discussed is specific to the Allen-Bradley PLC-5 family, most PLC manufacturers use an addressing scheme that identifies memory word locations, and may also give hardware locations.

SLC 500 AND MICROLOGIX 1000 AND 1500 ADDRESSING SCHEME

Like the PLC-5 family, the SLC 500 and MicroLogix family use the letter I for an input address and the letter O for an output address.

I = external input device
O = external output device

A typical address for an input device would be I:0.1/6. The I, of course, indicates that this is the address for an input device. The colon (:) is called an element delimiter. This means that the colon separates the input designator, I, from the rest of the address. The next character, the number 0, indicates the slot number that holds the actual input module. The slot number can range from slot 0, adjacent to the power supply in the first chassis, to a maximum of 30.

If the number of inputs is more than 16, a period (.), which is called a word delimiter, would be used after the slot number. The forward slash (/) is called the bit delimiter. The number that follows the forward slash is the bit number of the word as well as the terminal number of the I/O module. The digit 6 indicates the terminal number where the input device is wired. Figure 5–7 illustrates the addressing scheme for the SLC 500 and the MicroLogix family.

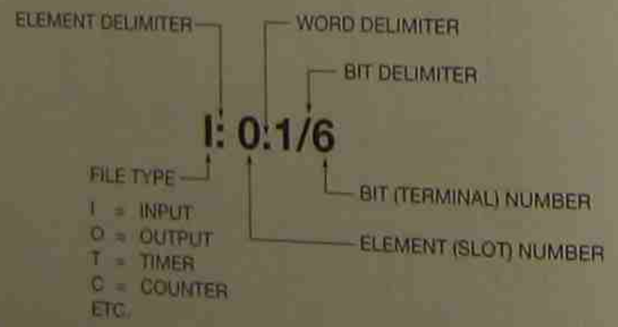


Figure 5–7 Allen-Bradley SLC 500 and MicroLogix Addressing Scheme

The first character of the address identifies the file type. The typical file types used are:

O	Output
I	Input
S	Status
B	Bit
T	Timer
C	Counter
R	Control
N*	Integer
F*	Float
St*	String
A*	ASCII

*Only available on selected processors.

In the address shown in Figure 5–7 the I indicates that the file type is an Input file and also indicates this is the address of an input device. The 0 after the :, which is the element delimiter, indicates that the input device is connected to slot 0. The period (.) after the slot number indicates that the inputs exceed 16 and require two words in the input image table. The number 1 indicates that this is word 1 in slot 0. The number 6 after the forward slash is the bit number.

Note: *The SLC 500 and MicroLogix do not use the octal numbering system like the PLC-5 family, but instead use the decimal numbering system.*

For 16 inputs, the bits are numbered 0–15. The address, by using the period word delimiter (.), indicates that there are more than 16 inputs installed in slot 0. Bit 6 is actually input device 22 (16 + 6 = 22). Because we started with bit 0, input device 22 is in reality the 23rd input device. Likewise, input address I:0.1/7 is the 24th input device.

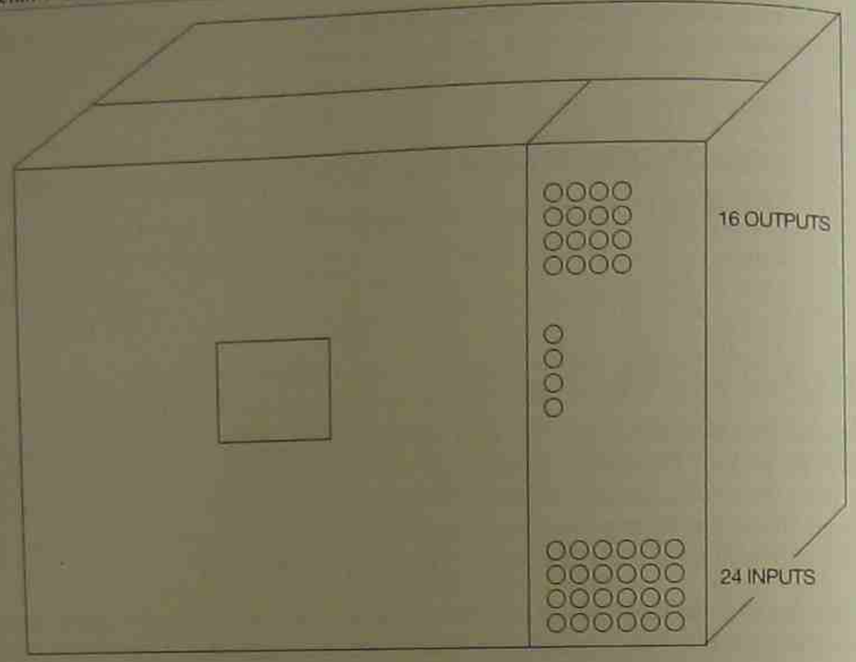
The SLC 500 controller is available in either fixed or modular I/O. The fixed I/O units have fixed I/O of 20 (12 inputs and 8 outputs), 30 (18 inputs and 12 outputs), and 40 (24 inputs and 16 outputs).

For fixed I/O controllers, all of the I/O are in slot 0. Figure 5–8 shows a fixed I/O controller that has 24 inputs and 16 outputs. Also shown are the input and output image tables for Data File 0 (output) and Data File 1 (input). Note that for 24 inputs, the Input Image table uses two words for slot 0. All 16 bits of word 0 (I:0) are used, whereas only the first 8 bits of word 1 (I:0.1) are used (bits 0 through 7). The unused bits of word 1—bits 8 through 15—are marked invalid and are not available for use.

An output address of O:0/4 indicates that this is the address of an output device in slot 0, terminal 4. This address also indicates that this is bit 4 of word O:0 in the Output Image table data file 0 as indicated by the *X* under bit 4 of output word O:0 (Figure 5–8).

An input address of I:0/15 indicates controller input 15 in slot 0. This address is indicated by the *X* placed under bit 15 of input word I:0 in the input image table (Figure 5–8).

An address of I:0.1/7 indicates an input device in slot 0, word 1, bit 7. This address is indicated by the *X* placed under bit 7 of input image Data File 1, word 1 in Figure 5–8. This address also indicates that this is input device 23. As we started with 0 as the first input device, the 23rd device is



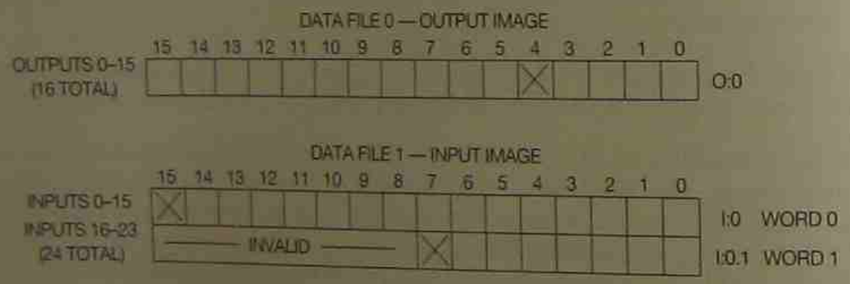


Figure 5–8 SLC 500 with 40 Fixed I/O

actually the 24th and last input device. The first word of the input image table, word I:0, held input device information for inputs 1 through 16, which are bits 0 through 15. The second word, word I:0.1, of the input image table holds the information for input devices 17 through 24, which are represented by bits 0 through 7. Bit 0 of word I:0.1 of Data File 1 represents input device 17. An alternate way to address input terminal 23 of slot 0 would be I:0/23.

Figure 5–9 shows an SLC 500 modular controller that consists of a seven-slot chassis interconnected to a ten-slot chassis.

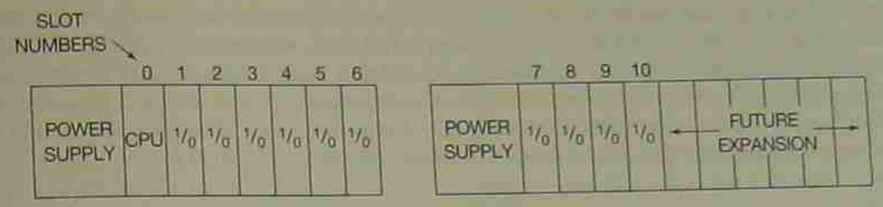


Figure 5–9 SLC 500 Modular Controller (Seven-Slot Chassis Connected to a Ten-Slot Chassis)

In this configuration, slot 0 contains the CPU, so slot 0 becomes an invalid I/O slot number. The controller is configured as follows:

SLOT	INPUTS	OUTPUTS
1	6	6
2	32	None
3	None	16
4	8	8
5	None	32
6	16	None
7	16	None
8	8	None
9	None	16
10	None	16

Figure 5–10 shows Data File 0 (the output image table) and Data File 1 (the input image table) as it would appear for the mix of I/O described above. Note that wherever 32-point I/O modules are used, the data table will require two 16-bit words. Slot 2 has a 32-point input module, and the input image table shows word I:2 (word 0 for inputs 0 through 15) and word I:2.1 (word 1 for inputs 0–15 which are actually inputs 17 through 32). Likewise, where 32 outputs are used in slot 5, the output image table for slot 5 shows that two words are used: O:5 and O:5.1.

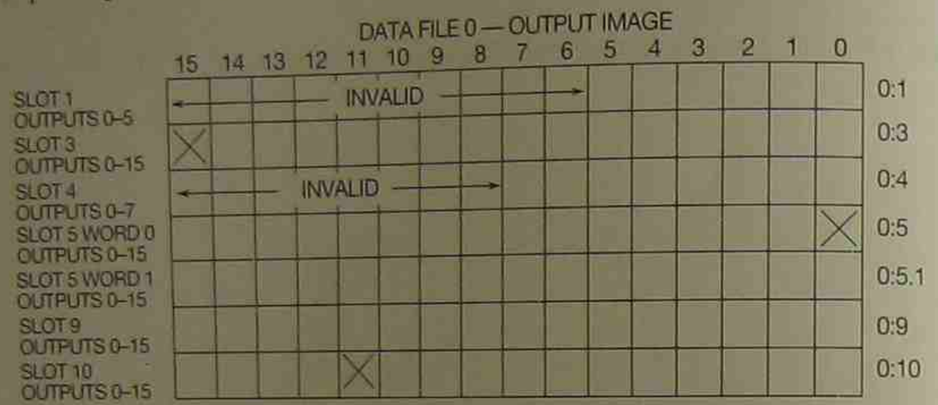


Figure 5–10 Output and Input Image Tables for the SLC 500 Configuration Shown in Figure 5–9

For the modular system shown in Figure 5–9, an address of O:3/15 indicates an output device 15 in slot 3. It also indicates bit 15 of word O:3 of the Output Image table, shown by an *X* under bit 15 of output word O:3 in Figure 5–10. An address of I:2.1/3 indicates input device 3 of word 1 in slot 2 (as indicated by an X under bit 3 of input word I:2.1). Slot 2 holds a 32-point input module which requires two words in the input image table: Word 0 (I:2) and Word 1 (I:2.1) shown in Figure 5–10. To further clarify the addressing scheme, address I:7/8 indicates input number 8 in slot 7 as indicated by the *X* under location 8 of slot 7 which is word I:7 of the input image table.

Note that slot 4 of both the input and output image tables has 8 inputs and 8 outputs. While only 8-bits of each word in the input and output image table are needed, the unused bits cannot be used for any other programming as indicated by the word *invalid* on the image tables.

Figure 5–11 shows an SLC 500 seven-slot chassis connected to a four-slot chassis using an Allen-Bradley 1746-C9 communications cable. The processor is installed in slot 0 of the first chassis which is adjacent to the power supply. Figure 5–11 shows an exploded view of the output/input module installed in slot 3. The output/input module is shown with the door that covers the terminals open. The inside of the door shows a pictorial view of the terminal layout. The first vertical row of terminal screws are for the six outputs plus an AC common connection. The second vertical row of terminals is for the six input devices and an AC common connection. The normally open pushbutton that is shown would have an address of I:3/0. I is for input, the 3 indicates that the input device is installed in slot 3, and the 0 indicates the input device is connected to input terminal 0.

