NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Electric and Electronic Engineering

ELECTROMAGNETIC CONVEYOR MACHINE USING BY PLC CONTROLLER

Graduation Project
EE 400

Students: TUNÇ SAMURKAŞ (990504)

Supervisor: ÖZGÜR C. ÖZERDEM

Lefkoşa - 2004

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor Mr. Özgür Özerdem., I successfully overcome many difficulties and learn a lot about Electronic and Programmable Logic Controller(PLC) and Graduation Project. When I faced problems in these fields, he always helped me.

I would like to thank My friends Melek Yanargöz, Cem U ludağ, Hakan Tuna ,Turgut Tuna, Bahadır Kara, Bahadır Öner, Ali Kurtekin, Nedim Perihanoğlu, Bora Karaçam, Bülent Kaptan, Iklim Çerkezoğlu, Birgül Kaya, Aydın Yılmaz, Emin Küçük, Umut Çağlar Çamoğlu, Okan Şen I wish to thank my parents because of their en dless support and encouragement

Also, I would like to thank my relatives that helps me even at my smallest problems. I feel lucky because I have these perfect people.

Finally, special thanks for my family, especially my parents for being patientful during my undergraduate degree study. I could never have completed my study without their

ABSTRACT

PLC technology has advanced, so have programming languages and communications capabilities, along with many other important features. Today's PLCs offer faster scan times, space efficient high-density input/output systems, and special interfaces to allow non -traditional devices to be attached directly to the PLC. Not only can they communicate with other control systems, they can also perform reporting functions and diagnose their own failures, as well as the failure of a machine or process. Size is typically used to categorize today's PLC, and is often an indication of the features and types of applications it will accommodate.

In this project FATEK FALCON FB SERIES PLC have 16 inputs, 12 outputs, 24 volt DC input 220 Volt AC output PLC is used, and an automation of a gate and a door are realized by PLC programs.

INTRODUCTION

A Programmable Logic Controller was defined by Capiel (1982) as: " A digitally operating electronic system designed for use in an industrial environment, which uses a programmable memory for the internal storage of instructions for implementing specific functions such as logic, sequencing, timing, counting and arithmetic to control through analog or digital input/output modules, various types of machines or processes." Which explains the device perfectly.

In the late 1960's PLC's were first introduced. The first PLC can be traced back to 1968 when Bedford Associates, a company in Bedford, MA, developed a device called a Modular Digital Controller for General Motors (GM). The MODICON, as it was known, was developed to help GM eliminate traditional relay-based machine control systems.

List of Figures

List of Tables

TABLE OF CONTENTS

CHAPTER 1 PROGRAMMABLE LOGIC CONTROLLERS

1.1 What is a PLC?

A programmable logic controller (PLC) is a device that was invented to replace the necessary sequential relay circuits for machine control. The PLC works by looking at its inputs and depending upon their state, turning on / off its outputs. The user enters a program, usually via software, that gives the desired results.

PLC are often defined as miniature industrial computers that contain hardware and software that is used to perform control functions. A PLC consists of two basic sections: the central processing unit (CPU) and the input/output interface system. The CPU, which controls all PLC activity, can further be broken down into the processor and memory system. The input/output system is physically connected to field devices (e.g., switches, sensors, etc.) and provides the interface between the CPU and the information providers (inputs) and controllable devices (outputs). To operate, the CPU "reads" input data from connected field devices through the use of its input interfaces, and then "executes", or performs the control program that has been stored in its memory system. Programs are typically created in ladder logic, a language that closely resembles a relay-based wiring schematic, and are entered into the CPU's memory prior to operation. Finally, based on the program, the PLC "writes", or updates output devices via the output interfaces. This process, also known as scanning, continues in the same sequence without interruption, and changes only when a change is made to the control program.

PLC's are used in many real world applications. If there is industry present, changes are good that there is a PLC present. If you are involved in machining, packaging material handling, and automated assembly or countless other industries you are probably already using them. If you are not, you are wasting money and time. Almost any application that needs some type of electrical control has a need for PLC.Let's assume that when a switch turned on, a solenoid is wanted to turn on for 5 seconds and then turn it off regularly of how long the switch is on. It can be done with a simple external timer.

If the process becomes more complicated, then we have to use a device the simplify that. We use PLC for this process. We can program the PLC to count its inputs and turn the solenoids for the specified time.

1.2 History of PLC

The primary reason for designing such a device was eliminating the large cost involved in replacing the complicated relay based machine control systems. Bedford Associates (Bedford, MA) proposed something called a Modular Digital Controller (MODICON) to a major US car manufacturer. Other companies at the time proposed computer based schemes, one of which was based upon the PDP-8. The MODICON 084 brought the world's first PLC into commercial production.

Since relays are mechanical devices they also have a limited lifetime which required strict adhesion to maintenance schedules. Troubleshooting was also quite tedious when so many relays are involved. Now picture a machine control panel that included many, possibly hundreds or thousands, of individual relays. The size could be mind boggling. How about the complicated initial wiring of so many individual devices! These relays would be individually wired together in a manner that would yield the desired outcome.

In the mid70's the PLC technologies were sequencer state-machines and the bit-slice based CPU. The AMD 2901 and 2903 were quite popular in Modicon and A-B PLCs. Conventional microprocessors lacked the power to quickly solve PLC logic in all but the smallest PLCs. As conventional microprocessors evolved, larger and larger PLCs were being based upon them. However, even today some are still based upon the 2903 (ref A-B's PLC-3) Modicon has yet to build a faster PLC than their 984A/B/X which was based upon the 2901.

Communications abilities began to appear in approximately 1973. The first such system was Modicon's Modbus. The PLC could now talk to other PLCs and they could be far away from the actual machine they were controlling. They could also now be used to send and receive varying voltages to allow them to enter the analog world. Unfortunately, the lack of standardization coupled with continually changing technology has made PLC communications a nightmare of incompatible protocols and physical networks.

It was also a time for reducing the size of the PLC and making them software programmable through symbolic programming on personal computers instead of dedicated programming terminals or handheld programmers. Today the world's smallest PLC is about the size of a single control relay!

The 90's have seen a gradual reduction in the introduction of new protocols, and the modernization of the physical layers of some of the more popular protocols that survived the 1980's. The latest standard (IEC 1131-3) has tried to merge plc programming languages under one international standard. We now have PLCs that are programmable in function block diagrams, instruction lists, C and structured text all at the same time! PC's are also being used to replace PLCs in some applications. The original company who commissioned the MODICON 084 has actually switched to a PC based control system.

1.3 General Physical Build Mechanism

1.3.1 Compact PLC'S

Compact PLC's are manufactured such that all units forming the PLC are placed in a case. They are low price PLC with lower capacity. Small or medium size machine manufacturers usually prefer them. In some types compact enlargement module is present.

1.3.2 Modular PLC's

Combining separate modules together in a board forms them. They can have different memory capacity, I / O numbers, power supply up to the necessary limits.

Some examples: SIEMENS S5-115U, SIEMENS S7-200, MITSUBISHI PC40, TEXAS INSTRUMENT PLC'S, KLOCKNER-MOELLER PS316, OMRON C200H.

1.4 Programmable Controller

1.4.1 Overview

Originally designed as a replacement for the hard-wired relay and timer logic to be found in traditional control panels, PLC's provides ease and flexibility of control based on programming and executing simple logic instructions. PLC's have internal functions such as timers, counters and shift registers, making sophisticated control possible using even the smallest PLC.

A programmable control operates by examining the input signals from a process and carrying out logic instructions on these input signals, producing output signal to drive process equipment or machinery. Standard interfaces build into PLC's allow them to be directly connected to process actuators and transducers without the need for intermediate circuitry or relays.

Through using PLC's it became possible to modify a control system without having the disconnect or re-route a signal wire. It was necessary to change only the control program using a keypad or VDU terminal. Programmable controllers also require shorter installation and commissioning times than do hardwired systems. Although PLC's are similar to conventional computers in terms of hardware technology, they have specific features suited to industrial control:

- Rugged, noise immune equipment;
- Modular plug-in construction, allowing easy replacement\addition of units (input\output);
- Standard input\output connections and signal levels;
- Easily understood programming language (ladder diagram and function chart),
- Ease of programming and reprogramming in-plant.

1.4.2 Background

The programmable controller was initially conceived by a group of engineers from General Motors in 1968, where an initial specification was provided: the controller must be:

1- Easily programmed and reprogrammed, preferably in-plant to alter its sequence of operations.

2- Easily maintained and repaired- preferably using plug-in modules.

a) More reliable in plant environment.

b) Smaller than it is relay equivalent.

3- Cost competitive, with solid-state and relay panels than in use.

The instruction sets quickly moved from simple logic instructions to include counters, timers and shift registers, than onto more advanced mathematical functions on the machines. Developments hardware were also occurring, with larger memory and greater numbers of input / output points featuring on new models. In 1976 became possible to control remote I / O racks, where large numbers of distant I / O points were monitored updated via a communications link, often several hundred meters from the main PLC. The Allan-Bradley Corporation in America introduced a microprocessor-based PLC in 1977. It was based on an 8080 microprocessor but used an extra processor to handle bit logic instruction at high speed.

The increased rate of application of programmable controllers within industry has encouraged manufacturers to develop whole families of microprocessor-based systems having various levels of performance. The range of available PLC's now extends from small self-contained units with 20 digital I / O points and 500 program steps, up to modular systems with add-on function modules:

- Analogue I/O;
- PID control (proportional, integral and derivative terms);
- Communications;
- Graphics display;
- Additional I/O;
- Additional memory.

