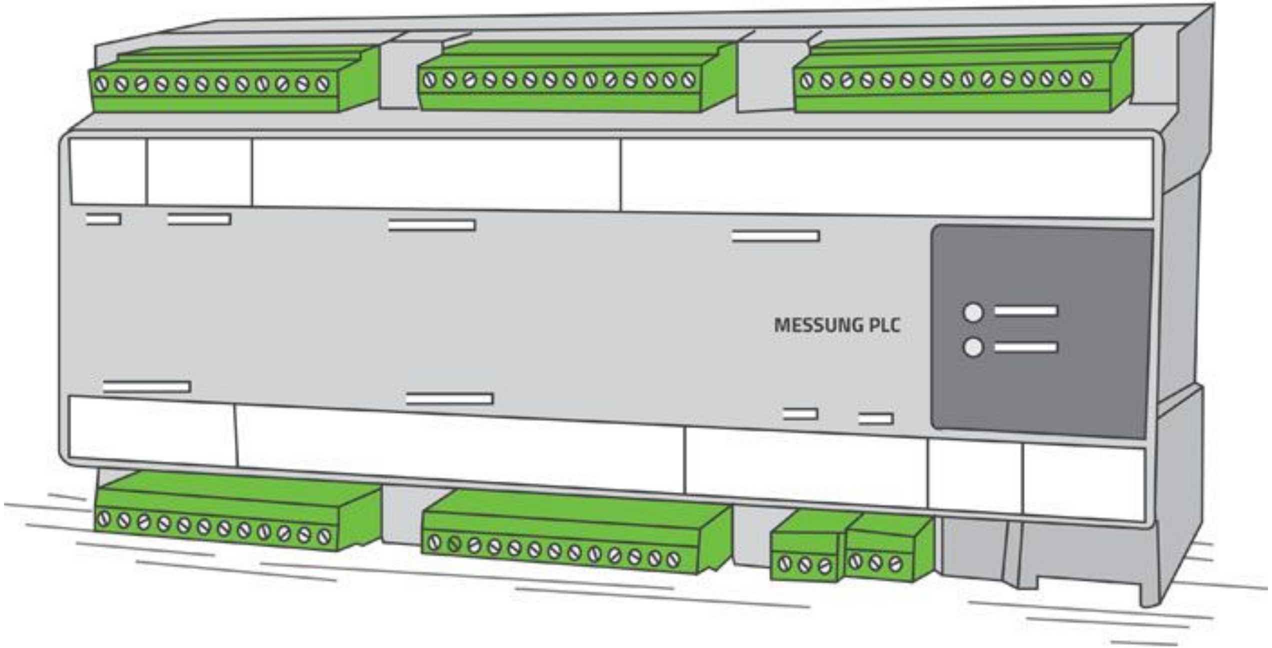
SY486K MICS
Lecture 5

Introduction to PLCs

CDR Brien Croteau, USNA Cyber Science Department, February 2023

Outline

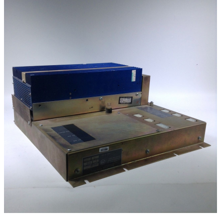
- Overview
- History
- Components
- Applications
- SCADA Organization
- Programming



What is a PLC?

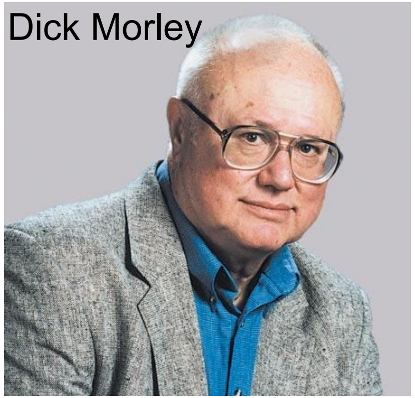
A programmable logic controller (PLC) or programmable controller is an industrial computer that has been ruggedized and adapted for the control of manufacturing processes, such as assembly lines, machines, robotic devices, or any activity that requires high reliability, ease of programming, and process fault diagnosis.

History

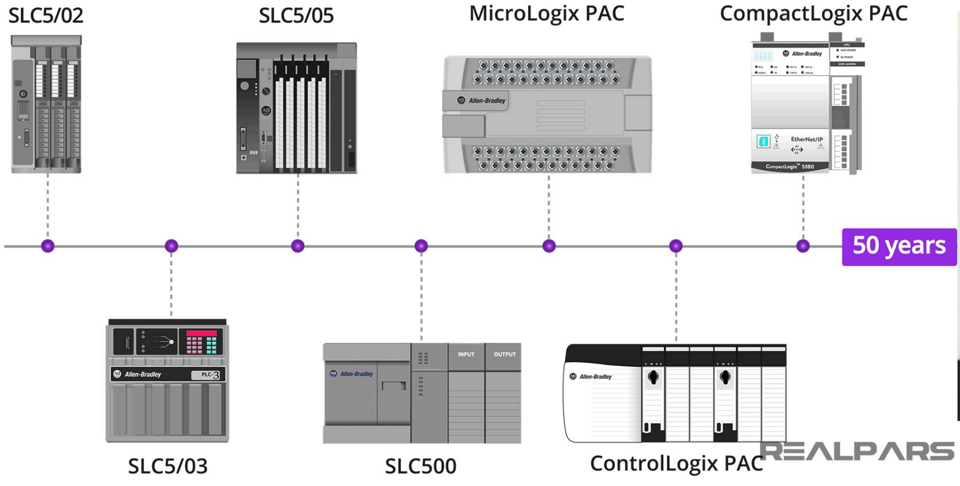


There are two men credited as being the "father" of the PLC.

