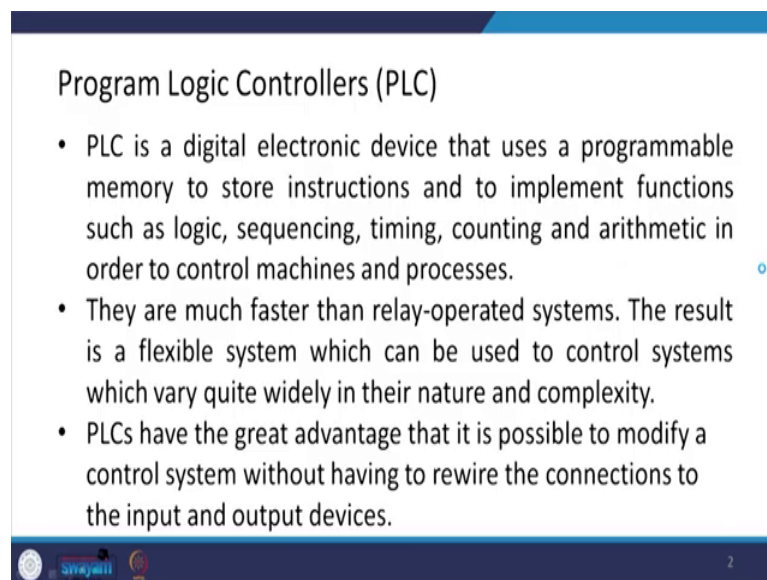


Mechatronics
Prof. Pushparaj Mani Pathak
Department of Mechanical and Industrial Engineering
Indian Institute of Technology, Roorkee

Lecture - 33
Program Logic Controllers

I welcome you all to this NPTEL Online Certificate Course on Mechatronics. Today, we are going to talk about Program Logic Controllers or PLCs. These are used in many industrial automation processes.

(Refer Slide Time: 00:49)



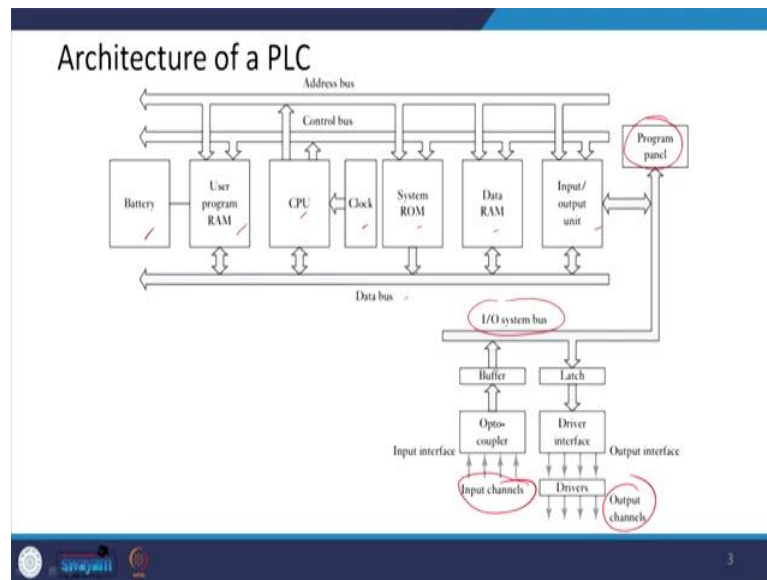
Program Logic Controllers (PLC)

- PLC is a digital electronic device that uses a programmable memory to store instructions and to implement functions such as logic, sequencing, timing, counting and arithmetic in order to control machines and processes.
- They are much faster than relay-operated systems. The result is a flexible system which can be used to control systems which vary quite widely in their nature and complexity.
- PLCs have the great advantage that it is possible to modify a control system without having to rewire the connections to the input and output devices.

swayamii 2

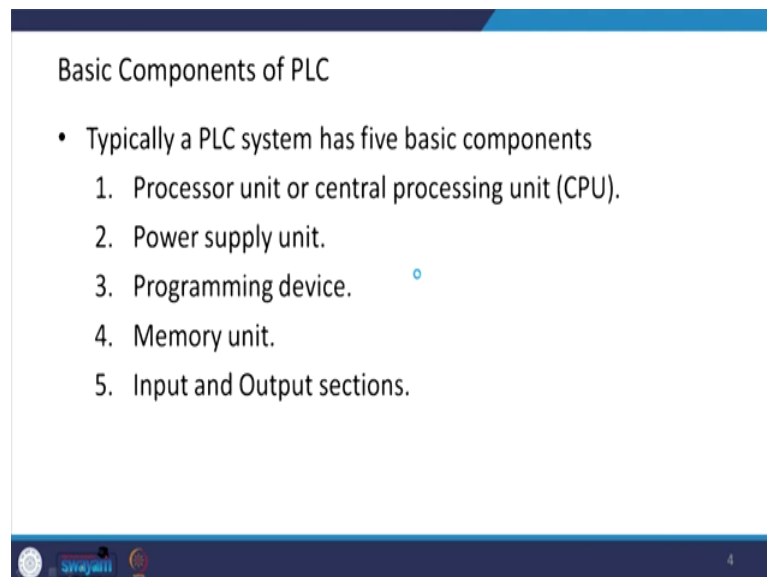
A program logic controller is a digital electronic device that uses a programmable memory to store instruction and to implement functions such as logic, sequencing, timing, counting, and arithmetic in order to control machines and processes. They are much faster than relay-operated systems. The result is a flexible system that can be used to control systems, which vary quite widely in their nature and complexity. PLCs have the great advantage that it is possible to modify a control system without having to rewire the connections to the input and output devices.

