Industrial Automation using PLC

Submitted in partial fulfillment of the requirements

For the degree of

Bachelor of Technology

By

Aditya Jagannathan (Roll No. U07PR837)

Supervisor:

Prof. S.N. Gaderia

PRODUCTION ENGINEERING DEPARTMENT
S.V. NATIONAL INSTITUTE OF TECHNOLOGY
SURAT
NOVEMBER 2010

Approval Sheet

Seminar entitled — “Industrial Automation using PLC” by

Aditya Jagannathan is approved.
1. Industrial Automation
   1.0 Introduction
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Other goals of automation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Advantages and Disadvantages of Automation</td>
<td>2</td>
</tr>
<tr>
<td>2. Programmable Logic Controllers</td>
<td>3</td>
</tr>
<tr>
<td>2.0 Introduction to Programmable Logic Controller</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Development of PLC</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Components of PLC</td>
<td>4</td>
</tr>
<tr>
<td>3. Comparison with Hard Wired Relay</td>
<td>6</td>
</tr>
<tr>
<td>3.0 A Brief overview about Hard wired Relay</td>
<td>6</td>
</tr>
<tr>
<td>3.1 Disadvantages of a classic control panel</td>
<td>6</td>
</tr>
<tr>
<td>3.2 Advantages of PLC Control Panel</td>
<td>7</td>
</tr>
<tr>
<td>4. Key concepts to understand PLC computing</td>
<td>8</td>
</tr>
<tr>
<td>4.0 Real Time Computing</td>
<td>8</td>
</tr>
<tr>
<td>4.1 The Importance of Time Based Control</td>
<td>8</td>
</tr>
<tr>
<td>4.1.0 Scan-Based Control</td>
<td>9</td>
</tr>
<tr>
<td>4.1.1 Event-Triggered Control</td>
<td>9</td>
</tr>
<tr>
<td>4.1.2 Time-Based Control</td>
<td>9</td>
</tr>
<tr>
<td>5. Ladder Logic</td>
<td>11</td>
</tr>
<tr>
<td>5.0 Introduction</td>
<td>11</td>
</tr>
<tr>
<td>5.1 Comparison to Relay Logic</td>
<td>11</td>
</tr>
<tr>
<td>6. Ladder Logic Programming</td>
<td>15</td>
</tr>
<tr>
<td>6.0 Basics of Ladder Logic</td>
<td>15</td>
</tr>
<tr>
<td>6.1 Basic Symbols and Notations</td>
<td>15</td>
</tr>
<tr>
<td>6.1.1 Normally Open Contact</td>
<td>15</td>
</tr>
<tr>
<td>6.1.2 Normally Open Coil</td>
<td>16</td>
</tr>
<tr>
<td>6.1.3 Normally Closed Contact</td>
<td>16</td>
</tr>
<tr>
<td>6.1.4 Normally Closed Coil</td>
<td>16</td>
</tr>
<tr>
<td>6.1.5 Basic AND &amp; OR Gates</td>
<td>16</td>
</tr>
</tbody>
</table>
5. Figure 3.1 Typical PLC Control Panel  
6. Figure 4.0 Jitter delays for Input-Process-Output Sequence  
7. Figure 5.0 Basic Ladder Logic Program  
8. Figure 5.1 Electromechanical Relay  
9. Figure 5.2. Basic Relay Logic Circuit  
10. Figure 5.3 Relay Logic Circuit with Jog function added  
11. Figure 5.4 Relay Logic Circuit with Jog function and Status Indicators  
12. Figure 5.5 Complex Ladder Diagram  
13. Figure 6.0 Simplified Logic Circuit  
14. Figure 6.1 Simplified AND gate  
15. Figure 6.2 Simplified OR gate  
16. Figure 6.3 On-Delay Timer (RTO)  
17. Figure 6.4 Up Counter  
18. Figure 7.0 Ladder Diagram in Relay Logic  
19. Figure 7.1 Relay Diagram with overload removed  
20. Figure 7.2 Relay Circuit with Addition of Jog Function  
21. Figure 7.3 Relay Circuit with Addition of Status Indicators  
22. Figure 7.4 Relay Logic Diagram Converted to PLC Ladder Logic  
23. Figure 8.0 Human Machine Interface (HMI)
24. Figure 8.1 Schematic diagram of an opto-isolator 22

List of tables:

1. Table 4.1 Relative delta jitter delay 10
Chapter 1. Industrial Automation

1.0 Introduction

Automation is the use of control systems and information technologies to reduce the need for human work in the production of goods and services. In the scope of industrialization, automation is a step beyond mechanization. Whereas mechanization provided human operators with machinery to assist them with the muscular requirements of work, automation greatly decreases the need for human sensory and mental requirements as well. Automation plays an increasingly important role in the world economy and in daily experience.