This modular approach allows the expansion or upgrading of a control system with minimum cost and disturbance.

Programmable controllers are developing at a virtually the same pace as microcomputers, with particular emphasis on small controllers, positioning\numeric control and communication networks. The market for small controllers has grown rapidly since the early 1980's when a number of Japanese companies introduced very small, low cost units that were much cheaper than others available at that time. This brought programmable controllers within the budget of many potential users in the manufacturing and process industries, and this trend continues with PLC's offering ever-increasing performance at ever-decreasing cost.

1.4.3 Terminology-PC or PLC

- PC programmable controller
- PLC programmable logic controller
- PBS programmable binary system

The smallest programmable controllers can now be equipped to process analogue inputs and outputs these labels are not representative of their capabilities. For these reason the overall term programmable controller has been widely adopted to describe the family of freely programmable controllers. However, to avoid confusion with the personal computer PC, this text uses the abbreviation PLC for programmable controller.

1.5 Types of PLC

The increasing demand from industry for programmable controllers that can be applied to different forms and sizes of control tasks has resulted in most manufacturers producing a range of PLC's with various levels of performance and facilities.
However, to evaluate properly any programmable controller we must consider many additional features such as its processor, cycle time language facilities, functions, and expansion capabilities.

A brief outline of the characteristics of small, medium of large programmable controller is given below, together with typical applications.

1.5.1 Small PLC's

Small and mini PLC's are designed as robust, compact units, which can be mounted on or beside the equipment to be controlled. They are mainly used the replaced hard-wired logic relays, timers, counters. That control individual items of plant or machinery, but can also be used to coordinate several machines working in conjunction with each other.

Small programmable controllers can normally have their total I/ O expanded by adding one or two I/ O modules, but if any further developments are required this will often mean replacement of the complete unit. This end of the market is very much concerned with non-specialist and users, therefore ease of programming and a 'familiar' circuit format are desirable. Competition between manufacturers is extremely fierce in this field, as they vie to obtain a maximum share in this partially developed sector of the market.

A single processor is normally used, and programming facilities are kept a fairly basic level, including conventional sequencing controls and simple standard functions: e.g. timers and counters. Programming of small PLC's is by way of logic instruction list or relay ladder diagrams.

Program storage is given by EPROM or battery-backed RAM. There is now a trend towards EEPROM memory with on-board programming facilities on several controllers.

1.5.2 Medium-sized PLC's

This construction allows the simple upgrading or expansion of the system. This construction allows the simple upgrading or expansion of the system by fitting additional I/ O cards in to the cards into the rack, since most rack, systems have space for several extra function cards. Boards are usually 'rugged zed' to allow reliable operation over a range of environments.

Communications of a single and multi-bit processor are likely within the CPU. For programming, standard instructions or ladder and logic diagrams are available. Programming is normally carried out via a small keypad or a VDU terminal. If different sizes of PLC are purchased from a single manufacturer, it is likely that programs and programming panels will be compatible between the machines.

In general this type of PLC is applied to logic control tasks that can not be met by small controllers due to insufficient I\O provision, or because the control task is likely to be extended in the future. This might require the replacement of a small PLC, where as a modular system can be expanded to a much greater extent, allowing for growth. A medium-sized PLC may therefore be financially more attractive in the long term.

1.5.3 Large PLC

Where control of very large numbers of input and output points is necessary and complex control functions are required, a large programmable controller is the obvious choice. Large PLC's are designed for use in large plants or on large machines requiring continuous control. They are also employed as supervisory controllers to monitor and control several other PLC's or intelligent machines. e.g. CNC tools.

Modular construction in Euro card format is standard, with a wide range of function cards available including analogue input output modules. There is a move towards 16-bit processor, and also multi-processor usage in order to efficiently handle a large range of differing control tasks.

This multi-processor solution optimizes the performance of the overall system as regards versatility and processing speed, allowing to PLC to handle very large programs of 100 K instructions or more. Memory cards can now provide several megabytes of CMOS RAM or EPROM storage.

1.5.4. Remote input\output

Large numbers of input / output points are located a considerable distance away from the programmable controller, it is uneconomic to run connecting cables to every point. A solution to this problem is to site a remote I/ O unit near to the desired I/ O points. This acts as a concentrator to monitor all inputs and transmit their status over a single serial communications link to the programmable controller. Once output signals have been produced by the PLC they are feedback along the communications cable to the remote I/ O unit, which converts the serial data into the individual output signals to drive the process.

1.5.5 Programming large PLC's

Any function can be programmed, using the familiar ladder symbols via a graphics terminal or personal computer. Parameters are passed to relevant modules either by incorporating constants in to the ladder, or via on screen menus for that module.

There is progress towards standardization of programming languages; with programs becoming easier to over-view through improvement of text handling, hand improved documentation facilities. This is assisted by the application of personal computers as workstations.

1.5.6. Developments

Information between PC's and PLC sand other devices within a automated plants has resulted in the provision of a communications capability on all but the smallest controller. The development of local area networks and in particular the recent MAP specification by General Motors provides the communication link to integrate all levels of control systems.

1.6 Today's PLC

Today's PLCs offer faster scan times, space efficient high-density input/output systems, and special interfaces to allow non-traditional devices to be attached directly to the PLC. Not only can they communicate with other control systems, they can also perform reporting functions and diagnose their own failures, as well as the failure of a machine or process. Size is typically used to categorize today's PLC, and is often an indication of the features and types of applications it will accommodate. Small, non-modular PLCs (also known as fixed I/O PLCs) generally have less memory and accommodate a small number of inputs and outputs in fixed configurations. Modular PLCs have bases or racks that allow installation of multiple I/O modules, and will accommodate more complex applications. When you consider all of the advances PLCs have made and all the benefits they offer, it's easy to see how they've become a standard in the industry, and why they will most likely continue their success in the future.

CHAPTER 2 STRUCTURE AND OPERATION OF PLC

2.1 Basic functional structure of a programmable controller system

The PLC can be considered as 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. These internal relays are simulated through bit locations in registers.

The main functional components in a programmable controller system is illustrated below. These functions communicate with each other and with the signals of the machine/process to be controlled.