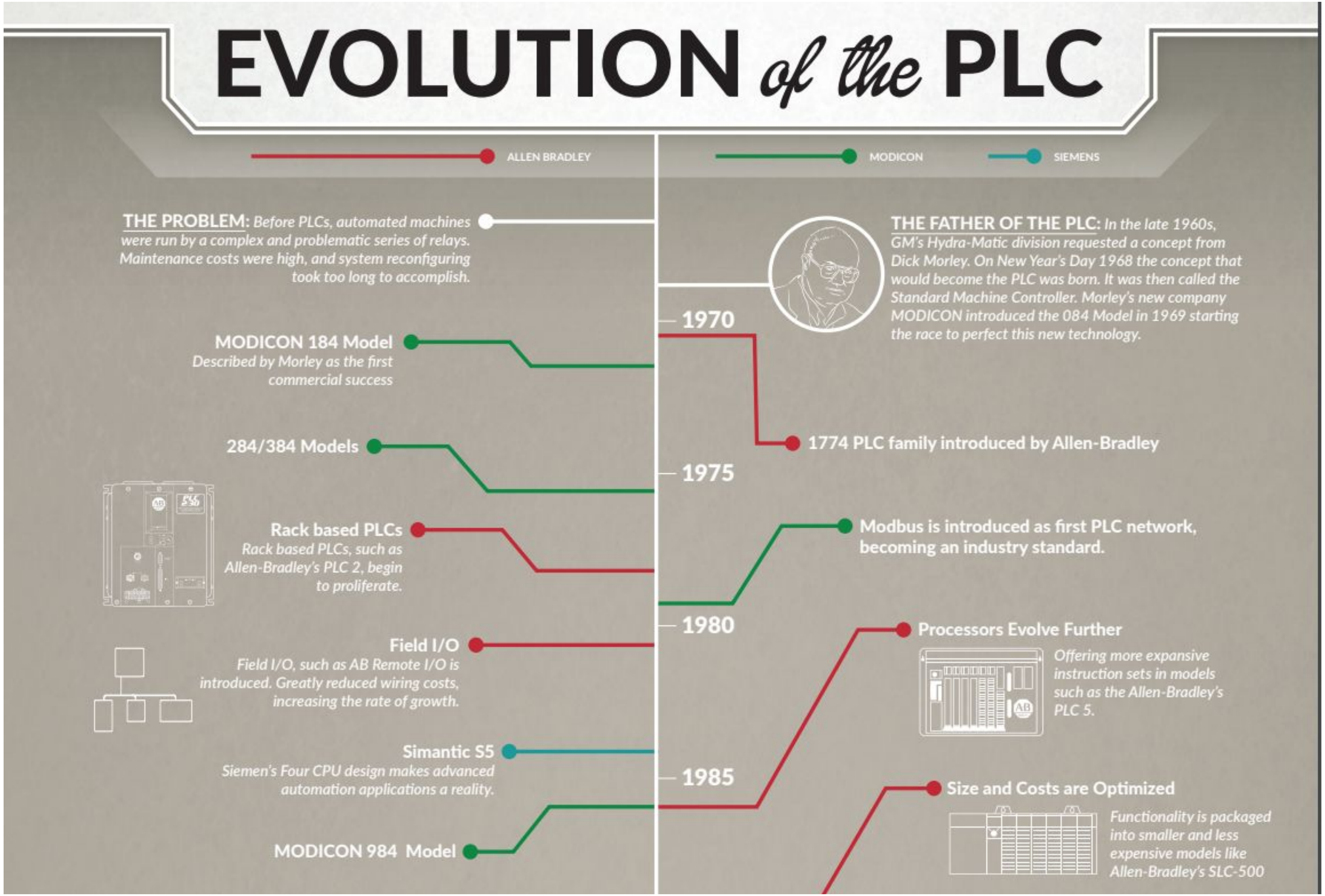
- Richard E. Morley (1932-2017) was an American mechanical engineer who was involved with the production of the first PLC for General Motors, Modicon, and Bedford Associates in 1968.
- Odo Josef Struger (1931-1998) was involved in the invention of the Allen-Bradley programmable logic controller (PLC) and coined that term, during 1958 to 1960 based on a concept developed in his doctoral dissertation at the Vienna University of Technology.



MODICON
Schneider Electric



Allen-Bradley
by ROCKWELL AUTOMATION



EVOLUTION of the PLC

🟥 ALLEN BRADLEY 🟩 MODICON 🟦 SIEMENS

THE PROBLEM: *Before PLCs, automated machines were run by a complex and problematic series of relays. Maintenance costs were high, and system reconfiguring took too long to accomplish.*

THE FATHER OF THE PLC: *In the late 1960s, GM's Hydra-Matic division requested a concept from Dick Morley. On New Year's Day 1968 the concept that would become the PLC was born. It was then called the Standard Machine Controller. Morley's new company MODICON introduced the 084 Model in 1969 starting the race to perfect this new technology.*

1970

MODICON 184 Model
Described by Morley as the first commercial success

1774 PLC family introduced by Allen-Bradley

284/384 Models

1975

Rack based PLCs
Rack based PLCs, such as Allen-Bradley's PLC 2, begin to proliferate.

