MEL – 334: Low Cost Automation

PROGRAMMABLE LOGIC CONTROLLER
Automation control systems. These control systems are used in manufacturing plants of all types, and some other applications that you may not have considered.

The control systems are built around special devices, designed to operate industrial machines, and processes. We call these devices Programmable Logic Controllers (PLC) and Programmable Automation Controllers (PAC).
PLC and PAC Systems

PLCs were introduced in the early 1970s. The term “PAC”, was developed to differentiate those older systems from today’s much more powerful, and flexible devices.

An analogy can be made between the VHS video tape, and a DVD. Both systems allow viewers to record TV programs for viewing at a latter time, but the DVD also can also be used to record music, data, and more.

PLCs were designed to control machinery. PACs can be used for machine control, process, motion control, and other applications.
We will use the term PLC generically to refer to both PLCs, and PACs.

This section will explore the various components that comprise a PLC system.
Basic Components of a PLC System

There are **five basic components** in a PLC system:

- The PLC processor, or controller
- I/O (Input /Output) modules
- Chassis or backplane
- Power supply
- Programming software that runs in a PC

In addition to these 5, most PLCs also have:

- A network interface

Let’s look at each in more detail..
Basic Components of a PLC System

Processor, Controller, or CPU

- Stores the control program and data in its memory
- Reads the status of connected input devices
- Executes the control program
- Commands connected outputs to change state based on program execution
  - For example: Turn a light on, start a fan, adjust a speed, or temperature
- Comes in various physical forms
Basic Components of a PLC System

I/O Modules

• Physically connect to field devices
• Input modules convert electrical signals coming in from input field devices such as pushbuttons, to electrical signals that the PLC can understand.
• Output modules take information coming from the PLC and convert it to electrical signals the output field devices can understand, such as a motor starter, or a hydraulic solenoid valve.
• I/O comes in various forms
Input Modules

- Input modules interface directly to devices such as switches and temperature sensors.
- Input modules convert many different types of electrical signals such as 120VAC, 24VDC, or 4-20mA, to signals which the controller can understand.

24 Volt DC System

Terminal block - this is where wires from the field devices are connected

Field device - this is showing a simple switch

+ 24 Volts

- 24 Volts
Input Modules

- Input modules convert real world voltage and currents to signals the PLC can understand. Since there are different types of input devices, there is a wide variety of input modules available, including both digital and analog modules.

- Pressure: 4-20 mA
- Temperature: 0-10V DC
- Switch: 24V DC
- Pressure: 4-20 mA
- Temperature: 0-10V DC
Discrete vs. Analog Modules

- **Discrete** modules use only a single bit to represent the state of the device. For example, a switch is either open or closed. Therefore, the bit is either a 0 (switch is open) or a 1 (switch is closed). *Discrete modules are also known as Digital modules.*

- **Analog** modules use words to represent the state of a device. An analog signal represents a value. For example, the temperature could be 5, 9, 20, 100, etc degrees. Analog modules use a value, such as 52, rather than a 0 or 1 to represent the state of the device.

The switch is either on or off. Therefore it is a digital device and will interface to a digital, or discrete input module.

The temperature could have many states or values. Therefore it is an analog device and will interface to an analog input module.
Discrete vs. Analog Modules

Discrete Modules

- Devices that are either **on or off**, such as a **pushbutton**, get wired to **discrete** modules. Discrete modules come in a variety of types, such as 24VDC or 120VAC. You can buy discrete modules that allow you to typically connect anywhere from 2 to 32 devices, with the most popular being 16 devices.

- Since it takes only 1 bit to represent the state of a device, a 16 point discrete module only requires 16 bits of memory in the controller to store the states of all the points on the module.
Discrete vs. Analog Modules

Analog Modules

• Devices that have a **number** associated with them, such as a **temperature sensor**, get wired to **analog** modules. Analog modules come in a variety of types, such as 4 to 20 mA or 0 to 10 VDC. You can buy analog modules that allow you to connect anywhere from 2 to 16 devices.

• Since it takes 1 word to represent a number, a 16 point analog module requires 16 words of memory in the controller to store the value of all the numbers on the module. Each word in a PLC takes 16 or 32 bits (depending on the PLC), therefore it takes 16 or 32 times the amount of PLC memory to store analog points vs. digital points.
Output Modules

- Output modules interface directly to devices such as motor starters and lights.
- Output modules take digital signals from the PLC and convert them to electrical signals such as 24VDC and 4 mA that field devices can understand.

Terminal block - this is where wires from the field devices are connected.

Field device - Motor starter, controlling an AC motor.

From PLC

D/A Conversion

Output Module

1 2 3 4 5 6 7 8 9 10 COM
Output Modules

Output modules take a signal from a PLC and convert it to a signal that a field device needs to operate. Since there are different types of output devices, there is a wide variety of output cards available, including both digital and analog cards.