Automation has had a notable impact in a wide range of highly visible industries beyond manufacturing. Once-ubiquitous telephone operators have been replaced largely by automated telephone switchboards and answering machines. Medical processes such as primary screening in electrocardiography or radiography and laboratory analysis of human genes, sera, cells, and tissues are carried out at much greater speed and accuracy by automated systems. Automated teller machines have reduced the need for bank visits to obtain cash and carry out transactions. In general, automation has been responsible for the shift in the world economy from industrial jobs to service jobs in the 20th and 21st centuries.

1.1 Other goals of automation (beyond productivity gains and cost reduction)

In manufacturing, the purpose of automation has shifted to issues broader than productivity and costs.

1) Reliability and precision

The old focus on using automation simply to increase productivity and reduce costs was seen to be short-sighted, because it is also necessary to provide a skilled workforce who can make repairs and manage the machinery. Moreover, the initial costs of automation were high and often could not be recovered by the time entirely new manufacturing processes replaced the old. (Japan's "robot junkyards" were once world famous in the manufacturing industry.)

Automation is now often applied primarily to increase quality in the manufacturing process, where automation can increase quality substantially. For example, automobile and truck pistons used to be installed into engines manually. This is rapidly being transitioned to automated machine installation, because the error rate for manual installment was around 1-1.5%, but has been reduced to 0.00001% with automation.

2) Health and environment

The costs of automation to the environment are different depending on the technology, product or engine automated. There are automated engines that consume more energy resources from the Earth in comparison with previous...
engines and those that do the opposite too. Hazardous operations, such as oil refining, the manufacturing of industrial chemicals, and all forms of metal working were always early contenders for automation.

3) Convertibility and turnaround time
Another major shift in automation is the increased demand for flexibility and convertibility in manufacturing processes. Manufacturers are increasingly demanding the ability to easily switch from manufacturing Product A to manufacturing Product B without having to completely rebuild the production lines. Flexibility and distributed processes have led to the introduction of Automated Guided Vehicles with Natural Features Navigation.

Digital electronics helped too. Former analogue-based instrumentation was replaced by digital equivalents which can be more accurate and flexible, and offer greater scope for more sophisticated configuration, parameterization and operation. This was accompanied by the fieldbus revolution which provided a networked (i.e. a single cable) means of communicating between control systems and field level instrumentation, eliminating hard-wiring.

1.2 Advantages and Disadvantages of Automation
The main advantages of automation are:

- Replacing human operators in tasks that involve hard physical or monotonous work.
- Replacing humans in tasks done in dangerous environments (i.e. fire, space, volcanoes, nuclear facilities, underwater, etc.)
- Performing tasks that are beyond human capabilities of size, weight, speed, endurance, etc.
- Economy improvement: Automation may improve in economy of enterprises, society or most of humanity. For example, when an enterprise invests in automation, technology recovers its investment; or when a state or country increases its income due to automation like Germany or Japan in the 20th Century.

The main disadvantages of automation are:

- Technology limits. Current technology is unable to automate all the desired tasks.
- Unpredictable development costs. The research and development cost of automating a process may exceed the cost saved by the automation itself.
- High initial cost. The automation of a new product or plant requires a huge initial investment in comparison with the unit cost of the product, although the cost of automation is spread in many product batches.
2.0 Introduction to Programmable Logic Controller
A programmable logic controller (PLC) or programmable controller is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines, amusement rides, or lighting fixtures. PLCs are used in many industries and machines. Unlike general-purpose computers, the PLC is designed for multiple inputs and output arrangements, extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. Programs to control machine operation are typically stored in battery-backed or non-volatile memory. A PLC is an example of a real time system since output results must be produced in response to input conditions within a bounded time, otherwise unintended operation will result.

A PLC is subject to many a conventions varying with the kind of manufacturer. The notations or what is called the ‘syntax’ in the programming language is specific to the manufacturer too.