(Refer Slide Time: 01:47)



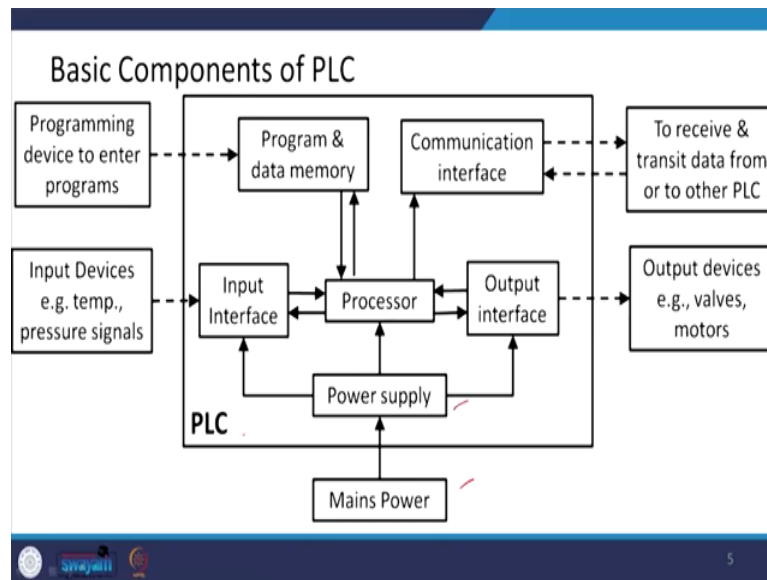
This is the architecture of a PLC. So, you have the input channel, and you have the output channel. Then, you have an input-output bus system, and there are program panels. There is a power supply, there is a RAM, CPU, clock, system ROM, data RAM, and the input-output unit, and you have the address bus as well as the control bus and the data bus.

(Refer Slide Time: 02:24)



Typically a PLC has five basic components. First is the processor unit, or what we call a CPU, then there is a power supply unit, programming device, memory unit, and input and output sections.

(Refer Slide Time: 03:00)



The PLC has a power supply, input interface, output interface, the processor is there, program and data memory is there, then communication interface is there. So, in input interface receives input from input devices such as temperature or pressure signals.

Program and data memory receives instructions that are programming devices to enter the programs. Communication interface to receive and transmit data from or other PLC and output interface, you have the output devices such as valves or motors.

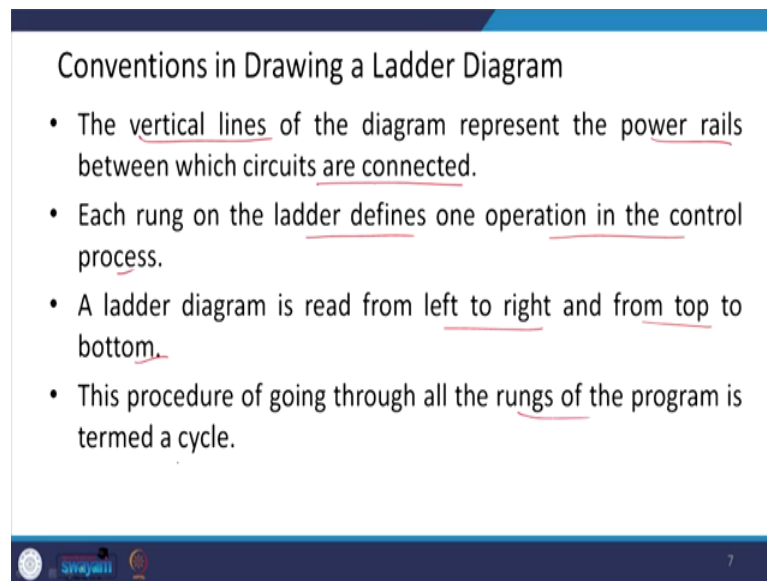
(Refer Slide Time: 03:41)

PLC Programming

- A PLC can be programmed by five different ways: ladder diagrams, instruction list, sequential flow charts, structured text and functional block diagrams.
- Out of these the LADDER Diagram is mostly adopted because it makes easier for engineers with no great knowledge of programming to write programs for PLCs.