Modbus is introduced as first PLC network, becoming an industry standard.

1980

Processors Evolve Further
Offering more expansive instruction sets in models such as the Allen-Bradley's PLC 5.

Field I/O
Field I/O, such as AB Remote I/O is introduced. Greatly reduced wiring costs, increasing the rate of growth.

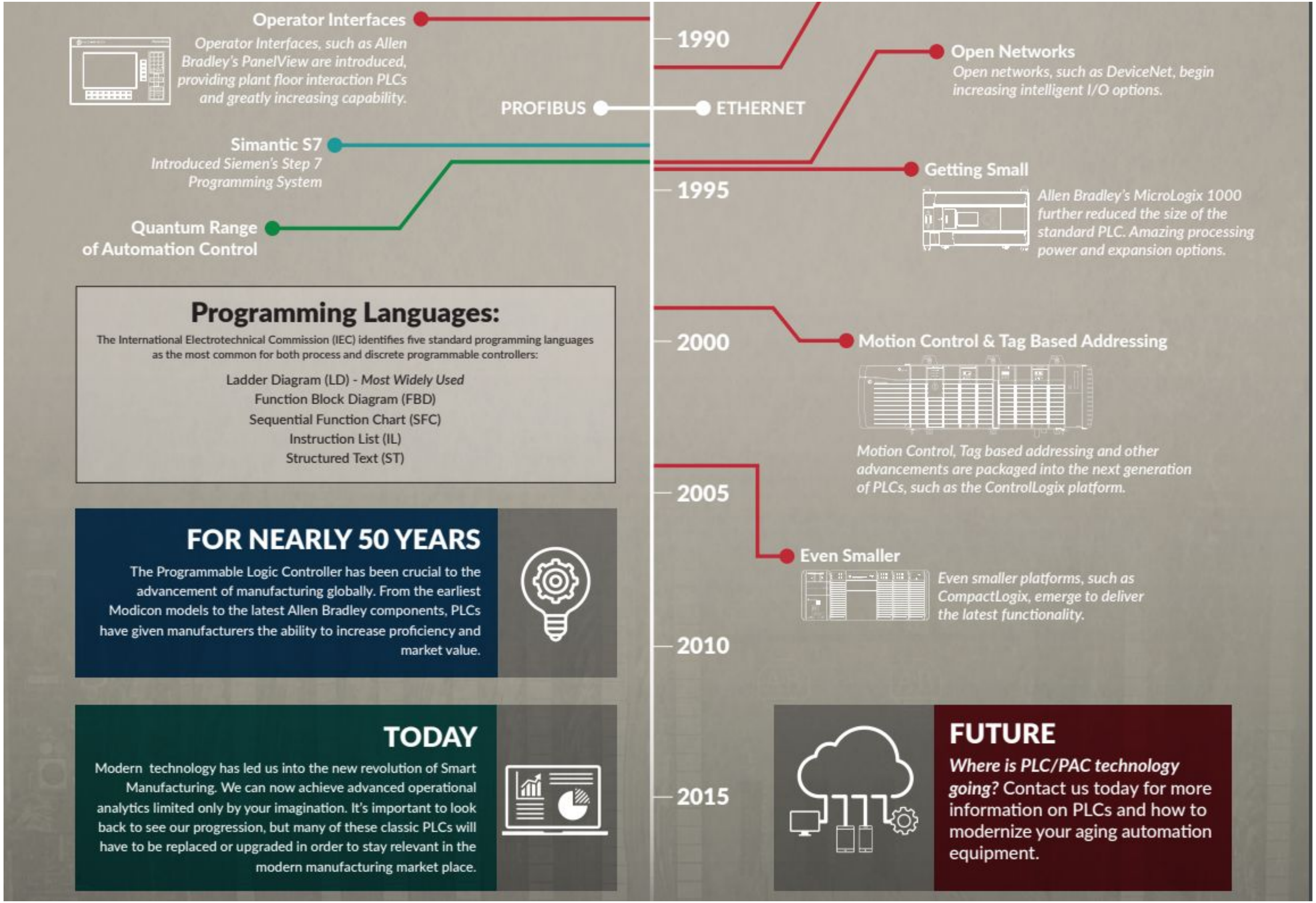
Simantic S5
Siemen's Four CPU design makes advanced automation applications a reality.

1985

Size and Costs are Optimized
Functionality is packaged into smaller and less expensive models like Allen-Bradley's SLC-500

MODICON 984 Model

https://eecoonline.com/inspire/evolution-of-the-plc-a-historic-timeline



Operator Interfaces

Operator Interfaces, such as Allen Bradley's PanelView are introduced, providing plant floor interaction PLCs and greatly increasing capability.

PROFIBUS

Simantic S7

Introduced Siemen's Step 7 Programming System

Quantum Range of Automation Control

Programming Languages:

The International Electrotechnical Commission (IEC) identifies five standard programming languages as the most common for both process and discrete programmable controllers:

Ladder Diagram (LD) - *Most Widely Used*
Function Block Diagram (FBD)
Sequential Function Chart (SFC)
Instruction List (IL)
Structured Text (ST)

FOR NEARLY 50 YEARS

The Programmable Logic Controller has been crucial to the advancement of manufacturing globally. From the earliest Modicon models to the latest Allen Bradley components, PLCs have given manufacturers the ability to increase proficiency and market value.

