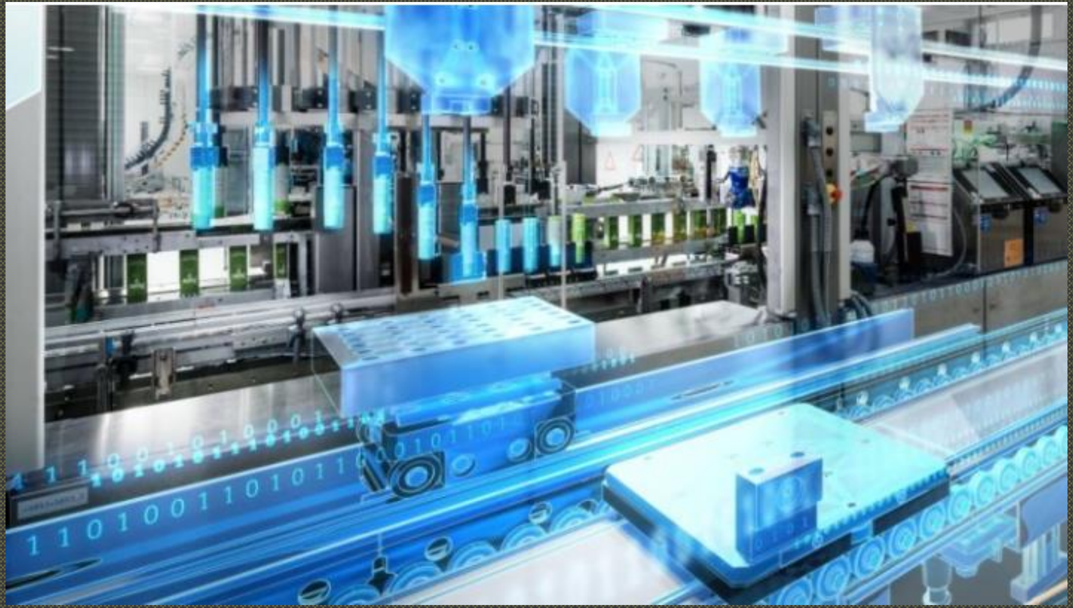
PROGRAMMABLE LOGIC CONTROLLERS MENG 3500

HUMBER

Automation Processes

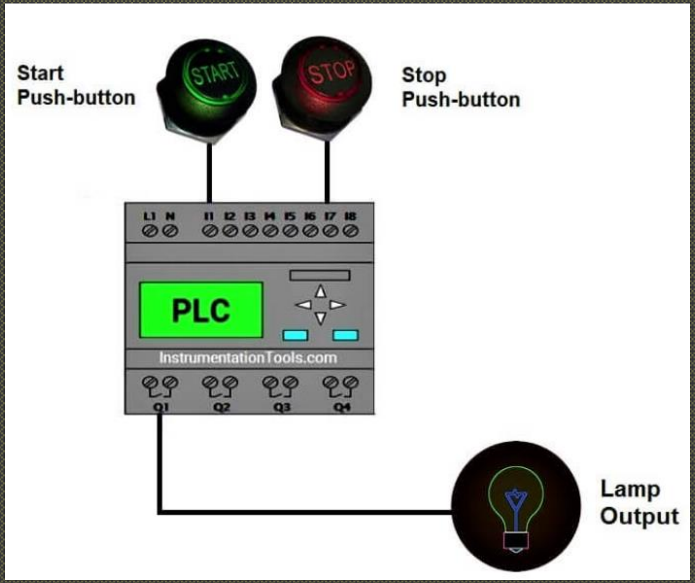

https://www.siemens.com/global/en/products/automation/systems.html

https://www.intel.com.br/content/www/br/pt/robotics/programmable-logic-controllers.html

Programming Standards: IEC 61131 has long been the industry's primary standard; many are considering a transition to IEC 61449 to maximize IT/OT convergence

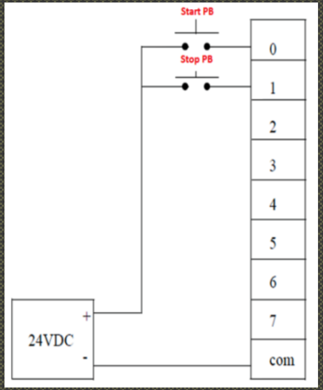
PROGRAMMABLE LOGIC CONTROLLERS

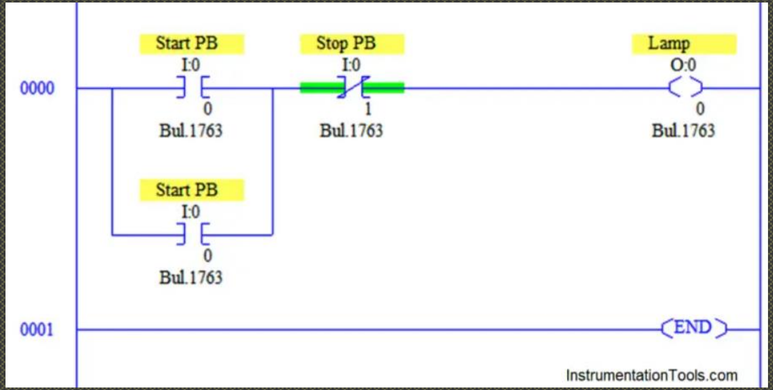
Basics

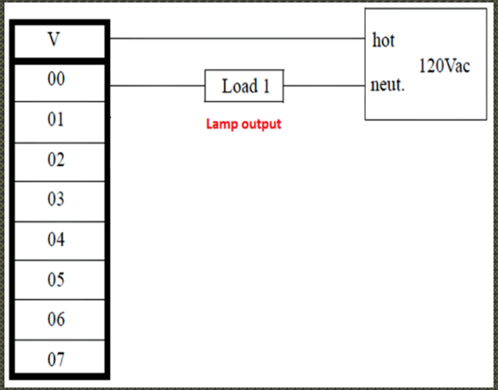


https://www.realpars.com/blog/plc-manufacturers

https://instrumentationtools.com/principle-of-operation-of-plc/







IEC 61131-3 Standard

Key Features:

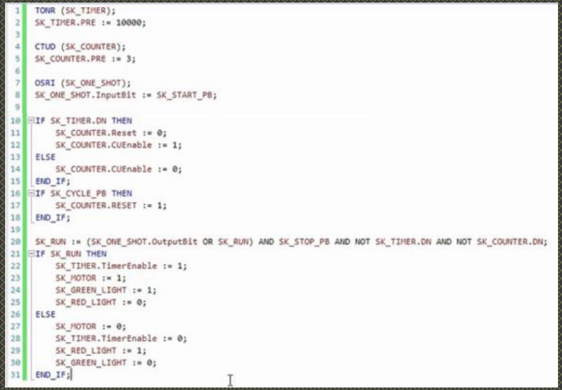
- Programming Languages:
  - Ladder Diagram (LD)
  - Function Block Diagram (FBD)
  - Structured Text (ST)
  - Instruction List (IL) *(deprecated in recent updates)*
  - Sequential Function Chart (SFC)

•Data Types: Defines basic, derived, and complex data types (Boolean, array, Timer/Counter).

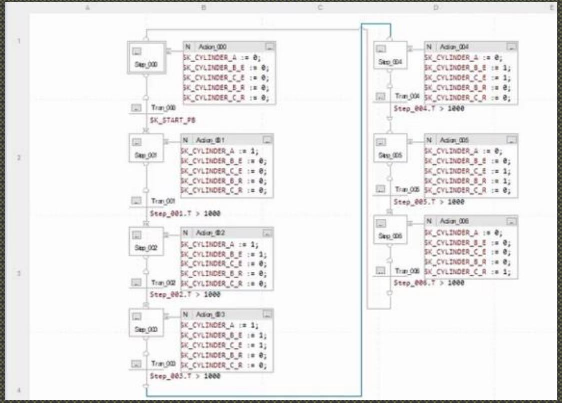
•Program Organization Units: Differentiates between Programs, Function Blocks, and Functions.

•Task Management: Defines cyclic and event-driven tasks.

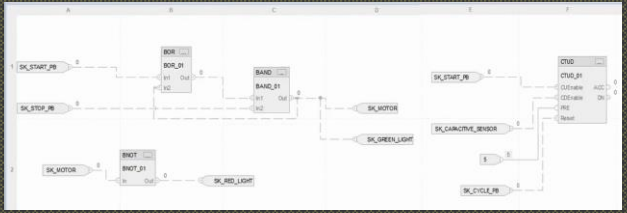
•Modularity and Reusability: Promotes reusable function blocks and modular code design.

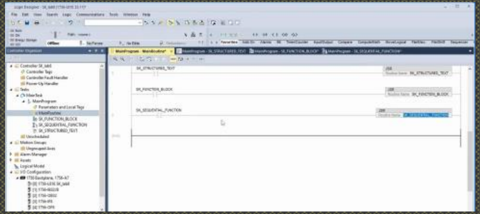


```
TONR (SK_TIMER);
SK_TIMER.PRE := 10000;

CTUD (SK_COUNTER);
SK_COUNTER.PRE := 3;

OSRI (SK_ONE_SHOT);
SK_ONE_SHOT.InputBit := SK_START_PB;

IF SK_TIMER.DN THEN
    SK_COUNTER.Reset := 0;
    SK_COUNTER.CUEnable := 1;
ELSE
    SK_COUNTER.CUEnable := 0;
END_IF;
IF SK_CYCLE_PB THEN
    SK_COUNTER.RESET := 1;
END_IF;

SK_RUN := (SK_ONE_SHOT.OutputBit OR SK_RUN) AND SK_STOP_PB AND NOT SK_TIMER.DN AND NOT SK_COUNTER.DN;
IF SK_RUN THEN
    SK_TIMER.TimerEnable := 1;
    SK_MOTOR := 1;
    SK_GREEN_LIGHT := 1;
    SK_RED_LIGHT := 0;
ELSE
    SK_MOTOR := 0;
    SK_TIMER.TimerEnable := 0;
    SK_RED_LIGHT := 1;
    SK_GREEN_LIGHT := 0;
END_IF;
```

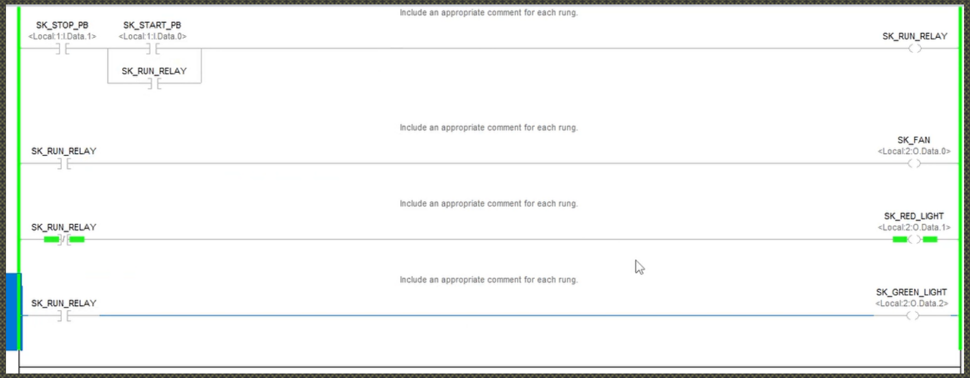


```
Action_000
SK_CYLINDER_A := 0;
SK_CYLINDER_B_E := 0;
SK_CYLINDER_C_E := 0;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 0;

Action_001
SK_CYLINDER_A := 1;
SK_CYLINDER_B_E := 0;
SK_CYLINDER_C_E := 0;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 0;

Action_002
SK_CYLINDER_A := 1;
SK_CYLINDER_B_E := 1;
SK_CYLINDER_C_E := 0;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 0;

Action_003
SK_CYLINDER_A := 1;
SK_CYLINDER_B_E := 1;
SK_CYLINDER_C_E := 1;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 0;

Action_004
SK_CYLINDER_A := 0;
SK_CYLINDER_B_E := 1;
SK_CYLINDER_C_E := 1;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 0;

Action_005
SK_CYLINDER_A := 0;
SK_CYLINDER_B_E := 0;
SK_CYLINDER_C_E := 1;
SK_CYLINDER_B_R := 1;
SK_CYLINDER_C_R := 0;

Action_006
SK_CYLINDER_A := 0;
SK_CYLINDER_B_E := 0;
SK_CYLINDER_C_E := 0;
SK_CYLINDER_B_R := 0;
SK_CYLINDER_C_R := 1;
```





IEC 61131-3 Standard



Naming Conventions

Consistent naming conventions improve readability and reduce errors.

- Tag Names: Use descriptive names for variables and tags.
  - Example: TEMP_SENSOR_01, MOTOR_START, ALARM_HIGH_TEMP
- Data Type Prefixes: Indicate the data type in the tag.
  - Example: b_ for boolean, i_ for integer, r_ for real numbers
- Function Block Naming: Prefix with FB
  - Example: FB_PumpControl
- Program Naming: Clearly define program scope and functionality.
- Best Practice: Follow a consistent naming standard throughout the project.

Basics

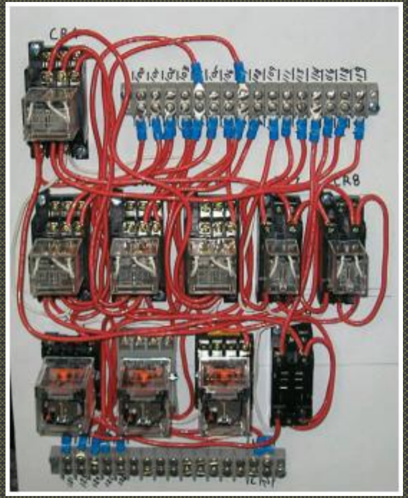
PLC technology plays a vital role in enhancing safety, automation, productivity, efficiency, and quality control across various industries, including:

- Manufacturing
- Automotive
- Energy and Utilities
- Food and Beverage
- Pharmaceuticals
- Water and Wastewater Treatment
- Mining
- Chemical Processing

Basics

A programmable logic controller (PLC) is an industrial grade computer that is capable of being programmed to perform control functions.

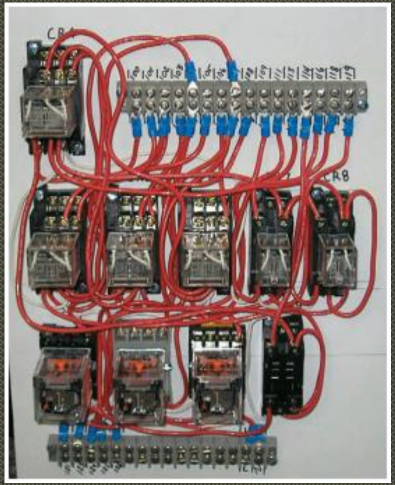
- Minimized much of the hardwiring associated with conventional relay control circuits,
- Fast response,
- Easy programming and installation,
- High control speed and network compatibility,
- Troubleshooting and testing convenience, and
- High reliability



(Frank D. Petruzella Programmable Logic Controllers 6th edition

PROGRAMMABLE LOGIC CONTROLLERS

- In the 1960s, automation processes primarily relied on control systems composed of logic circuits, control relays, electronic timers, and counters.
  - Adapting these traditional systems from one application to another required significant time and resource investment.
- The growing complexity of technical processes led to the need for flexible control systems capable of seamless adaptation across different applications.
- Programmable Logic Controllers (PLCs) emerged as specialized industrial computing devices designed to control and automate electromechanical processes in manufacturing and industrial environments.
  - PLCs are built to withstand harsh operating conditions and deliver reliable, real-time control over machinery and processes.
  - A key advantage of PLCs is their re-programmability, enabling quick and efficient adaptation from one automated application to another.