Now, let us look at how do we program a PLC. A PLC can be programmed in five different ways; ladder diagram, instruction list, sequential flow chart, structured text, and functional block diagrams. Out of these, the LADDER diagram is mostly adopted because it makes it easier for engineers with no great knowledge of programming to write the programs for PLCs. We will be seeing some examples of programming through the ladder diagram and a convention in drawing a ladder diagram. So, before we proceed further, let us see what the conventions are.

(Refer Slide Time: 04:36)



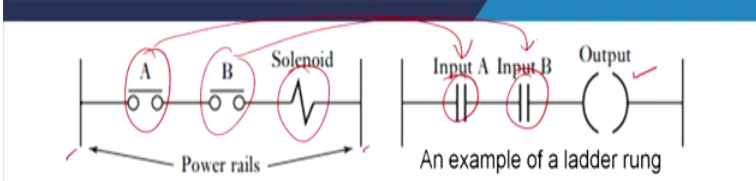
The slide is titled "Conventions in Drawing a Ladder Diagram" and contains four bullet points. The text is as follows:

- The vertical lines of the diagram represent the power rails between which circuits are connected.
- Each rung on the ladder defines one operation in the control process.
- A ladder diagram is read from left to right and from top to bottom.
- This procedure of going through all the rungs of the program is termed a cycle.

The slide also features a logo in the bottom left corner and the number "7" in the bottom right corner.

The vertical lines of the diagram represent the power rails between which the circuits are connected. Each rung on the ladder defines one operation in the control process. The ladder diagram is read from left to right and from top to bottom. So, this is how it is done. And this procedure is going through all the rungs of the program is termed a cycle.

(Refer Slide Time: 04:58)



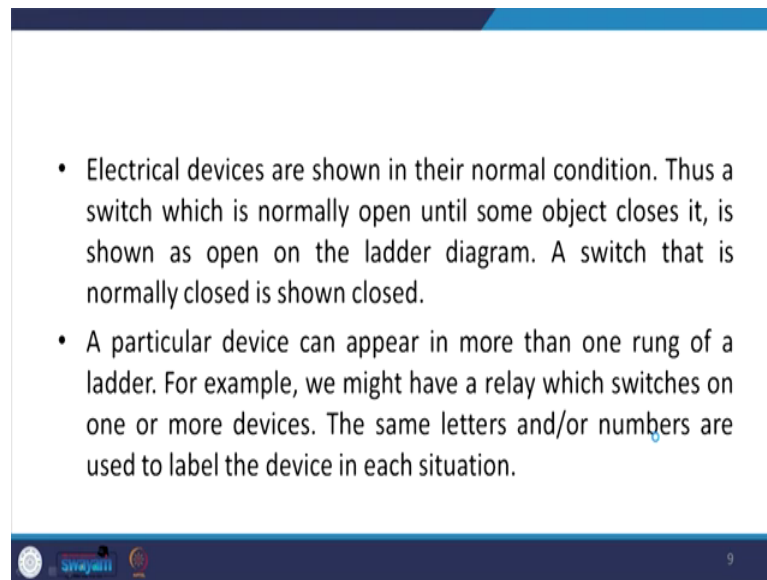
An example of a ladder rung

- Each rung must start with an input or inputs and must end with at least one output, input devices being represented by two short parallel lines to represent switching contacts and output devices being represented by circles.
- The term input is used for a control action, such as closing the contacts of a switch, used as an input to the PLC. The term output is used for a device connected to the output of a PLC, e.g. a motor.

Here each rung must start with input or inputs and must end with at least one output, input devices being represented by two short parallel lines as you can see over here to represent switching contact, and the output devices are represented by a circle as you can see over here. So, here we have the two input devices, input A and input B, and here, you have the output device, and the example could be that you have the power rails over here.

You have a switch A, corresponding to input device A, and you have switch B, corresponding to input device B, and the output could be the activation of a solenoid. So, this is how it is. The term input is used for a control action such as closing the contact of a switch, used as an input to the PLC. The term output is used for the device connected to the output of a PLC. For example, a motor.

