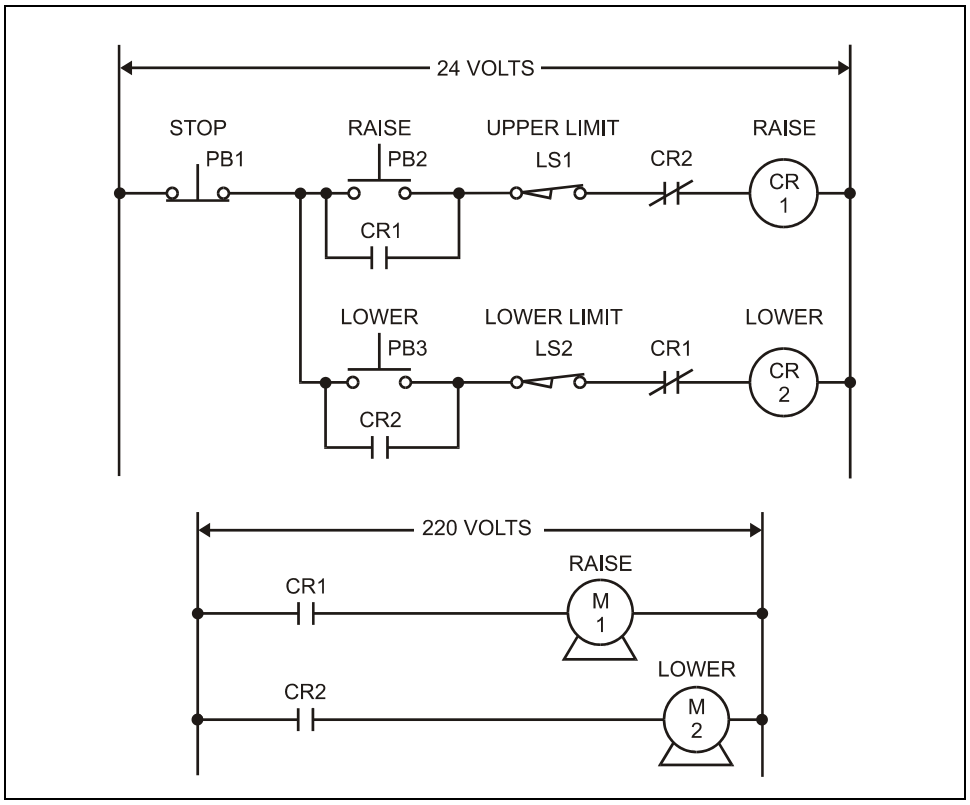

Figure 9-13. Door Raise and Lower Interlock

what to do; the *communications processor*, which handles communications with the outside world; a *battery*, which keeps time and maintains the program; and a *power supply*, which provides the energy necessary to run the processor.

The processor can vary in size, from a small part inside the one-piece PLC and commonly known as a "shoebox," to a single card in an input/output rack in some medium-sized PLCs, all the way up to an entire rack with one or more cards for each part of the processor in large PLCs. The central processing unit (CPU) can be a 4-, 8-, 16-, or 32-bit microprocessor. The larger the number of bits the microprocessor can handle at a time, the more work it can do, and, hence, the faster it will execute a program. With the advent of specialized microprocessors, the communications processor may very well be a microprocessor that is designed only to handle the communications between the real world and the memory used by the CPU. In other designs, several standard or custom chips may handle this function. Even the CPU itself may have some specialized functions (such as mathematical operations and PID control performed by microprocessors designed just for these operations). This use of several microprocessors within the same processor is called multiprocessing; it provides faster operation than would result from doing everything in one microprocessor. Some manufacturers now allow the user to select some of the PLC's functionality by adding communication or other modules to the processor rack. This can be less costly than making different models of a processor with varying numbers of ports and memory. The introduction of much faster back-plane buses and higher-speed communications from PCs has enabled this add-modules-as-needed approach. Previously, these components had to be integrated in the same module to work effectively. Now they can be distributed.

Some small PLCs were called shoebox PLCs because they were about the size and shape of shoe boxes.