TODAY

Modern technology has led us into the new revolution of Smart Manufacturing. We can now achieve advanced operational analytics limited only by your imagination. It's important to look back to see our progression, but many of these classic PLCs will have to be replaced or upgraded in order to stay relevant in the modern manufacturing market place.

1990

Open Networks

Open networks, such as DeviceNet, begin increasing intelligent I/O options.

ETHERNET

Getting Small

Allen Bradley's MicroLogix 1000 further reduced the size of the standard PLC. Amazing processing power and expansion options.

1995

2000

Motion Control & Tag Based Addressing

Motion Control, Tag based addressing and other advancements are packaged into the next generation of PLCs, such as the ControlLogix platform.

2005

Even Smaller

Even smaller platforms, such as CompactLogix, emerge to deliver the latest functionality.

2010

2015

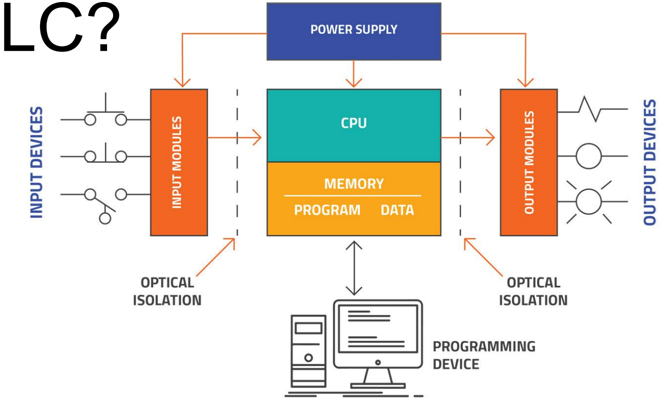
FUTURE

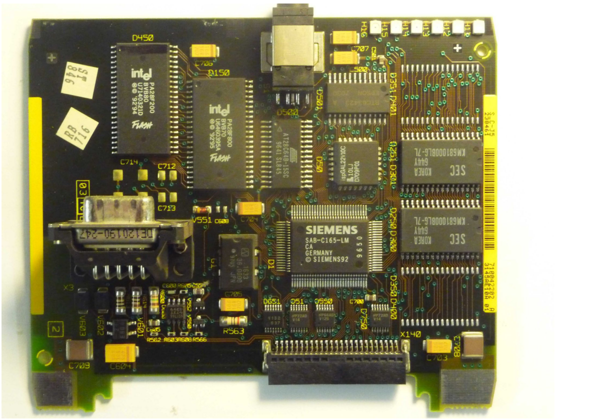
Where is PLC/PAC technology going? Contact us today for more information on PLCs and how to modernize your aging automation equipment.

https://eecoonline.com/inspire/evolution-of-the-plc-a-historic-timeline

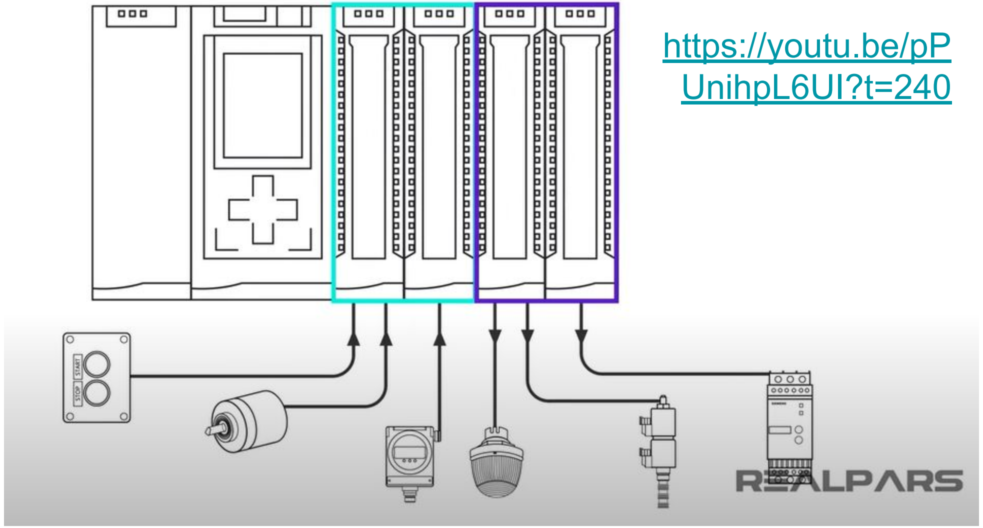
What is inside a PLC?

- Power Supply
- Processor
- Input Modules
- Output Modules
- Interface Modules
- Programming Interface