Figure 2.0 Pictorial View of PLC in Industry

2.1 Development of PLC
Early PLCs were designed to replace relay logic systems. These PLCs were programmed
in "ladder logic", which strongly resembles a schematic diagram of relay logic. This program notation was chosen to reduce training demands for the existing technicians.

Modern PLCs can be programmed in a variety of ways, from ladder logic to more traditional programming languages such as BASIC and C. Another method is State Logic, a very high-level programming language designed to program PLCs based on state transition diagrams.

Many early PLCs did not have accompanying programming terminals that were capable of graphical representation of the logic, and so the logic was instead represented as a series of logic expressions in some version of Boolean format, similar to Boolean algebra. As programming terminals evolved, it became more common for ladder logic to be used, for the aforementioned reasons. A primary reason for this is that PLCs solve the logic in a predictable and repeating sequence, and ladder logic allows the programmer to see any issues with the timing of the logic sequence more easily than would be possible in other formats.

![Figure 2.1 Example of a PLC: Festo IPC PS1 Professional](image)

**2.2 Components of PLC**

The function of an input module is to convert incoming signals into signals, which can be processed by the PLC, and to pass these to the central control unit. The reverse task is performed by an output module. This converts the PLC signal into signals suitable for the actuators. The actual processing of the signals is effected in the central control unit in accordance with the program stored in the memory. The program of a PLC can be created in various ways: via assembler type commands in ’statement list’, in higher-level, problem-oriented languages such as structured text or in the form of a flow chart such as represented by a sequential function chart. In Europe, the use of function block diagrams based on function charts with graphic symbols for logic gates is widely used. In America, the ‘ladder diagram’ is the preferred language by users. Depending on how the central control unit is connected to the input and output modules, differentiation can be made between compact PLCs (input module, central control unit and output module in one housing) or modular PLCs.
The PLC mainly consists of a CPU, memory areas, and appropriate circuits to receive input/output data. We can actually consider the PLC to be a box full of hundreds or thousands of separate relays, counters, timers and data storage locations. They don't physically exist but rather they are simulated and can be considered software counters, timers, etc. Each component of a PLC has a specific function.

- **Input Relays (contacts)** - These are connected to the outside world. They physically exist and receive signals from switches, sensors, etc. Typically they are not relays but rather they are transistors.
- **Internal Utility Relays** - These do not receive signals from the outside world nor do they physically exist. They are simulated relays and are what enables a PLC to eliminate external relays. There are also some special relays that are dedicated to performing only one task. Some are always on while some are always off. Some are on only once during power-on and are typically used for initializing data that was stored.
- **Counters** - These are simulated counters and they can be programmed to count pulses. Typically these counters can count up, down or both up and down. Since they are simulated they are limited in their counting speed. Some manufacturers also include high-speed counters that are hardware based. We can think of these as physically existing.
- **Timers** - These come in many varieties and increments. The most common type is an on-delay type. Others include off-delay and both retentive and non-retentive types. Increments vary from 1 millisecond through 1 second.
- **Output Relays (coils)** - These are connected to the outside world. They physically exist and send on/off signals to solenoids, lights, etc. They can be transistors, relays, or triacs depending upon the model chosen.
- **Data Storage** - Typically there are registers assigned to simply store data. They are usually used as temporary storage for math or data manipulation. They can also typically be used to store data when power is removed from the PLC. Upon power-up they will still have the same contents as before power was removed.

![Figure 2.2: Components of PLC](image-url)
Chapter 3. Comparison with Hard Wired Relay

3.0 A Brief overview about Hard wired Relay

At the outset of industrial revolution, especially during sixties and seventies, relays were used to operate automated machines, and these were interconnected using wires inside the control panel. In some cases a control panel covered an entire wall. To discover an error in the system much time was needed, especially with more complex process control systems. On top of everything, a lifetime of relay contacts was limited, so some relays had to be replaced. If replacement was required, machine had to be stopped and production as well. Also, it could happen that there was not enough room for necessary changes. A control panel was used only for one particular process, and it wasn’t easy to adapt to the requirements of a new system. As far as maintenance, electricians had to be very skillful in finding errors. In short, conventional control panels proved to be very inflexible. Typical example of conventional control panel is given in the following picture.