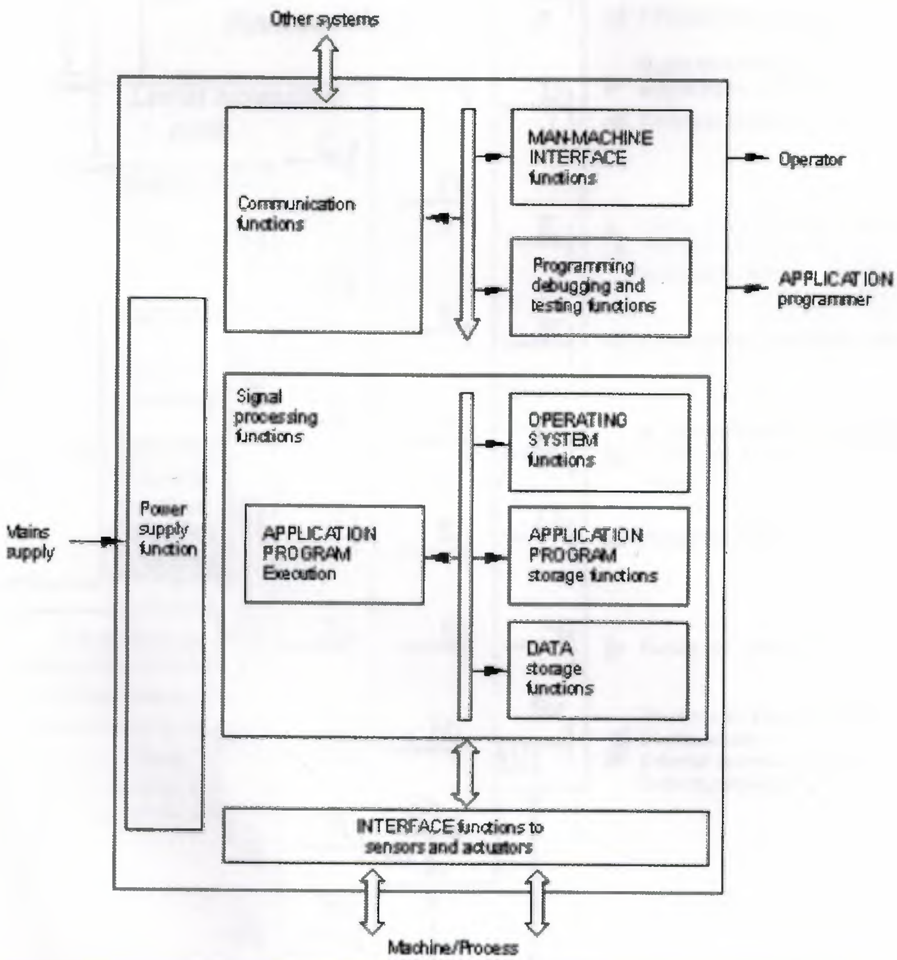


Figure 2.1.1 : Basic functional structure of a PLC-system

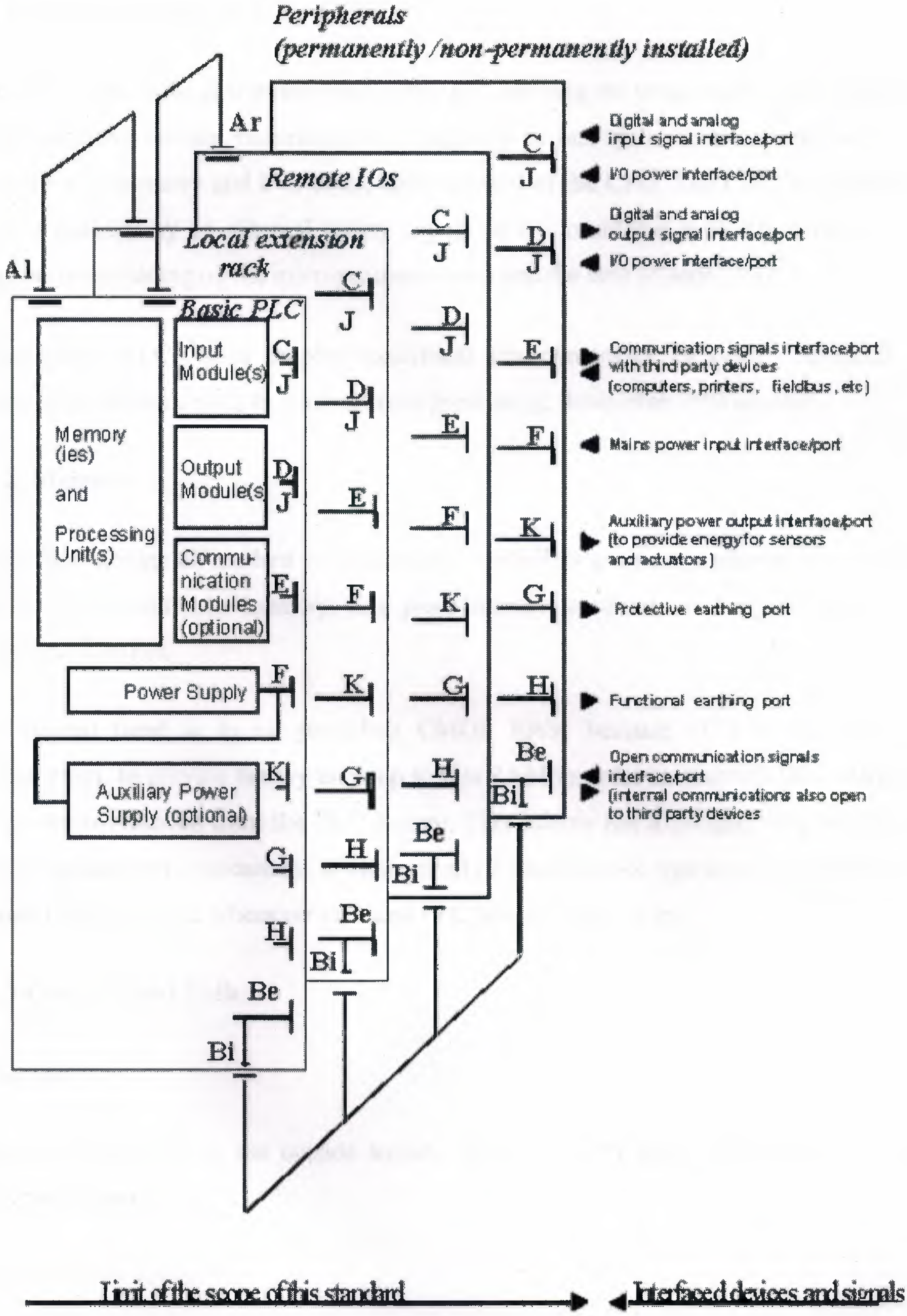


Figure 2.1.2: Typical Interface/Port Diagram of a PLC-system (from IEC 61131 - part)

2.1.1 Central Processing Unit (CPU)

The CPU controls all operations within the PLC, carrying out programmed instructions stored in memory. An internal communications highway or bus system carries information to and from the CP, memory and I/ O units, under control of the CPU. The CPU is supplied with a clock frequency by an external quartz crystal or RC oscillator, typically between 1 and 8 megahertz depending on the microprocessor used and the area of application.

Some larger PLC's also employ additional microprocessor to control complex, time-consuming functions such as mathematical processing, three terms PID control.

2.1.2 Memory

In the PLC storage all modern programmable controllers use semiconductor memory devices such as RAM read\write memory, or a programmable read-only memory of the EPROM or EEPROM families.

The current trend is to be providing CMOS RAM because of it is very low power consumption, to provide battery back up to this RAM in order to maintain the contents when the power is removed from the PLC system. This battery has a lifespan of at least one year before replacement is necessary, or alternatively a rechargeable type may be supplied with the system being recharge whenever the main PLC power supply is on.

2.1.3 Contacts and Coils

Input Relays:

These are connected to the outside world. They physically exist and receive signals from switches, sensors, etc.

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.

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.

2.1.4 Data storage

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.

2.1.5 Functions

**Communication function:*

**Human-machine interface (HMI) function*:

**Programming, debugging, testing and documentation functions*:

**Power supply functions*:

2.2 PLC Operation

The PLC's works by continually scanning a program. This scan cycle can be thought as consisting of 3 important steps. There are typically more than 3 but the important parts is focused on. Typically the others are checking the system and updating the current internal counter and timer values.

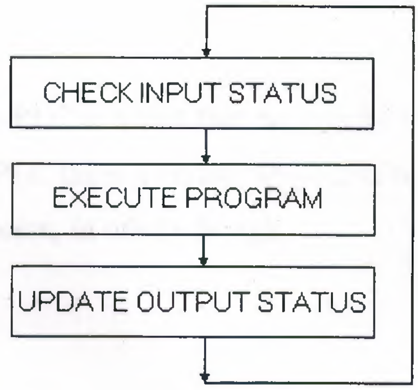


Figure 2.2.1 : Operation Flow of PLC

1-CHECK INPUT STATUS

First the PLC takes a look at each input to determine if it is on or off. In other words, is the sensor connected to the first input on.. It records this data into its memory to be used during the next step.

2-EXECUTE PROGRAM

The PLC executes your program one instruction at a time. Maybe the program said that if the first input was on then it should turn on the first output. Since it already knows which inputs are on/off from the previous step it will be able to decide whether the first output should be turned on based on the state of the first input. It will store the execution results for use later during the next step.

3-UPDATE OUTPUT STATUS

Finally the PLC updates the status of the outputs. It updates the outputs based on which inputs were on during the first step and the results of executing your program during the second step. Based on the example in step 2 it would now turn on the first output because the first input was on and program said to turn on the first output when this condition is true.

After the third step the PLC goes back to step one and repeats the steps continuously. One scan time is defined as the time it takes to execute the 3 steps listed above.

2.3 Response Time

The total response time of the PLC is a fact that have to be considered when shopping for a PLC. Just like our brains, the PLC takes a certain amount of time to react to changes. In many applications speed is not a concern, in others though...

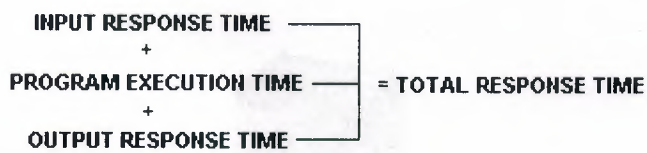


Figure 2.3.1 : Total response time

2.4 Relays

A relay can thought as an electromagnetic switch. Apply a voltage to the coil and a magnetic field is generated. This magnetic field sucks the contacts of the relay in, causing them to make a connection. These contacts can be considered to be a switch. They allow current to flow between 2 points thereby closing the circuit.

There is an example below.Bell is simly turned on whenever a switch is closed.There are 3 real-world parts. A switch, a relay and a bell. Whenever the switch closes we apply a current to a bell causing it to sound.

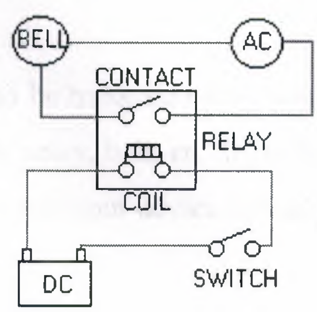


Figure 2.4.1 : Operation of relay

Notice in the figure 2.11 that there are 2 separate circuits. The bottom indicates the DC part. The top indicates the AC part.

Here dc relay is used to control an AC circuit.When the switch is open no current can flow through the coil of the relay. As soon as the switch is closed, however, current runs through the coil causing a magnetic field to build up. This magnetic field causes the contacts of the relay to close. Now AC current flows through the bell and it sounds.

Figure 2.4.2 : A typical industrial relay

2.4.1 Replacing Relays

Let's use a plc in place of the relay. The first thing that's necessary is to create what's called a ladder diagram. After seeing a few of these it will become obvious why its called a ladder diagram. One of these has to be created because, unfortunately, a plc doesn't understand a schematic diagram. It only recognizes code. Fortunately most PLCs have software which convert ladder diagrams into code. This shields users from actually learning the plc's code.

First step:

All of the items we're using have to be translated into symbols the plc understands. The plc doesn't understand terms like switch, relay, bell, etc. It prefers input, output, coil, contact, etc. It doesn't care what the actual input or output device actually is. It only cares that its an input or an output.

First the battery is replaced with a symbol. This symbol is common to all ladder diagrams. Bus bars are drawn. These simply look like two vertical bars. One on each side of the diagram. Think of the left one as being + voltage and the right one as being ground. Further think of the current (logic) flow as being from left to right. Next a symbol is given to the input. In this basic example we there is one real world input. (i.e. the switch) The symbol is given to the input that the switch will be connected to. This symbol can also be used as the contact of a relay.