(Frank D. Petruzella Programmable Logic Controllers 6th edition

PROGRAMMABLE LOGIC CONTROLLERS

Key features and functions of PLCs include:

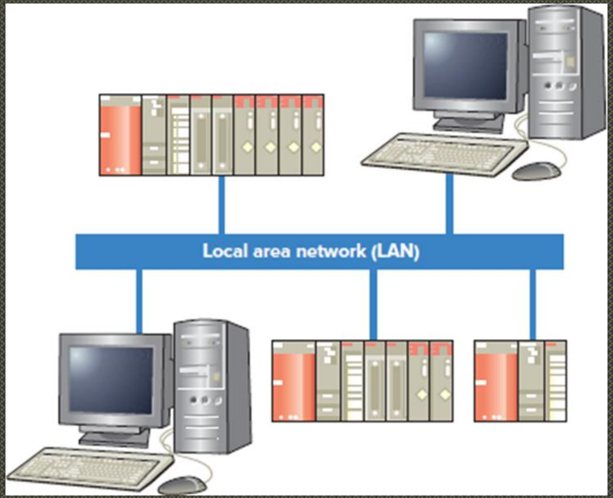
- Programming Capability: PLCs are programmable devices that can be configured to perform specific tasks based on the user's programming logic. This programming is typically done using ladder logic, a graphical programming language resembling relay logic diagrams.

- Input/Output (I/O) Handling: PLCs interface with various sensors and actuators through digital and analog inputs and outputs. They receive input signals from sensors, process the information, and send output signals to control actuators, such as motors, valves, and solenoids.

- Reliability: PLCs are known for their robustness and reliability in industrial environments. They are designed to operate in conditions with temperature variations, humidity, and electrical noise.

- Real-Time Operation: PLCs provide real-time control, ensuring that processes respond quickly to changes in input conditions. This is crucial for maintaining precise control over industrial machinery.

PROGRAMMABLE LOGIC CONTROLLERS



Additional features and functions of PLCs include:

- Modularity: PLC systems are modular, allowing for easy expansion and modification. Users can add or replace modules without requiring extensive changes to the entire system.

- Communication: PLCs often support communication with other devices and systems, enabling integration with supervisory control and data acquisition (SCADA) systems, human-machine interfaces (HMIs), and enterprise-level networks.

- Diagnostics and Monitoring: PLCs offer built-in diagnostics and monitoring features, making it easier to identify issues and troubleshoot problems in the control system.

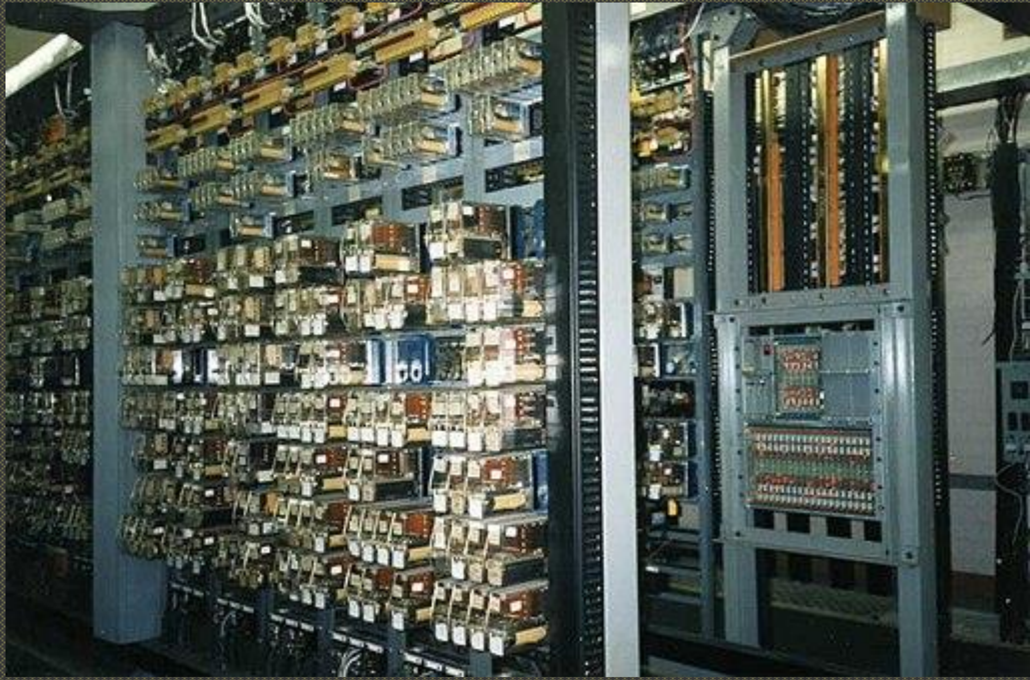
PLCs play a pivotal role in automating complex processes, improving efficiency, and ensuring precise control over industrial systems.

Evolution of PLCs

As industrial automation grew, factories needed multiple motors controlled by individual ON/OFF switches for each machine. The complexity of hardwiring relays led to the development of PLCs, offering a streamlined, electronic control solution.

https://ats.ae/resources/ats-knowledge-sharing-series-what-operational-technology-ot/

History of PLCs



The initial PLC, crafted by the Modicon company (Dick Morley), was introduced as a replacement for the control relay circuits at General Motors.

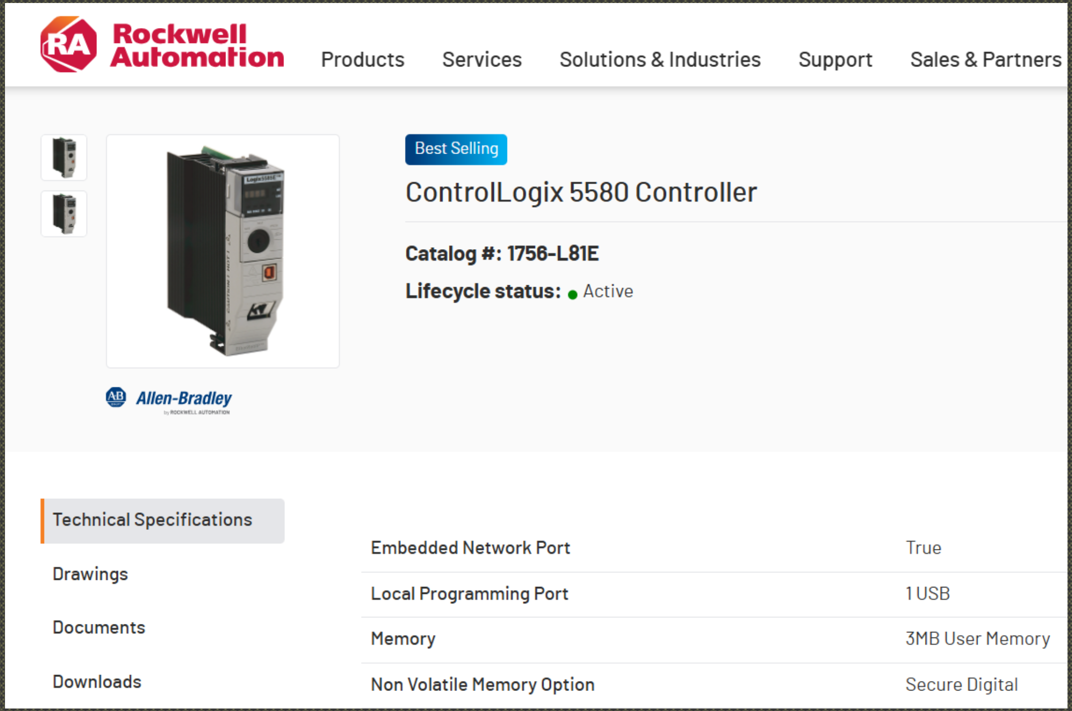
Engineers were presented with a compact device featuring input and output modules, streamlining the previously lengthy and costly process of redesigning existing circuits, rewiring, and making additions to the circuits.

https://library.automationdirect.com/history-of-the-plc/

HUMBER

PROGRAMMABLE LOGIC CONTROLLERS

Lab PLCs



Rockwell Automation | Products | Services | Solutions & Industries | Support | Sales & Partners

Best Selling

ControlLogix 5580 Controller

Catalog #: 1756-L81E

Lifecycle status: ● Active

Allen-Bradley

Technical Specifications

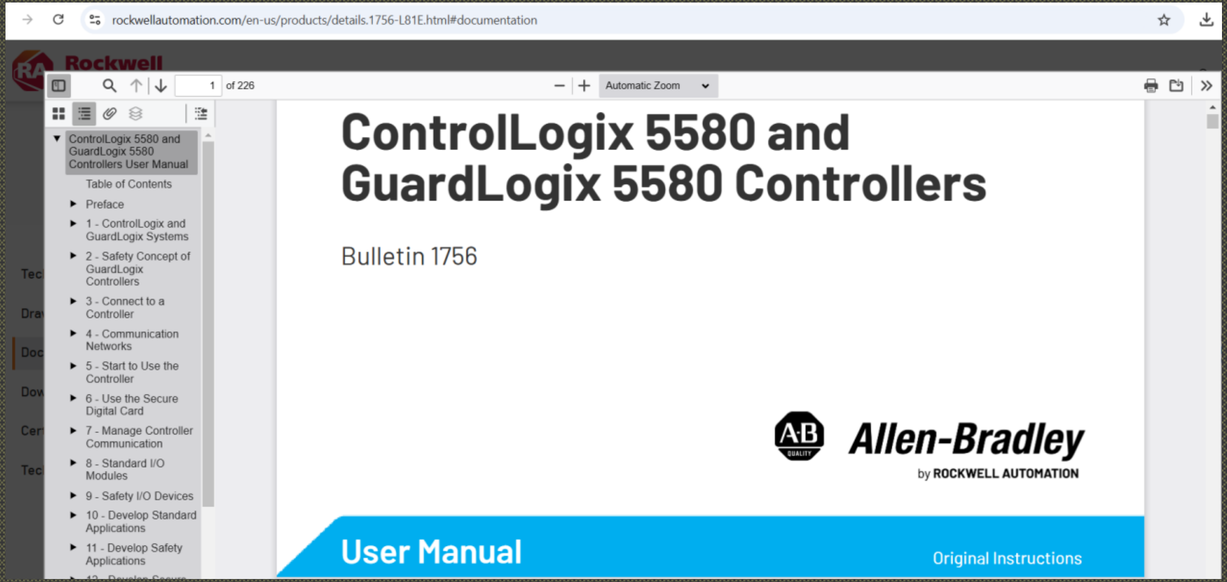
Drawings

Documents

Downloads

Embedded Network Port	True
Local Programming Port	1 USB
Memory	3MB User Memory
Non Volatile Memory Option	Secure Digital

https://www.rockwellautomation.com/en-il/products/details.1756-L81E.html



rockwellautomation.com/en-us/products/details.1756-L81E.html#documentation

1 of 226 | Automatic Zoom

ControlLogix 5580 and GuardLogix 5580 Controllers User Manual
- Table of Contents
- Preface
- 1 - ControlLogix and GuardLogix Systems
- 2 - Safety Concept of GuardLogix Controllers
- 3 - Connect to a Controller
- 4 - Communication Networks
- 5 - Start to Use the Controller
- 6 - Use the Secure Digital Card
- 7 - Manage Controller Communication
- 8 - Standard I/O Modules
- 9 - Safety I/O Devices
- 10 - Develop Standard Applications
- 11 - Develop Safety Applications

ControlLogix 5580 and GuardLogix 5580 Controllers

Bulletin 1756

Allen-Bradley by ROCKWELL AUTOMATION

User Manual Original Instructions

https://www.rockwellautomation.com/en-us/products/details.1756-L81E.html

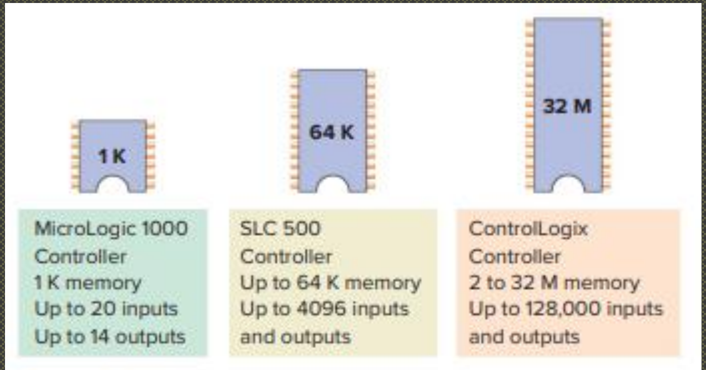
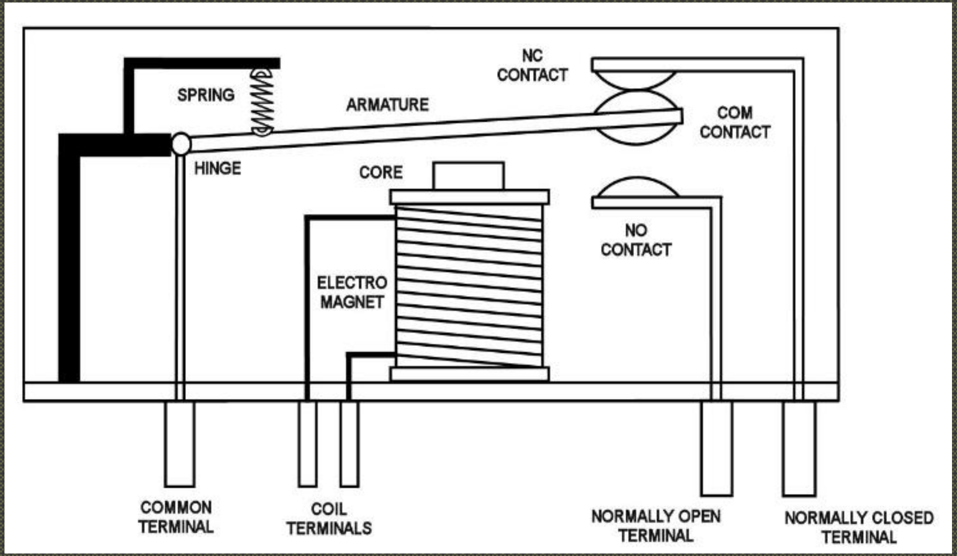

1 K
64 K
32 M
MicroLogic 1000 Controller 1 K memory Up to 20 inputs Up to 14 outputs
SLC 500 Controller Up to 64 K memory Up to 4096 inputs and outputs
ControlLogix Controller 2 to 32 M memory Up to 128,000 inputs and outputs

Figure 2-41 Typical PLC memory sizes.