Figure 3.0 Typical Small Scale Control Panel

In the Figure 3.0, you can see a large number of electrical wires, relays, timers and other elements of automation typical for that period. The pictured control panel is not one of the more complicated ones, so you can imagine what complex ones looked like.

3.1 Disadvantages of a classic control panel

The most frequently mentioned disadvantages of a classic control panel are:

- Large amount of work required connecting wires
- Difficulty with changes or replacements
- Difficulty in finding errors; requiring skillful/experienced work force
- When a problem occurs, hold-up time is indefinite, usually long
With invention of programmable controllers, much has changed in how a process control system is designed. Many advantages appeared. Typical example of control panel with a PLC controller is given in the following picture.

Figure 3.1 Typical PLC Control Panel

3.2 Advantages of PLC Control Panel

Advantages of control panel that is based on a PLC controller can be presented in few basic points:

- Compared to a conventional process control system, number of wires needed for connections is reduced by approximately 80%
- Use of PLC results in appreciable savings in Hardware and wiring cost
- Diagnostic functions of a PLC controller allow for fast and easy error detection.
- Change in operating sequence or application of a PLC controller to a different operating process can easily be accomplished by replacing a program through a console or using PC software (not requiring changes in wiring, unless addition of some input or output device is required).
- Needs fewer spare parts.
- It is much cheaper compared to a conventional system, especially in cases where a large number of Input/Output instruments are needed and when operational functions are complex
- Reliability of a PLC is greater than that of an electro-mechanical relay or a timer, because of less moving parts.
- They are compact and occupy less space
- Use of PLC results in appreciable savings in Hardware and wiring cost
Chapter 4. Key concepts to understand PLC computing

4.0 Real Time Computing
In computer science, real-time computing (RTC), or reactive computing, is the study of hardware and software systems that are subject to a "real-time constraint"—i.e., operational deadlines from event to system response. Real-time programs must execute within strict constraints on response time. By contrast, a non-real-time system is one for which there is no deadline, even if fast response or high performance is desired or preferred. The needs of real-time software are often addressed in the context of real-time operating systems, and synchronous programming languages, which provide frameworks on which to build real-time application software.

A real time system may be one where its application can be considered (within context) to be mission critical. The anti-lock brakes on a car are a simple example of a real-time computing system — the real-time constraint in this system is the short time in which the brakes must be released to prevent the wheel from locking. Real-time computations can be said to have failed if they are not completed before their deadline, where their deadline is relative to an event. A real-time deadline must be met, regardless of system load.

4.1 The Importance of Time Based Control
In traditional sequential control systems where input sensors, output actuators, and industrial controllers are distributed over a local area network, the control algorithms are typically scan-based and asynchronous, and consequently suffer from significant processing jitter. Some systems employ change of state or event-triggered techniques to improve performance. However, time-based control provides the best performance alternative.

Figure 4.0 Jitter delays for Input-Process-Output Sequence
4.1.0 Scan-Based Control

For scan-based control, the process is as follows for a simple input, control, output sequence. Input data from sensor devices are sent to the controller at a periodic rate. The controller runs its control algorithm at a periodic rate and output results are sent to the output actuators at a periodic rate. The inputs and outputs change state asynchronously to the periodic input and output scan.

This input-process-output sequence creates a very elastic or jittery input to output delay. The delay jitter will be a function of when the input changes in relation to the asynchronous periodic scans of the input, controller and output transfer, network transport delays, and internal device delays. These delays are illustrated in

4.1.1 Event-Triggered Control

Event-triggered or change-of-state control can significantly reduce jitter. With change-of-state operation the input, control, and output, scan delays are eliminated. When an input transition is detected by the input device, it immediately sends it to the controller. The controller is interrupted when the input arrives and immediately executes its processing algorithm and sends the result to the output device. When the output message arrives at the output device it immediately actuates the output.

This approach will still incur jitter delays due to network transport. If a large number of input transitions occur at once, network congestion and packet loss may occur resulting in additional jitter delays and possibly machine failure. Also, since many I/O devices do not support event trigger mechanisms, this approach is often less viable. In practice, a traditional control system will use a combination of scan-based and event-based control mechanisms.

4.1.2. Time-Based Control

For many applications, the jitter won’t matter as long as the application response times are satisfied. However, some applications require more precision and have a low tolerance to jitter. For these applications, a time-based control system can solve these problems more effectively.

