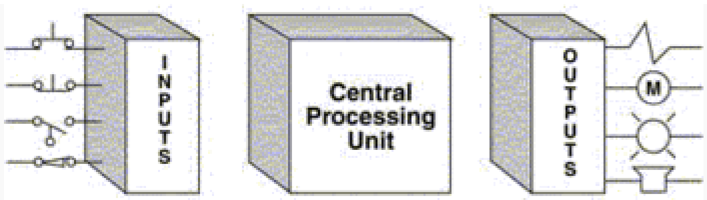
PLC

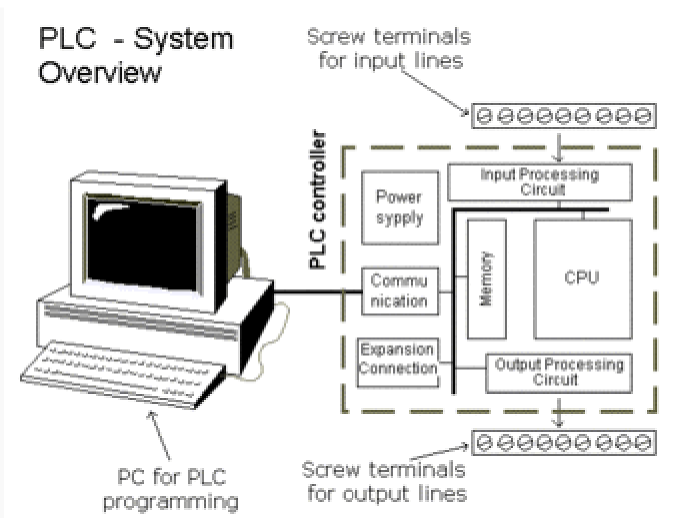
Introduction



Main parts of a typical PLC

A **P**rogrammable **L**ogic **C**ontroller (PLC) is an industrial computer control system that continuously monitors the state of input devices and makes decisions based upon a custom program, to control the state of devices connected as outputs.

Almost any production line, machine function or process can be automated using a PLC. The speed and accuracy of the operation can be greatly enhanced using this type of control system. But the biggest benefit in using a PLC is the ability to change and replicate the operation or process while collecting and communicating vital information.



PLC system overview

Example of the PLC input/output:

- Sensing Devices / Valves
- Switches and Pushbuttons / Solenoids
- Proximity Sensors / Motor

- Limit Switches / Actuators
- Pressure Switches / Pumps

In short, PLC operations consist of four steps:

- 1. Input Scan: Scans the state of the Inputs
- 2. Program Scan: Executes the program logic
- 3. Output Scan: Energize/de-energize the outputs
- 4. Operational commands to the controlled devices

Basic PLC sizing

Some of the typical PLC

The following steps will describe the basic sizing criteria to choose the most suitable PLC device:

- 1) **Determine whether your system is new or existing**: Will your system be installed from scratch or are there existing products already installed? The rest of your system will need to be compatible with new components.

Why this is important: Certain controller products may not be compatible with others. Making sure your existing products are compatible with any new products you are researching will save you time and money.

- 2) **Define any environmental issues that will effect your application**: Consider any environmental issues that will affect your application (temperature, dust, vibration, codes specific to your facility, etc.).

Why this is important: Certain environments may affect the operation of a controller. For example, typical controllers have an operating temperature of 0-55 °C (32-130 °F). If your application will include any extreme environmental conditions, or you have specific codes at your facility that must be met, you will need to either research products that meet those specifications or design the installation to meet requirements.

- 3) **Determine how many discrete and analog devices your system will have**: How many discrete and analog devices will you have? Which types (AC, DC, etc.) are needed?

Why this is important: The number and type of devices your system will include is directly linked to the amount of I/O that will be necessary for your system. You will need to choose a controller that supports your I/O count requirements and has modules that support your signal types.

- 4) **Determine whether your system will require any specialty features**: Will your application require high-speed counting or positioning? What about a real-time clock or other specialty feature?

Why this is important: Specialty functions are not necessarily available in a controller CPU or in standard I/O modules. Understanding the special functions your system may perform will help you determine whether or not you will need to purchase additional specialty modules.

- 5) **Determine the type of CPU you will need**: How much memory will your system require? How many devices will your system have (determines data memory)? How large is your program, and what types of instructions will your program include (determines program memory)? How fast a scan time do you need?

Why this is important: Data memory refers to the amount of memory needed for dynamic data manipulation and storage in the system. For example, counter and timer instructions typically use data memory to store setpoints, current values, and other internal flags. If the application requires historical data retention, such as measured device values over a long period of time, the size of the data tables required may determine the CPU model you choose. Program memory is the amount of memory needed to store the sequence of program instructions that have been selected to perform the application. Each type of instruction requires a specific amount of program memory, typically defined in a programming manual. Applications that are basically sequential in nature can rely on the I/O device rule of thumb to estimate program memory (five words of memory for each I/O device); complex applications will be more difficult to judge. If scan time is important in your application, consider the CPU processor speed as well as instruction execution speed. Some CPUs are faster at boolean logic but slower with data handling instructions. If special functions such as PID are required, the CPU you select may make those functions easier to perform. For program memory required, follow this rule of thumb: 5 words of program memory for each discrete device and 25 words for each analog device.

- 6) **Determine where your I/O will be located**: Will your system require only local I/O, or both local and remote I/O locations?

Why this is important: If subsystems will be needed at long distances from the CPU, you will need a controller that supports remote I/O. You will also have to determine if the remote distances and speeds supported will be adequate for your application. Serial and Ethernet-based I/O hardware are two typical choices available for most systems. This I/O may also be referred to as distributed I/O, and may require a particular protocol, such as Modbus.

- 7) **Determine your communication requirements**: Will your system be communicating to other networks, systems, or field devices?

Why this is important: Communication ports (other than the programming port) are not always included with a controller. Knowing your system communication requirements will help you choose a CPU that supports your communication requirements, or additional communication modules if necessary.

- 8) **Determine your programming requirements**: Does your application require only traditional programming instructions, or are special instructions necessary?

Why this is important: Certain controllers may not support every type of instruction. You will need to choose a model that supports all instructions that you may need for a specific application. For example, built-in PID functions are much easier to use than writing your own code to perform closed-loop process control. Typical instructions such as timers, counters, etc. are available in most controllers; note any other special instructions required here.

What now?

After the main requirements are determined, it will be much simpler to find a product with the necessary number of I/O points, features, memory, etc. that your application requires.

Source:

http://www.openelectrical.org/wiki/index.php?title=PLC