Figure 2.4.1.1 : A contact symbol

Next a symbol is given to the outputs. In this example one output (i.e. the bell) is used the symbol is given to the output that the bell will be physically connected to. This symbol is used *as the coil of a relay.*

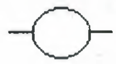


Figure 2.4.1.2 : A coil symbol

The AC supply is an external supply so we don't put it in our ladder. The plc only cares about which output it turns on and not what's physically connected to it.

2.5 Logic Instructions and Graphic Programming

Logic instructions are the basic programming language for programmable controllers. Although logic instructions are relatively easy to learn and use, it can be extremely time-consuming to check and relate large coded program to the actual circuit function. In addition, logic instructions tend to vary between different types of PLC. If a factory or plant is equipped with a range of different controllers (a common situation), confusion can result over differences in the instruction sets.

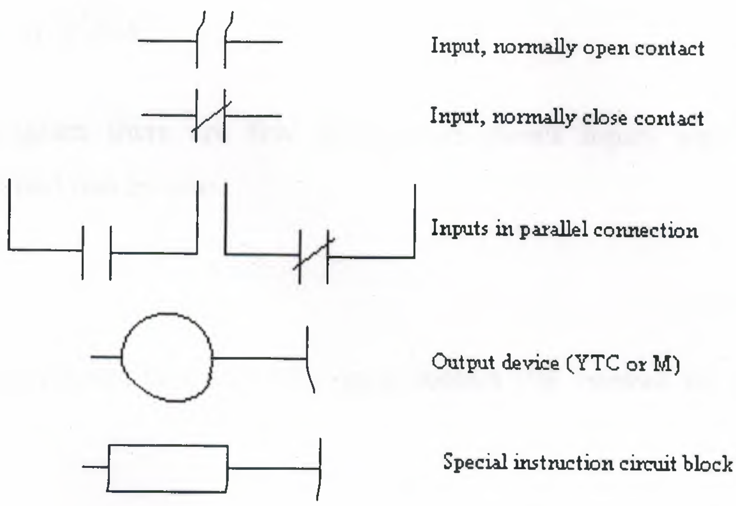


Figure 2.5.1 : Graphic ladder symbols.

A preferable alternative is to use a graphic programmer, as available for several programmable controllers including the small Mitsubishi and Toshiba models from Japan. Graphic programming allows the user to enter his program as a symbolic ladder circuit layout, using standard logic symbols to represent input contacts, output coils, etc, as shown. This approach is user-friendlier than programming with mnemonic logic instructions, and can be considered as a higher-level form of language.

The programming panel translates or complies these graphic symbols into machine (logic) instructions that are stored in the PLC memory, relieving the user of this task.

Different types of graphic programmer are normally used for each family of programmable controllers, but they all support similar graphic circuit conventions. Smaller, hand-held panels are common for the small to medium-sized PLCs, although the same programming panel is often used as a 'field programmer' for these and larger PCLs in the same family. However, the majority of graphic programming for larger systems is carried out on terminal-sized units. Some of these units are also semi-portable, and may be operated alongside the PLC system under commissioning or test in-plant. In addition to screen displays, virtually all graphic-programming stations can drive printers for hard copy of programs and/or status information, plus program storage via battery-backed RAM or tape/floppy disk. The facility to load (blow) resident programs into EPROM ICs may be available on more expensive units.

2.6 Basic Instructions

In the ladder diagram there are few instructions shows inputs and outputs. These are explained at each part one by one .

2.6.1 Load

The load (LD) instruction is a normally open contact.The symbol for a load instruction is shown below.

Figure 2.6.1.1 : A Load (contact) symbol

This is used when an input signal is needed to be present for the symbol to turn on. When the physical input is on it is easy to say that the instruction is True. If the input is physically on then the symbol is on. An on condition is also referred to as a logic 1 state.

This symbol normally can be used for internal inputs, external inputs and external output contacts. Internal relays don't physically exist. They are simulated software relays.

2.6.2 Loadbar

The Loadbar instruction is a normally closed contact. The symbol for a loadbar instruction is shown below.

Figure 2.6.2.1 : A Loadnot normally closed contact.

This is used when an input signal does not need to be present for the symbol to turn on. When the physical input is off it can be said that the instruction is True. If the input is physically off then the symbol is on. An off condition is also referred to as a logic 0 state.

This symbol normally can be used for internal inputs, external inputs and sometimes, external output contacts. It is the exact opposite of the Load instruction.

*NOTE- With most PLCs this instruction (Load or Loadbar) MUST be the first symbol on the left of the ladder.

Table 2.6.2.1 : Logic values of Load and Loadbar

Logic State	Load	LoadBar
0	False	True
1	True	False

2.6.3 Out

The Out instruction is sometimes also called an Output instruction. The output instruction is like a relay coil. Its symbol looks as shown below.

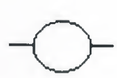


Figure 2.6.3.1 : An Out symbol

When there is a path of True instructions preceding this on the ladder rung, it will also be True. When the instruction is True it is physically On. This instruction can be thought as a normally open output. This instruction can be used for internal coils and external outputs.

2.6.4 Outbar

The Outbar instruction is sometimes also called an Outnot instruction. Some vendors don't have this instruction. The outbar instruction is like a normally closed relay coil. Its symbol looks like that shown below.

Figure 2.6.4.1: An Outbar (normally closed coil) symbol

When there is a path of False instructions preceding this on the ladder rung, it will be True. When the instruction is True it is physically On. This instruction can be thought as a normally closed output. This instruction can be used for internal coils and external outputs. It is the exact opposite of the Out instruction.

Table 2.6.4.1 : Logic values of Out and Outbar

Logic State	Out	Outbar
0	False	True
1	True	False

2.7 A Simple Example

The two figures below shows simple ladder diagram with its real world external physically connected relay circuit and the differences are easily seen.

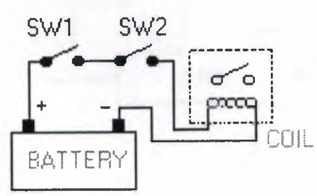


Figure 2.7.1 : Circuit of examle

In the above circuit, the coil will be energized when there is a closed loop between the + and - terminals of the battery. We can simulate this same circuit with a ladder diagram. A ladder diagram consists of individual rungs just like on a real ladder. Each rung must contain one or more inputs and one or more outputs. The first instruction on a rung must always be an input instruction and the last instruction on a rung should always be an output (or its equivalent).

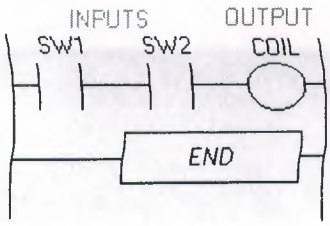


Figure 2.7.2 : Ladder diagram of the example

Notice in this simple one rung ladder diagram we have recreated the external circuit above with a ladder diagram. Here we used the Load and Out instructions. Some manufacturers require that every ladder diagram include an END instruction on the last rung. Some PLCs *also require an ENDH instruction on the rung after the END rung.*

2.8 PLC Registers

The previous example is taken and switch 2 (SW2) is changed to a normally closed symbol (loadbar instruction). SW1 will be physically OFF and SW2 will be physically ON initially. The ladder diagram now looks like figure 2.21:

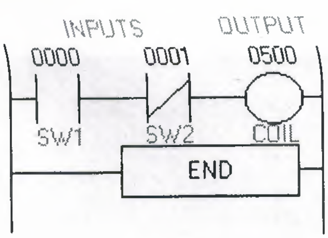


Figure 2.8.1 : *Addressed ladder diagram of previos example*

Notice also that an adrees was given to each symbol (or instruction). This address sets aside a certain storage area in the PLCs data files so that the status of the instruction (i.e. true/false) can be stored. Many PLCs use 16 slot or bit storage locations. In the example above two different storage locations or registers were used.

Table 2.8.1 : Register 00

REGISTER 00															
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
														1	0

Table 2.8.2 : Register 05

REGISTER 05															
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
															0

In the tables in register 00, bit 00 (i.e. input 0000) was a logic 0 and bit 01 (i.e. input 0001) was a logic 1. Register 05 shows that bit 00 (i.e. output 0500) was a logic 0. The logic 0 or 1 indicates whether an instruction is False or True. *Although most of the items in the register tables above are empty, they should each contain a 0.

Table 2.8.3 : Logical condition of diagram

LOGICAL CONDITION OF SYMBOL			
LOGIC BITS	LD	LDB	OUT
Logic 0	False	True	False
Logic 1	True	False	True

The plc will only energize an output when all conditions on the rung are TRUE. So, looking at the table 2.5, in the previous example SW1 has to be logic 1 and SW2 must be logic 0. Then and ONLY then will the coil be true (i.e. energized). If any of the instructions on the rung before the output (coil) are false then the output (coil) will be false (not energized).

Let's now look at a truth table of our previous program to further illustrate this important point. Truth table will show all possible combinations of the status of the two inputs.

2.9 Latch Instructions

Regular output coils are an essential part of our programs but must be remembered that they are only true when all instructions before them on the rung are also true. Then of course, the output will become false.

Think back to the bell example is done a few chapters ago. What would've happened if a push on/push off switch couldn't be found? Then would have been needed to keep pressing the button for as long as we wanted the bell to sound. The latching instructions let users use momentary switches and program the plc so that when it is push one time the output turns on and when it is pushed another time the output turns off.

2.10 Counters

A counter is a simple device intended to do one simple thing - count. Using them, however, can sometimes be a challenge because every manufacturer (for whatever reason) seems to use them a different way.

Types of counters:

There are up-counters (they only count up 1,2,3...). These are called CTU,(count up) CNT,C, or CTR. There are down counters (they only count down 9,8,7,...). These are typically called CTD (count down) when they are a separate instruction. There are also up-down counters (they count up and/or down 1,2,3,4,3,2,3,4,5,...) These are typically called UDC(up-down counter) when they are separate instructions.