In a time-based system, an association is made between input and output events and time. Time becomes an integral function of the control system and control algorithms. All devices in the system have the same notion of time. In such a system, the input events are time-stamped and output events are scheduled. The control system precisely knows when the input was sampled and can precisely determine when the output should be actuated. The output device can schedule the output to actuate at a predetermined time.
The only jitter sources for this system are those associated with accurately time-stamping inputs and outputs.

<table>
<thead>
<tr>
<th>Jitter or Delay Source</th>
<th>Delay</th>
<th>Delta Jitter (Max - Min)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Scan-Based</td>
<td>Event-Triggered</td>
</tr>
<tr>
<td>Input</td>
<td>0.2 msec</td>
<td>10 msec</td>
</tr>
<tr>
<td>Input Network</td>
<td>1 msec</td>
<td>1 msec</td>
</tr>
<tr>
<td>Controller</td>
<td>10 msec</td>
<td>100 msec</td>
</tr>
<tr>
<td>Output Network</td>
<td>1 msec</td>
<td>1 msec</td>
</tr>
<tr>
<td>Output</td>
<td>0.2 msec</td>
<td>10 msec</td>
</tr>
<tr>
<td>Total</td>
<td>12.4 msec</td>
<td>122 msec</td>
</tr>
</tbody>
</table>

**Table 4.1** shows relative delta jitter delays for the three control mechanisms discussed. The delay numbers indicate the processing delays for the component. The delta jitter is the maximum minus the minimum jitter for the component.
Chapter 5. Ladder Logic

5.0 Introduction

Ladder logic is one form of drawing electrical logic schematics, and is a graphical language very popular for programming Programmable Logic Controllers. Ladder logic was originally invented to describe logic made from relays. The name is based on the observation that programs in this language resemble ladders, with two vertical "rails" and a series of horizontal "rungs" between them. Figure 5.0 below is a very basic example of ladder logic used in a programmable logic controls program.

![Basic Ladder Logic Program](image)

**Figure 5.0 Basic Ladder Logic Program**

5.1 Comparison to Relay Logic

The program used in a controls schematic, called a ladder diagram, is similar to a schematic for a set of relay circuits. An argument that aided the initial adoption of ladder logic was that a wide variety of engineers and technicians would be able to understand and use it without much additional training, because of the resemblance to familiar hardware systems. This argument has become less relevant lately given that most ladder logic programmers have a software background in more conventional programming languages, and in practice implementations of ladder logic have characteristics such as sequential execution that make the analogy to hardware somewhat imperfect. Electricians and data cabling or control technicians still argue that this is the best graphical interface as they generally do not have any computer science or digital systems background, and are therefore taught with this interface in sequence with relay logic.
Relay logic is the precursor to ladder logic, and is a method of controlling industrial electronic circuits by using relays and contacts. Figure 5.1 shows an average mechanical relay used in older relay logic systems. The schematic diagrams for relay logic circuits are often called line diagrams, because the inputs and outputs are essentially drawn in a series of lines, with the lines representing actual wires run in the circuit. A relay logic circuit is an electrical network consisting of lines, in which each input/output group must have electrical continuity with all components in that group of devices to enable the output device. The Relay logic diagrams represent the physical interconnection of devices, while the relay logic circuit forms an electrical schematic diagram for the control of input and output devices. This is why electricians and control technicians can easily understand and interpret relay logic and ladder logic diagrams. Figure 5.2 below shows a basic relay logic circuit. Notice how it differs from the ladder logic circuit in Figure 5.0 in that the “virtual” inputs and outputs in the ladder logic circuit have replaced the actual relays and coils in the relay logic circuit.

Figure 5.2 is a small, basic relay logic circuit. You can see how in relay logic circuits the pushbuttons are represented with graphical drawings of a normally closed pushbutton for the stop button, and a normally open pushbutton for the start button. The coil that is marked “M” is a motor coil, and is a physical piece of equipment in the same location as the motor, which is represented by a circle with the letter M in the middle. The over current or overload device is represented by a normally closed coil symbol with “O.L.” over it. There would only be seven wires to connect in this circuit, so this would not be very difficult to wire, but when more inputs and outputs are added, the difficulty grows exponentially. Figure 5.3 shows an expanded relay circuit of Figure 5.2 in that a double pole single throw pushbutton is added into the diagram to be used as a “jog function”. As the diagram shows, a jog switch is used to run the output (motor). Only one component is added, but three wires need to be installed in the circuit for the component to be utilized in the intended manner.
Figure 5.3 Relay Logic Circuit with Jog function added

Figure 5.4 below adds four more components to the system. Two of them are just coils from the motor apparatus that are used as inputs and the other two are a red and green light to be utilized as output/motor status indicators for the user.