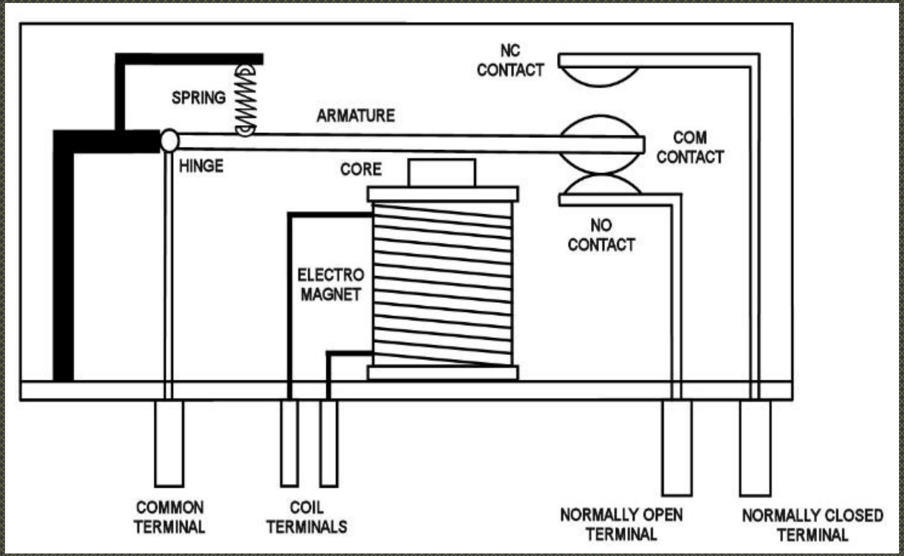
(Frank D. Petruzella Programmable Logic Controllers 4th edition)

Relay logic vs PLC Ladder logic

- De-energized relay



- Energized relay

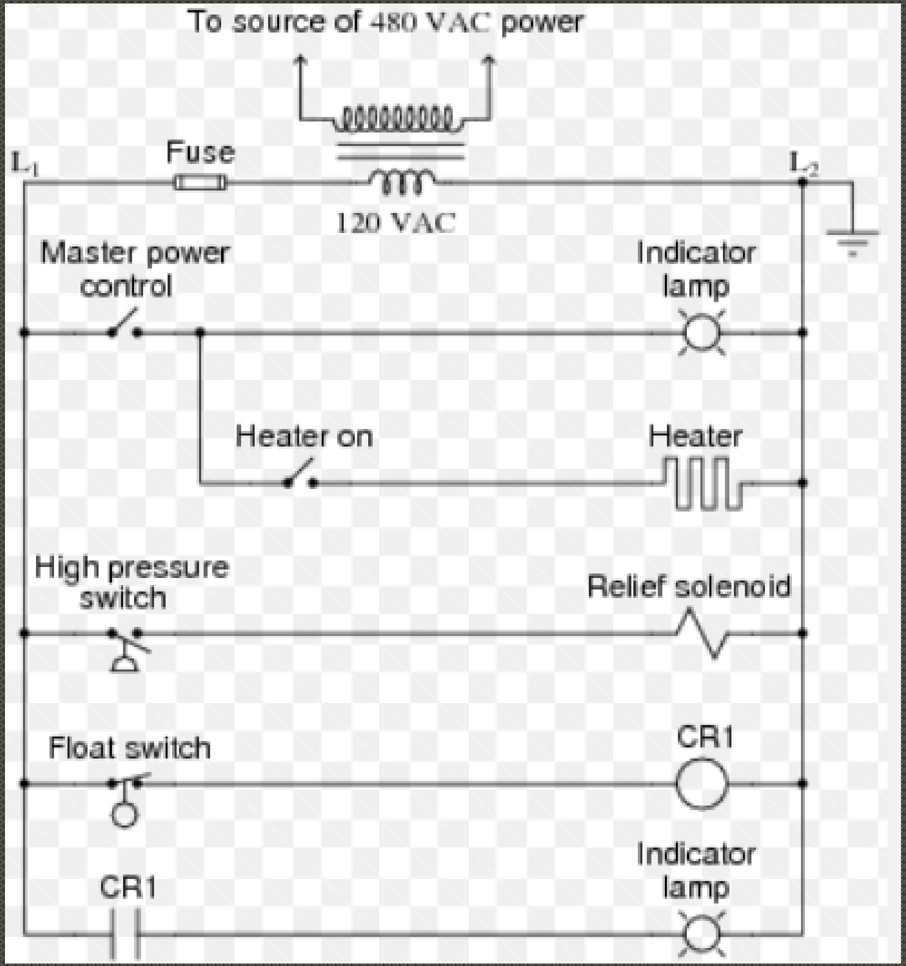


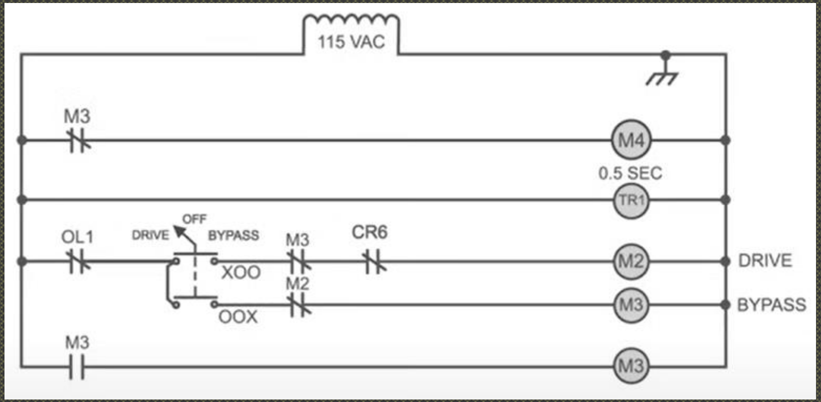
http://www.glolab.com/relays/relays.html

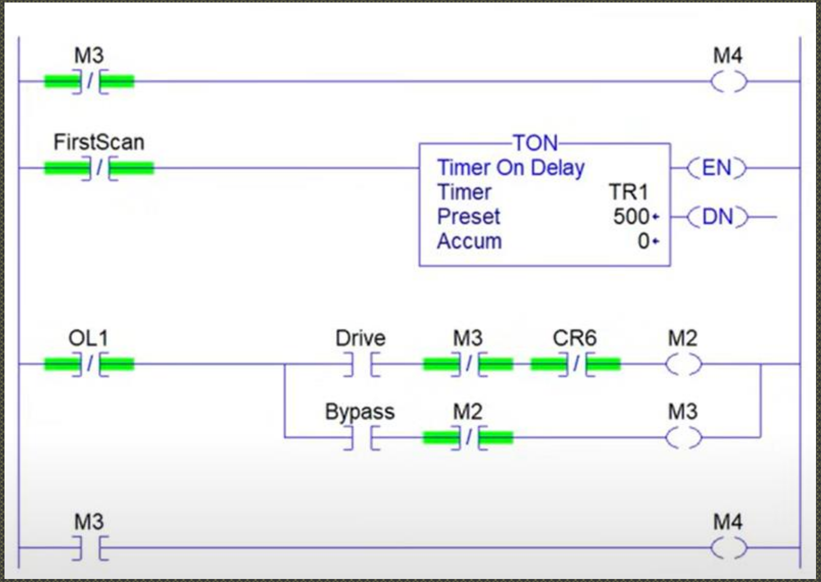
HUMBER

PROGRAMMABLE LOGIC CONTROLLERS

Relay logic vs PLC Ladder logic







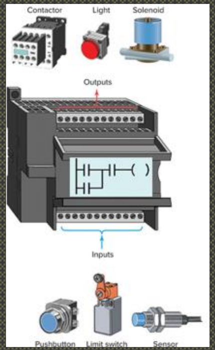
https://www.allaboutcircuits.com/worksheets/electromechanical-relay-logic/

https://www.youtube.com/watch?v=qaI48NCUvkA

HUMBER

PROGRAMMABLE LOGIC CONTROLLERS

Relay logic vs PLC Ladder logic



Relay Logic:

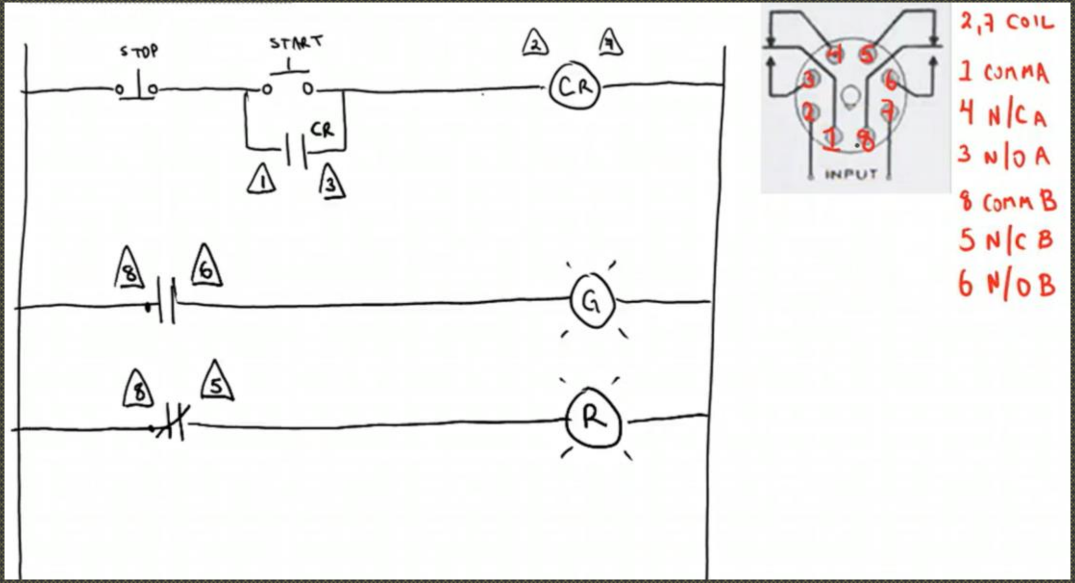
Technology: Relay logic is based on the use of electromechanical relays for switching and control. These relays are physical devices that open or close electrical contacts based on the input they receive.
Representation: The logic in relay systems is typically represented using ladder diagrams. These diagrams resemble electrical schematics and use symbols to represent relays, contacts, coils, and other components.
Programming: Relay logic is programmed by physically wiring the components together to create the desired control logic. This involves connecting relay contacts and coils using wires and terminal blocks.
Flexibility: While relay logic is straightforward, it can become complex and challenging to manage in large systems. Modifying or expanding the logic often requires rewiring, and troubleshooting can be time-consuming.

PLC Ladder Logic:

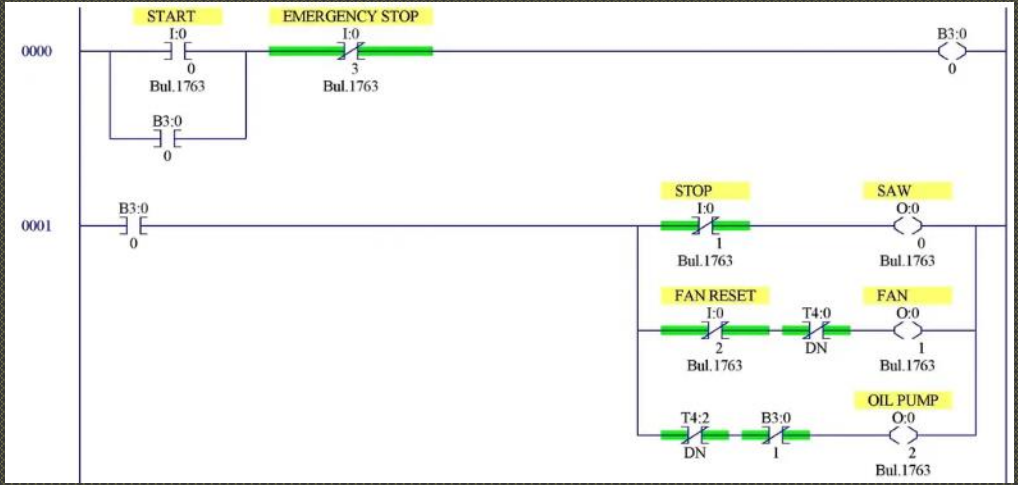
Technology: PLCs, on the other hand, are digital computers with input and output modules, a CPU, and memory.
Representation: PLC ladder logic is a graphical programming language that resembles relay ladder diagrams. It uses symbols to represent logical operations, timers, counters, and other control functions. However, it is entered and edited through software rather than physically rewiring components.
Programming: PLC ladder logic is programmed using software tools. Engineers or technicians use a programming interface to create and edit the ladder logic.
Flexibility: PLCs offer greater flexibility and ease of modification compared to relay logic. Changes to the control logic can be made through software without the need for physical rewiring. Additionally, PLCs often provide advanced features such as communication capabilities, data logging, and real-time monitoring.

PROGRAMMABLE LOGIC CONTROLLERS

Relay logic vs PLC Ladder logic



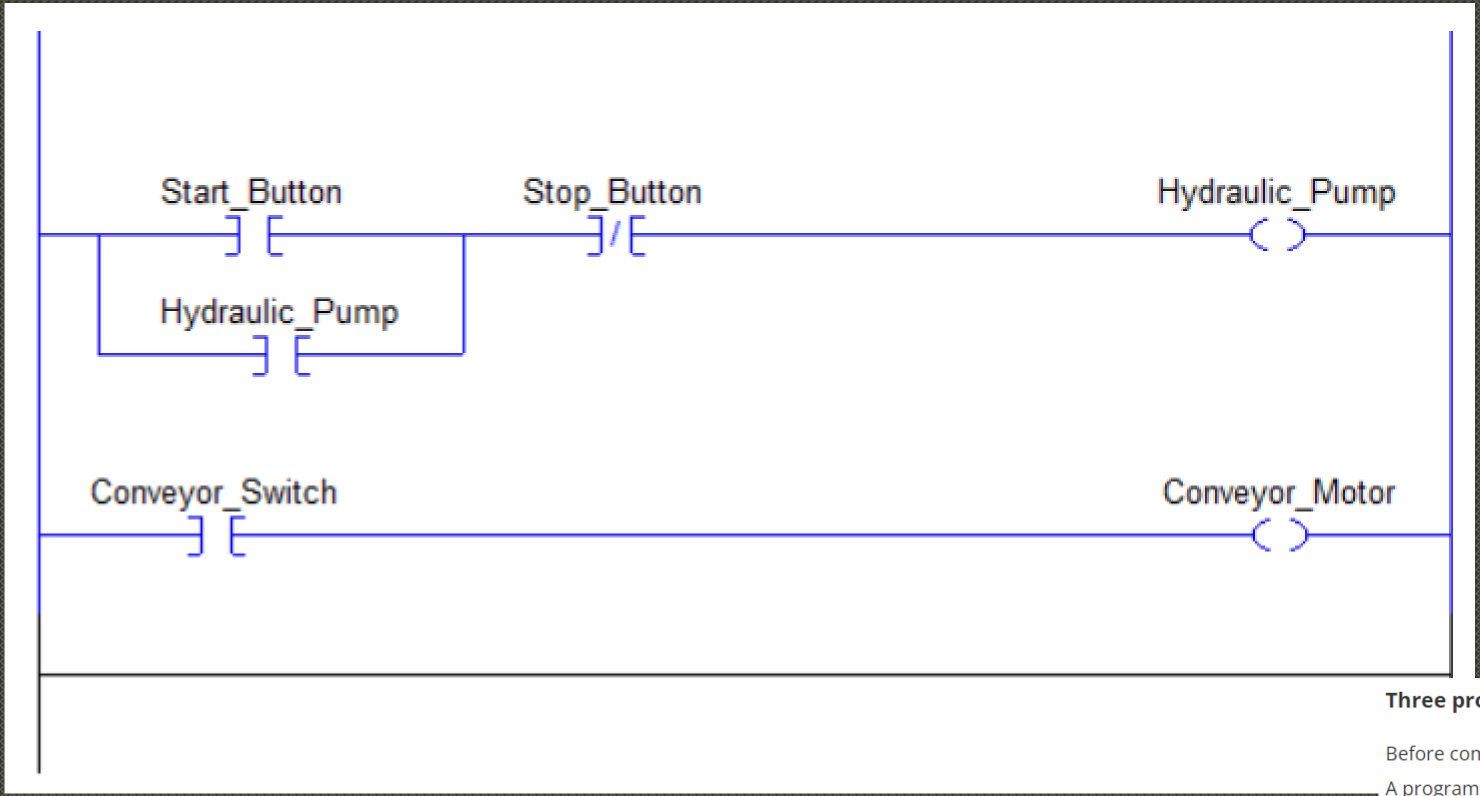
https://www.youtube.com/watch?v=aygXqlwCTis