Many manufacturers have only one or two types of counters but they can be used to count up, down or both. The theory is all the same regardless of what the manufacturers call them. A counter is a counter is a counter...

Most manufacturers also include a limited number of high-speed counters. These are commonly called HSC (high-speed counter), CTH (Counter high speed) or whatever. Typically a high-speed counter is a "hardware" device. The normal counters listed above are typically "software" counters. In other words they don't physically exist in the plc but rather they are simulated in software. Hardware counters do exist in the plc and they are not dependent on scan time.

When the program is running on the plc the program typically displays the current or "accumulated" value so the current count value can be seen.

Typically counters can count from 0 to 9999, -32,768 to +32,767 or 0 to 65535. Why the weird numbers? Because most PLCs have 16-bit counters.

Here are some of the instruction symbols that will be encountered (depending on which manufacturer be choosen) and how to use them. Should be remembered that while they may look different they are all used basically the same way.

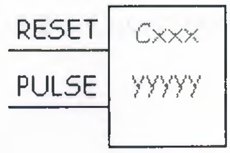


Figure 2.10.1 : Count-up counter

In this counter 2 inputs are needed.

One goes before the reset line. When this input turns on the current (accumulated) count value will return to zero.

The second input is the address where the pulses we are counting are coming from.

Cxxx is the name of the counter.If it is wanted to call it ''000'',then ''C000'' would be put.

yyyyy is the number of pulses we want to count before doing something. If 5 widgets are wanted to count before turning on a physical output to box them 5 would be put here. When the counter is finished . It will turn on a separate set of contacts that we also label Cxxx.

Note that the counter accumulated value ONLY changes at the off to on transition of the pulse input.

Here's the symbol on a ladder showing how a counter is set up(we'll name it counter 000) to count 100 widgets from input 0001 before turning on output 500. Sensor 0002 resets the counter.

Following figure is one symbol to encounter for an up-down counter. The same abbreviation is used as done for the example at the previous part (i.e. UDCxxx and yyyyy)

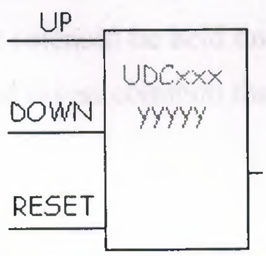


Figure 2.10.2 : Count up/down counter

One important thing to note is that counters and timers can't have the same name (in most PLCs). This is because they typically use the same registers.

Counters certainly are an essential tool.

2.11 Timers

Timer is exactly what the word says... It is an instruction that waits a set amount of time before doing something.

Different types of timers are available with different manufacturers. Here are most of them:

**On-Delay timer*:

On-Delay timer simply "delays turning on". In other words, after the sensor (input) turns on x-seconds be waited before activating a solenoid valve (output). This is the most common timer. It is often called TON (timer on-delay), TIM (timer) or TMR (timer).

**Off-Delay timer*:

Off-Delay timer is the opposite of the on-delay timer listed above. This timer simply "delays turning off". After the sensor (input) sees a target a solenoid (output) turnes on. When the sensor no longer sees the target the solenoid be held on for x-seconds before turning it off. It is called a TOF (timer off-delay) and is less common than the on-delay type listed above.

**Retentive or Accumulating timer*:

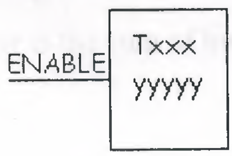
This type of timer needs 2 inputs. One input starts the timing event (i.e. the clock starts ticking) and the other resets it. The on/off delay timers above would be reset if the input sensor wasn't on/off for the complete timer duration. This timer however holds or retains the current elapsed time when the sensor turns off in mid-stream. For example,how long a sensor is on for during a 1 hour period is wanted to know. If one of the above timers is used they will keep resetting when the sensor turns off/on. This timer however, will give a total or accumulated time. It is often called an RTO (retentive timer) or TMRA (accumulating timer).

When the instructions before the timer symbol are true the timer starts "ticking". When the time elapses the timer will automatically close its contacts. When the program is running on the plc the program typically displays the elapsed or "accumulated" time for us so we can see the current value. Typically timers can tick from 0 to 9999 or 0 to 65535 times.

Most PLCs have 16-bit timers.The information about what this means is in a later chapter but for now suffice it to say that 0-9999 is 16-bit BCD (binary coded decimal) and that 0 to 65535 is 16-bit binary. Each tick of the clock is equal to x-seconds.

Typically each manufacturer offers several different ticks. Most manufacturers offer 10 and 100 ms increments (ticks of the clock). An "ms" is a milli-second or 1/1000th of a second. Several manufacturers also offer 1ms as well as 1 second increments. These different increment timers work the same as above but sometimes they have different names to show their timebase. Some are TMH (high speed timer), TMS (super high speed timer) or TMRAF (accumulating fast timer)

The figure shown at the following page is a typical timer instruction symbol (depending on which manufacturer be chosen). Remember that while they may look different they are all used basically the same way.



Figue 2.11.1 : On delay timer

The timer above is the on-delay type and is named Txxx. When the enable input is on the timer starts to tick. When it ticks yyyyy (the preset value) times, it will turn on its contacts that will be used later in the program. Remember that the duration of a tick (increment) varies with the vendor and the timebase used.

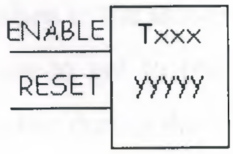


Figure 2.11.2 : Accumulating (Retentive) timer

This timer is named Txxx. When the enable input is on the timer starts to tick. When it ticks yyyyy (the preset value) times, it will turn on its contacts that will be used later in the program. Remember that the duration of a tick (increment) varies with the vendor and the timebase used. (i.e. a tick might be 1ms or 1 second or...) If however, the enable input turns off before the timer has completed, the current value will be retained. When the input turns back on, the timer will continue from where it left off. The only way to force the timer back to its preset value to start again is to turn on the reset input.

One important thing to note is that counters and timers can't have the same name (in most PLCs). This is because they typically use the same registers.

Always remember that although the symbols may look different they all operate the same way. Typically the major differences are in the duration of the ticks increments.

2.11.1 Timer Accuracy

There are two types of errors when using a timer. The first is called an input error. The other is called an output error. The total error is the sum of both the input and output errors.

**Input error*:

When the input turns on immediately after the plc looks at the status of the inputs during the scan cycle, the input error will be at its largest. This is because, the inputs are looked at once during a scan. If it wasn't on when the plc looked and turns on later in the scan, obviously there will be an error . Further should wait until the timer instruction is executed during the program execution part of the scan. If the timer instruction is the last instruction on the rung it could be quite a big error.

**Output error*:

Output error occurs depending upon when in the ladder the timer actually times out and when the plc finishes executing the program to get to the part of the scan when it updates the outputs. This is because the timer finishes during the program execution but the plc must first finish executing the remainder of the program before it can turn on the appropriate output. .

The hardware input error is caused by the time it takes for the plc to actually realize that the input is on when it scans its inputs. Typically this duration is about 10ms. This is because many PLCs require that an input should be physically on for a few scans before it determines its physically on.

The hardware output error is caused by the time it takes from when the plc tells its output to physically turn on until the moment it actually does. Typically a transistor takes about 0.5ms whereas a mechanical relay takes about 10ms.

2.12 Shift Registers

Shift registers are used to form a train of bits to store the previous on/off status. Each new change in status gets stored in the first bit and the remaining bits get shifted down the train.

The shift register called by many names. SFT (SHIFT), BSL (Bit Shift Left), SFR (Shift Forward Register) are some of the common names. These registers shift the bits to the left. BSR (Bit Shift Right) and SFRN (Shift Forward Register Not) are some examples of instructions that shift bits to the right. Note that not all manufacturers have shift registers that shift data to the right but most all do have left shifting registers.

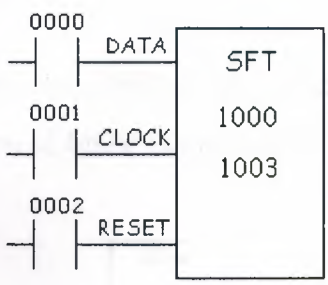


Figure 2.12.1 : Shift register

A typical shift register instruction has a symbol like that shown above. Notice that the symbol needs 3 inputs and has some data inside the symbol.

Data:

The data input gathers the true/false statuses that will be shifted down the train. When the data input is true the first bit in the register will be a 1. This data is only entered into the register on the rising edge of the clock input.

Clock:

The clock input tells the shift register to do its thing. On the rising edge of this input, the shift register shifts the data one location over inside the register and enters the status of the data input into the first bit. On each rising edge of this input the process will repeat.

Reset:

The shift register is most commonly used in conveyor systems, labeling or bottling applications, etc. Sometimes it's also conveniently used when the operation must be delayed in a fast moving bottling line. For example, a solenoid can't immediately kick out a bad can of beer when the sensor says its bad. By the time the solenoid would react the can would have already passed by. So typically the solenoid is located further down the conveyor line and a shift register tracks the can to be kicked out later when it's more convenient.

2.13 Getting and Moving Data

The single instruction is commonly called MOV> move. Some vendors also include a MOVN >move not. It has the same function of MOV but it transfers the data in inverted

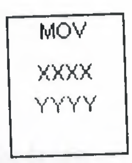


Figure 2.13.1 : MOV instruction symbol

The paired instruction typically is called LDA (Load Accumulator) and STA (Store Accumulator). The accumulator is simply a register inside the CPU where the plc stores data temporarily while its working. The LDA instruction typically looks like that shown below, while the STA instruction looks like that shown below to the right.