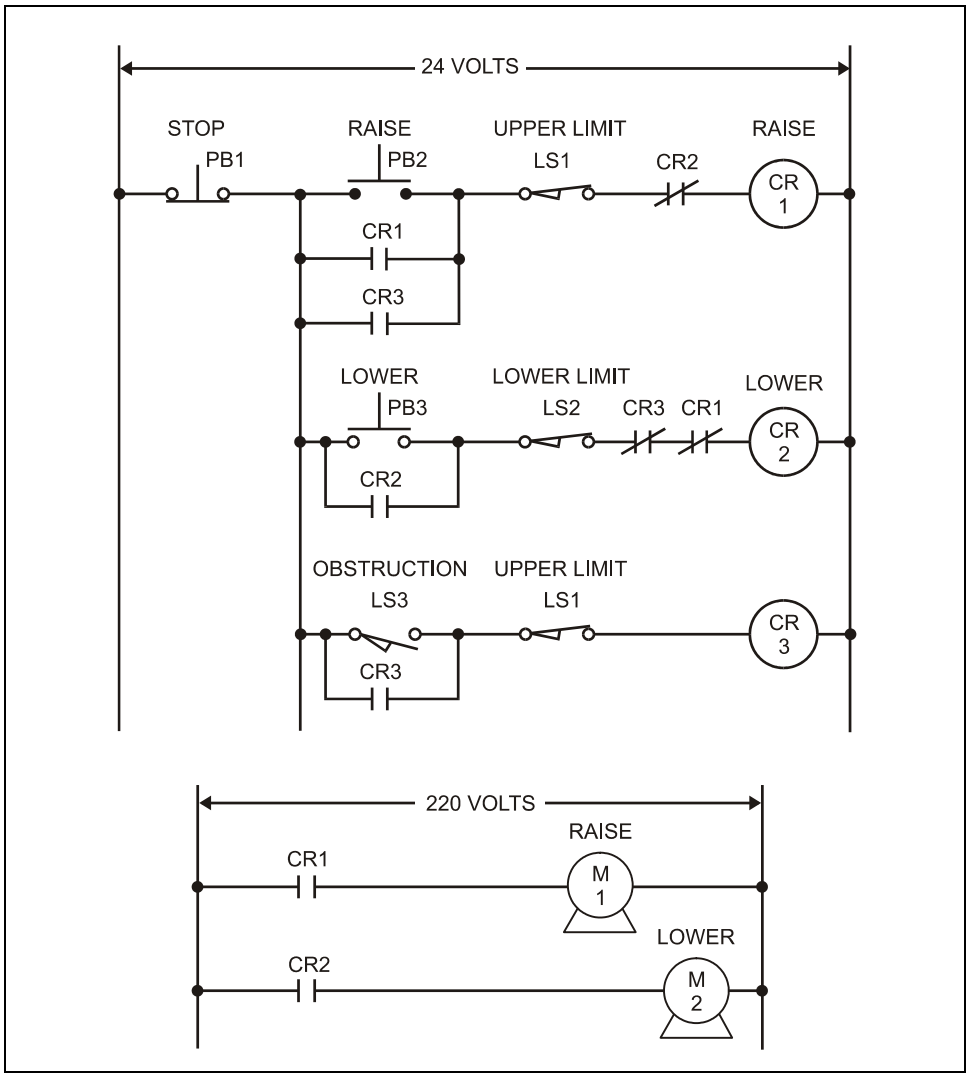

Figure 9-14. Door Opener Auto-Reverse Circuit

See Chapter 5 for a more complete discussion of the operation of computers and microprocessors.

Processor Scan

Unlike the office or general-purpose computer, the classic PLC executes only one program, and it keeps on doing it over and over again. As we mentioned at the beginning of the chapter, the time the PLC takes to execute this program once is called the scan time. Not only is the scan time itself of interest, but so is its variance under differing operating conditions. Figure 9-16 is a block diagram of the steps involved in a complete cycle. Many PLCs now support multi-tasking, in which a number of tasks or programs can be run on a scheduled basis, thus allowing the user to configure the PLC to suit his or her application.

Note the distinction between "program scan time" and "complete scan time." The difference can be as much as 100 percent.