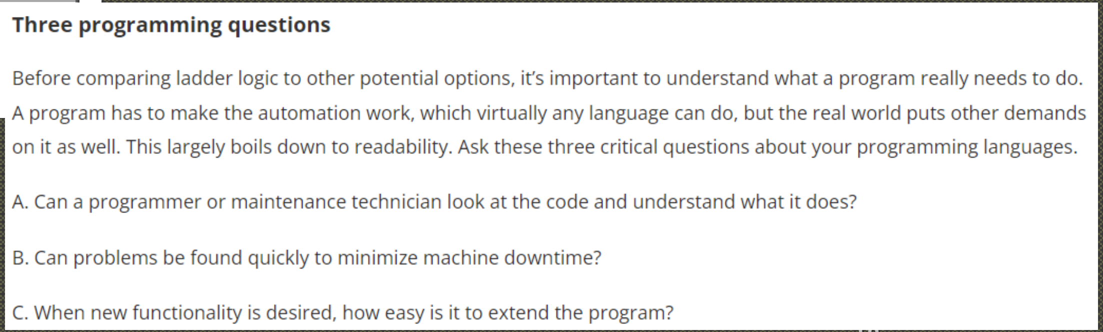


- https://instrumentationtools.com/ladder-logic-example-with-timers/
- https://instrumentationtools.com/plc-programming-examples/
- https://instrumentationtools.com/up-counter-plc-program/

PROGRAMMABLE LOGIC CONTROLLERS

Relay logic vs PLC Ladder logic





Three programming questions

Before comparing ladder logic to other potential options, it's important to understand what a program really needs to do. A program has to make the automation work, which virtually any language can do, but the real world puts other demands on it as well. This largely boils down to readability. Ask these three critical questions about your programming languages.

A. Can a programmer or maintenance technician look at the code and understand what it does?

B. Can problems be found quickly to minimize machine downtime?

C. When new functionality is desired, how easy is it to extend the program?

HUMBER

https://www.breen-machine.com/the-future-role-of-ladder-logic-in-automation-control-engineering/

PROGRAMMABLE LOGIC CONTROLLERS

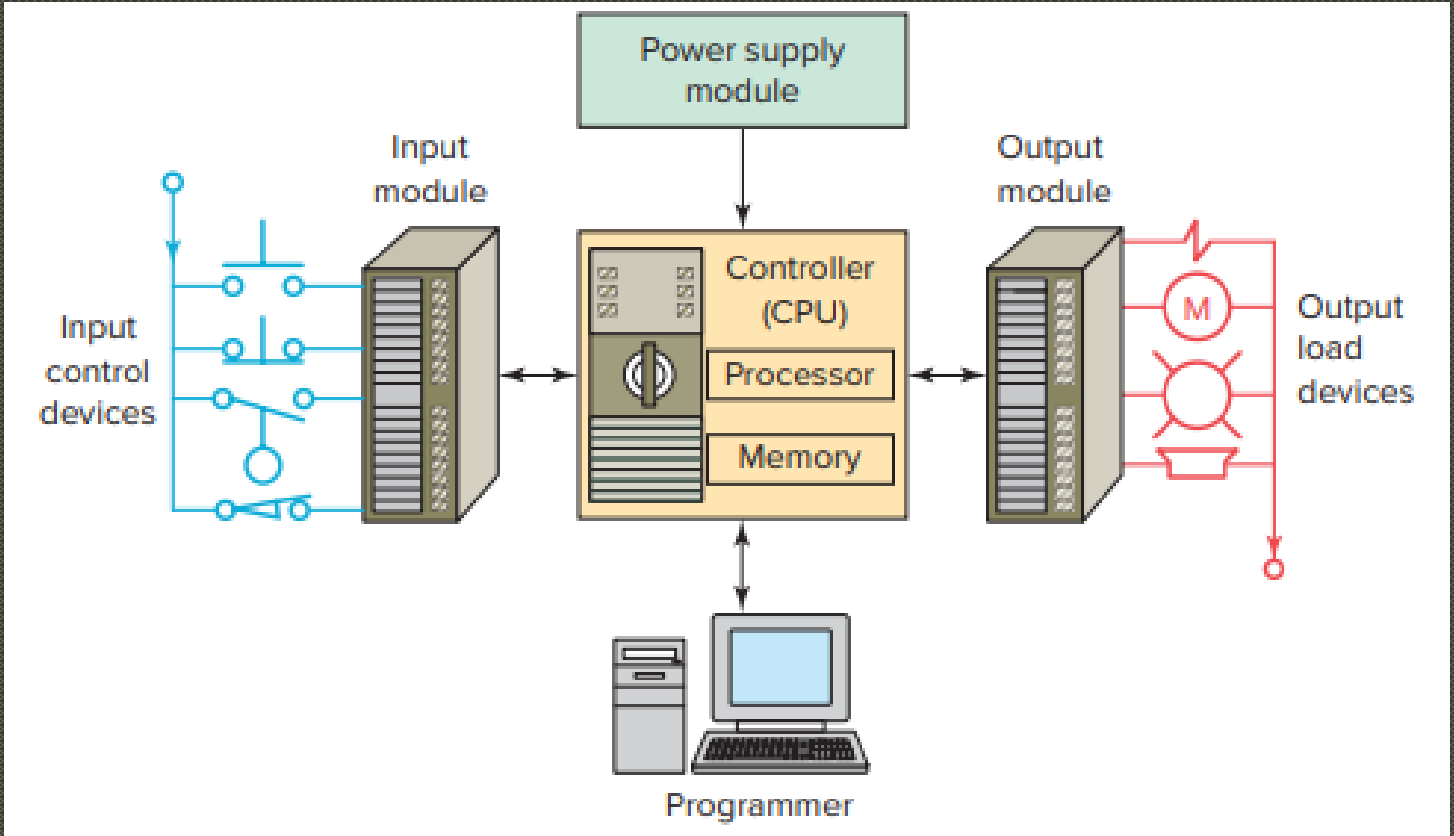
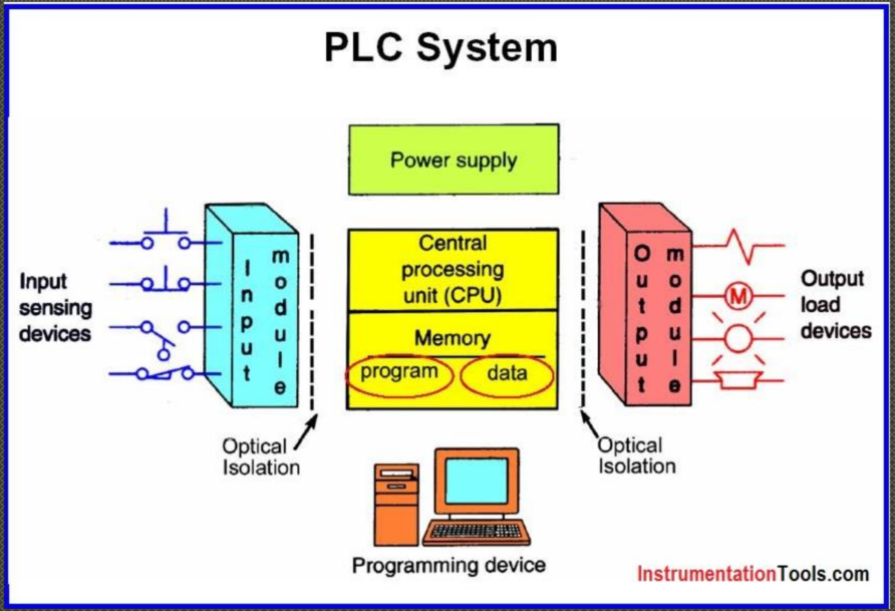


Figure 1-6 Typical parts of a programmable logic controller.

PLC architecture

- The basic PLC architecture is common for all brands. The device consists of the Central Processor Unit (CPU) with the internal memory, Input and Output modules.



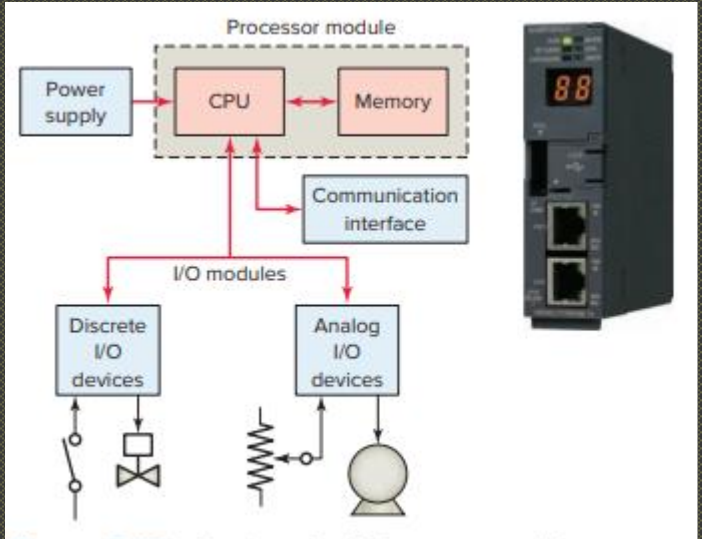
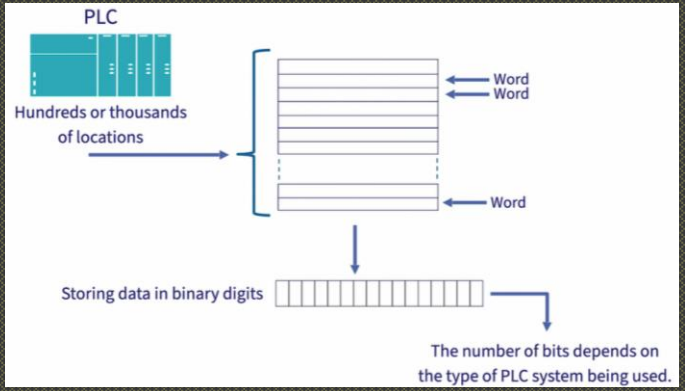
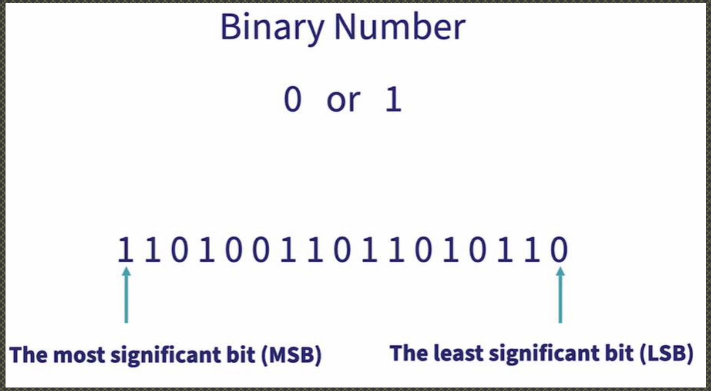


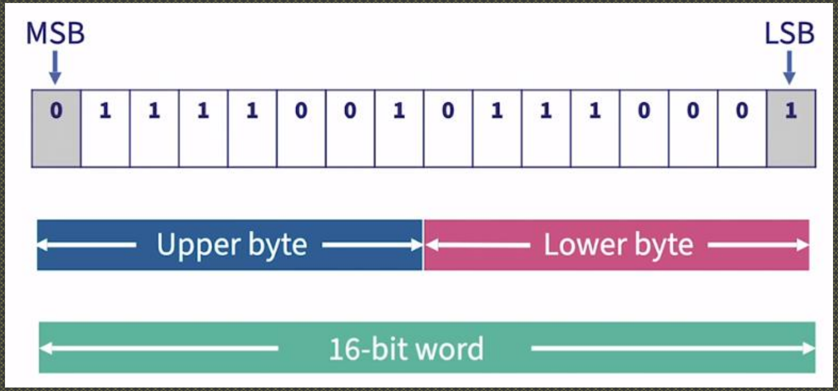
Figure 2-38 Sections of a PLC processor module.
Source: Courtesy Mitsubishi Automation.

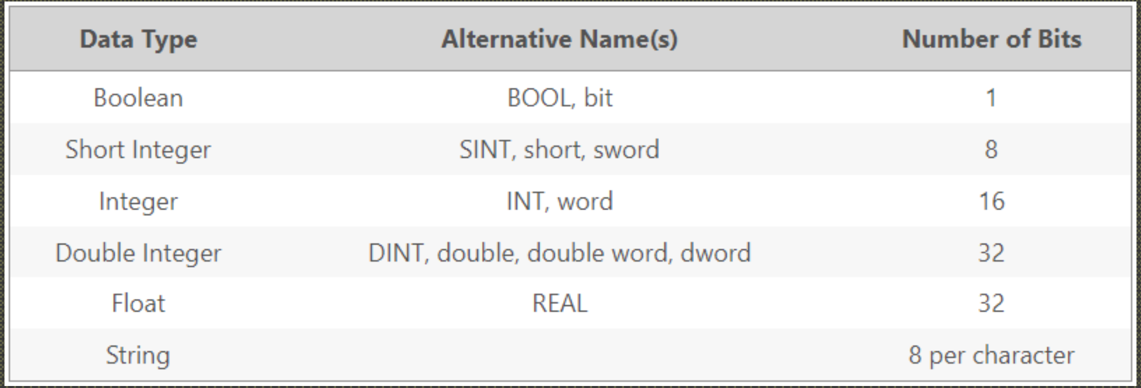
(Frank D. Petruzella Programmable Logic Controllers 4th edition)

Concepts of Bits and Words

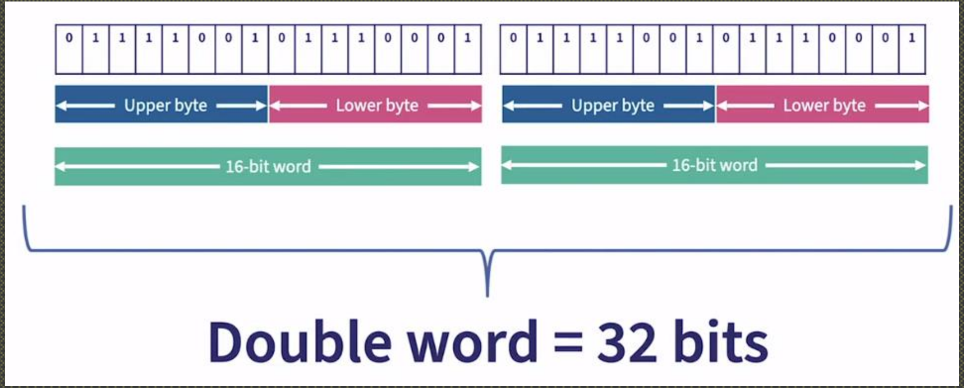

PLC
Hundreds or thousands of locations
Word
Word
Word
Storing data in binary digits
The number of bits depends on the type of PLC system being used.