Figure 5.4 Relay Logic Circuit with Jog function and Status Indicators

This circuit adds 6 additional wires to the original circuit in Figure 5.3. If both of the additions from Figure 5.3 and 5.4 were added to the original circuit, this would add 5 components and 9 additional wires. This illustrates how using a programmable logic controller is advantageous in that adding any number of relays takes much less effort. It doesn’t seem like a large amount of work to connect just 9 additional wires, but in a real world situation, the motor in question may be on top of a grain silo, and the start/stop station may be a few hundred feet away in a control booth. Pulling all these control wires would take hours instead of a few minutes sitting in front of a programming terminal. Programmable logic controllers coupled with ladder logic can make some of the most labor intensive tasks become easy, enjoyable projects.
Figure 5.5 Complex Ladder Diagram

Ladder logic is the most widely used program for programmable logic controllers where sequential control of a process or manufacturing operation is required. Ladder logic is useful for simple but critical control systems, or for reworking old hardwired relay circuits. As programmable logic controllers became more sophisticated it has also been used in very complex automation systems. Figure 5.5 above shows a much more complicated ladder logic diagram than the one shown in Figure 5.0. It is relatable to the relay circuits in Figures 5.2, 5.3, and 5.4 as well in that some of the outputs are motors and status lights. In addition there are holding/latching contacts included, but they are not a piece of hardware. In fact, they are just the address of the respective output being referenced, which will be discussed in greater detail later. This is still not a very large program. Ladder logic programs can easily grow to more than 500 “rungs” to finish some functions.
Chapter 6. Ladder Logic Programming

6.0 Basics of Ladder Logic

Ladder logic or ladder diagrams are the most common programming language used to program a PLC. Ladder logic was one of the first programming approaches used in PLCs because it borrowed heavily from the relay diagrams that plant electricians already knew. The symbols used in relay ladder logic consist of a power rail to the left, a second power rail to the right, and individual circuits that connect the left power rail to the right. The logic of each circuit (or rung) is solved from left to right. A common mistake made by most people is trying to think of the diagram as having to have current across the rung for the output to function. This has given many people trouble because of the fact that some inputs are “not” inputs, which will be true when there isn’t current through this sensor. The symbols of these diagrams look like a ladder - with two side rails and circuits that resemble rungs on a ladder.

\[
\text{Input} \quad \text{Output}
\]

**Figure 6.0 Simplified Logic Circuit**

The logic of the rung above is such:
- If Input1 is ON (or true) - power (logic) completes the circuit from the left rail to the right rail - and Output1 turns ON (or true).
- If Input1 is OFF (or false) - then the circuit is not completed and logic does not flow to the right - and Output 1 is OFF (or false).

6.1 Basic Symbols and Notations

There are many logic symbols available in Ladder Logic - including timers, counters, math, and data moves such that any logical condition or control loop can be represented in ladder logic. With just a handful of basic symbols such as a normally open contact, normally closed contact, normally open coil, normally closed coil, timer and counter most logical conditions can be represented.

6.1.1 Normally Open Contact

This can be used to represent any input to the control logic such as a switch or sensor, a contact from an output, or an internal output. When solved the referenced input is examined for a true (logical 1) condition. If it is true, the contact will close and allow logic to flow from left to right. If the status is FALSE (logical 0), the contact is open and logic will not flow from left to right.
6.1.2 Normally Open Coil

This can be used to represent any discrete output from the control logic. When "solved" if the logic to the left of the coil is TRUE, the referenced output is TRUE (logical 1).

6.1.3 Normally Closed Contact

When solved the referenced input is examined for an OFF condition. If the status is OFF (logical 0) power (logic) will flow from left to right. If the status is ON, power will not flow.

6.1.4 Normally Closed Coil

When "solved" if the coil is a logical 0, power will be turned on to the device. If the device is logical 1, power will be OFF.