Valve
0-10V DC

Fan
120V AC

Light
24V DC
Basic Components of a PLC System

Chassis/Backplane

All PLCs need some method of communicating between the controller, I/O and communications modules. Here are three ways used to accomplish this communications between the various components that make up the PLC system.

- Modules are installed in the **same chassis** as the PLC and communicate over the chassis backplane.
- Modules are designed to "plug" into each other. The interconnecting plugs form a backplane. There is no chassis.
- Modules are **built into** the PLC. The modules come together in one physical block. The backplane in this case is transparent to the user.
Basic Components of a PLC System

Below is an example of a backplane in a chassis based system. You can see the backplane in the area where the modules are not inserted. The modules have connectors that plug into the black connectors on the backplane. All of the connectors on the backplane are connected together electrically.
Chassis and Backplane Examples

Example of a modular based PLC. The PLC slides into the **chassis along with** other modules.

Example of a PLC that **plugs into** adjacent modules to form a backplane with no chassis required.

Example of a PLC with the modules **built in**. Comes as one block.

Slide modules into available slots.

Each module plugs into the one to the left.

Connections for the built in I/O.
Benefits of the Different Forms

- Great flexibility in choice of modules. Modules can be easily installed or removed without affecting other modules.

- Great flexibility in choice of modules. In some cases modules cannot be removed without “breaking the chain” and affecting all modules downstream. No chassis cost.

- Low cost solution but limited flexibility. Generally used in smaller, simpler systems.

![Diagram showing module installation and connections](image-url)
Basic Components of a PLC System

Power Supply

A power supply is needed to provide power to the PLC and any other modules. Power supplies come in various forms:

- Power supply modules that fit into one of the slots in a chassis
- External power supplies that mount to the outside of a chassis
- Stand alone power supplies that connect to the PLC or I/O through a power cable
- Embedded power supplies that come as part of the PLC block.

Power supply mounted to the side of the chassis

Power supply that fits in a slot of the chassis
Basic Components of a PLC System

Programming Software

Software that runs on a PC is required to configure and program PLCs.

- Different products may require different programming software
- Software allows programs to be written in several different languages
Network Interface

Most PLCs have the ability to communicate with other devices. These devices include computers running programming software, or collecting data about the manufacturing process, a terminal that lets an operator enter commands into the PLC, or I/O that is located in a remote location from the PLC. The PLC will communicate to the other devices through a network interface.
Network Interface

In some cases the network interface is built directly into the PLC and will appear as a plug on the front of the PLC.

Modular systems allow the user to add network interface modules to the chassis. The controller communicates to the network interface over the backplane.
Typically, PLCs are installed in enclosures, on a “panel”

- Power supply mounted to the side of the chassis
- PLC controller
- I/O modules
- Network interface modules
- Wiring to field devices
- Backplane of chassis
The PLC system is the center of a control system, but it is not the entire control system. There are several other key pieces that must be added to a PLC system to make a complete control system. Examples are:

- Operator terminals
- Networks
- Distributed I/O devices (I/O that is in a different location than the PLC)
PLC PROGRAMMING
Every PLC has associated programming software that allows the user to enter a program into the PLC.

- Software used today is Windows based, and can be run on any PC.
- Different products may require different software: PLC5, SLC, and ControlLogix each require their own programming software.

Example of PLC programming software
Before a PLC can perform any control task, it must be programmed to do so. The most popular language used to program a PLC is ladder logic.

Example of a ladder logic program
In a conveyor system, we have several “requirements” to accomplish; for example, timing and counting parts on the conveyor. Each of these requirements must be programmed into the PLC so that it knows how to respond to different events.
The programmer develops the program, and connects their personal computer to the PLC through a network or cable and then downloads the program to the PLC.
Here, we can see an example of ladder logic. Each line of code is known as a “rung”. In this example there are 4 rungs, numbered 0, 1 and 2, and the end rung marking the end of the program.

The PLC executes the program 1 rung at a time, starting with the first rung and then working down.
Ladder logic rungs are basically IF-THEN statements. Each individual rung is executed from the left to the right. The outputs at the right side of each rung is set to a condition that reflects the status of the permissive contacts in a particular rung.
This is a very simple rung of logic, from a PLC program:

```
Start Button          Motor
```

Let’s take a look at this simple program in detail…
The first instruction is as an “eXamine If Closed”, or “XIC” instruction.

In this example, if the actual Start Button is on, then the value of all the XICs named Start Button, in the program will be true (also known as a ‘one’, or “closed”).

If the start button is off (not on) then the value of the Start Button XICs will be false (also known as a “zero”, or “open”).
The second instruction is known as an Output Energize, or “OTE”. This is an output instruction.