(Refer Slide Time: 06:23)



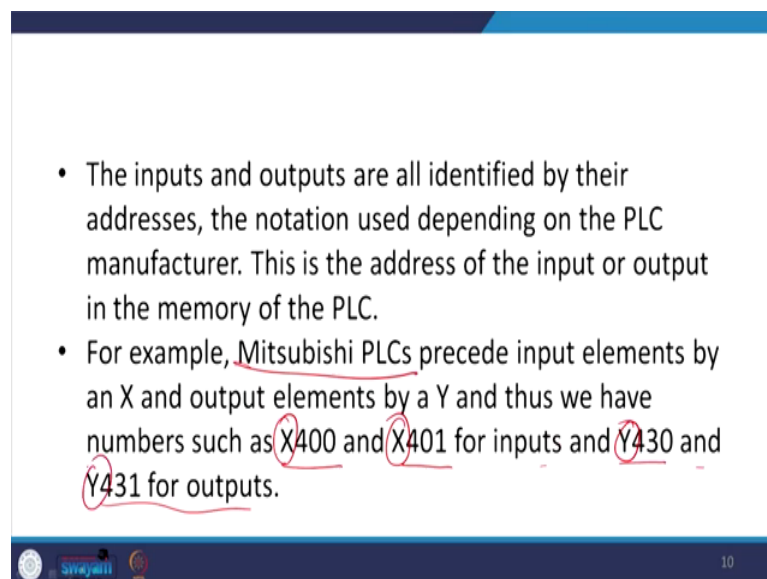
Slide 9 contains two bullet points explaining electrical devices in ladder diagrams. The first point states that devices are shown in their normal condition: a normally open switch is shown as open, and a normally closed switch is shown as closed. The second point states that a device can appear in multiple rungs, with the same labels used for each instance.

- Electrical devices are shown in their normal condition. Thus a switch which is normally open until some object closes it, is shown as open on the ladder diagram. A switch that is normally closed is shown closed.
- A particular device can appear in more than one rung of a ladder. For example, we might have a relay which switches on one or more devices. The same letters and/or numbers are used to label the device in each situation.

9

Electrical devices are shown in their normal condition. Thus, a switch that is normally open until some object closes is shown as open on a ladder diagram. A switch that is normally closed is shown as closed. A particular device can appear in more than one rung of a ladder. For example, we might have a relay that switches on one or more devices. The same relay and, or numbers are used to label the device in each situation. The input and output are all identified by their addresses. The notation used to depend on the PLC manufacturer.

(Refer Slide Time: 07:02)



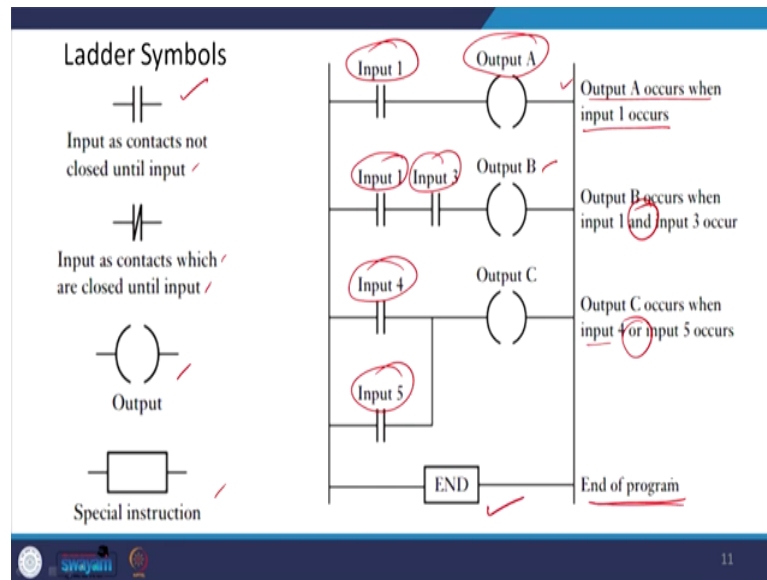
Slide 10 contains two bullet points about PLC addresses. The first point explains that inputs and outputs are identified by their addresses, which vary by manufacturer. The second point gives an example for Mitsubishi PLCs, where inputs are labeled with X and outputs with Y, followed by a number. The numbers X400, X401, Y430, and Y431 are circled in red in the original image.

- The inputs and outputs are all identified by their addresses, the notation used depending on the PLC manufacturer. This is the address of the input or output in the memory of the PLC.
- For example, Mitsubishi PLCs precede input elements by an X and output elements by a Y and thus we have numbers such as X400 and X401 for inputs and Y430 and Y431 for outputs.

10

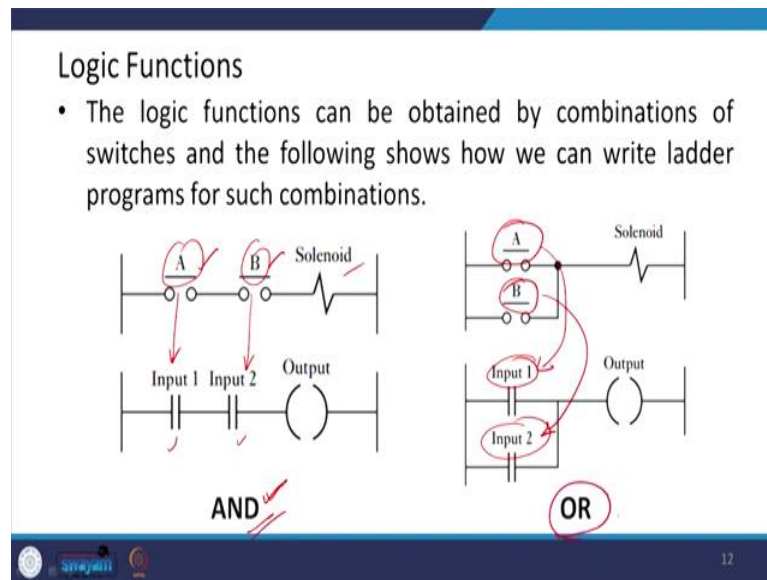
This is the address of the input or output in the memory of the PLC. For example, Mitsubishi PLCs precede input element by an X and output element by a Y, and thus, we have numbers such as X400 and X401 for inputs and Y430 and Y431 for output.