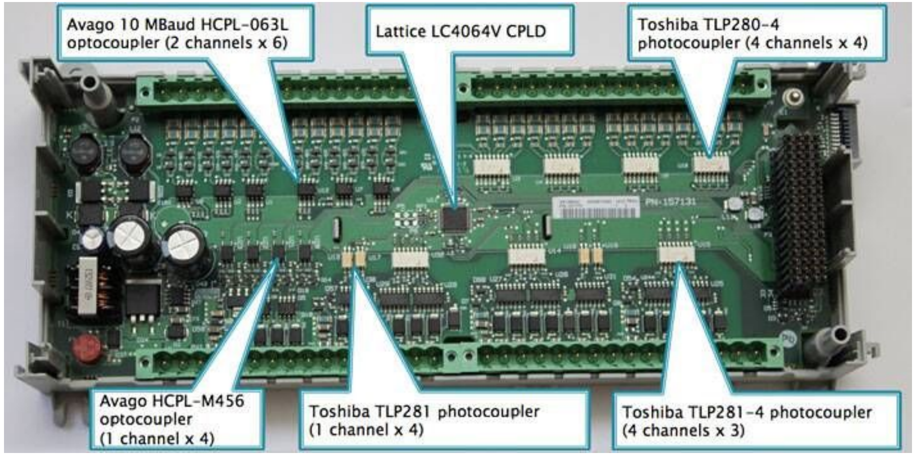




SIEMENS CPU314IFM with C165 series processor



https://youtu.be/pPUnihpL6UI?t=240



Avago 10 MBaud HCPL-063L optocoupler (2 channels x 6)
Lattice LC4064V CPLD
Toshiba TLP280-4 photocoupler (4 channels x 4)
Avago HCPL-M456 optocoupler (1 channel x 4)
Toshiba TLP281 photocoupler (1 channel x 4)
Toshiba TLP281-4 photocoupler (4 channels x 3)

Simple PLC Example

Figure 8 shows control of a manufacturing process being performed by a PLC over a fieldbus network. The PLC is accessible via a programming interface located on an engineering workstation, and data is stored in a data historian, all connected on a LAN.

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r3.ipd.pdf

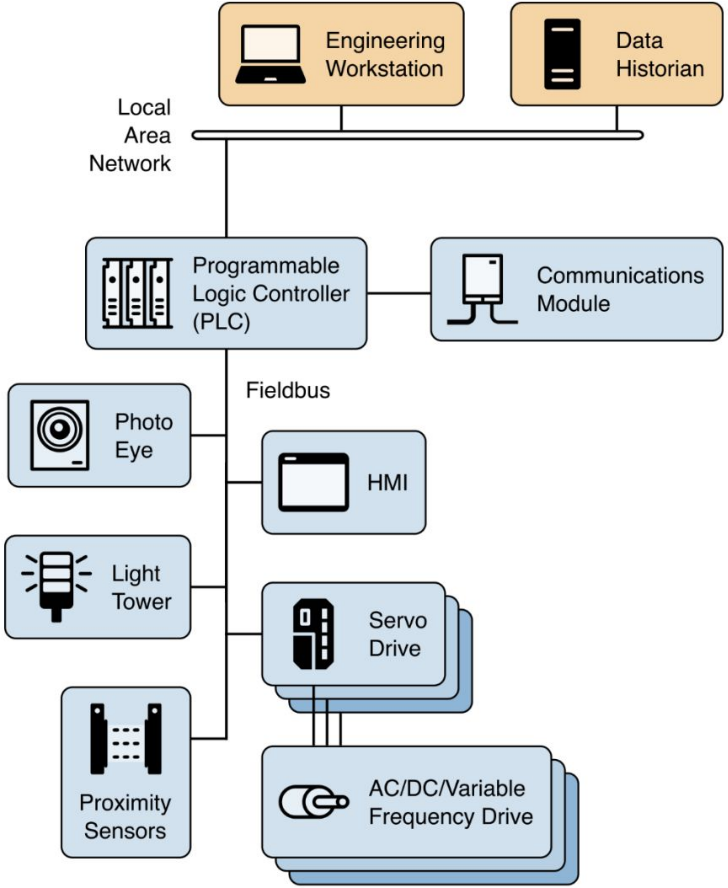
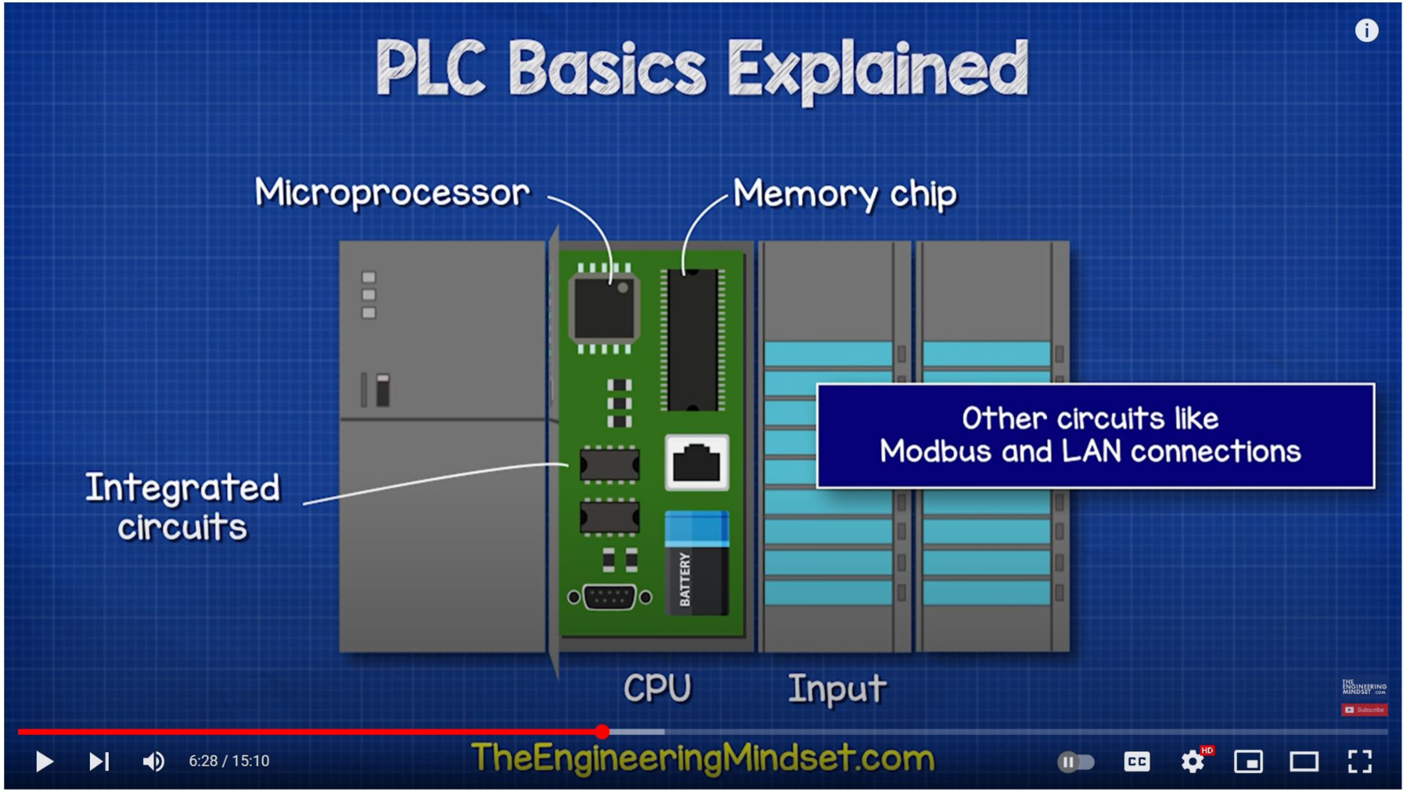