This instruction turns **on** if the logic to the left of the OTE is true. If the logic is false, then the output will be turned **off**. The OTE commands a physical output located on a output module to turn either on or off.
If there are multiple XIC’s on the rung, then all would have to be on for the rung to be true.

All the inputs have to be on for the rung to be true
If there are multiple OTE’s on the rung, then all would be turned on or off based on the rung condition (true or false).

All the outputs will turn on if the rung is true.
The text above the XIC and OTE is the **address** associated with the instruction. PLC addresses may appear in many different ways depending on the PLC being used.

- Start_Button
- Local:2:I.Data[0].1
- I:020/2

The address is used by the PLC to tell exactly which input to read or which output to command.
Since a PLC will be controlling real devices down on a plant floor, it has to have some way of communicating to the correct device. All PLCs use some sort of method of **I/O Addressing** to perform this function.

- I/O addresses are a means to tie a physical I/O point to a location in PLC memory
- An input address will represent the state of an input device, i.e. the switch is on or off.
- An output address will represent the commanded state for a device. i.e., turn the motor on or off.

**Example of input address representing an actual input**
Local:5:I.Data[0].1

**Example of output address representing an actual output**
Local:7:O.Data[6].11
Often, a descriptive name of the device connected to the I/O point is used in addition to, or in place of the base I/O address which describes the physical location of the module in the rack.

Example of symbolic input address:
Start Button
Local:5:I.Data[0].1

Example of base Input address:
Motor
Local:7:O.Data[6].11
I/O addresses are a means to tie a physical I/O point to a location in PLC memory. There are other addresses that do not connect to physical I/O, but are used to hold a value.

Data addresses store a value used for functions like timers, counters, or calculations.

Example of I/O address: `Part_detected`
Example of data address: `Counter part_count`
In this example, parts on a conveyor are counted.

The input, “Part_detected” looks at the I/O address of the actual sensor counting the parts.

The counter references a data address, and accumulates the counts in that location in memory.

Each time the “Part_detected” switch closes, the counter adds one more count to the area of memory called “part_count”
Next, we will take a look at an actual segment of ladder logic code. The code will be the code used to control our conveyor from the first lesson. Before we do that, let’s take a look at a few more ladder logic instructions:

- **OTL - Output Latch**: Turns on the output and keeps it even if the rung goes false.
- **OTU - Output Unlatch**: Turns off the output when the rung is true.
- **TON - Timer On Delay**: When the rung is true the timer will run. It will store the elapsed time in the “Accum” field (accumulator). As long as the rung remains true it will count until it reaches the preset value. If the preset value is hit the DN bit will go on (Done bit). When the rung goes false the timer will be reset.
Now let's take a look at a simple program for a conveyor application.

When a box is placed on the conveyor in front of Photoeye 1, Light 1, and Motor 1 will turn on, causing the box to move down the conveyor to the left. When the box passes in front of Photoeye 2, Motor 1 and Light 1 will turn off, stopping the conveyor.
Here’s the program for the conveyor: The first line of code turns on the motor and the light when a box is detected by photoeye1. Likewise, the motor and light are turned off when photoeye2 detects the box in the second line of code.
The third line begins a timer when the box passes by photoeye1, and if the box does not pass by photoeye2 in 30 seconds (the timer counts in milliseconds), the motor and light are shut off by line 4. This is the indication of a jam condition.
PLC Addressing Examples

- **Logix Controller sample addresses:**
  - `Motor_start` - Binary Tag
  - `Tank_temp` - Integer Tag
  - `Local:5:I.Data.0` - Input Tag (Local - same chassis as processor, in slot 5, is an input, data bit 0)

- **PLC-5 sample addresses:**
  - `B3:0/2` - Binary File
  - `N7:0` - Integer File
  - `I:012/3` - Input file (Describes: Rack#, Group, Bit)

- **SLC sample addresses:**
  - `B3:0/2` - Binary File
  - `N7:0` - Integer File
  - `I:3.0/2` - Input file (Describes: Slot.word/bit)
While ladder logic is the oldest and most popular language used in PLCs today, many other languages are gaining in popularity and are in wide use. Examples are:

- **Sequential Function Chart (SFC)**

- **Function Block**

- **Structured Text**

- **Higher level languages such as C.**
ARCHITECTURE
A complete control system is made up of a combination of PLCs, networks, I/O, terminals and software. All the components work together to form a complete control system.
Within a control system architecture there are many subsystems and terms used to describe them. This section will go over some of the popularly used terms to describe parts of control systems.

- Local I/O
- Distributed I/O
- Centralized I/O
- Centralized Control
- Distributed Control
- Control System
- Data Acquisition System
- Safety System
I/O Systems

I/O systems are often referred to as local or distributed.