Binary Number
0 or 1
1 1 0 1 0 0 1 1 0 1 1 0 1 0 1 1 0
The most significant bit (MSB)
The least significant bit (LSB)





Data Type	Alternative Name(s)	Number of Bits
Boolean	BOOL, bit	1
Short Integer	SINT, short, sword	8
Integer	INT, word	16
Double Integer	DINT, double, double word, dword	32
Float	REAL	32
String		8 per character



HUMBER

https://www.linkedin.com/learning/plc-program-flow-and-control-instructions/

PROGRAMMABLE LOGIC CONTROLLERS

PLC Programming Languages

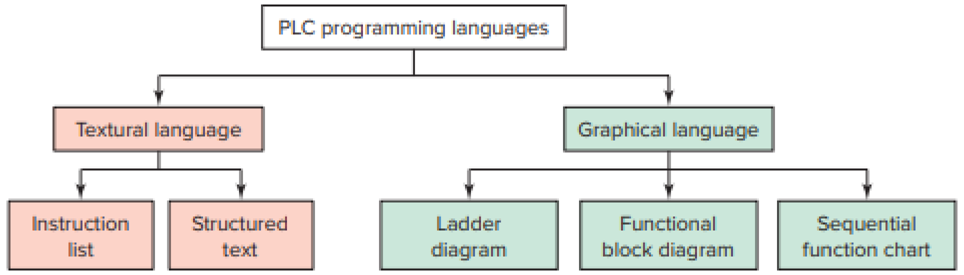
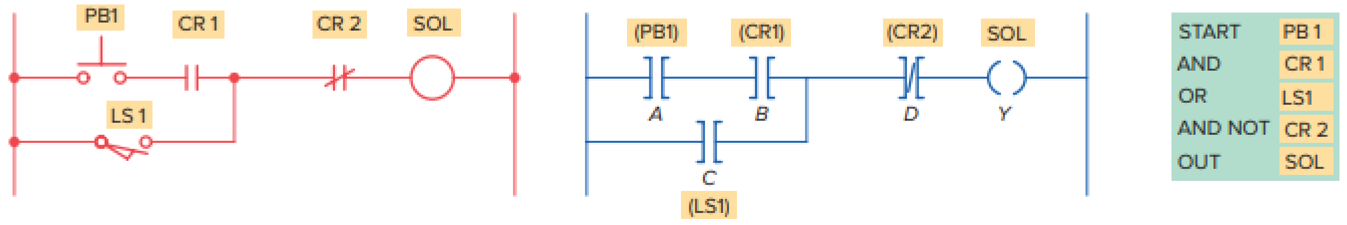


Figure 5-14 Standard IEC 61131 languages associated with PLC programming.



START	PB 1
AND	CR 1
OR	LS1
AND NOT	CR 2
OUT	SOL

(*a*) Hardwired relay control circuit (*b*) Equivalent ladder diagram (LD) program (*c*) Equivalent instruction list (IL) program

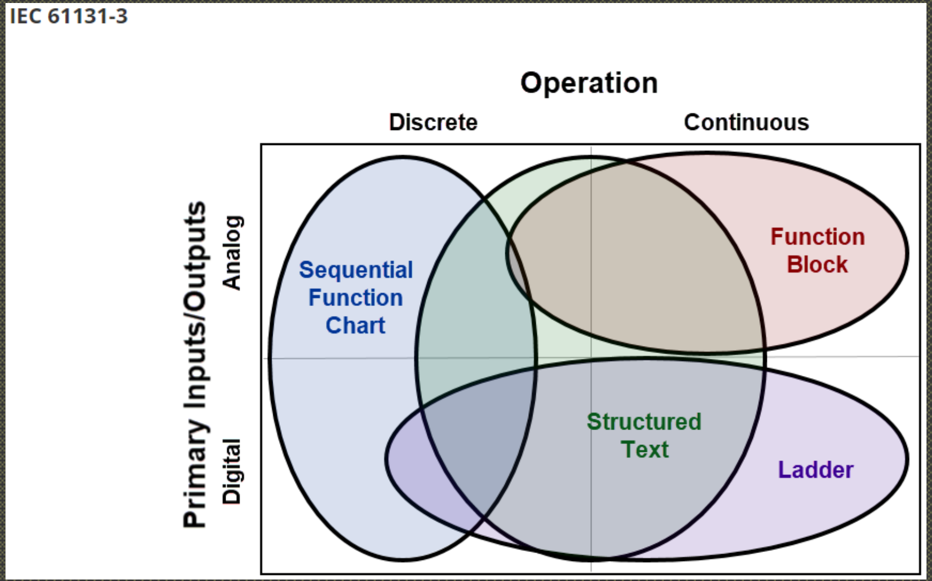
Figure 5-15 Comparison of ladder diagram and instruction list programming.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

Input and Output modules, Discrete and Analog modules

Among the all-PLC types, we can separate 2 groups of Modular and Fixed PLCs.

- The central processor unit (CPU) of all Modular PLCs does not have any input or output terminals. The CPUs are accompanied with input and output modules, communication modules, etc.
- The processors of the Fixed PLCs contain inputs and outputs on their base processor units. However, if necessary, the configuration of the fixed PLC for many models may be expanded by adding input and output modules.



https://www.breen-machine.com/the-future-role-of-ladder-logic-in-automation-control-engineering/

Input and Output DC Modules

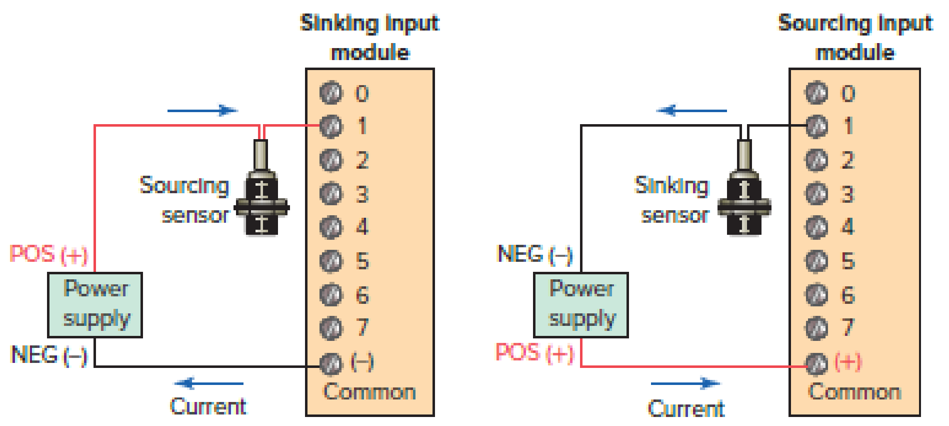


Figure 2-15 Sinking and sourcing inputs.

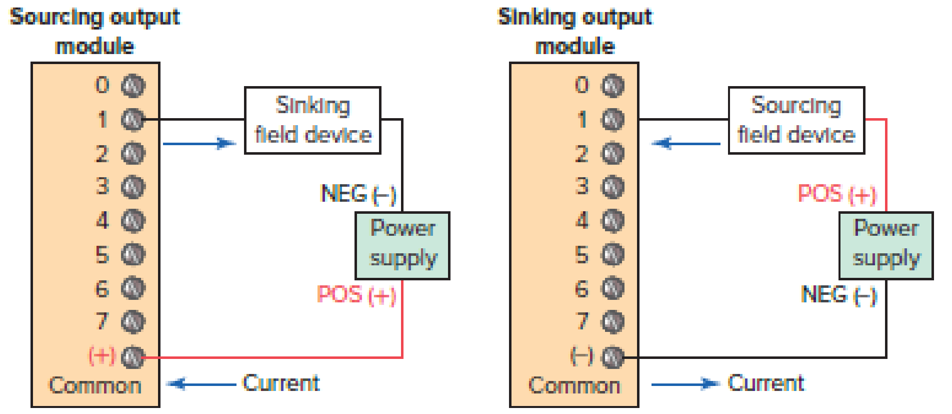


Figure 2-16 Sinking and sourcing outputs.

(Frank D. Petruzella Programmable Logic Controllers 6th edition)

PROGRAMMABLE LOGIC CONTROLLERS

AC INPUTS

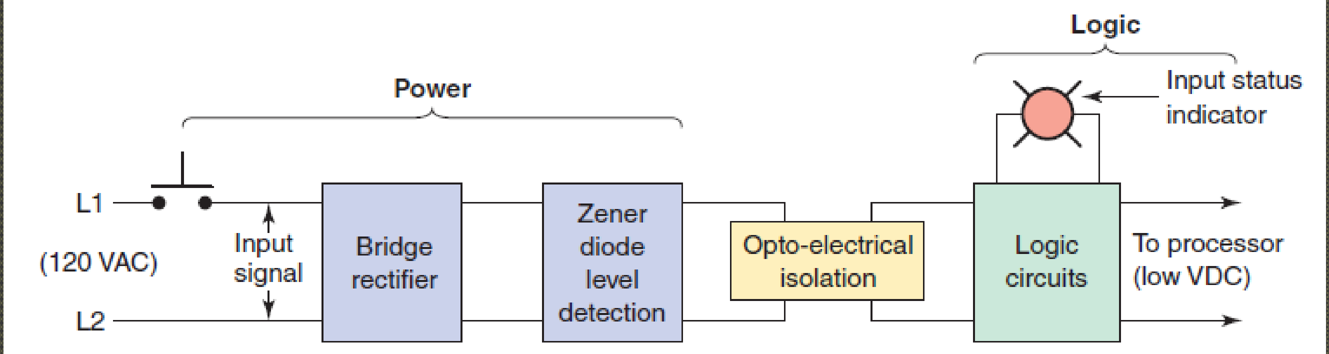


Figure 2-14 Discrete AC input module block diagram.

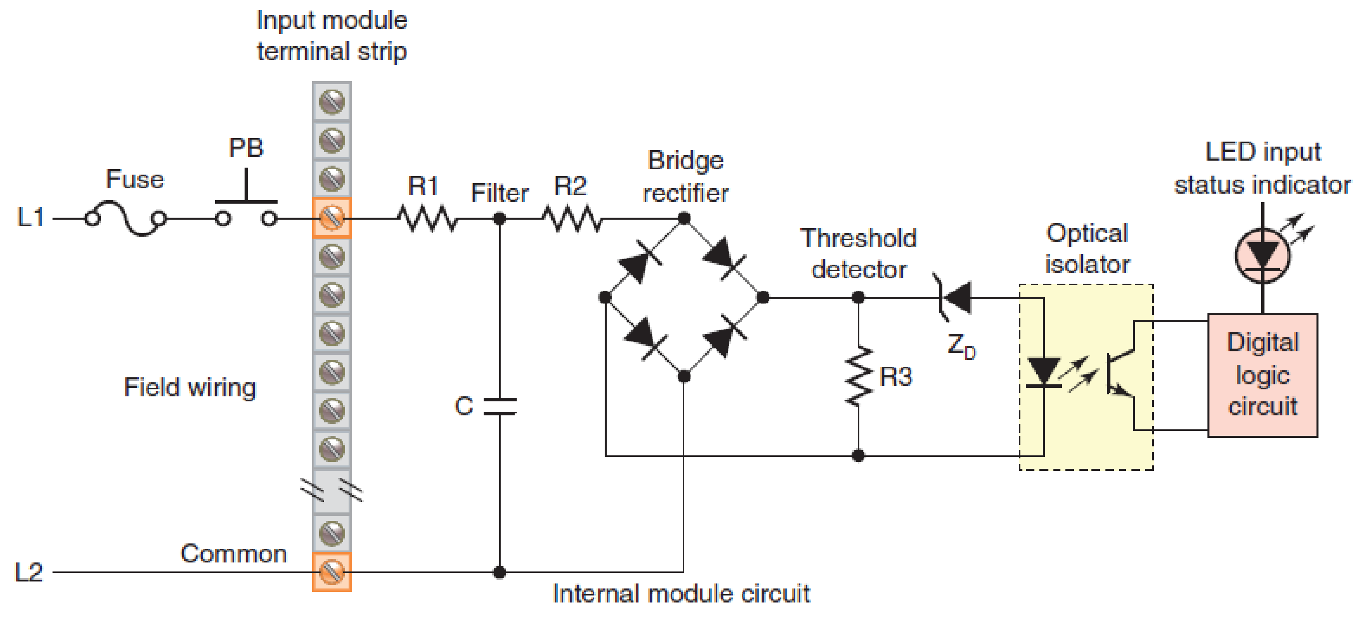


Figure 2-15 Simplified diagram for a single input of a discrete AC input module.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

PROGRAMMABLE LOGIC CONTROLLERS

AC OUTPUTS

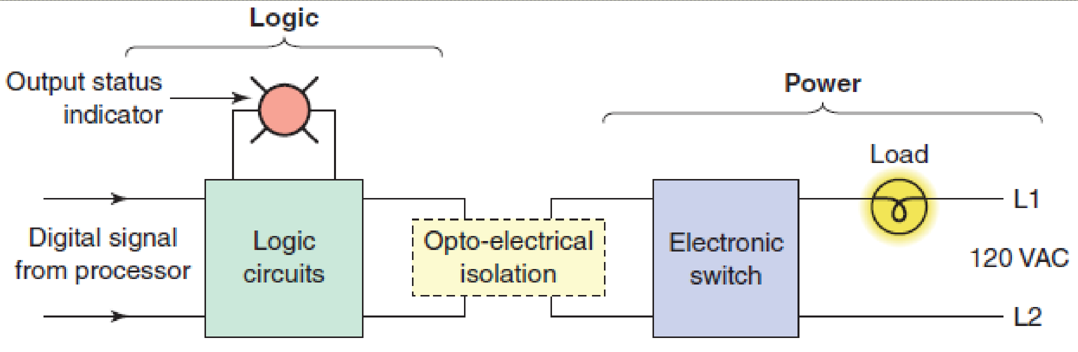


Figure 2-16 Discrete AC output module block diagram.