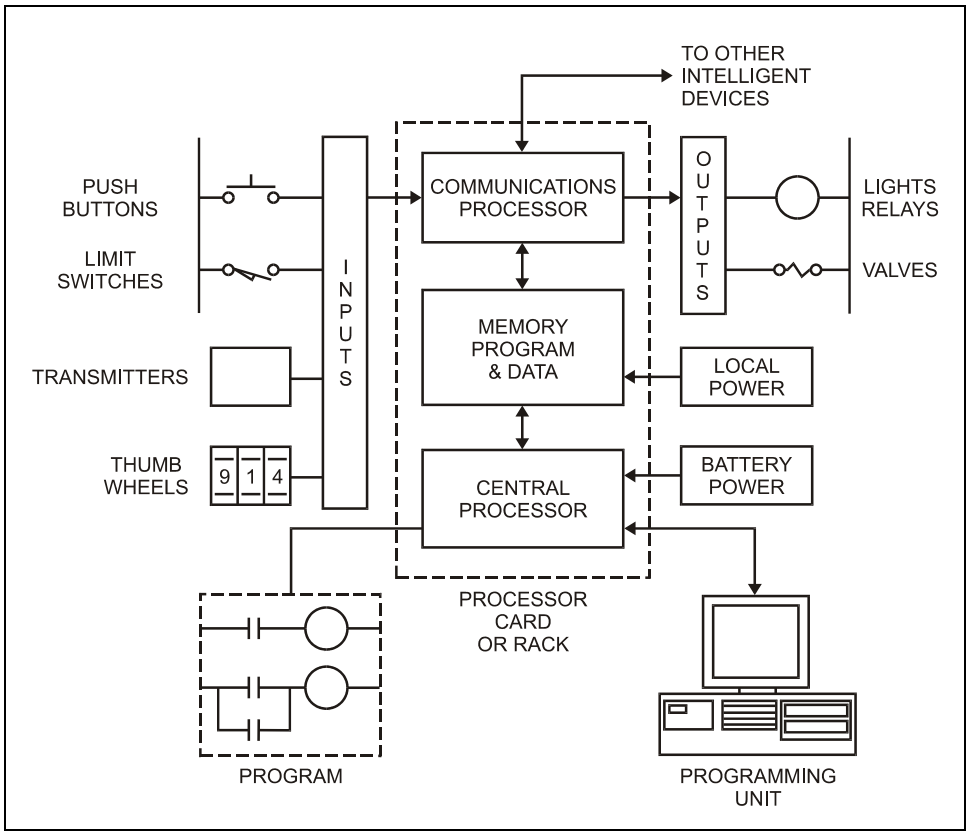

Figure 9-15. Generalized PLC Block Diagram

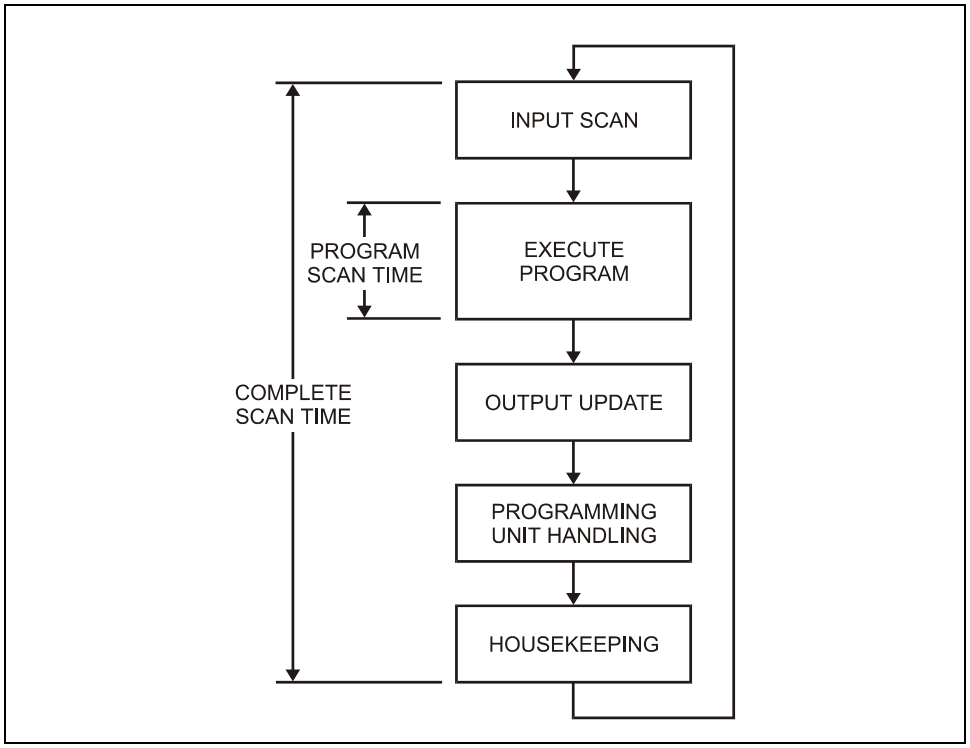

Figure 9-16. Steps in PLC Scan Cycle

Examine the PLC's execution time for instructions that accomplish the same function and then choose the faster one. It may be faster to multiply by 0.5 than to divide by 2. Similarly, a multipurpose function may be slower than a dedicated function. The only way to find out is to study the manufacturer's published information.

During the input scan, the processor updates the information on all the inputs. This may take a fixed amount of time, say, one millisecond, or it may be expressed as *x* milliseconds per *k* of inputs. The output update time is usually expressed the same way, and sometimes the two are combined into one figure.

Although a fast scan time is often desired, a repeatable scan time can be even more important, especially in a process where events must be reliably timed. The experienced designer can choose the logic functions that will result in the shortest execution time. Many PLCs will reduce their program scan time by stopping the solution of a line of logic once it has determined it cannot be true, regardless of the rest of the line. Figure 9-17 shows how this reduces scan time. The price one pays for this extra speed is often a scan time that is more variable. By the judicious ordering of contacts, the designer can tune a program to suit the specific need. With many of the newer PLCs, scan time is not a consideration since they are fast enough for most applications. Only when the application is very large or timing is critical will you have to resort to these techniques.