(Refer Slide Time: 07:49)



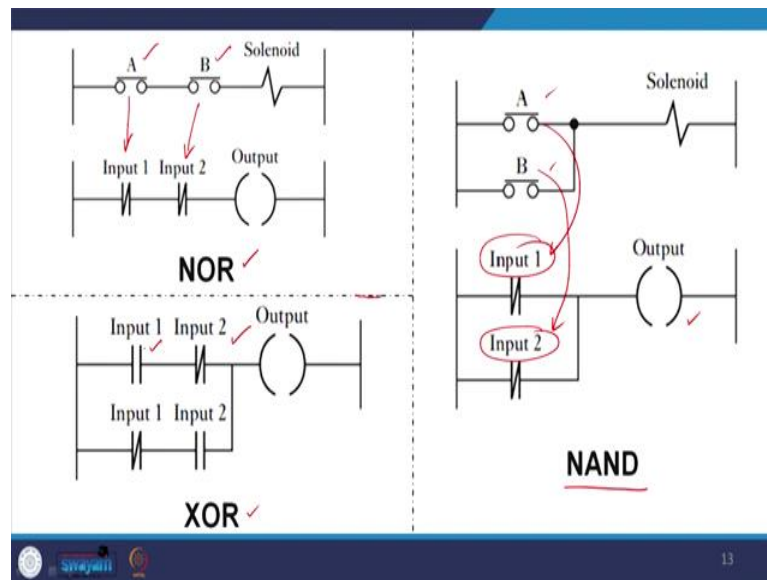
Here these proceed with X for the input and precede with Y for the output. These are the ladder symbols that are used. This is input as contacts not closed until input is available, and this is input as contacts which are closed until input is available, and this is for the output, and these are for any special instruction if you have. So, here if this is the diagram like this, then what does this mean? This means that output A occurs when input A occurs; this is what is meant by. So, you have input 1, and you have output A. Now, this one the second one output B occurs when input 1 and input 3 occurs. If input 1 and input 3 both are there, then only output B will be occurring, and here, output C occurs when input 4 or input 5 occurs. So, here these are in parallel. So, if either one either input 4 or input 5 occurs, then only output C will occur, and here, we have the 'and' one. And then, you can have a box, you can write END, and this indicates the end of the program. So, this is how using the ladder symbol and a PLC can be programmed. Now, let us look at the various logic functions which could be achieved using PLCs. So, the logic function can be obtained by a combination of switches, and the following shows how we can write a ladder program for such a combination.

(Refer Slide Time: 09:51)



Suppose you have a switch A and switch B and, there is a solenoid connected over here. So, when switch A will be on and switch A is in the open position. So, you have input 1 being shown here input 2 being shown here. This corresponds to this one, and this one corresponds to this one. Now, this solenoid will be on only when switching A and switch B both are on. So, if you have input here and input here, then you are going to have the output, and this type of logic is what is called the AND logic. We have just seen this in the previous slide. Similarly, I can define the OR logic, and here, in the case of OR, switch A and switch B, they are in parallel. So, we have input 1 and input 2, they are in parallel, and you have the output. So, in this case, you have the output if either of the input is available, and this is what we call the OR logic.

(Refer Slide Time: 11:08)



We can also see the other logics, such as NOR logic which is OR inverted, and that we can get with this type of combination. I am not explaining here the truth table corresponding to these because we have already seen the truth table in the digital logic lecture. So, if you want to see it corresponding to each gate what the truth table is going to be, please refer to my lecture on Digital Logic. So, here for the NOR, this is the switch A, and this is the switch B, which is in the normally closed position, and the inputs are going to be represented for this one by this way and for B this way and then only you will be getting the output. So, after the NOR gate, which is inverted OR, now let us look at the NAND gate, which is inverted AND. Inverted AND gate we can get by using the two switches A and B, which are normally closed, and corresponding inputs are going to be 1 and 2 over here. So, this is going to be there, and here, we are going to have the output. The truth table for the NAND gate is going to be the same as we have seen earlier in my lecture on Digital Logic. Now, let us look at the XOR gate and here what we use to program in a PLC. We use one input and one input which is normally a switch that is normally open, and one which is normally closed. So, they are put in series, and then, in parallel with both of these as shown in the figure here, we have one input 1, which is normally closed, and another one which is normally open. Then these are connected with output. Now, let us see the latching and internal relays. Latch circuit handles the situation, where it is necessary to hold a coil energized even when the input which energized it ceases. It is a self-maintaining circuit that is, after being energized, maintains that last state until other input is received.