Figure 8: A PLC control system implementation example

PLC Example Applications

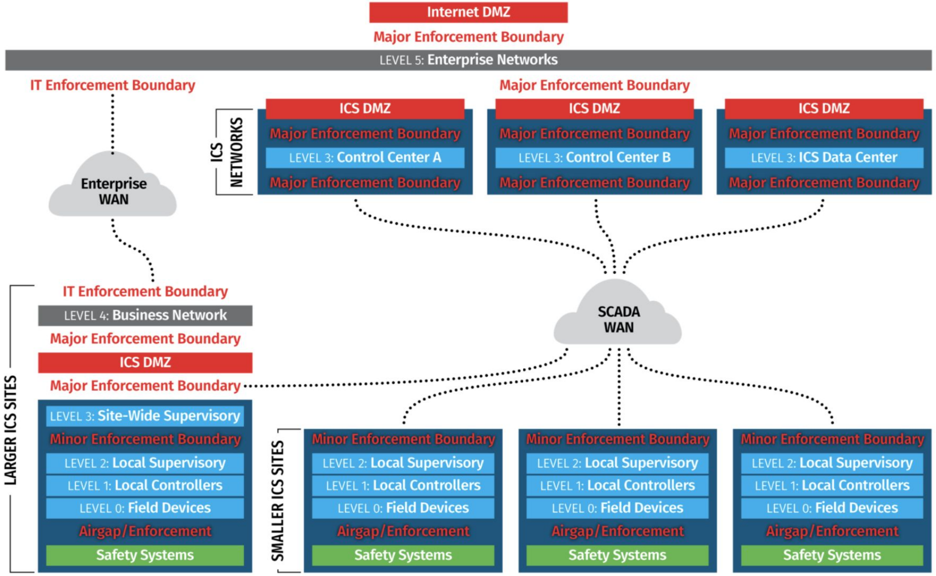
https://youtu.be/uOtdWHMKhnw?t=66

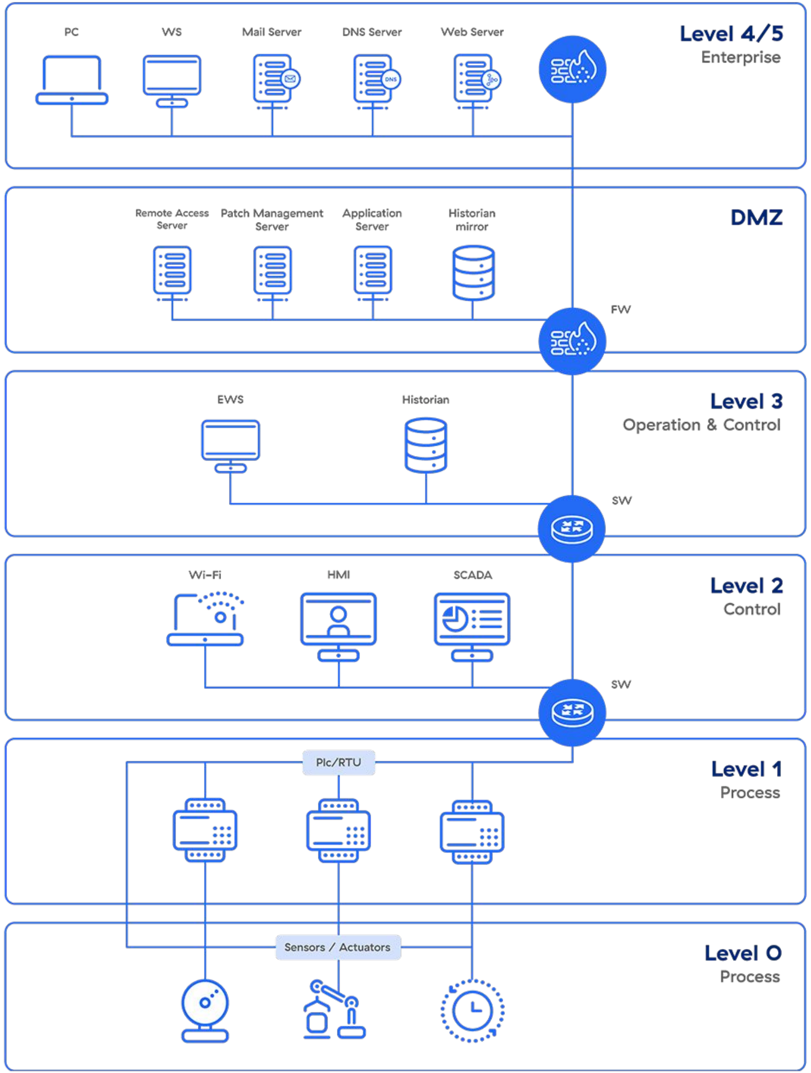


Programable Logic Controller Basics Explained - automation engineering

Where they fit into a SCADA system

The Purdue model, part of the Purdue Enterprise Reference Architecture (PERA), was designed as a reference model for data flows in computer-integrated manufacturing (CIM), where a plant's processes are completely automated. It came to define the standard for building an ICS network architecture in a way that supports OT security, separating the layers of the network to maintain a hierarchical flow of data between them.

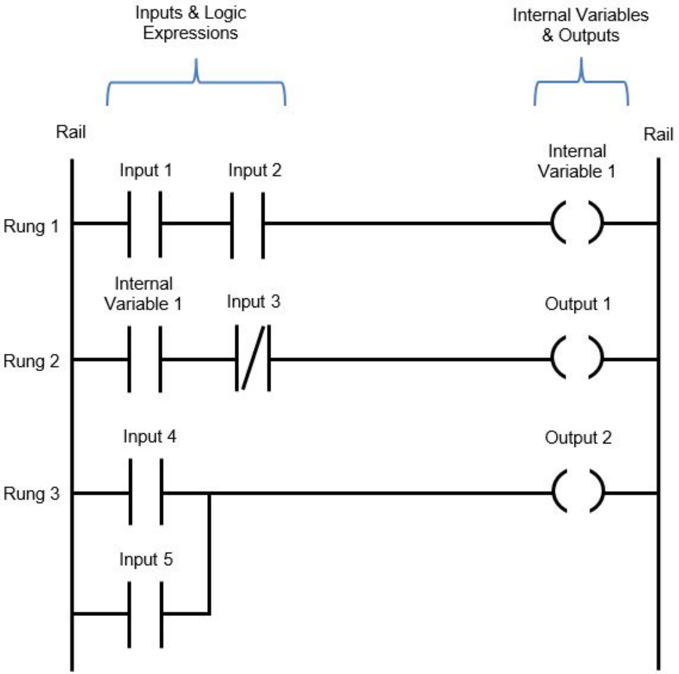

Internet DMZ
Major Enforcement Boundary
LEVEL 5: Enterprise Networks
IT Enforcement Boundary
Major Enforcement Boundary
Enterprise WAN
ICS NETWORKS
ICS DMZ
Major Enforcement Boundary