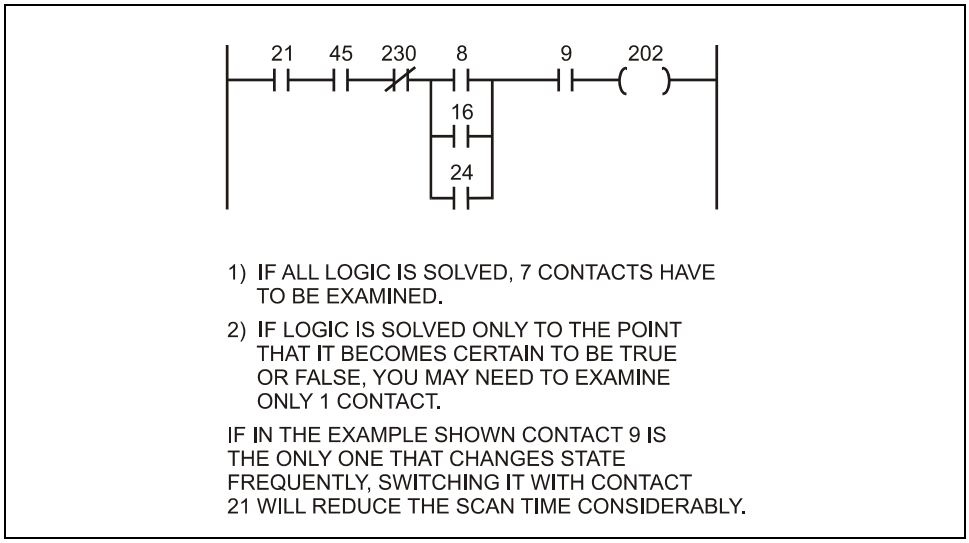

Figure 9-17. Reducing Scan Time in a PLC

In small PLCs, program execution is often expressed as *y* milliseconds per *k* of program. Medium and large PLCs give the time it takes for each instruction, often with an infinite number of details for handling all the variations of each command. The user is left to figure it all out. Many PLCs have an internal function that records the scan time and even gives the maximum and most recent values.

Once the PLC has read its inputs, executed its program, and set its outputs, one would expect it to be able to start over again right away. Unfortunately, a few other things must be taken care of first. The programming unit, if it is connected, needs some time to communicate with the processor, especially if the program logic is being changed. In some older PLCs, there was a noticeable effect on scan time when the programming unit was connected. Even more recent PLCs can have problems if the same communication network is used for both programming and operation. The use of intelligent display devices for HMI (human-machine interface) often takes up a significant amount of communication time, and we must consider it here. The alternative to taking up a lot of communication time and slowing down the scan time is to slow down the communications. However, this may result in unacceptable delays between when an operator initiates an action and when the action takes place. Once again, only the manufacturer can say for sure what the limitations are. The designer has to take these limitations into account in his or her design and choice of equipment.

The PLC also does some housekeeping things such as checking battery status, verifying memory integrity, checking power-supply voltages, and resetting the watchdog timer. Generally, these functions do not take much time, and they are often buried in the input/output scan or update.

The preceding description covers the single-program operation of a PLC. It is now time to consider the multi-tasking approach. For multi-tasking, each task is assigned a priority and the time between successive executions. The processor will start by executing the task with the highest priority. When it is done, it goes on to the next highest until all tasks are done. This works nicely if you can get everything done once before it is time to do the highest-priority task again. Usually, we are not so lucky. Before the last task is done, it is time to do the highest priority once again. In this case, the controller suspends execution of the lower-priority task, then executes the higher-priority task, and finally goes back to the lower-priority one again. This, of course, results in the lower-priority tasks taking more time to execute, but then they are not as important and can tolerate the longer processing time. The designer has to determine the priorities and maximum delays between execution based on his or her knowledge of the process or application. Obviously (or maybe not that obviously), for multi-tasking to work you must have a mix of priorities and execution delays so you have enough time to get everything done in the allotted time. If you have only high-priority tasks with fast execution times, then multi-tasking may not work. In that case, you will have to go to a faster processor, or use multiple processors, a dedicated card, or modules for some parts of the application. Also, it takes some time to switch between tasks, so frequent switching can make the overall program take even longer.

Input/Output Scan

The processor can use two methods to communicate with its input and output equipment. In the parallel mode, all the bits of a word are transferred at one time or in parallel. This method is fast, but it is limited to short distances because of the timing changes caused by cable capacitance. These changes cause the bits to get out of step with each other. In the serial mode, the bits of a word are transmitted one at a time and one after the other. This method is slower, but it can be made to work very well over almost any distance. Generally, the processor can update all its parallel inputs in one scan, but it requires several scans to update all its serial inputs. One can say that parallel I/Os are synchronous with the PLC scan but that serial I/Os are asynchronous. PLC manufacturers vary in the way they handle asynchronous I/Os. Some will access one rack in each scan, while others will provide data only on how long it takes to update the I/O table. Since the mid-1990s the data rate for serial I/O has increased substantially, and we now have serial I/O that is just as fast as some of the older parallel I/O. We have the Internet to thank for this development.