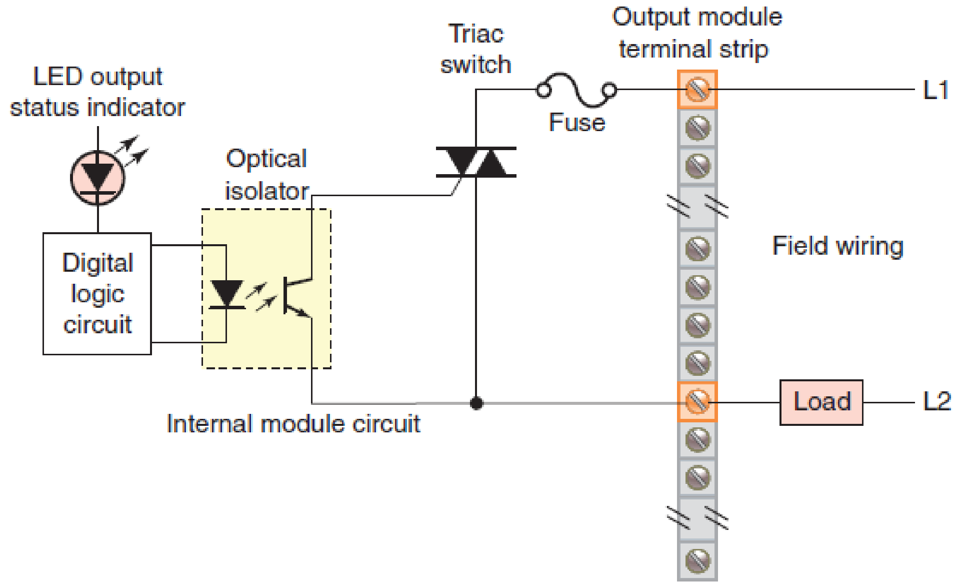


Figure 2-17 Simplified diagram for a single output of a discrete AC output module.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

Memory organization and addressing

The different types of Allen Bradley controllers deal with data in terms of storing and retrieving them according to the memory structure of the controller.

Allen Bradley controllers have two different memory structures; **rack-based fixed memory system** and **tag-based memory system**.

- MicroLogix controllers and SLC 500 series of controllers use a rack-based fixed memory structure.
- Logix controllers use a tag-based system that has a flexible memory structure

https://www.linkedin.com/learning/plc-memory-organization/rack-based-systems

https://www.linkedin.com/learning/plc-memory-organization/tag-based-system

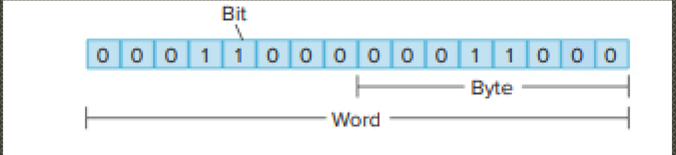


Figure 2-42 Memory bit, byte, and word.

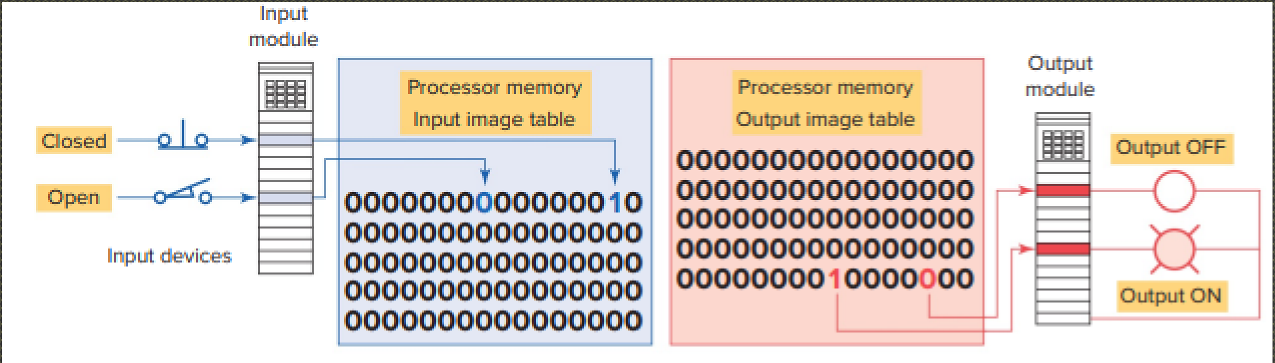
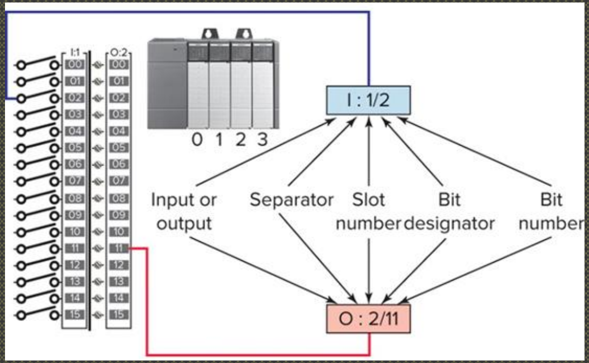


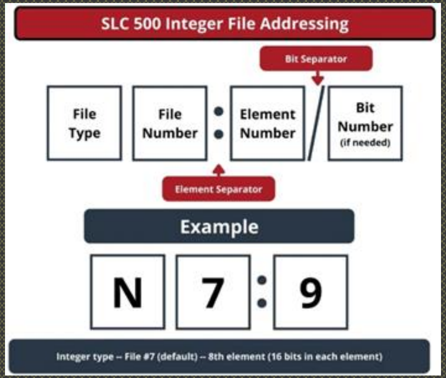
Figure 2-43 Input and output tables.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)



PROGRAMMABLE LOGIC CONTROLLERS

Rack-based Memory System



SLC 500 Integer File Addressing
Bit Separator
File Type
File Number
Element Number
Bit Number (if needed)
Element Separator
Example
N
7
9
Integer type – File #7 (default) – 8th element (16 bits in each element)

OUTPUT - O0 – set of 16-bit register map, representing the actual outputs. The typical address of a bit for RSLogix 500 programming software: O:2/12 means that the second slot after the processor unit is occupied by an output module; 12 – bit #12 which corresponds to the terminal #12 on the unit.

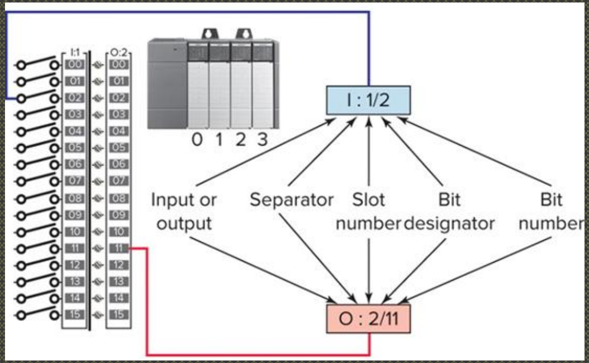
INPUT - I1 – A set of 16-bit register map, representing the actual inputs. Typical address: **I:1/10** can be understood similar to the case of outputs. "**I**" means input.

STATUS - S2 – This file contains information about the processor status or the status of different memory registers and bits. There is no way you can add or delete from the status file.

BINARY - B3 – A set of 16-bit register map. Each bit of the register, as well as the whole registers can be used for programming purposes. The B3 bits are sometimes called internal utility bits. You can use a whole word in the program, as well as separate bits: **B3:3/14** or **B3:0/0.**

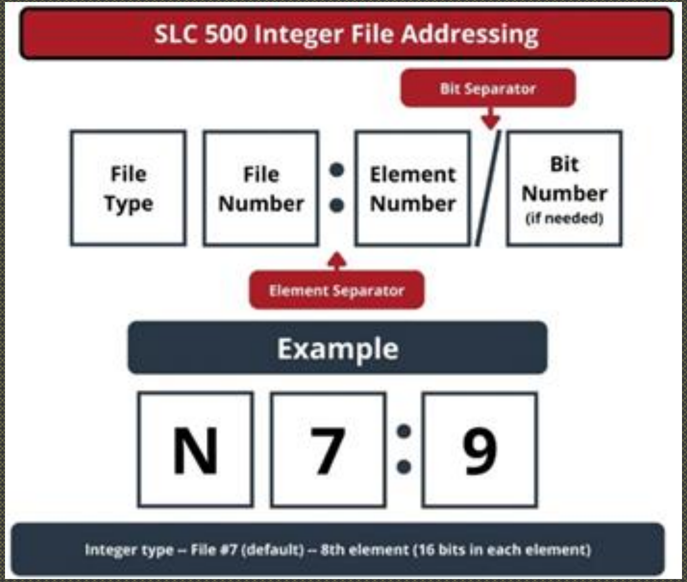
TIMER - T4 – an instruction used in programs, to provide delays or duration of actions. Each timer has a set of three words: Status (specific parameters called status bits), Accumulating, and Preset.

COUNTER - C5 – an instruction used in the program, to provide counting. Each counter also has a set of three words: Status (specific parameters called status bits), Accumulating, and Preset.



PROGRAMMABLE LOGIC CONTROLLERS

Rack-based Memory System


SLC 500 Integer File Addressing
Bit Separator
File Type
File Number
Element Number
Bit Number (if needed)
Element Separator
Example
N
7
:
9
Integer type – File #7 (default) – 8th element (16 bits in each element)

CONTROL - **R6** – the control addresses work as three-word elements (WORD 0, WORD 1 and WORD 2) used in specific instructions. WORD 0 is a status word and contains variety of control bits which are used for programming purpose. WORD 1 contains information about the length of array that is used by specific procedure. WORD 2 indicates the position of the pointer in the array.

INTEGER - **N7** – A set of 16-bit registers containing integer numbers in the range from -32768 to +32767. Maximum number of the words in the N7 file is 256 (counting from 0 – 255). Similar to B3 words, each bit of the N7 word can be used for programming purposes (N7:0/6, N7:13/12, …).

FLOAT - **F8** – These are containers of decimal numbers, either positive or negative. There is one **F8:0** address by default. Maximum number of addresses is 256. Float data cannot be accessed at the bit level.

PROGRAMMABLE LOGIC CONTROLLERS

Rack-based Memory System

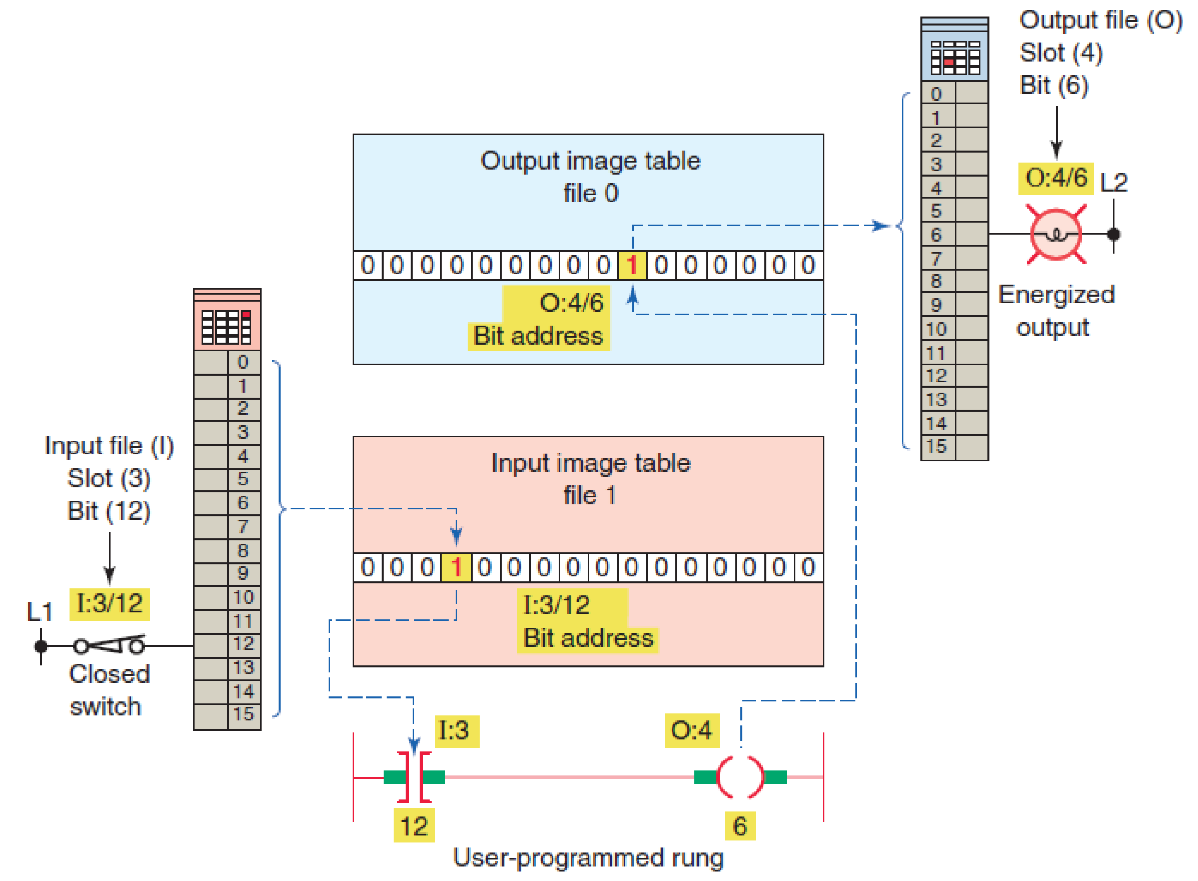
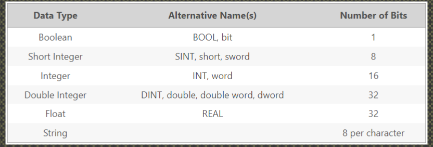


Figure 5-24 Addressing format for an Allen-Bradley SLC 500 controller.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

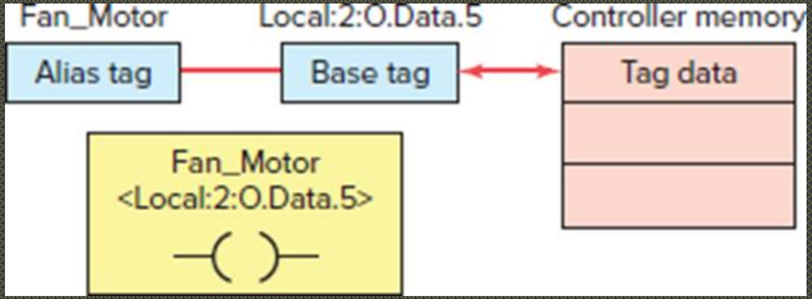
HUMBER

PROGRAMMABLE LOGIC CONTROLLERS



Data Type	Alternative Name(s)	Number of Bits
Boolean	BOOL, bit	1
Short Integer	SINT, short, sword	8
Integer	INT, word	16
Double Integer	DINT, double, double word, dword	32
Float	REAL	32
String		8 per character

Tag-based Memory System



The tag-based memory system is a way of organizing and referencing data in the Programmable Logic Controller (PLC) program using user-defined tags instead of traditional memory addresses.

Tags are user-defined names for variables, data structures, and I/O points. Instead of using memory addresses, tags provide a more intuitive and descriptive way to reference data within the PLC program.