LEVEL 3: Control Center A
Major Enforcement Boundary
ICS DMZ
Major Enforcement Boundary
LEVEL 3: Control Center B
Major Enforcement Boundary
ICS DMZ
Major Enforcement Boundary
LEVEL 3: ICS Data Center
Major Enforcement Boundary
SCADA WAN
LARGER ICS SITES
IT Enforcement Boundary
LEVEL 4: Business Network
Major Enforcement Boundary
ICS DMZ
Major Enforcement Boundary
LEVEL 3: Site-Wide Supervisory
Minor Enforcement Boundary
LEVEL 2: Local Supervisory
LEVEL 1: Local Controllers
LEVEL 0: Field Devices
Airgap/Enforcement
Safety Systems
SMALLER ICS SITES
Minor Enforcement Boundary
LEVEL 2: Local Supervisory
LEVEL 1: Local Controllers
LEVEL 0: Field Devices
Airgap/Enforcement
Safety Systems
Minor Enforcement Boundary
LEVEL 2: Local Supervisory
LEVEL 1: Local Controllers
LEVEL 0: Field Devices
Airgap/Enforcement
Safety Systems
Minor Enforcement Boundary
LEVEL 2: Local Supervisory
LEVEL 1: Local Controllers
LEVEL 0: Field Devices
Airgap/Enforcement
Safety Systems