The asynchronous nature of serial I/Os has serious implications for scan time. The scan time is no longer a good indication of how fast the PLC is or how fast an input change can be seen by the PLC. For example, a PLC with a 40-millisecond scan time can easily handle events that last for 100 milliseconds if it is using parallel I/Os. If, on the other hand, it is using serial I/Os and the update time is 100 milliseconds, the PLC could still have a 40-millisecond scan time and not be sure of seeing a 100-millisecond event. In this case, the minimum event duration needed to ensure that the PLC saw it is 200 milliseconds, or five times the processor scan time.

To get around some of these I/O update limitations, most PLCs have a feature that will do an immediate update of some I/Os during the scan instead of waiting for the end of the scan. This works very well with parallel I/Os and fast I/O but is useless with older, slower serial I/Os. The immediate update also slows down the scan time since the processor has to stop executing the user's program while it updates the I/Os. Again, if you have a newer model PLC with faster scan time, this may not be a handicap. In all cases, the I/O's immediate update works only on a limited number of bits, usually one word. This is still a very useful tool, however,

when only a small amount of high-speed logic is required. It is often a real life-saver in high-speed discrete manufacturing applications where only a few parts of the application require high speed.

Power Supply

Needless to say, PLCs need power to operate. Two sources of power are used in PLCs: battery power and local power from the plant or utility. In many cases, the battery is used to maintain the user's program, keep the time-of-day clock running, and retain the status of certain bits and registers. Some small PLCs do not use a battery and so cannot maintain a real-time clock. Whenever a battery is part of the PLC, it should be monitored in the user's program, with adequate warning provided when it is getting low. Memory and program backups are covered in the next section, and power supply wiring and installation are discussed in the section "Guidelines for Installation, Start-up, and Maintenance." Early PLC batteries had a life of only about a year, but many now have batteries that will last up to five years.

Power quality will affect the operation of a PLC. There are two places to provide this quality: in the power supply itself or in extra equipment added to the "upstream" side of the power supply. There is no free lunch in this respect because skimping on the power supply too often requires that equipment be added that can wipe out all the savings. A well-designed PLC will include circuitry in the power supply or elsewhere that continually monitors all necessary voltages and performs an orderly shutdown when the voltage drops below the level required to maintain reliable operation. Similarly, there will be circuitry to allow the processor to run only after it determines that all the voltages needed for operation are present and stable. It is important to have a large enough dead band between shutdown and startup to avoid oscillations or other problems if you are near one of the change points.

Look at both the range above and below nominal when considering power-supply capabilities. The general practice for utilities is ±10 percent of nominal, but manufacturers of PLCs may give narrower limits such as ±5 percent or +5 percent -10 percent. Although low voltages have been the most common problem, high voltages can cause as many, if not more, problems. Since high line voltage is not as common as low line voltage, designers have not always designed for continuous operation at the maximum voltage. This has led to overheating of components and premature failure. It has also led to erratic operation in devices that employ self-resetting over temperature, over load over voltage protection. A high line voltage may trigger one of these protection circuits, which shuts the device down, and once it has cooled or the condition has been removed, it will start up again all by itself. If somebody is not around to notice what has happened you can look for a long time before finding the problem.

Programming Devices

Unlike computers, most PLCs do not come with the software and hardware necessary to write, document, and debug the programs that run in them. With a personal computer, one needs to buy only a compiler or an interpreter (software) to write programs; the hardware is already there. PLCs necessitate the purchase of software, or sometimes hardware, to create, debug, and document programs. The most basic programming device is a handheld single-instruction device that allows one to program one instruction at a time and see the result on an LCD screen. This device is inexpensive and easy to use as long as the program is short. In the case of the small-relay replacers it is built into the PLC. It is also very limited and is generally used only on the small shoebox PLCs. A step up is the handheld line-of-logic device. This device has a larger LCD display that shows an entire line of logic in relay ladder symbols, instead of only one instruction. It is clearly much easier to use and more flexible, but it is also more expensive. The line-of-logic device is available for the small shoebox PLCs and some medium-sized controllers. Some handheld units have the capability to save a program to a memory card. Older ones would save to a separate tape unit and then reload the program from the same tape. However, these units are now obsolete.