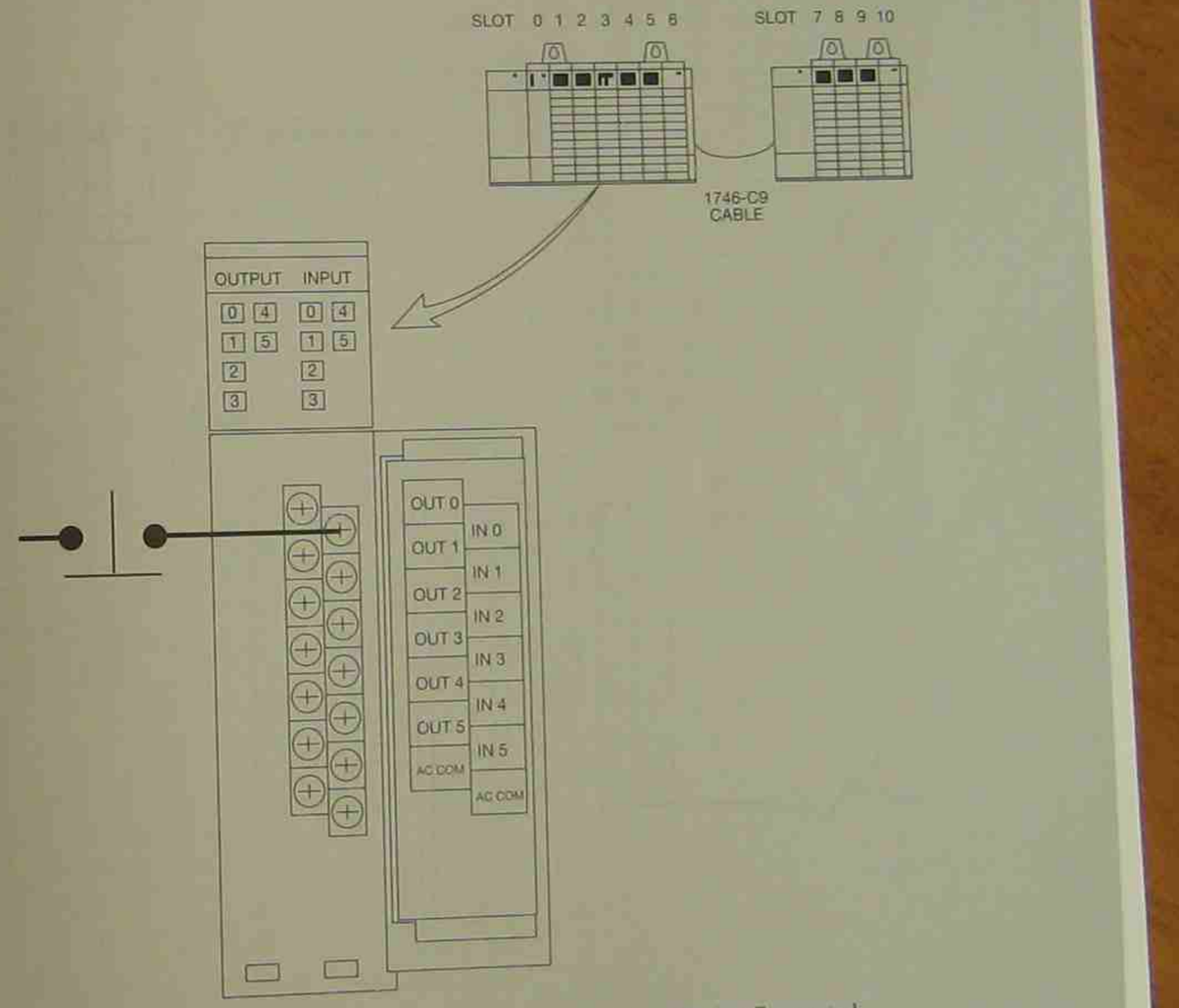


Figure 5–11 SLC 500 Modular Controller with Input Device Connected

Figure 5–12 shows the same seven-slot chassis connected to a four-slot chassis. In this figure, an output device (solenoid) is connected to one terminal of a 16-point output module that is installed in slot 8. As before, the terminal door is open and shows the layout of the terminals. The first terminal is for connecting the AC power. Notice that the output numbers alternate from Out 0, then Out 1, Out 2, and so on. The very last terminal is for the AC common, or neutral connection. The address for the solenoid connected as shown would be O:8/12. The O indicates an output device, the 8 indicates that the module is installed in slot 8, and the 12 indicates that the solenoid is connected to output terminal 12. This address also indicates that the status of the output device is found in bit 12 of Output Image Table word 8.

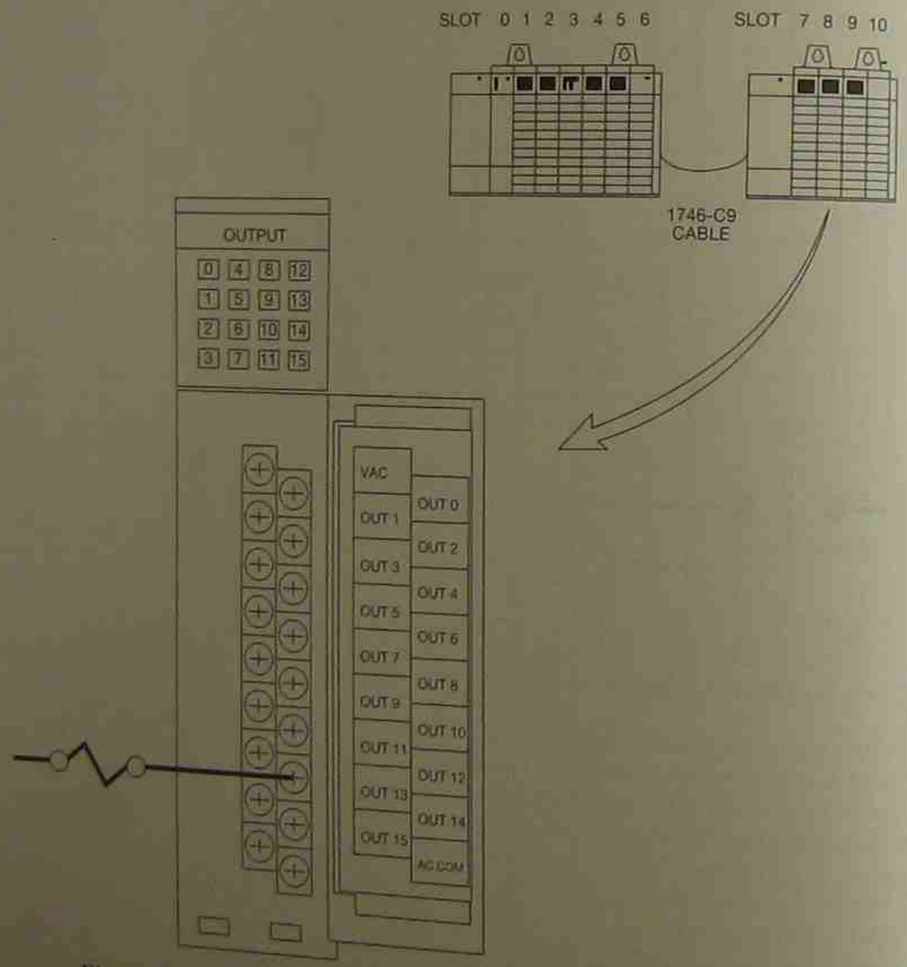


Figure 5–12 SLC 500 Modular Controller with Output Device Connected

Figure 5–13 shows an analog input module installed in slot 10. The exploded view of the module shows a sensor connected to the module at terminals 3 and 4. As connected, the address of the analog sensor is I:10/1. From the layout on the terminal door, we can see that terminal 3 and 4 are identified as Input 1+ and Input 1–.

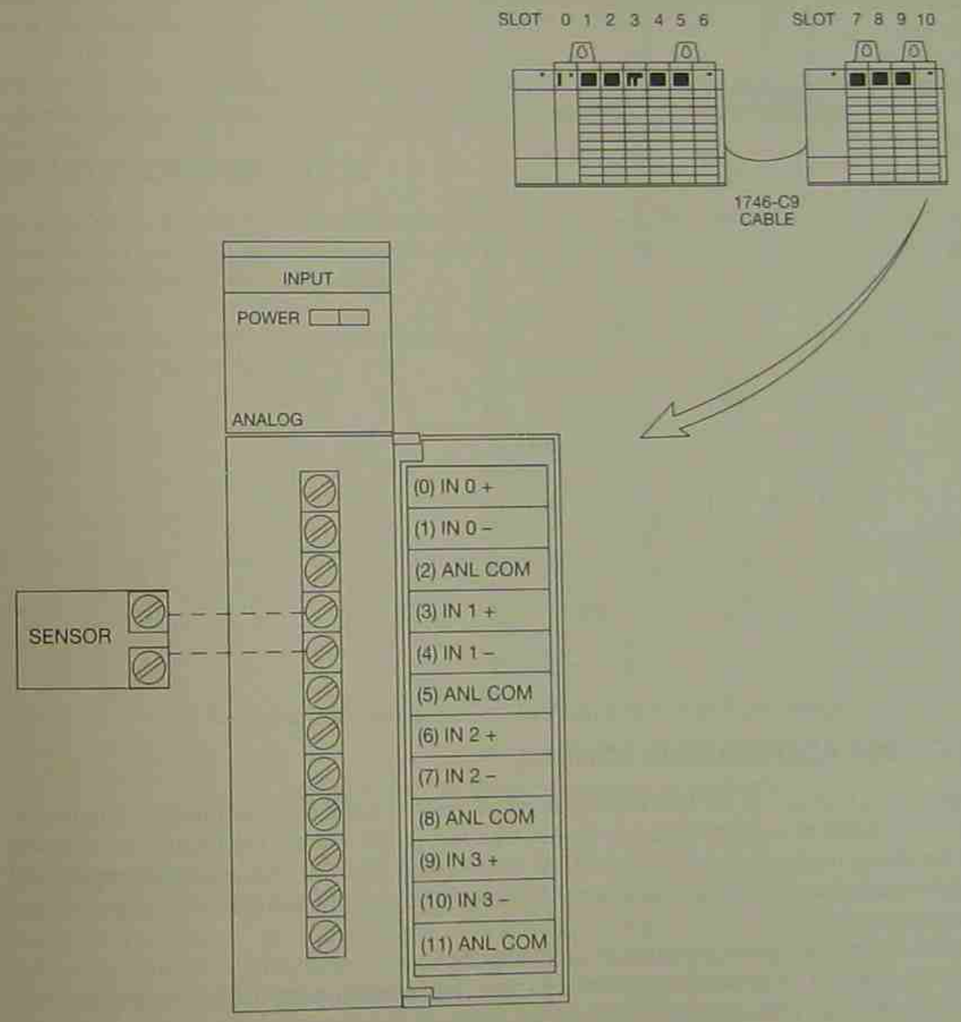


Figure 5–13 SLC 500 with Analog Input Device Connected

Figure 5–14 shows an analog output module installed in slot 6. The analog output in this illustration is an actuator connected to terminals 6 and 7. The address for the actuator is O:6/3.