6.1.5 Basic AND & OR Gates

The AND is a basic fundamental logic condition that is easy to directly represent in Ladder Logic. Figure 6.1 shows a simplified AND “gate” on a ladder rung.

Figure 6.1 Simplified AND gate

In order for Light1 to turn TRUE, Switch1 must be TRUE, AND Switch2 must be TRUE. If Switch1 is FALSE, logic (not power) flows from the left rail, but stops at Switch1. Light1 will be TRUE regardless of the state of Switch2. If Switch1 is TRUE, logic makes it to Switch2. If Switch2 is TRUE, power cannot flow any further to the right, and Light1 is FALSE. If Switch1 is TRUE, AND Switch2 is TRUE - logic flows to Light1 solving its state to TRUE.

The OR is a logical condition that is easy to represent in Ladder Logic. Figure 6.2 shows a simple OR gate. Notice the differences in logic between the OR and AND gates.

Figure 6.2 Simplified OR gate

If Switch1 is TRUE, logic flows to Light1 turning it to TRUE. If Switch2 is TRUE, logic flows through the Switch2 contact, and up the rail to Light1 turning it to TRUE. If Switch1 AND Switch 2 are TRUE Light1 is TRUE. The only way Light1 is FALSE is if
Switch1 AND Switch2 are FALSE. In other words, Light1 is TRUE if Switch1 OR Switch2 is TRUE.

6.2 Basic Timers & Counters
Many times programs will call for action to be taken in a control program based on more than the states of discrete inputs and outputs. Sometimes, processes will need to turn on after a delay, or count the number of times a switch is hit. To do these simple tasks, Timers & Counters are utilized.

Figure 6.3 On-Delay Timer (RTO)

A timer is simply a control block that takes an input and changes an output based on time. There are two basic types of timers. There are other advanced timers, but they won’t be discussed in this report. An On-Delay Timer takes an input, waits a specific amount of time, and allows logic to flow after the delay. An Off-Delay Timer allows logic to flow to an output and keeps that output true until the set amount of time has passed, then turns it false, hence off-delay. Figure 6.3 above shows an On-Delay Timer with a 10 second delay before it passes the logic through it.

Figure 6.4 Up Counter

A counter simply counts the number of events that occur on an input. There are two basic types of counters called up counters and down counters. As its name implies, whenever a triggering event occurs, an up counter increments the counter, while a down counter decrements the counter whenever a triggering event occurs. Figure 6.4 shows the typical graphical representation of an Up Counter.
Chapter 7. Building a PLC/Ladder Logic Program

7.0 Ladder Logic Programming

Building a small ladder logic program to run on a PLC network is quite easy. For the beginner, it is easier to see the ladder diagram in the form of relay logic. Figure 7.0 below shows a basic start/stop station for a motor in relay logic.

Figure 7.0 Ladder Diagram in Relay Logic

Just as in Figure 7.0 above, relay logic shows all components in the system. This is because relay logic is the same as the wiring diagrams that the electricians use, so all the wiring needs to be shown for the logic to work. Because of this, some components may not need to be included in the plc ladder logic diagram.

Figure 7.1 Relay Diagram with overload removed

Figure 7.1 above shows the same circuit as in Figure 7.0 with the overload removed. The overload is needed in relay logic because you have to have an overload device on any circuit; therefore it needs to be in the wiring diagram. This way, if you push too much current to the motor, the overload device will interrupt the circuit. Overloads are included internally in most any device anymore, but you will still see this in diagrams. There is still an overload device in a plc ladder logic circuit, but ladder logic shows only those components that have an input or output address, so you do not see it. In Figure 7.1, you can see that the start and stop buttons along with the motor relay will all be turned to inputs in the plc diagram and the motor, signified by a circle with an ‘M’ in the middle will be an output. The motor relay will not be a physical entity in the plc ladder diagram as it is in this relay logic. It will simply be an input that uses the same I/O address as the motor output. The stop button input can be located on either side of the start button/relay gate, as long as it is still in series with it.
Figure 7.2 Relay Circuit with Addition of Jog Function

Figure 7.2 above shows the addition of a ‘Jog Function’ to the relay circuit. The jog function is generally added to any circuit for troubleshooting purposes only. Most jog functions are set up so that the only time the motor will run with the help of the jog function is when the ‘Jog Button’ is pushed. In Figure 7.2 above, you can see this with the relay logic. As the circuit looks right now, when the Start Button is pressed, the motor will start, energizing the relay, and going across the Jog Button’s normally closed contacts. The motor will stay running this way until the Stop Button is pressed. If instead the Jog Button is pressed, the current will travel across the normally open Jog contacts that are now closed. The motor will stay running until the Jog Button is no longer pressed.