The dedicated programming terminal is designed solely for programming the PLCs of a single manufacturer and, often, only one family or type of that manufacturer's product line. As with the handheld units, ladder logic is displayed on a screen, often only one line at a time, but other features such as documentation, searching, and copying are offered. The program can be saved, usually on tape either in an integral unit or in a separate device. These programming devices are rugged industrial units, but they are also heavy, expensive, and not useful for any-

thing else. Because of such limitations, this type of programming device has disappeared. However, one still occasionally hears a request for one on list servers.

The most powerful programming device is the personal computer equipped with a suitable adapter card and software. This equipment is available from the PLC manufacturer or a third-party vendor; both sources have pros and cons. The software usually provides all the features found in the dedicated programming device, plus such features as saving and loading from diskette, formatted printing, more extensive documentation, and cross referencing. In addition, the computer can be used for other tasks. To make the desktop computer more portable, some people place it on a dolly or computer cart and even provide a uninterruptible power source (UPS) so it can be moved from place to place without being powered down.

Another programming tool is the palmtop computer with PLC programming software built in. The limited memory of these units does not allow you to store many programs, but they are very much more portable than a laptop or desktop mounted on a dolly. One consideration against laptops and palmtops is the risk that they will be stolen if left unattended on the plant floor or in the field.

A few points must be considered when choosing a computer-based programming system. Many PLC manufacturers offer an industrialized computer with the necessary hardware and software to program their PLCs. In creating this package, the PLC manufacturer may use a nonstandard basic input-output (I/O) system (BIOS), special cards or boards, and even a different operating system. This can create problems for users who want to run word processing, database, or other programs. Installing software from another PLC manufacturer in the same computer can cause even more problems. Fortunately, most PLC manufacturers and all third-party software manufacturers offer software that will run on any standard computer that has the specified memory, drives, and monitor. Almost all options require the user to buy the adapter card that allows the computer to communicate with the PLC, or you can use an Ethernet connection. Many of the original adapter cards used the ISA bus in the PC, but since the ISA bus has been replaced by the PCI bus, you have to get new adapter cards when changing your PC. Another point to consider is the operating system. Some of the newer programming PLCs require Windows NT or 2000 and will not run on anything else. If you are starting from scratch it is not a problem. But if you are adding a new PLC to those you already have, you may have problems getting the older software to co-exist on the PC. Also, for those of you who use languages other than English on your PCs there can be problems if the language of the PC is not the same as that of your software (this is not unique to PLC software; other types of software suffer from the same problems).

The selected programming device must permit one to:

(1) copy the program to some secure media (one option is copying the screen display of a handheld device by hand onto a sheet of paper),

(2) reload a program from the storage media,

(3) monitor the program while it is running,

(4) make changes to the program, and

(5) develop new programs.

The ease, flexibility, and cost of the programming device can vary considerably, and they are often major considerations when choosing a PLC. In the case of PLCs that are supplied with a machine it may be the machine builder, rather than the end user, who makes the choice for their own reasons—which may not be as convenient for the end user. If you are never going to change the machine and will rely on the manufacturer for service it does not really matter how the program is developed.

The Memory System

As with any computer, a PLC needs memory to store its operating system, data, and program. Unlike personal computers, the PLC must not lose its program when power fails. It must also restart itself automatically after a power failure and, if desired, continue from where it left off. These requirements can be met in many different ways, depending on the size of PLC, its speed, and cost targets. Bear these differences in mind when comparing different PLCs for the same job. Sometimes, the PLC manufacturers will bring them to a buyer's attention, especially if they feel their particular design has an advantage over a competitor. One should not rely on this, however.

Structure

The two basic parts of PLC memory are the system memory and the user memory. The system memory contains the programs that tell the microprocessor how to operate as a PLC and some temporary memory for the microprocessor to use in executing its programs. The system memory is programmed by the PLC builder and is clearly not under the user's control. The system memory has three parts: the executive programs, the scratch pad, and the system status areas. The scratch pad and system status areas are constantly being changed while the PLC is running. For older PLCs, the executive programs do not change and are thus located in programmable read-only memory (PROM). Some manufacturers may even put the executive in a custom-made read-only memory (ROM), but this requires extremely high volumes to be economically viable.

Under some circumstances, the executive memory may be changed. If the PLC manufacturer adds features, enhancements, or bug corrections in the executive program, these changes may be made available to users who have the old version. Sometimes these upgrades are free, usually if the upgrade involves corrections of bugs or features that were promised but not available at the original delivery time. On other occasions, there is a charge for such upgrades. Some manufacturers simply send the chips along with instructions and new labels for the PLC, and the user changes the chips himself. Others require that the old PLC be sent back for an upgrade. Some users prefer to change the chips themselves because this involves work that is easier to schedule and requires only minimal downtime. However, many users do not have the tools or the desire to get into chip changing, especially where static-sensitive complementary metal-oxide silicon (CMOS) chips are concerned. In many cases, the board or PLC must be returned to the manufacturer for upgrade because the changes involved more than merely changing a plug-in chip, and the manufacturer judged it best to make these changes for the user.