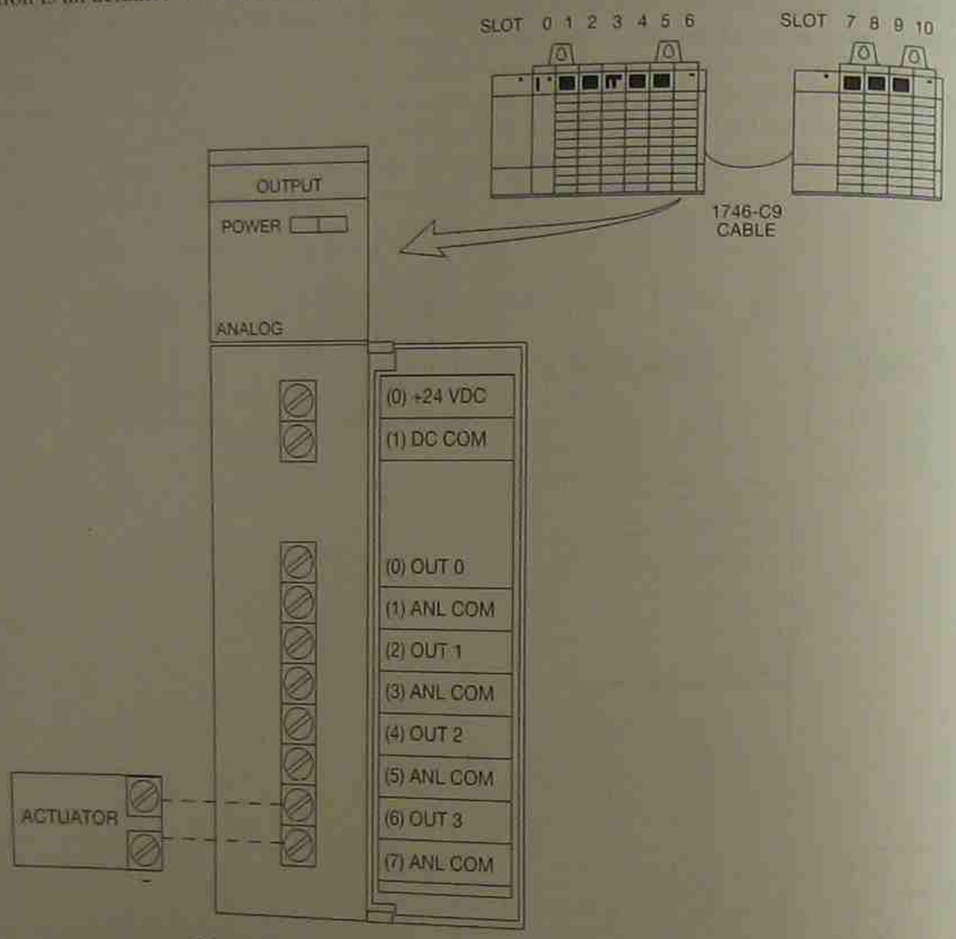


Figure 5–14 SLC 500 with Analog Output Device Connected

MODICON 984 ADDRESSING SCHEME

The Modicon 984 uses a 1 as the first number in an address for a discrete input device, and uses a 0 for the first number of a discrete output device. The rest of the address information is assigned by the programmer using a software utility called the I/O Map. The I/O Map is used to assign reference addresses to specific I/O modules, slots, and rack locations. By using the I/O Map software utility, the programmer defines:

- What slots in a given rack contain I/O modules.
- What type of I/O module is expected to be in a given slot, as opposed to what is actually there, as with the Allen-Bradley PLC-5 family.
- What addresses are to be assigned to which modules.

Instead of being hardware-driven like the Allen-Bradley addressing scheme, the 984 is software-driven and programmer-defined.

Discrete *input device* addresses use a five-digit number beginning with a 1 (1XXXX). The remaining four numbers that will make up the address are assigned by the programmer based on how the system is configured. Input registers are used for analog devices, absolute encoders, thermocouples, and the like, and these addresses will start with the number 3 (3XXXX).

Discrete *output devices* begin with the number 0 (0XXXX). For analog output devices such as meters, transducers, and variable speed drive motors, output registers are used and have addresses that start with the number 4 (4XXXX).

Once you gain experience using the I/O Map utilities portion of the Modicon programming software to assign addresses, the process becomes quite easy and very understandable.

MEMORY ORGANIZATION

At this point, it should be no surprise to find out that not all PLC manufacturers have organized their memories in the same way, or that they do not all use the same terminology for the configuration or make-up of their memories.

As discussed in Chapter 3, there are two general classifications of memory: storage memory and user memory (Figure 5–15).

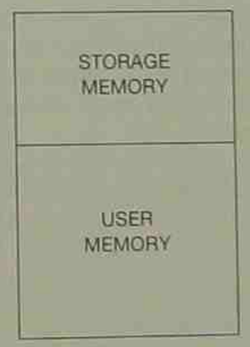


Figure 5–15 Two Broad Categories of Memory

Storage Memory

Storage memory is that portion of memory that will store information on the status of input and output devices, preset and accumulated values of timers and counters, internal relay equivalents, numerical values for arithmetic functions, and so on. The entire storage memory is called a data table, a register table, or other names, depending on the PLC manufacturer. A register is defined as an area for storing information (logic or numeric). Although the names or titles which are given to sections or subsections of the storage memory vary, the principles involved do not.

For example, the section of the memory that stores the status of the real-world input devices may be referred to as an input image table, input register, input status table, or external input section. No matter what name is used, the information is stored in the same way. The status (*ON* or *OFF*) of each input device is stored as either a 1 or a 0 (*ON* or *OFF*) in one bit of a memory word. When the processor is executing the user program (ladder diagram), it scans the input device status stored in the storage memory to determine which inputs are *ON* or *OFF*.

The section of storage memory set aside for output status may be referred to as the output image table, output register, output status table or external output section. Again, the name does not change the function of this section of the storage memory, or the method by which information is placed in memory for control of the actual output devices. As the processor executes the user program, it sends binary data (1s or 0s) to the output section of memory to control the output devices. Each output device is represented by one bit of a memory word.

Numeric information for timer or counter preset and accumulated values, arithmetic functions, sequencer functions, data manipulation, etc., uses a part of the storage memory that is called data registers or internal storage. Information is entered and stored in this part of memory using the binary, BCD, or hexadecimal numbering systems (the various numbering systems are covered in Chapter 6). The numbering system(s) used depend on the PLC hardware and system requirements. The storage of numeric information requires that several bits be used of one word to represent numbers. In a practical sense, any word used to store numerical information is not available for additional storage, even if all the bits of word are not used.

In general, any unused bits of a word that are not used for storing numeric values can be used as internal relays. Internal relays will replace the numerous control relays used in most hardwired control circuits. Many PLCs have a portion of memory set aside just for internal relays. Internal, or dummy relays, may also be programmed in the output section of memory when all the words or bits of words are not being used for real-world output devices. The concept and use of internal, or dummy relays, is covered later in the text.

Figure 5–16 shows the memory organization for an Allen-Bradley Mini PLC-2/15. The 2/15 is part of the PLC-2 family of Allen-Bradley processors. Allen-Bradley also has the newer PLC-5 family of processors.

Note: *While the PLC-2 family is the older version, there are literally thousands of this type still in use in industry today, and an understanding of their memory structure is meaningful.*

For the PLC-2 processors, Allen-Bradley uses 16-bit words. The word and bit addresses are numbered using the octal numbering system, as shown in Figure 5–11. The first eight words (000–007) of the data table are set aside for the processor. Words 010–017 and 020–026 are the output image table. An output device addressed 01000 is bit 00 of word 010; similarly, an address of 02617 is bit 17 of word 026. Note; word 027 is used by the processor for a *BAT LOW* condition, message generation, and data highway. The next 40-word section of memory, words 030–037, 040–047, 050–057, 060–067, and 070–077, is used for storing timer or counter accumulated values (numeric) or for internal storage.

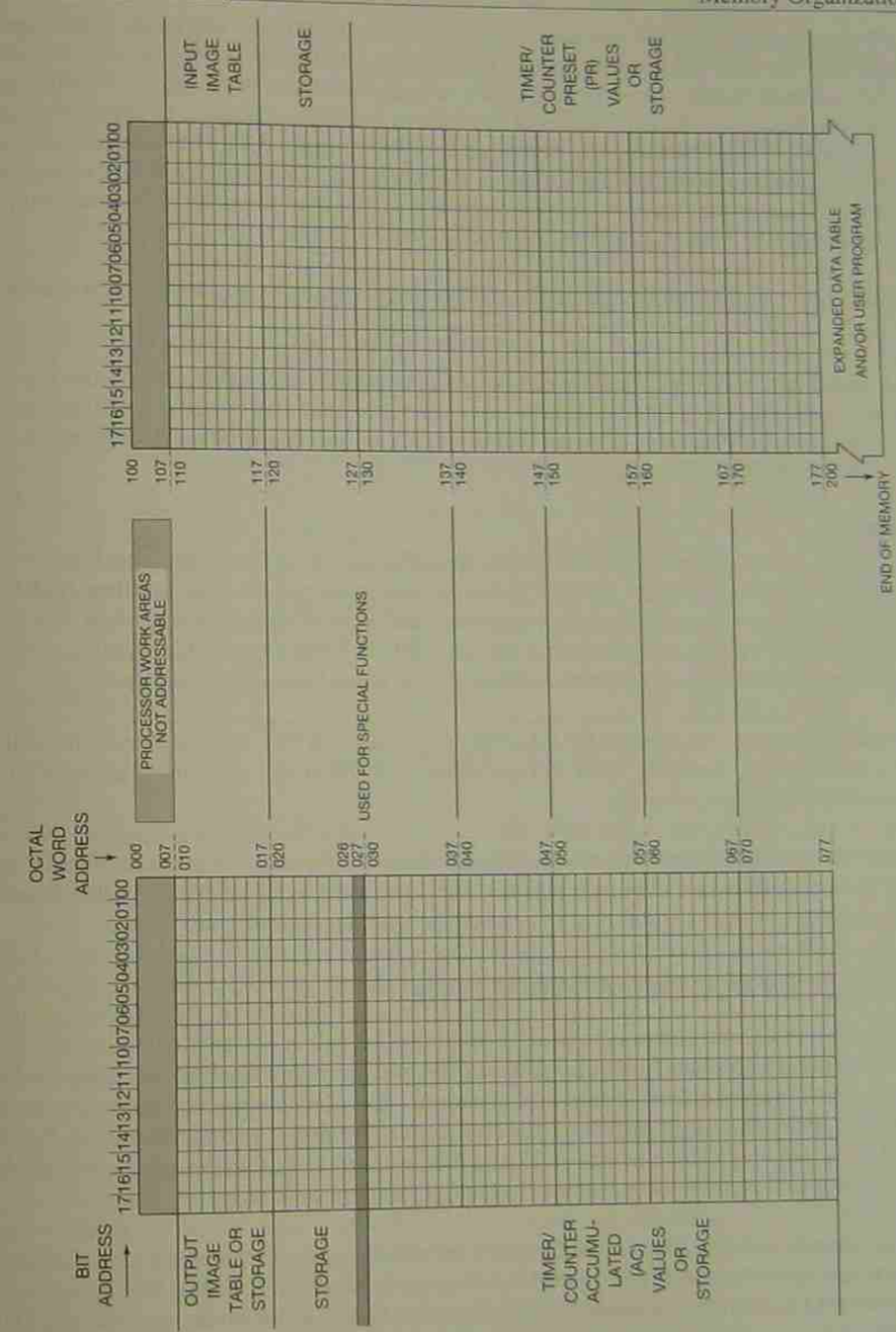


Figure 5–16 First 128 Words of an Allen-Bradley PLC-2/15 Memory

The next 8 words are the second processor work area. Words 110–127 (16 words) make up the input image table for discrete input addresses. Input address 12406 is bit 06 of word 124. Words 125 and 126 are used by the processor to indicate remote I/O faults. Words 130–177 (40 words) are used to store the preset values of timers or counters or for internal storage. If word 030 is used for a timer, word 130 automatically stores the preset value of timer 030. The accumulated value for timer 030 is stored in word 030. Any timer or counter in an Allen-Bradley PLC-2 system will store the preset value in the word that is numbered 100 higher than the timer or counter number. For example, counter 047 has its preset value stored in word 147.

Below the timer or counter preset values and internal storage section is the user program section of memory.

The actual configuration of the data table can be changed to meet user needs. Input/output image tables can be increased or expanded to handle more discrete inputs and outputs. Additional information on expanding the data table of Allen-Bradley PLCs is available from their local representative or from their technical publications.

User Memory

The user memory, or logic memory as it is sometimes called, is where the programmed ladder diagram is entered and stored. Within the user memory, words are set aside as **holding registers**. Holding registers typically store information generated and used by the processor when it is solving the user program. Holding registers that are set aside to store intermediate values or other short-term bits of information are sometimes referred to as *scratch areas* or *scratch pads*.

The user memory accounts for most of the total memory of a given PLC system. A system with an 8K memory (8192 words) typically has a storage memory of 2K or less, and the balance of memory (6K) is available for user memory.

Once the user program has been entered into the user memory, by either a programming device, tape loader, computer, or by means of a telephone interface, the programmable controller is ready to control the process or driven equipment in accordance with the user program logic.

ALLEN-BRADLEY PLC-5 FILE STRUCTURE

The Allen-Bradley PLC-5 processors are usually programmed with a personal computer and software specific for the PLC-5 family, and the areas of memory are often referred to as files, not tables, as is the case with the PLC-2 memory structure. Although there are still two memory sections (storage [data] and user [program]), the PLC-5 memory map, or structure, is very flexible in the way that the memory can be allocated. Figure 5–17 shows the PLC-5 **default** memory structure. Default refers to the initial value, setting, or configuration prior to any user changes.

In the data or storage memory section file 0 is the output image file. This file has 32 words of 16 bits each, and can hold the status of 512 real-world output devices (32 × 16). The status of the outputs (*ON* or *OFF*) is updated once each scan. On some PLC-5 models, like the PLC-5/25, the size of the file can be increased to accommodate more output devices.