(Refer Slide Time: 14:26)

- When Input 1 is energised and closes, there is an output.
- However, when there is an output, a set of contacts associated with the output is energised and closes.
- These contacts OR the Input 1 contacts.
- Thus, even if Input 1 contacts open, the circuit will still maintain the output energised.
- The only way to release the output is by operating the normally closed contact Input 2.

Latch Circuit

When input 1 is energized and closes, there is an output. And however, when there is an output, a set of contact associated with the output is energized and closes. These contacts OR the input 1 contact. Thus, even if input 1 contacts open, the circuit will still maintain the output energized. The only way to release the output is by operating the normally closed contact input 2. So, this is how the latch circuit can be represented in the ladder diagram. We have input 1, we have input two, and here is the output, and we have the same output being put over here. Now, let us look at the internal relays.

(Refer Slide Time: 15:45)

Internal Relays

- The internal relay, auxiliary relay or marker behave like relays with their associated contacts, but in reality are not actual relays but simulations by the software of the PLC. ◦
- Some have battery back-up so that they can be used in circuits to ensure a safe shut-down of plant in the event of a power failure.
- Internal relays are often used when there are programs with multiple input conditions.

The internal relay or auxiliary relay or marker behaves like relays with their associated contacts but, in reality, are not actual relays but simulations by the software of the PLCs. Some have battery backup so that they can be used in the circuit to ensure a safe shut-down of the plant in the event of a power failure. Internal relays are often used when there are programs with multiple input conditions. Consider the situation where the excitation of output depends on the two different input arrangements.

(Refer Slide Time: 16:34)

- Consider the situation where the excitation of an output depends on two different input arrangements.
- The first rung shows one input arrangement being used to control the coil of internal relay IR1.
- The second rung shows the other input arrangement controlling the coil of internal relay IR2.
- The contacts of the two relays are then put in an OR situation to control the output.

An output controlled by two input arrangements

The first rung shows one input arrangement being used to control the coil of the internal relay IR1. The second rung shows the other input arrangement controlling the coil of internal relay IR2. The contacts of the two relays are then put in an OR situation to control the outputs. So, here you can see these IR1 and IR2 are put in parallel to control the output. So, that is there. So, here you can see that output is controlled by two input arrangements.

(Refer Slide Time: 17:23)

- **Starting of multiple outputs:** Another use of internal relays is for the starting of multiple outputs.
- When the start contacts are closed, the internal relay is activated and latches the input.
- It also starts Output 1 and makes it possible for Outputs 2 and 3 to be activated.

18

We may also have the starting of the multiple outputs. Another use of internal relays is for the starting of multiple outputs. When the start contacts are closed, the internal relay is activated and latches the input as shown over here. And it also starts output one and makes it possible for outputs 2 and 3 to be activated.

Now, let us look at timers. Timers behave like relays with a coil which, when energized, result in the closure or opening of contact after some preset time. The timer is thus treated as an output for a rung, with control being exercised over pairs of contact elsewhere.

(Refer Slide Time: 18:33)

Timers

- Timers behaves like relays with coils which when energised result in the closure or opening of contacts after some preset time.
- The timer is thus treated as an output for a rung with control being exercised over pairs of contacts elsewhere.

Timer contacts

Delay-on Timer

19

So, we could have an input and timer, and then you can have the output. So, time delays before the input signal reach the output, and here we could have some input to get the output for this and time delay before output could be achieved by this way. Others consider a timer as a delay block in a rung that delays the signal in the rung reaching the output. So, this is how it could be. So, this could be input, and you have this much delay, and this is your timer output.

(Refer Slide Time: 19:24)

- Others consider a timer as a delay block in a rung which delays signals in that rung reaching the output.

The diagram consists of two parts. The left part is labeled 'On-Delay Time TON'. It shows an 'Input' signal that is a rectangular pulse. Below it, a 'Timer output' signal is shown. The output pulse starts after a certain time delay from the start of the input pulse. A double-headed arrow labeled 'Delay' indicates this time interval. The right part is labeled 'Off-Delay Time TOFF'. It shows an 'Input' signal that is a rectangular pulse. Below it, a 'Timer output' signal is shown. The output pulse continues for a certain time delay after the input pulse has ended. A double-headed arrow labeled 'Delay' indicates this time interval.

- Such a timer waits for a fixed delay period before turning on, e.g. a period which can be set between 0.1 and 999 s in steps of 0.1 s. Other time-delay ranges and steps are possible.