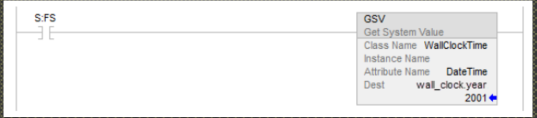
Each tag is associated with a specific data type, such as integers, floats, strings, arrays, or user-defined data structures. This helps define the nature of the information stored in the tag.

Tags are often organized hierarchically to create a structured and easily navigable system. For example, a tag might be organized as follows: Controller1\IO\DigitalInputs\StartButton.

Tags at the controller level may include system parameters, status flags, and configuration settings. These tags are typically associated with the overall operation and health of the PLC.

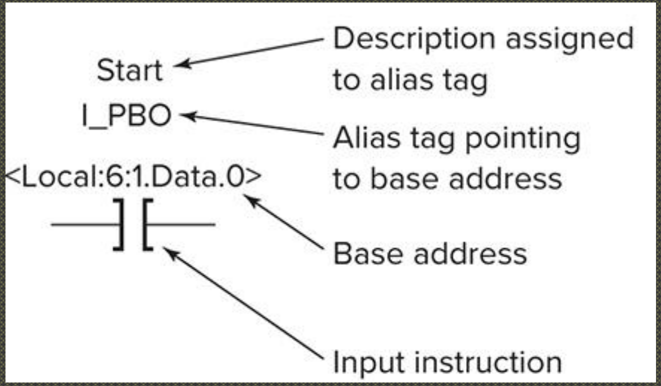
Tags used within the control program are organized based on their functionality. For instance, tags related to timers, counters, and other programmatic elements are grouped together to enhance program clarity and organization.

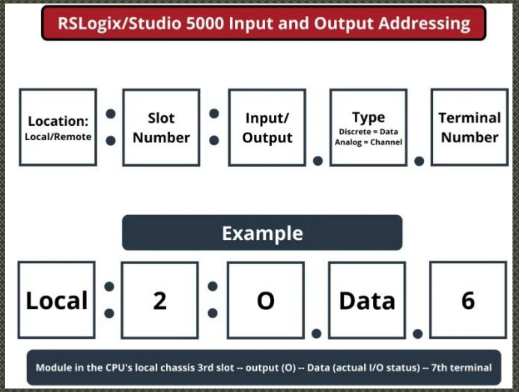
https://control.com/textbook/programmable-logic-controllers/memory-maps-and-i-o-addressing/



PROGRAMMABLE LOGIC CONTROLLERS

Tag-based Memory System


Start
I_PBO
<Local:6:1.Data.0>
Description assigned to alias tag
Alias tag pointing to base address
Base address
Input instruction


RSLogix/Studio 5000 Input and Output Addressing
Location: Local/Remote
:
Slot Number
:
Input/ Output
.
Type Discrete = Data Analog = Channel
.
Terminal Number
Example
Local : 2 : O . Data . 6
Module in the CPU's local chassis 3rd slot -- output (O) -- Data (actual I/O status) -- 7th terminal

Tags are created for interfacing with physical field devices, representing input and output points. These tags are associated with specific I/O modules and channels.

Aliases allow users to create alternative names for tags. This provides flexibility in programming and can make it easier to understand the purpose of each variable, especially when dealing with complex or commonly used tags.

In a tag-based system, data is addressed using the tag names rather than numerical memory addresses. For example, instead of referencing a memory location like M100, you might use a tag like Motor1\Speed.

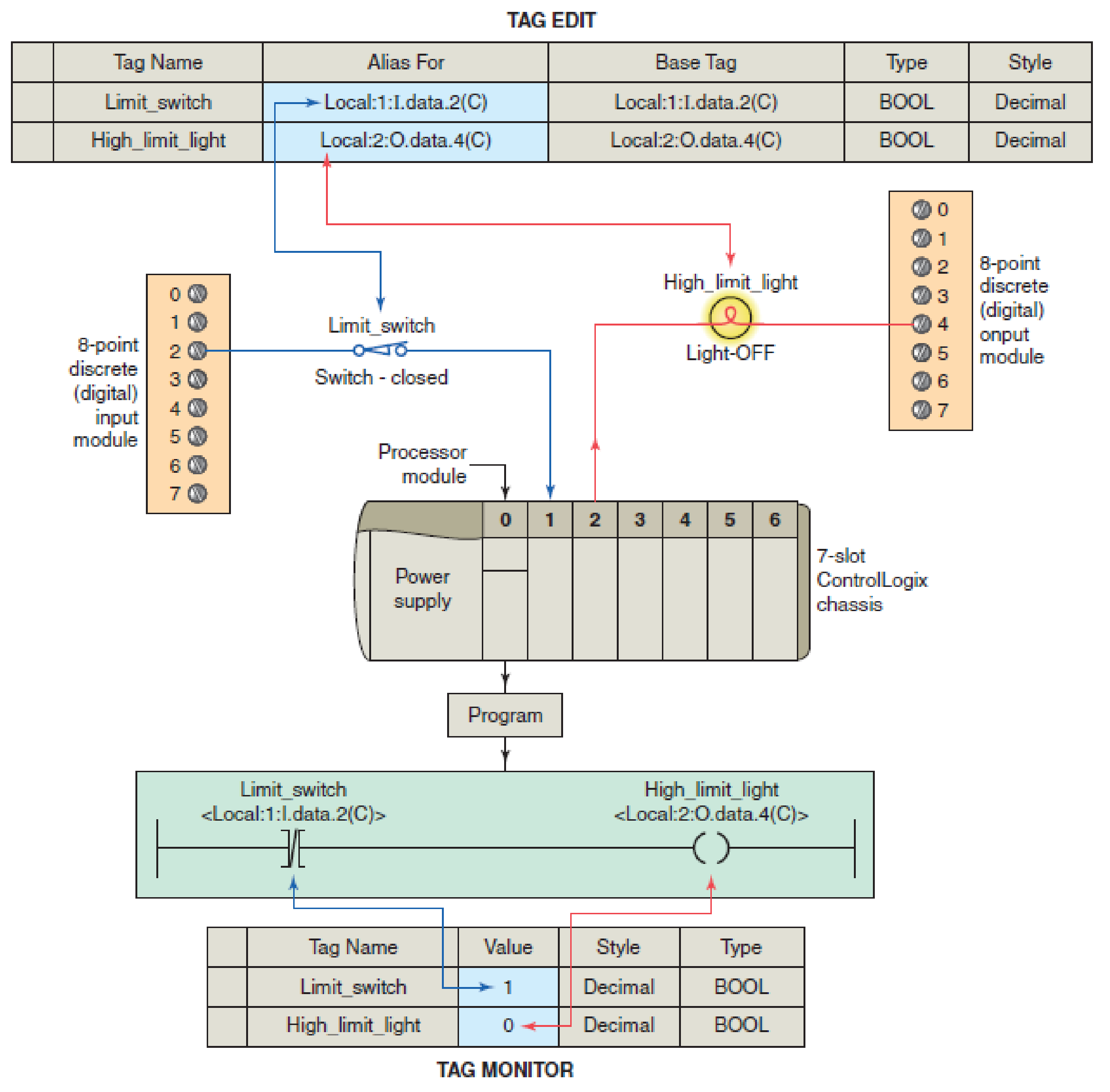
Allen Bradley programming environments allow users to edit and create tags offline. This means you can structure the program before connecting to the physical PLC, providing a convenient way to plan and organize the control logic.

Memory Allocation: The allocation and management of memory are handled by the controller firmware based on the tag definitions. Users are abstracted from the underlying memory details, focusing on the logical organization of data.

https://control.com/textbook/programmable-logic-controllers/memory-maps-and-i-o-addressing/

PROGRAMMABLE LOGIC CONTROLLERS

Tag-Based Memory System



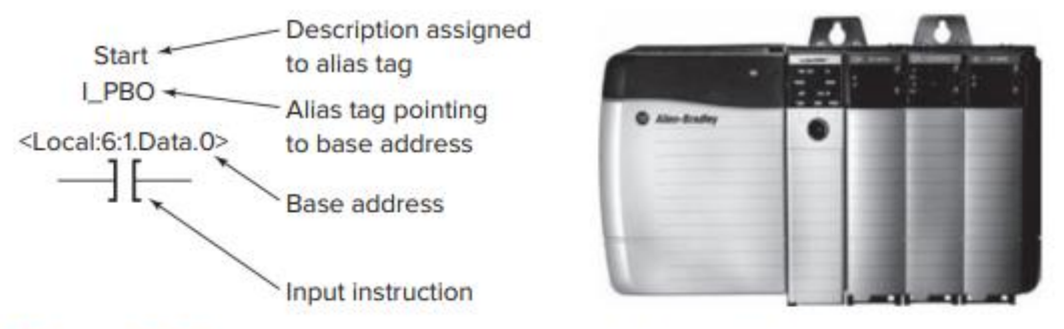
TAG EDIT

	Tag Name	Alias For	Base Tag	Type	Style
	Limit_switch	Local:1:I.data.2(C)	Local:1:I.data.2(C)	BOOL	Decimal
	High_limit_light	Local:2:O.data.4(C)	Local:2:O.data.4(C)	BOOL	Decimal

TAG MONITOR

	Tag Name	Value	Style	Type
	Limit_switch	1	Decimal	BOOL
	High_limit_light	0	Decimal	BOOL

Figure 15-30 Tag-based address implementation.



Start
I_PBO
<Local:6:1.Data.0>
Description assigned to alias tag
Alias tag pointing to base address
Base address
Input instruction

Figure 2-5 Allen-Bradley ControlLogix tag-based addressing format.
Source: Image Courtesy of Rockwell Automation, Inc.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

HUMBER

PROGRAMMABLE LOGIC CONTROLLERS

PLC processor Scan

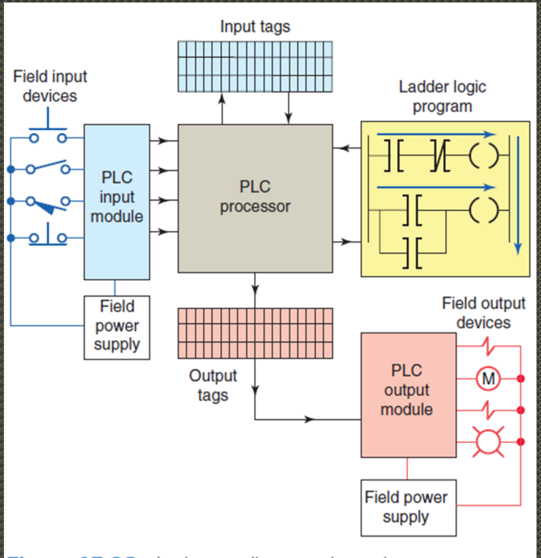


Figure 15-26 Logix controller operating cycle.

The program scan in a ControlLogix PLC (Programmable Logic Controller) refers to the cyclic execution of the control program stored in the controller's memory.

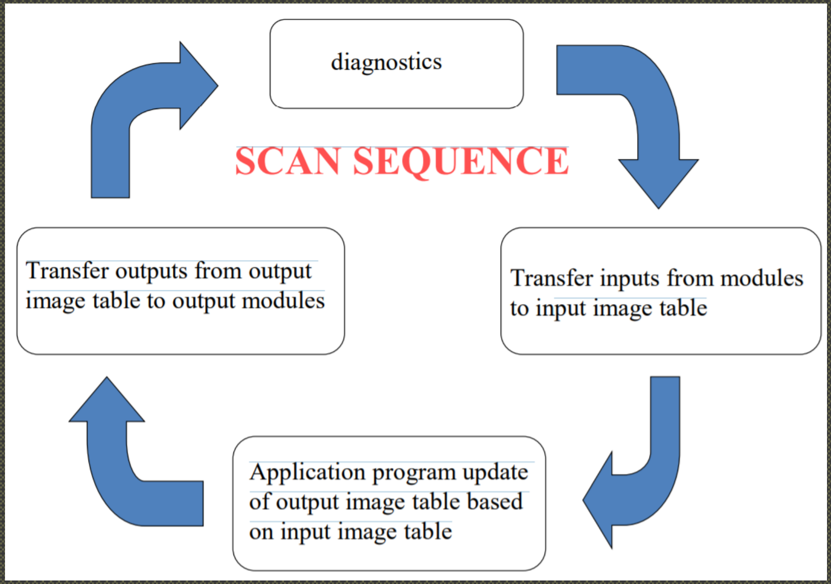
- The program scan begins with the initiation of a scan cycle, which is recurrent. It is triggered based on the controller's scan time, which is the time it takes to complete one full scan of the control program.
- The first phase of the program scan involves reading the status of input devices. The PLC checks the state of digital and analog input modules, updating the values of corresponding tags in memory.
- The control program, constructed using a tag-based language like ladder logic, is executed by the PLC processor.
- Based on the evaluation of the control program, the PLC determines the state of the output devices.

HUMBER

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

PROGRAMMABLE LOGIC CONTROLLERS

PLC processor Scan


diagnostics
SCAN SEQUENCE
Transfer inputs from modules to input image table
Application program update of output image table based on input image table
Transfer outputs from output image table to output modules

- Throughout the program scan, tag values are updated in memory based on the current state of inputs, outputs, and any internal calculations performed by the program.

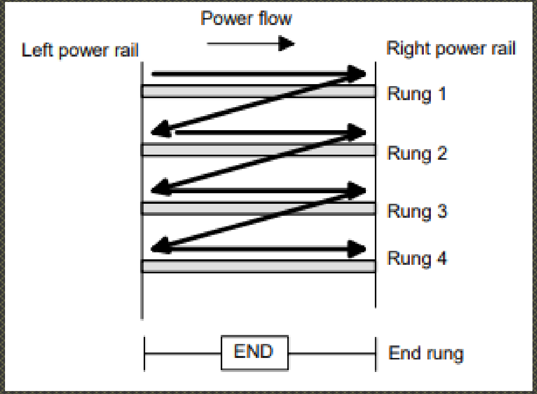
- ControlLogix PLCs often feature communication modules that facilitate data exchange with other devices on the network. During the program scan, PLC communication tasks may occur.

- Once all phases of the program scan are executed, the scan cycle is considered complete. The PLC then starts a new cycle, repeating the process to continually monitor and control the connected system.

- The scan time is crucial in PLC applications, and efforts are often made to optimize the control program to ensure fast and efficient execution. This includes managing the complexity of the program, minimizing scan time, and optimizing the use of memory and resources.