	MAXIMUM NUMBER OF ELEMENTS		FILE NUMBER	DESIGNATION
DATA OR STORAGE MEMORY	32	OUTPUT IMAGE	0	O
	32	INPUT IMAGE	1	I
	32	STATUS (CLOCK FAULT TABLE, ETC.)	2	S
	1000	BIT (INTERNAL RELAYS)	3	B
	100	TIMER	4	T
	1000	COUNTER	5	C
	1000	CONTROL (MATH, SEQUENCER)	6	R
	100	INTEGER (WHOLE NUMBERS)	7	N
	1000	FLOATING POINT (DECIMAL NUMBER, I.E., 3.2)	8	F
USER OR PROGRAM MEMORY		ASSIGN FILE TYPE AS NEEDED	9-999	
		ASCII	0	A
		RESERVED	1	
		MAIN (LADDER LOGIC)	2	
		SUBROUTING	3	
		FAULT ROUTINE		
			999	

Figure 5–17 PLC-5 File Structure

File 1 is the input image file. This file, like file 0, has 32 words of memory and can store the status of 512 input devices. The status (*ON* or *OFF*) of the input devices, like the output image file, is updated once each scan and can be increased in size on many PLC-5 models.

Both files 0 and 1 use the octal numbering system, and the memory locations (bits) are also numbered using the octal numbering system (there are no 8s or 9s in the octal numbering system). The digits are 0–7, 10–17, 20–27, and so forth.

File 2 is the status, or S file. This file is used to store information on general processor status, fault codes, real time clock and calendar, major and minor fault bits, and program scan times in msecs. Information from this file is used or incorporated into the user program. The size of this file changes depending on the processor that is being used.

File 3 is the B, or bit file, and is used primarily for internal or dummy relays. The default size of this file is one word, but can be expanded to 1000 words if needed. All addresses from this file must start with B3. Another B or bit file may be created using the other areas of the memory. A B10 file could be created that would also have internal or dummy relays. The addressing B10 versus B3 is used for organization and ease of identification. The B3 file may be associated with one

piece of equipment, while the B10 file could be associated with another piece of equipment and/or operation. Up to 999 files can be created in a PLC 5/15 processor memory, as long as you do not try to allocate more memory than is physically available in the processor. The B3 file is typical of the remaining files in flexibility, as well as being addressed using the decimal numbering system.

File 4 is the T, or timer file. All timer addresses must start with T4 unless new timer files have been created (i.e., T 9, T 10, and T 11). When creating files, the same number cannot be used twice. If file 10 is used as a B file (B10), then file 10 cannot be used as a timer file. Each timer that is programmed uses three words of memory from its timer file.

File 5 is the C, or counter file. All counters that are programmed have C5 as the start of their addresses. Each counter, like timers, use three words of the counter file memory.

File 6 is the R, or control file. The words in this file are used with special functions like sequencer, file moves, word to file moves, and math functions. File 7 stores whole numbers (integers) and is called the N, or integer file. The integer file is used to store numeric values for data compare, arithmetic functions, and the like. For storing numbers with a decimal point, or floating point, file 8, the F file is used. Files 9–999 can be used or assigned as needed. They may be used to expand the size of input or output files, timer and counter files, etc.

The program portion, or user portion, of memory (Figure 5–12) is used to store information that relates to the user program, or for information that is needed for the processor to operate. File 0 is used to store ASCII information, while File 1 is reserved for internal use by the processor. File 2 is where the user program is stored in RELAY LADDER LOGIC. Files 3–999 are for storing subroutines, fault routines, and selectable timed interrupt (STI) as they are needed.

Figure 5–18 shows the data file structure used by the various PLC-5 models. Note that the I/O section varies from 32 words for the input image table and 32 words for the output image table for the PLC-5/10, 5/12, 5/15, 5/11, 5/20, and 5/20E, while there are 192 words for both the input and output image tables for the PLC-5/60, 5/60L, and 5/80.

The second column of the chart shows the file numbers for the default files. Files 0, 1, and 2 are fixed and cannot be changed. Files 3–8, however, can be changed from the default settings and used as required. For example, if one wanted to use file 3—the binary file—for a timer file, it would be necessary to delete the binary file. The binary file is deleted from the data table map screen and then used as a timer file. Because files 3–8 can be changed, files 3–999 can then be used for timer files, counter files, and the like. However, it is much easier to use files 9–999 when additional files are needed.

SLC 500 AND MICROLOGIX FILE STRUCTURE

When a PLC program for either the SLC 500, the MicroLogix 1000, or the MicroLogix 1500 is being developed, the information needed for the program to function properly is created and stored in processor files. These files are classified into two general types: Program Files (User) and Data Files (Storage).

*DATA TABLE FILES	FILE DESCRIPTION		NUMBER (Default file)	MAXIMUM SIZE OF FILE (16-BIT WORDS)				MEMORY USED	MEMORY USED
				PLC -5/10, -5/12, -5/15	PLC -5/11, -5/20, -5/20E	PLC -5/25	PLC -5/30 PLC -5/40, -5/40E, -5/40L	CLASSIC PLC-5 PROCESSORS	ENHANCED PLC-5 PROCESSORS
PLC-5 MEMORY data table program	OUTPUT	O	0	32	32	64	64 PLC -5/60, -5/60L, -5/80 128	2/file + 1/word	6/file + 1/word
	INPUT	I	1	32	32	64	64 128	2/file + 1/word	6/file + 1/word
	STATUS	S	2	32	128	32	128 128	2/file + 1/Word	6/file + 1/word
	BIT (BINARY)	B	3-999 (3)	1000				2/file + 1/word	6/file + 1/word
	TIMER	T	3-999 (4)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	COUNTER	C	3-999 (5)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	CONTROL	R	3-999 (6)	1000 structures of 3				2/file + 3/structure	6/file + 3/structure
	INTEGER	N	3-999 (7)	1000				2/file + 3/word	6/file + 3/word
	FLOATING POINT	F	3-999 (8)	1000				2/file + 2/float word	6/file + 2/float word
	ASCII	A	3-999	1000				2/file + 1/2 per character	6/file + 1/2 per character
	BCD	D	3-999	1000				2/file + 1/word	6/file + 1/word
	BLOCK TRANSFER[1]	BT	3-999	1000 structures of 6					6/file + 6/structure
	MESSAGE[1]	MG	3-999	585 structures of 56					6/file + 56/structure
	PID[1]	PD	3-999	399 structures of 82					6/file + 82/structure
	SFC STATUS[1]	SC	3-999	1000 structures of 3					6/file + 3/structure
	ASCI STRING[1]	ST	3-999	780 structures of 42					6/file + 42/structure
	EXTRA STORAGE		3-999						

[1]enhanced PLC-5 processors only.

Figure 5–18 Data Table Map File Structure for the PLC-5 Family
(Courtesy of Allen-Bradley)

Program files typically contain controller information (type of processor, I/O configuration, etc.), the ladder logic program, subroutine programs, and interrupt subroutines. The specific program files for the SLC 500 are shown in the tables below.

SLC 500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
Subroutine ladder program file (file 3-255)	Stores any subroutines not created in the main ladder diagram.

SLC 500 Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.
Floating Point (file 8)* *This file applies to selected SLC 500 processors.	Stores numbers with a decimal point or floating point.
String (user defined file)*	*This file applies to selected SLC 500 processors.
ASCII (user defined file)*	*This file applies to selected SLC 500 processors.

MicroLogix processors have the same program files as the SLC 500, and three additional files.

MicroLogix 1000 and 1500 Program Files	
System program file (file 0)	Used to store information about the processor and the I/O configuration.
Reserved file (file 1)	Reserved for internal use of the processor and is not user accessible.
Main ladder program file (file 2)	Stores the instructions entered by the user that determine controller operation.
User Error Fault Routine (file 3)	File is executed when a recoverable fault occurs.
High-Speed Counter Interrupt (file 4)	File is executed when a high-speed counter interrupt occurs. Can also be used for a subroutine ladder program.
Selectable Timed Interrupt (file 5)	Executed when a selectable time interrupt occurs. Can also be used for a subroutine ladder program.
Subroutine ladder program file (file 6-15)	Stores any subroutines that have been created in the main ladder diagram.

Note: *The MicroLogix 1000 and 1500 are set up to always have 15 files. These files cannot be deleted in the ladder program.*

MicroLogix Data Files	
Output (file 0)	Stores the status, *ON* or *OFF*, of output devices wired to the controller.
Input (file 1)	Stores the status, open or closed, of the input devices wired to the controller.
Status (file 2)	Stores controller operation information. This file is useful for troubleshooting controller and program operation.
Bit (file 3)	Stores the logic for internal or dummy relays.
Timer (file 4)	Stores the preset values, accumulated values, and status bits for timers.
Counter (file 5)	Stores the preset values, accumulated values, and status bits for counters.
Control (file 6)	Stores information on sequencers and shift registers.
Integer (file 7)	Stores numeric values.

In summary the SLC 500 memory is organized as follows:

Program Files	File Number
System Program	0
Reserved	1
Main Ladder Program	2
Subroutine Ladder Program	3-255

Data Files	Identifier	File Number
Output	O	0
Input	I	1
Status	S	2
Bit	B	3
Timer	T	4
Counter	C	5
Control	R	6
Integer	N	7
Floating Point (Selected model only)	F	8

User Defined Files (9-255)	Identifier
Bit	B
Timer	T
Counter	C
Control	R
Integer	N
Float	F
String	St
ASCII	A

MicroLogix 1000 and 1500	
Program Files	**File Number**
System Program	0
Reserved	1
Main Ladder Program	2
User Error Fault Routine	3
High-Speed Counter Interrupt	4
Selectable Timed Interrupt	5
Subroutine Ladder Program	6–15

Data Files	Identifier	File Number
Output	O	0
Input	I	1
Status	S	2
Bit	B	3
Timer	T	4
Counter	C	5
Control	R	6
Integer	N	7

The MicroLogix 1000 and 1500 programmable controllers use both volatile RAM and nonvolatile EEPROM memory. Program data downloaded into the processor from a programming device is stored first in the RAM memory and then copied to the EEPROM memory where it is stored as both backup data and retentive data. It is important to remember that all data in the RAM memory is lost in the event of a power failure, whereas information stored on the EEPROM memory is not affected by power loss.

While the names of files are different for each PLC manufacturer, the flexibility of assigning file areas and file size is typical for all PLCs. As the names and structure vary, the only way to really understand the memory structure is to obtain the literature for the specific PLC that you are dealing with. Salesmen, saleswomen and technical representatives are all invaluable resources when trying to gather information or clarification about a particular PLC.

Chapter Summary

All data, logic, and numerics are stored with binary digits that are represented as either a 1 or a 0. By storing binary data, the processor can rapidly scan and execute the user program and update the I/O section. I/O addresses in many cases not only identify the word and bit that is associated with the I/O, but also indicate hardware location (rack, module group, and terminal).

The names given to memory sections or subsections are unique to each PLC manufacturer, but the memories all work in basically the same manner.

The processor memory (storage and user) stores the I/O status, the user program, and numeric data used by the processor.

Review Questions

1. The following types of information are normally found and/or stored in one of the PLC's two memory categories (user and storage). Place an S (for storage memory) or a U (for user memory) before the information type to indicate in which category it is normally found and/or stored.
   a. status of discrete input devices
   b. preset values of timers and counters
   c. numeric values of arithmetic
   d. holding registers
2. Identify the following PLC-5 files:
   a. I b. O c. N
   d. S e. B f. T
   g. R h. F i. C
3. Define the term *byte*.
4. In a PLC-2, in what rack and module group number would address 11103 be located?
5. What word and bit number are represented by PLC-5 address O:010/01?
6. Using SLC 500 addressing, what do the following addresses indicate?
   a. O:3/15
   b. I:2.1/3
   c. O:5/0
   d. I:7/8
7. Using Allen-Bradley PLC-5 address format, what would address I:013/12 indicate?
8. Using SLC 500 or MicroLogix address format, what would address I:1.0/4 indicate?
9. Referring to Figure 5-14, what would address O:6/2 indicate?
10. The Modicon 984 PLC uses a _____ driven addressing scheme.

CHAPTER

6 Numbering Systems

Objectives

After completing this chapter, you should have the knowledge to:

- Understand decimal, binary, octal, hexadecimal, and binary coded decimal (BCD) numbering systems.
- Convert from one numbering system to another.
- Express negative numbers in 2s complement.
- Add signed numbers.
- Convert a negative binary display to its decimal equivalent.
- Complete a subtraction problem using 2s complement and addition.