Here, you have this in input, and you have timer output. So, this much is the delay. So, this is Off-Delay Time or TOFF, and this is the On-Delay Time TON. Such a timer waits for a fixed delay period before turning on. For example, a period that can be set between 0.1 and 999 seconds in steps of 0.1 seconds. Other time-delay ranges and steps are possible. The timer can be used for sequencing for different activities.

(Refer Slide Time: 20:12)

- The timer can be used for sequencing as explained below. When the input In 1 is on, the output Out 1 is switched on. The contacts associated with this output then start the timer. The contacts of the timer will close after the preset time delay. When this happens, output Out 2 is switched on.

Timed sequence

For example, when input 1 is on, the output out 1 is switched on. So, this is there. The contact associated with this output and then start the timer. So, this output starts the timer, and the contact of the timer will close after the preset time delay. When this happens, output 2 is switched on. So, from here, this timer, you have the output 2 .

(Refer Slide Time: 20:56)

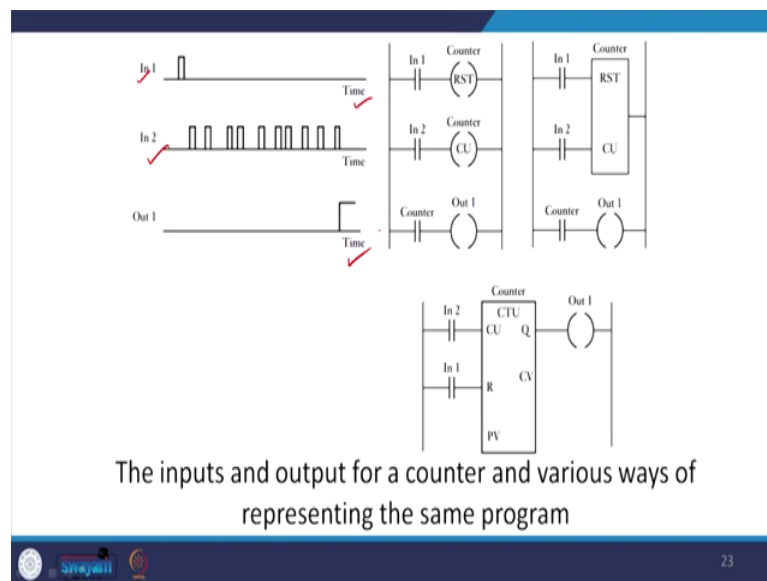
Counters

- Counters are used when there is a need to count a specified number of contact operations, e.g. where items pass along a conveyor into boxes, and when the specified number of items has passed into a box, the next item is diverted into another box.
- **Down-counter** : The counter counts down from the present value to zero, i.e. events are subtracted from the set value. When zero is reached the counter's contact changes state.
- **Up-Counter** : count up to the preset value, i.e. events are added until the number reaches the set value. When the set value is reached the counter's contact changes state.

Now, let us look at the counters. Counters are used when there is a need to count a specified number of contact operations. For example, when items pass along a conveyor into boxes and when the specified number of items has passed into a box, the next item is diverted

into another box. So, this type of operation requires counters. So, we may have the down-counter, and we may have the up-counter. In the down-counter, the counter counts down from the present value to 0. That is, events are subtracted from the set value, and when 0 is reached, the counter contact state changes. In the up-counter, the counts up to the preset value. So, events are added until the number reaches the set value, and when the set value is reached, the counters contact state changes.

(Refer Slide Time: 22:06)



This is how it is the input and output for a counter and various ways of representing the same in program 1. So, you have input 1, and then you have here, then you here you have input 2, and you have the output 1. This is how it can be represented over here. Then, the master control relay, a whole block of outputs can be simultaneously turned off or on by using the same internal relay contact in each output rung so that switching it on or off affects every one of the rungs.

(Refer Slide Time: 22:58)

Master Control Relay

- A whole block of outputs can be simultaneously turned off or on by using the same internal relay contacts in each output rung so that switching it on or off affects every one of the rungs.

24

It can have the master control relay MC 1 over here and that controlling the others.

(Refer Slide Time: 23:10)

Jump

- Jump enables programs to be designed so that if a certain condition exists then a section of the program is jumped.

25

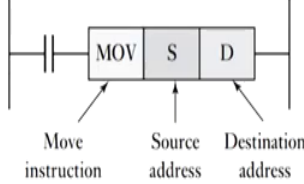
Then, let us look at the jump. The jump enables the program to be designed so that if a certain condition exists, then a section of the program is jumped. So, that is there. So, you could have carried out a program A. Is input 1 is on? Yes. Then, you move on here, carry on the program C and if it is not, then carry on the program B. So, you, we could have it being represented like this program A, program B, and then, you can have the program C.

Now, let us look at the handling of data. So, in data handling, we have data movement, data comparison, arithmetic operation, and code conversion. So, how the data movement is there in the case of PLCs?