PROGRAMMABLE LOGIC CONTROLLERS

PLC processor Scan



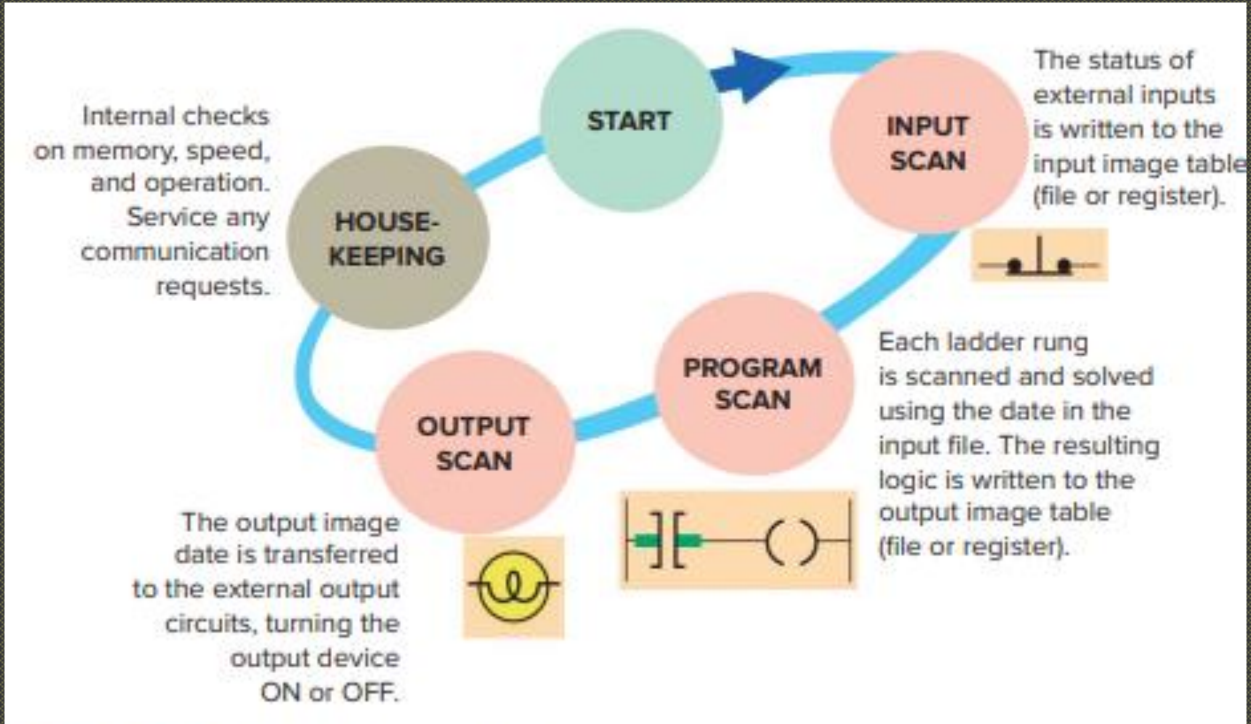

START
INPUT SCAN
The status of external inputs is written to the input image table (file or register).
HOUSE-KEEPING
Internal checks on memory, speed, and operation. Service any communication requests.
PROGRAM SCAN
Each ladder rung is scanned and solved using the date in the input file. The resulting logic is written to the output image table (file or register).
OUTPUT SCAN
The output image date is transferred to the external output circuits, turning the output device ON or OFF.

Figure 5-8 PLC program scan cycle.

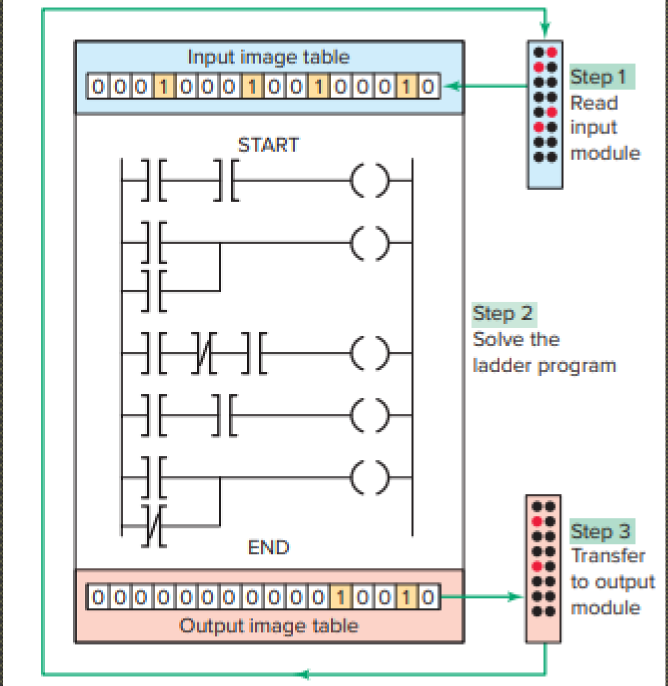


Figure 5-12 Scan process applied to a multiple rung program.

PLC mixes scan-based and event-driven program execution, whereas PAC software is typically event-driven.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

PROGRAMMABLE LOGIC CONTROLLERS

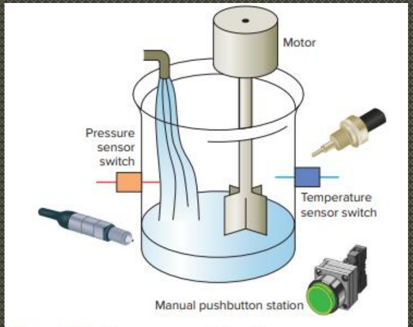


Figure 1-12 Mixer process control problem.

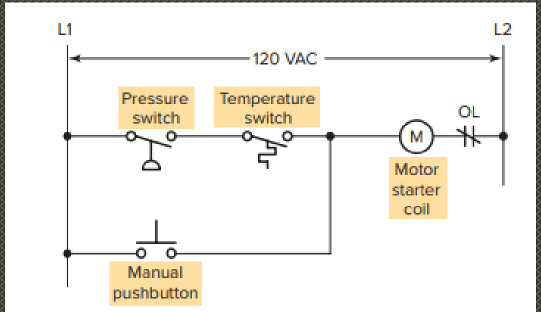


Figure 1-13 Process control relay ladder diagram.

- A mixer motor is to be used to automatically stir the liquid in a vat when the temperature and pressure reach preset values.
- In addition, direct manual operation of the motor is provided by means of a separate pushbutton station.
- The process is monitored with temperature and pressure sensor switches that close their respective contacts when conditions reach their preset values.

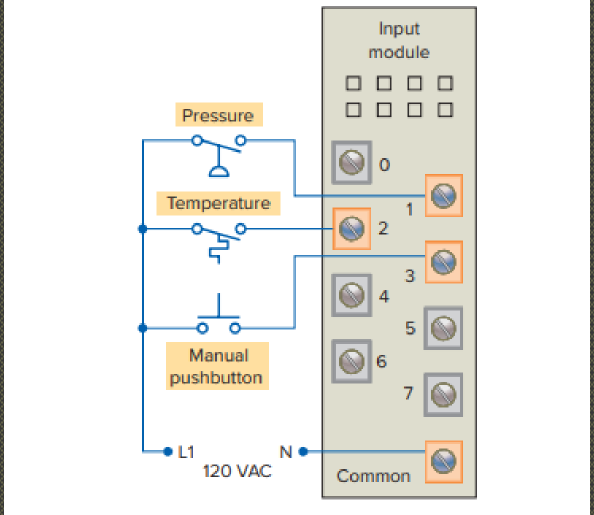


Figure 1-14 Typical wiring connections for a 120 VAC modular configured input module.

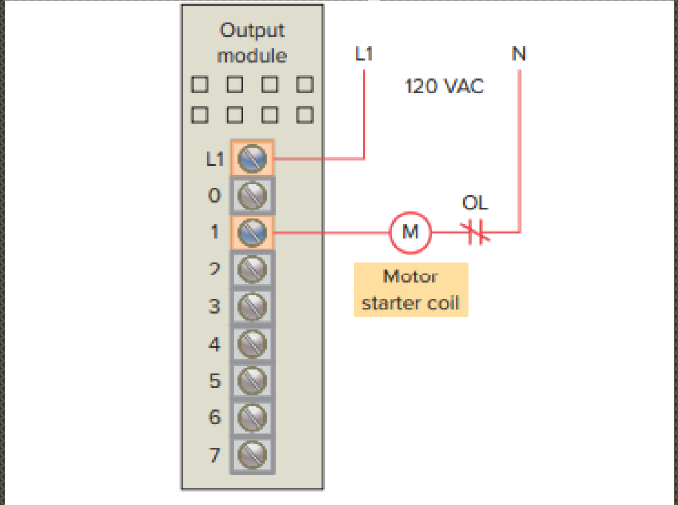


Figure 1-15 Typical wiring connections for a 120 VAC modular configured output module.

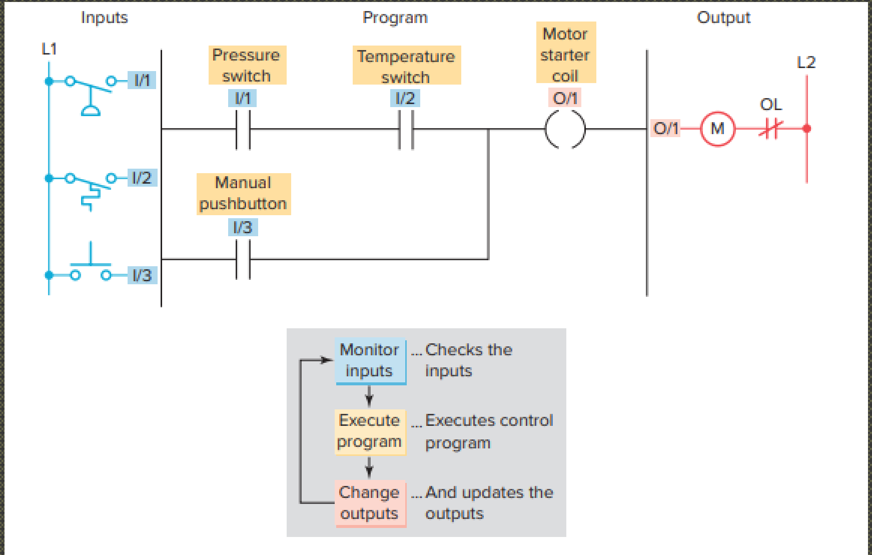


Figure 1-16 Process control PLC ladder logic program with typical addressing scheme.

(Frank D. Petruzella Programmable Logic Controllers 6th edition)

PROGRAMMABLE LOGIC CONTROLLERS

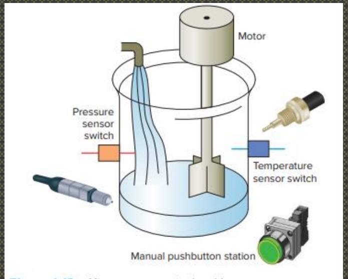


Figure 1-12 Mixer process control problem.

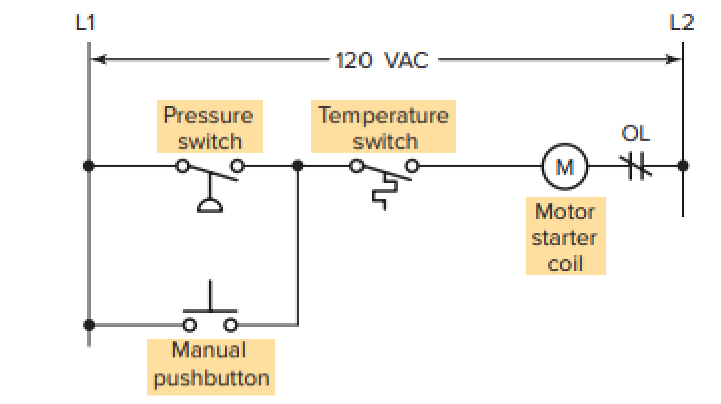


Figure 1-18 Relay ladder diagram for the modified process.

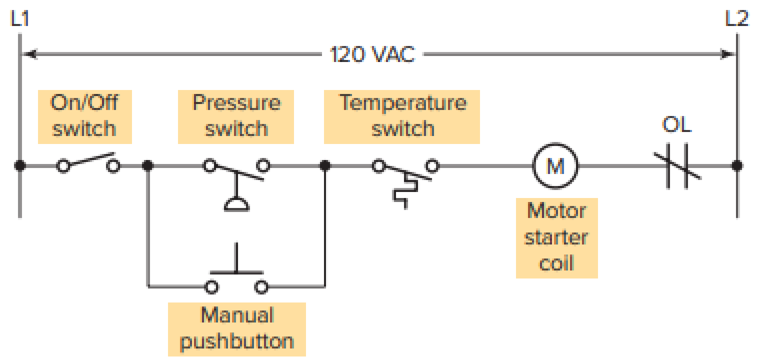


Figure 1-20 Modified relay ladder for the addition of an ON/OFF switch.

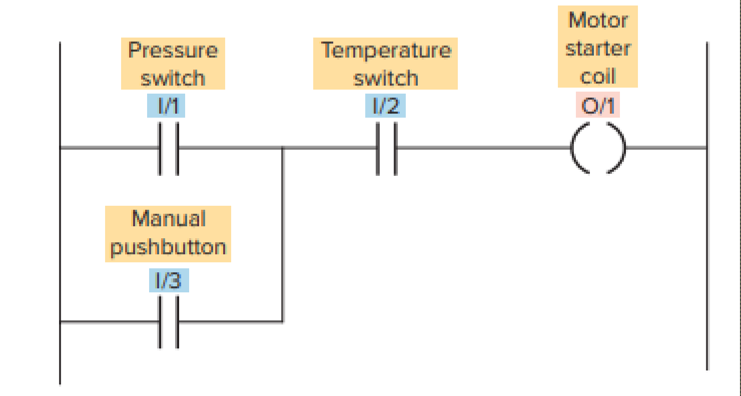


Figure 1-19 PLC ladder logic program for the modified process.

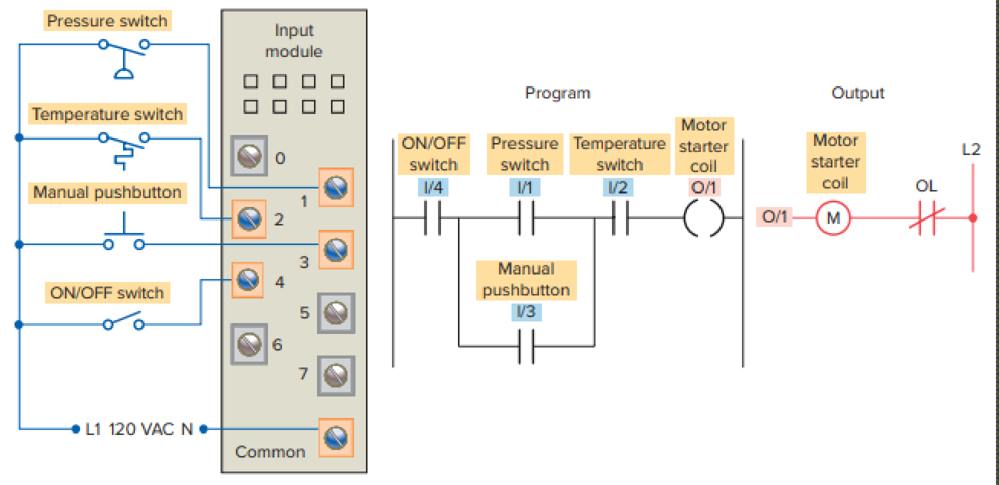


Figure 1-21 Modified PLC ladder logic program and ON/OFF switch module connection.

HUMBER

(Frank D. Petruzella Programmable Logic Controllers 6th edition)

PROGRAMMABLE LOGIC CONTROLLERS

Erickson, K. (2016) Programmable logic controllers: An emphasis on design and application (3rd edition). Rolla MO: Dogwood Valley Press.

Chapter 1
Chapter 2.1- 2.5
Chapter 3.1-3.4
Chapter 4.1 – 4.3

HUMBER

Thank you!

Discussions?