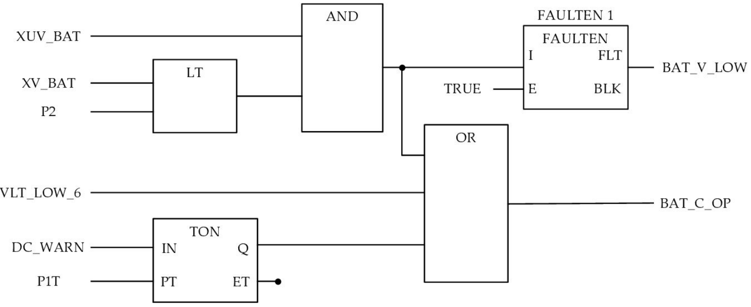


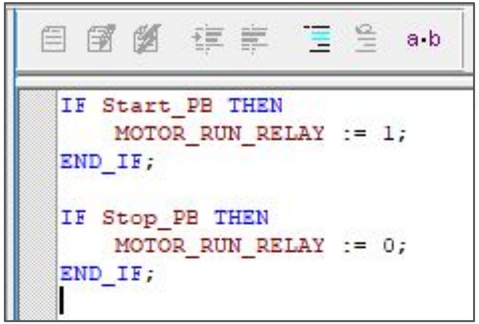
PLC Programming

Standardized with IEC 61131-3 which defines three graphical and two textual programming language standards:

- Ladder diagram (LD), graphical
- Function block diagram (FBD), graphical
- Structured text (ST), textual
- Instruction list (IL), textual (deprecated)
- Sequential function chart (SFC), graphical



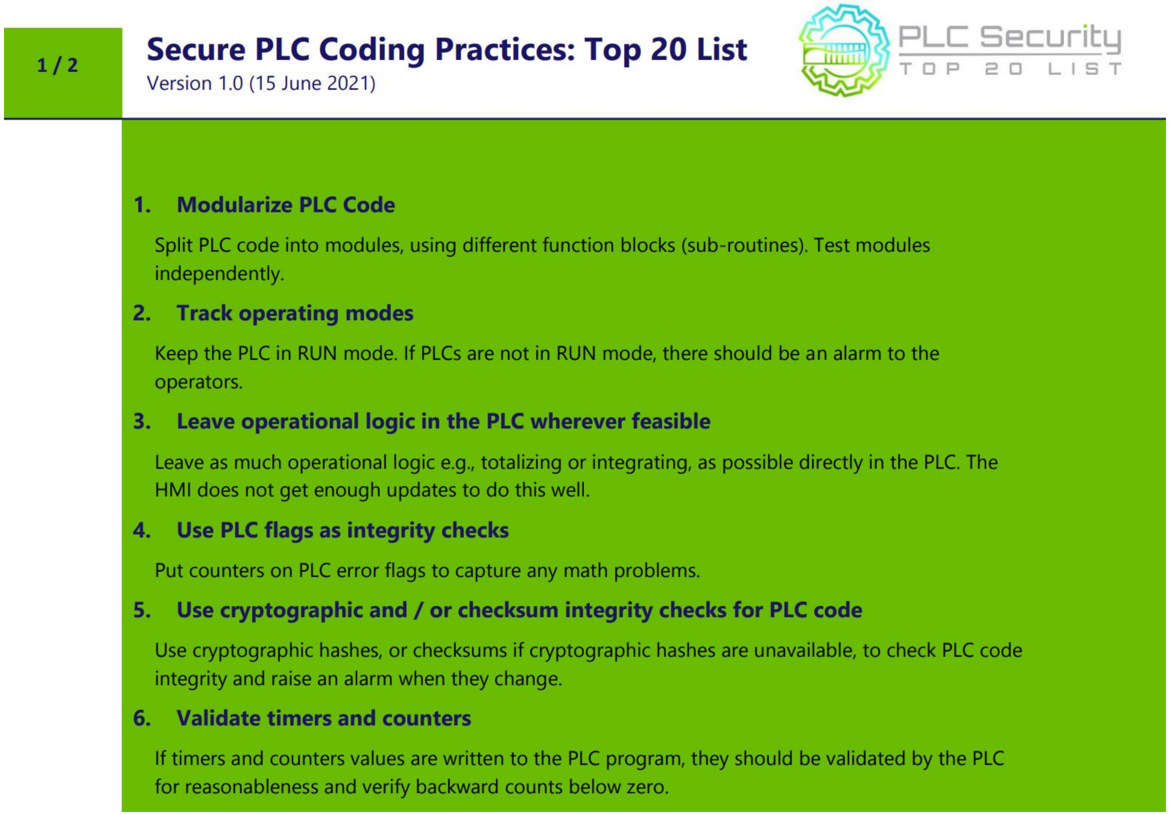




```
IF Start_PB THEN
    MOTOR_RUN_RELAY := 1;
END_IF;

IF Stop_PB THEN
    MOTOR_RUN_RELAY := 0;
END_IF;
```

PLC Secure Coding Practices



1 / 2

Secure PLC Coding Practices: Top 20 List

Version 1.0 (15 June 2021)

PLC Security
TOP 20 LIST

1. **Modularize PLC Code**

   Split PLC code into modules, using different function blocks (sub-routines). Test modules independently.

2. **Track operating modes**

   Keep the PLC in RUN mode. If PLCs are not in RUN mode, there should be an alarm to the operators.

3. **Leave operational logic in the PLC wherever feasible**

   Leave as much operational logic e.g., totalizing or integrating, as possible directly in the PLC. The HMI does not get enough updates to do this well.

4. **Use PLC flags as integrity checks**

   Put counters on PLC error flags to capture any math problems.

5. **Use cryptographic and / or checksum integrity checks for PLC code**

   Use cryptographic hashes, or checksums if cryptographic hashes are unavailable, to check PLC code integrity and raise an alarm when they change.

6. **Validate timers and counters**

   If timers and counters values are written to the PLC program, they should be validated by the PLC for reasonableness and verify backward counts below zero.