(Refer Slide Time: 24:14)

Data Movement

- For data movement the instruction will contain the move data instruction, the source address of the data and the destination address of the data.



- Such data transfers might be to move a constant into a data register, a time or count value to a data register, data from a data register to a timer or counter, data from a data register to an output, input data to a data register, etc.

So, for data movement, the instruction will contain the move data instruction that is the source address of the data and the destination address of the data. So, you have the move-in instruction, you have the source address, and you have the destination address. Such data transfer might be to move a constant into a data register, a time or counter value to a data register, data from a data register to timer or counter, data from a data register to an output, or input data to a data register etcetera.

(Refer Slide Time: 25:02)

Data Comparison

- PLCs can generally make the data comparisons of
 - Less than (usually denoted by $<$ or LES),
 - Equal to ($=$ or EQU),
 - Less than or equal to (\leq or \leq or LEQ),
 - Greater than ($>$ or GRT),
 - Greater than or equal to (\geq , \geq or GEQ) and
 - Not equal to (\neq or \neq or NEQ).

Compare instruction Source address Destination address

Compare Data

28

Then, for the data comparison, PLCs can generally make the data comparison of less than usually denoted by less-than symbol or LES; equal to denoted by equal sign or EQU; less than or equal to this way; greater than or equal to or not equal to. So, here, you have the compare instruction, you have the source address, and you have the destination address. Such a comparison might be used when the signal from two sensors is to be compared by the PLC before an action is taken.

(Refer Slide Time: 25:44)

- Such a comparison might be used when the signals from two sensors are to be compared by the PLC before action is taken.
- For example, an alarm might be required to be sounded if a sensor indicates a temperature above 80°C and remain sounding until the temperature falls below 70°C .

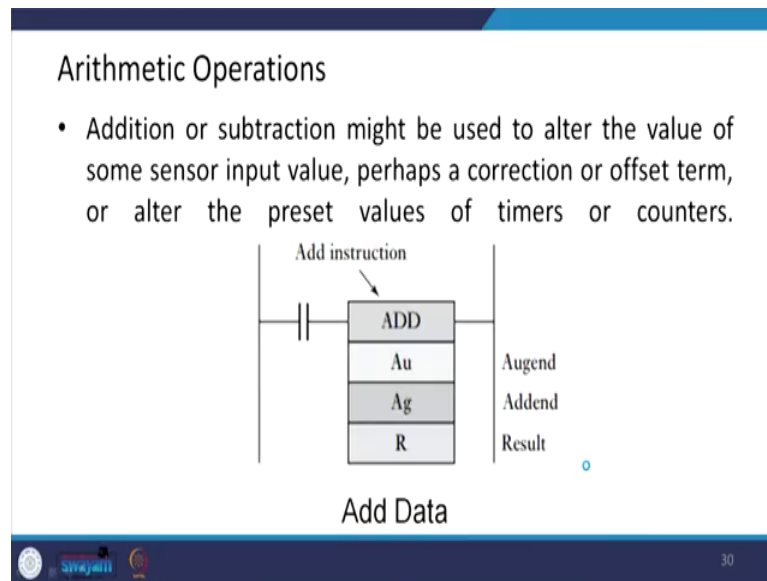
Temperature Alarm

29

For example, an alarm might be required to be sounded if a sensor indicates a temperature above $80^{\circ}C$ and remain sounding until the temperature falls below $70^{\circ}C$. So, this is how it could be done. So, you could have the temperature alarm, and a PLC programming could be like this. So, for a $K = 80$, that is $80^{\circ}C$, you could have an alarm over here, and this is for $K = 70^{\circ}C$.

Then, let us look at the arithmetic operations.

(Refer Slide Time: 26:43)



Addition or subtraction might be used to alter the value of some sensor input value, perhaps a correction or offset terms or alter the present value of the timer or counter. So, we could have the ADD instructions here or Augend, Addend, or the Result. So, this is how we could have the arithmetic operations. Then, the code conversion. All the internal operations in the CPU of a PLC are carried out using binary numbers.

(Refer Slide Time: 27:19)

Code Conversions

- All the internal operations in the CPU of a PLC are carried out using binary numbers.
- Thus, when the input is a signal which is decimal, conversion to BCD is used. Likewise, where a decimal output is required, conversion to decimal is required.
- The data at the source address is in BCD and converted to binary and placed at the destination address.

BCD to Binary

Convert to binary instruction Source address Destination address

31

Thus, when the input is a signal which is decimal, conversion to BCD is used. Likewise, where a decimal output is required, conversion to the decimal is required. So, the data at the source address is in the BCD and converted to binary and placed at the destination over here. The BCD to binary could be converted like this; convert to binary instruction, source address, and you could have the destination address.

For further reading, you can refer to *The Mechatronics* by Bolton, where you will find out an elaborate discussion on PLCs.

Thank you.