An electrician, technician, or other personnel who are required to program, modify, or maintain a PLC must have a "working" knowledge of the different numbering systems that are used. For example, the input/output addresses may use the octal numbering system; the timer and counter addresses may use the decimal numbering system; accumulated and preset values of the timers and counters may use the binary numbering system; operator interfaces, such as thumbwheels and seven segment displays, may require information be sent and received using the BCD format; and the hexadecimal system may be used for loading information into sequencers. The numbering system used in each area discussed varies with the different PLC manufacturers, but it is obvious that to fully understand and program a PLC, an understanding of the various numbering systems is necessary.

DECIMAL SYSTEM

The decimal numbering system is used every day by electricians and technicians, and it is a system they are comfortable with. This system uses ten unique numbers, or digits, which are 0 through 9. A numbering system that uses 10 digits is said to have a base of 10. The value of the decimal number depends on the digit(s) used, and each's place value. Each position can be represented as a power of 10, starting with 10^0 as shown in Figure 6–1. In the decimal system, the first

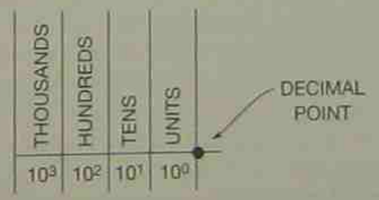


Figure 6–1 Place Value and Corresponding Power of Ten

position to the left of the decimal point is called the units place, and any digit from 0–9 can be used. The next position to the left of the units place is the tens place; next is the hundreds place, the thousands place, and so on, with each place extending the capability of the decimal system by ten, or a power often.

Note: *Any number that uses an exponent of 0, such as 10^0 has a place value of 1. Exponent 10^0 equals 1.*

A specific decimal number can be expressed by adding the place values as shown in Figure 6–2.

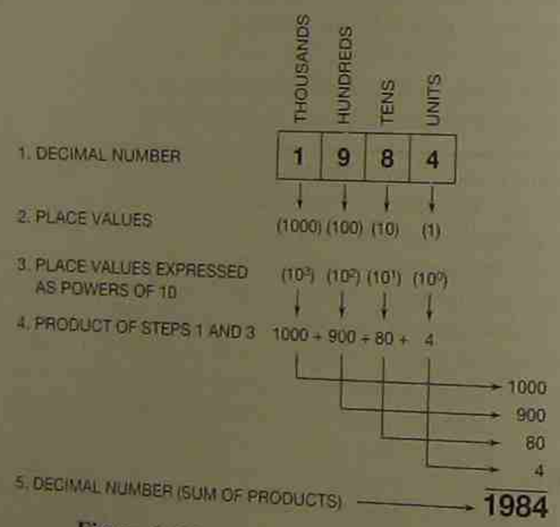


Figure 6–2 Decimal Numbering System

Mathematically, each place value is expressed as a digit number times a power of the base, or 10, in the decimal numbering system.

Another example is shown in Figure 6–3 using the decimal number 239.

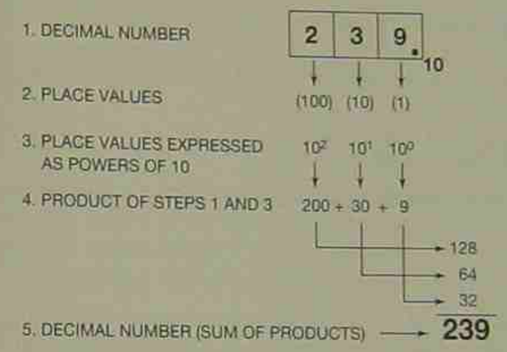


Figure 6–3 Decimal Numbering System

BINARY SYSTEM

The binary system uses only two digits: 1 and 0. Since only two digits are used, this system has a base of 2. Like the decimal system—and all numbering systems for that matter—each digit has a certain place value. The first place to the left of the starting point, or binary point, is the units or 1s location (base 2^0). The next place, to the left of the units place, is the 2s place, or base 2^1 as shown in Figure 6–4. The next place value is the 4s place, or base 2^2, then the 8s place, or base 2^3, and so forth. A binary number is always indicated by placing a 2 in subscript to the right of the units digit. Figure 6–4 illustrates how a binary number is converted to a decimal equivalent number. Note the subscripted 2 at the lower right-hand corner of the binary number line that indicates a base 2, or binary number.

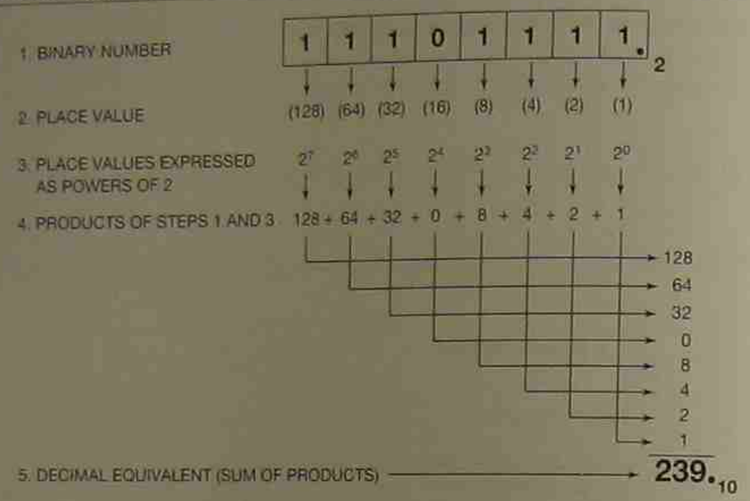


Figure 6–4 Converting a Binary Number to a Decimal Number

To convert a decimal number into a binary number, or to any numbering system for that matter, use the following procedure as shown in Figure 6–5. Divide the decimal number by the base you wish to convert to, in this case 2. The remainder is the 1s value (see Step 1 in the figure). Now divide the quotient from the first division again; the remainder becomes the value that is placed in the 2s location (see Step 2). The quotient of each preceding division is then divided by the base 2 until the base can no longer be divided (see Step 8), and the remainder (1) becomes the last digit in the binary number.

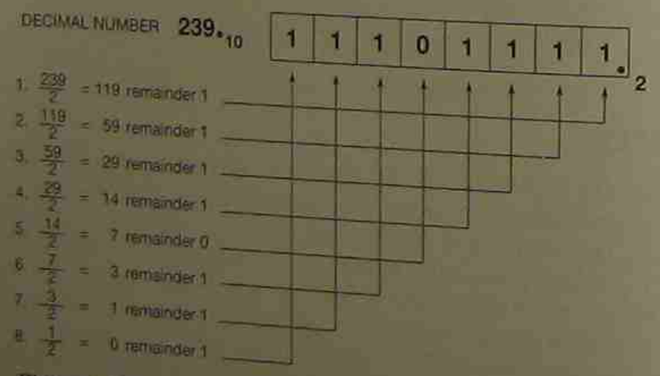


Figure 6–5 Converting a Decimal Number to a Binary Number

It is important to arrange the remainders correctly when making the decimal-to-binary conversion. The first digit placed in the 1s position is called the *least* significant digit, whereas the last digit is called the *most* significant digit. The last digit placed has the highest place value (128s) which is why it is called the most significant digit. This reference to least and most significant digits is common, and refers to the relative position of any given digit within a number.

The following steps summarize this decimal-to-binary conversion.

Step 1. The decimal number is divided by 2 (base of the binary numbering system). The quotient is listed (119) as well as the remainder (1).
Step 2. Divide the quotient of Step 1 (119) by base 2, and list the new quotient (59) and the remainder (1).
Step 3. Divide the quotient of Step 2 (59) by base 2, and list the new quotient (29) and remainder (1).
Step 4. Divide the quotient of Step 3 (29) by 2, and list the new quotient (14) and the remainder (1).
Step 5. Divide the quotient of Step 4 (14) by 2, and list the new quotient (7) and remainder (0).
Step 6. Divide the quotient of Step 5 (7) by 2, and list the new quotient (3) and remainder (1).
Step 7. Divide the quotient of Step 6 (3) by 2, and list the new quotient (1) and remainder (1).
Step 8. Divide the quotient of Step 7 (1) by 2, and list the new quotient (0) and remainder (1).

Note: *When using a calculator to do the division, the value to the* right *of the decimal must be multiplied by the base to get the actual remainder. For example, when 239 is divided by 2 (Step 1) on a calculator, the answer is 119.5. To find the actual remainder, the 0.5 is multiplied by 2, the base, to find the remainder 1. This procedure is true for any numbering system. The base times the value to the right of the decimal point equals the actual remainder.*

The binary numbering system is used to store information in the processor memory in the form of *bits* (BInary digiTS).

2S COMPLIMENT

Virtually all programmable controllers, computers, and other electronic calculating equipment perform counting functions using the binary system. For those PLCs that are programmed to perform arithmetic functions, a method of representing both positive (+) and negative (–) numbers must be used. The most common method is 2s complement. The 2s complement is simply a convention for binary representation of negative decimal numbers.

Before going any further with a discussion of 2s complement, a review of adding binary numbers may be helpful. In decimal addition, numbers are added according to an addition table. A partial addition table is shown in Fig. 6–6.

	0	1	2	3	4	5
0	0	1	2	3	4	
1	1	2	3	4	5	
2	2	3	4	5	6	
3	3	4	5			
4	4	5				

Figure 6–6 Decimal Addition System

To use the table, the first number to be added is located on the vertical line, and the second number on the horizontal line. The sum, or total, is found where the two imaginary lines intersect. For example, 3 + 2 = 5 (as shown in Figure 6–7).

	0	1	2	3	4	5
0	0	1	2	3	4	
1	1	2	3	4	5	
2	2	3	4	5	6	
3	3	4	⑤			
4	4	5				

Figure 6–7 Adding 2 and 3

For binary addition, a similar addition table is constructed. The table is small because the binary system only has two digits (1 and 0) (Figure 6–8).

	0	1
0	0	1
1	1	10

Figure 6–8 Binary Addition Table

To use the table, the first number (digit) to be added is located on the vertical line, the second digit is located on the horizontal line. The sum, or total, is found where the two imaginary lines intersect. Figure 6–9 shows an example of adding 1 + 0 = 1.

	0	1
0	0	1
1	①	10

Figure 6–9 Adding Binary 1 and 0

Notice that if 1 and 1 are added, the table shows 1 0, not 2, as might be expected. 1 0 is the binary representation of 2 (Figure 6–10).

8	4	2	1	
0	0	1	0	$_2$

Figure 6–10 Binary Representation of 2

Figure 6–11 shows how binary numbers 1011_2 and 110_2 are added.

	16	8	4	2	1	
	0	1	0	1	1	$_2$
+	0	0	1	1	0	$_2$
=	1	0	0	0	1	$_2$

Figure 6–11 Adding Binary Numbers

In the 1s column 1 + 0 = 1.
In the 2s column 1 + 1 = 0 with a carry over of 1.
In the 4s column 1 + 0 + 1 = 0, with a carry over of 1.
In the 8s column 1 + 1 + 0 = 0, with a carry over of 1.
In the 16s column 1 + 0 + 0 = 1.
The sum (total) of 1011_2 and 110_2 is, therefore, 10001_2.

To verify our results we can convert the binary numbers to decimal equivalent numbers and add them as shown in Figure 6–12.

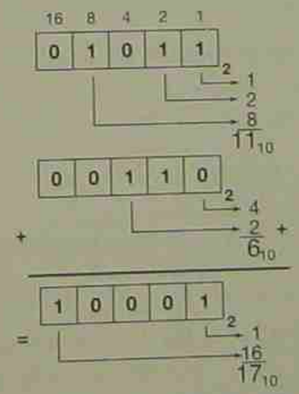


Figure 6–12 Converting Binary Numbers to Decimal Equivalents

Another example of adding binary numbers is shown in Figure 6–13 when 11011_2 and 11_2 are added.

$$\begin{array}{r} {\scriptstyle 1\,1} \\ 11011 \\ +\quad 11 \\ \hline 11110_2 \end{array}$$

Figure 6–13 Addition of Binary Numbers

In the 1s column 1 + 1 = 0, with a carry over of 1.
In the 2s column 1 + 1 + 1 = 1, with a carry over of 1.

Note: *1 + 1 + 1 = 3. The binary equivalent of* 3_{10} *is* 11_2.