Figure 2.13.2 : LDA and STA instructions

Regardless of whether the one symbol or two symbol instruction set that used there is no choice as it depends on whose plc is used. They work the same way.

Let's see the single instruction first. The MOV instruction needs to know 2 things from user.

The two symbol instruction works in the same method but looks different. To use them two things must be supplied, one for each instruction:

2.14 Math Instructions

In general, PLCs almost always include these math functions:

Addition:

The capability to add one piece of data to another. It is commonly called ADD.

Subtraction:

The capability to subtract one piece of data from another. It is commonly called SUB.

Multiplication:

The capability to multiply one piece of data by another. It is commonly called MUL.

Division:

The capability to divide one piece of data from another. It is commonly called DIV.

Figure 2.14.1 : ADD symbol (dual method)

Many PLCs also include other math capabilities. Some of these functions could include:

- Square roots
- Scaling
- Absolute value
- Sine
- Cosine
- Tangent
- Natural logarithm
- Base 10 logarithm
- X^Y (X to the power of Y)
- Arcsine (tan, cos)

Some PLCs can use floating point math as well. Floating point math is simply using decimal points. In other words, can be said that 10 divided by 3 is 3.333333 (floating point). Or 10 divided by 3 is 3 with a remainder of 1(long division). Many micro/mini PLCs don't include floating point math. Most larger systems typically do.

2.15 Standard PLC Functions

These internal functions are not physical input or output. They are simulated with in the controller. Each function can be programmed with related contacts (again simulated), which may be used to control different elements in the program.

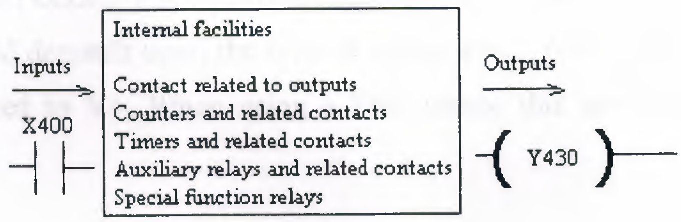


Figure 2.15.1 : Standard PLC functions.

As with physical inputs and outputs, certain number ranges are allocated to each block of function. The number range will depend both on the size of PLC and the manufacturer. For example, for the Mitsubishi F40- series, the details are as follows (octal numbering has been used):

Timers (T)	450-457 550-557	16 points (Elements)
Counters (C)	460-467 560-567	16 points

Table 2.15.1 : Timers and Counters Contacts

2.16 DC Inputs

Typically, dc input modules are available that will work with 5, 12, 24, and 48 volts.

DC input modules allow to connect either PNP (sourcing) or NPN (sinking) transistor type devices to them. If a regular switch is used typically don't have to worry about whether we wire it as NPN or PNP. Note that most PLCs won't let us mix NPN and PNP devices on the same module. When using a sensor however, have to worry about its output configuration. Always verify whether it's PNP or NPN.

The only things accessible to the user are the terminals labeled COMMON, INPUT 0000, INPUT 0001, INPUTxxxx... The common terminal either gets connected to V+ or ground. Where it's connected depends upon the type of sensor used. When using an NPN sensor this terminal is connected to V+. When using a PNP sensor this terminal is connected to 0V (ground).

A common switch (i.e. limit switch, pushbutton, toggle, etc.) would be connected to the inputs in a similar fashion. One side of the switch would be connected directly to V+. The other end goes to the plc input terminal. This assumes the common terminal is connected to 0V (ground). If the common is connected to V+ then simply connect one end of the switch to 0V (ground) and the other end to the plc input terminal.

2.17 AC Inputs

Ac input modules are available that will work with 24, 48, 110, and 220 volts.AC input modules are less common these days than dc input modules. The reason being that today's sensors typically have transistor outputs. A transistor will not work with an ac voltage. Most commonly, the ac voltage is being switched through a limit switch or other switch type. If the application is using a sensor it probably is operating on a dc voltage.

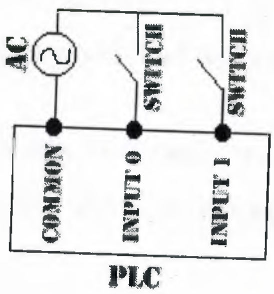


Figure 2.17.1 : AC device connected to PLC

The only things accessible to the user are the terminals labeled COMMON, INPUT 0000, INPUTxxxx... The common terminal gets connected to the neutral wire.

A common switch. Would be connected to the input terminals directly. One side of the switch would be connected directly to INPUT XXX. The other end goes to the ac hot wire. This assumes the common terminal is connected to neutral..

An ac input takes longer than a dc input for the plc to see. In most cases it doesn't matter to the programmer because an ac input device is typically a mechanical switch and mechanical devices are slow. It's quite common for a plc to require that the input be on for 25 or more milliseconds before it's seen. This delay is required because of the filtering which is needed by the plc internal circuit. Remember that the plc internal circuit typically works with 5 or less volts dc.

2.18 Relay Outputs

One of the most common types of outputs available is the relay output. A relay can be used with both AC and DC loads. A load is simply a fancy word for whatever is connected to outputs. It is called load because output is loading with something. If no load is connected to the output the outputs would be damaged.

Its circuit diagram typically looks like that shown above. When ladder diagram tells the output to turn on, the plc will internally apply a voltage to the relay coil. This voltage will allow the proper contact to close. When the contact closes, an external current is allowed to flow through external circuit. When the ladder diagram tells the plc to turn off the output, it will simply remove the voltage from the internal circuit thereby enabling the output contact to release. The load will than have an open circuit and will therefore be off.

Some common forms of a load are a solenoid, lamp, motor, etc. These "loads" come in all sizes. Electrical sizes, that is. Always the specifications of the load should be checked before connecting it to the plc output..

CHAPTER 3 COMPONENTS USED IN THE MACHINE

3.1 DC Motor

Figure 3.1.1 : DC motor

The *stator* is the stationary outside part of a motor. The *rotor* is the inner part which rotates. In the motor animations, *red* represents a magnet or winding with a north polarization, while *green* represents a magnet or winding with a south polariztion. Opposite, *red* and *green*, polarities attract.

The stator of a permanent magnet dc motor is composed of two or more permanent magnet pole peices. The rotor is composed of windings which are connected to a mechanical commutator. In this case the rotor has three pole pairs. The opposite polarities of the energized winding and the stator magnet attract and the rotor will rotate until it is aligned with the stator. Just as the rotor reaches alignment, the brushes move across the commutator contacts and energize the next winding. Commutator contacts are brown and the brushes are dark grey. A yellow spark shows when the brushes switch to the next winding.

Notice that the comutator is staggered from the rotor poles. If the connections of a dc motor are reversed the motor will change directions. Though it will not always work as well in both directions.

This is a permanent magnet dc motor. Two other types of dc motors are *series wound* and *shunt wound* dc motors. These motors also use a similar rotor with brushes and a commutator. However, the stator uses windings instead of permanent magnets. The basic principle is still the same. A series wound dc motor has the stator windings in series with the rotor. A shunt wound dc motor has the stator windings in parallel with the rotor winding. The series wound motor is more common. A series wound motor is also called a *universal* motor. It is universal in the sense that it will run equally well using either an ac or a dc voltage source. Reversing the polarity of both the stator and the rotor cancel out. Thus the motor will always rotate the same direction irregardless of the voltage polarity. A universal motor is in a sense an ac motor in that it will operate from an ac power source.

3.1.1 Brushless DC Motor

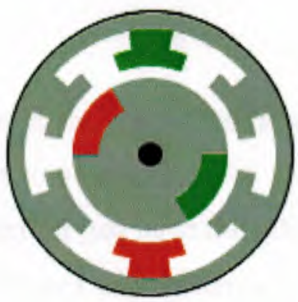


Figure 3.1.1.1 : DC Brushless motor

Brushless DC motors are refered to by many aliases: brushless permanent magnet, permanent magnet ac motors, permanent magnet synchronous motors ect. The confusion arises because a brushless dc motor does not directly operate off a dc voltage source. The basic principle of operation is similar to a dc motor.

A brushless dc motor has a rotor with permanent magnets and a stator with windings. It is essentially a dc motor turned inside out. The brushes and commutator have been eliminated and the windings are connected to the control electronics. The control electronics replace the function of the commutator and energize the proper winding.

The winding are energized in a pattern which rotates around the stator. The energized stator winding leads the rotor magnet, and switches just as the rotor aligns with the stator.

There are no sparks, which is one advantage of the bldc motor. The brushes of a dc motor have several limitations; brush life, brush residue, maximum speed, and electrical noise. BLDC motors are potentially cleaner, faster, more efficient, less noisy and more reliable. However, BLDC motors require electronic control.

3.2 Transformer

Transformers are one of the most powerful instruments in the field of AC electronics. More often then not the transformer is the first component to be used in AC circuit because its ability to radically change electrical values in a small area with a substantial cost and labor reduction. Although the transformer has many applications and has just as many shapes, this discussion is aimed in giving you an overall introduction to the transformer and details will come in the near future.

As the name implies, the transformers main function is to have an AC signal for the input and transform ether voltage or current to a desired level. Whether it be to increase or decrease voltage, increase or decrease current, or just have the efficiency level so low that the output is many times lower then the input. This is why in terms of electronics are about stepping up or stepping down current or voltage

The basic transformer is constructed of two coils "primary and secondary" that are then wrapped around a soft iron core. Although the coils never electrically touch "transfer electrons", the primary coil does influence the secondary coil so that it looks as if there was current flowing through the transformer.Can be easily seen how the basic transformer is constructed in the illustration.