Figure 7.3 Relay Circuit with Addition of Status Indicators

Figure 7.3 above shows that same circuit with Status Indicators added. These are used in control rooms to inform users of the status of their motors or other moving parts. Green is the generally accepted color for a motor going, while red is stopped. The green light is energized when the normally open contact is energized by the moving motor, closing it. The red light is energized whenever a normally closed relay is closed, so it will turn off whenever the motor starts to run. From Figure 7.0 to Figure 7.3, one can see that with every component added, many wires need to be connected as well. Depending on how far away these components are away from each other, this can be very difficult and time consuming.
Figure 7.4 Relay Logic Diagram Converted to PLC Ladder Logic

Figure 7.4 above was converted from the relay logic in Figure 7.3 to the PLC ladder logic seen here. If the PLC logic here was used in Figures 7.0-Figures 7.3, adding the various component would’ve taken much less time than physically wiring each component. PLC ladder logic can differ from relay logic in that different components are used as well. In the relay diagrams, a single button double pole switch was used so that it could perform two different functions. In PLC ladder logic, just a single pole button is needed, because the computer can be asked to look for a on or off state. For the status lights, instead of running wires to the motor relays the PLC diagram just looks for a true or false state of the motor output.
Chapter 8. Input and Output Components

8.0 Human Machine Interface

In the industrial design field of human-machine interaction, the user interface is (a place) where interaction between humans and machines occurs. The goal of interaction between a human and a machine at the user interface is effective operation and control of the machine, and feedback from the machine which aids the operator in making operational decisions. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology.

A user interface is the system by which people (users) interact with a machine. The user interface includes hardware (physical) and software (logical) components. User interfaces exist for various systems, and provide a means of:

- Input, allowing the users to manipulate a system, and/or
- Output, allowing the system to indicate the effects of the users' manipulation.

Generally, the goal of human-machine interaction engineering is to produce a user interface which makes it easy, efficient, and enjoyable to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human.

Ever since the increased use of personal computers and the relative decline in societal awareness of heavy machinery, the term user interface has taken on overtones of the (graphical) user interface, while industrial control panel and machinery control design discussions more commonly refer to human-machine interfaces.

![Human Machine Interface (HMI)](image)

Figure 8.0 Human Machine Interface (HMI)
8.1 Use of Optocoupler

In electronics, an opto-isolator, also called an optocoupler, photocoupler, or optical isolator, is "an electronic device designed to transfer electrical signals by utilizing light waves to provide coupling with electrical isolation between its input and output". The main purpose of an opto-isolator is "to prevent high voltages or rapidly changing voltages on one side of the circuit from damaging components or distorting transmissions on the other side." Commercially available opto-isolators withstand input-to-output voltages up to 10 kV and voltage transients with speeds up to 10 kV/μs.

![Figure 8.1 Schematic diagram of an opto-isolator showing source of light (LED) on the left, dielectric barrier in the center, and sensor (phototransistor) on the right](image)

An opto-isolator contains a source (emitter) of light, almost always a near infrared light-emitting diode (LED), that converts electrical input signal into light, a closed optical channel (also called dielectrical channel), and a photosensor, which detects incoming light and either generates electric energy directly, or modulates electric current flowing from an external power supply. The sensor can be a photoresistor, a photodiode, a phototransistor, a silicon-controlled rectifier (SCR) or a triac. Because LEDs can sense light in addition to emitting it, construction of symmetrical, bidirectional opto-isolators is possible. An optocoupled solid state relay contains a photodiode opto-isolator which drives a power switch, usually a complementary pair of MOSFET transistors. A slotted optical switch contains a source of light and a sensor, but its optical channel is open, allowing modulation of light by external objects obstructing the path of light or reflecting light into the sensor.
References

http://en.wikipedia.org/wiki/Automation

http://en.wikipedia.org/wiki/Programmable_logic_controller


http://www.controldesign.com/

http://www.festo-didactic.com/int-en/

http://www.webstergriffin.com/