In the 4s column 1 + 0 + 0 = 1.
In the 8s column 1 + 0 = 1.
In the 16s column 1 + 0 = 1.
The sum of 11011_2 and 11_2 is 11110_2.

To verify this method, convert the binary numbers to decimal numbers, and add them as shown in Figure 6–14.

$$\begin{array}{rcr} 11011_2 & = & 27_{10} \\ +\quad 11_2 & = & 3_{10} \\ \hline 11110_2 & = & 30_{10} \end{array}$$

Figure 6–14 Comparing Binary and Decimal Addition

To represent negative numbers using the binary numbering system, one bit is designated as a signed bit. If the designated bit is a 0 (zero), the number is positive, and if the bit is a 1, the number is negative.

Using a 4-bit word length, and using bit 4 as the designated signed bit, 0001_2 represents +1 decimal (see Figure 6–15).

Figure 6–15 Four-Bit Word with a Signed Bit

The table in Figure 6–16 shows all of the possible numbers for a 4-bit word using 2s complement.

BINARY NUMBER	DECIMAL
0111	+7
0110	+6
0101	+5
0100	+4
0011	+3
0010	+2
0001	+1
0000	0
1111	–1
1110	–2
1101	–3
1100	–4
1011	–5
1010	–6
1001	–7
1000	–8

Figure 6–16 2s Complement Numbers for an Eight-Bit Word

Notice that the negative numbers go to –8 while the positive numbers only go to +7. In this case, the signed bit is used for its place value, which is 8. The same holds true for 8- and 16-bit words. The maximum negative number is always one number *higher* than the maximum positive number.

To display a negative binary number requires that the same value positive number be complemented (all 1s changed to 0s and all 0s changed to 1s) and a value of 1 added. The result is 2s complement of the number. Figure 6–17 shows the steps to express –5 in 2s complement using a 4-bit word.

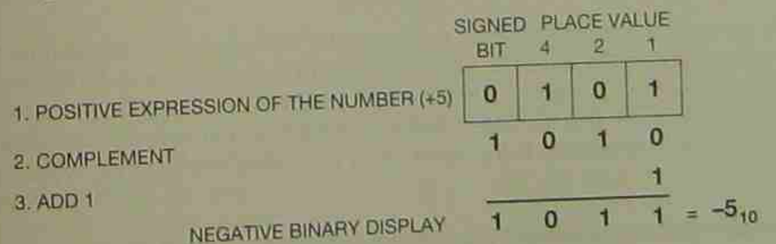


Figure 6–17 Expressing –5 in 2s Complement

Another example of 2s complement is shown in Figure 6–18 with the steps required to express –7 in 2s complement.

	SIGNED BIT	PLACE VALUE 4	2	1	
1. POSITIVE EXPRESSION OF THE NUMBER (+7)	0	1	1	1	
2. COMPLEMENT	1	0	0	0	
3. ADD 1				1	
NEGATIVE BINARY DISPLAY	1	0	0	1	$= -7_{10}$

Figure 6–18 Expressing –7 in 2s Complement

To convert a negative binary number to the decimal equivalent, the negative binary display is complemented, 1 is added, the binary sum is converted to decimal, and the negative sign (–) is added. Figure 6–19 shows what steps are necessary to determine the negative value of 1110.

	SIGNED BIT	PLACE VALUE 4	2	1	
	1	1	1	0	
1. COMPLEMENT	0	0	0	1	
2. ADD 1				1	
BINARY SUM	0	0	1	0	
3. ADD NEGATIVE SIGN					-2_{10}

Figure 6–19 2s Complement to Decimal Equivalent

An easy way to convert a negative binary number to its equivalent negative decimal number is to subtract the place value of the signed bit from the value of the binary digits. In Figure 6–19, the binary sum 1110 is equal to -2_{10}, which is the decimal number –2. In this example, a four-bit word is used with bit 4 being the signed bit. The fourth bit normally would have a place value of 8. In this example, the value 8 is subtracted from the value of the binary number 110_2 which is equal to 6. 6 – 8 = –2.

Another example using this method of converting negative binary numbers to their decimal equivalent is shown in Figure 6–20 using an 8-bit word with bit 8 being the signed bit.

PLACE VALUE	128	64	32	16	8	4	2	1
	1	1	0	1	1	1	1	0

SIGNED BIT

Figure 6–20 Eight-Bit Word 2s Complement

In this example, the signed bit would have a value of 128 (2^7), whereas the numeric value of the other bits would be 94 as shown in Figure 6–21.

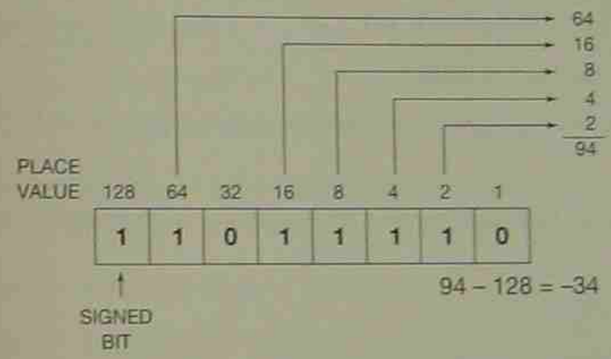


Figure 6–21 Eight-Bit Word (2s Complement) with Bits Added Together

Subtracting the place value of the signed bit (128) from the numeric value of the other bits (94) gives us the decimal number: 94 – 128 = –34. To verify this answer, complement the original binary number 1101 1110 to get 0010 0001. Then add 1.

$$\begin{array}{r} 0010\ 0001 \\ +\quad\quad 1 \\ \hline 0010\ 0010 \end{array}$$

The answer is $2^5 + 2^1$ or 32 + 2 = 34; adding the negative sign gives us a final answer of –34, the same answer we got when we subtracted the place value of the signed bit (128) from the numeric value of the binary number 94.

If the PLC system uses 2s complement for arithmetic, the highest positive number that a 16-bit word can represent is +32,767 as shown in Figure 6–22.

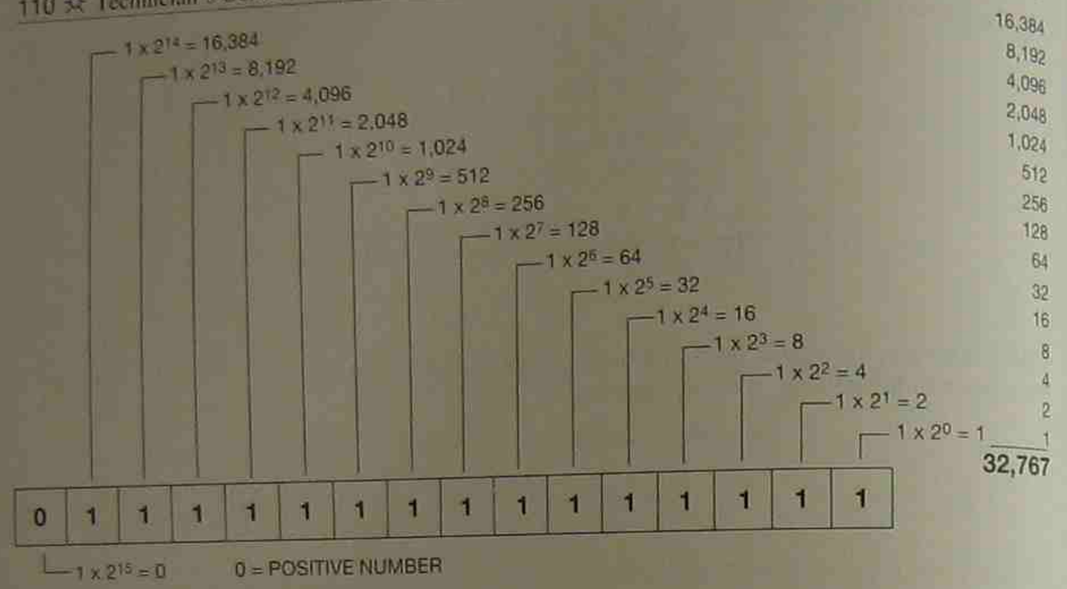


Figure 6–22 Maximum Positive Value of 2s Complement 16-Bit Word

The largest negative number that can be represented by a 16-bit word is –32,768 as shown in Figure 6–23.

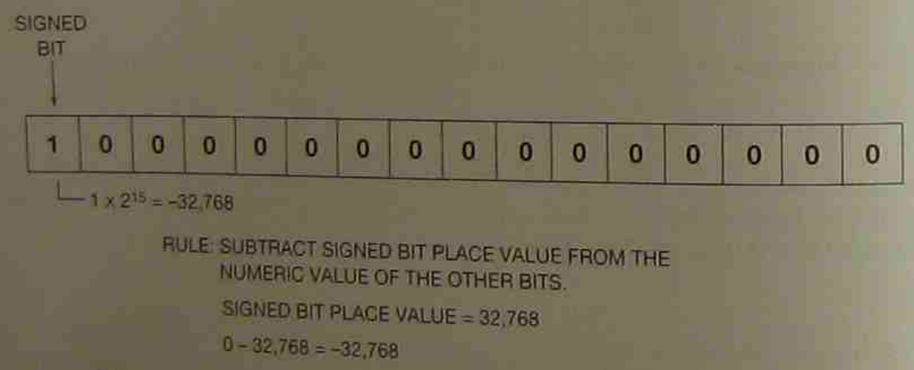


Figure 6–23 Maximum Negative Value of 2s Complement 16-Bit Word

Another method of converting positive numbers to 2s complemented negative numbers is as follows: starting at the least significant bit and working to the left, copy each bit up to and including the first 1 bit, and then complement or change each remaining bit. Figure 6–24 shows this alternate method of expressing –2 in 2s compliment using a 4-bit word.

1. ORIGINAL POSITIVE NUMBER (+2)	0	0	1	0
2. COPY UP TO FIRST 1 BIT			1	0
3. COMPLEMENT THE REMAINING BITS	1	1	1	0
2S COMPLEMENT –2	1	1	1	0

Figure 6–24 Alternate Method of 2s Complement

A further example is shown in Figure 6–25 for 2s complementing the value 24 using an 8-bit word.

1. ORIGINAL POSITIVE NUMBER (+24)	0	0	0	1	1	0	0	0
2. COPY UP TO FIRST 1 BIT					1	0	0	0
3. COMPLEMENT REMAINING BITS	1	1	1	0	1	0	0	0
2S COMPLEMENT –24	1	1	1	0	1	0	0	0

Figure 6–25 2s Complement of –24 Decimal

By using 2s complement, negative and positive values can now be added. The two steps for adding -7_{10} and $+5_{10}$ using 2s complement with a 4-bit word is shown in Figure 6–26.

	BIT	4	2	1	
1. EXPRESS –7 IN 2S COMPLEMENT					
a. POSITIVE EXPRESSION OF NUMBER (+7)	0	1	1	1	
b. COMPLEMENT	1	0	0	0	
c. ADD 1				1	
NEGATIVE BINARY DISPLAY	1	0	0	1	
2. ADD –7 AND +5					
	1	0	0	1	
	+ 0	1	0	1	
BINARY SUM	1	1	1	0	$= -2_{10}$

Figure 6–26 Adding Positive and Negative Numbers

Note: *When adding signed binary numbers, any carry over from the signed bit column is discarded.*

Once addition of signed numbers is possible, the other arithmetic functions (subtraction, multiplication and division) are also possible, because they are achieved by successive addition on a PLC.

EXAMPLE: Subtracting the number 20 from 26 is accomplished by complementing 20 to obtain –20, and then performing addition.

Subtracting 20 from 26 by complementing 20 and performing addition using an 8-bit word is shown in Figure 6–27.

	SIGNED BIT	PLACE VALUE 64	32	16	8	4	2	1
+26	0	0	0	1	1	0	1	0
+ –20	1	1	1	0	1	1	0	0
$+6_{10}$	0	0	0	0	0	1	1	0

Figure 6–27 Subtraction by Addition

OCTAL SYSTEM

The octal system, or base 8, is made up of eight digits: numbers 0 through 7. The first digit to the *left* of the octal point is the units place, or 1, and has a base or power of 8^0. The next place is eights (8s) or base 8^1. The next place is sixty-fours (64s) or base 8^2, followed by five hundred twelves (512s) or base 8^3, and four thousand ninety-sixes or base 8^4, and so on. An octal number will always be expressed by placing an eight in subscript to the right of the units digit as shown in Figure 6–28.

$357._8$

Figure 6–28 Octal Number

The method of converting an octal number to a decimal equivalent number is illustrated in Figure 6–29.