More recent-model PLCs have executive memory (or an operating system) that can be changed or flashed in the field to update it to a more recent version. This makes updating easier but means you have to keep track of what version of operating system is in a PLC and that you usually also have to update the programming software and sometimes convert the program. The difficulty may be that at the same time you will also have to update various other devices (drives, displays, communications adapters) to the correct versions to keep your machine working. I have seen an update take all day on a production line, and I have also seen it take only an hour. But it is still easier and more flexible than the old method of changing chips.

This brings us to the really interesting part—the user memory. User memory contains the following items: input data table, output data table, internal bits, internal registers, the user program, and for some PLCs, the tag names. Each PLC manufacturer will have proprietary terms for these items. They differ slightly from the terms we have used here, but they should still be easily recognized.

There are three ways to allocate the memory among these five items: fixed allocation, user-configurable hardware, and user-configurable software.

In fixed allocation, the PLC manufacturer determines all the allocations for a particular model beforehand and leaves the user to choose only what model suits his or her needs. For user-configurable hardware, the user can choose from a limited list of hardware options to suit the application. In the user-configurable software version, the user makes all the choices, subject to a maximum memory limitation. Table 9-2 summarizes these choices.

Table 9-2. Memory Types and Configurations

ITEM	USE	MEMORY TYPE	HARDWARE CONFIGURABLE	MEMORY CONFIGURABLE	SIZE 1 WORD = 8, 16, or 32 BITS	COMMENTS
EXECUTIVE	Operating system for the PLC	Read-only	System	System		
SCRATCH PAD	Temporary data for the system	Read/write	System	System		
SYSTEM STATUS	Clock error conditions, scan time, etc.	Read/write	I/O system	System	64 words	
INPUT DATA TABLE	Status of inputs, force table	Read/write	I/O system	User memory	32 to 256 words	At least 2 bits per input
OUTPUT DATA TABLE	Status of outputs, force table	Read/write	I/O system	User memory	32 to 256 words	At least 2 bits per output
INTERNAL BITS	Single bits used for program control	Read/write	Data memory	User memory	16 to 128 words	
REGISTERS	Data and values used in the user program	Read/write	Data memory	User memory	512 to 4096 words	At least 2 registers per timer/counter
USER PROGRAM	Controls the machine	Read/write	Program memory	User memory	2048 to 32768 words	
TAG Names	Provide human-understandable names for everything	Read/write	Program memory	User memory	10 to 500K words	Depends on user

The fixed allocation is usually found in the small shoebox PLCs. Here, there may be ten inputs, twenty outputs, sixty-four timers/counters, two hundred internal bits, and so forth, and the only way to change this is to buy a different model. The fixed allocation makes it easier for the system programmer, who has complete control of everything. It may also make the PLC faster since the hardware and software can be optimized for one configuration.

With user-configurable hardware, the user gets to choose a configuration to suit the specific application, for example, from among 256, 512, 1024, 2048, or 4096 I/Os and from among 1K, 2K, or 4K registers. Some manufacturers may allow any values for I/Os and registers to be selected, while others limit the choice to a smaller range (such as 1K I/Os and 1K registers, 2K I/Os and 2K registers, or 4K I/Os and 4K registers). Since the hardware is usually on a card, the penalty for not getting enough is buying a new I/O memory card (sometimes a credit is available for the old one). This is because the cards are different, and one cannot simply plug in a few more chips and upgrade from 1K to 2K. The user program memory is on another card and is usually field expandable, either by adding chips or by adding another card, or both. Care must be taken when reading the PLC manufacturer's literature and manuals so one understands the various upgrade possibilities and their costs.

User-configurable memory occupies a single block of memory so as to provide all the functions required by the system. Some partitions, such as I/O tables, may be of fixed length based on the maximum I/Os the PLC can handle. Others, such as internal bits and registers, are user definable to suit the needs of the program. This overhead reduces the memory available for the program itself. The original 8K of memory could become 6K or even 4K after all the necessary bits and registers have been given the space they need. Configurable memory provides greater flexibility, but the user has to manage it and understand the limitations.

If many copies of the same application are to be made, a better approach is to do a pilot project with whatever it takes. Then, once the application is well developed, see what can be done to optimize the memory requirements.

It's easy to see the user's dilemma—how much memory is needed for I/Os? Internal bits and registers? Program space? How does this change if a different PLC is used? All these questions must be answered before the program is written. One can estimate the needs reasonably well if one has a well-defined application with a complete I/O list, a completed description of what the PLC should be doing, and a good knowledge of the PLCs concerned. If any one of these three things is missing, and one usually is, the decision is not so easy. Most designers will choose to buy more memory, believing that the cost of trying to program around a memory limitation is going to be far more than the cost of the extra memory. With the relatively low cost of memory, this is justified most of the time. The only difficulty is when a manufacturer makes memory size a function of the I/O available and other options, so that to increase memory you have to buy a larger and more expensive processor. This is fine if you need the other options, not so fine if you only need the memory.