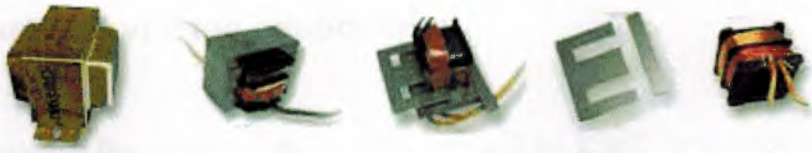


***Figure 3.2.1** : Construction of basic transformer*

When the alternating current flows through the primary coil a field is created that is proportional to the strength of power. So when AC, that is Alternating Current it traveling through the coil an alternating magnetic field is created. This satisfies the conditions needed for power generation for the secondary coil.

The voltage and static are very closely related, in fact static is what gives voltage its force even though voltage is just a measurement of static and electron pressure. The static is the dominant force in any object because it binds atoms together to form all the elements in the universe. Also magnetism is a property of static electricity, with this in mind can be then concluded that voltage, static, and magnetism are all related.

When these magnetic lines travel through the secondary coil electrons are ripped from their atoms and then replaced with electrons from Ground, returning alternation, or electrons that have completed the loop.

3.2.1 Concept of windings

The concept of stepping up or down certain values has to do with the number of turns each coil contains, that is, how many loops of wire are within the coil. This is important because the number of turns means that there is more efficiency within the transformer and as a result more influence from the primary coil to the secondary coil.

To step up or down a certain value what is needed to be aware of is which coil has more turns. This is because if the primary coil has more turns than the secondary coil, result would be an output of more current but less voltage.

If the secondary coil has more turns than the primary coil then the output of the transformer would be more voltage but less current.

3.2.2 Current and voltage relationship

To determine what the output of the transformer will be it will be needed to be aware of the math behind the transformer. Ratios are used when specifying the turns but simple dividing and multiplying are the only tools that will be needed to determining the output of the transformer and both equations can be seen below.

Ratio = secondary coil turns: Primary coil turns

Output Current = Input current * (Primary coil / Secondary coil)

or

I = I(Cp / Cs)

Output Voltage = Input voltage * (Secondary coil / Primary coil)

or

E = E(Cs / Cp)

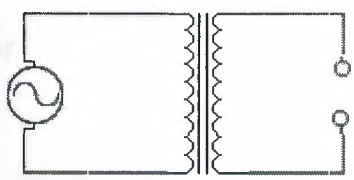


Figure 3.2.2.1 : Transformer

Let's say that in the picture above the primary coil that has the AC input has an amperage of 2A input and the secondary coil has 1A output.

- It is known from the sections above if the primary coil has more turns, then the output will be more current.
- But the output is less current so it means that the primary coil has less turns then the secondary coil.

- The output current is exactly half of the input
- The primary coil has exactly half the turns

3.3 Resistor

The resistor is one of the most diverse and easiest of all the electrical components be found in the average radio or TV set. This is because it has been around for many years and plays such a vital role that it will continue to in many new shapes and sizes to come. Today there are many different resistors in circulation, all of which will be explained shortly but for now lets go over some of the most important details.

The resistor is a component that has one purpose and that is to resist current and voltage by means of combining conductive material with a nonconductive one to form a substance that allows electrons to flow through its self but not as efficiently as a typical wire. The unit of measuring how much the resistor will oppose current is measured in ohms and to determine the outcome of the resistor we would use mathematical formulas known as ohms law.

There are three main types of resistors. These are low power resistors, high power resistors and variable resistors.

3.3.1 Low power resistor

Figure 3.3.1.1 : The symbol of low power and high power resistors

The carbon film resistor is composed up of a resistive material like graphite that is then cut into blocks or wrapped, or grafted in a desired way. For example, the length of the resistive material will determine how much resistance there will be while the width of the resistive material will determine what kind of power it can handle, the wider the more power it can handle. The schematic symbol can be seen in the figure above while the different types of carbon film resistors can be seen below There are some three distinct types of carbon film resistors which as follows:

The standard film resistor (A) :

A circular resistor with two pins extending from opposite sides or the barrel- shaped resistor.

The chip resistor (B) :

This type of resistor was introduced in the late 80's to accommodate for the ever shrinking computer components where there can be up to 6 layers per circuit board.

The network resistor (C) :

This type of resistor comes in (SIPP) form and can contain up to 12 resistors in a compact space that can not compare.

3.3.1.1 Color Code Explanation & Chart

Leaded resistor values are marked onto the body of the resistor using a series of colored bands. These give the value of the resistor as well as other information including the tolerance and sometimes the temperature coefficient. The band closest to the end of the resistor body is taken to be Band 1. The first two bands are the significant figures of the value, and the third is a multiplier, i.e. red black brown would be 2 0 x 10^1 or 200 ohms .

Table 3.3.1.1.1 : Resistor color codes

Color	Band 1 1st Figure	Band 2 2nd Figure	Band 3 3rd Figure	Band 4 Tolerance
Black	0	0	10^0	
Brown	1	1	10^1	1%
Red	2	2	10^2	2%
Orange	3	3	10^3	
Yellow	4	4	10^4	
Green	5	5	10^5	
Blue	6	6	10^6	
Violet	7	7	10^7	
Gray	8	8	10^8	
White	9	9	10^9	
Gold			10^{-1}	5%
Silver			10^{-2}	10%
None				20%

3.3.2 High power resistor

The most common wire wound resistor is composed up a fairly resistive wire wrapped around a ceramic cylinder and typically has a power range form 5 to 50 watts and is most often found in power supplies and amplifiers. It is common to find these components to heat up to levels that burns to the touch and is why they are made up of ceramic, a fire resistant material. The schematic symbol is the same of the carbon film resistor so it is also quite easy to remember.

Figure 3.3.2.1 : Appearances of some wire wound resistors

3.3.3 Variable resistor

The variable resistor is a very important component that is found in many electrical for such things as tone and bass controls as well as volume. This is due to the fact that resistors can be joined together with other components to form filters for a desired levels. They can also be found in computer monitors for color or positioning as well as the dimming switch.

This is done through digital to analog and analog to digital circuits, one great advantage to this is that you are able to turn a knob instead of typing a value in every time you want to change the tint or brightness.

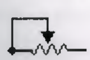


Figure 3.3.3.1 : The symbol of the variable resistor

The schematic for the variable resistor has stayed the same for quite some time. As seen it looks somewhat like a typical resistor but is an arrow coming out from one side pointing to the center of the resistor.