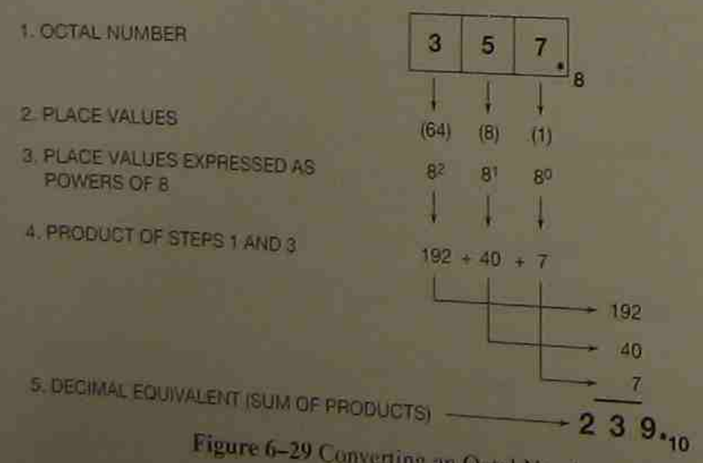


Figure 6–29 Converting an Octal Number to a Decimal Number

The decimal number 239 is converted to an octal number in Figure 6–30.

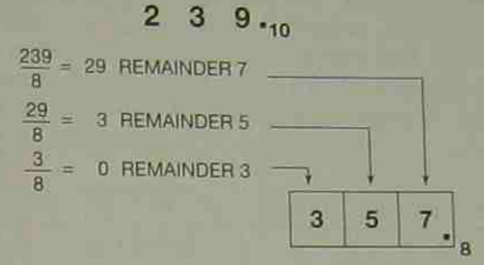


Figure 6–30 Converting a Decimal Number to an Octal Number

Step 1. The decimal number 239 is divided by 8 (base for the octal numbering system). The quotient is listed (29) as well as the remainder (7). A calculator shows the answer as 29.875. The quotient is 29, and the remainder is 0.875 × 8, or 7.

Step 2. Divide the quotient of Step 1 (29) by 8, and list the new quotient (3) and the remainder (5). A calculator gives the answer 3.625. The quotient is 3, and the remainder is 0.625 × 8, or 5.

Step 3. Divide the quotient from Step 2 (3) by 8, and list the new quotient (0) and remainder (3). The quotient 3 divided by 8 equals 0.375. The new quotient is 0, and the remainder is 0.375 × 8, or 3.

The decimal number 239 is the same as octal number 357.

Since the largest single number that can be expressed using the octal numbering system is seven (7), each octal digit can be represented by using only three (3) binary bits (base 2). Figure 6–31 illustrates how to convert an octal number to a binary number. The figure shows three sets of binary bits and the place value of each bit. For the *least* significant digit (7), a one (1) must be placed in the 1s place, the 2s place, and the 4s place to equal 7. The middle digit (5), a 1, is placed in the 1s place and the 4s place, while a 0 is placed in the 2s place. This combination equals 5. For the *most* significant digit (3), a 1 is placed in the 1s place and the 2s place, and a 0 is placed in the 4s place. This combination adds up to 3.

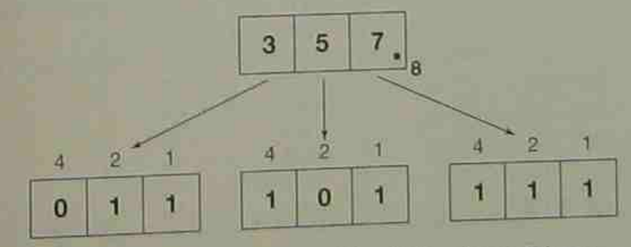


Figure 6–31 Conversion of Octal Number to Binary

Allen-Bradley uses the octal numbering system for I/O addressing and for numbering words and bits for the PLC-2 and PLC-5 families. The terminals of the input and output modules are labeled 00 through 07 and 10 through 17, rather than 0 through 15 as would be the case with decimal numbering that is used for the SLC 500 and MicroLogix PLCs. When using the octal numbering system, words are labeled 000–007, 010–017, 020–027, and so forth, whereas the bits are labeled 00–07 and 10–17. Figure 6–32 shows a memory word with the internal bits addressed using the octal numbering system.

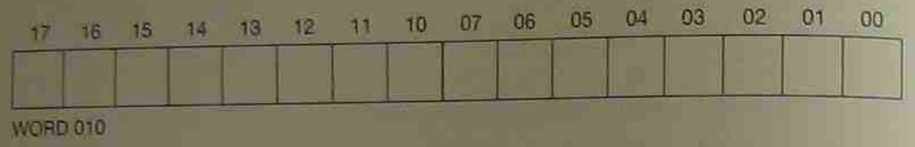


Figure 6–32 Word and Bit Labeling Using the Octal Numbering System

HEXADECIMAL SYSTEM

The hexadecimal system, often referred to as HEX, consists of a number system with base 16.

It seems logical that the numbers used in base 16 would be 0 through 15. However, only numbers 0 through 9 are used, and the letters A through F represent numbers 10–15, respectively. The place values from the hexadecimal point are 1s—16^0, 16s—16^1, 256s—16^2, 4096s—16^3, and so on.

Each hexadecimal digit is represented by four (4) binary digits. The binary equivalents are shown in the table in Figure 6–33.

HEXADECIMAL	BINARY	DECIMAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Figure 6–33 Hexadecimal Equivalents for Binary and Decimal

The decimal number 4,780 is converted to hexadecimal as illustrated in Figure 6–34.

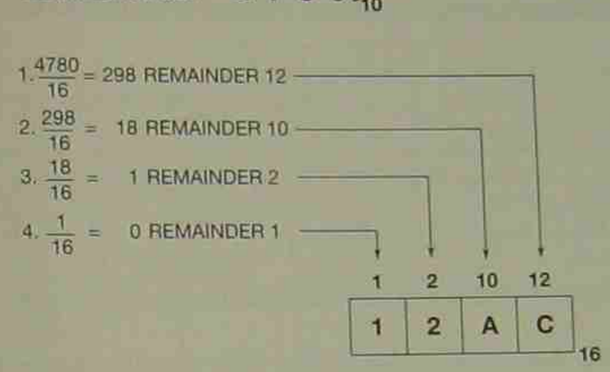


Figure 6–34 Converting a Decimal Number to a Hexadecimal Number

Step 1. The decimal number is divided by 16 (base for the hexadecimal numbering system). The quotient is listed (298) as well as the remainder (12). A calculator provides the answer 298.75. The quotient is 298, and the remainder is 0.75 × 16, or 12.

Step 2. Divide the quotient of Step 1 (298) by 16, and list the new quotient (18) and the remainder (10). The answer is 18.625. The quotient is 18, and the remainder is 0.625 × 16, or 10.

Step 3. Divide the quotient from Step 2 (18) by 16, and list the new quotient (1) and the remainder (2). Eighteen divided by 16 equals 1.125. The quotient is 1, and the remainder is 0.125 × 16, or 2.

Step 4. Divide the quotient from Step 3 (1) by 16, and list the new quotient (0) and the remainder (1). One divided by 16 equals 0.0625. The quotient is 0, and the remainder is 0.0625 × 16, or 1.

Converting a hexadecimal number to a decimal number is illustrated in Figure 6–35.

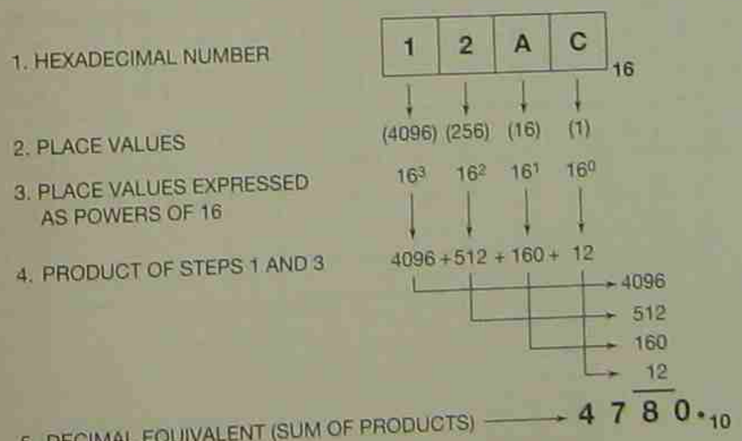


Figure 6–35 Converting a Hexadecimal Number to a Decimal Number

Note: *Remember that A is equivalent to 10, and C is equivalent to 12.*

The binary equivalent of the hexadecimal number 12AC is shown in Figure 6–36.

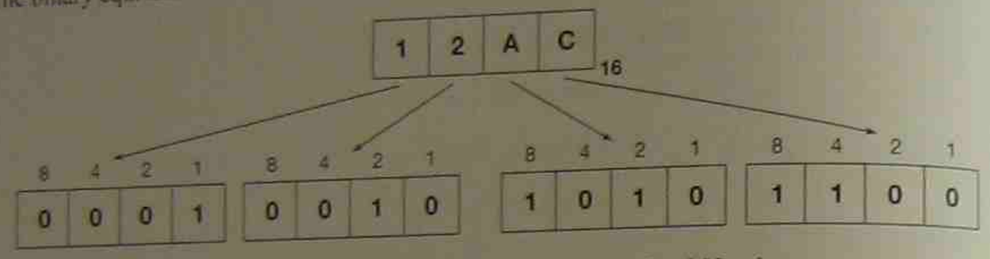


Figure 6–36 Binary Equivalent of a Hexadecimal Number

Since the largest number that can be displayed using the hexadecimal numbering system is 15, or F (as shown in the table in Figure 6–33), only four binary bits are needed to display each hexadecimal digit. The conversion to binary simply places the 1s in the correct binary locations to duplicate the hexadecimal digit (1 through F), as illustrated. The C has a value of 12, so a 1 is placed in the 8s and 4s location, while zeros (0) are placed in the 2s and 1s location for a total binary value of 12. The same procedure is followed for the remaining digits A (10), 2, and 1.

The HEX system is used when large numbers need to be processed. The hexadecimal system is also used by some PLCs for entering output instructions into a sequencer.

Figure 6–37 shows the conversion of a 16-bit binary number to its hexadecimal equivalent.

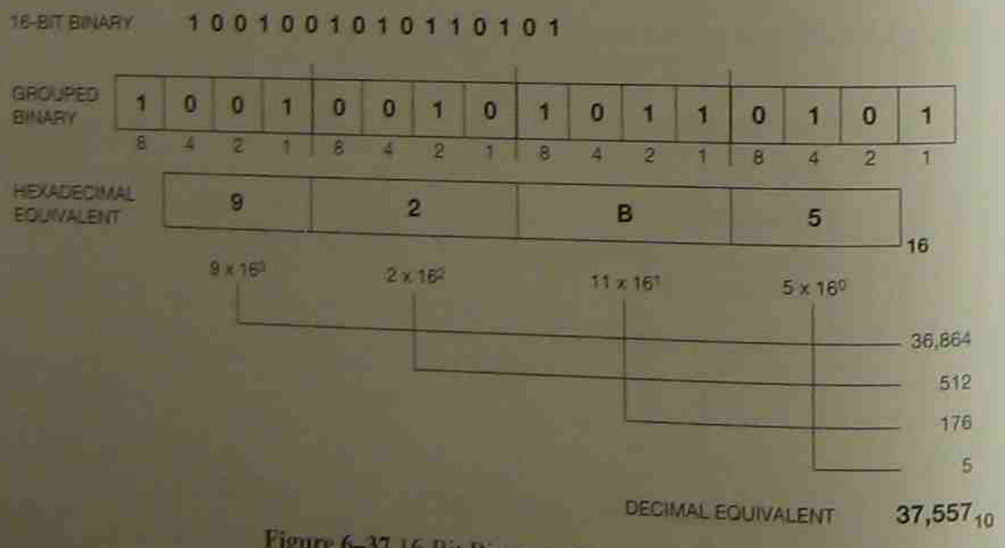


Figure 6–37 16-Bit Binary to Hexadecimal

The first step in converting 16-bit binary to hexadecimal is to group the 16-bit binary word into groups of four (conversion to Binary Coded Decimal [BCD]). Each group of four digits is converted to its hexadecimal equivalent. In Figure 6–37, the hexadecimal number is $9\ 2\ B\ 5_{16}$.