- **Local** refers to the I/O being attached directly to the Controller or on the same backplane as the Controller.
- **Distributed** refers to I/O which is not on the same backplane as the Controller. Distributed I/O is connected using a network.

The distributed input module sends the inputs across the backplane to the adapter.

The adapter sends them over the I/O network to the PLC (Controller).
I/O architectures are made up of I/O systems. The architectures are referred to as Centralized and Distributed.

- **Centralized** refers to the I/O being located near or in the same cabinet as the processor. Wires from field devices are brought back to the I/O, and can be quite long.

- **Distributed** refers to I/O that is located near the field devices. The wires from the field devices are short. The network cable is run out to the Distributed I/O instead of running the field wires back to the I/O.
• I/O systems are part of an I/O architecture.
  – If all the I/O is located near or in the same cabinet as the processor, it is a centralized architecture. Within the centralized architecture could be either local or distributed I/O, or both.

Local I/O - I/O that is attached to or in the same chassis as the PLC

Distributed I/O - I/O that is connected to the PLC over a network.

Centralized architecture made up of both local I/O and distributed I/O
Systems vs. Architectures

• I/O systems are part of an I/O architecture.
  – If some of the I/O is located remotely from the processor it is a distributed architecture. Within the distributed architecture is distributed I/O, or both local and distributed I/O.

Distributed architecture made up of both local I/O and distributed I/O
Local vs. Distributed I/O System

Why use Local I/O?

• Faster than distributed I/O
• Easy to install - add a module to the chassis
  • In some cases the I/O is already attached to the processor
• Less expensive than adding distributed
  • Use the I/O on the processor or simply add a module to the backplane. Don’t need to add a chassis, power supply, etc.

Why use Distributed I/O?

• Field devices distributed around the machine - too much wiring to take back to one chassis
• Out of local I/O
  • local I/O limited by number of slots in the backplane or fixed I/O attached to the processor
• Local I/O does not meet your needs
  • module type, current capability, etc.
Centralized vs. Distributed I/O Architecture

Why use a Centralized Architecture?
- If using local only, then it is faster than distributed I/O
- If the only centralized I/O is local I/O, then you don’t have to buy additional chassis, power supply, etc.
- The field devices are near the cabinet or processor so there is no wiring cost savings of going distributed

Why use a Distributed Architecture?
- Field devices distributed around the machine - too much wiring to take back to one chassis
- I/O mounted to machine - becomes a part of it instead of in a central cabinet.

*** Many systems will combine both ***
Centralized control refers to a control system where a single (usually large) PLC controls all of the I/O and performs all the control for the system.

A single PLC controls the local I/O and all the networked I/O.
Centralized Control

Advantages and disadvantages of centralized control

• Advantages
  – The control program is all in one place
  – Easier to troubleshoot system problems
  – I/O performance throughout entire system

• Disadvantages
  – Programs can get quite large
  – related I/O performance slower compared to distributed control
  – Single PLC failure shuts down entire system
Distributed control refers to a control system where multiple PLC controllers share the responsibility of controlling the system. The PLC’s usually communicate frequently with each other.

Multiple PLCs share control of the networked I/O. The sections are often broken up into logical systems. For example, one controller might control the conveyor bringing ingredients to the ovens while the other 2 are controlling the ovens.
Advantages and disadvantages of distributed control

- **Advantages**
  - Segmented programs to specific tasks
  - Easier to troubleshoot local problems
  - Performance on local network

- **Disadvantages**
  - System problems harder to troubleshoot
  - Performance from I/O across PLC’s
  - Cost
  - Maintain multiple programs
The control system is the system that is responsible for the control of the process. This is the system that includes the PLC, all of the I/O and any Human Machine Interfaces (HMI).
The Data Acquisition system is generally responsible for collecting data about the control system, and storing it on master computers or servers, or displaying it on terminals. The data is often used later for reporting or charting purposes.

Made up of devices and networks which are responsible for acquiring data about the process. Not responsible for direct control of the process.
The network used for data acquisition is often Ethernet. While data acquisition devices can exist directly on the control network, a gateway is often used to separate network traffic between the data acquisition system and the control system.
The **Function of a Safety System** is to monitor and control conditions on a machine or process that are hazardous in themselves or, if no action were taken, may give rise to hazardous situations.

The Safety System runs in parallel with the Control System. The Control System and Safety System may share components:

- Focus of Control System is Throughput
- Focus of Safety System is Protection

A Safety system is designed to protect:

- People
- Environment
- Machinery
The safety system is often referred to as “safety control” while the PLC system controlling the devices that produce the end product is often referred to as the “standard control”.

The PLC performing standard control would control the robot while it performs tasks related to producing the desired product.

The safety control would monitor the light curtain to make sure that the operator never moved into the robot area while the robot was moving. If the operator broke the light curtain, the safety system would remove power to the robot so that it could not move and injure the operator.