Memory Types

The first consideration when choosing memory types is which method to use to access the data—random or sequential. In random access memory (RAM for short), any bit or word can be accessed directly without having to read the bits or words that come before the desired bit. Sequential memory requires that the memory locations be accessed one after another in exact physical sequence, like reading a tape. All PLC memory is RAM of one type or another. Sequential memory is found only in some types of charge-coupled devices (CCDs for short), which are used in vision systems to read the camera image.

The next consideration has to do with the operations that are permitted on the memory—reading and writing. Memory can be read-only, read and write, or write-only. The only standard acronym is ROM for “read-only memory.” Some examples of write-only memory are the printer and the display screen.

The last consideration is how long the memory will retain the data stored in it. Volatile memory loses the data as soon as power is removed. Nonvolatile memory will retain the data intact when power is removed.

Of the types of nonvolatile read-only memory, the most stable is the ROM chip. This chip is made like a memory chip except that the data values are designed right into the chip and cannot be changed. This process is used only for high-volume applications such as the executive program. Changing the program requires designing a new chip.

Programmable read-only memory (PROM) can be programmed once and only once. The chip is made with fusible elements for each bit, and thus comes with all bits set to 1. To program the chip, a high current that melts the fusible link is passed through those bits that are to become zero. PROMs are often used for smaller-volume applications that do not justify the time and expense of ROMs.

A further development is the erasable programmable read-only memory (EPROM). This chip has a window over the storage area, which allows the chip to be erased by exposing it to an ultraviolet (UV) light source. Complete erasing usually takes several minutes, but several chips can be done at the same time. The programming process does not physically destroy parts of the circuit. Rather, it

places a charge at the desired location in the memory in order to indicate a 1. Since the memory cells are made with no discharge path, the charge remains until it is erased. There is a limit to the number of times an EPROM can be reprogrammed, but this limitation is not usually considered a problem. The chip can be inadvertently erased if the window over the memory area is not covered and a source of UV light is present, as from a welding torch or a fluorescent light. To avoid this problem, the window is usually covered with a suitable sticker after programming. There have also been questions about the long-term memory retention of EPROMs, considering that such things as cosmic rays could cause a single bit to change. These considerations are probably irrelevant for most PLC users.

At the other extreme is the volatile read/write memory, in which a solid-state memory chip uses a collection of transistors to form a bistable memory cell that can be read without destroying the data in the cell. For details on this, consult an electronics textbook on the subject. Various technologies are used, each with its own costs and benefits, and all this is constantly changing as existing technologies are improved and new ones are developed.

The chip has four sets of connections—power, control, address, and data. The address lines are used to specify the exact memory location to be read or written, and the control lines specify the type of operation to be performed. The data lines contain the data that is to be written or the values found in the memory cell. The proper manipulation of these lines permits data to be read and written. The major drawback is that power is required to maintain the data. Remove the power and the memory is lost. When the power is restored, the individual memory cells may take any value; so one of the processor's startup tasks is often to set all memory to 1 or 0 before going on.

Consider now the nonvolatile RAM. One of the older types is EEPROM, sometimes called E2PROM, for “electrically erasable programmable read-only memory.” These chips are similar to the UV erasable ones, except that the erasing can be done electrically. This makes them ideal for storing programs and other data that must not be lost. Although these chips can be programmed in the PLC, the programming does take a long time compared to the PLC scan time. In addition, the limits on the number of times the chip can be erased are more stringent. These limitations are more than offset by the advantages of having a nonvolatile memory. Another option is battery power for the memory. In this case, a regular volatile RAM chip is used for the memory, and power is supplied to it from a battery in the PLC. Since the battery is powering only the memory, it can maintain the data for at least a year, and some of the newer lithium batteries are said to be good for five years.

Note that although battery-backed memory is nonvolatile, it is not as long lived as the other types of nonvolatile memory. You should realize that removing the battery with the power off clears the memory.

Some manufacturers put the battery right on the memory card so the memory card can be removed without losing the program. Others put the battery in the power supply card; this means that removing either the memory card or the power supply card will clear the memory. This has caught some users! So-called NOVRAM chips combine a regular RAM, EEPROM, and control circuitry, all in one chip. The PLC sees only the RAM, and it activates a control line in the NOVRAM to copy the RAM to EEPROM, or vice versa. In some applications, this copying is automatic. In others, it is up to the user to issue the save instruction. Some systems have been made that have enough energy storage (provided by the capacitor) to allow the NOVRAM to make a copy of the RAM to EEPROM when it senses that the power is failing.