The conversion of $9\ 2\ B\ 5_{16}$ to decimal is $37,557_{10}$. Figure 6–38 shows the conversion of the original 16-bit binary number to its decimal equivalent.

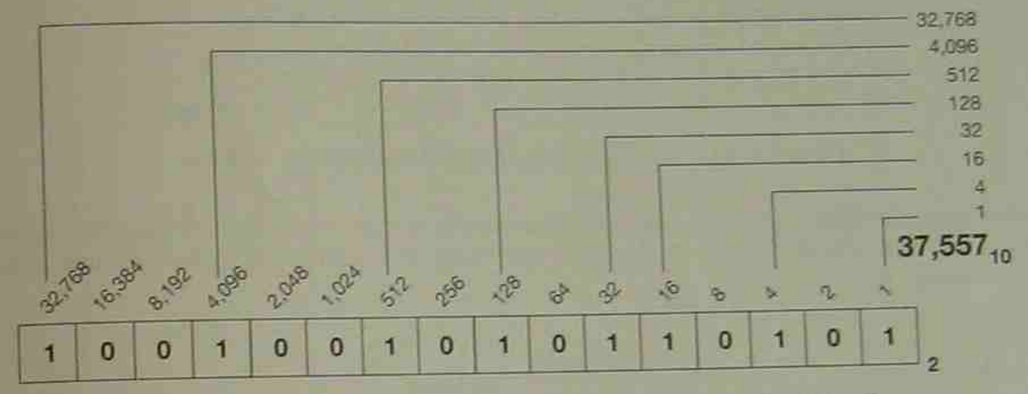


Figure 6–38 Converting a 16-Bit Digital Number to a Decimal Number

BINARY CODED DECIMAL (BCD) SYSTEM

When large decimal numbers are to be converted to binary for memory storage, the process becomes somewhat cumbersome. To solve this problem and speed conversion, the Binary Coded Decimal (BCD) system was devised. In the BCD system, four binary digits (base 2) are used to represent each decimal digit. To distinguish the BCD numbering system from a binary system, the designation BCD is subscripted and placed to the lower right of the units place. Converting a BCD number to a decimal equivalent is shown in Figure 6–39.

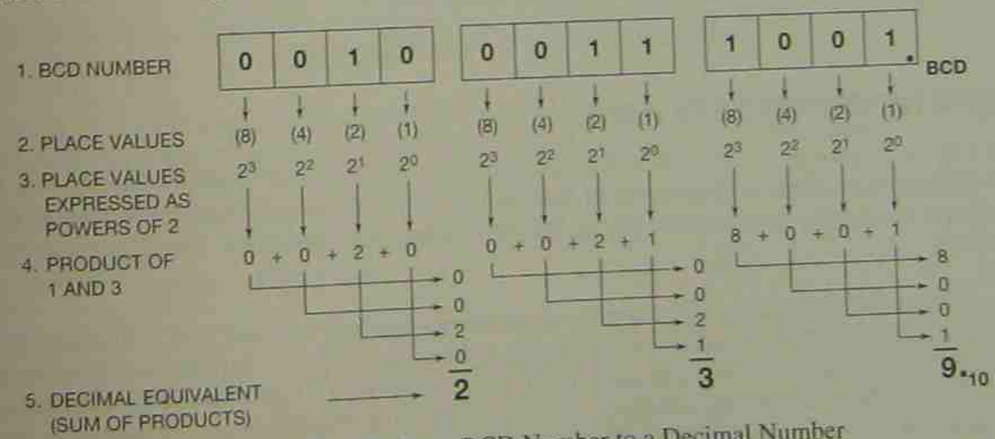


Figure 6–39 Converting a BCD Number to a Decimal Number

When using a BCD numbering system, three decimal numbers may be displayed using 12 bits (3 groups of 4), or 16 bits (4 groups of 4) may be used to represent four decimal numbers or digits.

When only three decimal digits are to be represented, using 12 bits, they are further identified as *most significant digit* (MSD), *middle digit* (MD), and *least significant digit* (LSD) (Figure 6–40).

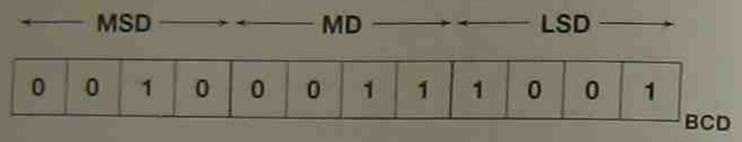


Figure 6–40 Converting a BCD Number to a Decimal Number

When using the BCD system, the largest decimal number that can be displayed by any four binary digits is 9. The table in Figure 6–41 shows the four binary digit equivalents for each decimal number 0 through 9.

PLACE VALUE				DECIMAL EQUIVALENT
2^3 (8)	2^2 (4)	2^1 (2)	2^0 (1)	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Figure 6–41 Binary to Decimal Equivalents

USING NUMBERING SYSTEMS

The keyboards used by various types of programming devices may have electric symbols and special function keys, along with numeric, or number keys, for addressing. Many keyboards also have alphanumeric (letters and numbers) keys for report generation and other special programming functions.

The alphanumeric keys of many programming terminals generate standard ASCII characters and control codes. **ASCII** is an acronym for **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange. The ASCII code uses different combinations of 7 bit binary (base 2) information for communication of data. The data may be communicated to a printer, tape loader, floppy or hard disc, or be displayed on the VDT of the programmer and/or computer.

Note: *ASCII information is often expressed in hexadecimal (base 16.) Figure 6–42 shows the 128 standard ASCII control code and character set with both the binary and hexadecimal numbering systems.*

LSB LEAST SIGNIFICANT BIT ↓ HEX		MSB MOST SIGNIFICANT BIT → BINARY	000	001	010	011	100	101	110	111
			0	1	2	3	4	5	6	7
000	0		NUL	DLE	SP	Ø	@	P	\	p
0001	1		SOH	DC1	!	1	A	Q	a	q
0010	2		STX	DC2	"	2	B	R	b	r
0011	3		ETX	DC3	#	3	C	S	c	s
0100	4		EOT	DC4	$	4	D	T	d	t
0101	5		ENQ	NAK	%	5	E	U	e	u
0110	6		ACK	SYN	&	6	F	V	f	v
0111	7		BEL	ETB	'	7	G	W	g	w
1000	8		BS	CAN	(	8	H	X	h	x
1001	9		HT	EM	)	9	I	Y	i	y
1010	A		LF	SUB	*	:	J	Z	j	z
1011	B		VT	ESC	+	;	K	[	k	{
1100	C		FF	FS	,	<	L	\	l	¦
1101	D		CR	GS	-	=	M	]	m	}
1110	E		SO	RS	.	>	N	^	n	~
1111	F		SI	US	/	?	O	–	o	DEL

Figure 6–42 Standard ASCII Control Code and Character Set

The digital or hexadecimal number is determined by first locating the vertical column where the code or character is located, and then the horizontal line.

EXAMPLE: The letter A is in column 4, horizontal line 1. The binary number that transmits the letter A is 100 0001. The hexadecimal number is 41. The symbol # is 010 0011 in binary and 23 in HEX.

An eighth bit is often used by programmers to provide error checking of information that is transmitted. This eighth bit is called the **parity bit**.

For *even* parity, the parity bit (the eighth bit) is added to the seven bits that represent the ASCII codes and characters so that the number of 1s will always add up to an even number.

EXAMPLE: The binary number for the # symbol is 010 0011. The 1s add up to three, an odd number. By adding an eighth bit and making it a 1, the total of 1s are now 4, or even, as shown in Figure 6–43.

	PARITY BIT	ASCII CODE BITS
EVEN PARITY	1	0 1 0 0 0 1 1

Figure 6–43 Parity Bit Set to 1 for Even Parity

The letter A, which is the binary number 100 0001, has two 1s and is already even. In this case, the parity bit would be a 0, as shown in Figure 6–44.

	PARITY BIT	ASCII CODE BITS
EVEN PARITY	0	1 0 0 0 0 0 1

Figure 6–44 Parity Bit Set to 0 for Even Parity

The ASCII control code BS (backspace) is binary number 000 1000. For even parity, a 1 is added for the parity bit shown in Figure 6–45.

	PARITY BIT	ASCII CODE BITS
EVEN PARITY	1	0 0 0 1 0 0 0

Figure 6–45 Even Parity

By checking each character or control code that is sent for an even number of 1s, transmission errors can be detected when an odd number of 1s is found.

For systems that operate on *odd* parity, the parity bit is used to make the total of 1s add up to an odd number.

EXAMPLE: The number 5 has a binary number of 011 0101. The 1s add up to 4. The parity bit is set to 1, making the 1s total 5, or an odd number. Figure 6–46 illustrates this concept.

	PARITY BIT	ASCII CODE BITS
ODD PARITY	1	0 1 1 0 1 0 1

Figure 6–46 Parity Bit Set to 1 for Odd Parity

For systems that do not use a parity bit for error checking, the eighth bit is always a zero (0).

Chapter Summary

There are several numbering systems that are used to store information in the form of binary digits (bits) into the memory system of a processor. The specific numbering system or the combination of numbering systems used depends on the hardware requirements of the specific PLC manufacturer. The important thing to remember, however, is that no matter which numbering system or systems are used, the information is still stored as 1s and 0s.

For programmable controllers to perform arithmetic functions, a way must be found to represent both positive and negative numbers. One of the most common methods used is called 2s complement. Using 2s complement, negative and positive numbers can be added, subtracted, divided, and multiplied. In reality, however, all arithmetic functions are accomplished by successive addition.

Review Questions

1. When information is stored using only 1s and 0s, it is called a _____ system.
2. A *bit* is an acronym for _____ .
3. The decimal numbering system uses 10 digits, or a base of 10. List the base for each of the following numbering systems.
   a. binary base _____
   b. hexadecimal base _____
   c. octal base _____
4. Convert *binary* number 11011011 to a *decimal* number.
5. Convert *decimal* number 359 to a *binary* number.
6. Convert *hexadecimal* number 14CD to a *decimal* number.
7. Convert *decimal* number 3247 to a *hexadecimal* number.
8. Convert *decimal* number 232 to an *octal* number.
9. How do we prevent binary numbers 10 and 11 from being confused as decimal numbers?
10. Convert the following *binary* values to *decimal*.
    a. 10011000
    b. 01100101
    c. 10011001
    d. 00010101
11. Convert the following *BCD* values to *decimal*.
    a. 1001 1000
    b. 0110 0101
    c. 1001 1001
    d. 0001 0101
12. The BCD value 1001 0011 0101 is *not*
    a. 935 decimal
    b. 0011 1010 0111 binary
    c. 647 octal
    d. 3A7 hexadecimal
13. The hexadecimal value 2CB is *not*
    a. 715 decimal
    b. 1313 octal
    c. 0010 1100 1011 binary
    d. 0111 0001 0011 BCD
14. Express the following signed decimal numbers in 2s complement. Use 8-bit words. Show all work.
    a. (–)7
    b. (–)4
    c. (–)3

15. Convert the following decimal numbers to 2s complement and add. Use 8-bit words. Show all work.
    a. (+)4
       (−)7

    b. (−)10
       (+)22

    c. (+)22
       (+)33

CHAPTER

7 Understanding and Using Ladder Diagrams

Objectives

After completing this chapter, you should have the knowledge to:
- Identify a wiring diagram.
- Identify the parts of a wiring diagram.
- Convert a wiring diagram to a ladder diagram.
- List the rules that govern a ladder diagram.

There are basically two types of electrical diagrams: wiring diagrams and ladder diagrams.

WIRING DIAGRAMS

The wiring diagram shows the circuit wiring and its associated devices (relays, timers, motor starters, switches, and the like) in their relative physical location (Figure 7–1). While this type of diagram assists in locating components and shows how a circuit is actually wired, it does not show the circuit in its simplest form. To simplify understanding of how a circuit works, and to show the electrical relationship of the components (not the physical relationship), a ladder diagram is used.

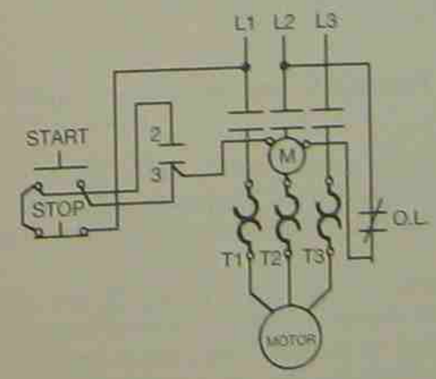


Figure 7–1 Wiring Diagram