Figure 3.3.3.2 : Appearances of some variable resistors

CHAPTER 4 APPLICATION OF ELECTROMAGNETIC CONVEYOR MACHINE

4.1 BLOCK DIAGRAM OF THE ELECTROMAGNETIC CONVEYOR MACHINE

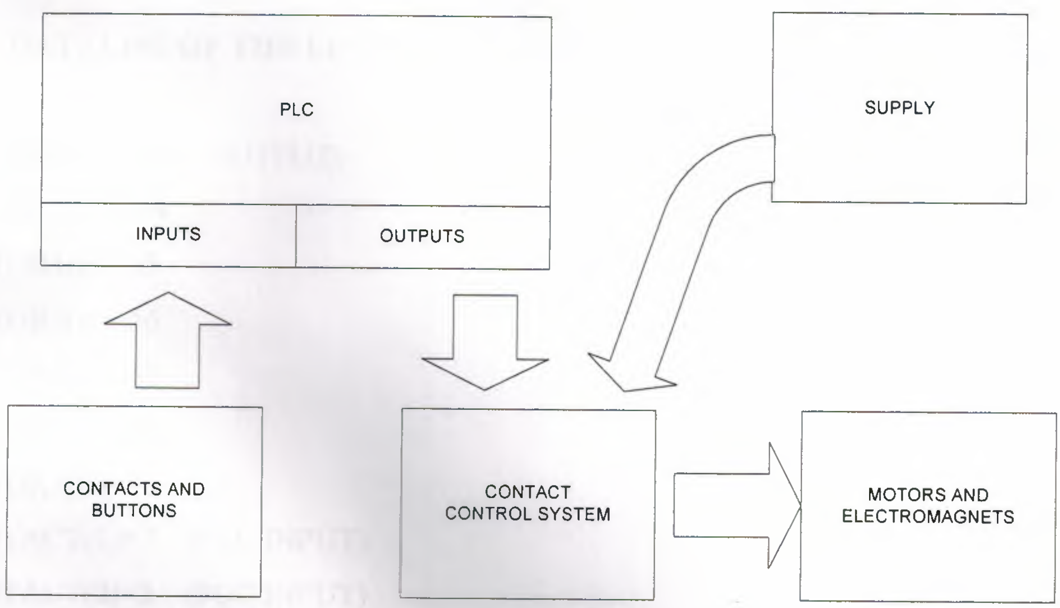


Figure 4.1.1 : Block Diagram of the System

4.2 MOTOR CONNECTION OF THE PLC

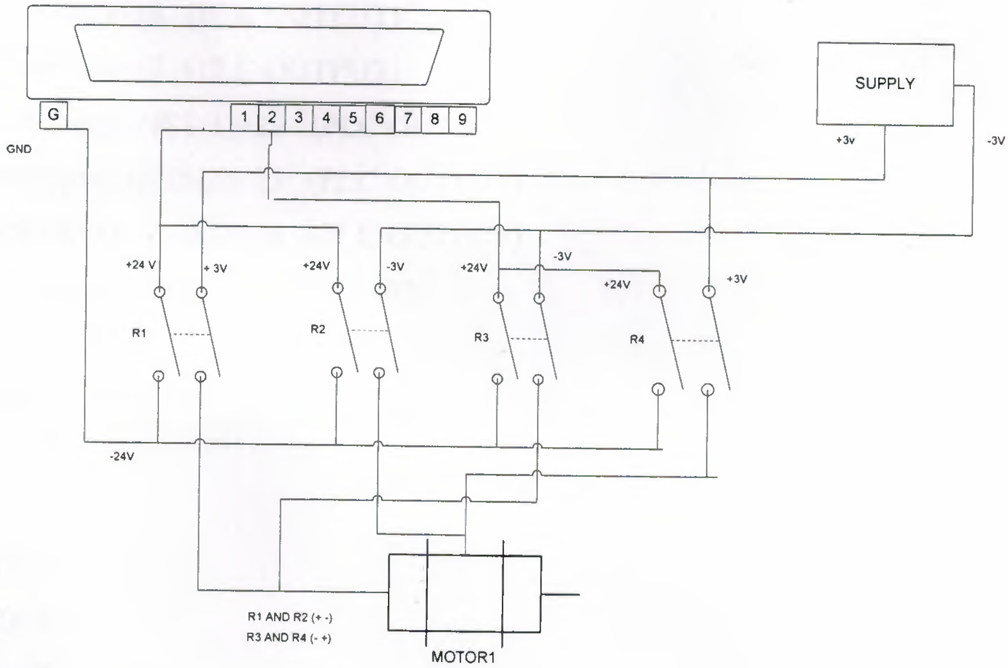


Figure 4.2.1 : Motor Connection of the System

4.2.1 CATALOG OF THE LINES

GROUND=0 (PLC OUTPUT)

MOTOR1 (A)=34

MOTOR1(B)=35

MOTOR2(A)=56

MOTOR2(A)=57

MOTOR3(A)=62

MOTOR3(B)=63

CONTACT(1)=7 (PLC INPUT)

CONTACT(2)=2 (PLC INPUT)

CONTACT(3)=3 (PLC INPUT)

CONTACT(4)=4 (PLC INPUT)

CONTACT(5)=5 (PLC INPUT)

CONTACT(6)=6 (PLC INPUT)

(M1R1-M1R2)=72 (PLC OUTPUT)

(M1R3-M1R4)=73 (PLC OUTPUT)

(M2R1-M2R2)=74 (PLC OUTPUT)

(M2R3-M2R4)=75 (PLC OUTPUT)

(M3R1-M3R2)=76 (PLC OUTPUT)

(M3R3-M3R4)=77 (PLC OUTPUT)

ELEKTROMAGNET(1)=22 (PLC OUTPUT)

ELECTROMAGNET(2)=23 (PLC OUTPUT)

ELECTROMAGNET(3)=24 (PLC OUTPUT)

ELEKTROMAGNET(COMMON)=02

(M1R1)=99

(M1R2)=98

(M1R3)=97

(M1R4)=96

(M2R1)=95

(M2R2)=94

(M2R3)=93

(M2R4)=92

(M3R1)=90

(M3R2)=89

(M3R3)=88

(M3R4)=87

GREEN BUTTON(START)=42 (PLC INPUT)

RED BUTTON(STOP)=43 (PLC INPUT)

SWITCH (CLOSE CONTACT)=44 (PLC INPUT)

COMMON (CONTACT AND BUTTONS)=0 (PLC INPUT)

Picture 4.1 : Electromagnetic Conveyor Machine

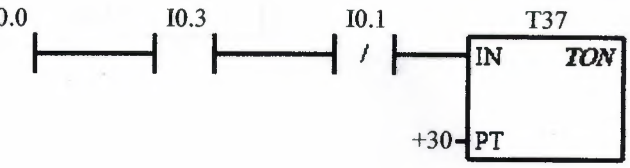
Picture 4.2 : Electromagnetic Conveyor Machine

GRAM TITLE COMMENTS

F1 for help and example program

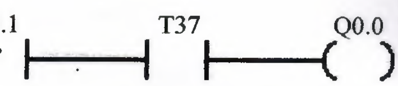
rk 1 Wait for 3 seconds and go up

or 3 seconds and go up



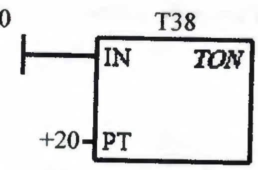
rk 2 electromagnet operate

rk 3 go up

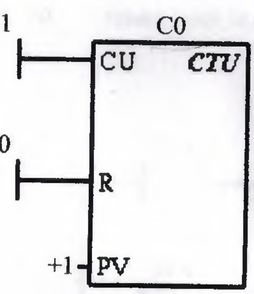


rk 4 electromagnet for right carrier operate

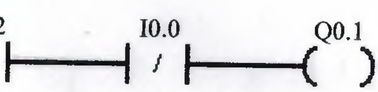
k 5 leave the object after two seconds



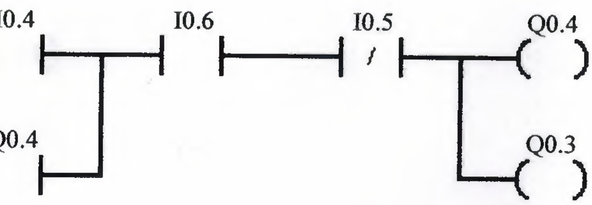
: 6 count when go up and reset when down



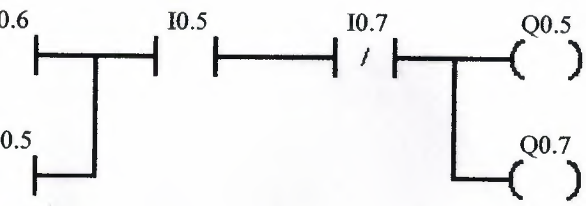
7 come down



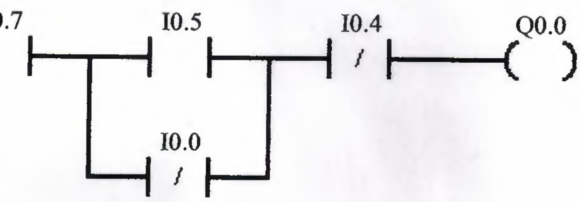
ork 8 CARRY OBJECT TO RIGHT



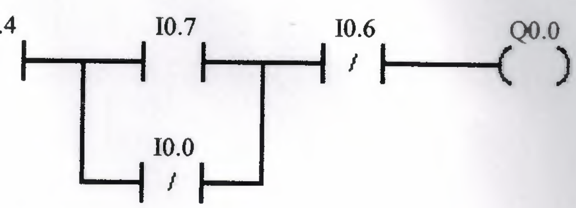
rk 9 carry object to left



rk 10 return back to right carrier



k 11 return back to left carrier



ork 12

ND)

```
//
//PROGRAM TITLE COMMENTS
//
//Press F1 for help and example program
//

NETWORK 1    //Wait for 3 seconds and go up
//
//Wait for 3 seconds and go up
//
//
LD      I0.0
A       I0.3
AN      I0.1
TON     T37, +30

NETWORK 2    //electromagnet operate
LDN     T38
=       Q0.0

NETWORK 3    //go up
LDN     I0.1
A       T37
=       Q0.0

NETWORK 4    //electromagnet for right carrier operate
LD      I0.3
=       Q0.3

NETWORK 5    //leave the object after two seconds
LD      C0
TON     T38, +20

NETWORK 6    //count when go up and reset when down
LD      I0.1
LD      I0.0
CTU     C0, +1

NETWORK 7    //come down
LDN     Q0.2
AN      I0.0
=       Q0.1

NETWORK 8    //CARRY OBJECT TO RIGHT
LD      I0.4
O       Q0.4
A       I0.6
AN      I0.5
=       Q0.4
=       Q0.3

NETWORK 9    //carry object to left
LD      I0.6
O       Q0.5
A       I0.5
AN      I0.7
=       Q0.5
=       Q0.7

NETWORK 10   //return back to right carrier
LD      I0.7
LD      I0.5
ON      I0.0
ALD
AN      I0.4
=       Q0.0

NETWORK 11   //return back to left carrier
LD      I0.4
```

```
LD      I0.7
ON      I0.0
ALD
AN      I0.6
=       Q0.0

NETWORK 12
MEND
```

REFERENCES

[1] Grob, Bernard, ‘‘Basic Electronics’’, McGraw Hill, Seventh Edition, 1993, p:40-51, 472-492, 484-486, 876

[2] Lander, C. W., ‘‘Power Electronics’’, McGraw Hill, Second Edition, p:3-5

[3] Özerdem, Ö. C., ‘‘Programmable Logic Controller and Programming’’, Near East University Press, 2002.

[4] Stenerson, Jon, ‘‘Fundamentals of Programmable Logic Controllers, Sensors of Communications’’, Prentice Hall, 1993

[5] Vithayathil, Joseph, ‘‘Power Electronics’’, McGraw Hill, 1995, p:65-81

[6] Warnock, G., ‘‘Programmable Controllers: Operation and Application’’, Prentice Hall, 1988

[7] Wilhelm, Robert E., ‘‘Programmable Controller Handbook’’, Hayden Books, 1987

[8] Lister E. C., Rusch R. J., "Electric Circuits And Machines", McGraw Hill, Seventh Edition, 1993, p: 178-205, 250-275

[9] Simatic S7-200 Programmable Controller, System Manual